MODULE- 1
MANAGEMENT

INTRODUCTION

- Management is a function of guidance and leadership control of efforts of a group or individuals in order to achieve goals/objectives of an organization.

MEANING

- Management is a critical element in the **economic growth** of a country. Management is the dynamic, life-giving element in every organization. The managing has become essential to ensure the coordination of individual efforts. The definitions of the management are:
  - Management is the **art of getting things done** through people.
  - Management is a process consisting of **planning, organizing, actuating and controlling, performed to determine and accomplish** the objectives by use of people and resources.

NATURE AND CHARACTERISTICS OF MANAGEMENT

- The production process involves land, labour, capital, organization and entrepreneurship.
- The production is the result of their combined efforts.
- The success of production depends on their effective combination and cooperation.
- The effective balance is a challenging task. Proper skills and knowledge is the management.
- The important characteristics of management are:
  - Stable
  - Applicable to all kinds of organizations
  - Transparent
  - Approaches to be clear and goal oriented
  - Simple and effective
  - It should be responsive to many external elements like economic, technological, social, political and ethical factors.
  - Well defined goals and effective means to accomplish the goals.
  - Good planning, organizing, staffing, directing and controlling functions.
  - Provide conducive atmosphere of work.

SCOPE OF MANAGEMENT

- The management is a must for every organization.
- Management plans the activities, coordinates and utilizes the available resources effectively and efficiently at minimum cost.
- The scope of management is not limited only to business organization but it is extended to Business establishments
- The scope can be extended to important areas of life such as:
  - Development management
  - Distribution management
  - Financial management
  - Marketing management
  - Personnel management
  - Production management
  - Office management
  - Transport management
  - Purchase management
  - Sales management
  - Supply chain management
  - Business management

- Its scope starts from self management to home management and to the management of organizations.

MANAGEMENT AS A SCIENCE, ART OR PROFESSION

- The art is doing things in the light of the realities of a situation under "art" one normally learns the "how" of a phenomenon.
- It is the art of getting things done through others in dynamic and mostly non- repetitive situations.
- The science is an organization knowledge.
- The essential feature of any science is the application of scientific methods to the development of knowledge.
- So management is partly an art and partly a science. Management does not possess the characteristics of a profession.
- A profession is expected to have organized and systematic knowledge, formalized methods of acquiring training and experience.
- But nowadays, management has become a profession than art or science.

MANAGEMENT AND ADMINISTRATION

- People feel that administration involves "thinking". It is a top level function that centers around the preparation of plans, rules, policies and objectives of an organization.
- Whereas management involves "doing" and is a lower level function, concerning with execution and direction of policies and operations.
- So administration is more important at lower levels. The time spent in administrative and managerial function at different levels is as in fig.



- Management is regarded as comprehensive generic function covering entire process of planning, organizing, directing and controlling.
- Administration is regarded as a branch of management that has two functions- planning and controlling.
- The function of management is divided into two categories the upper level management called as administrative management and lower level management termed as operative management.
- This can also be defined as managing of business enterprises is called management and managing non-business organizations is called administration.
- Administration determines the policies upon which the enterprise is to be conducted while the function of management is to carry out the policies that are laid down by the administrative group.
- Differences between administration and management

Characteristics	Administration	Management
Main Function	Planning ,Organising and staffing	Leading & Controlling
Status	Act as owner	Act as an agency
Skills	Good Administrative skills	More technical skills
Level in the organization	Top level	Middle/Lower level
Position	MD, Owner, CEO	Manager , Supervisor, foremen etc...
Objective	Makes the polices & objective to be achieved	Implements the plans and policies
Involvement	No direct involvement in production or service	Directly involves in the execution of plans and achieving objectives

ROLES OF MANAGER

- The three important roles played by a manager are
  - Interpersonal Roles
  - Informational Roles
  - Decisional Roles

- **Interpersonal Roles**
  - Figure head: The manager has to perform some duties that are casual and informal such as attending the functions, taking for lunch etc.
  - Leader: Manager must motivate and encourage his employees. He must try to reconcile their individual needs with the goals of the organization.
  - Liaison: Manager must cultivate contacts outside to collect information useful for his organization.
- **Informational Roles**
  - Monitor: The manager monitors his environment and collects information through his personal contacts.
  - Disseminator: The manager passes some of his information directly to his subordinates.
  - Spokesman: The manager communicates the information/goals of organization to his superiors. He also communicates the performance of company to shareholders and the rules and responsibility to his subordinates.
- **Decisional Roles**
  - Entrepreneur: For the improvement of the organization the manager continuously looks for new ideas, which may result in the development of new products or service or finding a change in work environment.
  - Disturbance Handler: manager works as a problem solver. He finds solutions of various un-anticipated problems both within and outside the organization.
  - Resource Allocator: The manager must divide work and delegate authority among his subordinates. He must decide who will get what.
  - Negotiator: The manager has to spend considerable time in negotiations. Must try to resolve any internal problems like trade agreements, strikes and solve the problems of employees.

LEVELS OF MANAGEMENT

- Almost all managers perform the same functions of management like planning, organizing, directing and controlling there are levels among them.
  - Top management
  - Middle management
  - First line or supervisory or Lower level management
- Top management consists of Chairman. Directors, Company Presidents, Vice- Presidents, CEO's. These are the people who make policies for the company, set goals and targets. They should possess conceptual and design skills.
- Middle management include finance manager, sales manager, marketing manager, personnel manager, department heads etc.
- Lower level managers are supervisors and foremen.
- Skills at different levels of management or managerial skills
  - Conceptual skills: The ability to view of organization and its future, think in abstract, analyze the forces working in a situation, creative and innovative ability and the ability to assess the environment and changes taking place.
  - Human relations skill: Ability to interact effectively with people at all levels. This skill develops the ability to recognize the feelings and sentiments, to judge the possible reactions, to examine his own concepts and values to develop more useful attitudes.
  - Technical skill: The person's knowledge and proficiency in any type of process or technique.

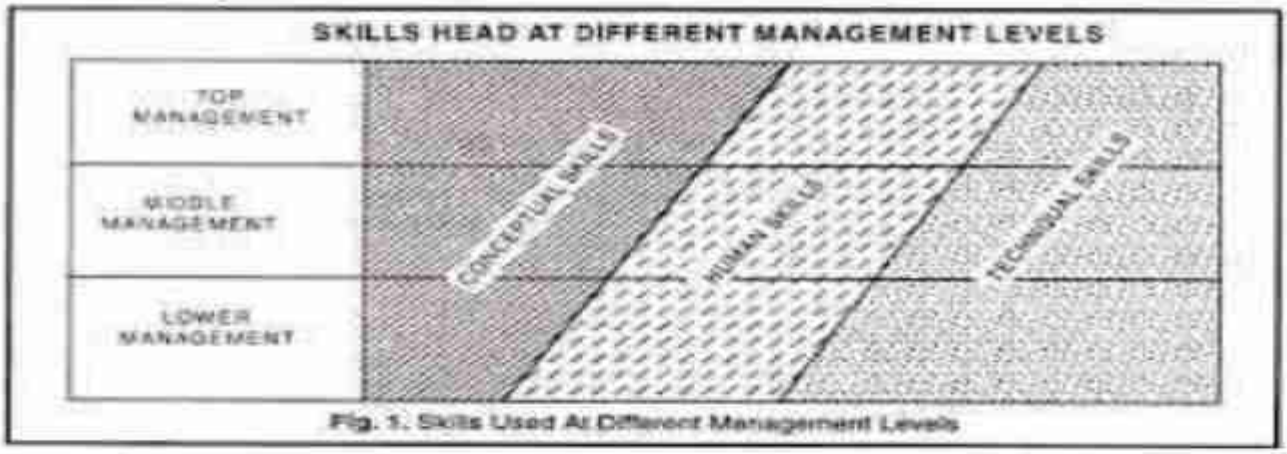


Fig. 1. Skills Used At Different Management Levels

FUNCTIONS OF MANAGEMENT OR PROCESS OF MANAGEMENT

- The five functions of management are:
  - Planning
  - Organizing
  - Staffing
  - Directing (leading)
  - Controlling
- **Planning:** Determines in advance what should be done. Planning is the function that is referred to as decision making. It involves mission and objectives and action to achieve them.
- This involves the following:
  - Setting short and long term goals for organization.
  - Selecting objectives , strategies and polices for accomplishing the planned goals.
  - Deciding in advance what to do, how to do, who has to do, when to do and where to do.
  - Planning bridges the gap from where we are now to where we want to be in future.
- **Organizing :** Organizing is a part of management that involves in establishing an intentional structure of roles for people to fill in an organization. To organize a business well, it is required

to provide all the useful things for its proper functioning. They are raw materials, tools, capital and personnel. This involves :

- Determination of activities required to achieve goals.
- Grouping these activities into department.
- Assigning groups of activities to manager.
- Forming delegation of authority.
- Making provisions for coordination of activities.

- **Staffing:** This makes the provision for man power to fill different positions. This is done by identifying work-force requirements, recruiting new staff, selecting, placing, promoting, appreciating, planning their career, training the staff to accomplish their tasks effectively and efficiently. This involves:
  - Finding the right person for right job.
  - Selecting the personnel.
  - Placement, training and developing new skills required for present and future jobs.
  - Creating new positions.
  - Apprising the staff and planning their growth and promotions etc.
- **Directing:** Directing involves three sub functions namely communication, leadership and motivation.
- Communication is the process of passing information and understanding from one person to another.
- Leadership is the process by which a manager guides and influences the work of his subordinates.
- Motivation means encouraging the minds of employees of an organization to perform their best.
- **Controlling:** Controlling is measuring and correcting of activities of subordinates to make sure that the work is going on as per the plans. Controlling relates to the measurement of achievement. This involves:
- Establishing standards of performance.
- Measuring performance and comparing with established standards.
- Taking necessary corrective action to meet the set standards.

Management Cycle



PLANNING

- **NATURE OF PLANNING**
- Planning is the most basic function of management. It is "deciding in advance" what to do, how to do, when to do and who has to do it etc.
- It is an intellectual process, which requires a manager to think before acting. It is nothing but thinking in advance. A manager should continuously watch the progress of the plans like a

navigator. Planning involves selection of objectives and goals and determines the ways and means of achieving them.

- Planning bridges the gap from where we are to where we want to be. Planning is vital at all levels of an organization.
- Planning involves four essential qualities:
  - It must contribute to accomplish purpose and objectives.
  - It must be considered as parent exercise in all processes.
  - It must spread through all management functions.
  - It must be efficient in such a manner so as to achieve the designed goals at the least cost.

IMPORTANCE OF PLANNING

- Planning is important because:
  - ➢ It overcomes uncertainty and change and minimizes risk.
  - ➢ It facilitates effective control.
  - ➢ It focuses attention and concentration only on the objectives of enterprise.
  - ➢ It makes economic operation and leads to success.
  - ➢ It forms the bridge between the present and the future.
- Uncertainty and minimize risk: Planning provides logical facts and procedure to managers for making decisions. This logical decision making based on plans to organization minimizes uncertainty and risk.
- Effective control: Planning sets goals, targets and means to accomplish these goals. These goals and plans become standards or bench marks against which performance can be measured. Thus good plans help effective control on the activity.
- Focuses attention and concentration on the objectives of the enterprise: Planning helps the managers to focus their attention on the goals and activates of organization. This makes entire organization to accomplish the goals.
- Economic operation and leads to success: Planning does not ensure success, but leads to success. If the work is planned in advance, there will be no confusion and things will happen as per plan and can even reduce uncoated expenditure.
- Bridge between the present and the future: There is a vast gap between what we are today and what we want to be in future. A proper and systematic plan forms the bridge between the present and future. Hence planning is important for the success of any organization.

PURPOSE OF PLANNING

- The purpose of planning are:
  - To select from many available alternatives so as to achieve the objectives of the enterprise, economically, effectively and efficiently.
  - To direct all other functions of management.
  - To set up the goals of an enterprise in perspective, within the environment.
  - To help planned goals of an enterprise to break-up into more easily hand able additive-segmented goals.
  - To form the basis for budget.
  - To forecast the future to avoid uncertainty and change.
  - To provide effective control.
  - To search for alternatives and adopt the best way of accomplishing the work.
  - To focus the vision on the objectives and goals.

TYPES OF PLANS

- Based on nature of planning, the planning is classified as strategic planning (Long range planning) and tactical planning (short range planning). The difference between them are:

Forms of Planning	
Strategic Planning	Tactical Planning
It is long term.	It is short term.
Done at top management.	Done at lower levels of management.
It decides the major goals and policies of allocation of resources to achieve these goals.	It decides the detailed use of resources for achieving each goal.
It is generally based on long-term forecasts about technology, political environment etc. and is more uncertain.	It is generally based on the past performance of the organization and is less uncertain.
It is less detailed because it is not involved with the day-to-day operations of the organization.	It is more detailed because it is involved with the day-to-day operations of the organization.

- The hierarchy of organizational plans are:



- Objectives are the broad end of the organization which are achieved by means of strategies. Strategies has two major groups plans – single-use plans and standing plans. Single-use plans, are developed to achieve a specific end; when end is reached the plan is dissolved. The plans are programmes and budgets. Standing plans are designed for situations that recur often.
- **Objectives**
  - Objectives are goals or aims which the management wishes the organization to achieve. These are end points towards which all business activities like organizing, staffing, directing and controlling are directed.
  - The objectives should have "vision" "purpose" and "mission".

    - The vision is the dream that an entrepreneur creates and the root for all objectives.
    - The purpose of an organization is its primary role defined by the society.
    - The mission is the unique aim that sets the organization apart from others.
    - Objectives are the specific target to be reached by an organization.
- Characteristics of Objectives:
    - Objectives are multiple in number.
        - These are market standing, innovation, productivity, physical and financial resources, profitability, manager performance and development, worker performance and attitude and public responsibility.
    - Objectives are either tangible or intangible.
    - Objectives have a priority.
    - Objectives are generally arranged in a hierarchy.
    - Objectives sometimes clash with each other.
- Requirements of sound objectives:
    - Objective must be both clear and acceptable.
    - Objectives must support one another.
    - Objectives must be precise and measurable.
    - Objectives should always remain valid.
- Advantages of objectives:
    - Provide a basis for planning
    - They act as motivators
    - They eliminate haphazard action
    - They facilitate coordinated behavior.
    - Function as a basis for managerial control.
    - They facilitate better management.
    - They lessen misunderstanding and conflict.
    - They provide legitimacy to organization's activity.

Strategies

- Strategies is a plan which account the environmental opportunities and threats and the organizational strengths and weaknesses and provides an optimal match between the firm and the environment.
    - Environment Appraisal: An analysis of the relevant environment result in the identification of threats and opportunities. Some of the key factors are:
        - Political and legal factors
        - Economic factors
        - Competitive factors
        - Social and cultural factors
    - Corporate Appraisal: This involves an analysis of the company's strengths and weaknesses.

Standing Plans

- Policies: A policy is general guideline for decision-making. It sets up boundaries around decisions, including those that can be made and shutting out those that cannot.
- Policy is verbal, written or implied overall guide, setting up boundaries that supply the general limits and directions in which managerial action will take place.
- Policy deals with "how to do" the work.
- Advantages of polices
    - Policies ensure uniformity of action and makes action more predictable.
    - Policies speed up decisions at lower levels.

    - Policies make it easier for the superior to delegate more and more authority to his subordinates.
    - Policies give a practical shape to the objectives.
- Types of policies: They are classified on the basis of sources, functions or organizational level.
- Classification on the basis of sources:
    - On this basis, policies are divided into originated, appealed, implied and externally imposed policies.
- Classification on the basis of organizational level:
    - On the basis of business functions, policies may be classified into production, sales, finance, personnel policies etc.
- Classification on the basis of organizational level:
    - Policies range from major company policies through major departmental policies to minor or derivative policies applicable to the smallest segment of the organization.
- Guidelines for effective policy-making:
    - Policies should be in writing
    - Policies should reflect the objectives
    - Top managers and subordinates must participate in the formation of policies.
    - Policy must strike a reasonable balance between stability and flexibility.
    - Policies in the organization should not pull in different directions.
    - Policies should not be detrimental to the interest of society.
    - Policies must be comprehensive.
    - Policies should be periodically reviewed.
- Procedures: A procedure provides a detailed set of instructions for performing a sequence of actions involved in doing a certain piece of work.
- Procedures are to be followed every time when activity is performed.
- Difference between policy and procedure:

Policy	Procedure
General guidelines of the organization.	General guidelines at the action level.
Top level activity.	Departmental activity.
Policies fulfill the objectives of an organization	Procedures guide the way to implement the policies.
Policies are often made without any study or analysis.	Procedures are always made after through study and analysis of work.

- Methods:
    - A method is a prescribed way in which one step of a procedure is to be performed. Thus method is a part of procedure.
    - Method help in increasing the effectiveness and usefulness of the procedure.
    - A procedure has a number of steps, each step may have number of methods to do it.
- Rules:
    - Rules are detailed and recorded instructions that a specific action must or must not be performed in a given situation.
- Single-use plans
    - Programmes

- Budgets

STEPS IN PLANNING

- **Establishing goals/objectives**
  - To determine the enterprise objectives and once the organizations objectives are determined, the section wise or department wise objectives are planned at the lower level.
- **Establishing planning premises**
  - That is the conditions under which planning activities will be undertaken.

The planning premises can be classified as below:

- Internal and External premises.
- Tangible and Intangible premises.
- Controllable and non-controllable premises.

- **Deciding the planning period**

The factors which influence the choice of a period are:

- Lead time in development and commercialization of a new product.
- Time required for recovering capital investment or the pay back period.
- Length of commitment already made.

- **Identification of alternatives course of action**
  - A particular objective can be achieved through various actions.
- **Evaluation and selection of course of action**
- **Developing derivative/supportive plans**
- **Measuring and controlling the process**

LIMITATIONS OF PLANNING

- Planning is time consuming Process. It involves significant amount of money energy and risk without any assurance of the fulfillment of organization's objectives.
- Planning sometimes restricts the organization to the most rational and risk free opportunities.
- Scope of planning is said to be limited in the case of organizations with rapidly changing situations.
- Establishment of advance plans tend to make administration inflexible.
- Difficulty of formulating accurate premises.
- Planning may sometimes face people's resistance to it.

DECISION-MAKING

Meaning

- A decision is choice between two or more alternatives.
- Decisions are made by the managers and actions are taken by others.
- Decision- making is commitment to something, a point of view, a principle or course of action.

It implies three things:

- When manager make decisions they are choosing-they are deciding what to do on certain basis.
- Managers have alternatives available when they are making a decision. It requires wisdom and experience to select best one.
- Managers have a purpose in mind when they make decision.

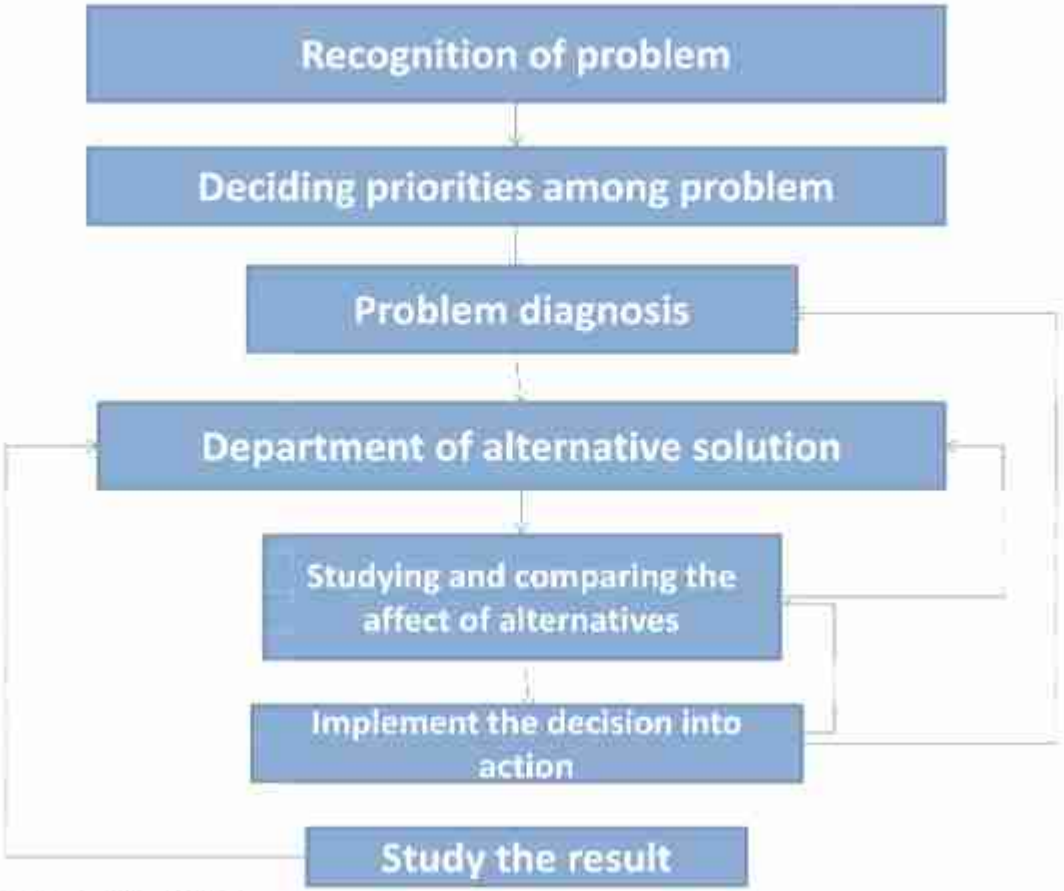
TYPES OF DECISIONS

- **Decisions are classified as:**
  - **Programmed and non-programmed decisions:**
    - Programmed or Pragmatic are those decisions taken within purview of the policies, rules or procedure. These are routine decisions or structured decisions.

        - Non-programmed or non Pragmatic decisions are called as strategic decisions. These involve heavy expenditure and are generally taken by top management.
    - **Individual and collective decisions**
        - Group decision-making has advantages like increased acceptance, better communication and better coordination.
    - **Major and minor decisions**
        - Minor decision are related to day to day periodical occurrences.
            - Example: Purchase of stationary, granting leaves and permissions etc.
        - Major decisions generally done by top management.
            - Example purchase of new machinery, employing new technology, hiring new people etc.
    - **Strategic and routine decisions**
        - Strategic decisions are generally done by top management.
        - Example price increase/ discount, Change in product range etc
        - Routine decisions are day to day decisions.
    - **Simple and complex decisions**
    - **Temporary (Ad-hoc) and permanent decisions.**
        - Some decisions are to be taken depending on situation till the solution is found.
        - Permanent decisions are taken on permanent basis.

Decision Making Process: Steps in Rational Decision Making

- The steps in decision making are:
    - Recognizing the problem.
    - Deciding priorities among the problems.
    - Diagnosing the problem.
    - Developing alternative solutions or courses of action.
    - Measuring and comparing the consequences of alternative solutions.
    - Converting the decision into effective action and follow up of action.



- Recognition of problem
  - A problem may exist either due to deviation from past experience, deviation from plan, when competitors outperform and when people bring problems to the manager.
- Deciding priorities among problem
  - The manager must identify the problems which he can solve, which he feels that his subordinates can solve and which are to be referred to higher officials.
- Problem diagnosis
  - Managers should follow system approach in diagnosing a problem. A thorough study need to done before arriving at solution.
- Development of alternative solutions
  - Every problem there will be some alternative solutions. Alternatives do exist. Manager has to find it out.
- Studying and comparing the affect of alternatives
  - The alternative solutions are measured and compared as well as quality and acceptability is analyzed.
- Implementation of the decision into action
  - This requires the communications of the decisions to the concerned employees in clear and simple terms.
- Study of results
  - The manager has to carry out the follow up action. If the result is not satisfactory, the manager has to take necessary corrective action or modify his decision. Some of the difficulties faced by manager are: Incomplete information, Non-conductive environment, Opposition by subordinates, improper communication, Wrong timing, Statutory regulations, Government policies, External influences and Lack of support.

MODULE- 2

Organizing and Staffing & Directing and Controlling

ORGANIZING

- **MEANING**
- Organization provides the structure, the frame on which the management of the enterprise is based.
- An organization is a social unit or human grouping deliberately structured for the purpose of attaining specific goals.
- An organization as the rational coordination of the activates of a number of people for the achievement of some common goal.
- The process of identifying and grouping of the work to be performed, defining and delegating responsibility and authority and establishing relationships for the purpose of enabling people to work most efficiently together in accomplishing the objectives.

Nature and Characteristics of Organization

- Organizing is the grouping of activates necessary to attain objectives.
- An organization should be designed to clarify who is to do what and who is responsible for what results.
- Two types of organizations are:
    - Formal Organization:
        - It must be flexible and there should be room for discretion , for taking advantage of creative talents.
    - Informal Organization:
        - It is any joint personal activity without conscious joint purpose, even though possibly contributing to joint results.

Characteristics

- Every organization has a purpose, goals which are the task of planning.
- A clear concept of the major duties or activities required to achieve the purpose.
- Classification of activities into jobs.
- Establishment of relationships between these jobs.
- The organizations are classified into four categories. Organizations which benefit their **owners, members, clients and whole society.**

PROCESS OF ORGANISING

- Organizing means designing the organization structure. The manager differentiates and integrates the activities of the organization.
- Differentiation is meant the process of departmentalization or segmentation of activities on the basis of some homogeneity.
- Integration is the process of achieving unity of effort among the various departments.
- The seven-step procedure are:
    - Consideration of objectives: Objectives determine the various activities which need to be performed and the type of organization which need to be build for this purpose.
    - Deciding Organizational Boundaries

- Grouping of Activities into Departments: To identify the activities necessary to achieve them and to group the closely related and similar activities into departments and sections.
- Deciding which Departments will be Key Departments: the departmental activates essential for the fulfillment of goals.
- Determining levels at which various types of Decisions are to be made: The levels at which various major and minor decisions are to be made must be determined.
- Determining the span of management: Designing the structure is to determine the number of subordinates who should report directly to each executive.
- Setting up a Coordination mechanism: As individuals and departments carry out their specialized activates, the overall goals of the organization may become submerged or conflicts among organization members may develop so need to be checked. The activities, decision and relations analysis need to be done.

PRINCIPLES OF ORGANISING

- The principals are:
- Objectives: The objectives of enterprise influence the organization structure and hence the objectives of the enterprise should first be clearly defined.
- Specialization: The activities of the enterprise should be divided according to functions and assigned to persons according to their specialization.
- Span of Control: As there is a limit to the number of persons that can be supervised effectively by one boss, the span of control should be as far as possible, the minimum.
- Management by Exception principle: only complex problems should be referred to higher level and other problems should be dealt by lower level.
- Scalar Principal (chain of command): The line of authority from the chief executive at the top to the first-line supervisor at the bottom must be clearly defined.
- Unity of Command: Each subordinate should have only one superior whose command he has to obey.
- Delegation: Proper authority should be delegated at the lower levels of organization also. The authority delegated should be equal to responsibility.
- Responsibility: The superior should be held responsible for the acts of his subordinates.
- Authority: Authority is the tool by which a manager is able to accomplish the desired objectives.
- Efficiency: The organization structure should enable the enterprise to function effectively and accomplish its objectives with the lowest possible cost.
- Simplicity: Organization structure should be as simple as possible and levels should be as far as possible, be minimum.
- Flexibility: The organization should be adaptable to changing circumstances.
- Balance: A reasonable balance in the size of various departments, between centralization and decentralization.
- Unity of Direction: One objective and one plan for a group of activities having the same objectives.
- Personal Ability: There is need for proper selection, placement and training of staff.
- Acceptability: The structure of the organization should be acceptable to the people who constitute it.

DEPARTMENTALISATION

- The horizontal differentiation of tasks or activities into discrete segments is called departmentalization.
- There are several bases for departmentalization, each of which is suitable for particular corporate size, strategies and purposes.
- **FUNCTIONS**
  - Each major function of the enterprise is grouped into a department. Various functions are Marketing, Engineering, Production, Finance, Personnel, Purchase etc.
  - Advantages:
    - Simple form of grouping activities for small organizations which manufacture only a limited number of products or render only a limited number of services.
    - It promotes excellence in performance.
    - It leads to improved planning and control of the key functions.
    - It ensures economy, there is only one department related to one function for the entire function.
  - Drawbacks
    - De-emphases overall company objectives.
    - Leads to over specialization of people.
    - Reduces coordination between functions.
    - Slow adoption to changes in environment.
    - Limits development of general managers.
- **PRODUCTS**
  - This is more suitable for a large organization that manufactures a vast variety of products.
  - Under this separate groups or departments are created and each department is controlled by a manager who will be responsible for all the activities of that sub group.
  - Each subgroup will have its own facilities required for manufacture, purchase, marketing and accounting etc.
  - **Advantages:**
  - Top management is relieved of operational task enabling them to concentrate more on common goals.
  - Enables top management to compare the performances of different products and invest more resources in profitable products.
  - Managers of individual products put better effort to improve his area compared to others.
- Drawbacks:
  - This form results in duplication of staff and facilities.
  - Extra expenditure is incurred in maintaining a sales force for each product line.
  - Employment of large number of managerial personnel is required.
  - May result in under utilization of facilities and equipments.
- **CUSTOMERS**
- An enterprise may be divided into a number of departments on the basis of the customers that it services.

- The advantage is it ensures full attention to major customer groups which have a very different set of criteria governing their decision to purchase.
- The disadvantages of this form are
    - It may result in under- utilization of resources and facilities.
    - May be duplication of facilities.
- **REGIONS or TERRITORY**
- When several production or marketing units of an organization are geographically dispersed in various locations, it is logical to departmentalize those units on geographical basis.
    - **Advantages:**
- It motivates each regional head to achieve high performance.
- It provides each regional head an opportunity to adapt to his local situation.
- It affords valuable top-management training and experience to middle-level executives.
- It enables Organization to compare Regional Performances & invest more in Profitable Areas & withdraw Resources from Unprofitable Ones.

- Complex business world, some of the administrative tasks cannot be performed by a single person alone.
- Such situations may call for two or more persons to perform such tasks.
- A committee is a group of people pooled to carry out a defined objective.
- A committee is a body of persons appointed or elected to meet on an organized basis for the consideration of matters brought before it.
- The area of operation of a committee is determined by its constitution.
- **Principles:**
- The no. of persons in a committee should depend upon need(5 to 10).
- Responsibility, authority, objectives, duties of the committee should be clearly defined.
- Agenda of the committee should be proposed or communicated to committee at least a week before they meet for discussion.
- Problems which can be taken care by individual should not be included in the agenda of the committee.
- Committee meeting should begin and end on prefixed timings.
- Problems not related to the subject should not be discussed because it will be a waste of time.
- The recommendation made by committee should be published and circulated to interested and concerned persons.
- The committee should be appraised , if this action is taken based upon its recommendation.
- A committee must be dissolved after its purpose.

Various functions of committee are:

- Collect the necessary information from different sources and arrange them in order.
- The collected information is critically examined and analyzed.
- Draft a detail report containing the recommendations for the purpose of implementation.
- Farming the policies of the organization.
- Selection of personnel, directing and controlling the officers at regular intervals to achieve the goals.

Types of committee :

Classified according to the place and purpose,

- Standing or permanent committee
- Temporary committee.
- General advisory committee.
- Joint consultative committee.
- Academic committee.
- Religious committee.
- Educational.
- **Advantages of Committees**
- Committees provide a forum for the pooling of knowledge and experience of many persons of different skills, ages and backgrounds.
- Committees are excellent means of transmitting information and ideas both upward and downward.

- Committees are impersonal in action and hence their decisions are generally unbiased and are based on facts.
- When departmental heads are members of committee, people get an opportunity to understand each others problems and hence improve coordination.
- **Weaknesses of Committees**
- In case a wrong decision is taken by committee, no one is held responsible which may results in irresponsibility among members.
- Committees delay action
- Committees are expensive form of organization.
- Decisions are generally arrived at on the basis of compromise and hence they are not best decision.
- As committee consists of large number of persons, it is difficult to maintain secrecy.

SPAN OF CONTROL

- The span of control indicates the number of subordinates who can be successfully directed by a supervisor.
- It is often referred to as span of management, span of supervision, span of authority.
- Span of management is important because of two reasons.
- First is span of management affects the efficient utilization of managers and the effective performance of the subordinates.
- If the span is too wide, managers are overburdened and subordinates receive little guidance. If the span of management is too narrow, the managers are under utilized and subordinates are over controlled.
- The second reason is there is relationship between span of management and organization structure.
- A narrow span results in tall organization with many levels of supervision between top management and lowest organizational levels which creates more communication and cost problems.
- On the other hand, a wide span for the same number of employees results in flat organization with fewer management levels between top and bottom.
- **Factors Affecting the Span of Management**
- Ability of the manager
- Ability of the employees
- Type of work
- Geographic location
- Well-defined authority and responsibility
- Level of management
- Economic considerations

Centralization Vs Decentralization of Authority and Responsibility

- Centralization refers to systematic reservation of authority at central points within the organization.
- Centralization means retention or concentration of managerial authority in few key managerial positions at the nerve center of an organization i.e., at the top level.

- Everything that goes to reduce the subordinate's role in decision making is centralization.
- Decentralization means dispersal of decision-making power to lower levels of the organization.
- In decentralized setup, ultimate authority to command and ultimate responsibility for the results is localized as far down in the organization.
- "Decentralization refers to the systematic effort to delegate to the lower levels all authority except that which can only be exercised at central points".
- In decentralized setup large number of decision, important decisions in large number of areas are made consulting few people.

- Advantages in decentralization
    - It eases the burden of top level managers.
    - Decentralization permits quicker and better decision making.
    - With decentralization capable managers can be developed
    - Promotes participation in decision making and improves morale and motivation.
    - Decentralization facilitates diversification of products.
    - Disadvantages of decentralization
- It increases administrative expenses.
- It may create problems in bringing coordination among various units.
- It may bring about inconsistencies in the company, because uniform procedures may not be followed for the same type of work in various divisions.
- **DELEGATION VS DECENTRALIZATION**
    - Delegation is a process while decentralization is the end result of a deliberate policy of making delegation widespread in the organization.
    - Delegation takes place between a superior and a subordinate while decentralization is company wide delegation as between top management and the departments or division of the organization.
    - In delegation, delegator exercises supervision and control over the delegate, while in decentralization, top management exercises broad minimum control.
- Principles of Delegation
- Receptiveness
- Willingness to let go
- Willingness to let others make mistakes
- Willingness to trust subordinates
- Willingness to establish and use broad controls
- **Authority and Power**

Authority	**Power**
It is the institutionalized right of a superior to command and compel his subordinates to perform a certain act.	**It is ability of a person to influence others.**
It rests in the position	**It rests in the individual.**
It is delegated to an individual by his superior.	**It is earned by individual**

It is well defined	**It is undefined and infinite.**
It is what exists in the eye of law.	**It is what exists in fact. It is a de facto concept.**

- **Responsibility**
- The obligation of a subordinate, to whom a duty has been assigned, to perform the duty.
- Responsibility arises from a superior- subordinate relationship.
- Responsibility is divided into two parts namely operating responsibility and ultimate responsibility.
- If the subordinate fails to perform the job (operating responsibilities) the superior is held responsible for this failure (Ultimate responsibility)

NATURE AND IMPORTANCE OF STAFFING

- Staffing is filling and keeping filled, positions in the organization structure. It is human resource management.
- The staffing is important because
  - It helps in discovering talented and competent workers and developing them to move the organizational ladder.
  - Staffing is important to put the right person at right job which results in increased production.
  - It avoids sudden disruption of the production due to shortage of workers, since it plans and fills the positions.
  - It maintains harmony and creates healthy atmosphere.
  - It plans the requirement of man power at various time and levels of project.
- Elements of staffing
  - Proper placement of people.
  - Rational recruitment and selection.
  - Proper positioning and fixation of salaries.
  - Providing necessary training for the people to carry out job.
  - Good promotional policies and retirement scheme.
- Functions of staffing
  - Man Power planning
  - Development
  - Fixing employment standards
  - Source of selection
  - Selection
  - Training
  - Routine functions

RECRUITMENT

- Recruitment is the process of identifying the sources for prospective candidates and to stimulate them to apply for the jobs.
- It is a linking activity that brings together those offering jobs and those seeking jobs.

- Recruitment refers to the attempt of getting interested applicants and providing a pool of prospective employees so that the management can select the right person for the right job from this pool.
- The various sources of recruitment are divided into two categories:
    - **Internal Sources.**
    - **External Sources.**

Internal sources

- Internal sources involve transfer and promotion.
- Transfer involves the shifting of an employee from one job to another.
- Many companies follow the practice of filling higher jobs by promoting employees who are considered fit for such positions.
- Filling higher positions by promotion motivates employees, boots employee's morale.

External Sources

- Direct recruitment
- Unsolicited applications
- Advertising
- Employment agencies
- Educational institutions
- Labour contractor
- Recommendations

SELECTION

- The process of selection leads to employment of persons having the ability and qualifications to perform the jobs which have fallen vacant in an organization.
- It divides the candidates into two categories; those will be offered employment and those who will not be.
- The basic purpose of the selection process is choosing right type of candidates to man various positions in the organization.

Selection process:

- Receipt of applications
- Screening of applications
- Employment tests
- Interviews
- Background investigation and medical examination
- Final selection

DIRECTING AND CONTROLLING

MEANING AND NATURE OF DIRECTING

- Directing means issuing of orders leading and motivating subordinates as they go about executing orders.
- Direction is a function of management which is related with instructing, guiding and inspiring human factor in the organization to achieve organizational mission and objectives. There are three elements of direction, namely communication leadership and motivation.

- Directing is the interpersonal aspect of managing by which subordinates are led to understand and contribute effectively to the attainment of enterprise objectives.
- Directing consists of the process and techniques utilized in issuing instructions and making certain that operations carried on of originally planned.
- Principles of Directions
    - Harmony of objectives
    - Unit direction of command
    - Efficiency
    - Direct supervision
    - Effective communication
    - Effective control
    - Follow-through
- **LEADERSHIP**
- Leadership is the art of process of influencing people so that they will strive willingly towards the achievement of group goals.

LEADERSHIP STYLES

- Traits approach
    - The leadership dealt with personal abilities
- Behavioral approach
    - The theories based on leadership behavior style are
        - Leadership based on the use of authority
        - Likert's four systems of managing
        - The managerial grid
        - Leadership involving a variety of styles and level of use of power and influence.
- Contingency approach to leadership
    - Which combines both autocratic and democratic

Motivational Theories

- Maslow's Need Hierarchy Theory
- Hertzberg's Two Factors Theory
- McClelland's Achievement Theory
- Victor Vroom's Expectancy Theory
- Adam's Equity Theory
- Skinner's Behavior Modification Theory
- Carrot and Stick Approach

Maslow's Theory of Motivation

Abraham Maslow is considered the Pioneer in contributing a systematic scheme of Hierarchy. After a proper research he came to the conclusion that there are some needs of the employees which they expect to be satisfied once they join an organisation upon which they get motivated to work. A gap between the two may lead them to work slow or refuse to work

Maslow's Hierarchy of Needs

Maslow maslow suggested the following needs for human motivation which could be organised hierarchically as :

- Physiological needs
- Safety needs
- Social needs
- Esteem needs
- Self actualization



- **Physiological needs** : This being the lowest order includes the need for food water clothing and shelter for a person who lacks everything this could be the major motivation.This should be the first need to be satisfied.
- **Safety needs** : This is the next important after physiological needs these are the needs for protection against danger or loss of existing physiological needs.
- **Social needs** : this being III in line comprises of social needs that is giving and receiving of love friendship affection belongingness association and acceptance. If a person census the absence of his friends this will motivate him towards affectionate relations.
- **Esteem needs** : this been the fourth level can be subdivided into two categories :

1. Need for achievement strength and freedom.
2. Need for self esteem or self worth

- **Self actualization** : this being the highest level of hierarchy is the need to realise one's potential for continued self development and desire to become more of what one is capable of. This is known as self-realization or self attenuation.

Motivation hygiene theory

This theory being the basis of effective utilisation of human resources was developed by **Frederick Herzberg** of the University of Utah.

He started collecting information from various interviews that What were they asked that made them unsatisfied and what made them satisfied ?

He concluded with 2 categories of people, the satisfied and dissatisfied. Satisfied peoples are concerned about the work which is called motivation factors where has the this satisfied people are concerned about the work environment which is the hygiene factor.

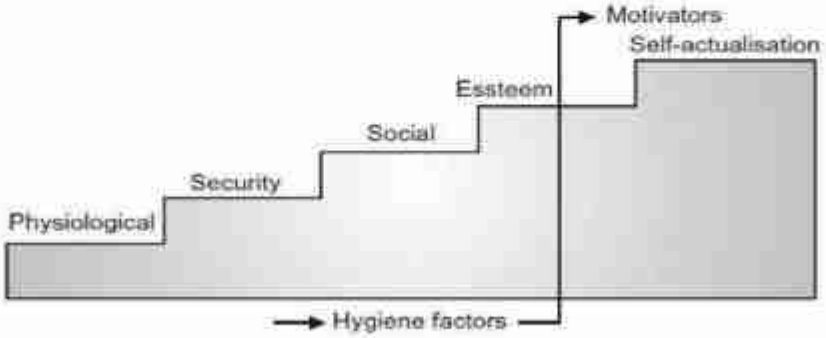
Comparison between hygiene and motivation factors

Hygiene factors	**Motivating factors**
• Environment	• The job
• Policies and administration	• Achievement
• Supervision	• Recognition for accomplishment
• Working conditions	• Challenges and responsibility
• Interpersonal relations	• Growth and development
• Money status in security	



- **Motivators** : it is the feeling of achievement or recognition in jobs. These often increase productivity.
- **Hygiene factors** : these may not be an integral part of the job, it has not produced the growth in workers output capacity But only prevent output losses in worker performance due to retraction.

The lower of maslows hierarchy of needs came under hygiene factor while esteem in half hygiene and half motivating factor. Self actualization is completely motivating factor.



Communication – Meaning and Importance

- Communication is an exchange of facts, ideas, opinions or emotions by two or more persons.
- Systems of communication
  - According to organizational structure
    - Formal
    - Informal
  - According to direction of expression
    - Downward
    - Upward
    - Horizontal
  - According to the way of expression
    - Oral or Verbal
    - Written

Coordination- Meaning and Importance

- Need to coordinate the activities to achieve common objectives.
- Coordination deals with the task of blending efforts in order to ensure successful attainment of an objective. It is accomplished by means of planning, organizing directing and controlling.
- Importance of coordination
- Unity in diversity
- Specialization
- Team work
- Large number of employees
- Empire building
- Functional differences

- Recognition of goals
- Communication
- Interdependency
- Techniques of coordination
    - Rules, Procedures and Policies
    - Planning
    - Hierarchy
    - Direct Contact
    - Task Force
    - Committees
    - Induction
    - Indoctrination
    - Incentives
    - Liaison Department
    - Workflow

Controlling – Meaning

- Controlling is to identify the variations between the set standards and actual performance and then to take necessary steps to correct it and prevent such deviations in future.
- Control is checking current performance against predetermined standards contained in the plans, with the view to ensuring adequate progress and satisfactory performance.
- Controlling is determining what is being accomplished, that is, evaluating the performance and if necessary applying corrective measures so that the performance takes place according to plans.

Steps in Controlling

- Three steps in control process are
- Establishing standards
    - The standards must be established against which results can be measured. Standards may be qualitative or quantitative. Following types are :
        - Physical standards
        - Cost standards
        - Revenue standards
        - Capital standards
        - Intangible standards
        - Working practice standards
- Measuring and comparing actual results against standards
    - Measurement of performance can be done by personal observation or by a study of various figures, reports, charts and statements. Comparison is very easy if the system of control is well defined.
- Taking corrective action
    - The management has to take necessary corrective action based on the nature of causes of deviation.

MODULE- 3

Social Responsibilities of Business & Entrepreneurship

Meaning of Social Responsibility

- Social responsibility is the manager's responsiveness to public agreement.
- Two types of business obligations are:
  - Social-economic obligation
    - The economic consequences of its action do not affect public welfare.
  - Social-human obligation
    - To nurture and develop human values.
- Different views on Social Responsibility
  - Communist view
  - Capitalist view
  - Pragmatic view
  - Trusteeship view

Social Responsibilities of Business towards Different Groups

- Towards Employees and Workers
  - A fair wage to the workers
  - Just selection, training and promotion
  - Social security measures and good quality of life
  - Good human relations
  - Freedom, self-respect and self-realisation.
  - Increase in productivity and efficiency by recognition of merit, by providing opportunities for talent.
- Towards shareholders and other businesses
  - Promoting good governance through internal accountability and transparency.
  - Fairness in relations with competitors.
- Towards the state
  - Shunning active participation in and direct identification with any political party.
  - Observing all the laws of land.
  - To provide safeguard against errant business practices.
  - To allocate limited resources according to social priorities and preferences.
  - To enforce distributive justice to weaker selection of community.
  - To implement rural uplift and secure balanced development of the country.

Social audit

- A systematic study and evaluation of the organizations social performance as distinguished from its economic performance.
- Benefits
  - It supplies data for comparison with the organization's social policies and standards.
  - It develops a sense of social awareness among all employees.
  - It provides data for comparing the effectiveness of different types of programmes.
  - it provides information for effective response to external groups which make demands on the organization.

- Limitations
  - A social audit is a process audit rather than an audit of results. It determines only what an organization is doing in social areas.
  - They are difficult to measure.
  - Their classification under good or bad is not universally accepted.
  - Most of them occur outside the organization, makes difficult to secure data.

Business Ethics and Corporate Governance

- **Business Ethics**
- Business ethics is the application of moral principles to business problems. It involves goodness or badness of an act. The four important factors are:
  - Government legislation
  - Business codes
  - Pressure groups
  - Personal values of the manager himself.
- **Corporate Governance**
- In this companies run in an open and honest manner in the best interest of all stake-holders. The key elements are transparency and accountability. Major recommendations of this committee are:
  - Formal selection process
  - Companies should have remuneration committees
  - Companies should have Audit committees
  - Audit partners should be rotated.
- Benefits
  - It creates overall market confidence and long –term trust in the company.
  - It leads to an increase in company's share prices.
  - It ensures the integrity of company's financial report.
  - It maximizes corporate security.
  - It limits the liability of top management.
  - It improves strategic thinking at the top.

Entrepreneurship

- **Definition of Entrepreneur**
- The entrepreneur is essentially a visualizer and an actualizer. He can visualize something, and when he visualizes it he sees exactly how to make it happen.
- An individual who bears the risk of operating a business in the face of uncertainty about the future conditions.
- He is the one who innovates, and introduces something new in the economy.
- An entrepreneur is a person who buys factor services at certain prices with a view to selling its product at uncertain prices.
- **IMPORTANCE OF ENTREPRENEURSHIP**
- The significance are economic development, industrial development, regional growth and employment generation.

- Entrepreneur provides opportunities to unemployed youth, increase in per capita income, higher standard of leaving and increased individual savings, revenue to the government in the form of TAX.
- They invented new products. Also to capitalize on new opportunities and to create wealth and new jobs.
- In an entrepreneurship, it is you who set the goals, plan the action and reap the satisfaction and rewards of having achieved the goal.
- If you become an entrepreneur
- You will be your own boss and boss to other people and make decisions that are critical to the business success or failure.
- You will have the chance to put your ideas into practice.
- You will make money for yourself rather than for someone else.
- You will participate in every aspect of running a business and learn and gain experience in a variety of disciplines.
- You will have chance to work directly with your customers.
- You will have the personal satisfaction of creating and running a successful business.
- You will be able to work in the field or area that you really enjoy.
- You will have the chance to build retirement value.
- **Difference between a Manager and an Entrepreneur.**

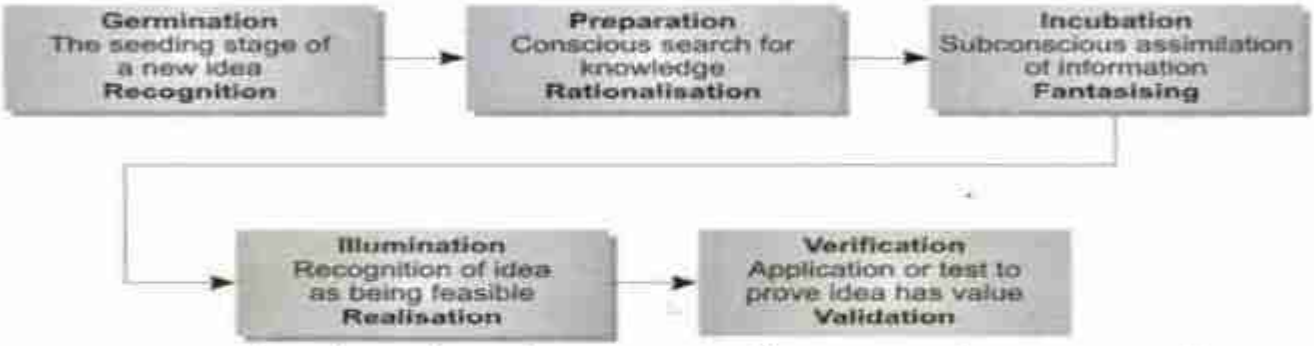
Entrepreneur	Manager
Own boss	Salaried employee
Takes own decisions	Executes the decisions of the owner
Hires employees	He is an employee
Uncertain rewards which can be unlimited.	Fixed rewards and salary

CONCEPTS OF ENTREPRENEURSHIP

- Entrepreneur is a person who creates an enterprise. The process of creation is called as entrepreneurship.
- Entrepreneurial development is a key to socio-economic transformation of the region.
- Entrepreneurship can be defined as a process of action an entrepreneur undertakes to establish his enterprise.
- The word entrepreneur is derived from the French Verb *enterprendre*, means *to undertake*. This refers to those who undertook the risk of new enterprises.
- Entrepreneurial development process is in five stages.
    - Stage 1: Perceiving, identifying and evaluating an opportunity.
    - Stage 2: Drawing up a business plan.
    - Stage 3: Marshalling resources.
    - Stage 4: Creating the enterprise.
    - Stage 5: Consolidation and management.

CHARACTERISTICS OF A SUCCESSFUL ENTREPRENEUR

- The characteristics are:
    - Creativity: The ability to bring something new into existence.
    - The creative process is



    - Innovation: The act that endows resources with a new capacity to create wealth.
- Dynamism: Dynamism constitutes a potential contribution and revises the targets of the enterprise.
- Leadership: Leadership is the basic quality of an entrepreneur and takes him forward.
- Teambuilding: An entrepreneur should have an ability to build a team. A team is group of individuals with a common purpose. The skills are:
    - Wanting to feel better
    - Identifying the problem and needs of the enterprise
    - Creating a vision
    - Setting goals for the group
    - Reviewing progress
- Achievement motivation: Entrepreneurs have a high need for achievement and are guided by their inner self, motivating their behavior towards accomplishment.
- Problem solving: the ability to solve problem and not to avoid them. The six steps are:
    - Define a problem
    - Gather information
    - Identify various solutions
    - Evaluate alternatives and select the best option.
    - Take action
    - Evaluate the action taken.
- Goal orientation: Goal setting is the achievement of targets and objectives for successful performance of an entrepreneur. The three steps are:
        - Definition of goal
        - Specific goals
        - Feedback about goal achieved
    - *Aspects of goal setting*
        - Clarity of goals
        - Reformulation of goals
        - Restating goals
        - Plan of action
        - Defining standards of performance and measurement

- Recognizing risk and obstacles
- Goal reaffirmation
- Goal attainment.

- Types of Goals
  - Target oriented
  - Achievement oriented
  - Specific
  - Overall
  - Secondary
  - Long range/ short range
  - Personal/ social
- Risk taking and decision- making ability: Entrepreneurs are persons who take decisions under conditions of uncertainty, and therefore are willing to bear risk, but never gamble with results.
- Commitment: The amount of willpower.

CLASSIFICATION OF ENTREPRENEURS

a) Based on Functional Characteristics
   1. Innovative entrepreneur
   2. Imitative or adoptive entrepreneur
   3. Fabian entrepreneur
   4. Drone entrepreneur

b) Entrepreneurial Core Competencies

	Core competencies	Entrepreneurial activities
1.	Initiative	Does things before asked for or forced to by events and acts to extend the business to new areas, products or services.
2.	Perceiving opportunities	Identifies business opportunities and mobilizes necessary resources to make good an opportunity.
3.	Persistence	Takes repeated or different actions to overcome obstacles.
4.	Information gathering	Consults experts for business and technical advice. Seeks information of client or supplier's needs. Personally undertakes market research and make use of personal contacts or information networks to obtain useful information.
5.	Concern for quality work	States desire to produce or sell a better quality product or service. Compares his performance favorably with that of others.
6.	Commitment to contractual obligations	Makes a personal sacrifice or expands extraordinary effort to complete a job, accepts full responsibility in completing a job contract on schedule, pitches in with workers or work in their place to get the job done and shows utmost concern to satisfy the customer.
7.	Efficiency orientation	Finds ways and means to do things faster, better and economically.

8.	Planning	Various inter-related jobs are synchronized according to plan.
9.	Problem solving	Conceives new ideas and finds innovative solutions.
10.	Self-confidence	Makes decisions on his own and sticks to it in spite of initial setbacks.
11.	Experience	Possesses technical expertise in areas of business, finance, marketing, etc.
12.	Self-critical	Aware of personal limitations but tries to improve upon by learning from his past mistakes or experiences of others and is never complacent with success.
13.	Persuasion	Persuades customers and financiers to patronize his business.
14.	Use of influence strategies.	Develops business contacts, retains influential people as agents and restricts dissemination of information in his possession.
15.	Assertiveness	Instructs, reprimands or disciplines for failing to perform.
16.	Monitoring	Develops a reporting system to ensure that work is completed and quality norms.
17.	Credibility	Demonstrates honesty in dealing with employees, suppliers and customers even if it means a loss of business.
18.	Concern for employee welfare	Expresses grievances. concern for employees by responding promptly to their.
19.	Impersonal relationship	Places long-term goodwill over short-term gain in a business relationship.
20.	Expansion of capital base	Reinvests a greater portion of profits to expand capital of the firm.
21.	Building product image	Concerned about the image of his products among consumers and does everything possible to establish a niche for his products in the market.

b) Based on the Development Angle

1. Prime mover
2. Manager
3. Minor innovator

4. Satellite
5. Local trading

c) Based on Types of Entrepreneurial Business.
1. Manufacturing
2. Wholesaling
3. Retailing
4. Service

d) Based on the Nine Personality Types of Entrepreneurs
1. The Improver
2. The Advisor
3. The Superstar
4. The Artist
5. The Visionary
6. The Analyst
7. The Fireball
8. The Hero
9. The Healer

e) Based on schools of Thought on Entrepreneurship

Entrepreneurial model	*Central focus or purpose*	*Behaviours and skills*	*Situation*
"Great Person" school	The entrepreneur has an intuitive ability – a sixth sense–and traits and instincts he is born with.	Intuition, vigour, energy, persistence, and self-esteem	Start up
Psychological Characteristics school	Entrepreneurs have unique values, attitudes, and needs, which drive them.	Personal values, risk taking, need for achievement, and others	Start up
Classical school	The central characteristic of entrepreneurial behaviour is innovation.	Innovation, creativity, and discovery	Start up and early growth
Management school	Entrepreneurs are organisers of an economic venture; they are people who organise, own, manage, and assume the risk.	Production planning, people organisation, capitalisation and budgeting	Early growth and maturity
Leadership school	Entrepreneurs are leaders of people; they have the ability to adapt their style to the needs of people.	Motivating, directing, and leading	Early growth and maturity
Intrapreneurship school	In corporations intrapreneurship is the development of independent units to create, market and expand services.	Alertness to opportunities, maximising decisions	Maturity and change

INTRAPRENEUR:

- Intrapreneurship is the entrepreneurship with in an existing business structure.
- These *"New Breed"* of Entrepreneurs are emerging from Big Companies.
- In this Case the Entrepreneurial Spirit of Small Companies is transferred to Big Companies.
- Such Entrepreneurs of Big Companies are called as *"Intrapreneurs"*, an Emerging Class of Entrepreneurs.
- *Intrapreneurship* is a Method of Stimulating & then Capitalizing on Individuals in an Organization *who think that something can be done differently & better.*

- The differences in the entrepreneurial and managerial domains have contributed towards an increased need for intrapeneureship.
- This need is growing due to variety of events occurring on social, cultural and business levels.
- In social level, there is an increasing interest in 'doing your own thing' and doing it on one's own terms.
- Some individuals having self confidence, self motivation and belief in their own talent, often desire to innovate on their own.
- They want own responsibility and to work in their own way.
- They become frustrated if this freedom is not given to them and get demotivated.
- Intrapreneurship is "one such method of providing freedom, simulating and capitalizing on individuals in an organization who think that things can be done in different and better way".
- The resistance against flexibility, growth and diversification can be overcome by developing a spirit of entrepreneurship within the organization which is called as intrapreneurship.
- Intrapreneurship is reflected more strongly in entrepreneurial activities and also in top management.

Elements of intrapreneurship :

1. New business venturing:
   - The creation of new business with in the organization.
   - Redefining the company's product or service,
   - Development of new market segment.
2. Innovations:
   - Innovation is the development of new product.
   - Improvement of existing products,
   - Development and simplified production methods and procedures.
3. Self – renewal:
   - The transformation of an organization through the renewal of main ideas.
   - This includes a redefinition of a business concept,
   - Reorganization or modifications in the system with an aim to initiate innovating.
4. Proactiveness:
   - Proactiveness includes initiative and risk taking.
   - Competitiveness and dashing to take new challenges.

Brief comparison of Intrapreneurs, Entrepreneurs and Managers is given below:

Point	Intrapreneur	Entrepreneur	manager
1. Goal management	Independent, innovates new ideas	Independent, stars new venture and leads direct involvement	Delegates and supervises more than direct involvement .
2. Status	Not concerned about traditional status, but wants recognition	Not concerned about status	Concerned about status symbol

3. Risk	Owns moderate risk	Bears all the risk and uncertainty.	Does not bear any risk
4. Rewards	Gets fixed reward for his work. Many get extra for his work.	Since there is risk, he may get profit or loss depending his innovations	Workers for salary for his service which is fixed and definite.
5.Innovation	Innovative	Very innovative	Need not be innovative. He manages the ideas of top management.
6. Decision making	Moderate, limited to his work.	Very much involved in decision making.	Delegates the decisions of top management.

Myths of Entrepreneurship

- Entrepreneurs are born, not made
- Entrepreneurs are academic and social misfits
- Entrepreneurs fit an ideal profile.
- All you need is money to be an Entrepreneur.
- All you need is luck to be an Entrepreneur.
- A great idea is the only ingredient in a recipe for business.
- My best friend will be a great business partner.
- Having no boss is great fun.
- I can make lots of money
- I will definitely become successful
- Life will be much simpler if I work for myself.

Entrepreneurial Development Models

- The models suggested for the development of Entrepreneurship are:
    - Psychological Models
    - Sociological Models
    - Integrated Models
    - Psychological Models:
        - This model signifies that psychological factors are responsible for the development of entrepreneurial behavior in individuals.
        - McClelland: Importance to achievement motives
        - Latent need for achievement among adults
        - Motivation-training programme to make entrepreneurs willing and eager to exploit the opportunities provided
        - Creative Personality - status withdrawal (Everett Hagen's theory of social change)
        - John Kunkel – entrepreneurial supply as a function of the surrounding social structure. Can be influenced by manipulating economic and social incentives.
- Sociological models

- This model considers societal factors responsible for the development of entrepreneurial behavior in individuals.
- Young's theory: Change based upon Society's incorporation of relative subgroups
- Relativeness of a subgroup which has a low status in a larger society will lead to entrepreneurial behavior if the group has better institutional resources than others in the society at the same level.
- Suggests the creation of supporting institutions in society as the determinant of entrepreneurship

- Integrated models
- – TV Rao – Entrepreneurial disposition
  - Need for motive
  - Long-term involvement either at thinking level or activity level
  - Personal, social and material resources
  - Suitable socio-political system for establishment and development of enterprise

Five stages for promoting small entrepreneurship

- Stimulation
- Identification
- Development
- Promotion
- Fellow up

Entrepreneurial Development Cycle

- The activities are:
  - Stimulatory activities
  - Support Activities
  - Sustaining Activities



- Stimulatory activities: These activities ensure the emergence of entrepreneurs in the society. The activities are:
  - Entrepreneurial education
  - Planned publicity for entrepreneurial opport-unities.
  - Identification of potential entrepreneurs through scientific methods.
  - Motivational training to new entrepreneurs.
  - Help and guidance in selecting products and preparing project reports.
  - Making available techno-economic information and product profiles.

        - Evolving locally suitable new products and process.
        - Creating entrepreneurial forums.
        - Recognition of entrepreneurs.
    - Support Activities : These activities help a person to develop into an entrepreneur. They nurture and help entrepreneurship to grow. The activities are:
- Registration of unit
- Arranging finance
- Providing land, shed, power, water and so on.
- Guidance for selecting and obtaining machinery.
- Supply of scare raw materials.
- Getting licenses/ import licenses
- Providing common facilities
- Garneting tax relief or other subsidy.
- Offering management consultancy services.
- Help marketing the product.
- Providing information.

- Sustaining activities: These activities are all those that help in the continuous and efficient functioning of entrepreneurship. The activities are:
    - Help modernization
    - Help diversification/ expansion/ substitute production.
    - Additional financing for full capacity utilization.
    - Deferring replacement/ interest.
    - Diagnostic industrial extension/ consultancy services.
    - Production unit legislation/ policy change.
    - Quality testing and approving services.

Problems faced by Entrepreneurs and capacity building for Entrepreneurship.

- The problems faced by entrepreneurs are:
    - Internal Problems
    - External Problems
    - Specific management Problems

Internal Problem

- Planning
    - Technical feasibility
    - Economic viability
- Implementation
- Production
    - Production management
    - Labour management
    - Marketing management
    - Financial management
    - Administrative management

External Problems

- Infrastructural
- Financial
- Specific management problems
    - Management deficiency
    - Finance
    - Manufacturing and technical problems
    - Product planning
    - Selection of equipment, plant and machinery
    - Human resource development
    - Technical know-how
    - Preparation of project report.

Capacity building for Entrepreneurship

- To build capacity for entrepreneurship following measures can be taken such as availability of credit, imported raw materials, skilled labour, technology and equipment, infrastructural facilities, advisory services and access to market.
- Along with to
    - Create the right environment for success.
    - Ensure that entrepreneurs have access to the right skills.
    - Ensure that entrepreneurs have access to smart capitals.
    - Enable networking and exchange

MODULE- 4

Modern Small Business Enterprises & Institutional Support for Business Enterprises

Modern Small Business Enterprises

- Small businesses are privately owned corporations, partnerships, or sole proprietorships that have fewer employees and/or less annual revenue than a regular-sized business or corporation.
- Businesses are defined as "small" in terms of being able to apply for government support and qualify for preferential tax policy varies depending on the country and industry.
- Include service or retail operations such as convenience stores, small grocery stores, bakeries.

Role of Small Scale Industries

- Dynamic and vibrant sector of the economy.
- Prominent role in socio-economic transformation of the country.
- Accounts for 35 percent of industrial production, 40 percent of exports and 90 percent of employment opportunities.
- The Small scale industry sector has, over the past six decades, acquired a prominent place in the socio- economic development of the country.
- Importance of Small-scale Industries
    - Provide increased employment through labour-intensive process.
    - Require lower gestation period.
    - Easy to set up in rural and backward areas.
    - Need small/ local market.
    - Encourage growth of local entrepreneurship.
    - Create a decentralized pattern of ownership.
    - Foster diversification of economic activities.
    - Introduce new products particularly to cater to local needs.
    - Influence the standard of living of local people.
    - Provide equitable dispersal of industries throughput rural and backward areas.
- Impact of Globalization and the world Trade Organization (WTO) on Small-Scale Industries (SSI)
    - Tighter patent laws through regulation of intellectual property rights under the Trade-Related Aspect of Intellectual Property Rights (TRIPS) agreement.
    - Worldwide market and opportunity to export.
    - Intensified competition in the domestic market.
    - Import of high quality goods.

Characteristics of SSI

- Capital investment is small and most of them have small number of workers
- Generally owned by a single or at the most two persons and engaged in production of small goods
- Most of them are family owned business
- Workers are not well recognized and they may do different types of works as need arises
- Funded by owner's savings or short-term loans
- SSI activity is mainly dependent on owner's entrepreneurship
- Exploitation of human and natural resources

- Generally management and organizations are very poor
- Profit margin is less
- Innovation and risk bearing is high
- Generally found in urban and semi urban areas.

OBJECTIVES OF SSIs

- Creation of employment opportunity
- Improvement of output , income and better standard of living
- Elimination of economic backwardness of rural and underdeveloped areas.
- To reduce regional imbalance
- To provide employment and means of a regular source of income to the needy people living in rural and semi-urban areas
- To improve the quality of industrial products produced in cottage industries and increase production and profits.
- To facilitate import substitution.
- To encourage entrepreneurship and self-reliance.
- To mobilize regional resources of capital

Concepts and definitions of SSI Enterprises

- Small-scale Industrial Unit : An industrial undertaking in which the investment in fixes assets in plant and machinery. Does not exceed Rs.100 lakh.
- Ancillary Unit: This is a sub-class of SSIs. An industrial undertaking which is engaged or is proposed to be engaged in
    - The manufacture of parts, components.
    - Rendering of services, or supplying.
- Export-oriented Unit (EOU) : The SSI has an obligation to export 30% of production.
- Tiny Unit: The investment limit in plant and machinery in respect of tiny enterprises is Rs. 25 Lakh.
- Women Entrepreneurs: 51% equity holding by women
- Small-scale services and Business (industry-related) Enterprises (SSSBEs): SSSBEs with investment in fixed assets, excluding land and building, up to Rs. 10 Lakh. The services are advertising agencies, Marketing consultancy, Industrial consultancy etc.
- Activities not recognized as SSIs and SSSBEs : Transportation, storage, Retail/wholesale, Hotels etc.

Definition of SSI

- Conventional Definition: cottage and handicraft industries that employ conventional labour–oriented methods to produce conventional products, mainly in the rural areas. Some examples are handloom and handicrafts.
- Operational definition: "all the undertakings having an investment in fixed assets in plant and machinery, whether held on ownership terms or by lease or hire-purpose, not exceeding Rs. 60 lakhs".
- National Income Accounting: A unit engaged in manufacturing in manufacturing , servicing, processing, servicing and preservation of goods having investment in plant and machinery, at an original cost not exceeding Rs.60 lakhs"

Government policy and development of the Small Scale sector in India

- Administratively, India's SSI sector is divided into seven industry groups
- Traditional Sector:

1. Handicrafts
2. Handlooms
3. Khadi, village and cottage industries
4. Coir
5. Sericulture

- Modern Sector

6. Power looms
7. Residual small-scale industries

- Indian definition of small-scale enterprises (SSE) based largely on cumulative amount of investment in plant and machinery whose limits have been periodically revised upwards.
- Emerging liberalized and competitive economic environment necessitated structural and fundamental changes in the policy framework, causing a shift in focus from protection to promotion.
- Govt. has announced several objectives and intentions through SSI industrial Policy Resolutions(IPRs).
- Different Policies

IPR 1948:

- During 1948, SSIs are suited for using local resources to create employment for rural areas.
- Initially SSI faced lot of problems like shortage of raw materials etc.
- Main thrust of IPR 1948 was to provide protection to SSIs

IPR 1956:

- It provided continuing policy support to the small sector and aimed to ensure that decentralized sector gained enough self support.
- About 128 items were reserved for exclusive production in small sector.
- Small Scale Industries Board(SSIB) constituted a working group in 1959 for the purpose of examining and formulating a development plan for SSIs
- During this policy a specific developmental projects like "Rural Industries Projects" and "industrial Estate Projects" were started to strengthen SSIs.
- IPR-1956 for SSIs aimed at "Protection and Development" and initiated the modern SSI in India.

IPR 1977

- The main thrust was on effective promotion of cottage and small scale industries widely spread in rural and urban areas.
- IPR 1977 classified small scale into 3 categories:
    - Cottage and house hold industries
    - Tiny sector: Investment within 1 lakhs
    - SSI: Investment up to 15 lakhs

IPR 1980

- Main objective of IPR 1980 was defined as facilitation an increase in industrial production.

- It helped small sector by increasing the limit from Rs. 1 Lakh to Rs. 2 lakhs for tiny industries, from 10 lakhs to 20 lakhs in small scale units and from Rs. 15 lakhs to 25 lakhs for ancillaries (Providing necessary support to the primary activities of operation)

IPR 1990

- It gave lot of scope for SSIs to generate wage and self employment based opportunities.
- Investment ceiling in plant and machinery for tiny units is increased form 2 to 5 lakhs, 35L to 60L and 45L to 75L.
- Implemented a new scheme of central investment subsidy for SSIs.
- SSI technology up gradation programs are implemented under technology development centre in Small industries Development Organization(SIDO)

New small Enterprise policy 1991(Industrial Policy)

- Govt. of India issued a new policy named "Policy measures for Promoting and strengthening and supplementing small tiny and village enterprises.
- Its main theme is to improve growth rate of SSI which in turn improves country's growth.
- Implement Small Industry Development Organization(SIDO)
- Set up of Export development centre(EDC) in SIDO
- Small Industries Development Bank of India(SIDBI) was established in 1990 to provide credit facilities to SSIs.
- Emphasis to establish special cell in SIDO for developing and training for women entrepreneurs.
- Policy also proposed to encourage and support industry associations to establish counseling and common testing facilities and to pursue orientation programs of modernization etc.

Policy Initiatives since 1999

- Economic scenario changed and competitive environment causing a shift in focus from protection of SSIs to promotion.
- Govt. took number of steps including change in investment limits, facilities for foreign participation, export promotion, marketing assistance and so on.
- More Important steps are:
    - A new ministry of small scale industries and agro and rural industries was created on oct. 14, 1999.
    - A new credit insurance scheme was announced in the budget 1999-2000 for SSI security to banks and to improve flow of investment.
- Banks on the basis of 20% of their annual turnover determine the capital limit for SSI units. From 4 cr. to 5 cr.
- A national programme to boost rural industrialization has been announced to set up 100 rural clusters every year.

IPR 2000

- Industrial Policy 2000 was actually a comprehensive policy package for SSIs
    - Credit linked capital subsidy of 12% against loans for technology up-gradation was provided in specific industries
- The limit of Composite loan was increased from 10L to 25L.
- The family income eligibility limit of 24000 was enhanced to 40000 per annum under Prime Minister Rozgar Yozna (PMRY).

- The limit of Investment was increased in industry related service and business enterprise from 5lakhs to 10L

IPR 2001-2002

- Investment limit was enhanced from 1Crore to 5 cr. for units in handloom sectors.
- Credit guarantee fund scheme was increased from 125 Cr. to 200 cr.
- 14 items were de-reserved during June 2001 related to leather goods and toys.
- Market development assistance scheme was launched exclusively for SSI sector

Policy initiatives on SSI sector in 2003-2004

- 75 items reserved for manufacturing consisting of chemicals, leather products, laboratory reagents in 2003.
- Selective enhancement in plant & machinery form 1crore to 5 crore : 13 items in stationary and 10 items in the drugs and pharma.
- Union budget 2003-04 announced banks would provide interest rate of 2%.
- Composite loan limit for SSI enhanced from 25L and 50L.
- The lower limit of 5L on loans covered under credit guarantee scheme has been removed. All loans upto 25L made eligible are under credit guarantee scheme.
- 417 specialized bank branches made operational throughout country.
- Small and Medium Enterprise(SME) fund of 10000 crore was set up under SIDBI to solve problem of inadequate finance for SSIs.
- Laghu Udyami Credit card scheme was liberalized. Under this scheme the credit limit was increased to 10L from 2L.

IPR 2003-04

- It gave reservation for chemical and chemical products, leather products and laboratory reagents etc.
- Enhancing investment from 1 cr. to 5 cr.
- Loan limit to SSI was increased from 25L to 50L
- Specialized bank branches were made operational for SSIs.

IPR 2004-05

- It suggested measures considered necessary for improvement in the productivity of enterprises, generation of large scale employment opportunities, raw material supply, technology up gradation, skill development by training.
- The reserve bank of India raised the composite loan limit from Rs. 50L to 1Crore.

IPR 2005-06

- The ministry of SSI has identified 180 items for De-reservation
- Small and Medium enterprises were recognized in the services sector and were treated on par with SSIs in manufacturing sector.
- Technological facilities have been increased.
- Tax concessions for SSIs and so on.

Growth and Performance of Small Scale Industries in India

- Pre- and Post- liberalization Periods.
    - The potential for creating employment
    - The output level of SSI went an increase of nearly 20 times over a period of 20 years.

    - Exports have gone up by nearly 40 times.
    - The pace of growth was relatively lower compared to that of the pre-liberalization period.
- Performance of small-scale Industries in India.
    - The SSI contributes to the GDP
    - The value of production has increased.
    - The sector is estimated to grow at the rate of 7.5%
- Share of small-scale Industrial Units in Total Exports.
    - With the growing trend towards liberalization and globalization, export is one of the focus areas of SSI units.
    - The commodity-wise export of major products by SSI units has also increased over a period of time
- Sickness in SSI sectorSickness was identified through the latest definition of RBI given by the Kohli Committee. Incipient sickness was identified in terms of continuous decline in gross output.
- Sickness in the total SSI sector was to the tune of 1%, whereas in the registered and unregistered SSI sector it was 3.38% and 0.64% respectively.
- The maximum number of sick units-about 59.33% was located in West Bengal, Kerala, Maharashtra, Karnataka and Andhra Pradesh.
- Out of the units having lone outstanding with institutional source such as banks and financial institutions, sickness was reported to be about 19.6% in the registered SSI sector and 16.61% in the unregistered SSI sector. In the total SSI sector, this percentage was 17.8.
- Incipient sickness, Identified in terms of a continuous decline in gross output was 11.5% in the registered SSI sector and 6.48% in the unregistered SSI sector. In the total SSI sector, this percentage was 7.4.
- Combining the three yardsticks used to measure sickness:
    - Delay in repayment of loan over one year.
    - Decline in net worth by 50%
    - Decline in output in last three years, about 13.98% of the units in the registered SSI sector were identified to be either sick or incipient sick.
- This percentage was only 6.89 in the case of unregistered units. In the total SSI sector, this percentage was 7.82.
- The States of Kerala, Tamil Nadu, Andhra Pradesh, Karnataka, and Maharashtra had maximum number of sick/incipiently sick SSI units. These five states together accounted for 54.28% of the sick/incipiently sick SSI units in the country.
- "Lack of demand" and "Shortage of working capital" were the main reasons for sickness/incipient sickness in both the registered and unregistered SSI sectors.

Problems for Small Scale Industries

- The problems of small industries may be divided into groups – External and Internal. External problems are those which result from factors beyond the control of the industrialist such as the availability of power and other infrastructure facilities required for the smooth running of small-scale industries. Internal problems are those which are not influenced by external forces.
- Internal

    - Choice of idea
    - Feeble structure
    - Faulty planning
    - Poor project implementation
    - Poor management
    - Poor production
    - Quality
    - Marketing
    - Inadequate finance
    - Labour problems
    - Capacity utilization
    - Lack of vertical and horizontal integration
    - Inadequate training in skills
    - Poor and loose organization
    - Lack of strategies
- External
- Infrastructure
    - Location
    - Power
    - Water
    - Post Office and so on
    - Communication
- Financial
    - Capital
    - Working capital
    - Long-term funds
    - Recovery
- Marketing
- Taxation
- Raw Material
- Industrial and financial regulations
- Inspections
- Technology
- Policy
- Competitive and volatile environment
- Problems faced by SSI
- Difficulty in obtaining credit from commercial banks because of their general inability to provide security.
- Inability to offer liberal credit terms in the sale of their products.
- Absence of management expertise. Often managed by one person who performs a number of functions usually with no formal training.

- Difficulty in competing with imported products due to high production costs.
- Competition from other local entrepreneurs in the same line of business competing for the limited local market.
- Difficulty in obtaining industrial land in towns and cities. The shortage of industrial land is giving rise to more and more backward operations.
- Under capitalization
- Difficulty in identifying appropriate technology and technical assistance.
- Bureaucratic red tape and regulations.
- Identification of Industrial projects for development
- Project preparation and evaluation.
- Financial or credit support and investment promotion.
- Consultancy and counseling services.
- Development of infrastructure of various kinds in the appropriate areas.
- Entrepreneurship development
- Industrial training and skill formation
- Quality control and testing facilities.
- Market promotion, both domestic and export.
- Procurement of raw materials and equipment.
- Productivity increase through modernization.
- Incentive measures, by industry and by area.
- Local initiative.
- Regional and International technical and financial assistance.
- Cooperation among developing countries.

Impact of liberalization, privatization and Globalization on SSI

- Started in 1991
- Indian economy had grown to new heights
- Liberalization
    - Refers to the relaxation of the previous government, restriction usually in the area of social economic policies
    - Import of scarce and non-available raw materials
    - Best example: Electronic and computer industries
- Privatization
    - Privatization means transfer of ownership and/or management of an enterprise from the public sector to the private sector
    - ITI and LIC
- Globalization
    - Globalization implies integration of the economy of the country with the rest of the world economy and opening up of the economy for foreign direct investment by liberalizing the rules and regulation and by creating favorable socio – economic and political climate for global business
    - Export

Impact of WTO/GATT on SSIs

- General Agreement on Tariffs and Trade (GATT)
- Multilateral global initiative
- Agreement signed by group of 123 nations to promote trade among members in 1947
- political disputes and problems could be solved my GATT
- USA and British made it possible
- The challenges are
    - E-commerce
    - Agriculture commodities
    - Narcotics, nuclear and dangerous materials.
    - Political and regional forces on trade and industry.
    - Development of backward regions.
    - Barter deals.
    - Counter trade.
    - Bilateral trade agreement and monopolies
    - Regional trade blocks
    - Services
    - Trade information
    - Trade disputes in international business
    - High tariffs by some countries.
    - Trade restrictive practices.
    - Subsidies

World Trade Organization (WTO)

- WTO was establish in January 1st, 1995 as a successor to the GATT.
- It Serves as the legal and institutional foundation of the multilateral trading system.
- The Uruguay Round Table Agreement that created WTO also eliminated tariffs for some manufactures' goods ,reduced barriers to trade in agriculture ,expanded protection for copy rights, patents and other intellectual property, provided some reduction in barriers to services and foreign investment.
- It also reformed the multilateral trade process and included strong dispute settlement mechanism.
- WTO has 123 members, another 31 countries have applied for membership 45
- **Objectives of WTO**
    - Make Optimum utilization of world's resources for improving the incomes standard of living , promoting employment and expanding production and trade activities among various countries.
    - Encourage sustainable development and ensure that environment care and development activities together.
    - Efforts should be made that poor countries as well as developing countries get an opportunity to have a share in the growth in the world trade.
- **Functions of WTO**

- It covers all the commodities that are internationally traded and have formulated rules and procedures for each category as a guidelines to member countries.
- Though WTO does not interfere in economic and political issues of the member countries , it does involve in case of policies regarding international trade.
- WTO has various expert committees and subcommittees for different categories to review various subjects and also to work as consultants.
- It arranges visits of its experts to various countries for training development activities.
- WTO acts as an arbitrator to sort out disputes between various countries.
- It reviews and advises trade policies of various countries so that such policies are conducive for international trade.
- Implementation and monitoring of multilateral and bilateral trade agreements which are essence of the WTO.
- To keep a track of trade related activities, member countries will inform WTO about various trade activities they have been doing in the global market.
- Evaluation of international trade and seek explanation wherever abnormal variations in terms of over dominance or very 48 poor performance are observed.
- **Advantages of WTO**
    - Safe guards from unilateral actions of the developed countries.
    - Increased access to export markets.
    - Increased R&D efforts in the country.
    - Access to advanced technology to the existing and new industries.
    - Increased opportunity to sub contracting and job work for SSI.
    - Increased global competition resulting higher efficiency and improved quality.

Ancillary Industry and Tiny Industry

- Ancillary Industry
- Ancillary industries are small industries having investment in fixed assets, plant and machinery not exceeding Rs.75 lakhs and engaged in
    - Manufacturing of parts , components , sub assemblies ,tooling or intermediates
    - The rendering of services , supplying ,rendering or proposing to supply or render 30% production of the total services .
- All the service –oriented enterprises are now eligible for registration as SSIs and are entitled for the same concessions and incentives that are offered for SSIs.
- Several clarifications are issued on the above definitions of ancillaries by government from time to time. Some of them are:
    - In calculations of plant and machinery , the original price paid by owner irrespective of new or old is to be considered for accounting.
    - The cost of equipment.
    - Units producing intermediates are considered as ancillaries.
    - Units providing service facilities like machining , pressure cleaning , sand blasting etc., are recognized as ancillaries.
- Advantages of Ancillary Units
    - Providing an effective solution to the marketing problems of SSIs

- Complementary with regard to output and enable reduction in the production cost.
- Employment generation and utilization of money in untapped hands.
- Reduction in gestation period of production.

- **Tiny Industry**
- A unit is treated as Tiny Industry where investment in plant and machinery does not exceed Rs.5lakhs.
- The growth in Tiny industries –facilitates self – employment , results in wider dispersal of industrial and economic activities and ensures maximum utilization of local resources.
- **ADVANTAGES**
  - Utilization of local resources like men and material
  - Limited capital to start the industry
  - Less risk involved
  - Smaller gestation period
  - Production of goods as per choice of consumers
  - Exports, there by earning foreign exchange
  - Employment generation to rural people
  - Regional development
  - Entrepreneurial development

Institutional Support for Business Enterprises



Institutions Supporting Small-scale Industries

CENTRAL LEVEL INSTITUTIONS

- Small-scale Industries Board (SSI Board)
- Constituted in 1954 to facilitate the coordination and inter-institutional linkages for the development of SSI sector
- The Board is an apex advisory body constituted to render advice to the government on all issues pertaining to the SSI sector
- The office of the Development Commissioner (Small-Scale Industry) serves as the secretariat for the board
- The Board operates broadly in the following areas:
  - Policies & programmes

- Development of industries in specific region like Northeast
- Ancillary development, quality improvement, marketing assistance.
- Credit facilities, taxation and modernization
- Industrial sickness

Khadi and Village Industries Commission (KVIC)

- Statutory body created by an act of Parliament.
- It is charged with planning, promotion, organization and implementation of the program for the development of Khadi and other village industries in the rural areas in coordination with other agencies engaged in rural development.
- KVIC's functions also comprise building up a reserve of raw materials and implements for supply to producers, creation of common service facilities for processing of raw materials and provision of marketing of KVIC products.
- KVIC is entrusted with the task of providing financial assistance to institutions or persons engaged in the development and operation of Khadi and village industries and guide them through supply of designs, prototypes and other technical information.

Small Industries Development Organization (SIDO)

- Established in 1954 on recommendation of Ford Foundation
- Over the years, it had seen its role evolve into an agency for advocacy, handholding and facilitation for the small industries sector.
- SIDO provides facilities for testing, tool mending, training for entrepreneurship development, preparation of project and product profiles, technical and managerial consultancy, assistance for export, pollution and energy audits, and so on.
- SIDO provides economic information services and advises the government in policy formulation for the promotion and development of SSIs.

National Small Industries Corporation Ltd. (NSIC)

- Established in 1955 by GOI with the main objectives to promote, aid and foster the growth of SSIs in the country
- Over four decades of transition and growth in the SSI sector, NSIC has provided strength through a progressive attitude of modernization, up gradation of technology, quality consciousness, strengthening linkages with large and medium-scale enterprise and boosting exports of products from small enterprises
- Main services provided by NSIC are:
  - Machinery and Equipment (Hire Purchase / Lease scheme)
  - Financial Assistance Scheme
  - Assistance for Procurement of Raw Material
  - Government Store Purchase Program
  - Technology Transfer Centre (TTC)
  - Marketing Assistance

National Science and Technology Entrepreneurship Development Board (NSTEDB)

- Established in 1982 by GOI, is an institutional mechanism to help promote knowledge-driven and technology-intensive enterprises
- Major objectives are:

- promote and develop high-end entrepreneurship for S&T manpower as well as self-employment by utilizing S&T infrastructure and by using S&T methods
- facilitate and conduct various informational services relating to promotion of entrepreneurship
- network agencies of support system, academic institutions and R&D organizations to foster self-employment using S&T with special focus on backward areas
- act as a policy advisory body with regard to entrepreneurship

National Productivity Council (NPC)

- Autonomous institution functioning under the overall supervision of the Ministry of Industry, GOI
- Primary objective is to act as a catalyst in enhancing the productivity of all sectors of the economy, including industry and agriculture
- Administered by a tripartite Governing Council (GC) which has equal representation from the government, industry and trade unions
- Active in the field of consultancy and training and has a number of specialized divisions to provide tailor-made solutions to agriculture and industry. These divisions, manned by trained consultants, deal with issues related to industrial engineering, plant engineering, energy management, HRD, informal sector, agriculture and so on
- NPC is a member of the Asian Productivity Organization (APO), Tokyo, an umbrella body of all productivity councils in Asian region
- To channelise expertise of NPC to small-scale and informal sector, SIDBI has tied-up with NPC for enhancing technology in small units

National Institute for Small Industry Extension and Training (NISIET)

- Set up in early 1950s, NISIET acts an important resource and information centre for small units and undertakes research and consultancy for small industry development
- An autonomous arm of the Ministry of Small Scale Industries, the institute achieves its objectives through training, consultancy, research and education, to extension and information services
- In 1984, UNIDO has recognized NISIET as an institute of meritorious performance under its Centre of Excellence Scheme to extend aid

National Institute for Entrepreneurship and Small Business Development (NIESBUD)

- NIESBUD is an autonomous body under the administrative control of the Office of the DC(SSI)
- NIESBUD established in 1983 by the Ministry of Industry, GOI, as an apex body for coordinating and overseeing the activities of various institutions/agencies engaged in Entrepreneurship Development particularly in the area of small industry and business
- The policy, direction and guidance to the institute is provided by its Governing Council whose chairman is the Minister of SSI.
- Besides conducting national and international training programs, the institute undertakes research studies, consultancy assignments, development of training aids, etc.

State Level Institutions – DIs and DICs

- Directorate of Industries (DIs) – At the State level, the Commissioner/ Director of Industries implements policies for the promotion and development of small-scale, cottage, medium and

large scale industries. The Central policies for the SSI sector serve as guidelines but each State evolves its own policy and package of incentives. The Commissioner/ Director of Industries in all the States/UTs, oversee the activities of field offices, that is, the District Industries Centers (DICs) at the district level

- District Industries Centers (DICs) – In order to extend promotion of small-scale and cottage industries beyond big cities and state capitals to district headquarters, DIC program was initiated in May, 1978, as a centrally sponsored scheme. DIC was established with the aim of generating greater employment opportunities especially in rural and backward areas in the country. At present DICs operate under respective Sate budgetary provisions. DICs extend services of the following nature – (i) economic investigation of local resources (ii) supply of machinery and equipment (iii) provision of raw materials (iv) arrangement of credit facilities (v) marketing (vi) quality inputs (vii) consultancy

State Level Institutions – SFCs

State Financial Corporations (SFCs) – Main objectives are to finance and promote small and medium enterprises in their respective states for achieving balanced regional growth, catalyze investment, generate employment and widen ownership base of industry. Financial assistance is provided by way of term loans, direct subscription to equity/debentures, guarantees, discounting of bills of exchange and seed capital assistance. SFCs operate a number of schemes of refinance of IDBI and SIDBI and also extend equity type assistance. SFCs have tailor-made schemes for artisans and special target groups such as SC/ST, women, ex-servicemen, physically challenged and also provide financial assistance for small road transport operators, hotels, tourism-related activities, hospitals and so on. Under Single Window Scheme of SIDBI, SFCs have also been extending working capital along with term loans to mitigate the difficulties faced by SSIs in obtaining working capital limits on time.

State Level Institutions – SIDC / SIIC and SSIDC

- State Industrial Development / Investment Corporation (SIDC/SIIC) – Set up under the Companies Act, 1956, as wholly owned undertakings of the State governments, act as catalysts in respective states. SIDC helps in developing land providing developed plots together with facilities like roads, power, water supply, drainage and other amenities. They also extend assistance to small-scale sector by way of term loans, subscription to equity and promotional services. 11 out of 28 SIDCs in the country also function as SFCs and are termed as Twin-function IDCs
- State Small Industrial Development Corporations (SSIDC) – Established under Companies Act, 1956, as State government undertaking, caters to small, tiny and village industries in respective states. Being operationally flexible undertakes the activities like (i) procure and distribution of scarce raw materials, (ii) supply of machinery to SSI units on hire-purchase basis, (iii) product marketing assistance, (iv) construction of industrial estates, allied infrastructure facilities and their maintenance (v) extending seed capital assistance on behalf of State government and (vi) providing management assistance to production units.

Other State-level agencies Extending Facilities for SSI Promotion

- State Infrastructure Development Corporations
- State Cooperative Banks

- Regional Rural Banks
- State Export Corporations
- Agro Industries Corporations
- Handloom and Handicrafts Corporations

Other Agencies

National Bank for Agriculture and Rural Development (NABARD)	Set up in 1982, provide refinance assistance to State Cooperative Banks, Regional Rural Banks, and other approved institutions for all kinds of production and investment credit to SSIs, artisans, cottage and village industries, handicrafts and other allied activities. Helps SSI entrepreneurs to get loan for setting up SSIs in any part of the country
Housing and Urban Development Corporation Ltd. (HUDCO)	Wholly owned company of GOI, incorporated Apr.1970, as a Pvt. Ltd. Co. and subsequently, converted into a Public Ltd. Co. in 1986. Primary objective is to provide assistance for urban, social sector infrastructure, and the creation of housing facility, of late, to create SSI infrastructure. Also extends assistance for the promotion of building material industries, besides imparting consultancy, training and technical in related matters.
Technical Consultancy Organizations (TCOs)	Set up by all-India financial institutions during 70s and 80s to cater to consultancy needs of SMEs and new entrepreneurs. Services include preparing project profiles and feasibility studies, undertaking industrial potential surveys, identifying potential entrepreneurs and provision of technical and management assistance to them, undertake market research and surveys for specific products, carrying out energy audit and energy conservatism assignment, project supervision, taking up assignments on a turnkey basis, undertaking export consultancy for EOU

MODULE- 5

Project Management

Project

- An entrepreneur takes numerous decisions to convert his business idea into a running concern. His/ Her decision making process starts with project/product selection. The project selection is the first corner stone to be laid down in setting up an enterprise.
- The success or failure of an enterprise largely depends upon the project.
- A project has typically has a distinct mission that it is designed to achieve and clear termination point, the achievement of the mission.
- As a whole complex of activities involved in using resources to gain benefits.
- Project can differ in their size, nature of objectives, time duration and complexity.

A Project: Search for a Business idea

Introduction

- Industrialization is widely recognized not only as one of the important means to users in socio-economic transformation and achieving self-sufficiency but also for the accelerated development of agriculture, transport, trade, service and other potential sectors through forward and backward linkages.
- It is a process which accelerates economic growth.
- Real progress must ultimately depend on industrialization.

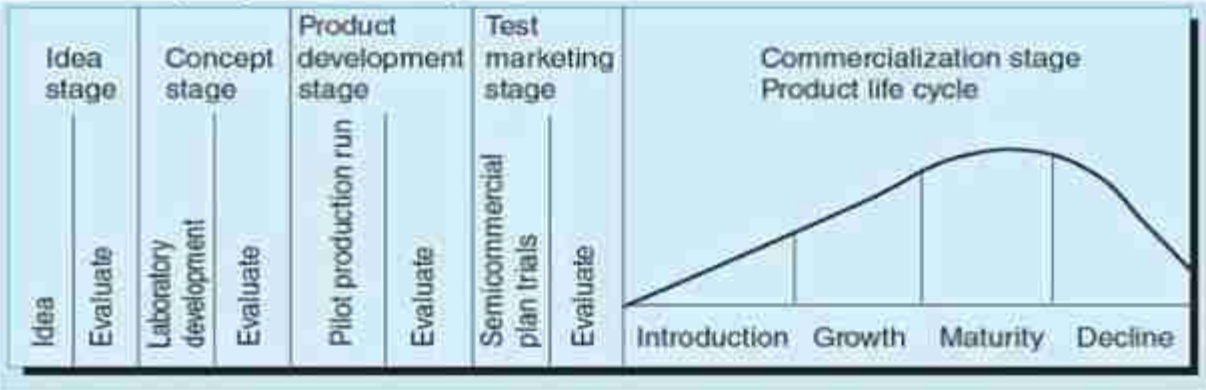
Choosing an Idea

- Establishing as a successful entrepreneur depends, in part, upon choosing a good idea.
- Idea must not only be good for the market, but good for the project and good for the entrepreneurs.
- It should be manageable by you without much dependence on others.
- The idea should give satisfaction results.
- As an entrepreneur develop more idea in parallel until one emerges so appropriate that it begins to dominate.
- To adopt one idea at a time had some disadvantage because constantly receiving random information from what we read and from people talk to. Also if pursuing a single idea by commitment then we may put our self in tight corner.
- Choosing an idea is difficult and the entrepreneur has to weigh objectively his intrinsic capabilities in finalizing an idea.
- In the idea stage, suggestions for new products are obtained from all possible sources: customers, competitors, R&D, distributors and company employees.
- Established objectives and defined areas provide a base for development.

Selection of product

- At this stage, the entrepreneur is concerned with identifying a particular product that hopes to market successfully at a reasonable profit.
- The selection of the right product is very essential for being successful in the business.
- The product should be marketed at a reasonable profit for the business growth.
- Various factors influence the entrepreneur in selecting the right product are:

- Whether import restrictions or the items selected are banned items would considerably weigh favorably or otherwise in the selection of the products.
- If the entrepreneur himself or his partners have gathered, substantial amount of experience in the manufacture and marketing of certain products , then selection of such product is advantage.
- The selection of product will also be based upon the degree of profitability that generally rules in the market.
- Many concessions are available from the government for producing a product which serves as an import substitute or even essential item.
- May products belongs to the priority industries or small-scale sector.
- The market for the product also plays a significant role in the selection of the product.
- Certain products are permitted for production only if the license is obtained from the appropriate authority while others belongs to the de-license category.
- Many products enjoy specific advantages in regard to the scale of manufacture or carry location advantage.
- If a product belongs to an ancillary unit and serves as a major component for the parent industry, it provides a ready demand.



The Adoption process

- An adoption process is a process bringing about a change in a buyer's attitude and perception.
- Customer adoption process covers the steps that a consumer usually goes through in determining the feasibility of buying new products:
  - Awareness: A person learns about a new idea, product or practice. He has general information about it. He has however, limited knowledge about special qualities, usefulness, performance, etc.
  - Interest: He now develops an interest in the innovation. He demands more detailed information about the new product, its utility, its performance, and so on. He listens to radio, TV ads, reads press ads, and learns more about it from others.
  - Evaluation: The accumulated information and evidences are weighed by the person in order to assess the basic soundness or worth of the innovation. He tries to weight the value of the new product and the extent to which it is good for him.
  - Trial: The person now is ready to put the idea into practice. Competent personal assistance is necessary to put the innovation to use.

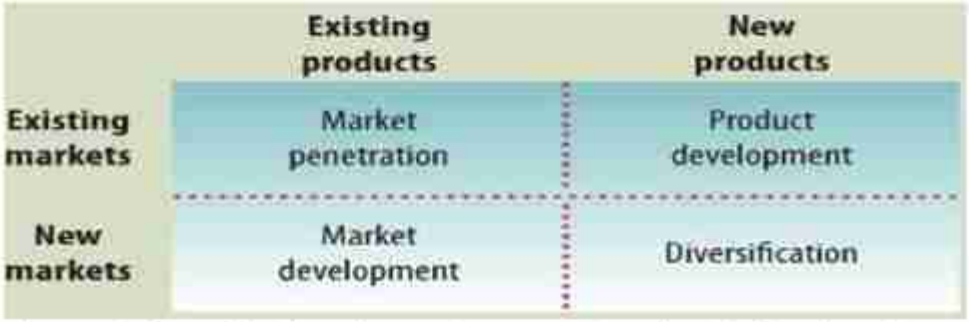
- Adoption: It is the final stage in which he makes a decision to buy. The person now decides to adopt the new idea, product or practice for continued use. If post purchase experience is good, he becomes a repeat buyer and a talking advertisement of the innovation.

Product Innovation

- Business enterprise must have two basic functions: marketing and innovation
- Innovation alone assures growth and survival while customers orientation assures survival. The evolution of new products is a practical business function and it is described as a process product management.
- The process of product planning and development is always adopted for product innovation.

Product Planning and Development Strategy

- Marketing have four alternative ways of bringing about an increase in sales and profit:
- Market penetration: It involves the expansion of sales of the existing products in the existing markets by selling more to present customers or gaining new customers in the existing markets. The firm can market its present products to existing markets. May have a temporary price cut to raise the volume of sales and penetrate the market in a big way. Penetrating the current market for higher usage rate is a conservative choice.



- Market development: In market development, a present product is introduced to a new market or segment. Market development is the creation of a new market by discovering new applications for existing goods. This is another alternative to expand market opportunity, prolong product life cycles, profitability and survival.
- Product development: Product development occurs when a firm introduces new products into market in which it is well established. Product development is the introduction of new products in the present market. The firms by offering new or improved products to present markets can satisfy the present customers better and stand assured of their loyalty.
- Product diversification: Diversification occurs when a firm seeks to enter a new market with a completely new product. Such a firm has neither market enterprise nor product knowledge. The firm may adopt a daring strategy by creating new products for entirely new markets. The innovations are introduced for the first time in the new market.

Product Planning and Development Process

- There are seven steps in planning and development of a new product.
- New Product Ideas: We visualize the detailed features of a model product. Ideas may be contributed by scientists, professional designers, rivals, customers. Sales force, top management, dealers, etc. We may need many ideas to get one commercially viable product.

- Idea Screening: We have to evaluate all ideas and inventions. Poor or bad ideas are dropped and through the process of elimination, only the most promising and profitable ideas are picked up for further detailed investigation and research.
- Concept Development and Testing: All ideas that survive the process of screening will be studied in detail. They will be developed into mature product concepts. We will have a precise description of the ideas and features of the proposed product. We can incorporate customer preference also.
- Business Analysis: Once the best product concept is picked up, it will be subjected to rigorous scrutiny to evaluate its market potential, capital investment, rate of return on capital, etc. Business analysis is a combination of marketing research, cost benefit analysis and assessment of competition. It eill prove the economic prospects of the new product concept.
- Product Development Programme: We have three steps in this stage when a paper idea is duly converted into a physical product.
    - Prototype development, giving a visual image of the product
    - Consumer testing of the model or prototype
    - Branding, packaging and labeling.
- Test Marketing: The entire product marketing programme is tried out for the first time in a small number of well-selected test markets. Test marketing is necessary to find out the viability of a full marketing programme for national distribution. Customer reactions can be tested under normal market conditions.
- Commercialization: Once the test marketing gives the green signal for the product with or without expected modifications, the company can proceed to finalize all features of the product. Mass production will start and all distribution channels will be duly organized. The product is now born and will start the life cycle.

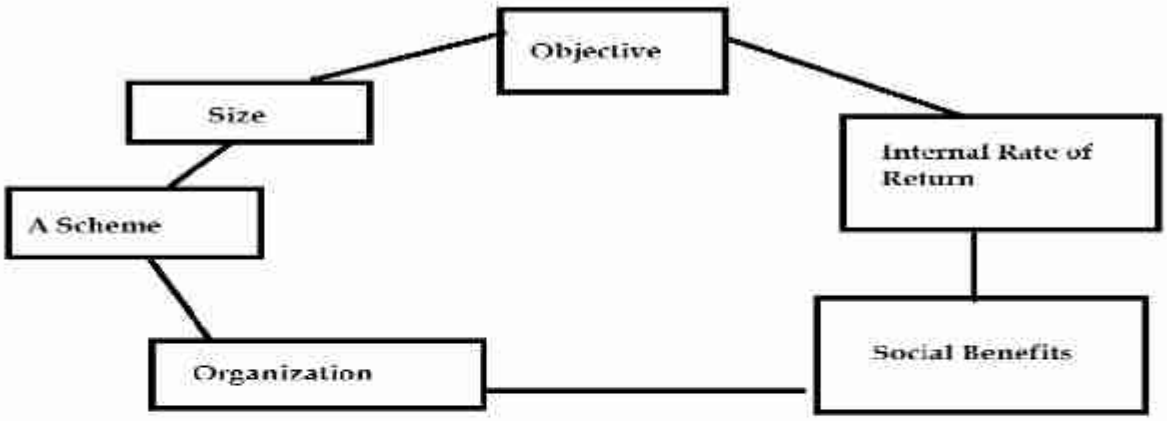
Concepts of Projects and Classification

Introduction :

- The project is an important groundwork of an enterprise and is also very crucial to the entrepreneur. Project is speculative imagination; a scheme of something to be done; a proposal for an undertaking. Two important aspects have to be borne in mind ie. A scheme and speculative imagination.

Meaning of Projects:

- The smallest unit of investment activity to be considered in the case of programming.
- It may be any item of investment activity which can separately be evaluated.
- The whole complex of activities involved in using resources to gain benefits.
- A project may be defined as a scientifically evolved work plan devised to achieve a specific objective with a specified period of time. The three basic attributes are a course of action, specific objective and defined time perspective.
- A project is a productive activity, which can be analyzed appraised and monitored independently.
- A project has specific objective in terms of a geographic location, specific starting and end point
- Serve the target population by achieving good returns on investment.
- Has an organization to implement it.



Basic Components of a project

Characteristics of a Project

- The four basic characteristics are:
  - Investment pattern
  - Benefits or gains
  - Time limit
  - Location
- The project is an economic activity with well-defined objectives and having a specific beginning and end.
- A well planned project includes a correct consideration of alternatives, identification of key issues, broad participation, compactness and enforceability. It should be neat, clear cut and specific.
- A project will involve allocation and consumption of resources, on the one hand and generation of resources, goods, or services, on the other.

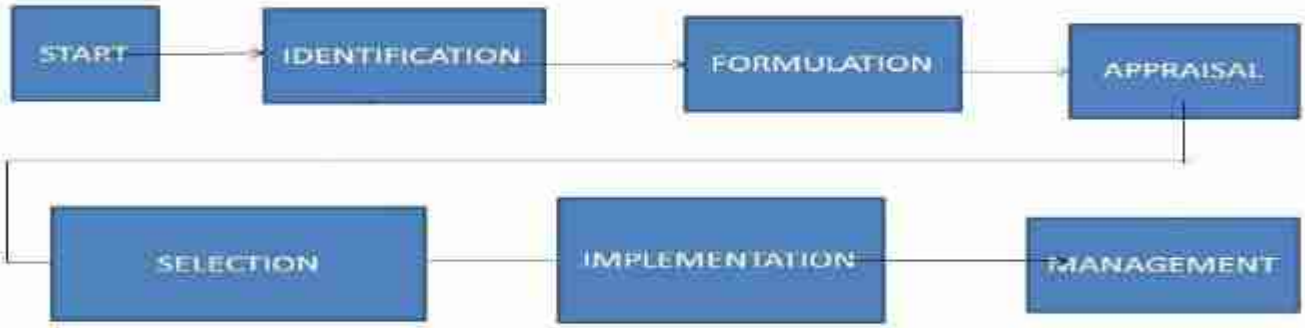
Project Levels

- The three levels are:
- At the national level, where national investment plans are formulated, priorities among sectors are established, and the macroeconomic framework of policies for economic growth is put in place.
- At the sector level, where priorities for investment within each sector are determined and the issues and problems affecting the development of the sector are addressed.
- At the project level, where individual projects are identified, prepared and implemented and attention is given to their technical, economical, financial, social, institutional and other dimensions.

Features and Phases of Project management

- Project Features
  - Simplicity and clarity
  - Availability of attractive technology to promote the project.
  - Integration of basic production services, especially those of input supply, credit, marketing and extension.
  - Compatibility of the project within the existing administrative mix.

- Phases of Project management
  - An entrepreneur has to consider carefully various factors from the start to the finish in converting profitable opportunities into realities.
  - The process of project management may be divided into six broad phases – identification, formulation, appraisal, selection, implementation and management of projects.



PHASES OF A PROJECT		
Sl.no.	**PHASE**	**REQUIREMENTS**
1	Identification	Selection of a project after a careful scanning of the environment of investment opportunity and its likely return
2	Formulation	Translation of the idea into a concrete project with scrutiny of its important preliminary aspects. Preparation of feasibility reports.
3	Appraisal	Searching scrutiny, analysis and evaluation of market technical, financial and economic variables. Assessing the profitability, return investment and break even points.
4	Selection	Rational choice of a project in the light of objectives and inherent constraints.
5	Implementation	Expeditious completion within the allocated resources
6	Management	Judicious operation of a project / enterprise with objectives like maximisation of net present value, maximisation of return and increase and increase in the rate of return of low risk.

Project Management Processes

- Operational Management Process
  - Planning
  - Executing
  - Controlling
- Additional management Process
  - Initiating
  - Closing
- Technical Processes

Project Identification

Feasibility Report

- Before starting a small-scale industry, it is mandatory for entrepreneurs to consult the Director of Industries Service Institute (SISI).
- The SISI guides entrepreneurs as to the type of industry to start, where to start and how to start it.
- The SISI help them odd select the various items of manufacture which have scope for development in different areas.
- It suggests the lines on which project report for the proposed units should be prepared for the consideration of various financial institution with a view to securing financial assistance.
- Similarly, technical guidance in the selection of proper raw materials and type of machinery is also provided.
- The SISI gives valuable information on various incentives available to the small-scale industries from various organizations.

Project Feasibility Analysis

- A project feasibility analysis includes market analysis, technical analysis, financial analysis and social profitability analysis.
- The starting point of a project analysis is the establishment of objectives to be attained. The next stage is the pre – selection stage – the advisability of having an in depth study. The analysis stage consists mainly of three factors – market, technical and financial analysis. A market analysis is a method of screening project ideas as well as means of evaluating a project's feasibility in terms of the market. A market analysis should cover the following areas:
    - A brief market description including the market are, methods of transportation, existing rates of transport, channels of distribution and general trade practices.
    - An analysis of past and present demand, determination of quantity value of consumption and identification of the major consumers of the product.
    - An analysis of past and present supply, broken down as source (whether imported or domestic) as well as information to assist in determining the competitive position of the product such as selling process quality and marketing practices of competitors.
- The technical analysis of a project feasibility study establishes whether project is technically feasible or not, and whether if offers basis for the estimation of costs. Moreover it provides an opportunity for a consideration of the effect of various technical alternatives on employment, ecology, infrastructure demands, capital services, support of other industries, balance of payments and other factors. A technical analysis should review the techniques or processes to be applied and should incorporate.
    - A description of the product including specification relating to its physical, mechanical and chemical properties, as well as the uses of the product.
    - A description of the selected manufacturing process, showing detailed flow charts and presenting alternative processes which may have been considered and the justification for the adoption of the selected process.
    - A determination of the plant size and production schedule which includes the expected volume for a given time period on the basis of start up and technical factors.
    - Selection of machinery and equipment including specifications, equipment to be purchased and its origin, quotations from suppliers, delivery dates, terms of payment and

a comparative analysis of alternatives in terms of cost reliability performance and spare parts availability.

- An identification of plant's location and as assessment of its desirability in terms of its distance from raw material sources and markets. For a new project this part may include a comparative study of different sites, indicating the advantages and disadvantages of each.
- A design of the plant layout and an estimate of the cost of the erection of the proposed buildings and land improvements.
- A study of the availability of raw materials and utilizes including a description of physical and chemical properties, quantities needed, current and prospective costs, terms of payment, location of sources of supply and continuity of supply.
- An estimate of labour requirements, including a detailed break – down of direct and indirect labour requirements and the supervision required for the manufacture of the product.
- A determination of the type and quantity of waste to be disposed of together with a description of the waste disposal method, its costs and the necessary clearance from proper authorities.
- An estimate of the production cost of the product.

- In the financial analysis of this feasibility study, the emphasis is on the preparation of financial statements, so that the project may be evaluated in terms of the different measures of commercial profitability followed by the magnitude of financing which requires the assembly of the market and also technical cost estimated in various proforma statements. If it is necessary to have more information on which to base an investment decision a sensitivity analysis or possibly a risk analysis may be conducted. This financial analysis should include:
    - For projects that involve new companies, statements of total project cost, for initial capital requirements and cash flows relative to the project schedule. For all projects financial projections for future time periods including income statements, cash flows and balance sheets.
    - For all projects supporting schedules for financial projection, stating the assumption made as to the collection period of sales, inventory levels, payment period of purchases and expenses and the element of production cost, selling, administrative and financial expenses.
    - For all projects a financial analysis showing returns on investments returns on equity , break even volume and price analysis.
    - For all projects, if necessary a sensitivity analysis to identify which have a substantial impact on profitability or possibly a risk analysis.
- For the small entrepreneur, the studies conducted during the analysis stage of the project provide the material for an assessment. If positive results are obtained, the entrepreneur in seeking finance, will want to prepare an investment proposal. The planners or government officials, however having obtained positive conclusions from the economic feasibility study will want to evaluate the element of social profitability.
- The purpose of the investment or loan application is to convince a lender (financial institution) that the project is a desirable investment; that it not only possesses the potential for profit but

that the proposed management team has the capability to achieve the potential. The investment proposal normally contains:

- General information on the product, company history, the nature of the industry and the reputation and qualifications of the existing or proposed management.
- A description of the project, which usually consists of extracts from economic feasibility selected manufacturing methods (with detailed indication of the cost of equipment and operational expenses) and a financial statement.
- Miscellaneous information such as the steps taken for the implementation of the project and the qualifications of the technical partners envisaged or selected.

Project Formulation

- **Meaning**
- Project formulation is defined as taking a first a look carefully and critically at a project idea by an entrepreneur to build up an all round beneficial to project after carefully weighing its various components.
- It is formulated by the entrepreneur with the assistance of specialists or consultants.
- The aim of project formulation is to achieve the project objectives with the minimum expenditure and adequate resources.
- In other words it is to derive maximum benefits from minimum expenses in a short span of time.



Fig: Phases of Project Formulation

Steps in Project formulation

- The objectives may be social, economic, or a combination of both and they can be defined under the following categories:
  - General objectives
  - Operational objectives
- A general objective merely states in broad terms the achievements expected whereas an operational objective specifically mentions results expected from the implementation of the project or scheme.
- The definition of objective in clear terms helps in quantifying physical, financial, human and other resources requirements.

Sequential Stages of Project Formulation

- The process of project development has been categorized into seven distinct and sequential stages. They are:
- **Feasibility analysis:** This is the very first stage in project formulation. At this stage the project idea is examined from the point of view of whether to go in for a detailed investment proposal or not. As project for a detailed investment proposal or not. As project idea is examined in the context of internal and external constraints three alternatives could be considered. First the project idea seems to be feasible, second the project idea is not a feasible one and third, unable

to arrive at a conclusion for want of adequate data. If it is feasible we proceed to the second step if not feasible we abandon the idea and if sufficient data are not available, we make more efforts to collect the required data and design development.

- **Techno – economic analysis:** In this step estimation of project demand potential and choice of optimal technology is made.
- **Project Design and Network analysis:** This important step defines individual activities which constitute the project and their inter – relationship with each other. The sequence of events of the project is presented. A detailed work plan of the project is prepared with time allocation for each activity and presented in a network drawing. Project design is the heart of the project entity. This paves the way for detailed identification and qualification of the project inputs, an essential step in the development of the financial and cost – benefit profile of the project.
- **Input analysis:** The step process the input requirements during the construction of the project and also during the operation of the project. In the earlier step a project was divided into several activities. Now it is better to see the inputs required for each activity and sum it up to get the total input requirements on qualitative and quantitative terms. Inputs include materials, human resources.
- **Financial analysis:** This stage mainly involves estimating the project costs, estimating its operating costs and fund requirements. Financial analysis also help in comparing various project proposals on a common scale, thereby aiding the decision maker. Some of the analytical tools used in financial analysis are discounted cash flow, cost – volume – profit relationship and ratio analysis.
- **Social cost benefit analysis:** When we talk of cost benefit analysis we not only take in to account the apparent direct costs and direct benefits of the project but also the costs which all entities connected with the project have to bear and the benefits which will be enjoyed by all such entities.
- **Pre – investment analysis:** The project proposal gets a formal and final shape at this stage. All the results obtained in the above steps are consolidated and various conclusions arrived at to present a clear picture. At his stage, the project is presented in such a way that the project – sponsoring body, the project – implementing body and the external consulting agencies are able to decide whether to accept the proposal or not.

Project Evaluation

- With a view to evaluate the project, the social cost-benefit analysis is used.
- All cost and benefits can be classified into three categories as:
  - Primary costs and benefits
  - Secondary costs and benefits
  - Tertiary costs and benefits.

Project Design and Network Analysis:

Introduction

- The execution of a project follows a definite path of planning, scheduling and controlling. The first and the foremost aspect of a project is the project design.
- It defines the individual activities which go into the corpus of the project and their interrelationship with each other.

- Project design enables to identify the flow of event which must take place for the successful implementation of the project.
- Network techniques help the management of an organization in performing these functions efficiently and effectively.



Fig: Relation between Project Design and Network

Importance of Network Analysis

- The network analysis has the potential to unfolding unknown snags involved in project estimates which, when detected, may provide management not only to improve on the ongoing project estimates but also to take serious lesson for future applications.
    - The whole project should be considered with reference to the sequence of activities and events.
    - This would also require that the events should be thought of in different streams of operations and their relationship understood clearly.
    - The whole project may be put on one network while different segments of the project may be detailed out in separate networks for final integration in the overall network.
    - The time estimates may be made taking into view two discrete aspects: one projects in which previous experience does not exist and two, time estimates may be deterministic.
    - Cost estimates would depend on the project time estimates and the changes in the prices of different factors of production.
    - The physical progress of the projects, individuality and simultaneity of events, jobs farmed out snags in different areas of project work would all require adequate notice and application of correctives in proper time.

Origin of PERT and CPM

- The Programme Evaluation and Review Techniques (PERT) and Critical Path Method (CPM) techniques were developed in U.S. independently, while CPM came into focus about 1957 as an offshoot of collaboration between Du Pont and Remington Rand.
- The emphasis was on the trade-off between the cost of the project and its overall completion time.
- PERT, which was developed about 1958 as a result of collaboration between the Operational Research Division of the United States Navy and a firm of business consultants.
- The emphasis was on completing the program in the shortest possible time. In addition PERT had the ability to cope with uncertain activity completion times.

Network

- A network comprises a set of exponents connected with each other in a sequential relationship with each step till the completion of a project.
- Network analysis is a system which plans both large and small projects by analyzing the project activities.
- Projects are broken down into simple activities, which are then arranged in a logical sequence.
- It is also decided as to which task will be performed simultaneously and which other sequentially.



Fig: Network Design

Network Techniques

- Several techniques of project scheduling and control such as Bar charts, Programme Evaluation and Review Techniques (PERT) and Critical path method (CPM) are used.
- Of these CPM have some to be widely used in project management as they are very useful in the basic management functions of planning, scheduling and control.
- These techniques can be applied in diverse kinds of projects like construction of a projects, scheduling ship construction and repairs, end of the month closing of accounts, large, research projects etc.

Need for Network Techniques

- Network analysis helps in designing, planning, coordinating, controlling and decision-making in order to accomplish the project economically in the minimum available time with the limited available resources. The conventional planning method like bar chart had limitations the following disadvantages are:
    - A bar chart becomes too cumbersome while dealing with big and complex projects.
    - A bar chart does not point out which tasks should be given priority as regards resources.
    - The effect of changes in schedule cannot be evaluated with the help of a bar chart.
    - A bar chart neither satisfactorily tells the time at which the activates begin and end nor does it indicate tolerances in activity timing.
- PERT is one of the management techniques which is considerably more useful.
- It works a method of minimizing production delays, interruptions, and conflicts; of co-ordinating and synchronizing the various parts of the overall job; and of expediting of projects.
- PERT is concerned with two concepts:
    - **Events:** An event is a specific accomplishment that occurs at a recognizable point of time and does not call for either the need of time or resources.
    - **Activities:** An activity is the work required to complete a specific event.

Steps in PERT

- The first step in the development of a PERT network is the establishment of objectives.
- The second step is to schedule work breakdown in great detail.

- In the third step both technical and managerial persons should begin to work together.
- The fourth step is that each person who participants in the application of PERT to the control of the project should have some basic familiarity with the general nature of the work and with the ultimate objective desired.
- In PERT, time is the basic measure. It is usually expressed in calendar weeks the project should be completed within stipulated optimistic time.
- In order to arrive at the most reliable estimate of time, three time estimates are usually employed under this technique as given below:
    - **The optimistic time:** It is the shortest time possible if everything goes perfectly well with no complications, the chance of this optimum actually occurring might be one in a hundred.
    - **The pessimistic time:** It is longest time conceivable, it includes time for unusual delays and thus the chance of its happening might be only one in a hundred;
    - **The most likely time:** It would be the best estimate of what normally would occur.
- The difference in these three times give a measure of the relative uncertainty involved in the activity.
- **ADVANTAGES OF PERT**
    - This technique gives the management the ability to plan the best possible use of resource to achieve a given goal within the overall time and cost limitations.
    - It helps management to handle the uncertainties involved in programmes where no standard time data of the Taylor – Gantt variety are available.
    - Use of this technique for active control of project requires frequent updating and revising the PERT calculations and this proves quite a costly affair.
- **LIMITATIONS OF PERT**
    - The basic difficulty comes in the way of time estimates for the completion of activities because activities are of non – repetitive type.
    - This technique does not consider resources required at various stages of the project.
    - Use of this technique for active control of a project requires updating and revising the PERT calculations and this proves quite a costly affair.
- **CRITICAL PATH METHODS (CPM)**
- CPM was developed in 1956 at the E.I. Dupont Nemours & Co. U.S.A in connection with the Periodic overhauling and maintenance of a chemical plant. It resulted in reducing the shut down period from 130 hours to 90 hours and saving hours and saving the company $ 1 million. CPM has two time cost estimates for each activity (one time cost estimates for the normal situation and the other estimate for the crash situation) but does not incorporate any statistical analysis in determining such time estimates. CPM operates on the assumptions that there is a precise known time that each activity in the project will take.
- **ADVANTAGES OF CPM**
    - It helps in ascertaining the time schedule.
    - With its aid, control by the management is made easy.
    - It makes better and detailed planning possible.

- It proves a standard method for communicating project plans, schedules time and cost performance.
- It identifies most critical elements and thus more attention can be paid to these activities.

- **LIMITATIONS OF CPM**
  - CPM fails to incorporate statistical analysis in determining the time estimates.
  - It operates on the assumption that there is a precise known time that each activity in the project will take but his may not be true in actual life.
  - It is difficult to use CPM as a controlling device for the simple reason that one must repeat the entire evaluation of the project each time when changes are introduced into the network. It may be remembered that CPM was initially developed as a static planning model and not as a dynamic controlling device .

Differences between PERT and CPM

PERT	CPM
The origin is military (Naval)	The origin is industrial.
It is an event-oriented approach.	It is an activity-oriented system.
There is allowance for uncertainty.	No such allowance.
It has three time estimates.	There is only one single estimate of time and the emphasis is on cost.
It is a probabilistic model with uncertainty in activity duration.	It is a deterministic model with well-known activity time based upon past experience.
It does not demarcate between critical and non-critical activities.	It marks critical activities
It is especially suitable when high prediction is required in time estimates.	It is suitable when reasonable prediction is required.
Time is averaged	No averaging of time is involved.
The concept of 'crashing' is not applied.	The concept of crashing is applied.
It is time - based.	It is cost - based.

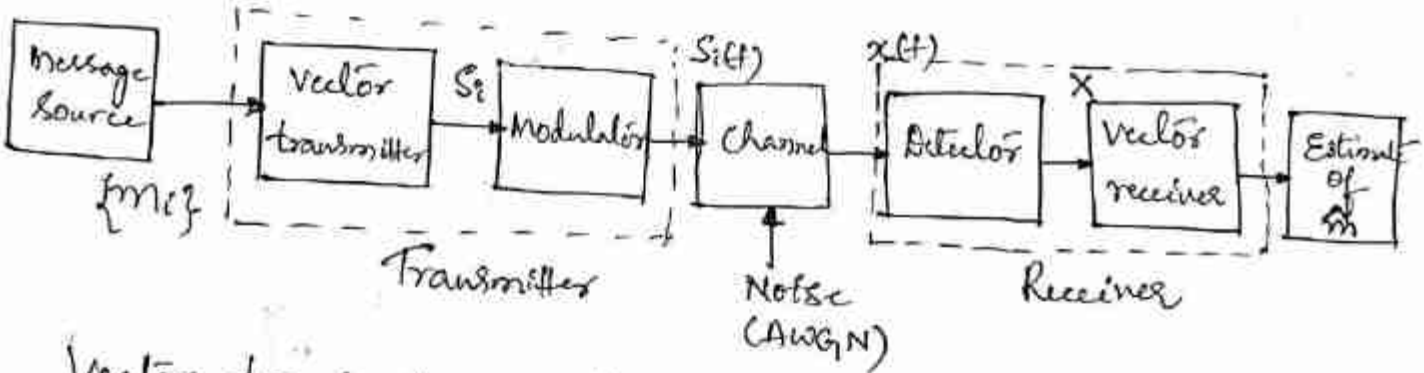
K. Prabhavathi
Asst Professor
Dept of ECE
BGSIT, BGNagar

Module - 2

Signaling Over AWGN Channels

Detection theory deals with the design and evaluation of a decision making receiver system, that observes the received signal corrupted by additive noise & guesses the most likely transmitted symbol. Since the effect of noise cannot be completely removed, the receiver is to be designed in such a way that the effect of noise is minimum.

* Model of a Digital Communication System :-



Vector transmitter produces a sequence S_i corresponding to each message m_i. Modulator converts each vector S_i into a continuous finite energy signal $S(t)$. The channel noise is assumed to be additive white gaussian noise (AWGN). At the receiver, the detector receives the signal corrupted by noise & processes the received signal to get an estimate $\hat{m}$ of message that was transmitted.

When the source symbol is m_i, the corresponding o/p of the vector transmitter is

Scanned with CamScanner

$$S_i = \begin{bmatrix} S_{i1} \\ S_{i2} \\ \vdots \\ S_{iN} \end{bmatrix}, \quad i = 1, 2, 3 \ldots M$$

where N is the dimension of the vector & $N \leq M$. The input to the modulator is S_i. The modulator then constructs a finite energy signal $S_i(t)$ of duration T seconds.

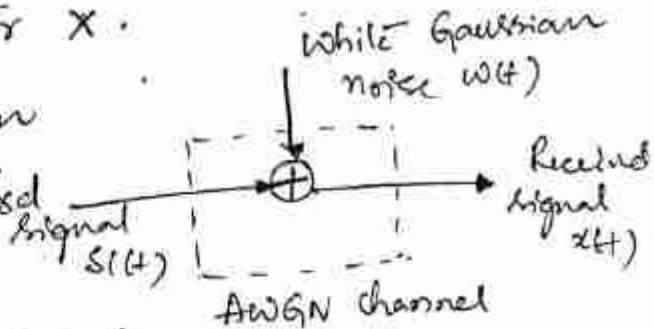
The finite energy of the signal $S_i(t)$ is given by

$$E_i = \int_0^T S_i^2(t)\, dt$$

If $S_i(t)$ is transmitted during symbol interval T seconds, then the received signal $x(t)$ is

$$x(t) = S_i(t) + W(t)$$

The receiver observes the received signal $x(t)$ & then generates an vector X. The receiver makes a decision regarding which symbol is transmitted by computing the estimate $\hat{m}$ of the rxd signal.



→ Geometric Representation of Signals :-

Geometric representation of signals is to represent set of M energy signals $\{S_i(t)\}$ as linear combination of N orthonormal basis functions, where $N \leq M$. i.e., For a set of real-valued energy signals, $S_1(t), S_2(t) \ldots S_M(t)$ each of duration T seconds.

Scanned with CamScanner

$$S_i(t) = \sum_{j=1}^{N} s_{ij}\,\phi_j(t) \quad \text{---(1)} \quad \begin{cases} 0 \le t \le T \\ i = 1, 2 \ldots M \end{cases}$$

The coefficients s_{ij} are given by,

$$s_{ij} = \int_0^T s_i(t)\,\phi_j(t)\,dt \quad \text{---(2)} \quad \begin{cases} i = 1, 2, 3 \ldots M \\ j = 1, 2, 3 \ldots N. \end{cases}$$

The basis functions $\phi_1(t), \phi_2(t) \ldots \phi_N(t)$ form an orthonormal set,

$$\int_0^T \phi_i(t)\,\phi_j(t)\,dt = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad \text{---(3)}$$

The first condition of above eqn(3) states that each basis function is normalized to have unit energy. The second condition states that the basis functions $\phi_1(t), \phi_2(t), \ldots \phi_N(t)$ are orthogonal to each other.

Figure below shows the scheme for generating the signal $s_i(t)$. It consists of a bank of N multipliers with each multiplier having its own basis function followed by a summer.

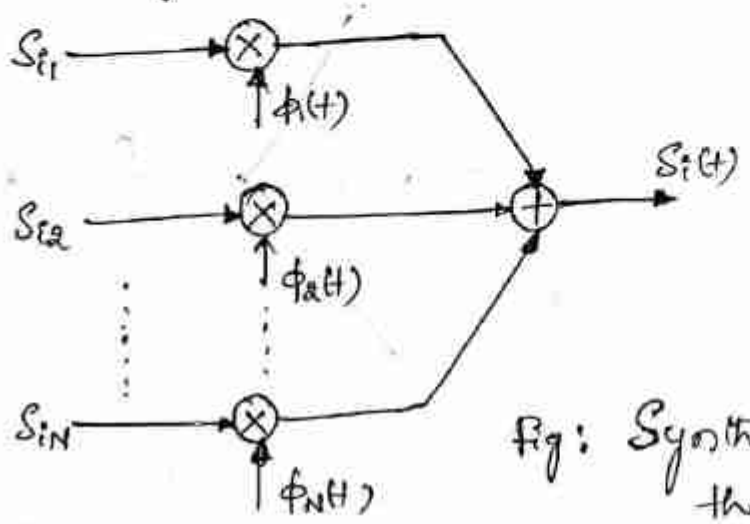


Fig: Synthesizer for generating the signal $s_i(t)$.

Scanned with CamScanner

Figure below shows the scheme for reconstructing the signal vector $\{S_i\}$. This scheme consists of a bank of N-correlators with a common input & with each one of them supplied with its own basis function.

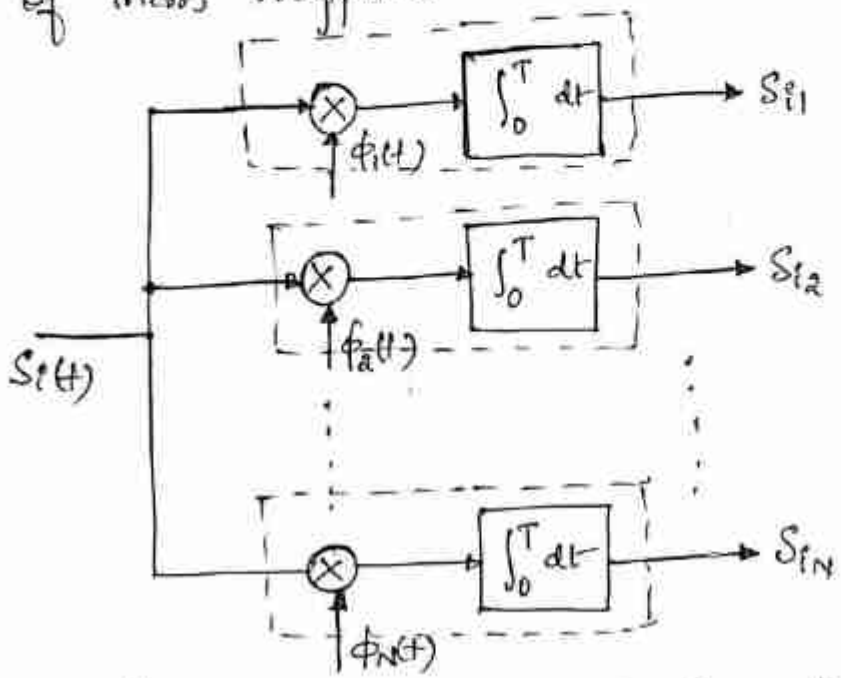


Fig:- Analyzer for reconstructing the signal vector $\{S_i\}$

Let $S_1 = \begin{bmatrix} S_{11} \\ S_{12} \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$ $S_2 = \begin{bmatrix} S_{21} \\ S_{22} \end{bmatrix} = \begin{bmatrix} -2 \\ 3 \end{bmatrix}$ $S_3 = \begin{bmatrix} S_{31} \\ S_{32} \end{bmatrix} = \begin{bmatrix} -3 \\ -3 \end{bmatrix}$

be the signal vectors

& $\phi_1(t)$ & $\phi_2(t)$ be the orthonormal basis funs.

i.e, $\boxed{N=2 \quad \& \quad M=3}$

$S_1(t) = S_{11}\phi_1(t) + S_{12}\phi_2(t)$

$\hookrightarrow = 3\phi_1(t)$ —①

$S_2(t) = S_{21}\phi_1(t) + S_{22}\phi_2(t)$

$\hookrightarrow = -2\phi_1(t) + 3\phi_2(t)$ —②

& $S_3(t) = -3\phi_1(t) - 3\phi_2(t)$ —③

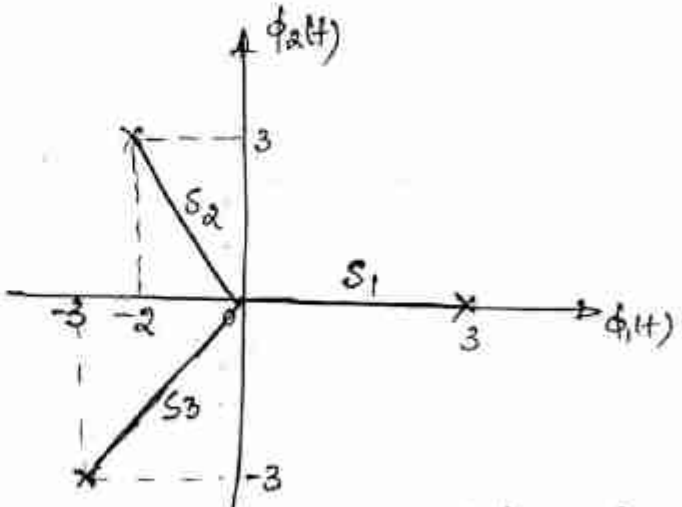


fig: Geometric representation of signals for N=2 & M=3.

Scanned with CamScanner

" The representation of signals in an N-dimensional signal space [Euclidian space] is called as signal constellation." The above figure illustrates the case of a two-dimensional signal space with three signals : i.e., N = 2 & M = 3.

→ **Relation between lengths of vector & signal energy :-**

For an N-dimensional Euclidian space, we define lengths of vectors & angles between vectors.

The length of a signal vector S_i is denoted by the symbol $\|S_i\|$. The squared length of any signal vector S_i is given by,

$$\|S_i\|^2 = S_i^T S_i$$

$$= \begin{bmatrix} S_{i1} & S_{i2} & \cdots & S_{iN} \end{bmatrix} \begin{bmatrix} S_{i1} \\ S_{i2} \\ \vdots \\ S_{iN} \end{bmatrix}$$

$$= S_{i1}^2 + S_{i2}^2 + \cdots S_{iN}^2$$

$$= \sum_{j=1}^{N} S_{ij}^2 \quad \text{where } i = 1, 2 \cdots M.$$

The energy E_i of a signal $S_i(t)$ is

$$E_i = \int_0^T S_i^2(t)\, dt$$

$$E_i = \int_0^T \left(\sum_{j=1}^{N} S_{ij}\, \phi_j(t) \right) \left(\sum_{k=1}^{N} S_{ik}\, \phi_k(t) \right) dt$$

Scanned with CamScanner

Interchanging the order of summation & integration

$$R_i = \sum_{j=1}^{N} \sum_{K=1}^{N} s_{ij}\, s_{ik} \int_0^T \phi_j(t)\, \phi_k(t)\, dt$$

When $j = K$

$$E_i = \sum_{j=1}^{N} s_{ij}^2 = \|s_i\|^2$$

Let s_i & s_j be two vectors in the set. Then cosine of the angle b/n two vectors s_i & s_j is defined as,

$$\cos\theta_{ij} = \frac{\text{Inner product of } s_i \text{ \& } s_j}{(\text{length of } s_i)\,(\text{length of } s_j)}$$

$$\cos\theta_{ij} = \frac{s_i^T s_j}{\|s_i\|\,\|s_j\|}$$

The two vectors s_i & s_j are orthogonal or perpendicular to each other if their inner product $s_i^T s_j$ is zero in which $\boxed{\theta_{ij} = 90^\circ}$.

Note :- A pair of signals $s_i(t)$ & $s_j(t)$ represented by the signal vectors s_i & s_j, then

$$s_i^T s_j = \int_0^T s_i(t)\, s_j(t)\, dt.$$

Scanned with CamScanner

* **The Schwarz Inequality :-**

Consider a pair of energy signals $s_1(t)$ & $s_2(t)$.

The Schwarz inequality states,

$$\left(\int_{-\infty}^{\infty} s_1(t)\, s_2(t)\, dt\right)^2 \leq \left(\int_{-\infty}^{\infty} s_1^2(t)\, dt\right)\left(\int_{-\infty}^{\infty} s_2^2(t)\, dt\right)$$

To prove this, let $s_1(t)$ & $s_2(t)$ be expressed in terms of the pair of orthonormal basis functions $\phi_1(t)$ & $\phi_2(t)$ as follows,

$$s_1(t) = s_{11}\,\phi_1(t) + s_{12}\,\phi_2(t)$$

$$s_2(t) = s_{21}\,\phi_1(t) + s_{22}\,\phi_2(t)$$

where, $$\int_{-\infty}^{\infty} \phi_i(t)\,\phi_j(t)\, dt = \begin{cases} 1 & \text{for } i = j \\ 0 & \text{for } i \neq j \end{cases}$$

Signal vectors, $s_1 = \begin{bmatrix} s_{11} \\ s_{12} \end{bmatrix}$ & $s_2 = \begin{bmatrix} s_{21} \\ s_{22} \end{bmatrix}$

$$\text{WKT, } \cos\theta = \frac{s_1^T s_2}{\|s_1\|\,\|s_2\|}$$

$$= \frac{\int_{-\infty}^{\infty} s_1(t)\, s_2(t)\, dt}{\left(\int_{-\infty}^{\infty} s_1^2(t)\, dt\right)^{1/2}\left(\int_{-\infty}^{\infty} s_2^2(t)\, dt\right)^{1/2}}$$

Recognizing that, $|\cos\theta| \leq 1$

Scanned with CamScanner

$$\int_{-\infty}^{\infty} s_1(t)\, s_2(t)\, dt \leq \left(\int_{-\infty}^{\infty} s_1^2(t)\, dt\right)^{1/2} \left(\int_{-\infty}^{\infty} s_2^2(t)\, dt\right)^{1/2}$$

Squaring

$$\left(\int_{-\infty}^{\infty} s_1(t)\, s_2(t)\, dt\right)^2 \leq \left(\int_{-\infty}^{\infty} s_1^2(t)\, dt\right)\left(\int_{-\infty}^{\infty} s_2^2(t)\, dt\right)$$

This inequality condition holds good if & only if $s_2(t) = C\, s_1(t)$, where C is a constant.

* Gram-Schmidt Orthogonalization procedure :—

** Gram-Schmidt procedure allows us to determine a finite set of N orthonormal functions such that each $s_i(t)$ can be expressed as a linear combinations of these N orthonormal functions.

Let $\phi_1(t), \phi_2(t) \ldots \phi_N(t)$ be the N orthonormal functions such that,

$$s_1(t) = s_{11}\phi_1(t) + s_{12}\phi_2(t) + \ldots + s_{1N}\phi_N(t) \quad \text{—①}$$

$$s_2(t) = s_{21}\phi_1(t) + s_{22}\phi_2(t) + \ldots + s_{2N}\phi_N(t) \quad \text{—②}$$

$$s_3(t) = s_{31}\phi_1(t) + s_{32}\phi_2(t) + \ldots + s_{3N}\phi_N(t) \quad \text{—③}$$

$$\vdots$$

$$s_M(t) = s_{M1}\phi_1(t) + s_{M2}\phi_2(t) + \ldots + s_{MN}\phi_N(t) \quad \text{—④}$$

Step 1 :— Set all the co-efficients except s_{11} to zero in eqn ①

Scanned with CamScanner

$$s_1(t) = s_{11}\,\phi_1(t)$$

$$\rightarrow \phi_1(t) = \frac{s_1(t)}{s_{11}} \quad \text{but } \phi_1(t) \text{ is normalized,}$$

$$\int_0^T \phi_1^2(t)\,dt = 1$$

$$\text{or} \quad \int_0^T \frac{s_1^2(t)}{s_{11}^2}\,dt = 1$$

$$\text{or} \quad \int_0^T s_1^2(t)\,dt = s_{11}^2$$

$$\text{or} \quad \boxed{s_{11} = \left[\int_0^T s_1^2(t)\,dt\right]^{1/2}}$$

Step 2:- Set all the coefficients except s_{21} & s_{22} to zero. in eqn (A)

$$s_2(t) = s_{21}\,\phi_1(t) + s_{22}\,\phi_2(t) \quad \text{---} \quad (*)$$

Multiply both side by $\phi_1(t)$

$$s_2(t)\,\phi_1(t) = s_{21}\,\phi_1(t)\,\phi_1(t) + s_{22}\,\phi_2(t)\,\phi_1(t)$$

Taking integration on both side over the interval 0 to T

$$\int_0^T s_2(t)\,\phi_1(t)\,dt = \int_0^T s_{21}\,\phi_1^2(t)\,dt + \int_0^T s_{22}\,\phi_2(t)\cdot\phi_1(t)\,dt$$

$$\int_0^T \phi_1^2(t)\,dt = 1 \quad \& \quad \int_0^T \phi_2(t)\,\phi_1(t)\,dt = 0$$

$$\Rightarrow \boxed{s_{21} = \int_0^T s_2(t)\,\phi_1(t)\,dt}$$

Scanned with CamScanner

from eqn (4)

$$S_{22}\,\phi_2(t) = s_2(t) - S_{21}\,\phi_1(t)$$

Squaring & integrating above eqn.

$$\int_0^T S_{22}^2\,\phi_2^2(t)\,dt = \int_0^T [s_2(t) - S_{21}\,\phi_1(t)]^2\,dt$$

$$\Rightarrow S_{22}^2 = \int_0^T [s_2(t) - S_{21}\,\phi_1(t)]^2\,dt \qquad \left\{ \because \int_0^T \phi_2^2(t)\,dt = 1 \right.$$

$$\therefore S_{22} = \left[\int_0^T [s_2(t) - S_{21}\,\phi_1(t)]^2\,dt \right]^{1/2}$$

Since S_{21} & S_{22} are known, $\phi_2(t)$ can be written as,

$$\boxed{\phi_2(t) = \frac{1}{S_{22}} [s_2(t) - S_{21}\,\phi_1(t)]}$$

Step 3 :- from eqn (3),

$$s_3(t) = S_{31}\,\phi_1(t) + S_{32}\,\phi_2(t) + S_{33}\,\phi_3(t)$$

by setting all coefficients to zero.

$$S_{31} = \int_0^T s_3(t)\,\phi_1(t)\,dt$$

$$S_{32} = \int_0^T s_3(t)\,\phi_2(t)\,dt$$

$$S_{33}^2 = \int_0^T [s_3(t) - S_{31}\,\phi_1(t) - S_{32}\,\phi_2(t)]^2\,dt$$

$$\therefore S_{33} = \left(\int_0^T [s_3(t) - S_{31}\,\phi_1(t) - S_{32}\,\phi_2(t)]^2\,dt \right)^{1/2}$$

Scanned with CamScanner

Since S_{31}, S_{32} & S_{33} are known,

$$\boxed{\phi_3(t) = \frac{1}{S_{33}}\left[S_3(t) - S_{31}\phi_1(t) - S_{32}\phi_2(t)\right]}$$

Step 4: This procedure continues till $\phi_N(t)$.

* Conversion of a Continuous AWGN channel into a Vector channel :-

The received signal $x(t)$ is given by,

$$x(t) = S_i(t) + w(t) \longrightarrow (1)$$

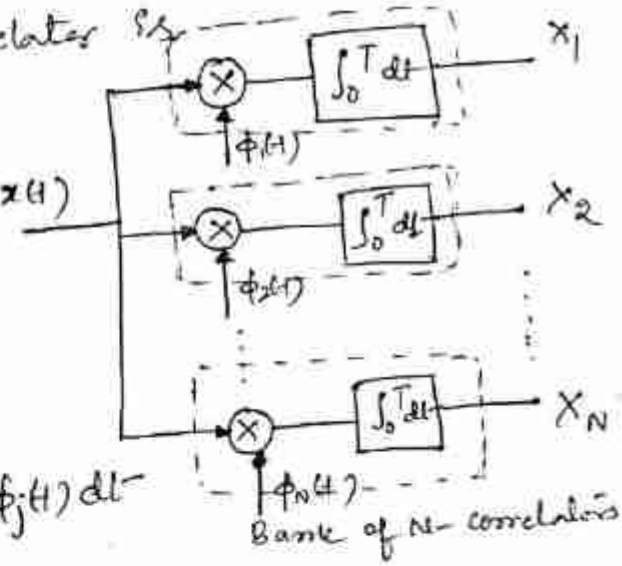
where $S_i(t)$ is the txd signal & $w(t)$ is a sample value of white Gaussian noise process.

The output of each correlator is,

$$x_j = \int_0^T x(t)\,\phi_j(t)\,dt$$

$$x_j = \int_0^T \left[S_i(t) + w(t)\right]\phi_j(t)\,dt$$

$$x_j = \int_0^T S_i(t)\,\phi_j(t)\,dt + \int_0^T w(t)\,\phi_j(t)\,dt$$



$$\boxed{x_j = S_{ij} + w_j} \longrightarrow (2) \text{ where } j = 1, 2, 3 \ldots N.$$

Consider a new stochastic process $x'(t)$ related to the received signal $x(t)$ as follows.

$$x'(t) = x(t) - \sum_{j=1}^{N} x_j\,\phi_j(t) \longrightarrow (3)$$

Scanned with CamScanner

$$x'(t) = s(t) + w(t) - \sum_{j=1}^{N} (s_{ij} + w_j)\phi_j(t)$$

$$x'(t) = \cancel{s_i(t)} + w(t) - \underbrace{\sum_{j=1}^{N} s_{ij}\phi_j(t)}_{s(t)} - \sum_{j=1}^{N} w_j \phi_j(t)$$

$$x'(t) = w(t) - \sum_{j=1}^{N} w_j \phi_j(t)$$

$$x'(t) = w'(t) \quad \text{——— (4)}$$

$$\therefore \quad x(t) = x'(t) + \sum_{j=1}^{N} x_j \phi_j(t) \quad \text{— (from (3) we can write)}$$

$$\text{or} \quad x(t) = \sum_{j=1}^{N} x_j \phi_j(t) + w'(t)$$

Statistical Characterization of the Correlator outputs

Since $w(t)$ is a gaussian random variable & is completely characterized by its mean value & variance

The mean of x_j is

$$\mu_{x_j} = E[x_j] = E[s_{ij} + w_j]$$

$$\hookrightarrow = s_{ij}$$

$\because$ the noise $w(t)$ has zero mean & variance $\frac{N_0}{2}$

The variance of x_j,

$$\sigma^2_{x_j} = E[w_j^2]$$

where $w_j = \int_0^T w(t)\phi_j(t)\,dt$

Scanned with CamScanner

$$\sigma_{x_j}^2 = E\left[\int_0^T w(t)\phi_j(t)\,dt \int_0^T w(u)\phi_j(u)\,du\right]$$

$$= E\left[\int_0^T\int_0^T \phi_j(t)\phi_j(u)\,w(t)\,w(u)\,dt\,du\right]$$

Interchanging the order of integration & expectation

$$\sigma_{x_j}^2 = \int_0^T\int_0^T \phi_j(t)\,\phi_j(u)\,E[w(t), w(u)]\,dt\,du$$

But, $E[w(t), w(u)] = R_w(t,u) = \frac{N_0}{2}\delta(t-u)$

↳ auto correlation function

$$\therefore \sigma_{x_j}^2 = \frac{N_0}{2}\int_0^T\int_0^T \phi_j(t)\,\phi_j(u)\,\delta(t-u)\,dt\,du.$$

Using shifting property,

$$\int_0^T \phi_j(u)\,\delta(t-u)\,du = \phi_j(t)$$

$$\therefore \sigma_{x_j}^2 = \frac{N_0}{2}\int_0^T \phi_j^2(t)\,dt$$

Since $\int_0^T \phi_j^2(t)\,dt = 1$ i.e, Unit energy.

$$\boxed{\sigma_{x_j}^2 = \frac{N_0}{2}}$$

Thus all the correlators o/ps $\{x_j\}$, $j = 1, 2, 3 \cdots N$ have mean values s_{ij} & variance $\frac{N_0}{2}$.

Also, $\{x_j\}$, $j = 1, 2, \cdots N$ are uncorrelated & statistically independent.

Scanned with CamScanner

The conditional probability density function $f_{x_j}(x_j|m_i)$ is

$$f_{x_j}(x_j|m_i) = \frac{1}{\sqrt{\pi N_0}} e^{\left[-\frac{1}{N_0}(x_j - s_{ij})^2\right]}$$

The likelihood functions of an AWGN channel are defined by,

$$f_x(X|m_i) = (\pi N_0)^{-\frac{N}{2}} e^{\left[-\frac{1}{N_0}\sum_{j=1}^{N}(x_j - s_{ij})^2\right]}$$

where $X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$ is the received vector

* Optimum Receivers using Coherent detection :-

*** Maximum likelihood decoding :-

Let X be the received vector & $\hat{m}$ be the estimate of the txd signal m_i. then the average probability of correct decision is given as the conditional probability, $P_c = P(m_i \text{ sent} | X \text{ received})$

The Average probability of Symbol error P_e is given by,

$$P_e = 1 - P_c$$

$$P_e = 1 - P(m_i \text{ sent} | X \text{ received})$$

The decision rule is such that P_e is to be minimized.

The likelihood function of the received vector X given that m_i was transmitted. is $f_x(X|m_i)$

Scanned with CamScanner

The decision rule is to choose estimate $\hat{m} = m_i$ if $f_x(x|m_k)$ is maximum for $k=i$

This decision rule is referred as maximum likelihood decision & the corresponding receiver structure is called maximum likelihood receiver.

The likelihood fun for AWGN channel is,

$$f_x(x|m_k) = (\pi N_0)^{-N/2} e\left[-\frac{1}{N_0}\sum_{j=1}^{N}(x_j - s_{kj})^2\right] \quad \text{---} (1)$$

$k = 1, 2, 3 \cdots m$

Taking natural log on both side,

$$\ln\left[f_x(x|m_k)\right] = -\frac{N}{2}\ln(\pi N_0) - \frac{1}{N_0}\sum_{j=1}^{N}(x_j - s_{kj})^2 \quad \text{---} (2)$$

Since logarithmic fun is a monotonically increasing function. The decision rule is now becomes

$\ln\left[f_x(x|m_k)\right]$ is maximum for $k=i$

In eqn (2), first term in the right hand side is constant. Log-likelihood function is maximum only if the magnitude of the summation is minimum i.e.,

eqn (2), $\sum_{j=1}^{N}(x_j - s_{kj})^2$ is minimum for $k=i$

Note:- As 1st term in the right hand side is constant (it does not depend on k) only and term is considered for analysis.

Scanned with CamScanner

The decision rule can be simplified by expanding the terms

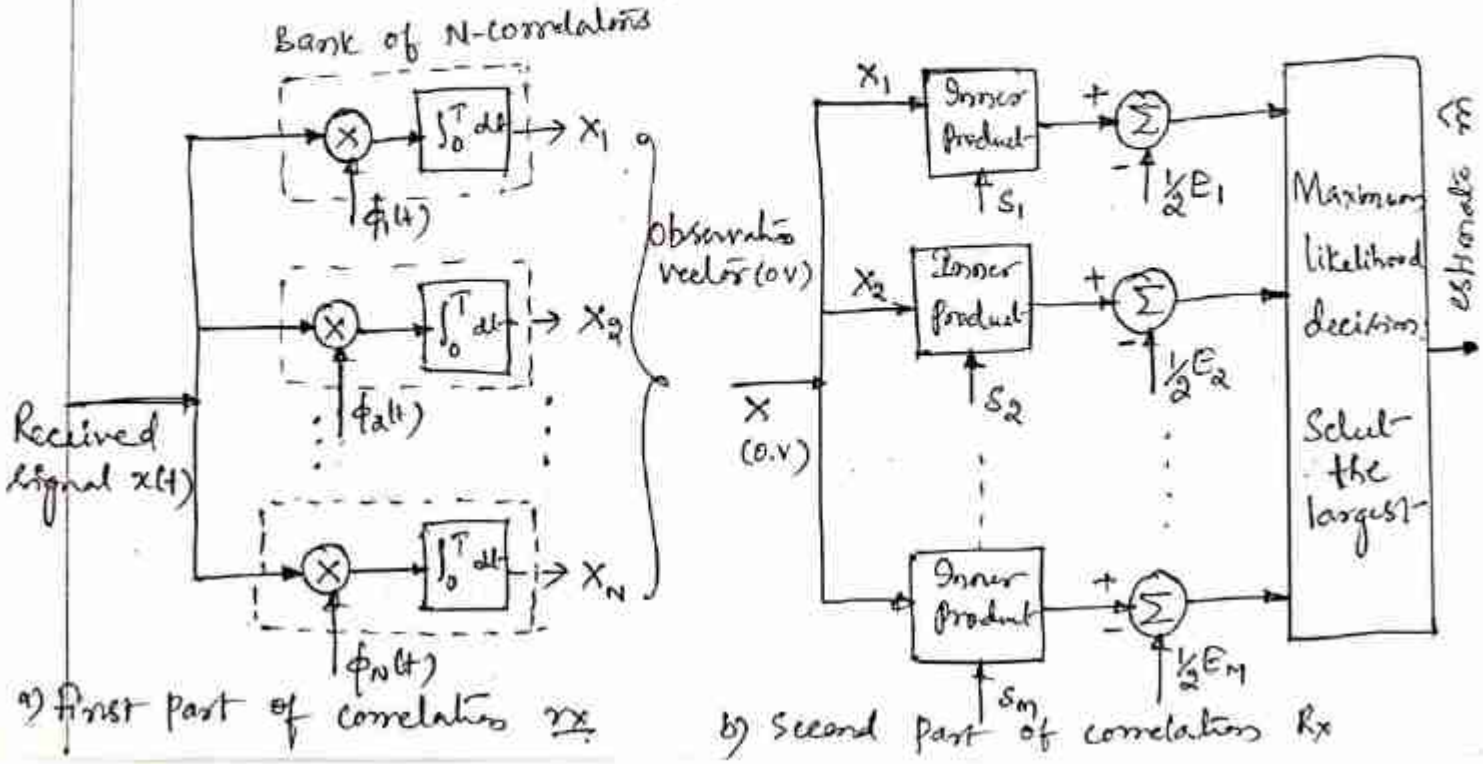
$$\sum_{j=1}^{N}(x_j - s_{kj})^2 = \sum_{j=1}^{N} x_j^2 - 2\sum_{j=1}^{N} x_j s_{kj} + \sum_{j=1}^{N} s_{kj}^2 \quad , \quad k = 1, 2 \ldots M$$

The first term in RHS gives energy of the received signal which is independent of k & hence can be ignored. The 3rd term gives the energy of the signal $s_k(t)$ & $\sum_{j=1}^{N} x_j s_{kj}$ is the inner product of received vector x & the signal vector s_k.

→ We can write 3rd term as $E_k = \sum_{j=1}^{N} s_{kj}^2$

Changing polarity of right hand side, this condition is equivalent to choosing that value of $k = i$ for which y_k is maximum, where

$$y_k = \sum_{j=1}^{N} x_j s_{kj} - \frac{1}{2} E_k \quad , \quad y_k \text{ is maximum for } k = i \text{ & } \hat{m} = m_i$$



a) first part of correlation Rx

b) Second part of correlation Rx

Scanned with CamScanner

Matched filter Receiver :-

The correlation receivers require analog multipliers & local reference of the orthonormal functions $\phi_1(t), \phi_2(t) \ldots \phi_N(t)$ which are time synchronized with respect to transmitted signals. This may be avoided using filters with appropriate impulse responses.

Consider a linear filter with impulse response $h_i(t)$, with the received signal $x(t)$ as input. The resulting filter output $y_i(t)$ is given by,

$$y_i(t) = \int_{-\infty}^{\infty} x(\tau)\, h(t-\tau)\, d\tau \quad \longrightarrow (1)$$

Let us assume the impulse response of filter as

$$h_i(t) = \phi_i(T-t)$$

where T is the symbol duration, [$h_i(t)$ is time reversed & delayed version of the function $\phi_i(t)$]

$$\therefore \quad y_i(t) = \int_{-\infty}^{\infty} x(\tau)\, \phi_i(T-t+\tau)\, d\tau$$

At $t = T$,

$$y_i(t)\Big|_{t=T} = y_i(T) = \int_{-\infty}^{\infty} x(\tau)\, \phi_i(\tau)\, d\tau$$

But $\phi_i(t) = 0$ outside the interval $0 \le t \le T$

$$\therefore \quad y_i(T) = \int_0^T x(\tau)\, \phi_i(\tau)\, d\tau$$

Comparing above eqn with the output of the correlator

$$x_i = \int_0^T x(\tau)\, \phi_i(\tau)\, d\tau$$

{ hint : $x_j = \int_0^T x(t)\phi_j(t)\,dt$

Scanned with CamScanner

We get $y_i(T) = x_i$

Where x_i is the i^{th} correlator o/p produced by received signal $x(t)$. Using a bank of N filters with impulse responses

$$h_i(t) = \phi_i(T-t), \quad i = 1, 2, 3 \ldots N$$

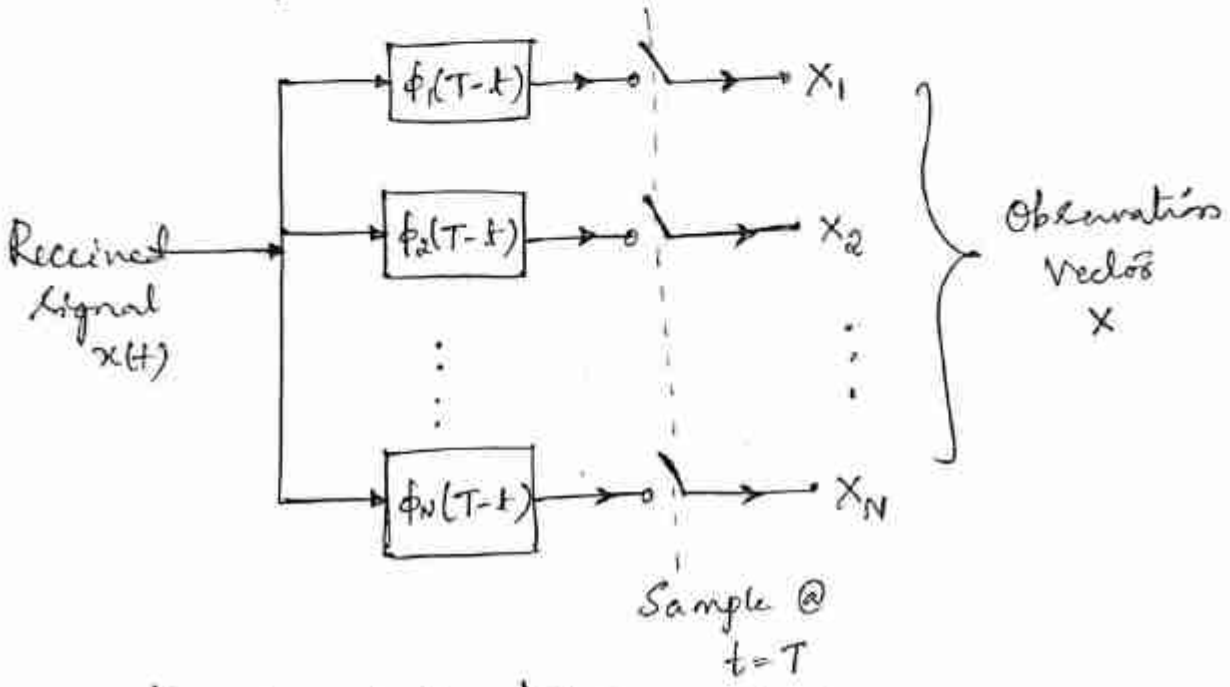


fig: Matched filter receiver.

Scanned with CamScanner

K. Prabhavathi
Asst Professor-1-
BGSIT, BG Nagar

Module-3

* Digital Modulation Techniques :-

Since base band signals are significant only at low frequencies, it cannot be transmitted over microwave link or satellite link. Therefore it is necessary to shift the spectrum of base band signal to high frequency range. & this can be done by modulating the carrier.

There are 3 types of digital modulation techniques.

1) Amplitude shift keying (ASK)
2) Frequency Shift Keying (FSK)
3) Phase Shift Keying (PSK).

1) Amplitude Shift Keying (ASK):-

In ASK, the amplitude of the carrier signal shifts between two levels in accordance with the input digital data.

Mathematically ASK signal can be written as

$$S(t) = \begin{cases} S_1(t) = A_c \cos 2\pi f_c t, & 0 \le t \le T_b \text{ Symbol '1'} \\ S_2(t) = 0 & 0 \le t \le T_b, \text{ Symbol '0'.} \end{cases}$$

2) Frequency Shift Keying (FSK):-

In FSK, the frequency of the carrier signal shifts in accordance with the input digital data keeping amplitude & phase constant.

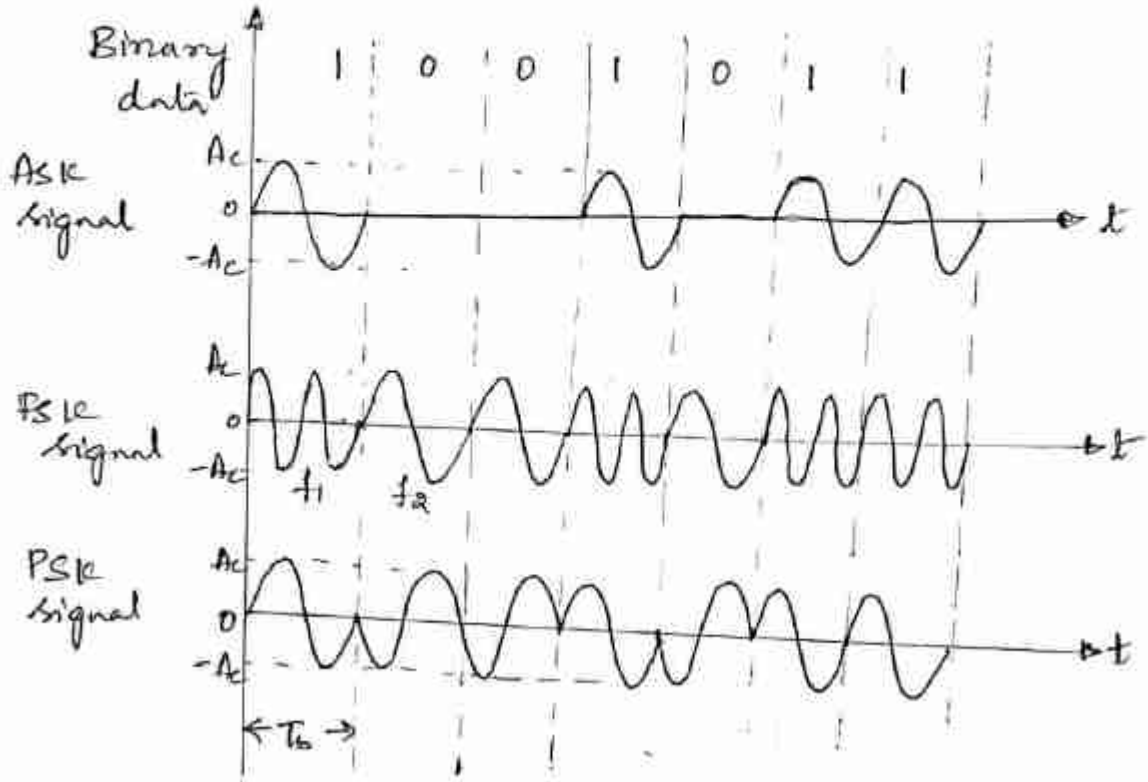
$$\text{i.e., } S(t) = \begin{cases} S_1(t) = A_c \cos 2\pi f_1 t & \text{Symbol '1', } 0 \le t \le T_b \\ S_2(t) = A_c \cos 2\pi f_2 t & \text{Symbol '0', } 0 \le t \le T_b. \end{cases}$$

3) Phase Shift Keying (PSK):-

In PSK, the phase of the carrier signal shifts in accordance with the input digital data keeping amplitude & frequency constant.

Scanned with CamScanner

$$S(t) = \begin{cases} S_1(t) = A_c \cos 2\pi f_c t & \text{Symbol '1'.} \\ S_2(t) = -A_c \cos 2\pi f_c t & \text{Symbol '0'.} \end{cases}$$



* Binary Phase Shift Keying (BPSK) :-

In BPSK, the transmitted signal S(t) can be written as,

$$S(t) = \begin{cases} S_1(t) = A_c \cos 2\pi f_c t, & 0 \le t \le T_b, \text{ Symbol '1'} \\ S_2(t) = -A_c \cos 2\pi f_c t, & 0 \le t \le T_b, \text{ symbol '0'} \end{cases}$$

Where $A_c = \sqrt{\frac{2E_b}{T_b}}$

$E_b \rightarrow$ transmitted energy per bit.

$T_b \rightarrow$ bit duration.

$$\therefore S(t) = \begin{cases} S_1(t) = \sqrt{\frac{2E_b}{T_b}} \cos 2\pi f_c t, & \text{Symbol '1'} \\ S_2(t) = -\sqrt{\frac{2E_b}{T_b}} \cos 2\pi f_c t, & \text{Symbol '0'} \end{cases}$$

Let $\phi_1(t)$ is the basis function (carrier signal)

$$\phi_1(t) = \sqrt{\frac{2}{T_b}} \cos 2\pi f_c t.$$

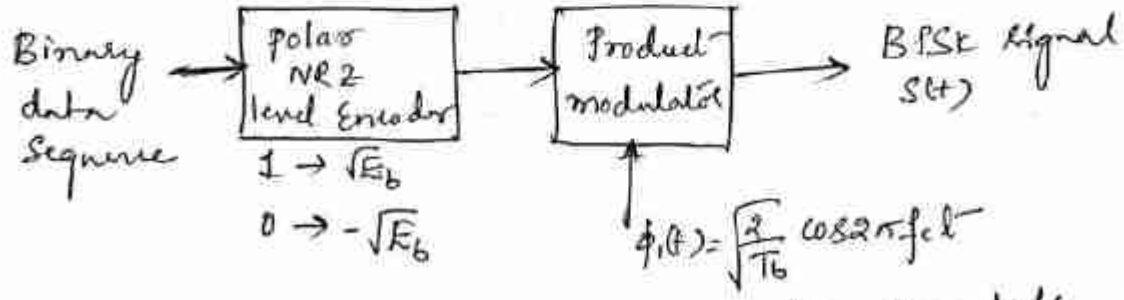
Scanned with CamScanner

BPSK signal in terms of $\phi_1(t)$ is

$$S(t) = \begin{cases} S_1(t) = \sqrt{E_b}\,\phi_1(t), & 0 \le t \le T_b, \text{ symbol '1'} \\ S_2(t) = -\sqrt{E_b}\,\phi_1(t), & 0 \le t \le T_b, \text{ symbol '0'} \end{cases}$$

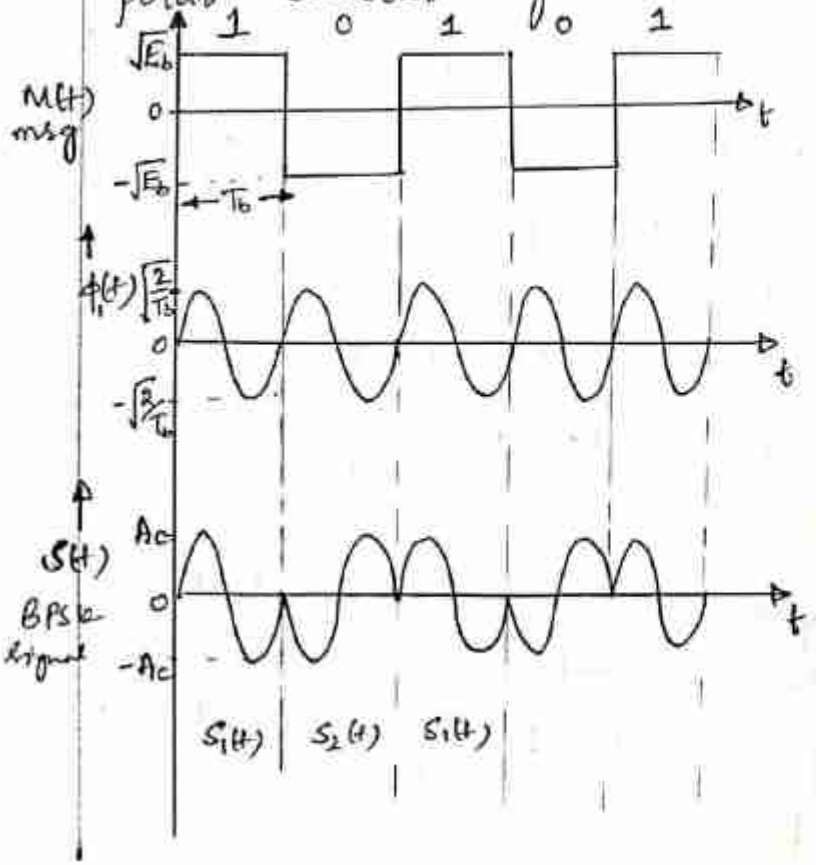
* Generation of BPSK :-

The generator (transmitter) consists of two components
1) Polar NRZ-level Encoder 2) Product Modulator.



→ Polar NRZ-level encoder represents symbols 1 and 0 of the incoming binary sequence by $\sqrt{E_b}$ and $-\sqrt{E_b}$ respectively.

→ product modulator multiplies the output of the NRZ polar encoder by the basis function $\phi_1(t)$ [carrier signal]



When symbol '1' is transmitted

$$S(t) = S_1(t) = \sqrt{\frac{2E_b}{T_b}}\cos 2\pi f_c t$$

When symbol '0' is transmitted

$$S(t) = S_2(t) = -\sqrt{\frac{2E_b}{T_b}}\cos 2\pi f_c t$$

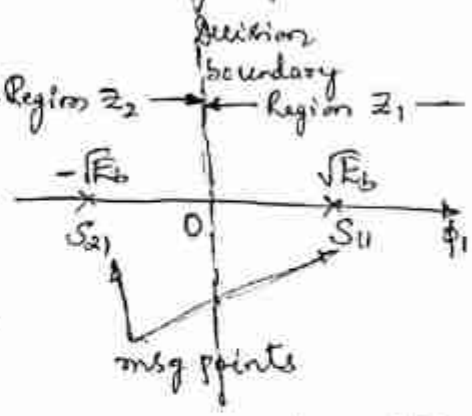
Scanned with CamScanner

Signal Space Diagram :-

Signal constellation consisting of two message points. The co-ordinates of the two message points are,

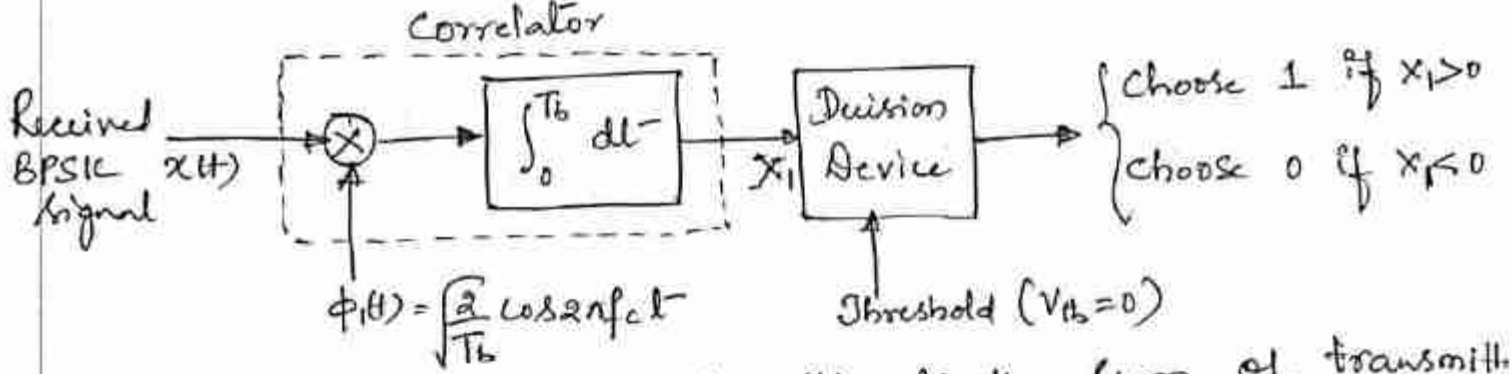
$$S_{11} = \int_0^{T_b} s_1(t)\,\phi_1(t)\,dt = \sqrt{E_b}$$

$$S_{21} = \int_0^{T_b} s_2(t)\,\phi_1(t)\,dt = -\sqrt{E_b}$$



It consists of two regions Z_1 & Z_2. Decision in region Z_1 corresponds to Symbol '1' is a set of all points that are close to S_{11}, while decision in region Z_2 corresponds to symbol '0' is a set of all points that are close to S_{21}.

* Coherent detection of BPSK signal :-



The received signal x(t) is the sum of transmitted signal and noise. The rxd signal x(t) is cross correlated with the locally generated carrier $\phi_1(t)$ on a bit-by-bit basis. For coherent detection, the locally generated carrier should be synchronized with the transmitter.

→ Decision device compares the correlator output against a Zero-threshold. If the threshold is exceeded, decision is made in favour of symbol 1; if not, the decision is made in favour of symbol 0.

Scanned with CamScanner

* **Probability of Symbol Error in BPSK :-**

Let $x(t)$ be the received signal then

$$x(t) = \begin{cases} s_1(t) + w(t), & 0 \le t \le T_b, \text{ symbol '1' is txd.} \\ s_2(t) + w(t), & 0 \le t \le T_b, \text{ symbol '0' is txd.} \end{cases}$$

The o/p of the correlator is given by, (for symbol '1')

$$x_1 = \int_0^{T_b} x(t)\, \phi_1(t)\, dt$$

$$x_1 = \int_0^{T_b} [s_1(t) + w(t)]\, \phi_1(t)\, dt$$

$$x_1 = \int_0^{T_b} s_1(t)\, \phi_1(t)\, dt + \int_0^{T_b} w(t)\, \phi_1(t)\, dt = \sqrt{E_b} + w_1$$

Where w_1 is the sample value of a AWGN having mean = 0 & variance = $\frac{N_0}{2}$.

Mean of $[x_1] = \sqrt{E_b}$

μ

Var $[x_1] = \frac{N_0}{2}$

$$\Rightarrow \boxed{\mu = \sqrt{E_b} \ \& \ \sigma^2 = \frac{N_0}{2}}$$

The Conditional probability density function (PDF) when '1' is txd is,

$$f_{x_1}(x_1|1) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x_1-\mu)^2}{2\sigma^2}}$$

$$f_{x_1}(x_1|1) = \frac{1}{\sqrt{2\pi \frac{N_0}{2}}}\, e^{-\left[\frac{x_1 - \sqrt{E_b}}{\sqrt{N_0}}\right]^2} = \frac{1}{\sqrt{\pi N_0}}\, e^{-\left[\frac{x_1 - \sqrt{E_b}}{\sqrt{N_0}}\right]^2}$$

Let $P_e(1)$ be probability of error when '1' is txd.

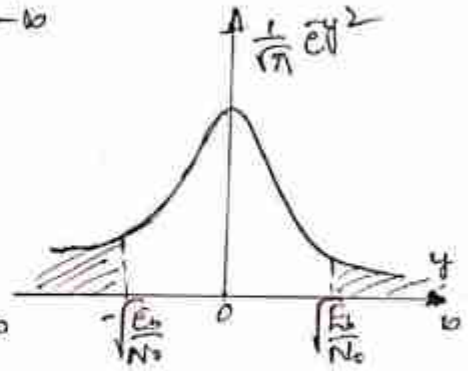
$$P_e(1) = \int_{-\infty}^{0} \frac{1}{\sqrt{\pi N_0}}\, e^{-\left[\frac{x_1 - \sqrt{E_b}}{\sqrt{N_0}}\right]^2} dx_1$$

Let $y = \frac{x_1 - \sqrt{E_b}}{\sqrt{N_0}}$ when $x_1 = -\infty$, $y = -\infty$; $x_1 = 0$, $y = -\sqrt{\frac{E_b}{N_0}}$

Scanned with CamScanner

$$dx_1 = \sqrt{N_0}\, dy$$

$$P_e(1) = \int_{-\infty}^{-\sqrt{\frac{E_b}{N_0}}} \frac{1}{\sqrt{\pi N_0}} e^{-y^2} \sqrt{N_0}\, dy = \int_{-\infty}^{-\sqrt{\frac{E_b}{N_0}}} \frac{1}{\sqrt{\pi}} e^{-y^2} dy$$



$$P_e(1) = \int_{\sqrt{\frac{E_b}{N_0}}}^{\infty} \frac{1}{\sqrt{\pi}} e^{-y^2} dy$$

Since $erfc(u) = \frac{2}{\sqrt{\pi}} \int_u^{\infty} e^{-y^2} dy$

erfc $\longrightarrow$ complementary error function.

$$\therefore P_e(1) = \frac{1}{2} erfc\left(\sqrt{\frac{E_b}{N_0}}\right) \quad \text{---} (1)$$

The o/p of the correlator when '0' is txd.

$$x_1 = -\sqrt{E_b} + w_1$$

Mean $[x_1] = \mu = -\sqrt{E_b}$ & Var $[x_1] = \sigma^2 = N_0/2$

$$\therefore \quad f_{x_1}(x_1|0) = \frac{1}{\sqrt{\pi N_0}} e^{-\left[\frac{x_1 + \sqrt{E_b}}{\sqrt{N_0}}\right]^2}$$

Let $P_e(0)$ be the probability of error when '0' is txd.

$$P_e(0) = \int_0^{\infty} \frac{1}{\sqrt{\pi N_0}} e^{-\left[\frac{x_1 + \sqrt{E_b}}{\sqrt{N_0}}\right]^2} dx_1$$

Let $y = \frac{x_1 + \sqrt{E_b}}{\sqrt{N_0}}$ $\quad \therefore P_e(0) = \int_{\sqrt{\frac{E_b}{N_0}}}^{\infty} \frac{1}{\sqrt{\pi N_0}} e^{-y^2} \sqrt{N_0}\, dy$

$x_1 = 0 \quad y = \sqrt{\frac{E_b}{N_0}}$

$x_1 = \infty \quad y = \infty$

& $dx_1 = \sqrt{N_0}\, dy$

$$P_e(0) = \int_{\sqrt{E_b/N_0}}^{\infty} \frac{1}{\sqrt{\pi}} e^{-y^2} dy$$

Scanned with CamScanner

$$P_e(0) = \frac{1}{2}\,erfc\sqrt{\frac{E_b}{N_0}} \quad \text{---} (2)$$

If symbols '1' & '0' are equiprobable, then

$$P(0) = P(1) = \frac{1}{2}$$

$$P_e = P(0)\,P_e(0) + P(1)\,P_e(1) = \frac{1}{2}\left[P_e(0) + P_e(1)\right]$$

$$\boxed{P_e = \frac{1}{2}\,erfc\sqrt{\frac{E_b}{N_0}}}$$

* Power Spectra of BPSK signals :-

To generate BPSK signal, base band signal used is in Polar NRZ format.

The PSD of a BPSK signal

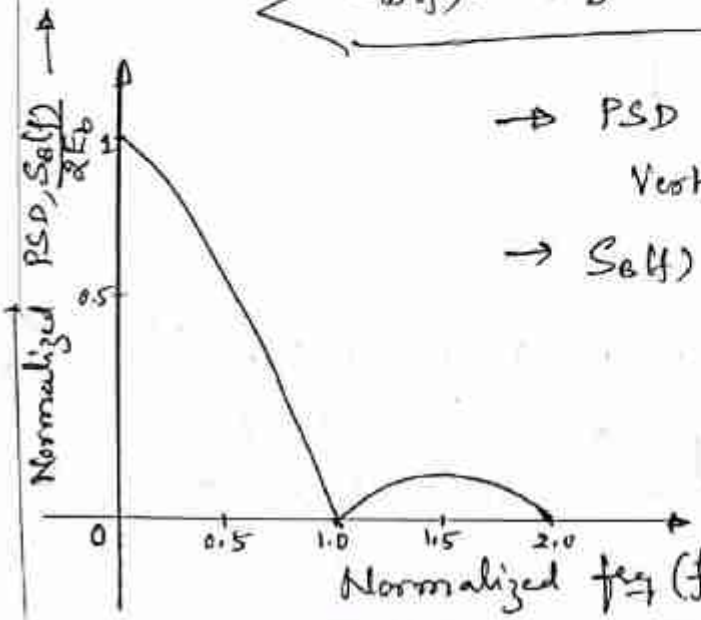
$$S_B(f) = a^2 T_b\, sinc^2(fT_b)$$

Where a = amplitude of rectangular pulse × amp of carrier

$$\hookrightarrow = \sqrt{E_b} \times \sqrt{\frac{2}{T_b}} = \sqrt{\frac{2E_b}{T_b}}$$

$$S_B(f) = \frac{2E_b}{T_b} \times T_b\, sinc^2(fT_b) = 2E_b\,\frac{\sin^2(\pi f T_b)}{(\pi f T_b)^2}$$

$$\boxed{S_B(f) = 2E_b\, sinc^2(fT_b)}$$



→ PSD $S_B(f)$ is symmetric about Vertical axis.

→ $S_B(f)$ goes to zero at multiples of bit rate i.e, $f = \pm\frac{1}{T_b}, \pm\frac{2}{T_b}$ ....

Scanned with CamScanner

* Binary Frequency Shift Keying (BFSK) :-

The BFSK signal can be represented as,

$$S(t) = \begin{cases} S_1(t) = \sqrt{\frac{2E_b}{T_b}} \cos 2\pi f_1 t, & 0 \le t \le T_b, \text{ Symbol '1'} \\ S_2(t) = \sqrt{\frac{2E_b}{T_b}} \cos 2\pi f_2 t, & 0 \le t \le T_b, \text{ Symbol '0'} \end{cases}$$

where,

f_1 & f_2 are two frequencies such that $f_1 > f_2$.

$E_b \rightarrow$ txd energy per bit

Let $\phi_1(t)$ & $\phi_2(t)$ are two basis functions defined as

$$\phi_1(t) = \sqrt{\frac{2}{T_b}} \cos 2\pi f_1 t \quad \& \quad \phi_2(t) = \sqrt{\frac{2}{T_b}} \cos 2\pi f_2 t$$

$$\therefore S(t) = \begin{cases} S_1(t) = \sqrt{E_b}\,\phi_1(t) & \text{Symbol '1'} \\ S_2(t) = \sqrt{E_b}\,\phi_2(t) & \text{Symbol '0'} \end{cases}$$

* Generation of BFSK :-

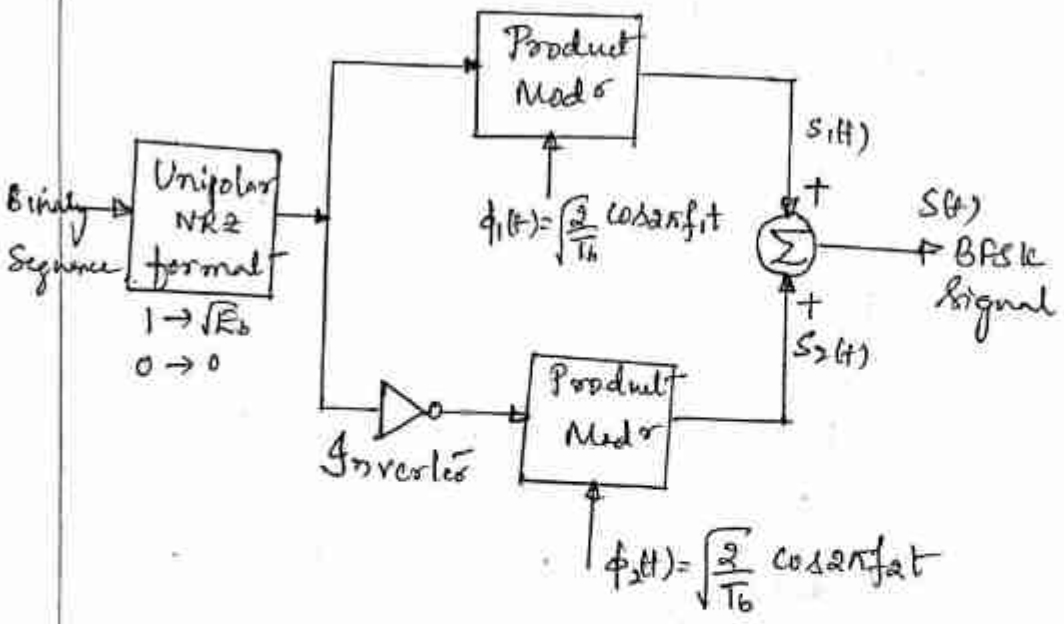
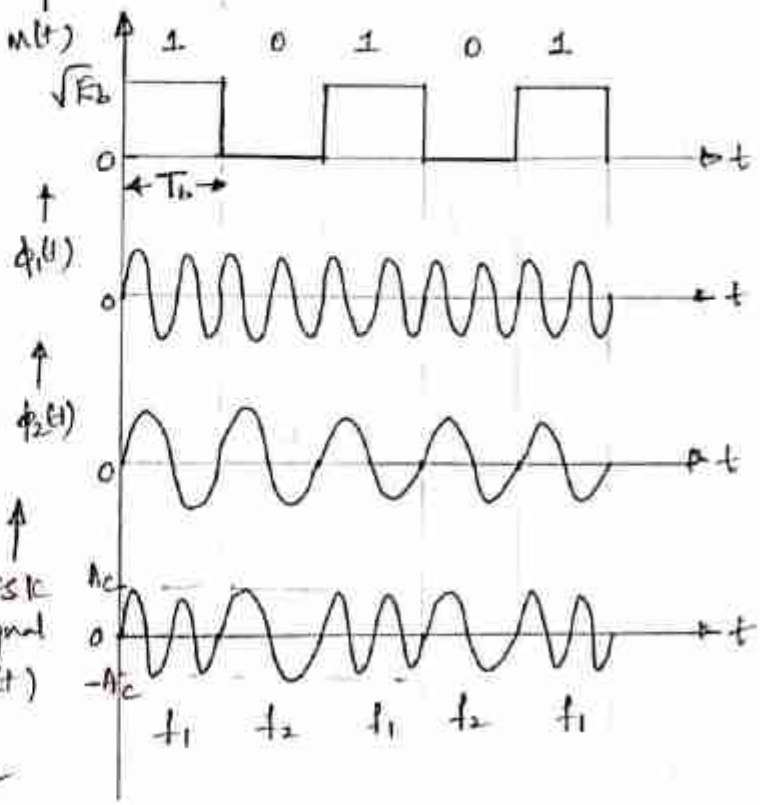


Fig above shows the block diagram of BFSK generation. Input binary data to unipolar NRZ format is applied to the top product mod & also to the bottom product mod through inverter

Scanned with CamScanner

When '1' is transmitted, the o/p of the summer is $s_1(t) = \sqrt{E_b}\,\phi_1(t)$.

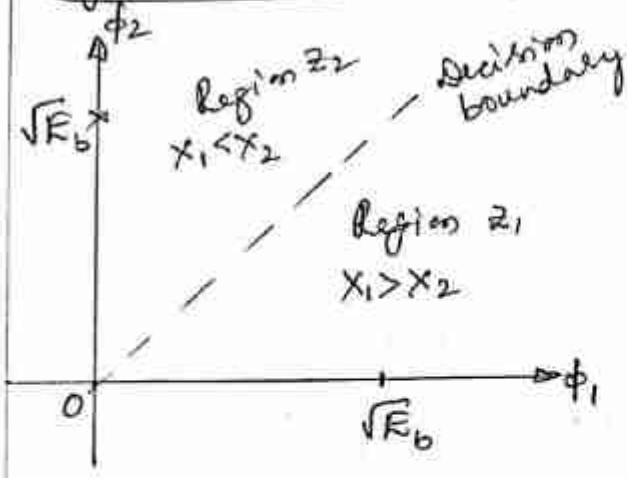
When '0' is transmitted the o/p of the summer is $s_2(t) = \sqrt{E_b}\,\phi_2(t)$.



For Symbol '1', only top multiplier will respond as the inverter o/p for symbol '1' is '0', bottom multiplier will give zero o/p.

For Symbol '0', since the inverter o/p '1' bottom xr will gives the o/p.

* Signal Space diagram :-



The co-ordinates of the two msg points.

For '1'

$$S_{11} = \int_0^{T_b} s_1(t)\,\phi_1(t)\,dt = \sqrt{E_b}$$

For '0'

$$S_{22} = \int_0^{T_b} s_2(t)\,\phi_2(t)\,dt = \sqrt{E_b}$$

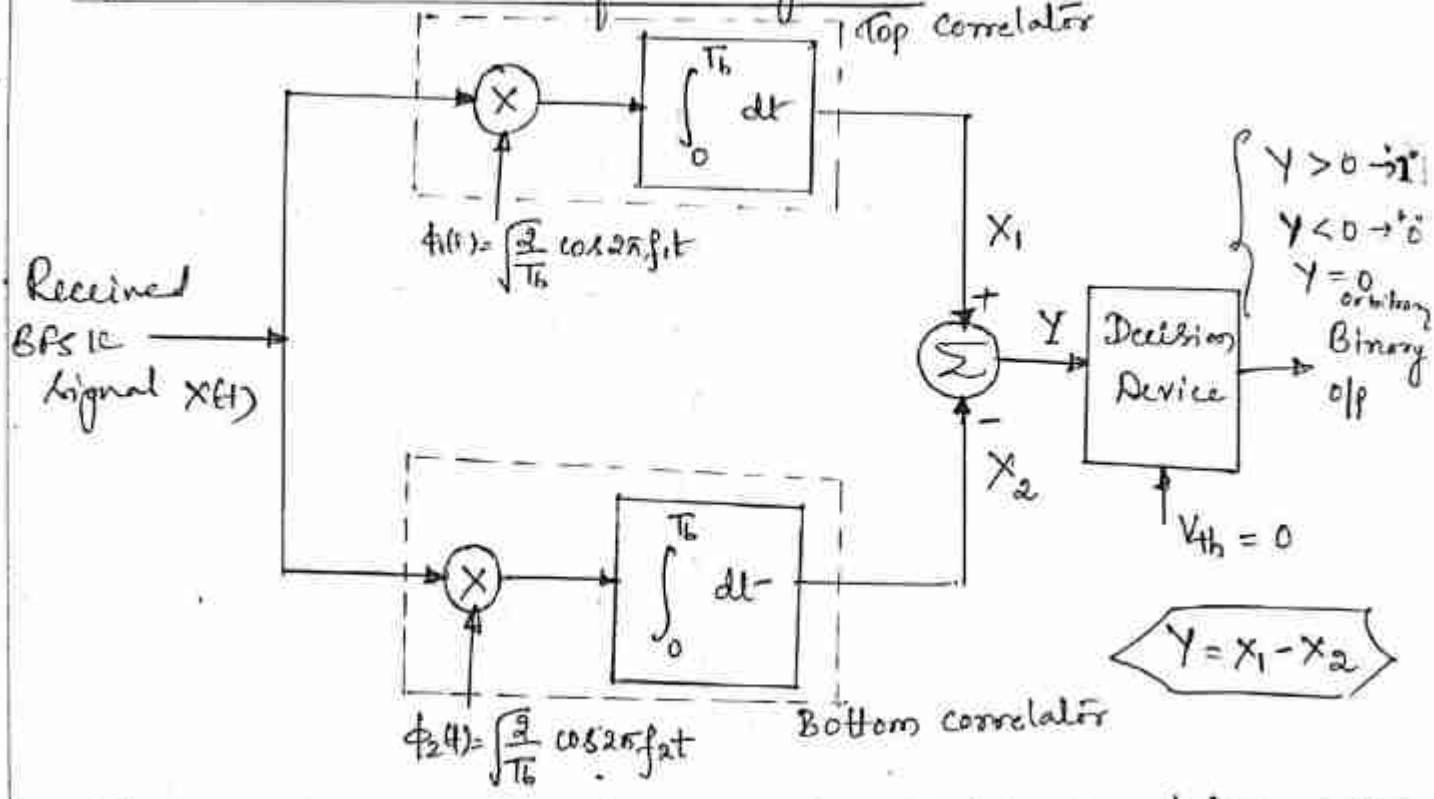
The region Z_1 corresponds to symbol '1' & the region Z_2 corresponds to symbol '0'

i.e., $x_1 > x_2$ Symbol '1'

$x_1 < x_2$ Symbol '0'

Scanned with CamScanner

* Coherent detection of Binary FSK :-



The received signal x(t) is cross correlated with two locally generated carriers $\phi_1(t)$ & $\phi_2(t)$ as shown. The outputs of the top and bottom correlators are X_1 & X_2 respectively. The o/p of the subtractor is $Y = X_1 - X_2$. If $Y > 0$, the decision is made in favour of Symbol '1' & If $Y < 0$ the decision is made in favour of Symbol '0'. If $Y = 0$, the decision is arbitrary.

* Probability of Symbol Error in BFSK :-

Let x(t) be the rxd BFSK signal

$$z(t) = \begin{cases} s_1(t) + w(t), & 0 \le t \le T_b, \text{ Symbol '1'} \\ s_2(t) + w(t), & 0 \le t \le T_b, \text{ Symbol '0'} \end{cases}$$

Where w(t) is the sample value of AWGN with Mean = 0 & PSD = Variance = $\frac{N_0}{2}$.

Scanned with CamScanner

The o/p of the top correlator is X_1 & o/p of the bottom correlator is X_2. Therefore $Y = X_1 - X_2$ which is the o/p of the subtractor.

If symbol '1' is txd, the rxd signal is $x(t) = s_1(t) + w(t)$. The o/p of the top correlator is

$$X_1 = \int_0^{T_b} [s_1(t) + w(t)] \phi_1(t)\, dt$$

$$X_1 = \int_0^{T_b} s_1(t)\, \phi_1(t)\, dt + \int_0^{T_b} w(t)\, \phi_1(t)\, dt$$

$$X_1 = \sqrt{E_b} + w_1$$

→ Mean $[X_1] = \sqrt{E_b}$ and Var $[X_1] = \frac{N_0}{2}$

The o/p of the bottom correlator is

$$X_2 = \int_0^{T_b} [s_1(t) + w(t)]\, \phi_2(t)\, dt$$

$$X_2 = \int_0^{T_b} s_1(t)\, \phi_2(t)\, dt + \int_0^{T_b} w(t)\, \phi_2(t)\, dt$$

$$X_2 = 0 + w_2$$

→ Mean $[X_2] = 0$ and Var $[X_2] = \frac{N_0}{2}$

WKT $Y = X_1 - X_2$

Mean $[Y]$ = Mean $[X_1]$ + Mean $[X_2]$

↳ $= \sqrt{E_b}$

& Var $[Y]$ = Var $[X_1]$ + Var $[X_2]$ = $\frac{N_0}{2} + \frac{N_0}{2} = N_0$

Therefore $\mu = \sqrt{E_b}$ & $\sigma^2 = N_0$ for symbol '1'

Scanned with CamScanner

Conditional probability density function [PDF] when '1' is txd.

$$f_Y(Y|1) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left[\frac{(Y-\mu)^2}{2\sigma^2}\right]}$$

$$f_Y(Y|1) = \frac{1}{\sqrt{2\pi N_0}} e^{-\left[\frac{Y-\sqrt{E_b}}{\sqrt{2N_0}}\right]^2}$$

Let $P_e(1)$ be the probability of error when '1' is transmitted,

$$P_e(1) = \int_{-\infty}^{0} f_Y(Y|1)\, dy = \int_{-\infty}^{0} \frac{1}{\sqrt{2\pi N_0}} e^{-\left[\frac{Y-\sqrt{E_b}}{\sqrt{2N_0}}\right]^2} dy$$

Let $Z = \frac{Y-\sqrt{E_b}}{\sqrt{2N_0}}$

When $y = -\infty$, $Z = -\infty$

$y = 0$, $Z = -\sqrt{\frac{E_b}{2N_0}}$

& $dy = \sqrt{2N_0}\, dZ$

$$P_e(1) = \int_{-\infty}^{-\sqrt{\frac{E_b}{2N_0}}} \frac{1}{\sqrt{2\pi N_0}} e^{-Z^2} \sqrt{2N_0}\, dZ$$

$$\Rightarrow P_e(1) = \int_{-\infty}^{-\sqrt{\frac{E_b}{2N_0}}} \frac{1}{\sqrt{\pi}} e^{-Z^2} dZ$$

Since $\frac{1}{\sqrt{\pi}} e^{-Z^2}$ is even symmetric function

[changing limit from $-\infty$ to $-\sqrt{\frac{E_b}{2N_0}}$ to $\sqrt{\frac{E_b}{2N_0}}$ to ∞]

Scanned with CamScanner

$$P_e(1) = \int_{\sqrt{\frac{E_b}{2N_0}}}^{\infty} \frac{1}{\sqrt{\pi}} e^{-z^2} dz$$

$$\Rightarrow P_e(1) = \frac{1}{2} \operatorname{erfc} \sqrt{\frac{E_b}{2N_0}} \quad \text{——— (1)}$$

Similarly if symbol '0' is transmitted, the received signal $x(t) = s_2(t) + w(t)$

The o/p of the top correlator is

$$x_1 = 0 + w_1$$

The o/p of the bottom correlator is

$$x_2 = \sqrt{E_b} + w_2$$

On simplifying we get

$$P_e(0) = \frac{1}{2} \operatorname{erfc} \sqrt{\frac{E_b}{2N_0}} \quad \text{——— (2)}$$

If symbols 0's & 1's are equiprobable then

$P(0) = P(1) = 1/2$

The probability of error in BPSK signal is,

$$P_e = P(0) P_e(0) + P(1) P_e(1)$$

$$P_e = \frac{1}{2}\left[P_e(0) + P_e(1) \right]$$

$$\boxed{P_e = \frac{1}{2} \operatorname{erfc} \sqrt{\frac{E_b}{2N_0}}}$$

Scanned with CamScanner

* Non-Coherent Orthogonal Modulation Techniques :-

* Non Coherent Binary FSK :-

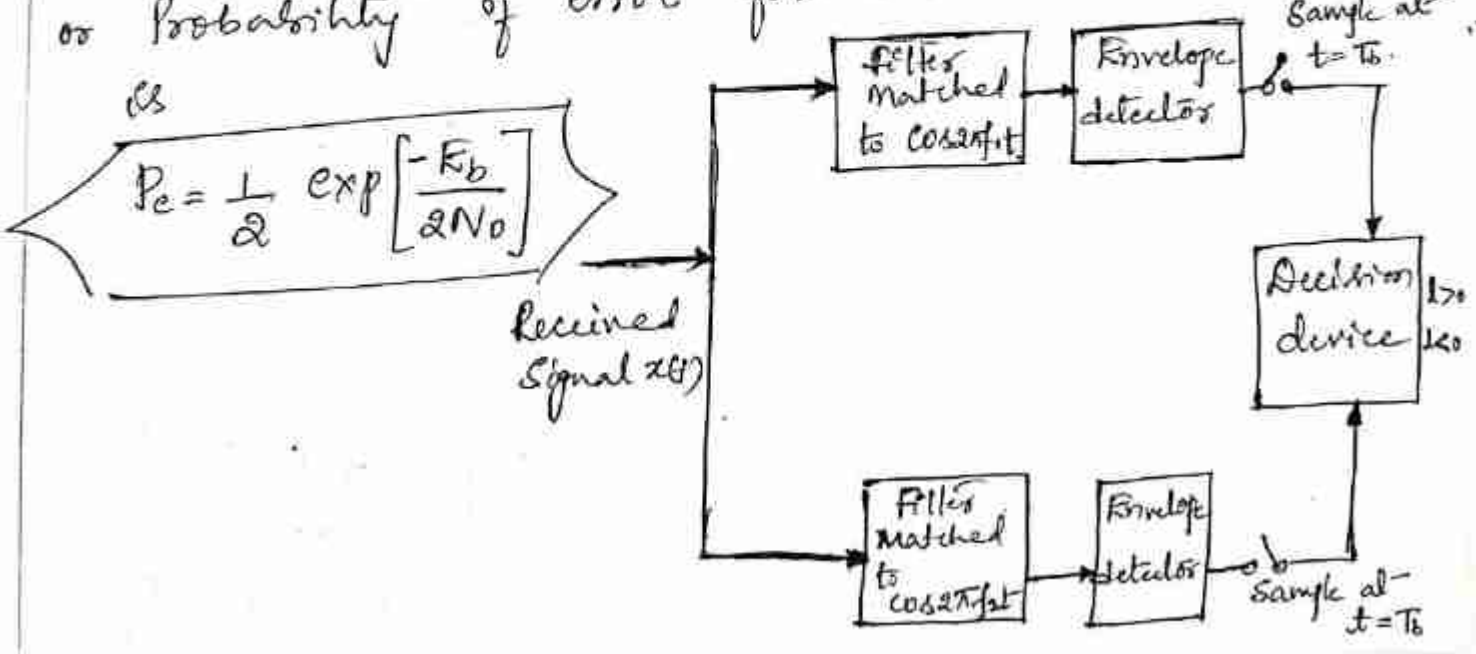
In Binary FSK, the transmitted signal corresponding symbol '1' & '0' is

$$S(t) = \begin{cases} S_1(t) = \sqrt{\frac{2E_b}{T_b}} \cos 2\pi f_1 t, & 0 \le t \le T_b \\ S_2(t) = \sqrt{\frac{2E_b}{T_b}} \cos 2\pi f_2 t, & 0 \le t \le T_b \end{cases}$$

Where E_b is the energy per bit and T_b is the bit duration.

The non-coherent receiver consists of a pair of matched filters followed by envelope detectors. The filter in the upper path is matched to $\cos 2\pi f_1 t$ & the filter in the lower path is matched to $\cos 2\pi f_2 t$. The resulting envelope detector o/p's are sampled at $t = T_b$ & their values are compared. The receiver decides in favour of symbol 1 if $l_1 > l_2$ & in favour of 0 if $l_1 < l_2$. The BER (bit error rate) or Probability of error for non coherent binary FSK is

$$P_e = \frac{1}{2} \exp\left[\frac{-E_b}{2N_0}\right]$$



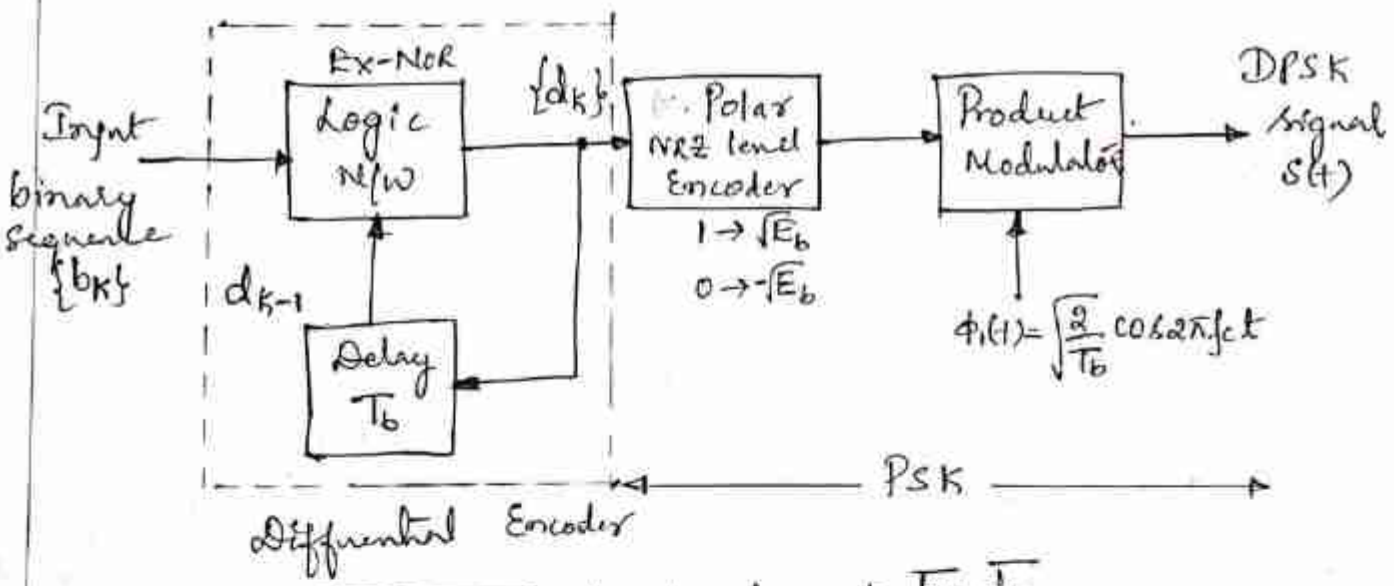
Scanned with CamScanner

* Differential Phase-Shift Keying [DPSK] :-

DPSK can be regarded as "non coherent version of PSK". The main advantage of DPSK is it eliminates the need for synchronizing the receiver to the transmitter by combining two basic operations at the transmitter.

1) Differential encoding of the input binary sequence
& 2) PSK of the encoded sequence, hence the name DPSK.

* Generation of DPSK Signal :-



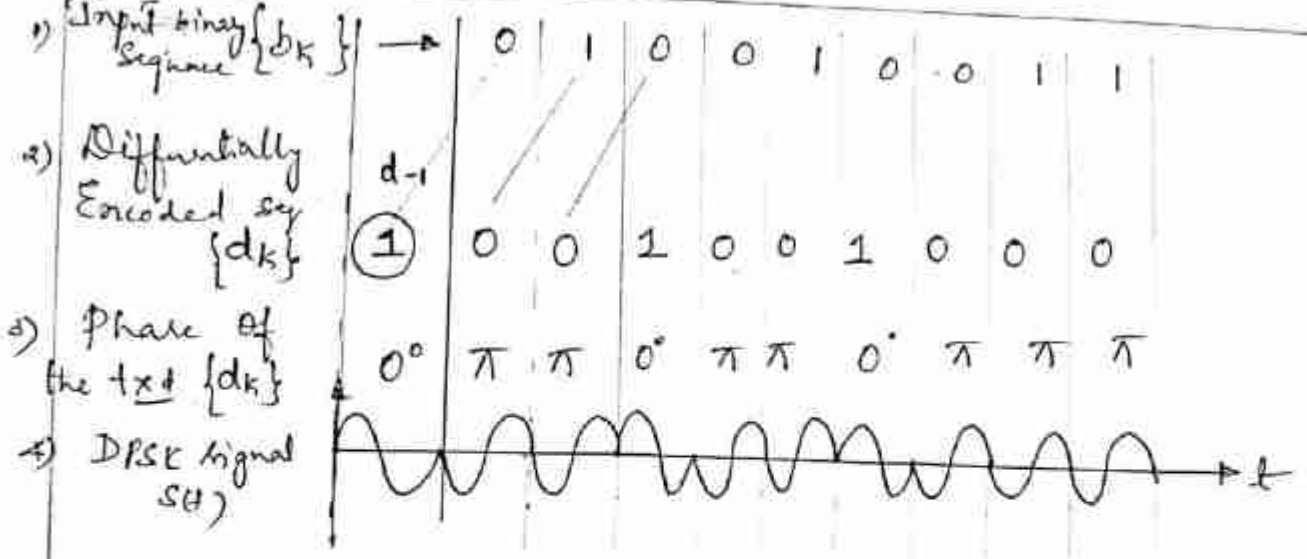
Where, $d_k = b_k \oplus d_{k-1} = b_k d_{k-1} + \overline{b_k}\,\overline{d_{k-1}}$

The transmitter consists of two functional blocks,

1) Logic n/w & one-bit delay (storage) element, which are interconnected so as to convert the i/p binary sequence $\{b_k\}$ into differentially encoded sequence $\{d_k\}$
2) PSK modr - The encoded sequence $\{d_k\}$ is represented in polar form & is multiplied with the carrier signal to get the desired DPSK signal.

Scanned with CamScanner

* Wave form :-

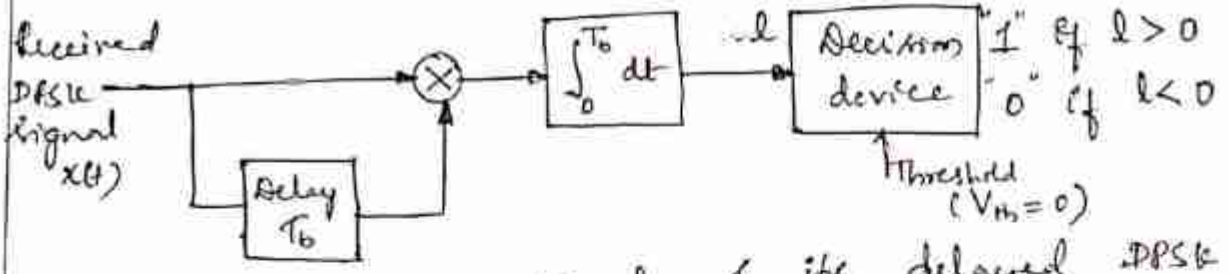


The product modr o/p is

For $d_k = 1$, $s(t) = \sqrt{\frac{2E_b}{T_b}} \cos 2\pi f_c t$

& For $d_k = 0$, $s(t) = -\sqrt{\frac{2E_b}{T_b}} \cos 2\pi f_c t$

* Detection of DPSK Signal :-



Received DPSK signal & its delayed DPSK wave is applied to the multiplier. The output of the multiplier is integrated every bit interval and the output is sampled and compared with $V_{Th} = 0$. If the o/p of the integrator is negative, then decision is in favour of symbol '0' and if the o/p of the integrator is positive, then the decision is in favour of symbol '1'.

⇒ The BER of DPSK is [Probability of symbol error]

$$P_e = \frac{1}{2} \exp\left[-\frac{E_b}{N_0}\right]$$

Scanned with CamScanner

* Quadrature phase Shift Keying [QPSK] :-

or Quadri phase shift keying

It is basically a band-width conserving modulation scheme.

The QPSK signal can be represented in time domain as,

$$S_i(t) = \sqrt{\frac{2E_b}{T}} \cos\left[2\pi f_c t + (2i-1)\frac{\pi}{4}\right] \quad \text{--- (1)}, \; 0 \le t \le T$$

Where $i = 1, 2, 3, 4$

$E_b \rightarrow$ fixed energy per bit

$T \rightarrow$ Symbol duration $[T = 2T_b]$

→ Since there are four distinct phases i.e., $\pi/4$, $3\pi/4$, $5\pi/4$ & $7\pi/4$, each phase can represent a pair of binary bits known as dibit.

Expanding eqn (1) using $\cos(A+B)$

$$S_i(t) = \sqrt{\frac{2E_b}{T}} \cos 2\pi f_c t \cos(2i-1)\frac{\pi}{4} - \sqrt{\frac{2E_b}{T_b}} \sin 2\pi f_c t \sin(2i-1)\frac{\pi}{4}$$

For $i = 1$

$$S_i(t) = \sqrt{\frac{E_b}{2}}\, \phi_1(t) - \sqrt{\frac{E_b}{2}}\, \phi_2(t)$$

Where, $\phi_1(t) = \sqrt{\frac{2}{T_b}} \cos 2\pi f_c t$

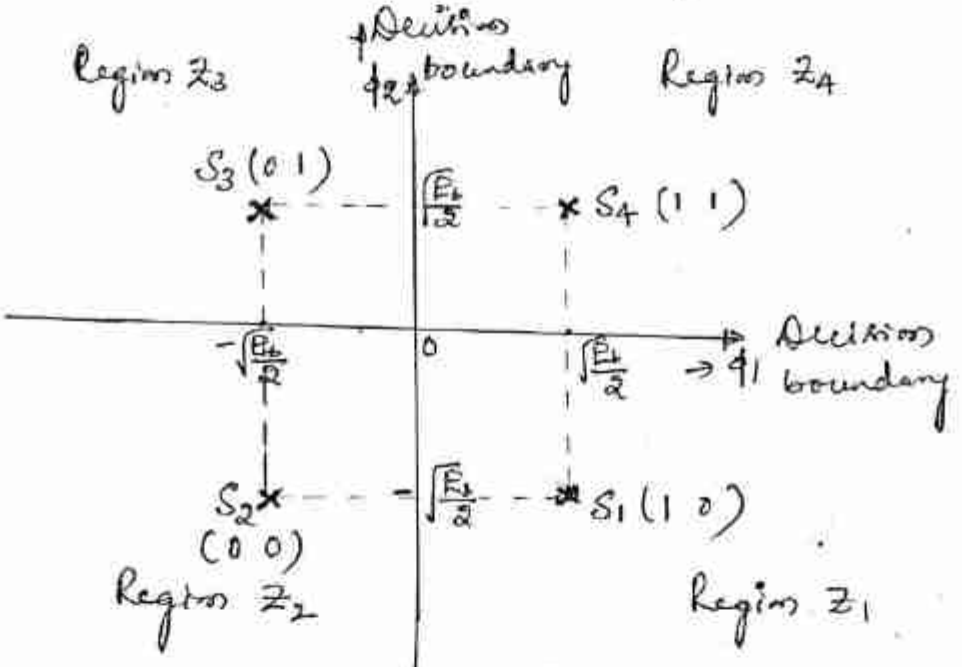
$\phi_2(t) = \sqrt{\frac{2}{T_b}} \sin 2\pi f_c t$ are two orthonormal basis functions & they are also called as quadrature carriers

Scanned with CamScanner

(Phase of QPSK signal)

i	$S_i(t)$	$\theta_i(t)$	s_{i1}	s_{i2}	Gray Encoded i/p dibit
1	$S_1(t)$	$\pi/4$	$\sqrt{\frac{E_b}{2}}$	$-\sqrt{\frac{E_b}{2}}$	1 0
2	$S_2(t)$	$3\pi/4$	$-\sqrt{\frac{E_b}{2}}$	$-\sqrt{\frac{E_b}{2}}$	0 0
3	$S_3(t)$	$5\pi/4$	$-\sqrt{\frac{E_b}{2}}$	$\sqrt{\frac{E_b}{2}}$	0 1
4	$S_4(t)$	$7\pi/4$	$\sqrt{\frac{E_b}{2}}$	$\sqrt{\frac{E_b}{2}}$	1 1

→ A QPSK signal has four message points (i.e. M = 4), which is shown in the signal-space diagram.



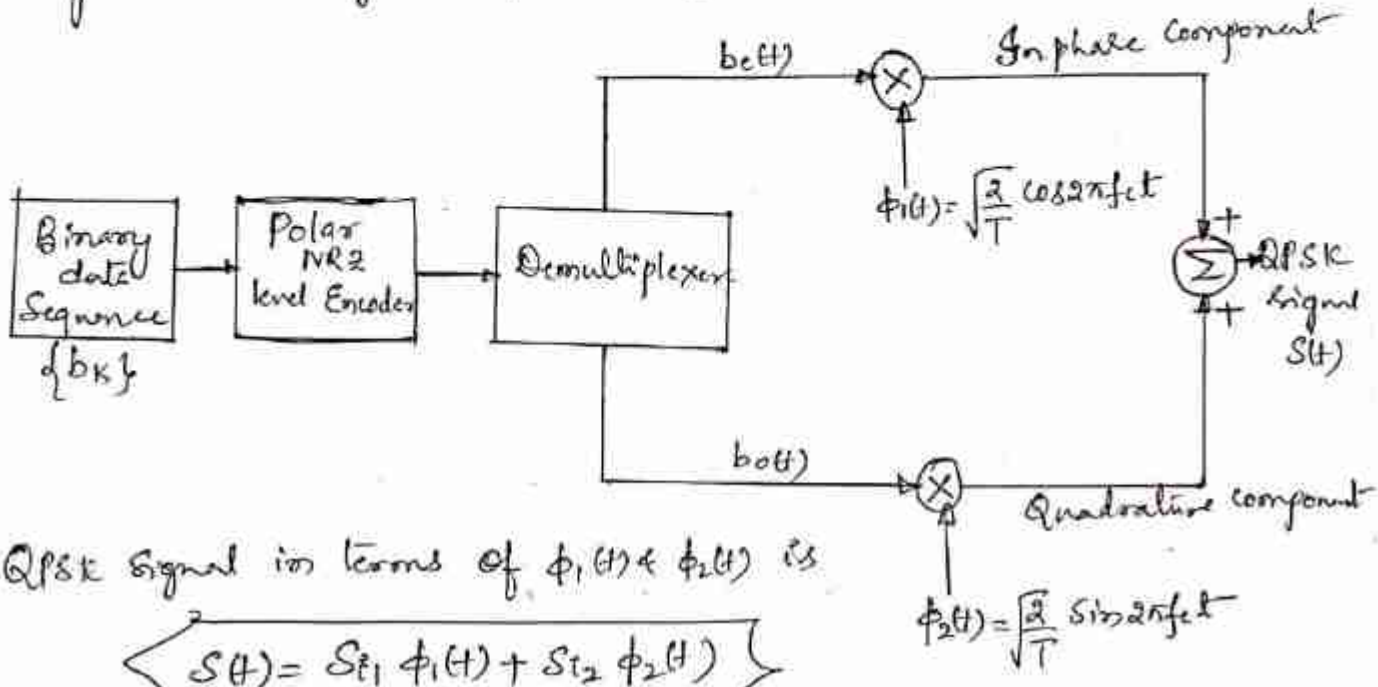
Signal-space diagram.

Generation of QPSK Signal :-

The function of the demultiplexer is to divide the binary wave produced by the polar NRZ level encoder into two separate binary waves, one of which represents the odd-numbered dibits & the other represents the even-numbered dibits.

Scanned with CamScanner

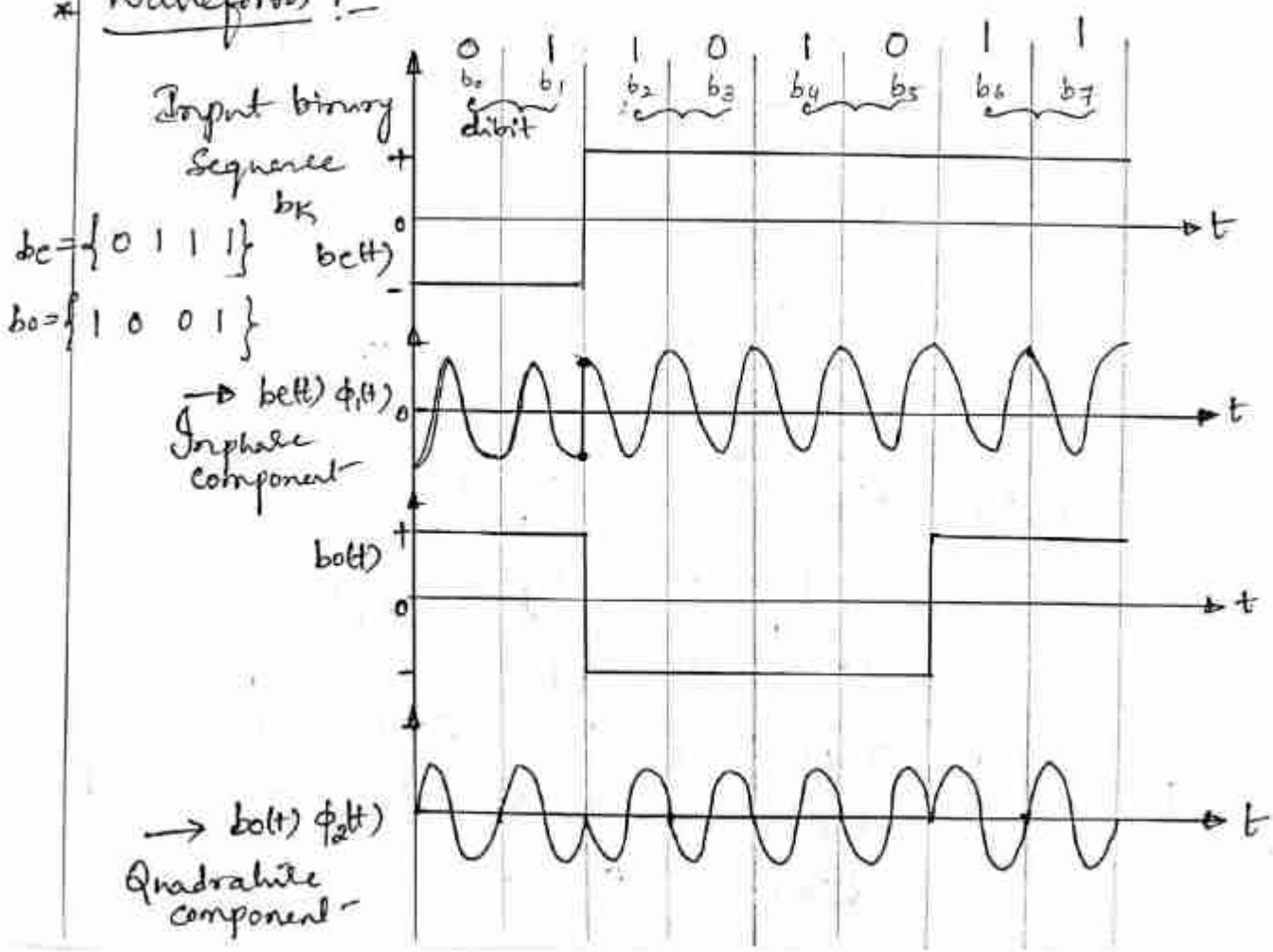
The QPSK transmitter may be viewed as two binary PSK generators that work in parallel, each at a bit rate equal to one-half the bit rate of the original binary sequence $\left[i.e.\ r_b = \frac{1}{2T_b}\right]$.



QPSK signal in terms of $\phi_1(t)$ & $\phi_2(t)$ is

$$S(t) = S_{i1}\,\phi_1(t) + S_{i2}\,\phi_2(t)$$

* Waveforms :-



Scanned with CamScanner

* Coherent detection of QPSK :-

The functional components of the QPSK receiver is as follows.

1) Pair of correlators, which have a common I/p $x(t)$. The two correlators are supplied with a pair of locally generated Orthonormal basis functions $\phi_1(t)$ & $\phi_2(t)$. The correlator outputs in response to the received signal $x(t)$ are denoted by x_1 & x_2 respectively.

2) Pair of decision devices,

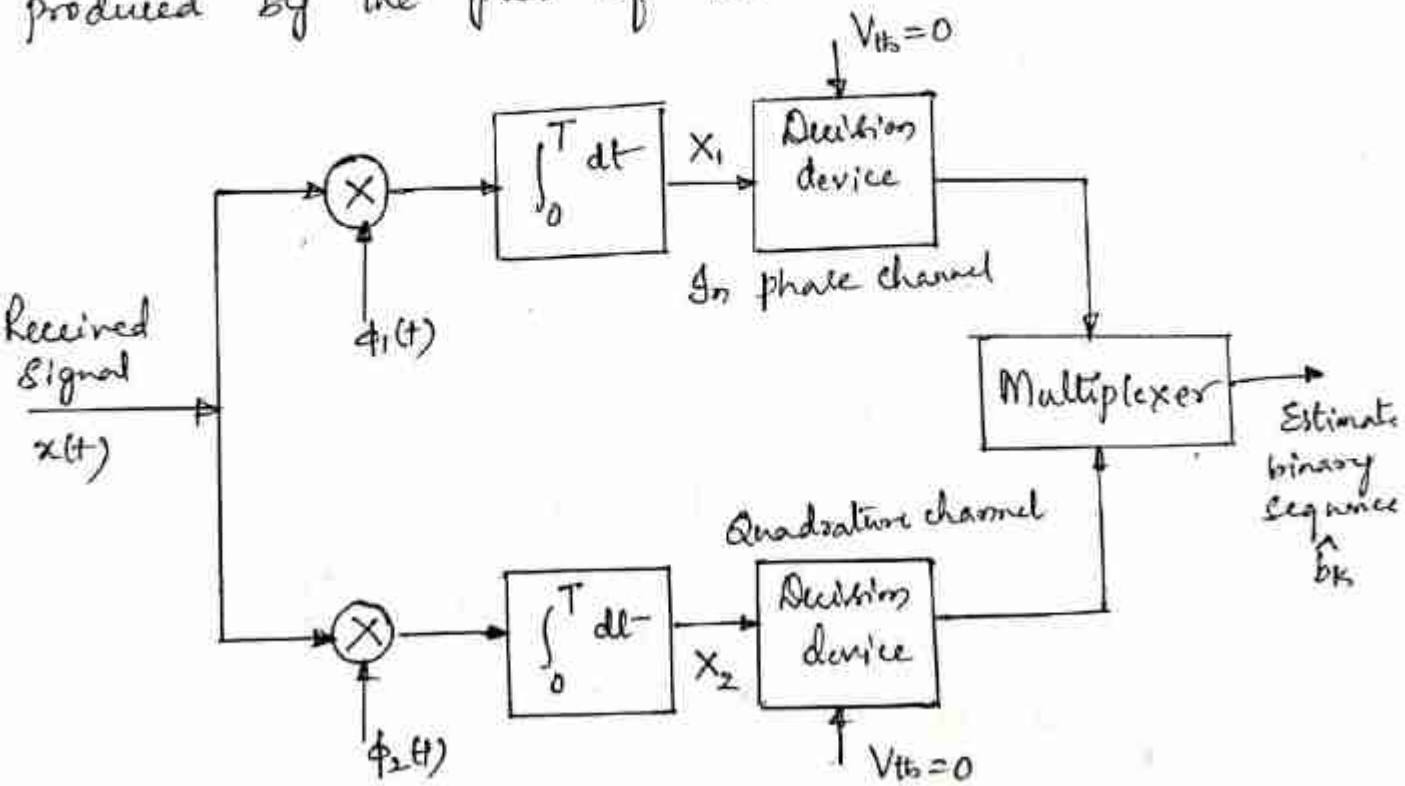
In phase channel if $x_1 > 0$ symbol is '1'

if $x_1 < 0$ Symbol is '0'

& Quadrature channel. if $x_2 > 0$ symbol is '1'

$x_2 < 0$ symbol is '0'

3) Multiplexer - is to combine the two binary sequences produced by the pair of decision devices.



Scanned with CamScanner

* Error probability of QPSK :—

In QPSK, the received signal $x(t)$ is defined as

$$x(t) = S_i(t) + w(t) \quad \begin{cases} 0 \le t \le T \\ i = 1, 2, 3, 4 \end{cases}$$

where $w(t)$ is the sample function of a white Gaussian noise of zero mean and power spectral density $\frac{N_0}{2}$.

If we calculate the probability of correct decision for one signal, this probability remains same for remaining three signals.

If $S_4(t)$ is the txd signal, then received signal will be, $x(t) = S_4(t) + w(t)$

The two correlators o/p's x_1 & x_2 are defined as follows.

$$x_1 = \int_0^T x(t)\,\phi_1(t)\,dt = \int_0^T [S_4(t) + w(t)]\,\phi_1(t)\,dt$$

$$x_1 = \sqrt{\frac{E_b}{2}} + w_1$$

and $$x_2 = \int_0^T x(t)\,\phi_2(t)\,dt = \int_0^T [S_4(t) + w(t)]\,\phi_2(t)\,dt$$

$$= \sqrt{\frac{E_b}{2}} + w_2$$

When signal $S_4(t)$ is txd, the received signal point lies in the decision region Z_4, if $x_1 > 0$ & $x_2 > 0$ leading to correct decision

Scanned with CamScanner

The probability of correct decision $P_c(S_4(t))$, when $S_4(t)$ is txd is given by.

$$P_c[S_4(t)] = \left[\frac{1}{\sqrt{\pi}} \int_{-\sqrt{\frac{E_b}{N_0}}}^{\infty} \exp(-y)^2 dy\right]^2$$

from the defn of complementary error fun,

$$\frac{1}{\sqrt{\pi}} \int_{-\sqrt{\frac{E_b}{N_0}}}^{\infty} e^{-y^2} dy = 1 - \frac{1}{2} erfc \sqrt{\frac{E_b}{N_0}}$$

$$\Rightarrow P_c[S_4(t)] = \left(1 - \frac{1}{2} erfc \sqrt{\frac{E_b}{N_0}}\right)^2$$

$$\hookrightarrow = 1 + \frac{1}{4} erfc^2\left(\sqrt{\frac{E_b}{N_0}}\right) - erfc \sqrt{\frac{E_b}{N_0}}$$

when $\frac{E_b}{N_0} \gg 1$, neglect $erfc^2\left(\sqrt{\frac{E_b}{N_0}}\right)$

$$\therefore P_c[S_4(t)] \cong 1 - erfc \sqrt{\frac{E_b}{N_0}}$$

Hence the probability of error when $S_4(t)$ is txd is

$$P_e(S_4(t)) = 1 - P_c(S_4(t))$$

$$\hookrightarrow \cong 1 - \left[1 - erfc \sqrt{\frac{E_b}{N_0}}\right]$$

$$\Rightarrow P_e[S_4(t)] = erfc \sqrt{\frac{E_b}{N_0}}$$

$$\text{Similarly } P_e[S_1(t)] = P_e[S_2(t)] = P_e[S_3(t)] = P_4[S_4(t)]$$

If all the four dibits occur with equal probability.

Scanned with CamScanner

$$P_e = \frac{1}{4}\left[P_e(S_1(t)) + P_e(S_2(t)) + P_e(S_3(t)) + P_e(S_4(t))\right]$$

$$\boxed{P_e = erfc\sqrt{\frac{E_b}{N_0}}}$$ BER of QPSK.

* **Power Spectra of QPSK signals:-**

In QPSK, both inphase & quadrature components have a common power spectral density.

To generate QPSK signal, baseband signal is in polar NRZ format.

$$S_B(f) = a^2 T \, sinc^2(fT) \quad \text{where } T = 2T_b$$

Where a = amp of the rectangular pulse x amp of the carrier. i.e, $\sqrt{\frac{E_b}{2}} \times \sqrt{\frac{2}{T_b}} = \sqrt{\frac{E_b}{T_b}}$

$$a = \sqrt{\frac{E_b}{T_b}} \Rightarrow a^2 = \frac{E_b}{T_b}$$

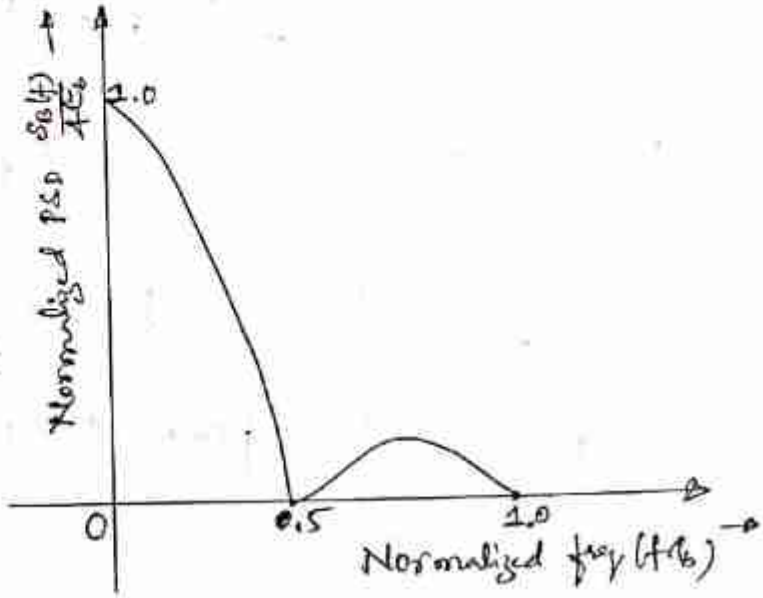
$$\therefore S_B(f) = \frac{E_b}{T_b} \times 2T_b \, sinc^2(fT) = 2E_b \, sinc^2(fT)$$

The PSD of the QPSK signal equals the sum of the individual PSD's of the in-phase & Quadrature components,

$$S_B(f) = 4E_b \, sinc^2(fT)$$

But $T = 2T_b$

$$\boxed{S_B(f) = 4E_b \, sinc^2(2fT_b)}$$



Scanned with CamScanner

Line codes.

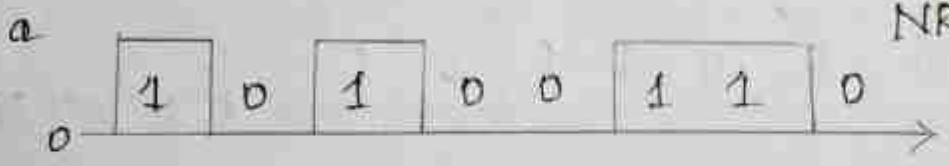
Digital data i.e binary digits are represented by different waveforms for the purpose of transmission over the channel. This process is known as line coding or transmission coding. It is used to avoid overlap and distortion

1) Unipolar format OR ON-OFF Signalling.

In the unipolar format, symbol 1 is represented by transmitting a pulse whereas symbol '0' is represented by switching off the pulse.

Unipolar NRZ-Signalling.

When the pulse occupies the full duration of a symbol, the unipolar format is said to be of unipolar non return-to-zero [NRZ] type.



NRZ-unipolar.

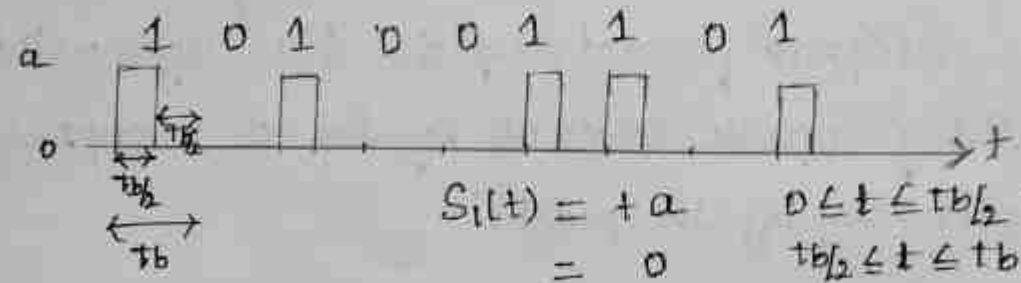
The signals are represented as

$s_1(t) = +a$ for $0 \le t \le T_b$ for symbol '1'

$s_2(t) = 0$ for $0 \le t \le T_b$ for symbol '0'

Unipolar RZ signalling

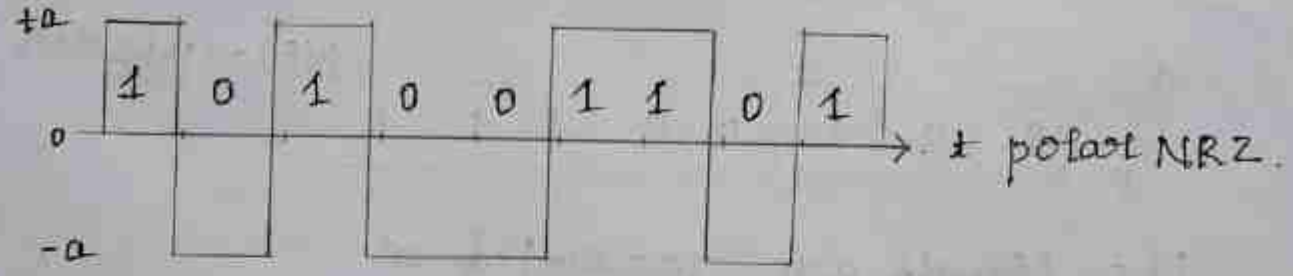
In unipolar RZ signalling, the symbol 1 is represented by positive pulse for half of the symbol bit duration & immediately returns to zero, shows the absence of pulse for remaining half of the symbol bit duration.



$S_1(t) = +a \quad 0 \le t \le T_b/2$
$= 0 \quad T_b/2 \le t \le T_b$ } for symbol 1

Polar format. $S_2(t) = 0 \quad 0 \le t \le T_b/2$ for symbol '0'

In polar signalling, the symbol 1 is represented by a positive pulse. while the symbol '0' is represented by a negative pulse



polar Non-return to zero format.

In polar NRZ signalling, the symbol 1 is represented by a positive pulse for full duration while the symbol 0 is represented by a negative pulse for full duration.

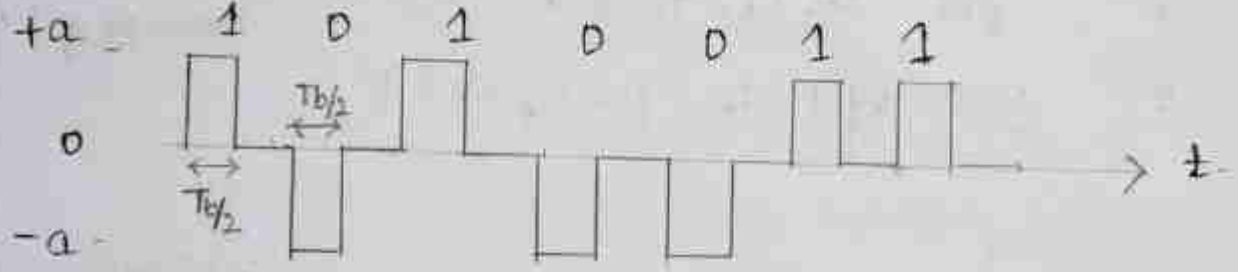
$S_1(t) = +a$ for $0 \le t \le T_b$ for symbol '1'

$S_2(t) = -a$ for $0 \le t \le T_b$ for symbol '0'

Polar return to zero format.

In polar return to zero signalling, the symbol '1' is represented by a +ve pulse for half of the symbol bit duration, returns to zero & shows the absence of pulse for remaining half of the symbol bit duration.

The symbol '0' is represented by negative pulse for half of the symbol bit duration and shows absence of pulse for remaining half of symbol bit duration.



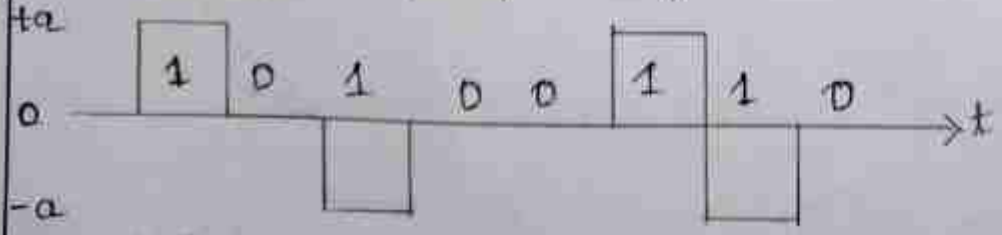
$$S_1(t) = +a \quad \text{for } 0 \le t \le T_b/2$$
$$= 0 \quad \text{for } T_b/2 \le t \le T_b \quad \Big\} \text{ for symbol '1'}$$

$$S_2(t) = -a \quad \text{for } 0 \le t \le T_b/2$$
$$= 0 \quad \text{for } T_b/2 \le t \le T_b \quad \Big\} \text{ for symbol '0'}$$

Bipolar format.

In bipolar signalling, the positive and negative pulses are used alternatively for the transmission of 1's and no pulses for the transmission of 0's. It uses Alternate mark inversion [AMI]

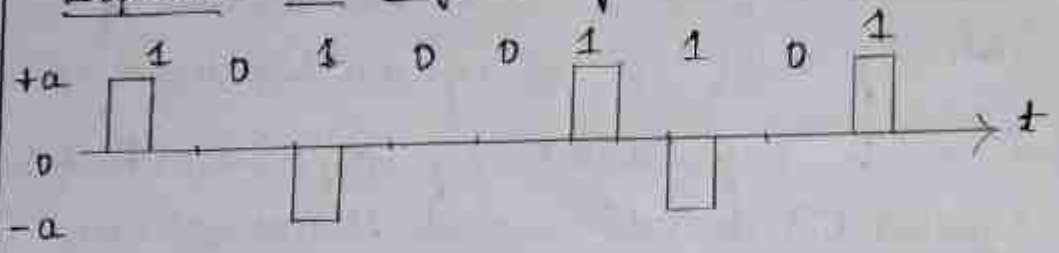
Bipolar NRZ signalling



$s_1(t) = \pm a$ for $0 \le t \le T_b$ for symbol '1' — odd no of 1's / even no of 1's

$s_2(t) = 0$ for $0 \le t \le T_b$ for symbol '0'

Bipolar RZ signalling



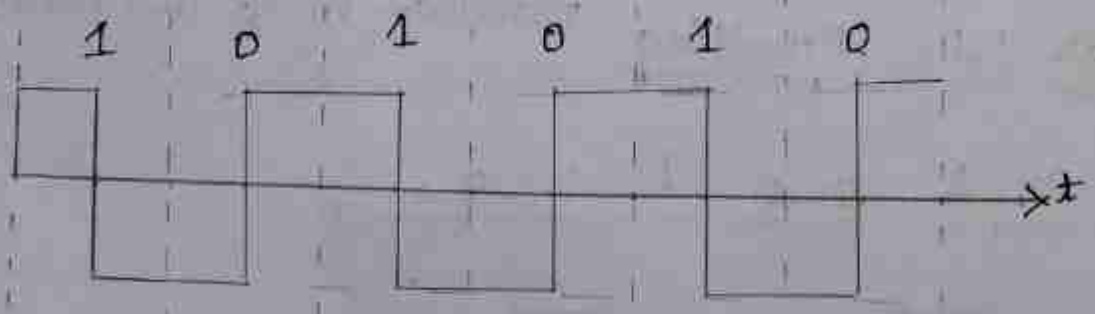
$s_1(t) = \pm a$ for $0 \le t \le T_b/2$
$= 0$ for $T_b/2 \le t \le T_b$
} for symbol '1' — odd no of 1's / even no of 1's

$s_2(t) = 0$ for $0 \le t \le T_b$ for symbol '0'

Manchester format or Biphase baseband signalling.

Symbol 1 is represented by transmitting a positive pulse for one-half of the symbol duration, followed by a negative pulse for the remaining half of the symbol duration.

Symbol '0' is represented by transmitting a negative pulse for one-half of the symbol duration followed by a positive pulse for the remaining half of the symbol duration.



$s_1(t) = +a$ for $0 \le t \le T_b/2$
$= -a$ for $T_b/2 \le t \le T_b$
} symbol 1, $s_1(t) = -a$ for $0 \le t \le T_b/2$; $+a$ for $T_b/2 \le t \le T_b$ } symbol '0'

Power spectral density of Line codes [PSD]

The function which describes the distribution of power of a signal at various frequencies in the frequency domain is called power spectral density.

* A discrete PAM signal or a line code is represented as a random process given by

$$X(t) = \sum_{k=-\infty}^{\infty} A_k . v(t-kT)$$

$A_k \rightarrow$ Amplitude value

$v(t) \rightarrow$ Basic pulse shape

$T \rightarrow$ Symbol duration.

The power spectral density of a line code $x(t)$ is given by.

$$S_x(f) = \frac{1}{T} |V(f)|^2 \sum_{n=-\infty}^{\infty} R_A(n) . e^{-j2\pi fnT}$$

$V(f) \rightarrow$ Fourier transform of basic pulse $v(t)$.

$R_A(n) \rightarrow$ Autocorrelation function.

* A_k is discrete random variable generated by random process. We can characterise random variable by its ensemble averaged autocorrelation function given by

$$R_A(n) = E[A_k . A_{k-n}]$$

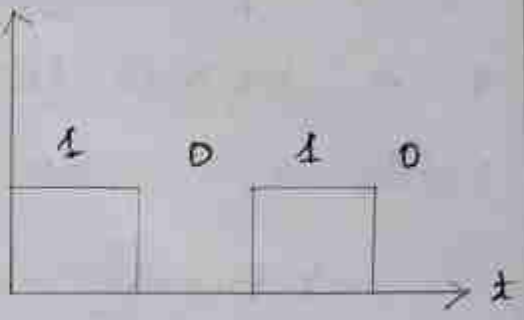
$$= \sum_{i=1}^{I} [A_k . A_{k-n}]_i . P_i$$

NRZ unipolar format

Consider unipolar NRZ form with symbol 1 and symbol '0' with equal probability.

i.e $P[A_k = 1] = 1/2$, $P[A_k = 0] = 1/2$

$$A_k = \begin{cases} +a & \text{for symbol 1} \\ 0 & \text{for symbol 0} \end{cases}$$



The autocorrelation function

$$R_A(n) = E[A_k . A_{k-n}]$$

Case i) For $n = 0$

$$R_A(n) = E[A_k . A_k] = E[A_k^2]$$

The probable values of $A_k . A_k$ are 0×0 & $a \times a$

$$R_A(0) = \sum_{i=1}^{2} [A_k . A_k] P_i$$

$$= [0^2] . \frac{1}{2} + [a^2] . \frac{1}{2}$$

$$\langle R_A(0) = a^2/2 \rangle$$

Case ii) For $n \neq 0$, A_k, A_{k-n} will have four possibilities with probabilities $1/4$ each.

$$R_A(n) = \sum_{i=1}^{4} [A_k . A_{k-n}]_i . P_i$$

$$= (0) \tfrac{1}{4} + (0) \tfrac{1}{4} + (0) \tfrac{1}{4} + a^2 (\tfrac{1}{4})$$

$$R_A(n) = a^2/4$$

A_k	A_{k-n}	$A_k . A_{k-n}$	P_i
0	0	0	$1/4$
0	a	0	$1/4$
a	0	0	$1/4$
a	a	a^2	$1/4$

The autocorrelation function $R_A(n)$ for unipolar NRZ format as:

$$R_A(n) = \begin{cases} a^2/2 & \text{for } n=0 \\ a^2/4 & \text{for } n \neq 0 \end{cases}$$

* The fourier transform of basic pulse $V(t)$ with unit amplitude is given by

$$\boxed{V(f) = F.T[V(t)] = T_b . \text{Sinc}(fT_b)}$$

* The power spectral density [PSD] of unipolar NRZ format is

$$S_X(f) = \frac{1}{T_b} |V(f)|^2 \sum_{n=-\infty}^{\infty} R_A(n) . e^{-j2\pi f n T_b} \longrightarrow (1)$$

Substitute the value of $V(f)$ and $R_A(n)$ in eq (1) we get.

$$S_X(f) = \frac{1}{T_b} . T_b^2 \text{Sinc}^2(fT_b) \left[\sum_{n=0} R_A(0) e^0 + \sum_{\substack{n=-\infty \\ n\neq 0}}^{\infty} R_A(n) e^{-j2\pi f n T_b} \right]$$

$$= \frac{1}{T_b} . T_b^2 \text{Sinc}^2(fT_b) \left[\frac{a^2}{2} + \sum_{\substack{n=-\infty \\ n\neq 0}}^{\infty} \frac{a^2}{4} e^{-j2\pi f n T_b} \right]$$

$$= T_b \text{Sinc}^2(fT_b) \left[\frac{a^2}{2} + \frac{a^2}{4} \left[\sum_{n=-\infty}^{\infty} e^{-j2\pi f n T_b} - 1 \right] \right]$$

[To remove $\sum_{n\neq 0}$, subtract 1 [at $n=0$] from $\sum_{n=-\infty}^{\infty} e^{-j2\pi f n T_b}$]

$$= \frac{a^2 T_b}{2} \text{Sinc}^2(fT_b) + \frac{a^2 T_b}{4} \text{Sinc}^2(fT_b) . \sum_{n=-\infty}^{\infty} e^{-j2\pi f n T_b} - \frac{a^2 T_b}{4} \text{Sinc}^2 fT_b$$

$$S_X(f) = \frac{a^2 T_b}{4} \text{Sinc}^2(fT_b) + \frac{a^2 T_b}{4} \text{Sinc}^2(fT_b) . \sum_{n=-\infty}^{\infty} e^{-j2\pi f n T_b} \longrightarrow (2)$$

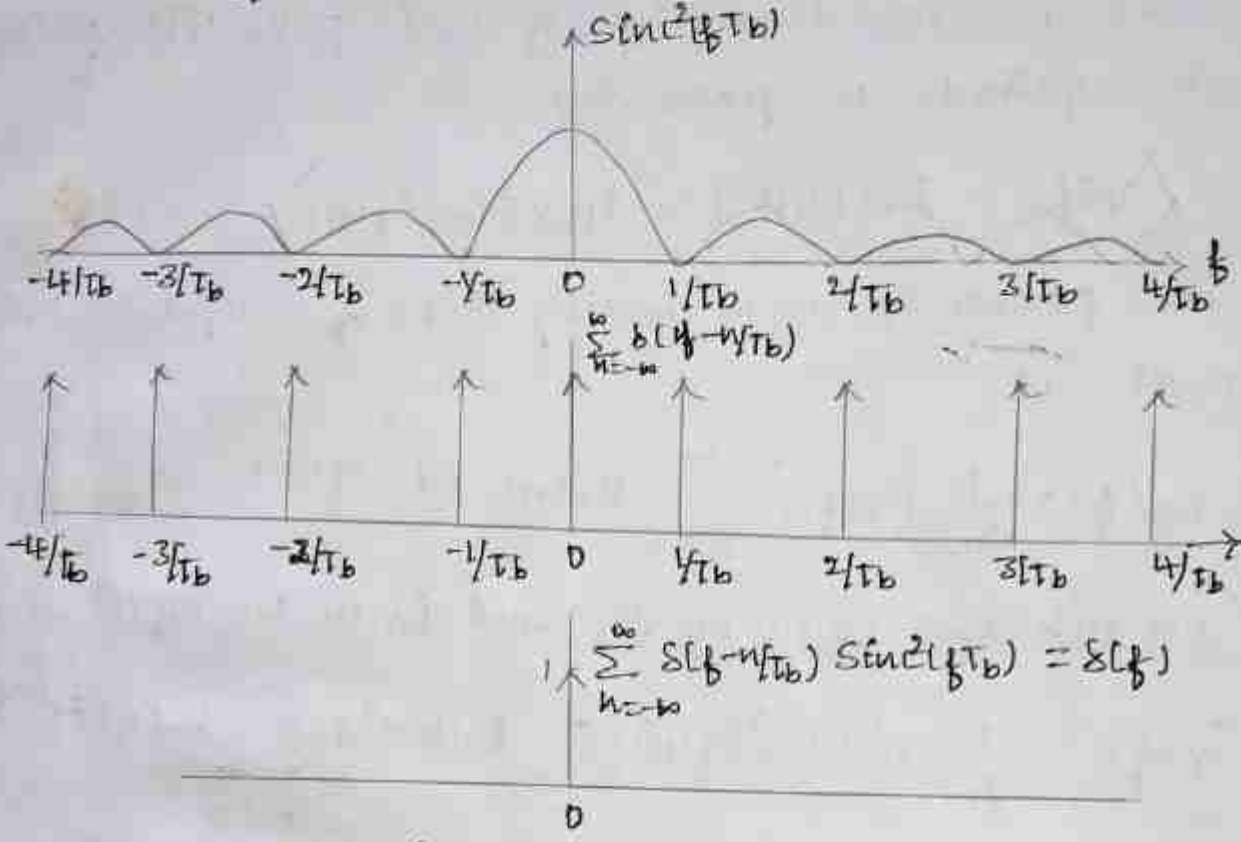
From Poisson sum formula

$$\sum_{n=-\infty}^{\infty} e^{-j2\pi f n T_b} = \frac{1}{T_b} \sum_{n=-\infty}^{\infty} \delta[f - n/T_b] \longrightarrow (3)$$

Substitute eq^n ③ in eq^n ②, we get.

$$S_X(f) = \frac{a^2 T_b}{4} \operatorname{sinc}^2(fT_b) + \frac{a^2 T_b}{4} \operatorname{sinc}^2(fT_b) \cdot \frac{1}{T_b} \sum_{n=-\infty}^{\infty} \delta\left(f - \frac{n}{T_b}\right) \rightarrow ④$$

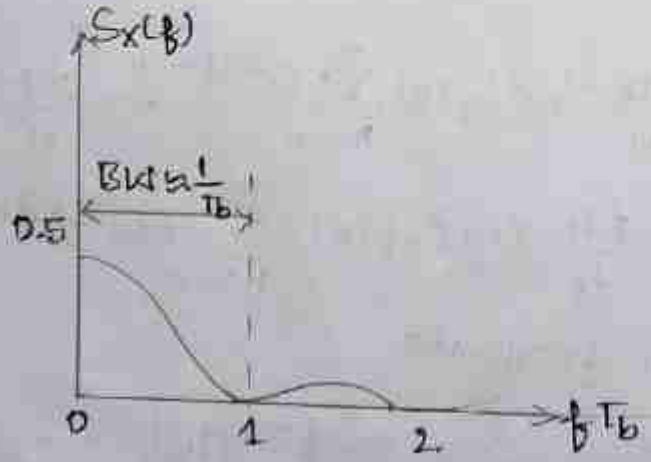
$$= \frac{a^2 T_b}{4} \operatorname{sinc}^2(fT_b) + \frac{a^2}{4} \operatorname{sinc}^2(fT_b) \cdot \sum_{n=-\infty}^{\infty} \delta(f - n/T_b) \rightarrow ④$$



$$\operatorname{sinc}^2(fT_b) \times \sum_{n=-\infty}^{\infty} \delta(f - n/T_b) \longrightarrow ⑤$$

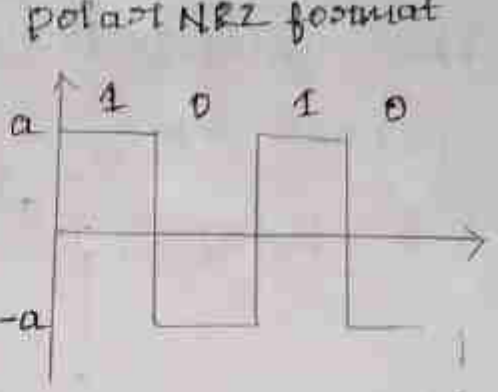
Substituting eq^n ⑤ in ④

$$\boxed{S_X(f) = \frac{a^2 T_b}{4} \operatorname{sinc}^2(fT_b) + \frac{a^2}{4} \delta(f)}$$



Normalized PSD for unipolar NRZ line code.

Polar NRZ format



* In polar NRZ format.

$$A_k = \begin{cases} +a & \text{for symbol 1} \\ -a & \text{for symbol 0} \end{cases}$$

* Symbol '0' & symbol '1' occurs with equal Probability

i.e $P[A_k = +a] = \frac{1}{2}$

$P[A_k = -a] = \frac{1}{2}$

* Auto correlation

$$R_A(n) = E[A_k . A_{k-n}]$$

case (i) For $n=0$

$$R_A(0) = E[A_k^2]$$

$$= \frac{a^2}{2} + \frac{a^2}{2}$$

$$\langle R_A(0) = a^2 \rangle$$

A_k	A_k^2	Equally probable
$+a$	a^2	$1/2$
$-a$	a^2	$1/2$

case (ii) For $n \neq 0$

A_k A_{k-n} will have four possibilities with probabilities $1/4$ each.

$$R_A(n) = E[A_k A_{k-n}]$$

$$= \frac{a^2}{4} - \frac{a^2}{4} - \frac{a^2}{4} + \frac{a^2}{4}$$

$$R_A(n) = 0.$$

A_k	A_{k-n}	$A_k A_{k-n}$	P_i
$-a$	$-a$	a^2	$1/4$
$-a$	a	$-a^2$	$1/4$
a	$-a$	$-a^2$	$1/4$
a	a	a^2	$1/4$

The autocorrelation function $R_A(n)$ for polar NRZ format is

$$R_A(n) = \begin{cases} a^2 & \text{for } n=0 \\ 0 & \text{for } n \neq 0. \end{cases}$$

The fourier transform of basic pulse $v(t)$ in the polar NRZ format is

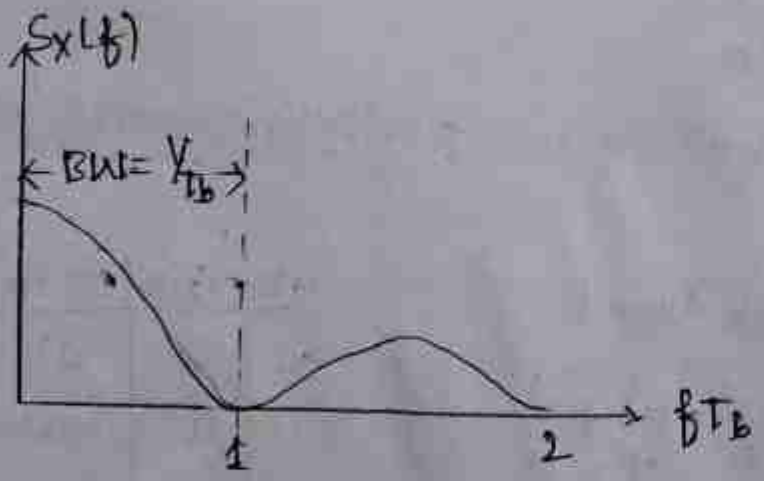
$$\langle V(f) = T_b \cdot \text{Sinc}(fT_b) \rangle$$

The power spectral density (PSD) of polar NRZ format is

$$S_X(f) = \frac{1}{T_b} |V(f)|^2 \sum_{n=-\infty}^{\infty} R_A(n) e^{-j2\pi f n T_b}$$

$$= \frac{1}{T_b} T_b^2 \text{Sinc}^2(fT_b) [a^2 + 0]$$

$$\langle S_X(f) = a^2 T_b \text{Sinc}^2(fT_b) \rangle$$



Normalised PSD of NRZ polar format.

Bipolar NRZ format

x In Bipolar NRZ format

$$A_k = \begin{cases} \pm a & \text{for symbol 1} \\ 0 & \text{for symbol 0} \end{cases}$$

* Let us assume that symbol '0' & '1' occur with equal probabilities

i.e $P[A_k = 0] = \frac{1}{2}$

$P[A_k = +a] = \frac{1}{4}$ &

$P[A_k = -a] = \frac{1}{4}$

* W.K.T autocorrelation function

$$R_A(n) = E[A_k A_{k-n}]$$

Case i: For $n=0$, we get

$$R_A(0) = E[A_k^2]$$

$$= 0 \cdot \frac{1}{2} + a^2 \cdot \frac{1}{4} + (-a)^2 \cdot \frac{1}{4}$$

$$\langle R_A(0) = \frac{a^2}{2} \rangle$$

Case ii: For $n=1$, we get

$$R_A(1) = 0\left(\tfrac{1}{4}\right) + 0\left(\tfrac{1}{4}\right) + 0\left(\tfrac{1}{4}\right) + (-a^2)\left(\tfrac{1}{4}\right)$$

$$\langle R_A(1) = -a^2/4 \rangle$$

A_k	A_{k-1}	$A_k A_{k-1}$	Pf, Equally probable
0	0	0	$\frac{1}{4}$
0	$\pm a$	0	$\frac{1}{4}$
$\pm a$	0	0	$\frac{1}{4}$
$\pm a$	$\mp a$	$-a^2$	$\frac{1}{4}$

NOTE: $R_A(1) = R_A(-1) = -a^2/4$

$$R_A(n) = R_A(-n)$$

Case iii: $n>1$

For $n>1$, the bits being considered are not adjacent and the $A_k A_{k-n}$ and their probabilities are given Eg. $\{0,0,0,0,0,-a^2,0,a^2\}$ & $\frac{1}{8}$ each respectively

A_k	A_{k-1}	A_{k-2}	$A_k A_{k-2}$	Equally probable P_i
0	0	0	0	$\frac{1}{8}$
0	0	1 $\begin{cases}+a\\-a\end{cases}$	0	$\frac{1}{8}$
0	1 $\begin{cases}+a\\-a\end{cases}$	0	0	$\frac{1}{8}$
0	1 $\begin{cases}+a\\-a\end{cases}$	1 $\begin{cases}-a\\+a\end{cases}$	0	$\frac{1}{8}$
1 $\begin{cases}+a\\-a\end{cases}$	0	0	0	$\frac{1}{8}$
1 $\begin{cases}+a\\-a\end{cases}$	0	1 $\begin{cases}-a\\+a\end{cases}$	$-a^2$	$\frac{1}{8}$
1 $\begin{cases}+a\\-a\end{cases}$	1 $\begin{cases}-a\\+a\end{cases}$	0	0	$\frac{1}{8}$
1 $\begin{cases}+a\\-a\end{cases}$	1 $\begin{cases}-a\\+a\end{cases}$	1 $\begin{cases}+a\\-a\end{cases}$	a^2	$\frac{1}{8}$

Then

$R(n>1)$

$$R(2) = E[A_k \cdot A_{k-2}]$$

$$= \left[0\left(\tfrac{1}{8}\right)+0\left(\tfrac{1}{8}\right)+0\left(\tfrac{1}{8}\right)+0\left(\tfrac{1}{8}\right)+0\left(\tfrac{1}{8}\right)-a^2\left(\tfrac{1}{8}\right)+0\left(\tfrac{1}{8}\right)+a^2\left(\tfrac{1}{8}\right)\right]$$

$$R(2) = 0$$

$$R_A(n>1) = 0$$

Let define $R_A(n)$ as

$$R_A(n) = \begin{cases} A^2/2 & n=0 \\ -A^2/4 & |n|=1 \\ 0 & |n|>1 \end{cases}$$

The PSD of the bipolar NRZ format is

$$S_X(f) = \frac{1}{T_b} |V(f)|^2 \sum_{n=-\infty}^{\infty} R_A(n) e^{-j2\pi f n T_b} \quad \text{---------- (1)}$$

Substitute the values of $V(f)$ and $R_A(n)$ in eq^n (1) we get.

$$S_X(f) = \frac{1}{T_b}\left[T_b^2 \operatorname{sinc}^2(fT_b)\right]\left[\sum_{n=-1} R_A(-1) e^{j2\pi f T_b} + \sum_{n=0} R_A(0) + \sum_{n=1} R_A(1) e^{-j2\pi f T_b}\right]$$

$$S_X(f) = T_b \operatorname{sinc}^2(fT_b)\left[R_A(-1) e^{j2\pi f T_b} + R_A(0) + R_A(1) e^{-j2\pi f T_b}\right]$$

$$R_A(1) = R_A(-1)$$

$$S_X(f) = T_b \operatorname{sinc}^2(fT_b)\left[R_A(1)\left[\underbrace{e^{j2\pi f T_b} + e^{-j2\pi f T_b}}_{2\cos 2\pi f T_b}\right] + R_A(0)\right]$$

$$= T_b \operatorname{sinc}^2(fT_b)\left[-\frac{a^2}{4}\left[2\cos^{(2\pi f T_b)}\right] + \frac{a^2}{2}\right]$$

$$S_X(f) = T_b \operatorname{sinc}^2(fT_b)\left[-\frac{a^2}{2}\cos^{(2\pi f T_b)} + \frac{a^2}{2}\right]$$

$$S_X(f) = T_b \operatorname{sinc}^2(fT_b)\frac{a^2}{2}\left[1 - \cos 2\pi f T_b\right]$$

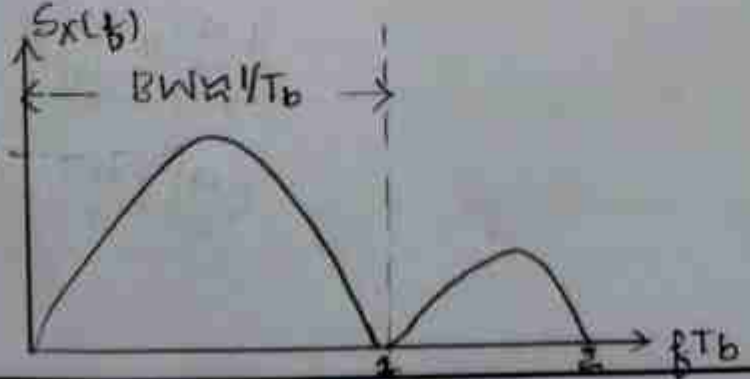
$$\frac{e^{j\theta} + e^{-j\theta}}{2} = \cos\theta$$

$$e^{j\theta} + e^{-j\theta} = 2\cos\theta.$$

W.K.T $\quad \cos 2\theta = 1 - 2\sin^2\theta.$

$$S_X(f) = \frac{a^2}{2}\left[T_b \operatorname{sinc}^2(fT_b)\left[2\sin^2 \pi f T_b\right]\right]$$

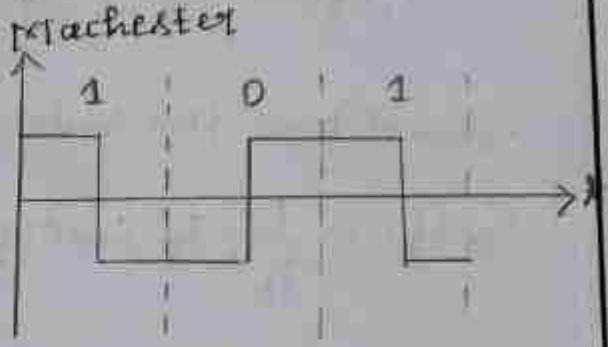
$$S_X(f) = a^2 T_b \operatorname{sinc}^2(fT_b)\sin^2(\pi f T_b)$$



PSD of bipolar NRZ format.

PSD of machester format.

Let us assume that symbol '0' & '1' occur with equal probabilities.



i.e $P[A_k=0) = 1/2$

$P[A_k=1) = 1/2$

In Machester format

$$A_k = \begin{cases} +a \\ -a \end{cases}$$ for symbol 1

$$\begin{cases} -a \\ +a \end{cases}$$ for symbol 0.

Autocorrelation function $R_A(n)$

$$R_A(n) = E[A_k \cdot A_{k-n}]$$

A_k	$A_{k-0} = A_k$	A_k^2	Equally probable
0 $\begin{cases} -a \\ +a \end{cases}$	0 $\begin{cases} -a \\ +a \end{cases}$	a^2	1/2
1 $\begin{cases} +a \\ -a \end{cases}$	1 $\begin{cases} +a \\ -a \end{cases}$	a^2	1/2

Case (i) For $n=0$,

$$R_A(0) = E[A_k^2]$$

$$= a^2 \times \frac{1}{2} + a^2 \times \frac{1}{2}$$

$$\langle R_A(0) = a^2 \rangle$$

Case (ii) For $n \neq 0$

$$R_A(n) = a^2(1/4) - a^2(1/4) - a^2(1/4) + a^2(1/4)$$

$$R_A(n) = 0$$

A_k	A_{k-1}	$A_k A_{k-1}$	Equally probable
0 $\begin{matrix} -a \\ +a \end{matrix}$	0 $\begin{matrix} -a \\ +a \end{matrix}$	a^2	1/4
0 $\begin{matrix} -a \\ +a \end{matrix}$	1 $\begin{matrix} +a \\ -a \end{matrix}$	$-a^2$	1/4
1 $\begin{matrix} +a \\ -a \end{matrix}$	0 $\begin{matrix} -a \\ +a \end{matrix}$	$-a^2$	1/4
1 $\begin{matrix} +a \\ -a \end{matrix}$	1 $\begin{matrix} +a \\ -a \end{matrix}$	a^2	1/4

We define $R_A(n)$ as

$$R_A(n) = \begin{cases} a^2 & \text{for } n=0 \\ 0 & \text{for } n \neq 0 \end{cases}$$

Pulse shaping function of machester format is

$$V(f) = jT_b \operatorname{sinc}(fT_b/2)\sin(\pi fT_b/2)$$

$$|V(f)|^2 = T_b^2 \operatorname{sinc}^2(fT_b/2)\sin^2(\pi fT_b/2)$$

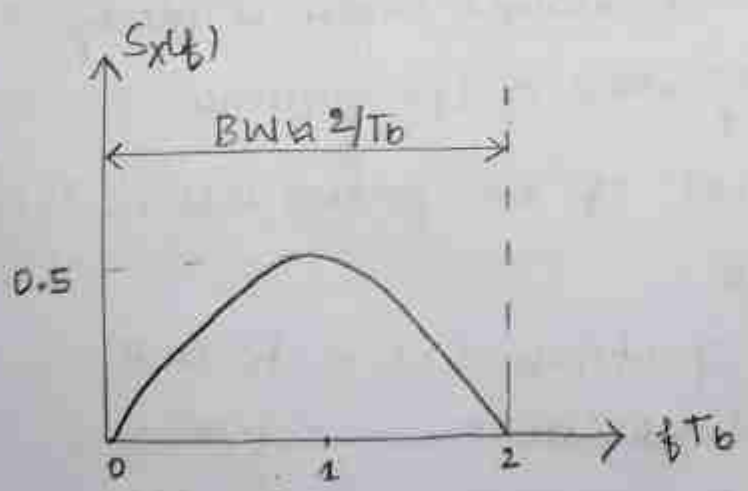
the PSD for manchester format is

$$S_X(f) = \frac{1}{T_b}|V(f)|^2 \sum_{n=-\infty}^{\infty} R_A(n)\, e^{-j2\pi fnT_b}$$

$$= \frac{1}{T_b} T_b^2 \operatorname{sinc}^2(fT_b/2)\sin^2(\pi fT_b/2)\left[\underset{n=0}{R_A(0)} + \underset{n>1}{0}\right]$$

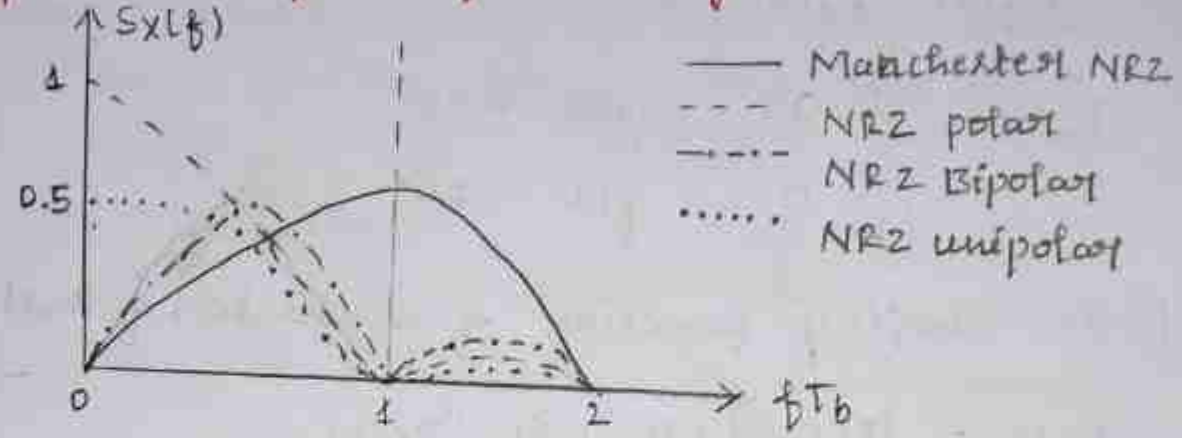
$$= T_b \operatorname{sinc}^2(fT_b/2)\sin(\pi fT_b/2)\cdot(a^2)$$

$$S_X(f) = a^2 T_b \operatorname{sinc}^2(fT_b/2)\sin(\pi fT_b/2)$$



Normalised power spectrum of Machester format

Comparision of PSD for Binary line codes.



Comparision of PSD for binary line codes

Unipolar NRZ : The null bandwidth (the width of the main lobe) of NRZ unipolar format is $1/T_b$. The presence of Dirac delta function amounts for the waste of power due to dc level and fact that the spectrum is not approaching zero near dc

polar NRZ : Most of the signal power lies in the null bandwidth of $1/T_b$. It has the disadvantage of having a large PSD near dc ($f=0$). Polar signalling is also transparent because there is always some (+ve or -ve) pulse regardless of bit sequence.

Bipolar NRZ : Most of the power lies in the null bandwidth of $1/T_b$

Advantages : 1) Its spectrum has a dc null

2) Its bandwidth is not excessive

3) It has single error detection capability

Disadvantages : 1) Requires twice as much power as a polar signal

2) It is not transparent.

Machester format : The null bandwidth is twice that of all other formats. However, Machester code has a zero dc level on a bit-by-bit basis.

High density bipolar signalling

The HDB scheme is an [ITU International telecommunication union] standard.

In this scheme, the problem of non transparency in bipolar signalling is eliminated by adding pulses. When the number of consecutive zeros (0) exceeds N, such a modified code is designated as high density bipolar coding [HDBN].

where N take any value 1, 2, 3. . . . . .

The most important of the HDB codes is HDB_3 format which has been adopted as an international standard.

The basic idea of the HDBN code is that when a run of N+1 zeros occurs, this groups of zeros is replaced by one of the special N+1 binary digit sequences

HDB3 format

HDB3 line coding is used in all levels of the European E-carrier system.

HDB_3 code replaces any instances of 4 [N+1 = 3+1] consecutive zeros with one of the patterns "000V" or "B00V". The choice is made to ensure that consecutive

violations are of differing polarity

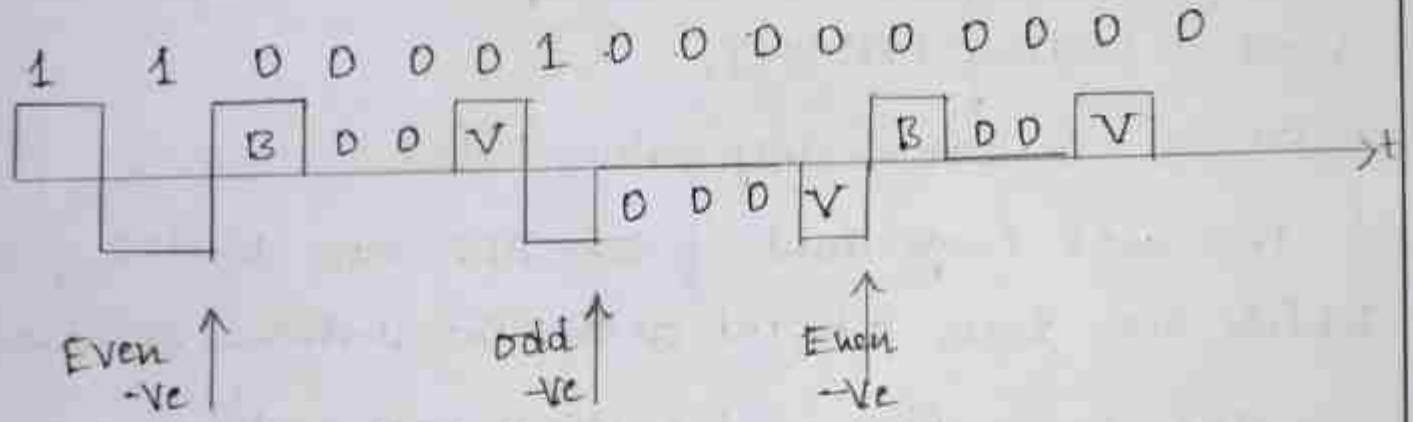
B=1 conforms to the bipolar rule [Balancing bit]

V=1 violates the bipolar rule [violating bit]

HDB3 coding

Number of +Ve and -Ve pulses since last substitution [V]	Pattern	polarity of previous pulse	Coded
Odd	000V	+	000+
		-	000-
Even	B00V	+	-00-
		-	+00+

Example



B3ZS format.

B3ZS format replaces any instances of three consecutive zeros with one of the pattern "000V" OR "B00V".

Everytime when there are 3 consecutive 0's in the code, they will be replaced by either 00-, 00+, -0- or +0+.

To determine which pattern to use one must count the number of positive and negative pulses since the last violation bit V.

If the result is odd number, then 00- or 00+ is used

If the result is even number then +0+, or -0- is used

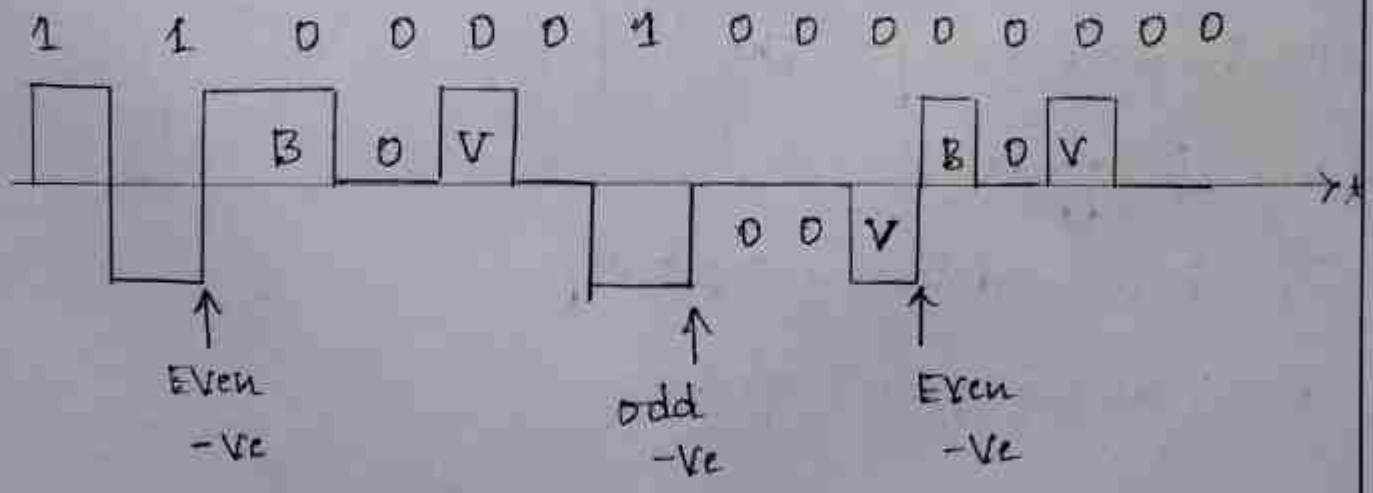
To determine which polarity to use, we look at the pulse preceeding the 3 zeros.

If 00V form must be used then V simply copies the polarity of last pulse.

If B0V form must be used then B and V chosen will have the opposite polarity of the last pulse.

Number of +ve or -ve pulses since last V	pattern	polarity of previous pulse	Coded
odd	00V	+	00+
		-	00-
Even	B0V	+	-0-
		-	+0+

Example.

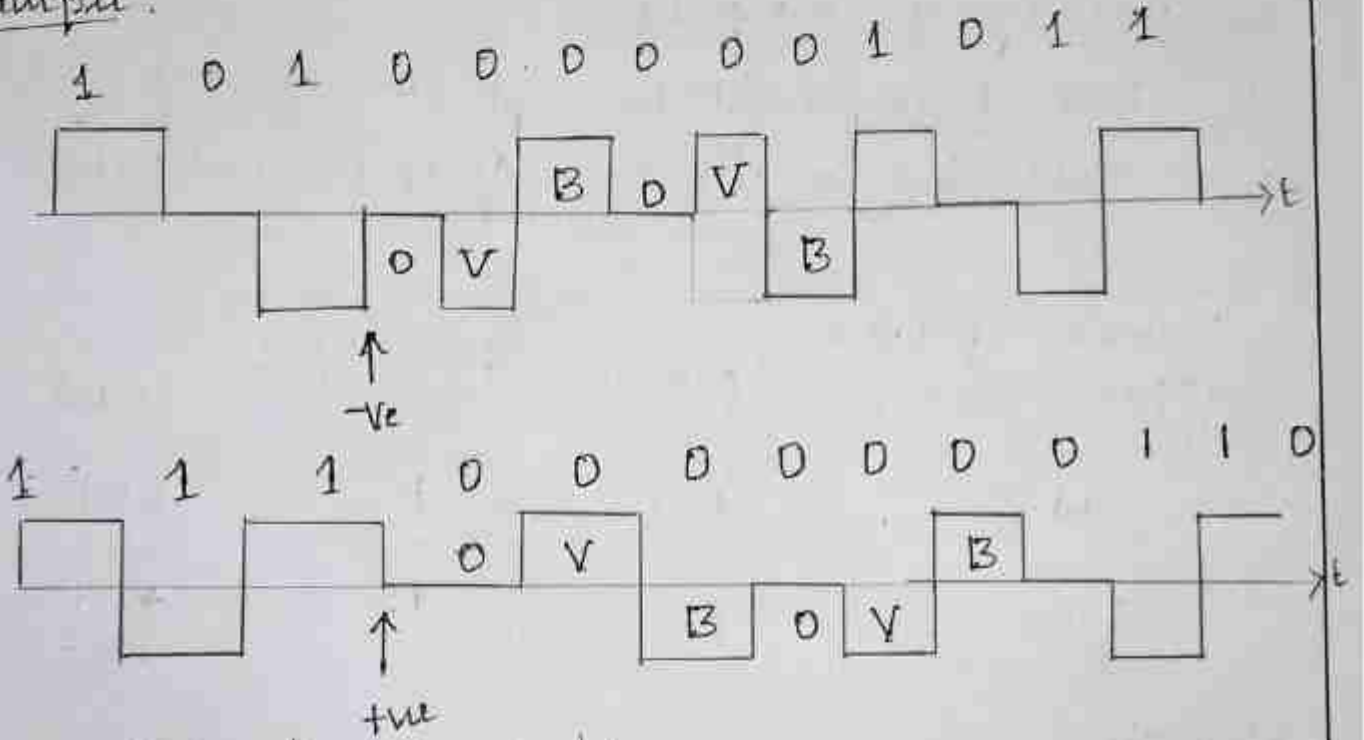


B6ZS format:

In B6ZS format, a string 6 zeros are replaced by the pattern OVBOVB. The polarity of the bits V & B are chosen based on the polarity of the previous pulse. ([0+-0-+] or [0-+0+-])

North American T_2 rate [6.312 Mbit/s]

Example:



Properties of Line codes

1) Transparency:

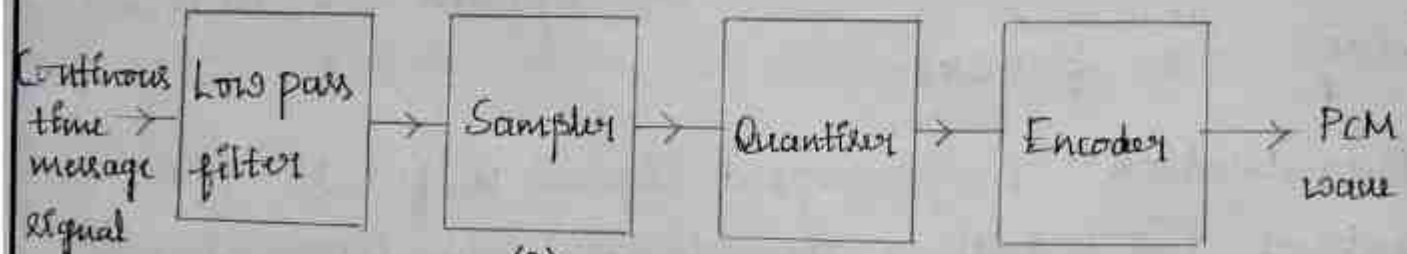
2) Ruggedness

3) Error detection capability

4) Matched power spectrum

Pulse code modulation.

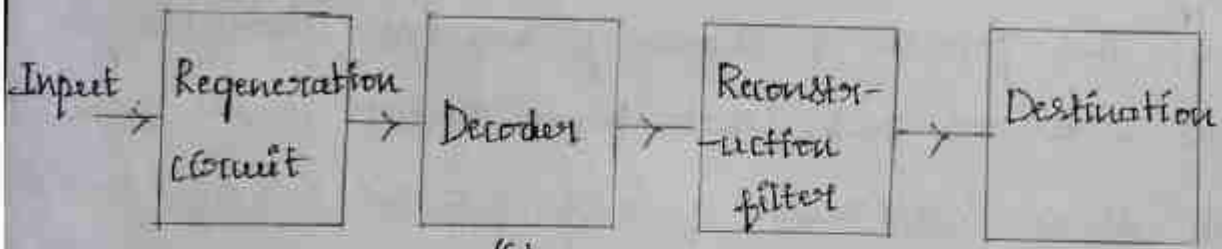
Pulse code modulation is an analog to digital converter where the information contained in the instantaneous samples of an analog signal are represented by digital codes in a serial bit stream manner



(a)



(b)



(c)

Eg: Basic elements of a PCM system

(a) Transmitter (b) Transmission path (c) Receiver

The Block diagram of PCM system consists of

a) Transmitter

b) Regenerative repeater

c) Receiver

1) The Low pass filter [pre-alias filter] is used before Sampler in order to limit the frequency greater than 'W' Hz. Hence message signal is bandlimited to W Hz.

2) Sampler : The Incoming message signal is sampled with a train of Narrow rectangular pulses. The sampling rate 'f_s' is selected above Nyquist rate to avoid aliasing i.e $f_s \geq 2W$

3) Quantisation : The sampled signal is fed to the Quantizer. The Quantizer approximates each input signal level to the nearest prefixed level.

The output of Quantizer is discrete time discrete Value signal known as Quantized signal.

4) Encoding : The Quantized samples are then encoded in the encoder. The process of encoding involves allocating some digital code to each level. These coded levels are transmitted as a bitstream of data i.e 0's & 1's.

The Encoder output consists of pulses depending on the code combination.

5) Regenerative repeater : The PCM signal is reconstructed by means of regenerative repeater located at sufficiently closed spacing along the transmission path.

The regenerative networks are used at intermediate points between transmitter and receiver in order to boost up the pulse amplitude.

6) Decoder: The first operation in the receiver is to generate the required pulses.

The decoder converts binary coded signal to a approximated pulses of discrete magnitude.

7) Reconstruction filter: The final operation in the receiver is to recover the original analog signal. This is done by passing the decoder output through a LPF.

The output of LPF is an analog signal.

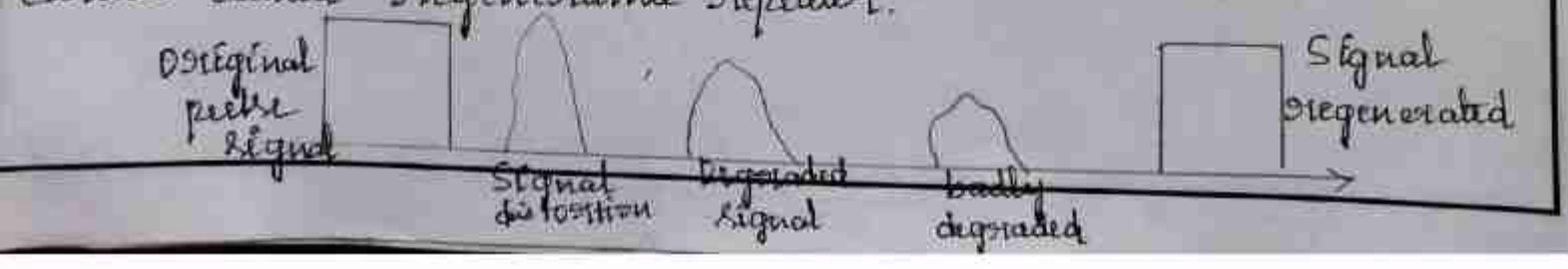
Advantages

1) Relatively inexpensive digital circuitry is involved in PCM
2) PCM signals can be multiplexed & transmitted over a common high speed communication link.
3) In long distance communication, clean waveforms can be regenerated using repeaters.
4) The noise performance of digital system is superior to that of an analog system.

Pulse degradation

The shape of the pulse is affected mainly by two mechanisms
1) Unwanted electrical noise or other disturbance.
2) Nonideal transfer function of the transmission medium

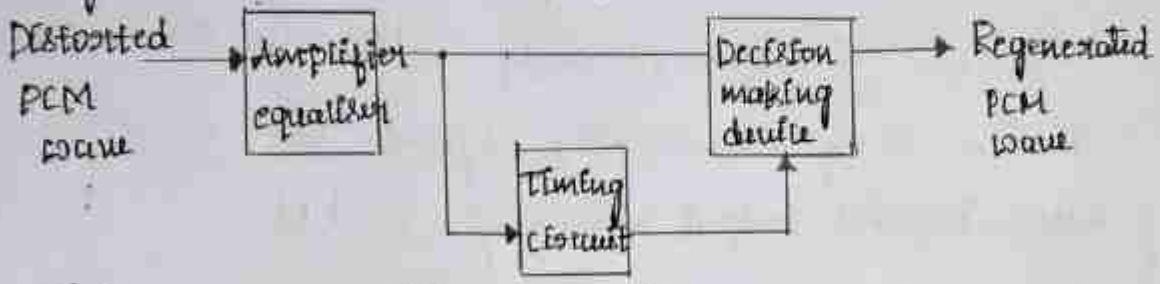
The shape of the pulse degraded as a function of the distance travelled. This degraded pulse can be regenerated by using circuit called regenerative repeater.



Regenerative repeater

The regenerative repeater is used to remove noise from the incoming PCM data.

Regenerative repeaters are installed at regular interval along the channel through which PCM signal is transmitted



The regenerative repeater performs three basic operations
1) Equalization 2) Timing 3) Decision making

Equalizer: The equalizer shapes the received pulse so as to compensate for the effect of amplitude and phase distortion produced by the transmission channel.

2) Timing circuitry: The Timing circuit provides a periodic pulse train derived from the received pulse.

The output of amplifier is sampled at different sampling instants where signal to noise ratio is maximum.

3) Decision making device: The sample extracted is compared with a predetermined threshold [λ] in the decision making device. During each bit interval, a decision is made whether the received symbol is a '1' or '0'.

If the threshold is exceeded, a clean new pulse representing symbol '1' transmitted, otherwise a clean base line representing '0' is transmitted.

Quantization

Quantization is a process of representing the sampled value of the amplitude by a finite set of levels. It means converting a continuous-amplitude sample in to a discrete time signal.

* The Quality of a Quantizer output depends upon the number of quantization levels used.
* The discrete amplitudes of the quantized output are called as representation levels
* The spacing between two adjacent representation levels is known as step size.
* There are two types of Quantization.
  1) Uniform Quantization
  2) Non uniform Quantization.

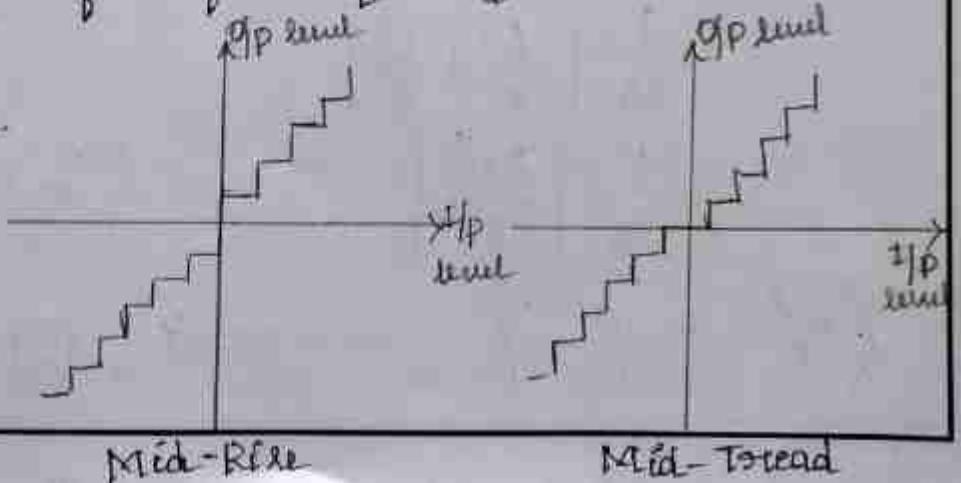
Quantization in which the quantization levels are uniformly spaced is termed as Uniform quantization.

Quantization in which quantization levels are unequal and relation between them is logarithmic is termed as Non-uniform Quantization

Uniform quantization.

There are two types of uniform quantization.

- Mid-Rise type
- Mid-tread type.



Quantization noise and Signal to Noise ratio in PCM system

Derive the equation for Signal to quantization noise ratio if probability of overload is less than 10^{-4} in the case of a uniform quantizer. Further, if a binary code of n-bits ($n > 6$), write the equation for (SNR).

>

Let the random variable 'Q' denotes the quantization error and 'q' its sample value.

Let us assume that the quantization error Q is uniformly distributed over a single quantizer interval 'Δ'.

Probability density function [PDF] of quantization error is

$$f_Q(q) = \begin{cases} 1/\Delta & \text{for } -\Delta/2 \le q \le \Delta/2 \\ 0 & \text{otherwise.} \end{cases}$$

The variance of quantization error is

$$\sigma_q^2 = \int_{-\Delta/2}^{\Delta/2} (q-m)^2 f_Q(q)\, dq$$

Mean quantization error, $m = 0$

$$= \int_{-\Delta/2}^{\Delta/2} (q-0)^2 \, 1/\Delta \cdot dq$$

$$= 1/\Delta \int_{-\Delta/2}^{\Delta/2} q^2 \cdot dq = \frac{1}{\Delta}\left[\frac{q^3}{3}\right]_{-\Delta/2}^{\Delta/2} = \frac{1}{3\Delta}\left[\left(\frac{\Delta}{2}\right)^3 - \left(\frac{-\Delta}{2}\right)^3\right]$$

$$= \frac{1}{3\Delta}\left[\frac{\Delta^3}{8} + \frac{\Delta^3}{8}\right]$$

$$= \frac{1}{3\Delta}\left[\frac{2\Delta^3}{8}\right]$$

$$\left\langle \sigma_q^2 = \frac{\Delta^2}{12} \right\rangle \longrightarrow (2)$$

The signal to quantization noise ratio [SNR] is given by

$$\frac{S}{N} = \frac{\text{Normalized Signal power}}{\text{Normalized Noise power.}} = \frac{\sigma_x^2}{\sigma_q^2}$$

$$= \frac{\sigma_x^2}{\Delta^2/12}$$

$$(SNR) = \frac{12\,\sigma_x^2}{\Delta^2} \quad \text{——— (3)}$$

Let x_{max} denote the absolute value of the overload level of the quantizer & Δ denotes its step size.

The number of levels in the quantizer is given by

$$L = 1 + \frac{2x_{max}}{\Delta} \quad \text{——— (4)}$$

Since the number of levels, L is odd for mid-tread quantizer,

$$L = 2^n - 1 \quad \text{——— (5)}$$

By equating eq^n

$$2^n - 1 = 1 + \frac{2x_{max}}{\Delta}$$

$$2^n = 2 + \frac{2x_{max}}{\Delta}$$

$$2^n = 2\left[1 + \frac{x_{max}}{\Delta}\right]$$

$$\frac{2^n}{2} = 1 + \frac{x_{max}}{\Delta}$$

$$2^n 2^{-1} = 1 + \frac{x_{max}}{\Delta}$$

$$2^{n-1} - 1 = \frac{x_{max}}{\Delta}$$

$$\left\langle \Delta = \frac{x_{max}}{2^{n-1} - 1} \right\rangle \quad \text{——— (6)}$$

If the probability of overload is $< 10^{-4}$, $x_{max} = 4\sigma_x$

$$\Delta = \frac{4\sigma_x}{2^{n-1} - 1}$$

We know that

$$(SNR)_0 = \frac{12\sigma_x^2}{\Delta^2}$$

$$(SNR)_0 = \frac{12\sigma_x^2}{\left(\frac{4\sigma_x}{2^{n-1}-1}\right)^2} = \frac{12\sigma_x^2}{\frac{16\sigma_x^2}{(2^{n-1}-1)^2}}$$

$$\left\langle (SNR)_0 = \frac{3}{4}\left[2^{n-1} - 1\right]^2 \right\rangle \quad \text{——————} \quad (7)$$

$$(SNR_0) = \frac{3}{4}\left[2^n \cdot 2^{-1} - 1\right]^2$$

$$= \frac{3}{4 \times 2}\left[2^n - 1\right)^2 = \frac{3}{16}\left[2^n - 1\right)^2$$

For largest n i.e $n > 6$, we can neglect 1 i.e $2^n - 1 = 2^n$

$$(SNR)_0 = \frac{3}{16} 2^{2n}$$

$$(SNR)_{dB} = 10 \log_{10}[3/16) + 10 \log_{10}(2^{2n})$$

$$= -7.2 + 20n \log_{10}(2)$$

$$= -7.2 + 6.02n$$

$$(SNR)_{dB} = 6.02n - 7.2.$$

A signal x(t) is uniformly distributed in the range ± x_{max}. Calculate signal to Noise ratio for pulse code modulation of this signal.

> We know that.

$$\left\langle SNR = \frac{\sigma_x^2}{\sigma_q^2} \right\rangle \quad \text{——— (1)}$$

The PDF of x(t) is

$$f_X(x) = \begin{cases} 1/2x_{max} & \text{for } -x_{max} \le x \le +x_{max} \\ 0 & \text{otherwise.} \end{cases}$$

* The mean square value of x(t) is

$$\sigma_x^2 = \int_{-x_{max}}^{x_{max}} [x-u]^2 f_X(x).\,dx \qquad u = 0$$

$$= \int_{-x_{max}}^{+x_{max}} [x-0]^2 \frac{1}{2x_{max}} .\,dx$$

$$= \frac{1}{2x_{max}} \int_{-x_{max}}^{x_{max}} x^2.\,dx = \frac{1}{2x_{max}} \left[\frac{x^3}{3}\right]_{-x_{max}}^{x_{max}} = \frac{1}{2x_{max} \times 3} \left[x_{max}^3 - (-x_{max})^3\right]$$

$$\sigma_x^2 = \frac{1}{6x_{max}} \cdot 2x_{max}^3$$

$$\left\langle \sigma_x^2 = \frac{x_{max}^2}{3} \right\rangle \quad \text{——— (2)}$$

The PDF of Quantization error 'q' is

$$f_Q(q) = \begin{cases} 1/\Delta & -\Delta/2 \le q \le \Delta/2 \\ 0 & \text{otherwise} \end{cases}$$

The variance of quantization error is

$$\sigma_q^2 = \int_{-\Delta/2}^{\Delta/2} (q-\mu)^2 f_Q(q)\, dq \qquad \mu = 0$$

$$= \int_{-\Delta/2}^{\Delta/2} (q-0)^2 \cdot \frac{1}{\Delta}\, dq$$

$$= \frac{1}{\Delta} \int_{-\Delta/2}^{\Delta/2} q^2 \cdot dq = \frac{1}{\Delta}\left[\frac{q^3}{3}\right]_{-\Delta/2}^{\Delta/2} = \frac{1}{3\Delta}\left[\left(\frac{\Delta}{2}\right)^3 - \left(\frac{-\Delta}{2}\right)^3\right]$$

$$= \frac{1}{3\Delta}\left[\frac{\Delta^3}{8} + \frac{\Delta^3}{8}\right]$$

$$= \frac{1}{3\Delta}\left[\frac{2\Delta^3}{8_4}\right]$$

$$\left\langle \sigma_q^2 = \frac{\Delta^2}{12} \right\rangle \quad \text{---------- (3)}$$

Substituting equation (2) and (3) in equation (1), we get.

$$SNR = \frac{\sigma_x^2}{\sigma_q^2}$$

$$= \frac{x_{max}^2/3}{\Delta^2/12} = \frac{x_{max}^2}{3_1} \times \frac{12_4}{\Delta^2} = \frac{4x_{max}^2}{\Delta^2}$$

$$\left\langle SNR = \frac{4x_{max}^2}{\Delta^2} \right\rangle \quad \text{---------- (4)}$$

we know that, $\Delta = \frac{2x_{max}}{M}$ ——→ (5)

Substituting equation (5) in equation (4).

$$SNR = \frac{4x_{max}^2}{\left(\frac{2x_{max}}{M}\right)^2} = \frac{4x_{max}^2}{\frac{4x_{max}^2}{M^2}} = M^2$$

$$\langle SNR = M^2 \rangle$$

We know that $M = 2^N$

$$(SNR) = (2^N)^2$$

$$\langle (SNR) = 2^{2N} \rangle$$

$$(SNR)_{dB} = 10 \log_{10} [2^{2N}]$$

$$= 2N . 10 \log_{10} (2)$$

$$= 20N (0.30)$$

$$= N \times 2.0 \times 0.3$$

$$\langle (SNR)_{dB} = 6N \rangle$$

Bandwidth of PCM Signal

* The quantizer uses 'N' number of binary digits to represent each level

∴ The number of levels that can be represented by 'N' digits will be 'L'. i.e $L = 2^N$

Eg: If $N = 3$, then total number of levels L

$L = 2^3 = 8$ levels.

* The number of bits per second is called Signalling rate of PCM & is denoted by 'R' i.e $R = Nf_s$ bits/sec

* Bandwidth needed for PCM transmission will be given by half of the signalling rate.

∴ Transmission bandwidth of PCM is

$$B_T \geq \frac{1}{2} R$$

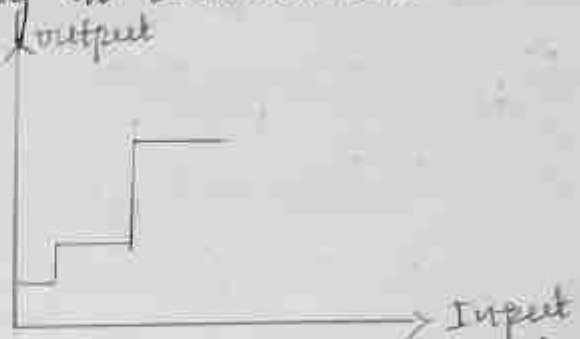
$$B_T \geq \frac{1}{2} Nf_s \qquad \langle B_T \geq NW \rangle$$

$$B_T \geq \frac{1}{2} . N . 2W$$

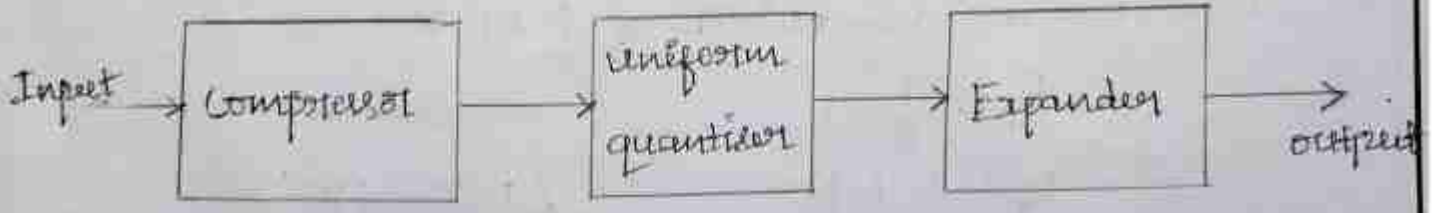
Non-uniform quantization or Robust quantization.

In Non-uniform quantization, the step size is not fixed. It varies according to certain law or as per signal amplitude.

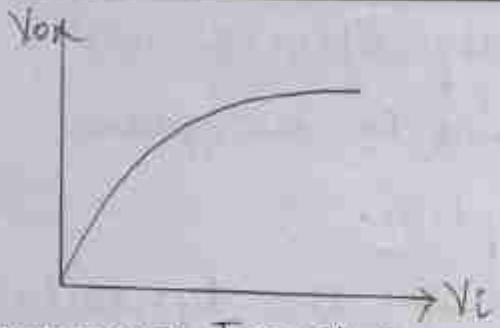




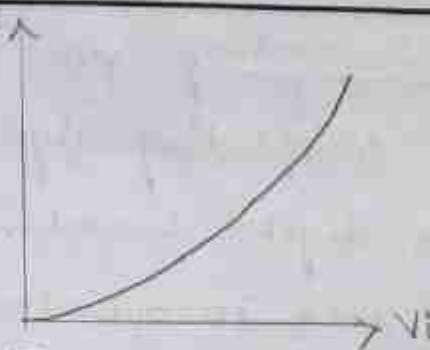
* The step size is small at low input signal levels. Hence quantization error is also small at these inputs. Therefore Signal to quantization noise [SNR] power ratio is improved at low signal levels.
* The step size is highest at high input levels. Hence Signal to noise [SNR] power ratio remains same throughout the dynamic range of quantizer.
* Maintaining constant Signal to noise power ratio [SNR] throughout the signal range is called Robust quantization.
* To achieve constant SNR, the signal is passed through a combination of compressor Expander circuit respectively at the transmitter & receiver. This process of compressing and expanding is called companding.



Model of Non-uniform quantizer



Compressor Transfer characteristics



Expander Transfer characteristics.

* To achieve robust quantisation, the signal is passed through a network which has an Input-output characteristics as shown in above fig.
* The signal is changed such that small amplitude signals are boosted up without altering the maximum amplitude of the signal, Small amplitude signals range through more quantisation levels.
* Any signal when passed through such a network gets compressed leading to signal distortion.
* To remove this distortion, the signal is passed through an Inverse network at the receiver called as Expander.
* The complete process of compressing and expanding the signal is referred to as companding

There are two types of companding

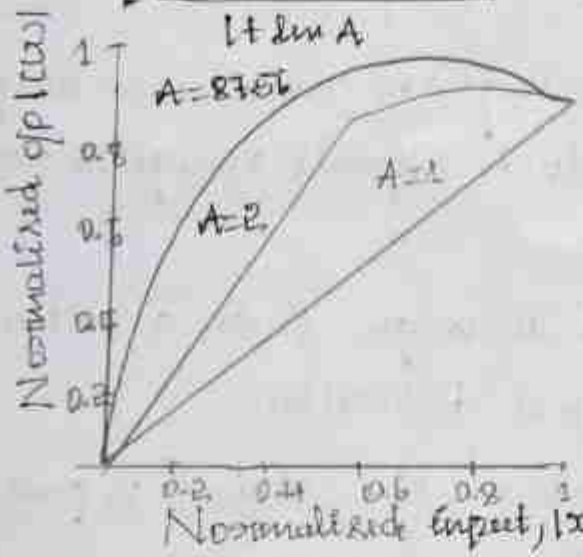
1) A-law companding
2) μ-law companding.

A-law companding: An A-law algorithm is a standard companding algorithm, used in European 8-bit PCM digital communication system.

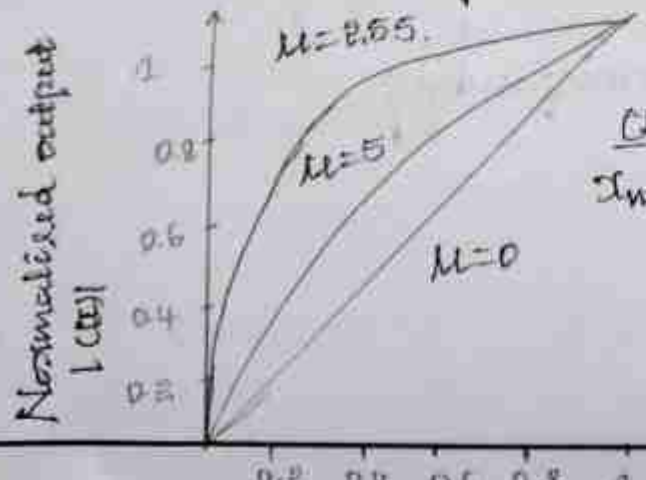
* It is used to optimize, i.e modify the dynamic range of an analog signal for digitizing.

A law is defined by following equation.

$$\frac{C(|x|)}{x_{max}} = \begin{cases} \frac{A|x|/x_{max}}{1+\ln A} & 0 \le \frac{|x|}{x_{max}} \le \frac{1}{A} \\ \frac{1+\ln(A|x|/x_{max})}{1+\ln A} & \frac{1}{A} \le \frac{|x|}{x_{max}} \le 1 \end{cases}$$



U law companding: The u-law algorithm is a companding algorithm, primarily used in 8-bit PCM digital telecommunication system in North America and Japan. It reduces the dynamic range of an audio signal.

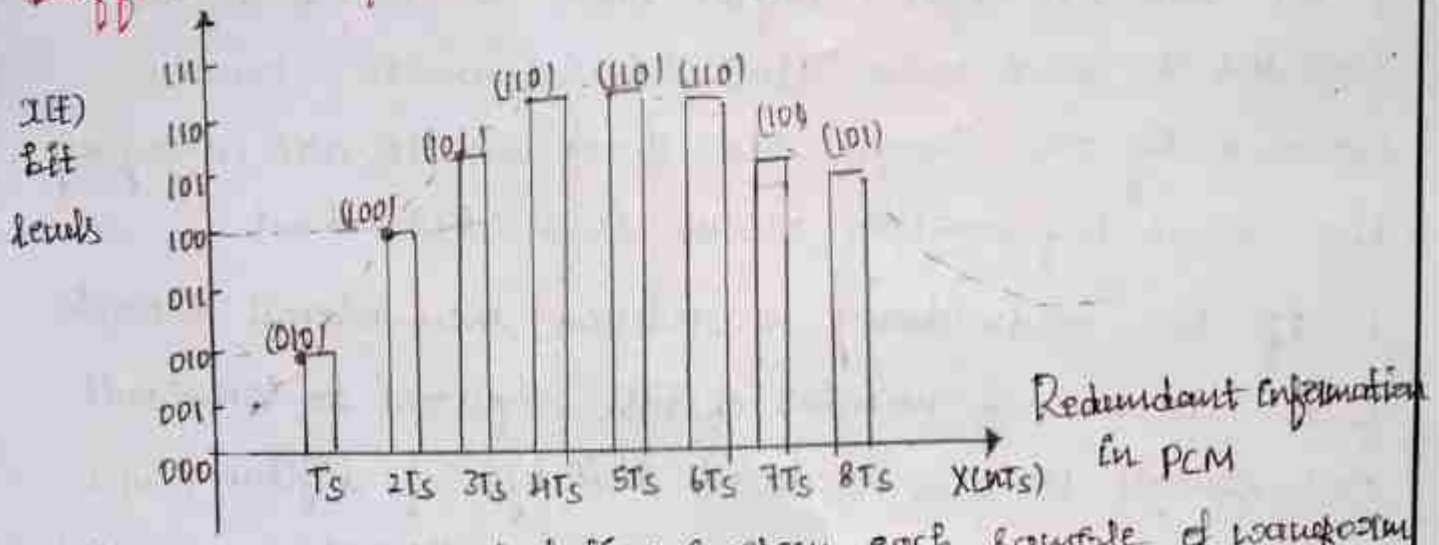


$$\frac{C(x)}{x_{max}} = \frac{\ln(1+\mu|x|/x_{max})}{\ln(1+\mu)} \qquad 0 \le \frac{|x|}{x_{max}} \le 1$$

Advantages of Non-uniform quantization.

1) Reduced quantization noise
2) High average SNR.

Differential pulse code modulation.



* In pulse code modulation system, each sample of waveform is encoded independently of other samples.

* The samples of signals are highly correlated, this is because
  i) Signal does not change fast.
  ii) The samples above the Nyquist rate i.e $f_s > 2W$

  Thus, the signal does not change rapidly from one sample to the next sample.

* When these highly correlated samples are encoded, the resulting encoded signal contains Redundant information. [duplicated information]

* The redundancy can be eliminated by DPCM.

* The fig(1) shows the continuous time signal x(t) Sampled used flattop sampling. This signal is sampled at the

instants, $T_s, 2T_s, 3T_s, \ldots\ldots nT_s$. The Sampling frequency is selected higher than the Nyquist rate & encoded using 3-bit encoder. The Samples quantized to nearest level is shown in fig4)

* We can see that Samples taken at $4T_s, 5T_s, 6T_s$ are encoded to same value "110". this information can be carried by one Sample. But three samples are carrying the same information means it is Redundant.

* If the redundancy is reduced, then overall bit rate will decrease & number of bits required to transmit one sample is also reduced. This type of digital pulse code modulation scheme is known as DPCM[Differential pulse code modulation].

Differential pulse code modulation

* When an analog signal is sampled at a rate slightly higher than the Nyquist rate, resulting high correlated samples. i.e the signal does not change rapidly from one sample to next.

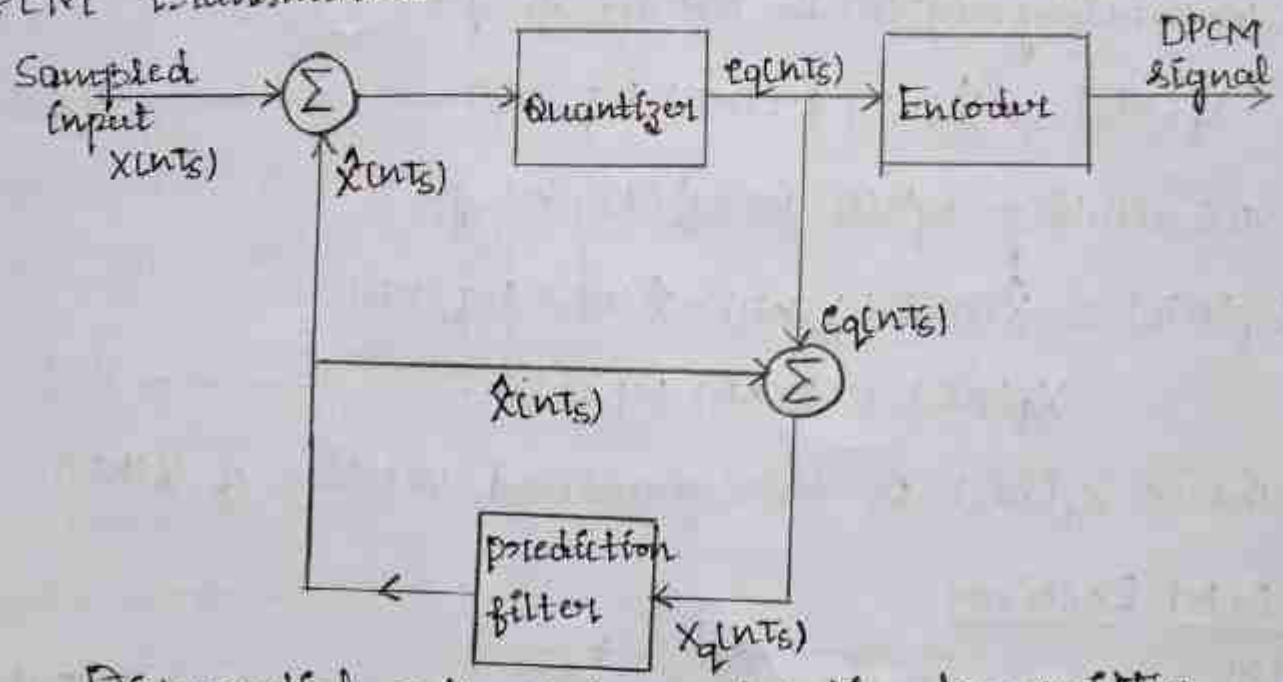
* If these highly correlated samples are encoded, the resulting encoded signal contains redundant information.

* If we remove this redundancy before encoding, efficiency of the coded signal can be increased.

* DPCM works on the principle of prediction. The value of the present sample is predicted from the

past values. The prediction may not exact but it is very clse to the actual sample value.

DPCM transmitter



Differential pulse code modulation transmitter.

* X(nTs) represents the sampled version of the analog signal x(t) [shown in fig (1)]

* The output of the comparator is the difference between the sampled input and prediction of its, $\hat{x}(nT_s)$

i.e $e(nT_s) = x(nT_s) - \hat{x}(nT_s)$

where $\hat{x}(nT_s)$ is the prediction of $x(nT_s)$

$e(nT_s)$ is the prediction error.

* The prediction error $e(nT_s)$ is the quantized to produce $e_q(nT_s)$

* In the quantizer, the noise $q_e(nT_s)$ gets added

∴ The o/p of the quantizer can be written as

$$e_q(nT_s) = e(nT_s) + q_e(nT_s)$$

The Input to the prediction filter is written as

$$X_q(nT_s) = \hat{X}(nT_s) + e_q(nT_s) \quad \text{---------- (3)}$$

Substituting eq^n[2] in eq^n[3], we get

$$X_q(nT_s) = \hat{X}(nT_s) + e(nT_s) + q_e(nT_s) \quad \text{---------- (4)}$$

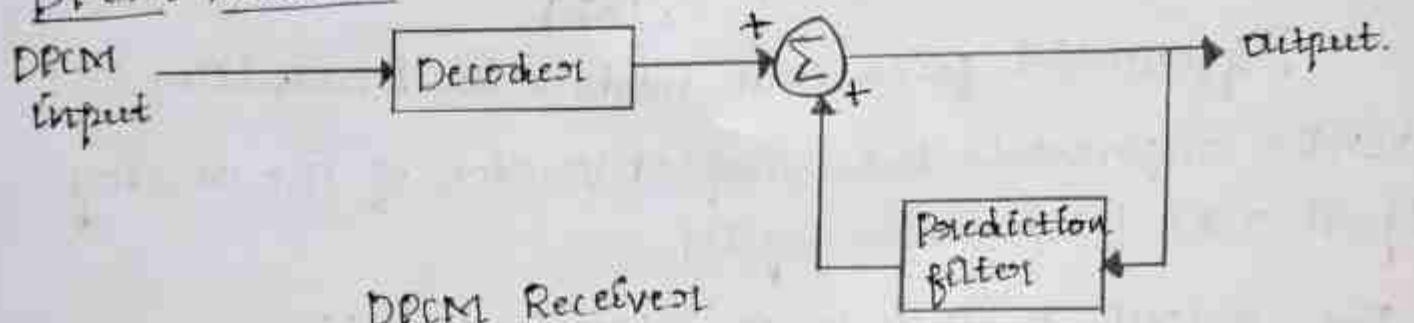
Substituting eq^n(1) in eq^n(4), we get

$$X_q(nT_s) = \hat{X}(nT_s) + X(nT_s) - \hat{X}(nT_s) + q_e(nT_s)$$

$$X_q(nT_s) = X(nT_s) + q_e(nT_s) \quad \text{---------- (5)}$$

Where $X_q(nT_s)$ is the quantised version of $X(nT_s)$

DPCM Receiver



DPCM Receiver

* The decoder reconstructs the quantized error signal from incoming binary signal.
* The prediction filter output & quantised error signals are summed up to give the quantized version of the original signal

$$\langle e_q(nT_s) + \hat{X}(nT_s) = X_q(nT_s) \rangle$$

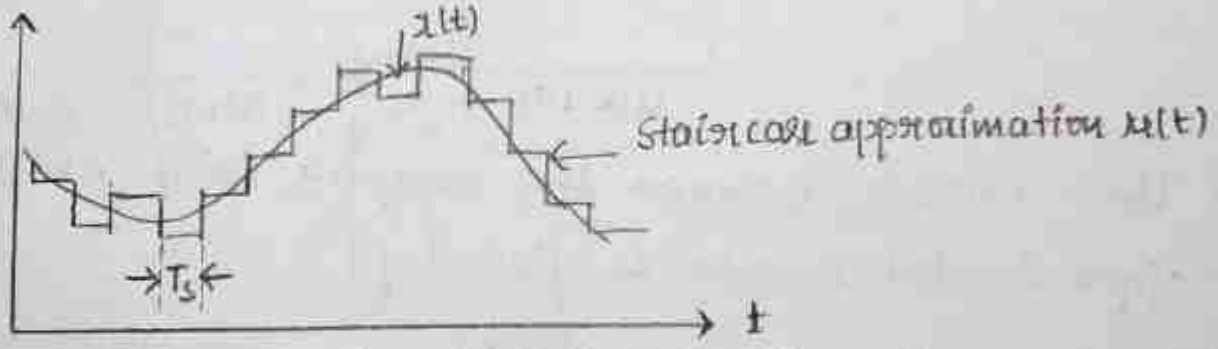
* Thus, the signal at the receiver differs from actual signal by quantization error $q_e(nT_s)$, which is introduced permanently in the reconstructed signal.

Delta modulation.

The Sampling rate of a signal should be higher than the Nyquist rate, to achieve better sampling. If the sampling interval in Differential PCM is reduced considerably, the sample to sample amplitude difference is very small, as if the difference is 1-bit quantization, then the stepsize will be very small. i.e Δ [delta]

The modulation, where the sampling rate is much higher and in which the stepsize after quantization is smallest value, such modulation is termed as Delta modulation.

Delta modulation transmits only one bit per sample. i.e the present sample value is compared with previous sample value & the indication, whether the amplitude is increased or decreased is sent.

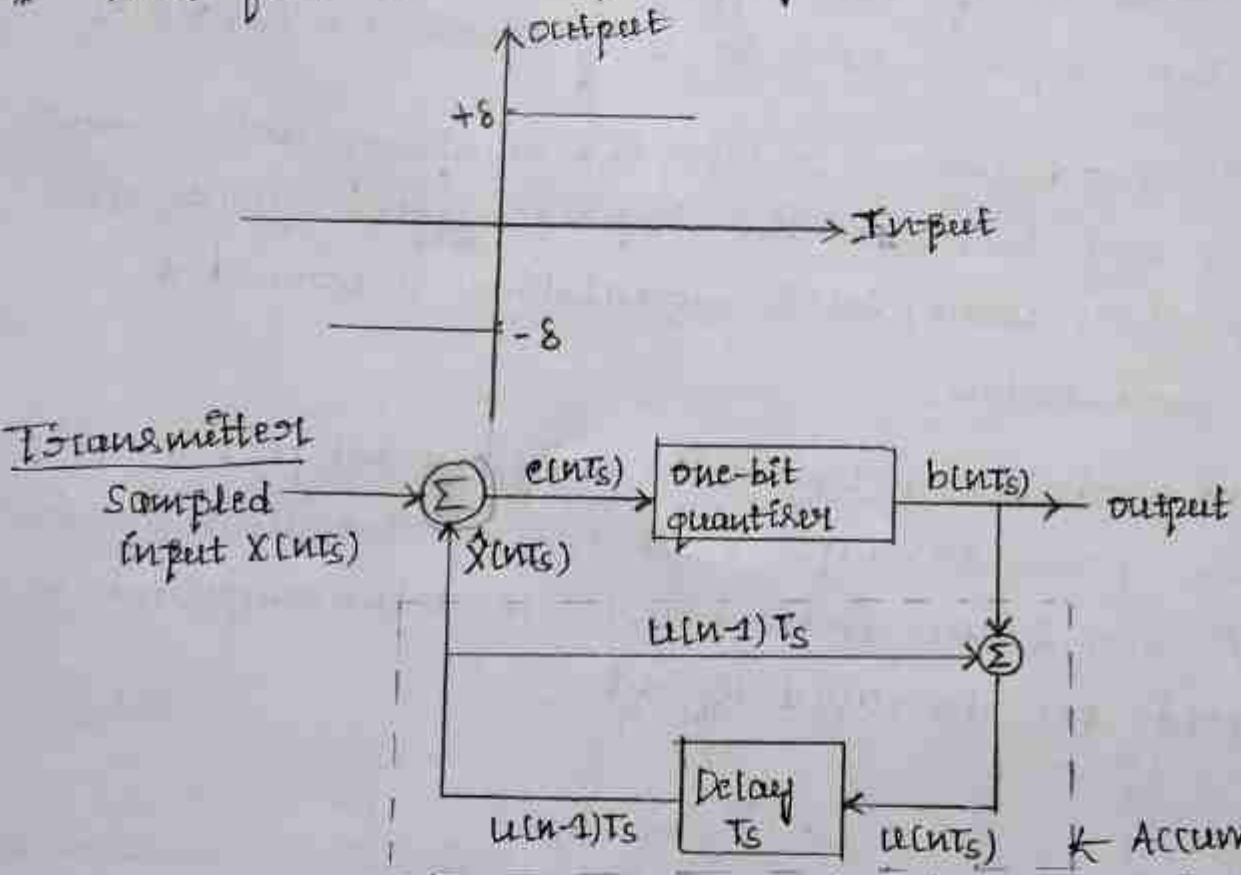


Binary sequence at modulator output 0 0 1 0 1 1 1 1 0 1 0 0 0 0

1) The Input signal x(t) is approximated to step signal by the delta modulator.

2) the difference between input signal x(t) and staircase approximated signal is quantized into only two levels i.e +δ or -δ.

* If the difference is +ve, then approximated signal is increased by one step .i.e +δ & bit 1 is transmitted.
* If the difference is -ve, then approximated signal is reduced by one step i.e -δ & bit 0 is transmitted.
* Thus for each sample only one bit is transmitted.



The error between the sampled value x(nTs) & last approximated sample is given by

$$e(nT_s) = x(nT_s) - \hat{x}(nT_s) \quad \text{---------- (1)}$$

Let u(nTs) be the present sample approximation of staircase output.

From the figure : $\hat{x}(nT_s) = u(n-1)T_s$

$$\hat{x}(nT_s) = u(nT_s - T_s) \quad \text{---------- (2)}$$

Substituting eq^n(2) in eq^n(1), we get

$$e(nT_s) = x(nT_s) - u(nT_s - T_s) \quad \text{---------- (3)}$$

* The binary quantity $b(nT_s)$ is the algebraic sign of the error $e(nT_s)$, except for the scaling factor δ

i.e $b(nT_s) = \delta \, Sgn[e(nT_s)]$ ——— (4)

$b(nT_s)$ depends on the sign of error $e(nT_s)$, the sign of step size 'δ' will be decided.

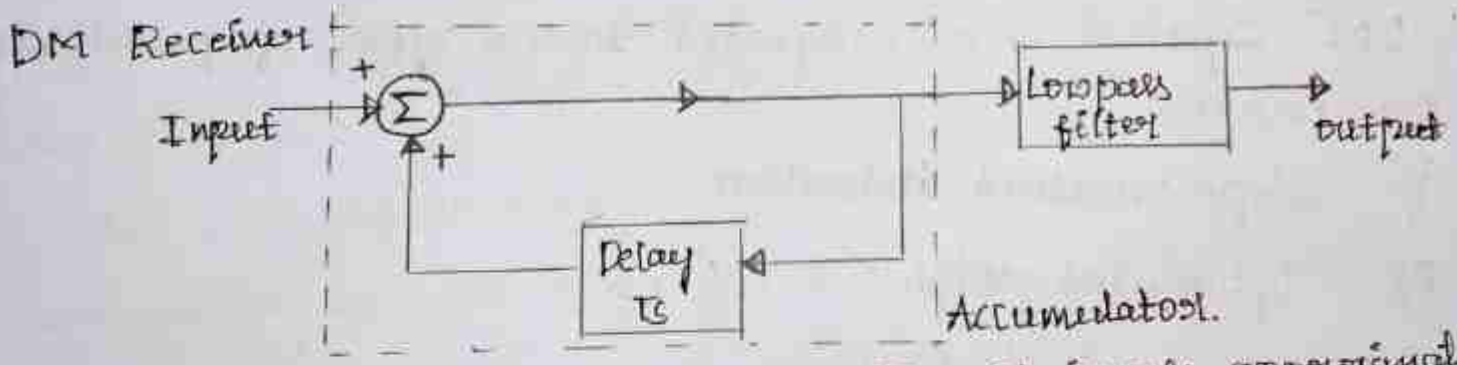
i.e $b(nT_s) = +\delta$, if $x(nT_s) \geq \hat{x}(nT_s)$

$b(nT_s) = -\delta$, if $x(nT_s) \leq \hat{x}(nT_s)$

* If $b(nT_s) = +\delta$, then binary '1' is transmitted.

If $b(nT_s) = -\delta$, then binary '0' is transmitted.

* The previous sample approximation $x[nT_s - T_s]$ is restored by delaying one sample period 'T_s'.



* The accumulator generates the staircase approximated signal output and is delayed by one sampling period 'T_s'. It is then added to the input signal.

* If Input is binary '1' then it adds $+\delta$ step to the previous output.

* If Input is binary '0', then one step 'δ' is subtracted from the delayed signal.

The Lowpass filter is used to remove step variation & to get smooth reconstructed message signal x(t).

Advantages of Delta modulation[DM]

The DM has following advantages over PCM

1) DM transmits only one bit for one sample. Thus the signalling rate & transmission channel bandwidth is quite small for DM

2) Simplicity of design for both the transmitter and the receiver

3) A one-bit code word for the output, which eliminates the need for word framing.

Drawbacks of DM:

DM systems are subjected to two types of quantization error:

1) Slope overload distortion

2) Granular noise.

Slope overload distortion

Slope overload distortion arises because of large dynamic range of the input signal.

As shown in fig, the rate of rise of input signal x(t) is so high that the staircase signal cannot approximate it, the step size δ becomes too small for staircase signal x(t). to follow the steep segment of x(t).

Thus large error between the staircase approximated signal & the original I/p signal x(t). This error is called slope overload distortion.

* To reduce this error, the step size should be increased when slope of the signal x(t) is high.

i.e Slope of the staircase u(t) ≥ Slope of the message signal

$$\frac{\delta}{T_s} \geq \max \left| \frac{d[x(t)]}{dt} \right|$$

Granular noise

This noise occurs when the step size is too large compared to small variations in the I/p signal. i.e for very small variations in the I/p signal, the staircase signal is changed by large amount because of large step size 'δ'.

* In the figure shown, I/p signal is almost flat, the staircase signal u(t) keeps on oscillating by ±δ around the signal.

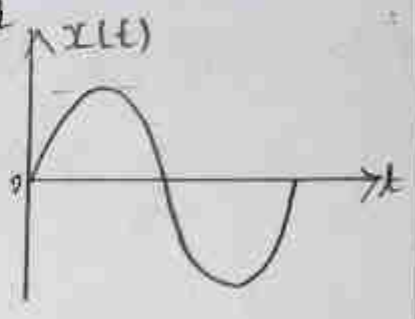
* The error between the I/p and approximated signal is called granular noise. The solution of this problem is to make stepsize small.

Consider a sine wave of frequency f_m & amplitude A_m applied to a delta modulator of step size δ.

ST Slope overload distortion will occur if

$$A_m > \frac{\delta}{2\pi f_m T_s}$$ where T_s is the sampling period

* Let the message signal be represented as $x(t) = A_m \sin(2\pi f_m t)$



* Consider uniform DM with step size δ & Sampling interval T_s.

$\therefore$ The Slope of DM $= \frac{\text{Step size}}{\text{Sampling period}} = \frac{\delta}{T_s}$

* The slope overload distortion will takes place if Slope of sine wave is greater than slope of demodulator

i.e $$\max \left| \frac{d}{dt}[x(t)] \right| > \frac{\delta}{T_s}$$

$$\max \left| \frac{d}{dt} A_m \sin(2\pi f_m t) \right| > \frac{\delta}{T_s}$$

$$\max \left| A_m \cos 2\pi f_m t \cdot 2\pi f_m \right| > \frac{\delta}{T_s}$$

Slope will be maximum when $\cos 2\pi f_m t = 1$
i.e at $t = 0$.

$$A_m(1) \cdot 2\pi f_m > \frac{\delta}{T_s}$$

$$\boxed{A_m > \frac{\delta}{2\pi f_m T_s}}$$

Derive an expression for signal to [noise] quantization noise power ratio for delta modulation. Assume that no slope overload distortion exists.

* The signal power $x(t) = A_0 \cos(2\pi f_0 t)$

The maximum slope of the signal $x(t)$ is given by

$$\left|\frac{d}{dt}[x(t)]\right|_{max} = \left|\frac{d}{dt} A_0 \cos(2\pi f_0 t)\right|_{max}$$

$$= \left|-A_0 \sin 2\pi f_0 t \cdot 2\pi f_0\right|_{max}.$$

$$\boxed{\left|\frac{d}{dt}[x(t)]\right|_{max} = A_0 2\pi f_0} \quad \text{——— (1)}$$

$\sin 2\pi f_0 t = 1$ at $t = 90^\circ$

* The condition for no slope overload distortion is

$$\boxed{\frac{\delta}{T_s} \geq \left|\frac{d[x(t)]}{dt}\right|} \quad \text{——— (2)}$$

Substituting eq (1) in eq (2), we get.

$$\frac{\delta}{T_s} \geq A_0 2\pi f_0$$

$$A_0 2\pi f_0 \leq \frac{\delta}{T_s}$$

$$A_0 = \frac{\delta}{2\pi f_0 T_s}$$

$$\left\langle A_0 = \frac{\delta f_s}{2\pi f_0} \right\rangle \quad \text{——— (3)}$$

* The maximum average power of the signal x(t) is given by

$$P_{max} = \frac{V^2}{R}$$

$V \rightarrow$ Rms value of the signal $\therefore V = \frac{A_0}{\sqrt{2}}$

* Normalised signal power is obtained by taking $R = 1$

$$P_{max} = \left(\frac{A_0}{\sqrt{2}}\right)^2 / 1$$

$$P_{max} = \frac{A_0^2}{2} \longrightarrow \langle 4 \rangle$$

Substituting eq ③ in eq ④, we get.

$$P_{max} = \frac{\delta^2 f_s^2}{4\pi^2 f_0^2} \times \frac{1}{2}$$

$$\left\langle P_{max} = \frac{\delta^2 f_s^2}{8\pi^2 f_0^2} \right\rangle \longrightarrow \langle 5 \rangle$$

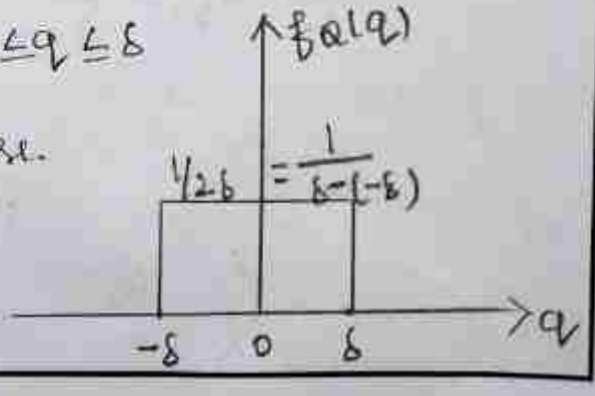
P_{max} is the signal power in delta modulation

Noise power

When there is no slope overload, the maximum quantization error is $\pm\delta$. Assuming that the quantization error is uniformly distributed.

* Consider the PDF of the quantization error i.e

$$f_Q(q) = \begin{cases} \frac{1}{2\delta} & \text{for } -\delta \le q \le \delta \\ 0 & \text{otherwise.} \end{cases}$$



The variance of quantization error is

$$\sigma_q^2 = \int_{-\delta}^{\delta} q^2 \cdot f_q(q) \cdot dq$$

$$= \int_{-\delta}^{\delta} q^2 \cdot \frac{1}{2\delta} \cdot dq = \frac{1}{2\delta} \int_{-q}^{q} q^2 \cdot dq$$

$$= \frac{1}{2\delta} \left[\frac{q^3}{3} \right]_{-\delta}^{\delta} = \frac{1}{2\delta} \left[\frac{\delta^3}{3} - \frac{(-\delta)^3}{3} \right]$$

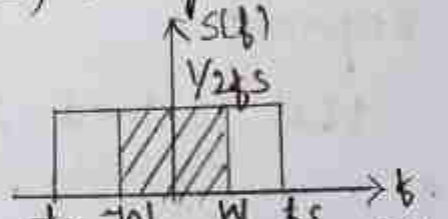
$$= \frac{1}{2\delta} \left[\frac{\delta^3}{3} + \frac{\delta^3}{3} \right]$$

$$= \frac{1}{2\delta} \left[\frac{2\delta^3}{3} \right]$$

$$\langle \sigma_q^2 = \delta^2/3 \rangle \quad \text{---------- (6)}$$

* The DM receiver contains a LPF whose cut-off frequency $f_c = W$ Hz.
* Assuming that the average power of the quantization error is uniformly distributed over a frequency interval extending from $-1/T_s$ to $+1/T_s$, we get

Average output noise power, N_o



$$N_o = \left(\frac{f_c}{f_s}\right) \frac{\delta^2}{3} = W T_s \frac{\delta^2}{3} \quad \text{---------- (7)}$$

* Signal to noise power ratio at the output of DM receiver is

$$\langle (SNR)_o = \frac{P_{max}}{N_o} \rangle$$

$$[SNR_o) = \frac{\delta^2 f_s^2 / 8\pi^2 f_o^2}{W T_s \delta^2/3} = \frac{\delta^2 f_s^2}{8\pi^2 f_o^2} \times \frac{3}{W T_s \delta^2}$$

$$[SNR_o) = \frac{3}{8\pi^2 f_o^2 T_s^2 . W T_s}$$

$$T_s^2 = 1/f_s^2$$

$$[SNR_o) = \frac{3}{8\pi^2 f_o^2 W T_s^3} \quad \text{---------- (8)}$$

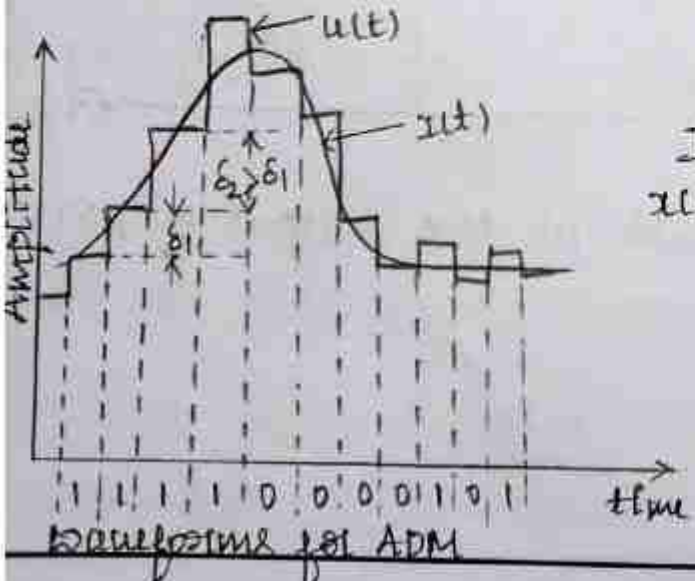
Eqn (8) shows that, under the assumption of no-slope overload distortion, the minimum output signal to noise ratio of a delta modulator is proportional to the sampling rate cubed.

Adaptive delta modulation.

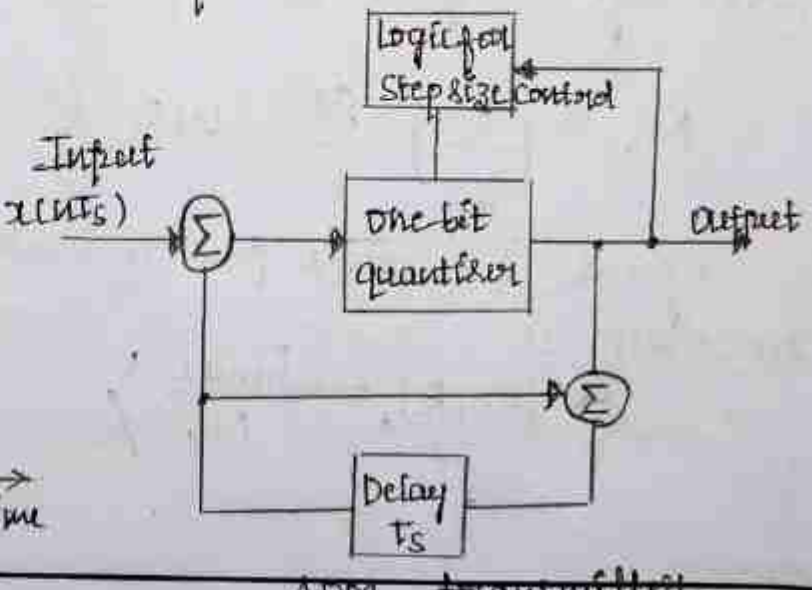
* To overcome slope overload distortion & granular noise, step size δ is made adaptive to variations in the input signal $x(t)$.

* The step size is increased for the steep segment of $x(t)$ & step size is decreased for slowly varying segment.

This method is called Adaptive delta modulation.



Waveforms for ADM



ADM transmitter.

* In practical adaptive DM, the step size $\delta[nT_s]$ is constrained to lie between maximum and minimum values

$$\text{i.e } \delta_{min} \leq \delta[nT_s] \leq \delta_{max}$$

* This upper limit δ_{max}, controls the amount of slope overload [condition] distortion
* The lower limit δ_{min}, controls the amount of granular noise
* Inside these limits the adaptive algorithm scale of $\delta[nT_s]$ expressed in general form

$$\boxed{\delta[nT_s] = g[nT_s]\,\delta[nT_s - T_s]}$$

Where the time varying function $g[nT_s]$ depends on the present & previous binary op of the delta modulator.

* The adaptive algorithm is initiated with a starting step

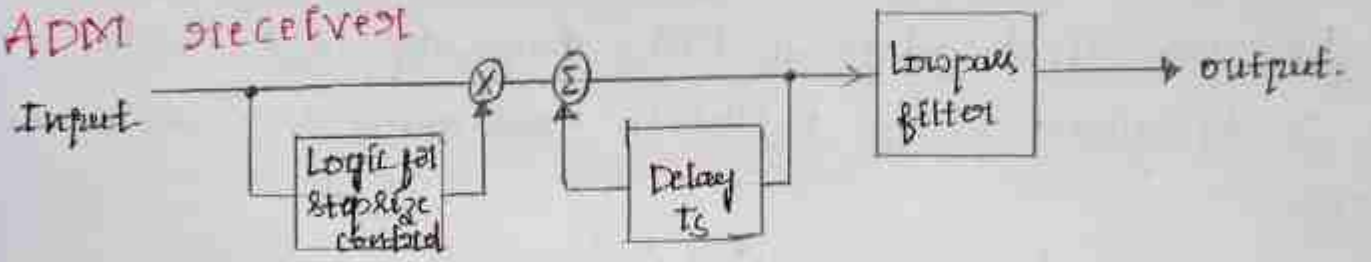
$$\boxed{\delta_{start} = \delta_{min}}$$

* The function $g[nT_s]$ is given by

$$g[nT_s] = \begin{cases} K & \text{if } b[nT_s] = b[nT_s - T_s] \\ K^{-1} & \text{if } b[nT_s] \neq b[nT_s - T_s] \end{cases}$$

* This adaptive algorith is called a constant factor ADM with one-bit memory
* The ADM is also known as continous variable slope DM

ADM receiver



* In the receiver, the 1st part generates step size from each incoming bit which is variable in size. The previous input & present input decides the step size.
* The LPF then smoothens out the staircase waveform to reconstruct the smooth signal.

Advantages of ADM

* The signal to noise ratio is better than ordinary delta modulation because of the reduction in the slope overload distortion & granular noise.
* Utilization of bandwidth is better than Delta modulation

Communication through Band limited channels

In base band data transmission, transmitted pulses tend to spread during transmission. The pulse spreading or pulse dispersion causes overlapping of pulses into adjacent time slots. This may result in an error at the receiver.

"The phenomenon of pulse spreading & resultant difficulty in discriminating the symbols at the receiver is termed as Inter Symbol Interference [ISI]."

** Digital PAM transmission through Band limited channels :-

The binary data $\{b_k\}$ is applied to the pulse generator which generates the signal $x(t)$.

i.e., $x(t) = \sum_{k=-\infty}^{\infty} A_k\, g(t - k T_b)$ where,

$T_b \rightarrow$ duration of each binary bit.

$g(t) \rightarrow$ is the shaping pulse normalized to 1V.

i.e., $g(0) = 1V$.

$$A_k = \begin{cases} +a & \text{if } b_k = 1 \\ -a & \text{if } b_k = 0 \end{cases}$$

The PAM signal $x(t)$ is then passed through the transmitting filter of transfer function $H_T(f)$. The signal is then passed through the channel having the transfer function $H_c(f)$. The channel may be co-axial cable or optical fibre.

The channel o/p is passed through a receiving filter of transfer function $H_R(f)$. The o/p of the rxg

Scanned with CamScanner

filter is $y(t)$. This $y(t)$ is noisy replica of the transmitted signal $x(t)$.

The o/p $y(t)$ is sampled with the sampling instants $t = i T_b$. The sampled signal $y(iT_b)$ is given to the decision device. The decision device compares the I/p signal with threshold 'λ'

If $y(iT_b) > \lambda$ symbol is '1'

If $y(iT_b) < \lambda$ symbol is '0'

The receiving filter output is,

$$y(t) = \mu \sum_{K=-\infty}^{\infty} A_K \; P(t - K T_b)$$

$\mu \rightarrow$ scaling factor

$P(t) \rightarrow$ normalized pulse i.e., $P(0) = 1$.

@ $t = iT_b$

$$y(iT_b) = \mu \sum_{K=-\infty}^{\infty} A_K \; P[\, iT_b - K T_b]$$

$$= \mu \sum_{K=-\infty}^{\infty} A_K \; P[(i-K)T_b]$$

$$y(iT_b) = \mu A_i \, P(0) + \mu \sum_{\substack{K=-\infty \\ K \neq i}}^{\infty} A_K \; P[(i-K)T_b]$$

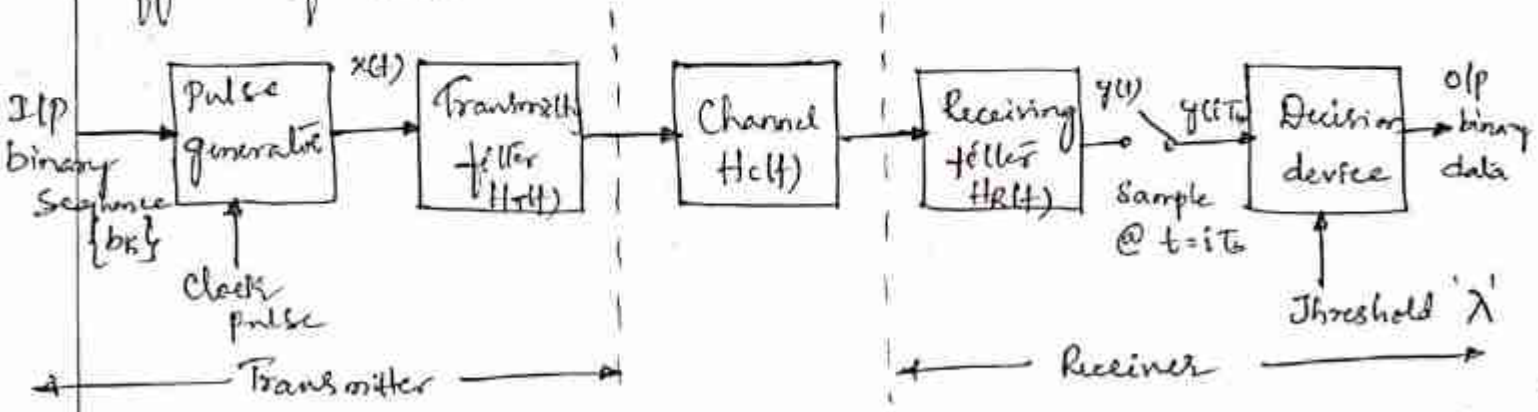
The first term represents the i^{th} transmitted symbol. The 2nd term represents the residual effect of all other bits transmitted before & after the sampling instants iT_b. This residual effect is called Inter Symbol Interference [ISI].

Scanned with CamScanner

In the absence of ISI

$$y(iT_b) = \mu A_i$$

The presence of ISI introduces error at the receiver output. Therefore while designing the transmitting & receiving filters it is necessary to minimize the effects of ISI.



The effect of ISI can be reduced by following methods

1) Nyquist pulse shaping criterion.
2) Raised cosine spectrum.
3) Correlative coding.

Nyquist criterion for distortionless base band binary transmission [Zero ISI] :-

Consider the equation,

$$y(iT_b) = \mu A_i + \mu \sum_{\substack{k=-\infty \\ k \neq i}}^{\infty} A_k \, p[(i-k)T_b] \quad \text{——} (1)$$

In the above equation, the second term must be zero to eliminate the effect of ISI. This is possible, if the received pulse $p(t)$ is controlled such that,

$$p[(i-k)T_b] = \begin{cases} 1 & \text{for } i=k \\ 0 & \text{for } i \neq k \end{cases} \quad \text{——} (2)$$

Scanned with CamScanner

If $p(t)$ satisfies the above eqn (condition), then we get an o/p which is free from ISI.

This condition is known as Nyquist criterion for zero ISI.

$\therefore y(iT_L) = \mu A_i$ for $i \neq k$. which implies zero ISI.

The equation ② gives more useful criteria in frequency domain.

If $p(t)$ is sampled using a Dirac comb with a period equal to T_L.

i.e., $p_\delta(t) = p(t)\,\delta(t)$ ——— ③

$$= p(t) \sum_{m=-\infty}^{\infty} \delta(t - mT_b)$$

$$= \sum_{m=-\infty}^{\infty} p(mT_b)\,\delta(t - mT_L)$$

Taking FT on both sides of eqn ③.

$$P_\delta(f) = P(f) * \delta(f)$$

$$P_\delta(f) = f_b \sum_{n=-\infty}^{\infty} P(f - nf_b) \quad \text{——— ④}$$

From the definition of F.T

$$P_\delta(f) = \int_{-\infty}^{\infty} p_\delta(t)\, e^{-j2\pi ft}\, dt.$$

$$P_\delta(f) = \int_{-\infty}^{\infty} \sum_{m=-\infty}^{\infty} p(mT_b)\,\delta(t - mT_L)\, e^{-j2\pi ft}\, dt.$$

Let the integer $m = i - k$ then $i = k$ corresponds to $m = 0$ only $i \neq k$ corresponds to $m \neq 0$.

Scanned with CamScanner

i.e, eqn ② becomes

$$P[(i-k)T_b] = P(mT_b) = \begin{cases} 1, & m=0 \\ 0, & m \neq 0 \end{cases}$$

For $i=k$ i.e., $m=0$

$$P_s(f) = \int_{-\infty}^{\infty} P(0)\, \delta(t)\, e^{j2\pi ft}\, dt$$

For $i \neq k$ i.e., $m \neq 0$

$$P_s(f) = 0$$

But from the defn of FT of a delta fun.

i.e., $$\int_{-\infty}^{\infty} \delta(t)\, e^{j2\pi ft}\, dt = 1$$

$\therefore$ $\boxed{P_s(f) = P(0)}$ for $i=k$.

$\Rightarrow$ $P_s(f) = 1$ for normalized value of $P(0)=1$

Equation ④ becomes.

$$1 = f_b \sum_{n=-\infty}^{\infty} P(f - nf_b)$$

or $$\boxed{\sum_{n=-\infty}^{\infty} P(f - nf_b) = \frac{1}{f_b} = T_b} \quad \text{—⑤}$$

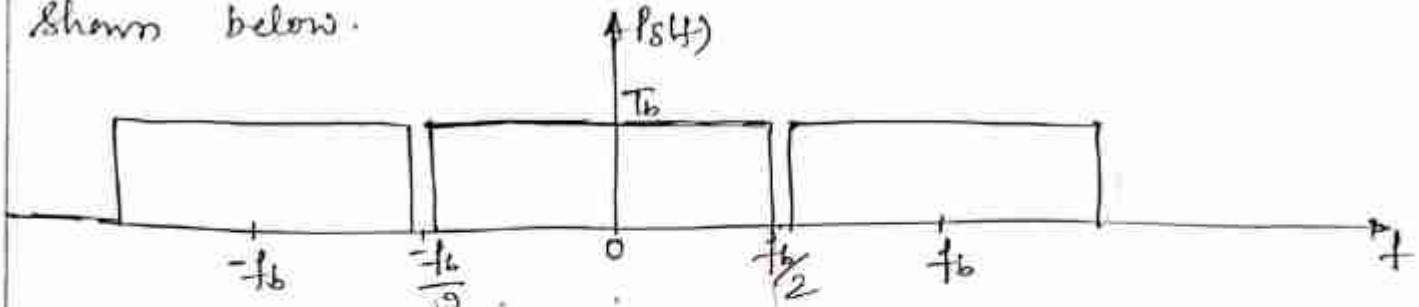
This is the frequency domain condition for zero ISI & this condn is called Nyquist criterion for distortionless base band transmission.

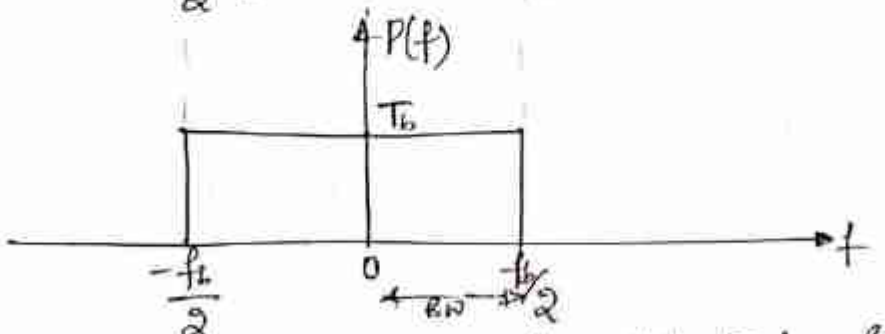
Ideal solution or Nyquist soln for zero ISI :—

The effect of ISI can be minimized by controlling $P(t)$ in time domain & $P(f)$ in frequency domain.

Scanned with CamScanner

Equation ⑤ represents the spectrum which repeats with period f_b & it has amplitude of T_b. This spectrum is shown below.





$P(f)$ shows the spectrum of the expected signal $P(t)$.
It can be represented by rect function as

$$P(f) = \frac{1}{f_b} \text{rect}\left(\frac{f}{f_b}\right)$$

$f_b = 2B_0$

Inverse FT of above function is

$$\frac{1}{f_b} \text{rect}\left(\frac{f}{f_b}\right) \longleftrightarrow \text{Sinc}(f_b t).$$

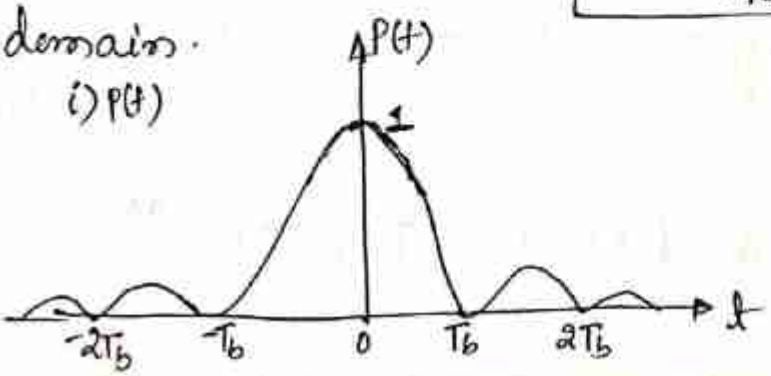
i.e., $P(t) = \text{Sinc}(f_b t)$

The B.W of the pulse can be represented by $B_0 = \frac{f_b}{2}$.

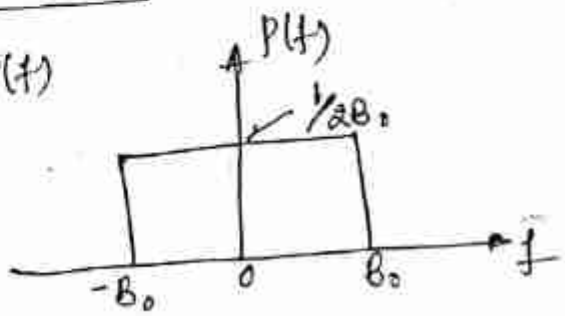
$\therefore P(t) = \text{Sinc}(2B_0 t)$.

The functions which gives this are $\boxed{P(t) = \text{Sinc}(2B_0 t)}$ in time domain & $\boxed{P(f) = \frac{1}{f_b} \text{rect}\left(\frac{f}{f_b}\right)}$ in frequency domain.

i) $P(t)$



ii) $P(f)$



Scanned with CamScanner

Nyquist B.W is given by.

$$B_0 = \frac{f_b}{2} = \frac{1}{2T_b}$$

$f_b = 2B_0$

"Nyquist B.W is defined as the minimum transmission B.W for zero ISI."

$$P(f) = \frac{1}{2B_0} \text{rect}\left(\frac{f}{2B_0}\right)$$

on simplification (using rect property).

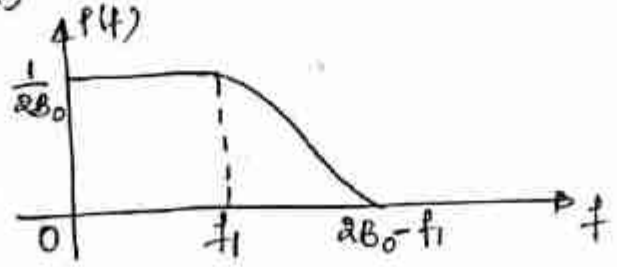
$$P(f) = \begin{cases} \frac{1}{2B_0} & \text{for } -B_0 \le f \le B_0 \\ 0 & \text{elsewhere.} \end{cases}$$

The spectrum of P(f) is flat from $-B_0$ to $+B_0$ & zero elsewhere. Since there is an abrupt transitions in the frequency at $\pm B_0$. Such frequency spectrum is not physically realizable. Hence the ideal sinc pulse cannot be physically generated.

Raised Cosine pulse shaping :-

In the raised cosine spectrum, the frequency response P(f) decreases towards zero gradually (i.e., there is no abrupt transition).

In otherwords, P(f) should have a flat portion & a roll-off portion that has the form of a raised cosine spectrum as shown below,



Scanned with CamScanner

Mathematically, a raised cosine spectrum is defined as

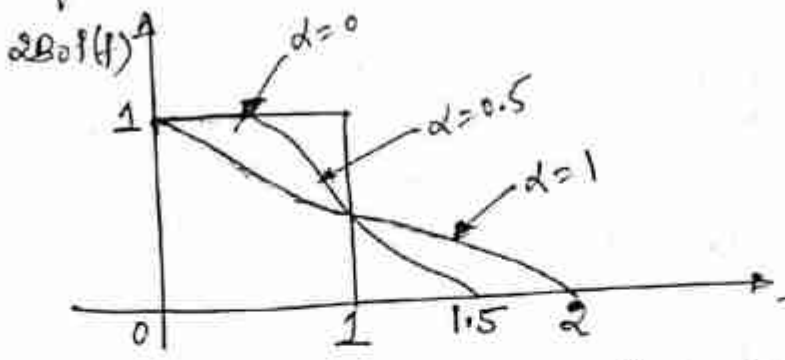
$$P(f) = \begin{cases} \frac{1}{2B_0} & \text{for } 0 \le |f| \le f_1 \\ \frac{1}{4B_0}\left[1 + \cos\left(\frac{\pi(|f| - f_1)}{2B_0 - 2f_1}\right)\right] & \text{for } f_1 \le |f| < 2B_0 - f_1 \\ 0, & |f| \ge 2B_0 - f_1 \end{cases}$$

In the above equation, the middle term represents the gradual roll off from f_1 to $2B_0 - f_1$. The freq f_1 & the Nyquist B.W B_0 are related by,

$$\alpha = 1 - \frac{f_1}{B_0}$$

The parameter α is called roll-off factor.

The frequency spectrum $P(f)$ is plotted for diffrent values of α i.e, $\alpha = 0, 0.5$ & 1 as shown below,



$0 \le \alpha \le 1$

fig: Amplitude response for diffrent roll-off factors.

For $\alpha = 1$, we get full cosine roll-off characteristics with the transmission b.w at $2B_0$ which is twice that of ideal solution.

The inverse F.T of $P(f)$ gives.

Scanned with CamScanner

$$P(f) = \text{Sinc}(2B_0 t)\,\frac{\cos(2\pi\alpha B_0 t)}{1-16\alpha^2 B_0^2 t^2}$$

When $\alpha = 1$, we get

$$P(f) = \frac{\text{Sinc}(4B_0 t)}{1-16B_0^2 t^2}$$

$$P(f) = \begin{cases} \frac{1}{4B_0}\left[1+\cos\left(\frac{\pi f}{2B_0}\right)\right], & 0 < |f| < 2B_0 \\ 0, & |f| \geq 2B_0 \end{cases}$$

Transmission B.W requirement :-

The frequency response $P(f)$ is non zero in the interval $[0, (2B_0 - f_1)]$. The transmission b.w is

$$B = 2B_0 - f_1$$

Where $B_0 = \frac{1}{2T_b}$ is the Nyquist B.W & $\alpha = 1 - \frac{f_1}{B_0}$

$$f_1 = B_0(1-\alpha)$$

$$B = 2B_0 - B_0(1-\alpha)$$

$$\boxed{B = B_0(1+\alpha)}$$

For $\alpha = 0$, $B = B_0$, this is same as ideal solution. As α increases, the B.W increases, when $\alpha = 1$, $B = 2B_0$ i.e., bandwidth required by the raised cosine spectrum is double of the Nyquist B.W.

Scanned with CamScanner

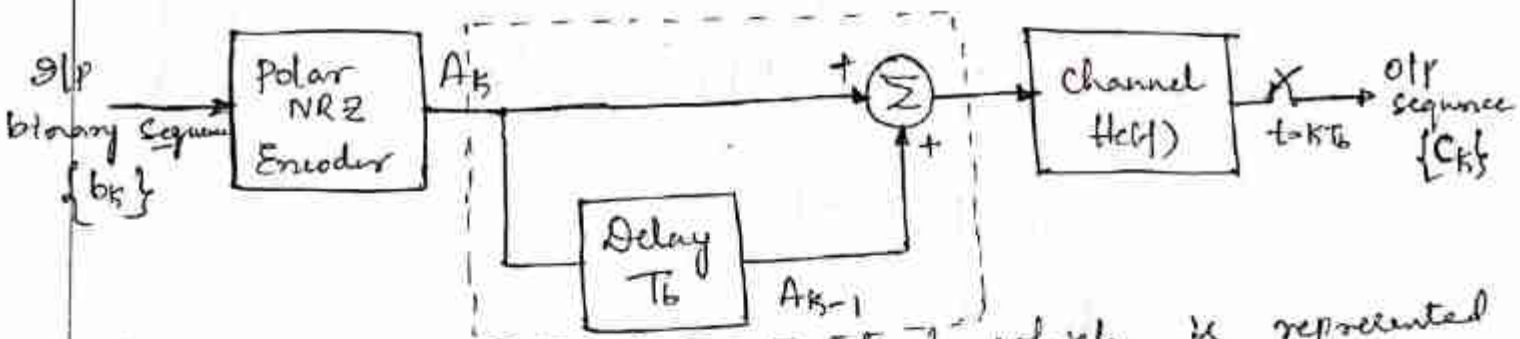
* Correlative Coding :-

In correlative coding two successive bits are used for evaluation. It is also called duobinary encoding.

Consider two successive pulses $p(t)$ and $p(t-T_b)$. If both $p(t)$ & $p(t-T_b)$ are positive, the sample value will be 2. If both $p(t)$ & $p(t-T_b)$ are negative, the sample value will be -2. If $p(t)$ & $p(t-T_b)$ are of opposite polarities, the sample value will be 0. This gives us the logic to make correct decisions.

The correlative coding is implemented by duobinary signaling & modified duobinary encoding.

* Duo binary Encoding :-



Consider the input sequence $\{b_k\}$ which is represented by polar NRZ sequence A_k.

$$\text{i.e., } \begin{cases} A_k = +1V, & \text{if } b_k = 1 \\ A_k = -1V, & \text{if } b_k = 0 \end{cases}$$

When this sequence is applied to a duobinary encoder, the output will contain three levels -2, 0 and $+2$. The o/p of the duo binary encoder can

Scanned with CamScanner

be expressed as,

$$C_k = A_k + A_{k-1}$$

Here the uncorrelated sequence $\{A_k\}$ of two level pulses is converted into correlated sequence of three level pulses.

Reconstruction: Let $\hat{A}_k$ represents the estimate of A_k. Then we can obtain $\hat{A}_k$ as

$$\hat{A}_k = C_k - \hat{A}_{k-1}$$

This shows that if C_k is received with error, then $\hat{A}_k$ will have error. This error will propagate in the output sequence.

Draw back :- Error propagation takes place in the decoder.

Example :-

Sequences k	0	1	2	3	4	5	6	7	8
1) Binary seq b_k	0	0	1	1	0	1	0	0	1
2) Polar representation A_k	-1	-1	+1	+1	-1	+1	-1	-1	+1
3) Duo binary encoder o/p $C_k = A_k + A_{k-1}$	-	-2	0	+2	0	0	0	-2	0
4) Estimate of $\hat{A}_k = C_k - \hat{A}_{k-1}$	-1	-1	+1	+1	-1	+1	-1	-1	+1
5) Estimated binary sequence.	0	0	1	1	0	1	0	0	1

Scanned with CamScanner

Duobinary Encoder with Precoder :-

Encoder :- Precoder is used in duobinary encoder to avoid error propagation. Fig shows the block diagram of duobinary encoder with precoder. Precoder is nothing but a differential encoder.

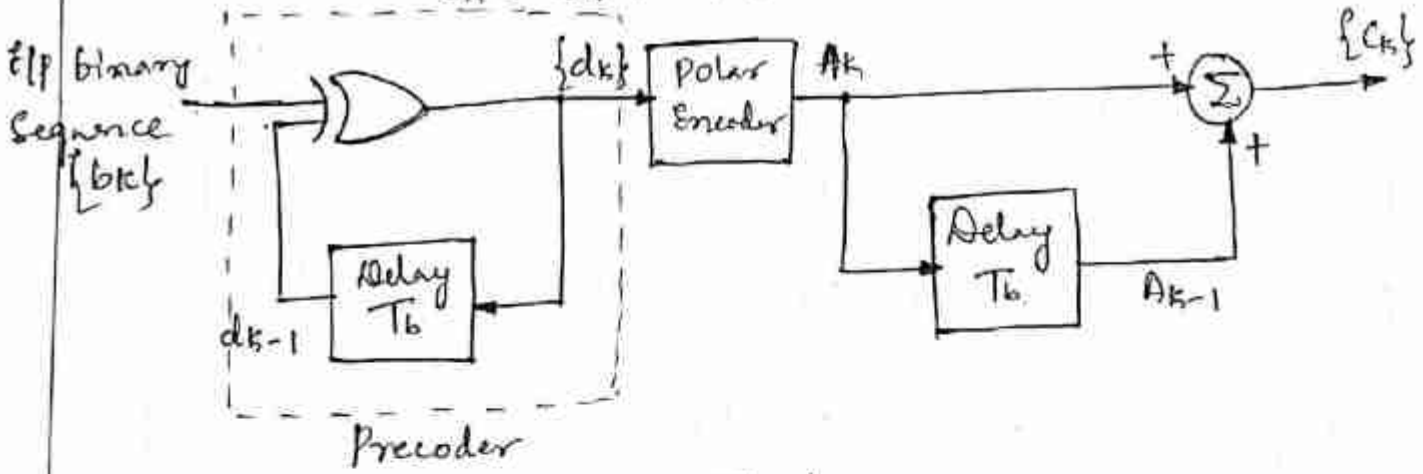
The O/p of the precoder is

$$d_k = b_k \oplus d_{k-1}$$

This is normal exclusive-OR operation.

The precoded sequence $\{d_k\}$ is applied to polar NRZ level encoder. The sequence $\{A_k\}$ is then applied to duobinary encoder. The output C_k is

$$C_k = A_k + A_{k-1}$$

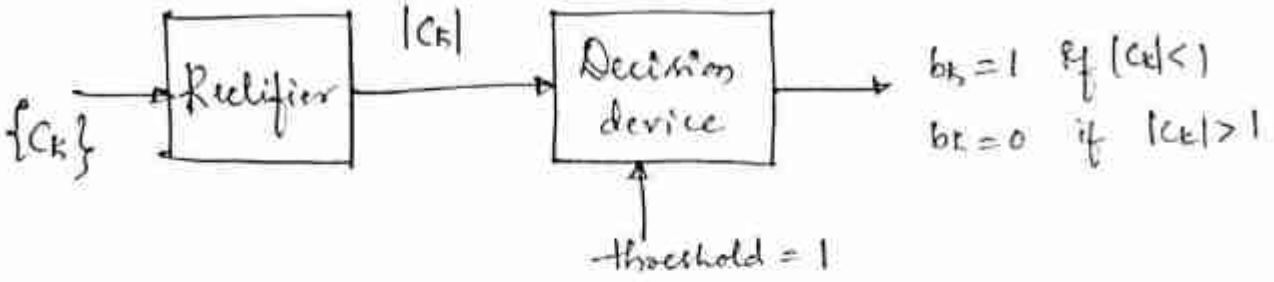


The value of C_k will be,

$$C_k = \begin{cases} \pm 2 & \text{if } b_k = 0 \\ 0 & \text{if } b_k = 1 \end{cases}$$

Decoder :- fig below shows the block diagram of duobinary decoder.

Scanned with CamScanner



The magnitude of the o/p sequence Ck is taken. It is compared with '1'. Then decisions are taken as follows,

If $|C_k| > 1$ then $b_k = 0$

$|C_k| < 1$ then $b_k = 1$

The o/p b_k depends upon the present value of Ck. Since previous value of the o/p is not required. There is no propagation of error in the system.

Example :-

Sequences k	-1	0	1	2	3	4	5	6
1) Binary Seq {bk}		0	0	1	0	1	1	0
2) Precoded Seq dk	d-1 (1)	1	1	0	0	1	0	0
3) Polar representation of dk {Ak}	+1	+1	+1	-1	-1	+1	-1	-1
4) Duo binary o/p Ck	-	+2	+2	0	-2	0	0	-2
5) Mag of |Ck|		+2	+2	0	+2	0	0	+2
6) o/p binary Seq bk		0	0	1	0	1	1	0

Scanned with CamScanner

* **Modified Duo binary (MDB) Encoding :-**
(Without Precoder)

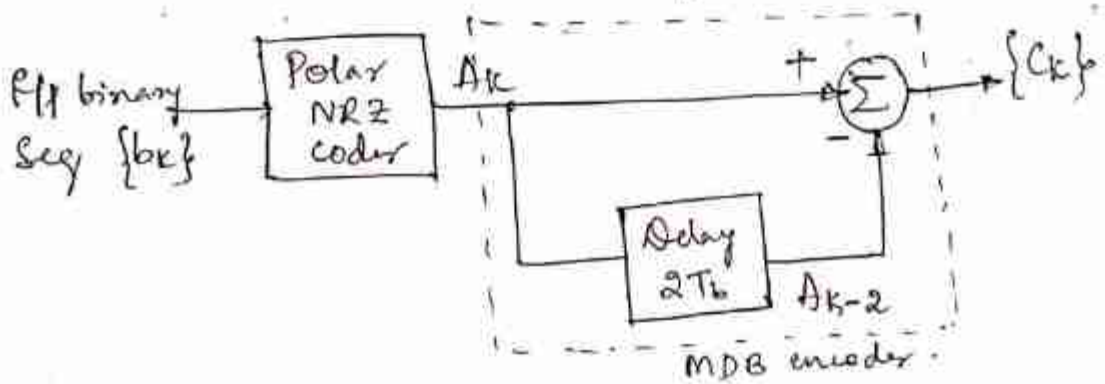
The correlation is over two binary digits in modified duo binary encoding. The output of the MDB is

$$C_k = A_k - A_{k-2}$$

Where $A_k = +1V$, If $b_k = 1$

$A_k = -1V$ If $b_k = 0$

Here the uncorrelated sequence $\{A_k\}$ of two level is correlated into correlated sequence of three level pulses.



Reconstruction: Let $\hat{A}_k$ represents the estimate of A_k. then we can write,

$$\hat{A}_k = C_k + \hat{A}_{k-2}$$

This shows that If C_k is received with error, then $\hat{A}_k$ will have error. This error will propagate to the output sequence.

* **MDB encoder with Precoder :-**

Encoder:- To avoid error propagation, a precoder (differential encoder) is used at the input. Fig shows the block diagram of MDB encoder with precoder.

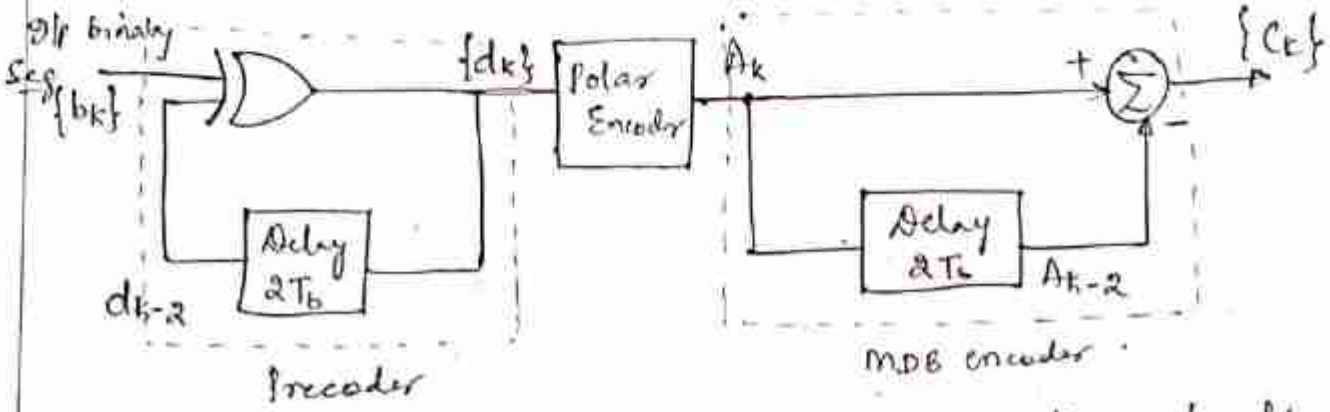
Scanned with CamScanner

The output of the precoder is

$$d_k = b_k \oplus d_{k-2}$$

This is normal Ex-OR operation. The precoded seq $\{d_k\}$ is applied to NRZ level encoder. The seq $\{A_k\}$ is then applied to MDB encoder. The o/p C_k is

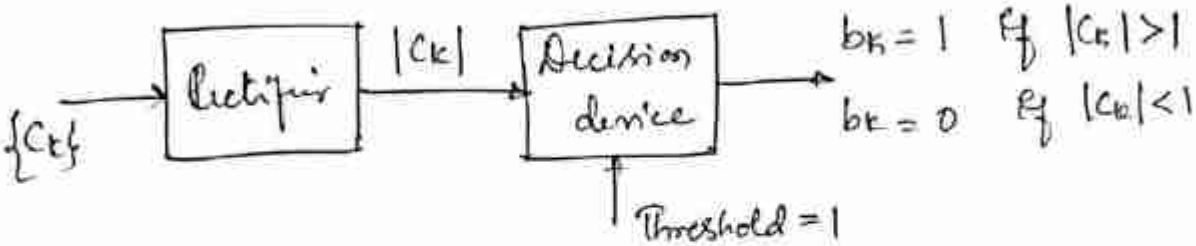
$$C_k = A_k - A_{k-2}$$



The sequence C_k is uncorrelated with three levels +2, 0, & -2.

$$C_k = \begin{cases} \pm 2, & \text{if } b_k = 1 \\ 0 & \text{if } b_k = 0 \end{cases}$$

Decoder :- Fig below shows the block diagram of MDB decoder.



The magnitude of the o/p sequence C_k is taken. It is compared with 1. Then decisions are taken as follows,

If $|C_k| > 1$ then $b_k = 1$

$|C_k| < 1$ then $b_k = 0$

Scanned with CamScanner

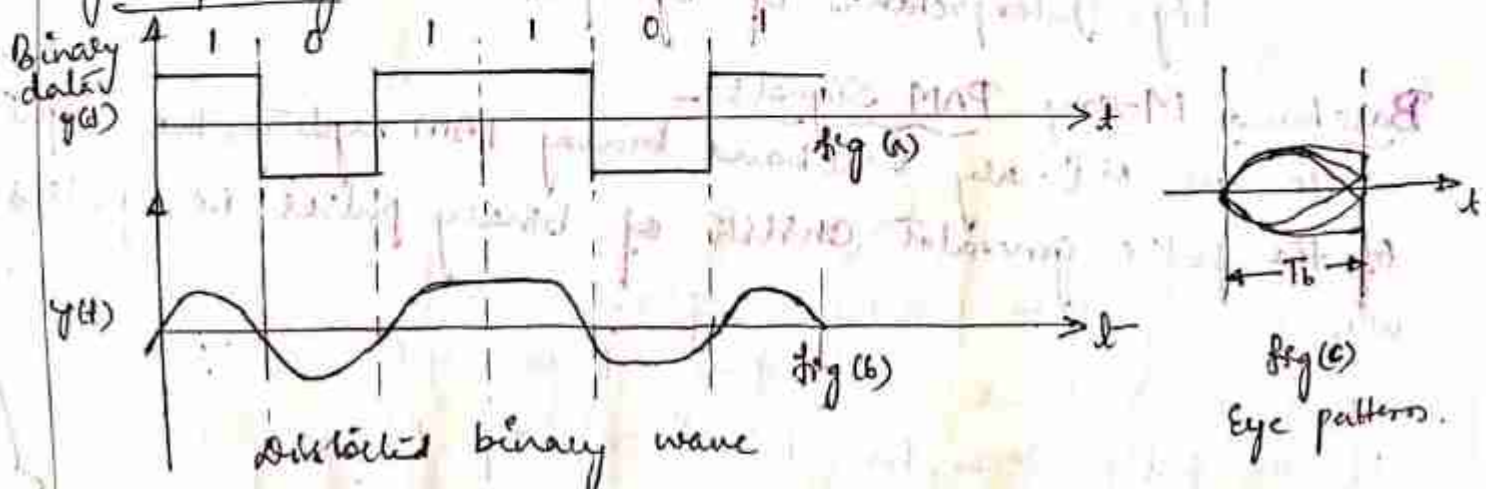
Thus the precoding of the data at the transmitter makes it possible to detect the received data on a symbol-by-symbol basis without having to look back at previously detected symbols. Thus, error propagation is avoided.

**Note:- Correlative coding is also called as symbol-by-symbol detection of data with controlled ISI. (with precoder)

Eye diagram:- [Eye pattern]

The Eye pattern is used to study the effect of ISI in baseband digital transmission.

Consider the transmission of binary data 10110 by NRZ polar signalling. If the channel is ideal with infinite bandwidth, pulses will be received without distortion [fig (a)] But because of channel distortion, the o/p will be distorted. If we cut the signal y(t) shown in fig (b) in each interval Tb & place it over one another, then we obtain the pattern as shown in fig (c). This pattern is called as eye pattern because of its resemblance to the human eye. The interior region of the eye pattern is called the eye opening.



Scanned with CamScanner

The eye pattern will provide the following information,

1) The width of the eye opening defines the interval over which the received wave can be sampled without error from interSymbol interference. It is preferable to sample the instant at which eye is open widest.

2) The sensitivity of the system to timing error is determined by the rate of closure of the eye as the sampling time is varied.

3) The height of the eye opening, at a specified sampling time, defines the margin over noise.

As the effect of ISI increases, the eye opening reduces. If the eye opening is closely completely, then it is not possible to avoid errors in the output.

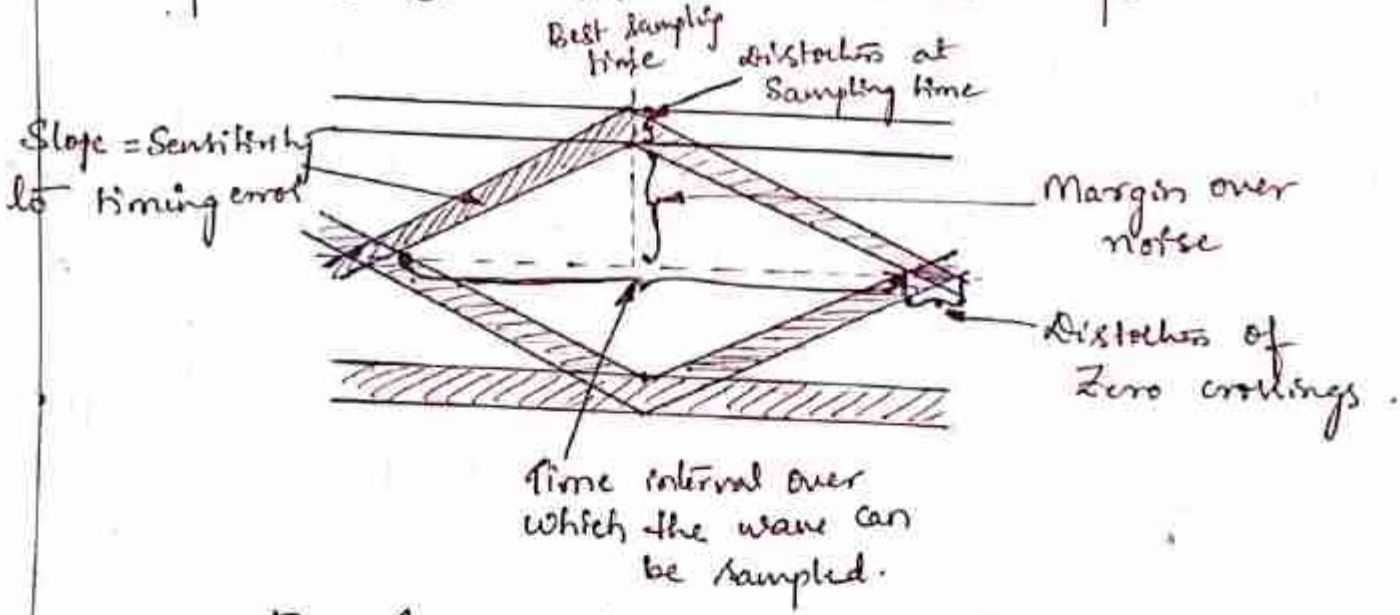


Fig: Interpretation of eye pattern.

Scanned with CamScanner

CBCS SCHEME

CENTRAL LIBRARY

USN | | | | | | | | | | |

15EC61

Sixth Semester B.E. Degree Examination, June/July 2019
Digital Communication

Time: 3 hrs. Max. Marks: 80

Note: *Answer any FIVE full questions, choosing ONE full question from each module.*

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and /or equations written eg, 42+8 = 50, will be treated as malpractice.

Module-1

1 a. Determine the Hilbert transform of the signal $g(t) = \sin c(t)$. **(04 Marks)**

b. Determine the pre-envelope and complex envelope of the signal shown in Fig.Q1(b). **(06 Marks)**

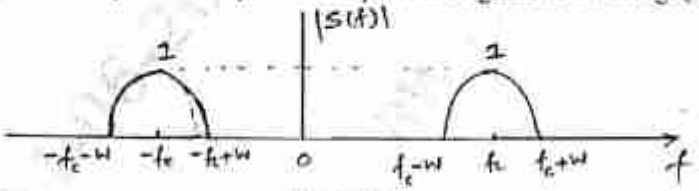


Fig.Q1(b)

c. Explain the time-domain procedure for the complex representation of band pass signals and systems. **(06 Marks)**

OR

2 a. For a binary sequence 0100000001011 construct i) RZBipolar format ii) Manchester format iii) B3Zs format iv) B6ZS format v) HDB3 format.
Also mention the application of B3ZS and B6Zs formats. **(07 Marks)**

b. Draw the power spectra of : i) RZAMI signal ii) NRZ polar signal. **(03 Marks)**

c. Consider a bandpass signal S(t) which is represented in terms of in-phase and quadrature components. Suggest a suitable scheme for :
i) extracting the in-phase and quadrature components from the band pass signal
ii) reconstructing the band pass signal from in-phase and quadrature components. **(06 Marks)**

Module-2

3 a. For the signals $s_1(t)$, $s_2(t)$, $s_3(t)$ and $s_4(t)$ shown in Fig.3(a), find a set of orthonormal basis functions using gram-Schmidt orthogonalization procedure. **(09 Marks)**

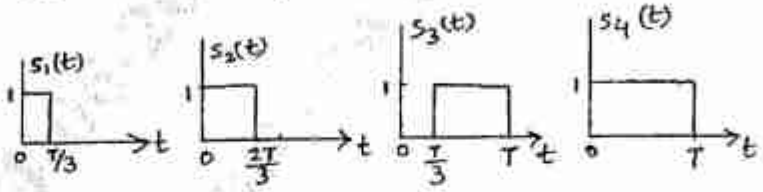


Fig.Q3(a)

b. Explain with neat diagram and necessary equations the matched filter receiver. **(07 Marks)**

OR

4 a. Obtain the decision rule for maximum likelihood decoding and explain the correlation receiver. **(08 Marks)**

b. Show that for a noisy input, the mean value of the j^{th} correlator output X_j depends only on S_{ij} and all the correlators outputs X_j, j = 1, 2, - - - - , N, have a variance equal to the PSD No/2 of the additive noise process w(t). **(08 Marks)**

Scanned with CamScanner

Module-3

5 a. Sketch the QPSK wave form for the sequence 01101000. (06 Marks)

b. Obtain the expression for average probability of symbol error for BPSK using coherent detection. (06 Marks)

c. Obtain the constcleation of QAM for M = 16 and draw the signal space diagram. (04 Marks)

OR

6 a. Explain the generation and coherent detection of BFSK system. (06 Marks)

b. The binary sequence 1100100010 is applied to the DPSK transmitter

i) Sketch the resulting wave from at the transmitter output.

ii) Applying this waveform to the DPSK receiver, show that in the absence of noise, the original binary sequence is reconstructed at the receiver output. (06 Marks)

c. An FSK system transmits binary data at the rate of 2×10^6 bps. During the source of transmission, AWGN of zero mean and two sided PSD 10^{-20} Watts/Hz is added to the signal. The amplitude of the received sinusoidal wave for digit 1 or 0 is 1μv. Determine the average probability of symbol error assuming non-coherent detection. (04 Marks)

Module-4

7 a. Explain the following terms with related equations and diagram with respect to base band transmission.

i) ISI and Nyquist condition for zero ISI

ii) Duobinary signal pulse

iii) Modified duobinary signal pulse

iv) Partial response signals

v) Raised cosine spectrum. (10 Marks)

b. Explain the need for precoder in a duobinary signaling. The binary sequence 1110100100001101 is the input to the precoder whose output is used to modulate a duobinary transmitting filter. Obtain the precoded sequence, transmitted amplitude levels, the received signal levels and the decoded sequence. (06 Marks)

OR

8 a. With a neat diagram, explain the concept of linear traversal filter. (06 Marks)

b. Consider a channel distorted pulse x(t), at the input to the equalizer, given by $x(t) = \frac{1}{1+\left(\frac{2t}{T}\right)^2}$ where 1/T is the symbol rate. The pulse is sampled at the rate 2/T and equalized by a zero-forcing equalizer. Determine the coefficients of a five-tap zero-forcing equalizer. (06 Marks)

c. Write a note on eye diagram. (04 Marks)

Module-5

9 a. With a neat diagram explain the generation of PN sequences and state its properties. (06 Marks)

b. A DS spread–spectrum signal is designed so that the power ratio P_R/P_S at the intended receiver is 10^{-3}. If the desired $E_b/N_0 = 10$ for acceptable performance, determine the minimum value of the processing gain. (04 Marks)

c. Explain with neat block diagram FH spread –spectrum system. (06 Marks

OR

10 a. Explain the generation and demodulation of DS spread spectrum signal. (06 Marks)

b. Write a note on application of spread spectrum in wireless LANs. (04 Marks)

c. With a neat block diagram, explain the IS–95 reverse link. (06 Marks)

Scanned with CamScanner

CBCS SCHEME

USN

15EC61

Sixth Semester B.E. Degree Examination, Dec.2018/Jan.2019
Digital Communication

Time: 3 hrs. Max Marks: 80

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and /or equations written eg, 42+8 = 50, will be treated as malpractice.

Module-1

1 a Define Hilbert transform. What are its applications. Prove that a signal g(t) and its Hilbert transform $\hat{g}(t)$ are orthogonal over the entire time interval $(-\infty, \infty)$. **(05 Marks)**

b Determine the pre-envelope and complex envelope of the RF pulse defined by $x(t) = A\,rect\left(\frac{t}{\tau}\right)\cos 2\pi f_c t$. **(06 Marks)**

c Compare the power spectra of various line codes in terms of bandwidth, DC component, Noise immunity and synchronization capability, with neat sketch. **(05 Marks)**

OR

2 a Express bandpass signal s(t) in canonical form. Also explain the scheme for deriving the inphase and quadrature components of the bandpass signal s(t). **(06 Marks)**

b Explain with relevant expressions, the procedure for computational analysis of a bandpass system driven by a bandpass signal. **(06 Marks)**

c What is the advantage of HDB3 code over conventional alternate mark inversion(AMI) code. Code the pattern "1010000011000011000000" using HDB3 encoding and AMI encoding. **(04 Marks)**

Module-2

3 a Explain the geometric representation of set of M energy signals as linear combination of N orthonormal basis functions. Illustrate for the case N = 2 and M = 3, with necessary diagrams and expressions. **(08 Marks)**

b Using the Gram-Schmidt orthogonalization procedure, find a set of orthonormal basis functions to represent the three signals $s_1(t)$, $s_2(t)$ and $s_3(t)$ shown in Fig.Q3(b). also express each of these signals in terms of the set of basis functions. **(08 Marks)**

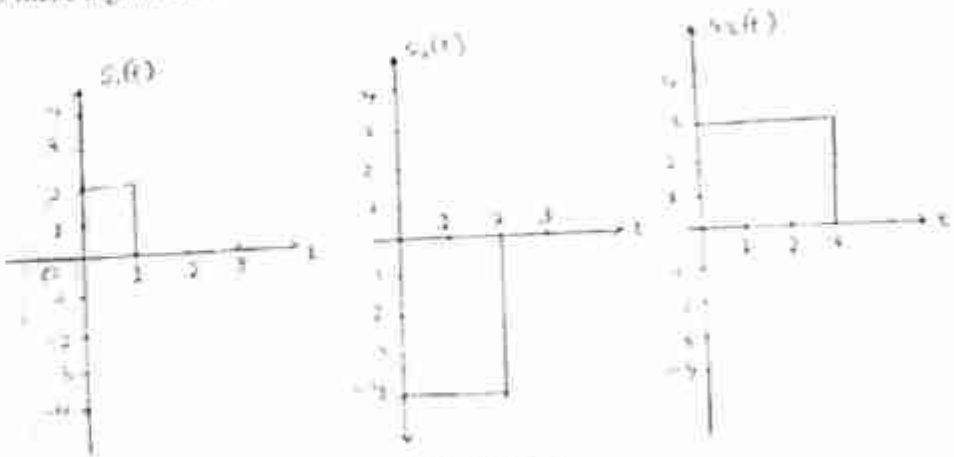


Fig Q3(b)

1 of 3

Scanned with CamScanner

OR

4 a. Explain the correlation receiver with neat diagrams and explanation of detector and the maximum –linklihood decoder blocks. **(08 Marks)**

b. Explain the matched filter receiver. Obtain the expression for the impulse response of the matched filter. **(08 Marks)**

Module-3

5 a. Derive the expression for error probability of binary PSK using coherent detection. **(06 Marks)**

b. Binary data are transmitted over a microwave link at the rate of 10^6 bits/sec and the power spectral density of the noise at the receiver input is 10^{-10} w/Hz. find the average carrier power required to maintain an average probability of error $Pe \leq 10^{-4}$ for the following cases.
Binary PSK using coherent detection
DPSK
Note : take erfc(2.63) = 2×10^{-4}, Q(3.7) = 10^{-4}. **(06 Marks)**

c. Define bandwidth efficieney. Tabulate and comment on the bandwidth efficiency of M-ary PSK signals for different values of M. **(04 Marks)**

OR

6 a. With neat diagram and expressions, explain binary FSK generation and non-coherent detection scheme. **(06 Marks)**

b. Explain the generation and optimum detection of differential phase – shift keying with neat block diagram. **(06 Marks)**

c. What is the advantage of M-ary QAM over M-ary PSK system? Obtain the constellation of QAM for M = 4 and draw signal space diagram. **(04 Marks)**

Module-4

7 a. With a neat block diagram, explain the digital PAM transmission through band limited baseband channels. Also obtain the expression for inter symbol interference. **(06 Marks)**

b. Explain the modified duo-binary signaling scheme, with pre-coding. Illustrate the encoding for the binary sequence "011100101". Assume previous pre-coder outputs as 1. **(07 Marks)**
With neat diagram, explain the timing features pertaining to eye diagram and it interpretation for baseband binary data transmission system. **(03 Marks)**

OR

8 a. With neat sketches and expressions, explain raised cosine spectrum solution to reduce ISI. **(06 Marks)**

b. What is the advantage of controlled ISI partial response signaling scheme? With block diagram, explain the duo-binary encoder with pre-coder. Mention the frequency response, impulse response and its features. **(06 Marks)**

c. With neat diagram and relevant expressions, explain the concept of adaptive equalization. **(04 Marks)**

Scanned with CamScanner

Module-5

9 a. Explain the working of Direct Sequence Spread Spectrum transmitter and receiver with neat diagram, waveform and expressions. **(08 Marks)**

b. A slow frequency Hopped/MFSK system has the following parameters,
   i) The number of bits/MFSK symbol = 4
   ii) The number of MFSK symbols per hop = 5
   iii) Calculate the processing gain of the system in decibels. **(03 Marks)**

c. List and briefly explain any 3 applications of direct sequence spread spectrum. **(05 Marks)**

OR

10 a. With a neat block diagram, explain frequency Hopped spread spectrum technique. Explain the terms chip rate, Jamming Margin and processing gain. **(08 Marks)**

b. What is a PN sequence? Explain the generation of maximum-length sequences (ML-sequence). What are the properties of ML sequences? **(04 Marks)**

c. In a DS/BPSK system, the feedback shift register used to generate the PN sequence has length m = 19. The system is required to have an average probability of symbol error due to externally generated interfering signals that does not exceed 10^{-5}. Calculate the following system parameters in decibels :
   i) Processing gain
   ii) Antijam margin
   (Assume Q(4.25) = 10 –5 or erfc(3) = 2×10^{-5}). **(04 Marks)**

* * * * *

Scanned with CamScanner

CBCS Scheme

USN | 4 | B | W | 1 | 5 | E | C | 1 | 0 | 6 |

15EC61

Sixth Semester B.E. Degree Examination, June/July 2018

Digital Communication

Time: 3 hrs. Max. Marks: 80

***Note:** Answer any FIVE full questions, choosing one full question from each module.*

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and /or equations written eg, 42+8 = 50, will be treated as malpractice.

Module-1

1 a. Define Hilbert transform. List the properties of the Hilbert transform. **(04 Marks)**

b. Obtain the canonical representation of band pass signals. **(06 Marks)**

c. What is line coding? For the binary stream 011010 sketch the following line codes:
i) Unipolar NRZ
ii) Polar NRZ
iii) Unipolar RZ
iv) Bipolar RZ
v) Manchester **(06 Marks)**

OR

2 a. Define pre-envelope of a real valued signal. Given a band pass signal s(t), sketch the amplitude spectra of signal s(t), pre-envelope $s_+(t)$ and complex envelope $\tilde{s}(t)$. **(04 Marks)**

b. Derive the expression for the complex low pass representation of band pass systems. **(08 Marks)**

c. Write a note on HDBN signaling. **(04 Marks)**

Module-2

3 a. Explain the geometric representation of signals. Show that energy of the signal is equal to the squared length of the vector representing it. **(08 Marks)**

b. Derive the expressions for mean and variance of the correlator outputs. Also show that the correlator outputs are statistically independent. **(08 Marks)**

OR

4 a. Explain the Gram-Schmidt orthogonalization procedure. **(06 Marks)**

b. Obtain the maximum likelihood decision rule for the signal detection problem. **(10 Marks)**

Module-3

5 a. Explain the signal space representation for binary phase shift keying modulation. Also derive the expression for the probability of error for the binary phase shift keying. **(10 Marks)**

b. With a neat block diagram, explain the generation and coherent detection of QPSK signals. **(06 Marks)**

OR

6 a. With a neat block diagram, explain the non-coherent detection of binary frequency shift keying technique. **(04 Marks)**

b. Derive an expression for probability of error of binary frequency shift keying technique. Also draw the block diagrams of BFSK transmitter and coherent receiver. **(10 Marks)**

c. For the binary sequence given by 10010011, illustrate the operation of DPSK. **(02 Marks)**

Scanned with CamScanner

Module-4

7 a. With a neat block diagram of digital PAM system obtain the expression for inter symbol interference (ISI). **(06 Marks)**

b. State and prove Nyquist condition for zero ISI. **(06 Marks)**

c. For the binary data sequence $\{d_n\}$ given by 11101001. Determine the precoded sequence, transmitted sequence, received sequence and the decoded sequence. **(04 Marks)**

OR

8 a. Explain the design of band limited signals with controlled ISI. **(10 Marks)**

b. What is a zero forcing equalizer? With a neat block diagram, explain the operation of linear transversal filter. **(06 Marks)**

Module-5

9 a. Explain the model of a spread spectrum digital communication system. **(06 Marks)**

b. Explain the generation and demodulation of direct sequence spread spectrum signals with necessary equation and block diagram. **(07 Marks)**

c. Write a note on low detectability signal transmission as an application of direct sequence spread spectrum. **(03 Marks)**

OR

10 a. With a neat block diagram, explain the frequency hopped spread spectrum. **(07 Marks)**

b. Explain the effect of despreading on a Narrow band interference in direct sequence spread spectrum systems. A direct sequence spread spectrum signal is designed to have the power ratio P_R/P_N at the intended receiver is 10^{-2}. If the desired $E_b/N_o = 10$ for acceptable performance, determine the minimum value of processing gain. **(06 Marks)**

c. Write a note on code division multiple access as an application of direct sequence spread spectrum. **(03 Marks)**

* * * * *

Scanned with CamScanner

6th Semester BE (CBCS) EC/TC Model Question Papers

15EC61

Visvesvaraya Technological University, Belagavi

MODEL QUESTION PAPER

6th Semester, B.E (CBCS) EC/TC

Course: 15EC61 - Digital Communication

Max Marks: 80 **Time: 3 Hours**

Note: (i) Answer Five full questions selecting any one full question from each Module.

(ii) Question on a topic of a Module may appear in either its 1st or/and 2nd question.

		Module 1	
1	(a)	Define Hilbert Transform. State the properties of it.	4
	(b)	Define the complex envelope of bandpass signals. Obtain the canonical representation of bandpass signals	6
	(c)	Derive the power spectral density of polar NRZ signals and plot the spectrum.	6
		OR	
2	(a)	Define the Pre-envelope. Show the spectral representations of pre-envelopes for low pass signals.	4
	(b)	Derive the expression for the complex low pass representation of bandpass systems.	7
	(c)	Given the data stream 1110010100. Sketch the transmitted sequence of pulses for each of the following line code. (i) Unipolar NRZ (ii) Polar NRZ (iii) Unipolar RZ (iv) bipolar RZ (v) Manchester code.	5
		Module 2	
3	(a)	Explain the Geometric representation of signals and express the energy of the signal in terms of the signal vector.	5
	(b)	Explain the Gram-Schmidt orthogonalization procedure.	5
	(c)	Explain the matched filter receiver with the relevant mathematical theory.	6
		OR	
4	(a)	Obtain the decision rule for Maximum likelihood decoding and explain the correlation receiver.	7
	(b)	The waveforms of four signals $s_1(t)$, $s_2(t)$, $s_3(t)$, and $s_4(t)$ described below. $s_1(t) = 1, \quad 0 < t < T/3,$ $s_2(t) = 1, \quad 0 < t < 2T/3,$ $s_3(t) = 1, \quad T/3 < t < T,$	9

Scanned with CamScanner

		$s_3(t) = 1$, $0 < t < T$, and zero otherwise. Using the Gram-Schmidt orthogonalization procedure, find an orthonormal basis for this set of signals and construct the corresponding signal-space diagram.	
		Module 3	
5	(a)	Define binary phase shift keying. Derive the probability of error of BPSK.	7
	(b)	Define M-ary QAM. Obtain the constellation of QAM for M=4 and draw the signal space diagram	4
	(c)	Given the input binary sequence 1100100001. Sketch the waveforms of the inphase and quadrature components of a modulated wave and next sketch the QPSK signal.	5
		OR	
6	(a)	Describe the QPSK signal with its signal space characterization. With a neat block diagram explain the generation and detection of QPSK signals.	6
	(b)	Obtain the expression probability of symbol error of coherent binary FSK.	7
	(c)	Illustrate the operation of DPSK for the binary sequence 10010011	3
		Module 4	
7	(a)	With a neat block diagram Explain the digital PAM transmission through bandlimited baseband channels and obtain the expression for ISI.	5
	(b)	What are adaptive equalizers? Explain the linear adaptive equalizer based on the MSE criterion.	6
	(c)	The binary sequence 100101100010 is the input to the precoder whose output is used to modulate a duobinary transmitting filter. Obtain the precoded sequence, transmitted amplitude levels, the received signal levels and the decoded sequence.	5
		OR	
8	(a)	What is eye pattern? What is the Nyquist criterion for zero ISI? Given an example of the pulse with zero ISI.	5
	(b)	Explain the design of bandlimited signals with controlled ISI. Describe the time domain and frequency domain characteristics of a duobinary signal.	5
	(c)	What is channel equalization? With a neat diagram explain the concept of equalization using a linear transversal filter.	6
		Module 5	
9	(a)	Draw the 4 stage linear feedback shift register with 1st and 4th stage is connected to Modulo-2 adder. Output of Modulo-2 is connected to 1st stage input. Find the output PN sequence and obtain the autocorrelation sequence.	6
	(b)	With a neat block diagram explain the frequency hopped spread spectrum.	7
	(c)	Explain the effect of dispreading on narrowband interference.	3
		OR	
10	(a)	Explain the generation of direct sequence spread spectrum signal with the relevant waveforms and spectrums.	6
	(b)	With a neat block diagram explain the CDMA system based on IS-95.	7
	(c)	Write a short note on application of spread spectrum in wireless LANs.	3

Scanned with CamScanner

SIGNALING OVER AWGN CHANNELS

Introduction:

Detection theory deals with the design and evaluation of a decision making receiver system, which observes the received signal corrupted by additive noise the most likely transmitted symbol. Since the effect of noise cannot be completely removed, the receiver is to be designed in such a way that the effect of noise is minimum.

Model Of a Digital Communication System:

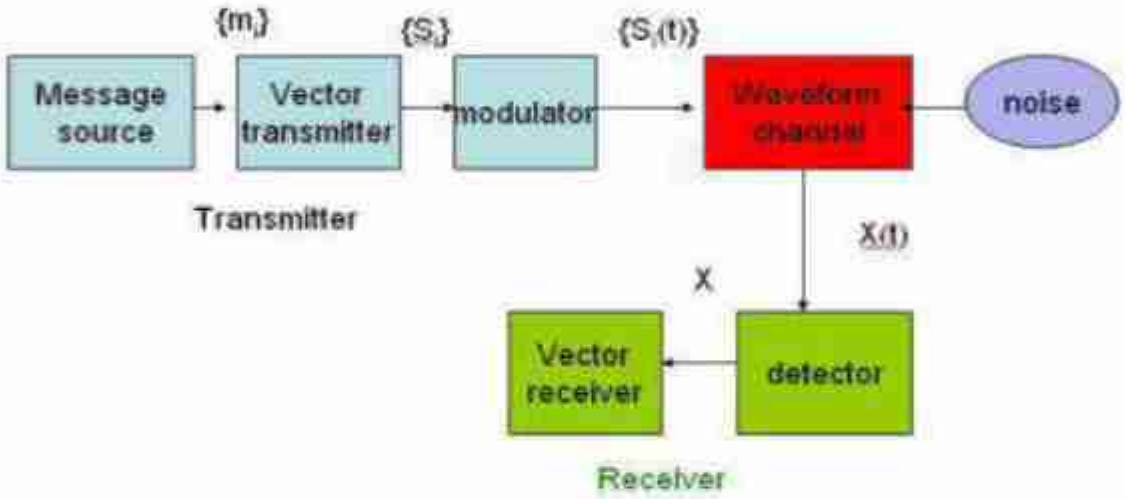


Fig: Block diagram of Digital Communication System

A digital communication system consists of six basic blocks. The functional blocks at the transmitter are responsible for processing the input message, encoding, modulating, and transmitting over the communication channel. The functional blocks at the receiver perform the reverse process to retrieve the original message.

The aim of a digital communication system is to transmit the message efficiently over the communication channel by incorporating various data compressions encoding and modulation techniques, in order to reproduce the message in the receiver with the least errors. The information input, which is generally in analog form, is digitized into a binary sequence, also known as an information sequence. *The* source encoder is responsible for compressing the input information sequence to represent it with less redundancy. The compressed data is passed to the channel encoder.

The channel encoder introduces some redundancy in the binary information sequence that can be used by the channel decoder at the receiver to overcome the effects of noise and interference encountered by the signal while in transit through the communication channel. Hence, the redundancy added in the information message helps in increasing the reliability of the data received and also improves the fidelity of the received signal. Thus, the channel encoder aids the receiver in decoding the desired information sequence. Some of the popular channel encoders are Low Density Parity Check codes, Turbo codes, Convolution codes,

and Reed-Solomon codes. The channel encoded data is passed to the channel modulator, which serves as the interface to the communication channel. The encoded sequence is modulated using suitable digital modulation techniques, i.e., Binary Phase Shift Keying (BPSK), Quadrature Phase Shift Keying (QPSK) and transmitted over the communication channel.

Geometric Representation Of Signals:

Geometric representation of signals can provide a compact characterization of signals and can simplify analysis of their performance as modulation signals.

Orthonormal bases are essential in geometry. Let {s1 (t), s2 (t), . . . , sM (t)} be a set of signals.

$\phi_1(t)$ and $\phi_2(t)$ are orthonormal if:

$$\int_0^{T_b} \phi_1(t)\phi_2(t)\mathrm{d}t = 0 \text{ (orthogonality)},$$

$$\int_0^{T_b} \phi_1^2(t)\mathrm{d}t = \int_0^{T_b} \phi_2^2(t)\mathrm{d}t = 1 \text{ (normalized to have unit energy)}.$$

$$\begin{aligned} s_1(t) &= s_{11}\phi_1(t) + s_{12}\phi_2(t), \\ s_2(t) &= s_{21}\phi_1(t) + s_{22}\phi_2(t). \end{aligned}$$

$$\text{where} \quad s_{ij} = \int_0^{T_b} s_i(t)\phi_j(t)\mathrm{d}t, \quad i,j \in \{1,2\},$$

$$\begin{aligned} s_1(t) &= s_{11}\phi_1(t) + s_{12}\phi_2(t), \\ s_2(t) &= s_{21}\phi_1(t) + s_{22}\phi_2(t), \\ s_{ij} &= \int_0^{T_b} s_i(t)\phi_j(t)\mathrm{d}t, \; i,j \in \{1,2\}, \end{aligned}$$

$\int_0^{T_b} s_i(t)\phi_j(t)\mathrm{d}t$ is the projection of $s_i(t)$ onto $\phi_j(t)$.

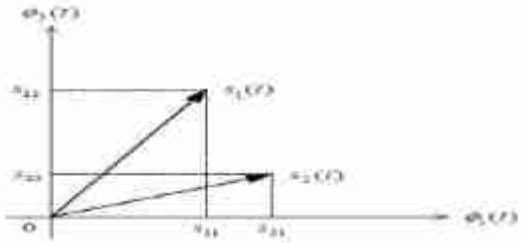


Fig: Geometric Representation of Signals

The representation of signals in an N-dimensional signal space is called as Signal Constellation. The above fig. illustrates the case of two-dimensional signal space.

Gram-Schmidt Orthogonalization Procedure:

Suppose we are given a signal set

$$\{s_1(t), \ldots, s_M(t)\}$$

Find the orthogonal basis functions for this signal set

$$\{\phi_1(t), \ldots, \phi_K(t)\}$$

Step 1: Construct the first basis function

Compute the energy in signal 1

$$E_1 = \int_{-\infty}^{\infty} s_1^2(t)dt$$

The first basis function is just a normalized version of s1 (t)

$$\phi_1(t) = \frac{1}{\sqrt{E_1}} s_1(t)$$

$$s_1(t) = s_{11}\phi_1(t) = \sqrt{E_1}\phi_1(t)$$

$$s_{11} = \int_{-\infty}^{\infty} s_1(t)\phi_1(t)dt = \sqrt{E_1}$$

Step 2: Construct the second basis function

Compute correlation between signal 2 and basic function 1

$$s_{21} = \int_{-\infty}^{\infty} s_2(t)\phi_1(t)dt$$

Subtract off the correlation portion

$$g_2(t) = s_2(t) - s_{21}\phi_1(t)$$

Compute the energy in the remaining portion

$$E_{g2} = \int_{-\infty}^{\infty} [g_2(t)]^2 \, dt$$

Normalize the remaining portion

$$\phi_2(t) = \frac{1}{\sqrt{E_{g2}}} g_2(t)$$

$$s_{22} = \int_{-\infty}^{\infty} s_2(t)\phi_2(t)dt = \sqrt{E_{g2}}$$

1st basis function is normalized version of the first signal. Successive basis functions are found by removing portions of signals that are correlated to previous basis functions and normalizing the result. This procedure is repeated until all basis functions are found. If gk=0, no new basis functions is added. The order in which signals are considered is arbitrary.

Conversion Of a Continuous AWGN Channel into a Vector:

Influence of the AWGN noise to the signal space concept

$$x(t) = s(t) + w(t)$$

where w(t) is zero-mean AWGN with PSD N0/2.

After the correlator at the receiver, we obtain:

$$\langle x(t), f_j(t) \rangle = \langle s_i(t), f_j(t) \rangle + \langle w(t), f_j(t) \rangle$$

Equivalently, $x_j = s_{ij} + w_j$

There is no "information loss" by the signal space representation.

Since {sij} is deterministic, the distribution of x is a mean shift of the distribution of w. Observe that w is Gaussian distributed because w(t) is AWGN. The distribution of w can, therefore, be determined by its mean vector and covariance matrix.

As a result, [w1, w2, …, wN] are zero-mean i.i.d. Gaussian distributed with variance N0/2. This shows that x is independent Gaussian distributed with common variance N0/2 and mean vector si = [si1, si2, …, siN]. Equivalently,

$$f(\boldsymbol{x} \mid \boldsymbol{s}_i) = \prod_{j=1}^{N} \frac{1}{\sqrt{\pi N_0}} \exp\left[-\frac{1}{N_0} (x_j - s_{ij})^2 \right]$$

It is possible that

$$w'(t) = w(t) - \sum_{i=1}^{N} w_i \cdot f_i(t) \neq 0$$

However, it can be shown that (as an error term) w'(t) is orthogonal to si(t) for 1 £ i £ M. Hence, w '(t) will not affect the decision error rate on message i.

<w'(t),si(t)> = 0 with probability 1.

Maximum Likelihood Decoding:

Decision regions for

N = 2 and M = 4

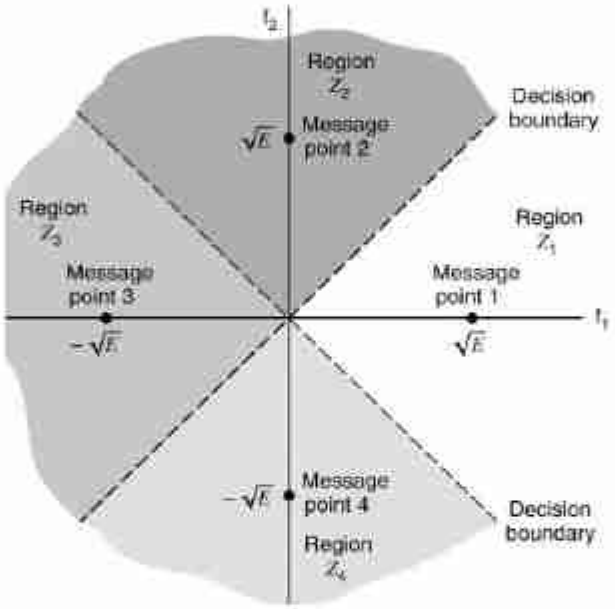


Fig: Maximum likelihood decoding

Usually, $s1$, s2, …, sM are named the message points. The received signal point x wanders about the transmitted message point in a Gaussian-distributed random fashion. In this case, the ML decision rule can be reduced to an inner-product.

$$\begin{aligned} d(\boldsymbol{x}) &= \arg\min_{1\le i\le M} \| \boldsymbol{x} - \boldsymbol{s}_i \|^2 \\ &= \arg\min_{1\le i\le M} \left(\| \boldsymbol{x} \|^2 - 2\langle \boldsymbol{x}, \boldsymbol{s}_i \rangle + \| \boldsymbol{s}_i \|^2 \right) \\ &= \arg\min_{1\le i\le M} \left(-2\langle \boldsymbol{x}, \boldsymbol{s}_i \rangle + E_i \right) \\ &= \arg\max_{1\le i\le M} \langle \boldsymbol{x}, \boldsymbol{s}_i \rangle, \text{ if } E_i \text{ is constant.} \end{aligned}$$

Correlation Receiver:

If signals do not have equal energy, we can use,

$$d(\boldsymbol{x}) = \arg\max_{1 \le i \le M} \left(\langle \boldsymbol{x}, \boldsymbol{s}_i \rangle - \frac{1}{2} E_i \right).$$

The receiver is coherent because the receiver requires to be in perfect synchronization with the transmitter (more specifically, the integration must begin at exactly the right time instance).

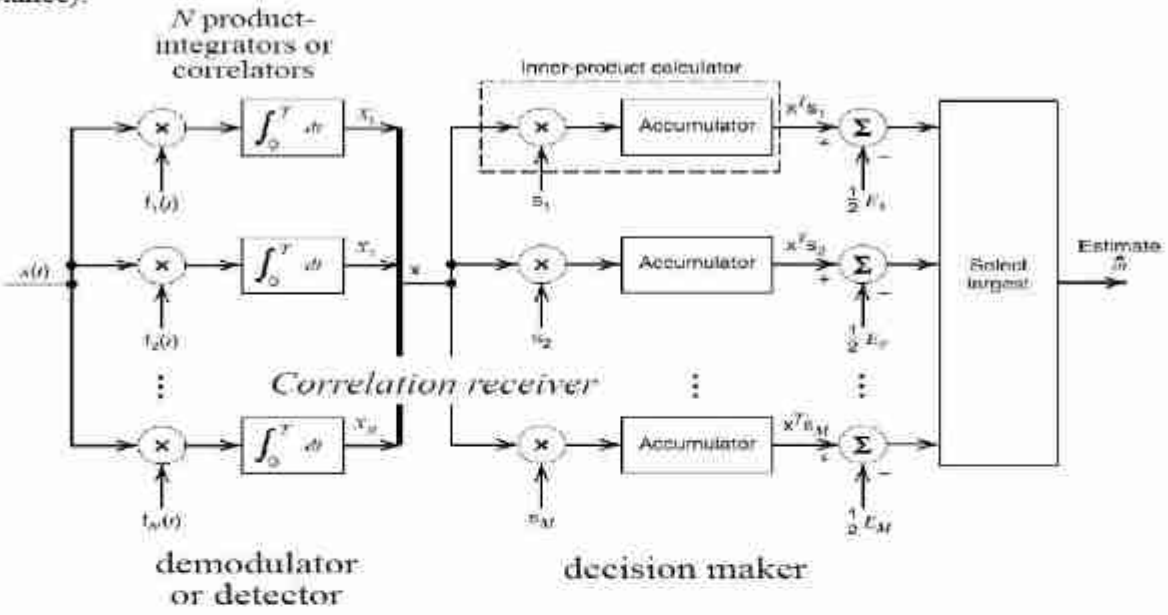


Fig: Correlation Receiver

Matched Filter Receiver:

The correlator and matched filter can be made equivalent. Specifically,

$$x_i = \int_0^T x(\tau) f_i(t) dt = \int_{-\infty}^{\infty} x(\tau) h_i(T - \tau) d\tau$$

if $h_i(t) = f_i(T - t)$ (and implicitly $f_i(t)$ is zero outside $0 \le t \le T$).

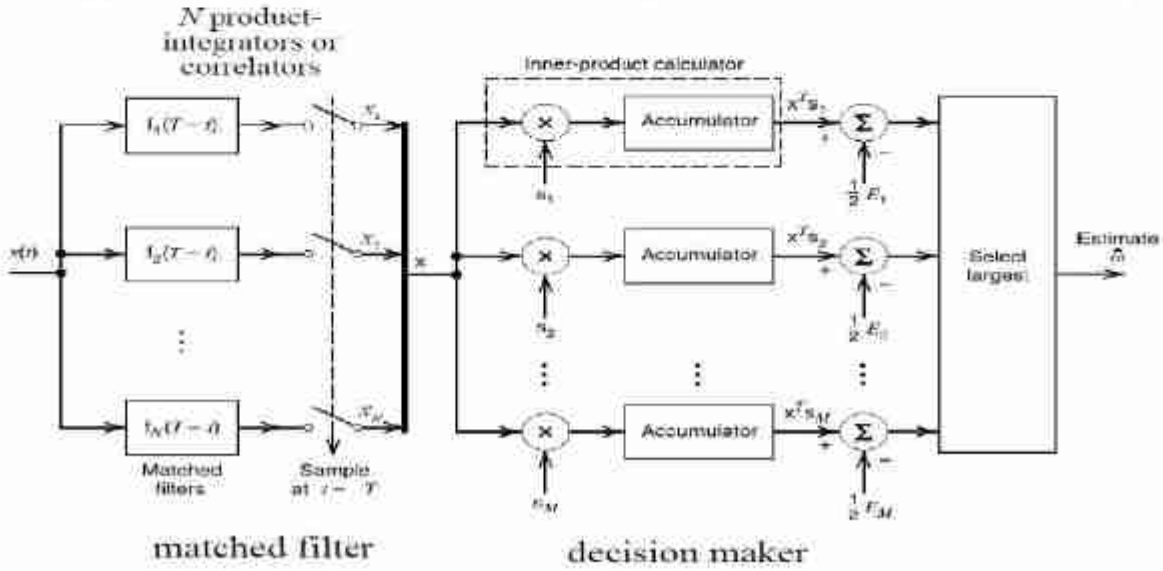


Fig: Matched Filter

Digital Modulation Techniques

Introduction:

Digital Modulation provides more information capacity, high data security, quicker system availability with great quality communication. Hence, digital modulation techniques have a greater demand, for their capacity to convey larger amounts of data than analog ones.

There are many types of digital modulation techniques and we can even use a combination of these techniques as well. In this chapter, we will be discussing the most prominent digital modulation techniques.

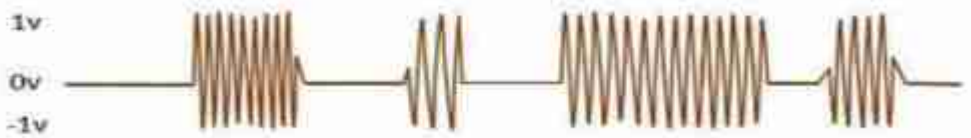
if the information signal is digital and the amplitude (IV of the carrier is varied proportional to the information signal, a digitally modulated signal called amplitude shift keying (ASK) is produced.

If the frequency (f) is varied proportional to the information signal, frequency shift keying (FSK) is produced, and if the phase of the carrier (0) is varied proportional to the information signal, phase shift keying (PSK) is produced. If both the amplitude and the phase are varied proportional to the information signal, quadrature amplitude modulation (QAM) results. ASK, FSK, PSK, and QAM are all forms of digital modulation

Amplitude Shift Keying: The amplitude of the resultant output depends upon the input data whether it should be a zero level or a variation of positive and negative, depending upon the carrier frequency.

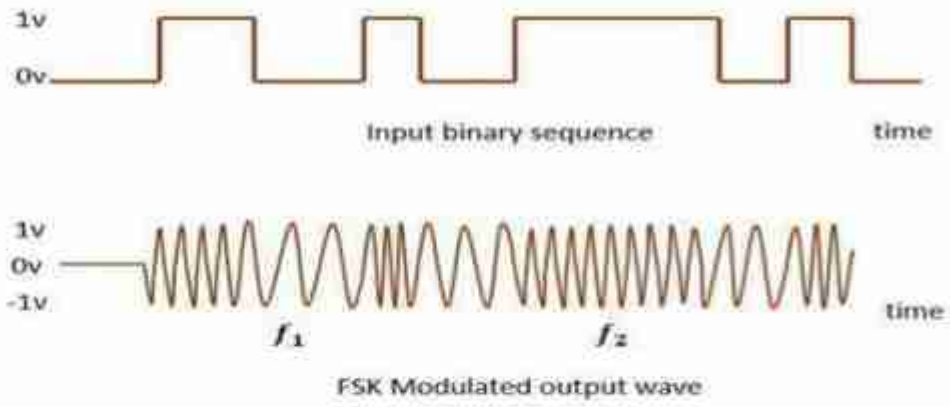


Input binary sequence



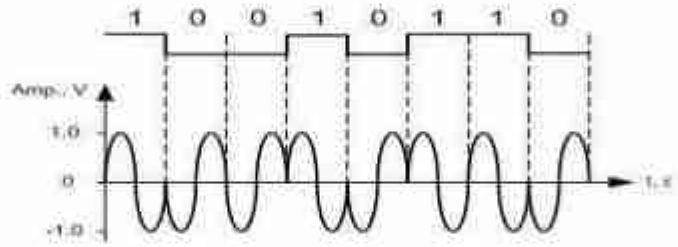
ASK Modulated output wave

Frequency Shift Keying: FSK is the digital modulation technique in which the frequency of the carrier signal varies according to the discrete digital changes. FSK is a scheme of frequency modulation.



FSK Modulated output wave

Phase Shift Keying: The phase of the output signal gets shifted depending upon the input. These are mainly of two types, namely BPSK and QPSK, according to the number of phase shifts. The other one is DPSK which changes the phase according to the previous value.



Phase shift keying (PSK)

Binary Phase Shift Keying [BPSK]:

For $M = 2$, we obtain $\phi_m \in \{0, \pi\}$ and thus the two waveforms

$$s_1(t) = \sqrt{2P}\cos(2\pi f_0 t),$$
$$s_2(t) = \sqrt{2P}\cos(2\pi f_0 t + \pi) = -s_1(t)$$

Orthonormal functions:

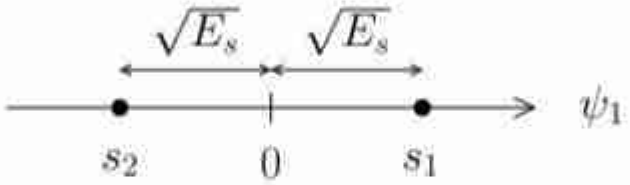
$$\psi_1(t) = \frac{1}{\sqrt{E_s}} s_1(t) = \sqrt{\frac{2}{T}} \cos(2\pi f_0 t)$$

for $t \in [0, T]$.

Geometrical representation of the waveform,

$$s_1(t) = +\sqrt{PT}\psi_1(t) \quad \longleftrightarrow \quad s_1 = +\sqrt{PT} = +\sqrt{E_s}$$
$$s_2(t) = -\sqrt{PT}\psi_1(t) \quad \longleftrightarrow \quad s_2 = -\sqrt{PT} = -\sqrt{E_s}$$

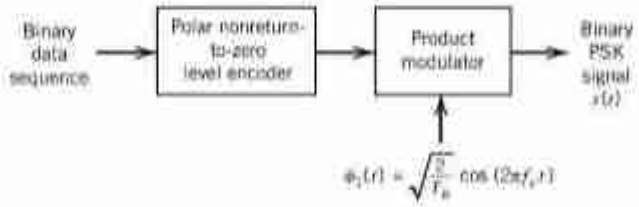
Signal constellation,



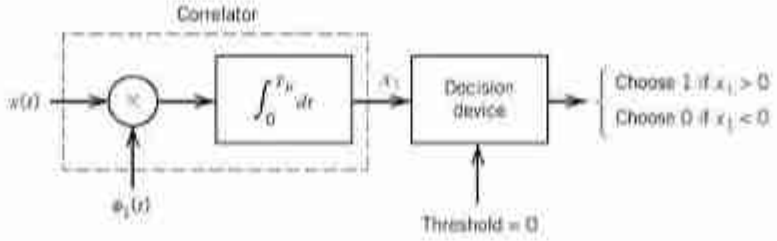
Vector encoder,

$$x = \text{enc}(u) = \begin{cases} s_1 & \text{for } u = 1, \\ s_2 & \text{for } u = 0. \end{cases}$$

Generation & Detection of BPSK Signal:



Polar NRZ-Level encoder represents symbol 1 & 0 of incoming binary sequence. Product modulator multiplies the output of the NRZ polar encoder by the basis function.



Let x(t) be the received signal which is the sum of AWGN noise & transmitted signal i.e x(t)=s(t)+w(t).

if x1>0, the decision is taken in favour of symbol 1

if x1<0, the decision is taken in favour of symbol 0.

Coherent Binary Frequency Shift Keying:

Frequency Shift Keying (FSK) modulation is a popular form of digital modulation used in low-cost applications for transmitting data at moderate or low rate over wired as well as wireless channels. In general, an M-ary FSK modulation scheme is a power efficient modulation scheme and several forms of M-ary FSK modulation are becoming popular for spread spectrum communications and other wireless applications. In this lesson, our discussion will be limited to binary frequency shift keying (BFSK).

Two carrier frequencies are used for binary frequency shift keying modulation. One frequency is called the 'mark' frequency (f2) and the other as the space frequency (f1). By convention, the 'mark' frequency indicates the higher of the two carriers used. If Tb indicates the duration of one information bit, the two time-limited signals can be expressed as :

$$s_i(t) = \begin{cases} \sqrt{\frac{2E_b}{T_b}} \cos 2\pi f_i t, & 0 \le t \le T_b, i = 1,2 \\ 0, \text{ elsewhere.} \end{cases}$$

The binary scheme uses two carriers and for special relationship between the two frequencies one can also define two orthonormal basis functions as shown below.

$$\varphi_i(t) = \sqrt{\frac{2}{T_b}} \cos 2\pi f_i t \quad ; \quad 0 \le t \le T_b \text{ and } j = 1,2$$

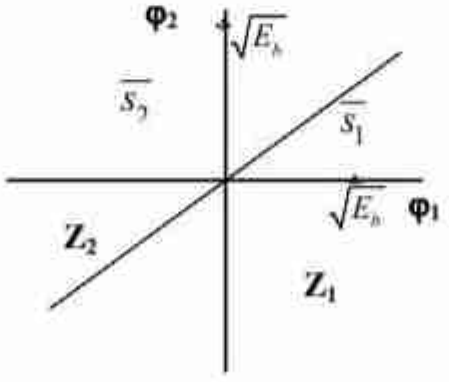
$$s_{ij} = \int_0^{T_b} s_i(t)\varphi_j(t)\,dt = \begin{cases} \sqrt{E_b}, i = j \\ 0, \quad i \neq j \end{cases} \quad ; \quad \text{i} = 1,2 \quad \text{and j} = 1,2$$

$$\text{i.e.} \quad \left.\begin{aligned} s_{11} = s_{22} = \sqrt{E_b} \\ s_{12} = s_{21} = 0 \end{aligned}\right| \begin{aligned} i &= 1,2 \\ j &= 1,2 \end{aligned}$$

So, the two signal victors can be expressed as:

$$\overline{s_1} = \begin{bmatrix} \sqrt{E_b} \\ 0 \end{bmatrix} \quad \text{and} \quad \overline{s_2} = \begin{bmatrix} 0 \\ \sqrt{E_b} \end{bmatrix}$$

Please note that one can generate an FSK signal without following the above concept of orthogonal carriers and that is often easy in practice. possible FSK modulated waveform. Notice the waveform carefully and verify if the two carriers are orthogonal. An obvious feature of an FSK modulated signal, analogous to analog FM signal is that envelop of the modulated signal is constant. All modulation schemes which exhibit constant envelope, are preferable for high power digital transmission because, operation of the power amplifier in a non-linear region may not produce considerable harmonics. conceptual diagram for generating binary FSK modulated signal. Note that the input random binary sequence is represented by '1' and '0' where '0' represents no voltage at the input of the multipliers. A '0' input to the inverter results in a '1' at its output. That is, the inverter, along with the two multipliers and the summing unit, may be thought to behave as a 'switch' which selects output of one of the two oscillators.



Generation & Detection of BFSK Signal:

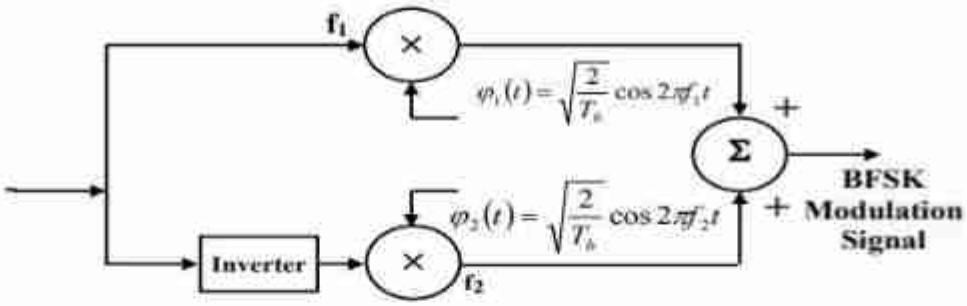
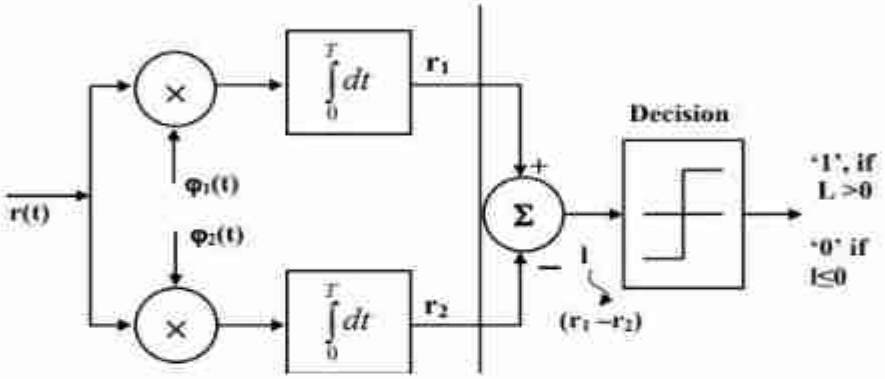


Fig shows the block diagram of BFSK generation, input binary data in unipolar NRZ format is applied to the top modulator & also to bottom product modulator through the inverter.



The detector consists of two correlators that are individually tuned to 2 different frequencies. The output of top & bottom correlator be denoted b x1 & x2 respectively. The x1 & x2 are given to the subtractor. The output of subtractor l=x1-x2.

If l>0, the decision is taken in favour of symbol 1

If l<0, the decision is taken in favour of symbol 0.

Quadrature Phase Shift Keying:

In digital modulation techniques a set of basis functions are chosen for a particular modulation scheme. Generally the basis functions are orthogonal to each other. Basis functions can be derived using 'Gram Schmidt orthogonalization procedure. Once the basis function are chosen, any vector in the signal space can be represented as a linear combination of the basis functions. In Quadrature Phase Shift Keying (QPSK) two sinusoids (sin and cos) are taken as basis functions for modulation. Modulation is achieved by varying the phase of the basis functions depending on the message symbols. In QPSK, modulation is symbol

based, where one symbol contains 2 bits. The following equation outlines QPSK modulation technique

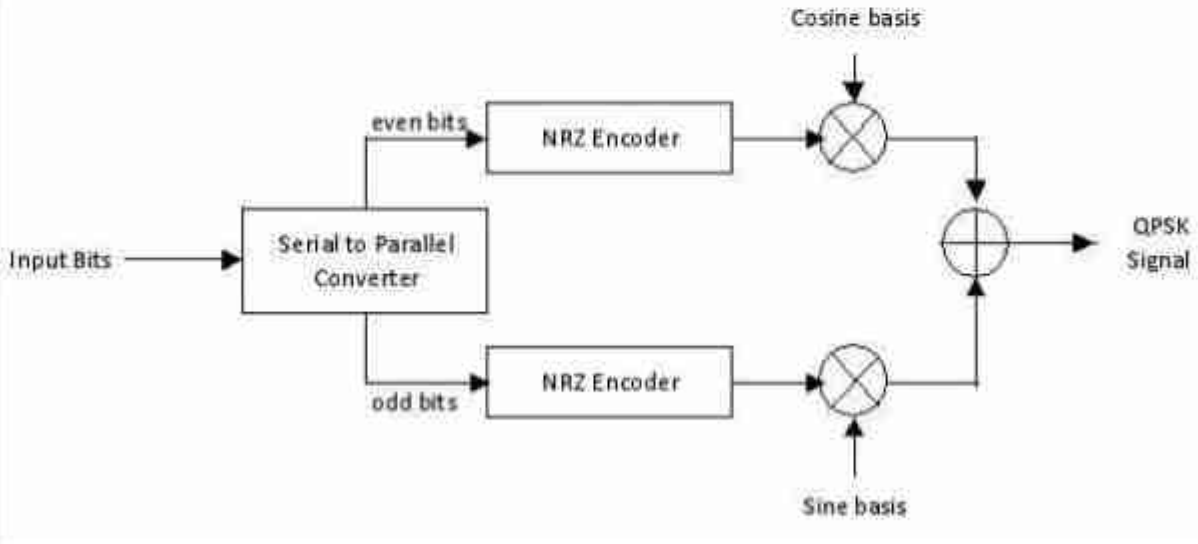
$$s_i(t)=2E_sT\sqrt{\cos(2\pi f_c t+(2n-1)\pi 4)},n=1,2,3,4$$

When n=1, the phase shift is 45 degrees. This is called pi/4 QPSK. The constellation diagram of QPSK will show the constellation points lying on both x and y axes. This means that the QPSK modulated signal will have an in-phase component (I) and also a quadrature component (Q). This is because it has only two basis functions.

A QPSK modulator can be implemented as follows. A de-multiplexer (or serial to parallel converter) is used to separate odd and even bits from the generated information bits. Each of the odd bits (quadrature arm) and even bits (in-phase arm) are converted to NRZ format in a parallel manner.

Generation and detection of QPSK:

The signal on the in-phase arm is multiplied by cosine component and the signal on the quadrature arm is multiplied by sine component. QPSK modulated signal is obtained by adding the signal from both in-phase and quadrature arm

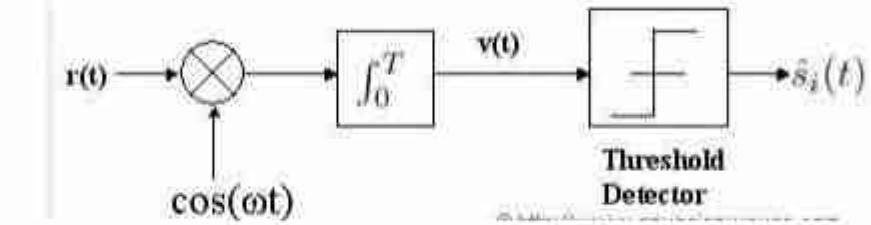


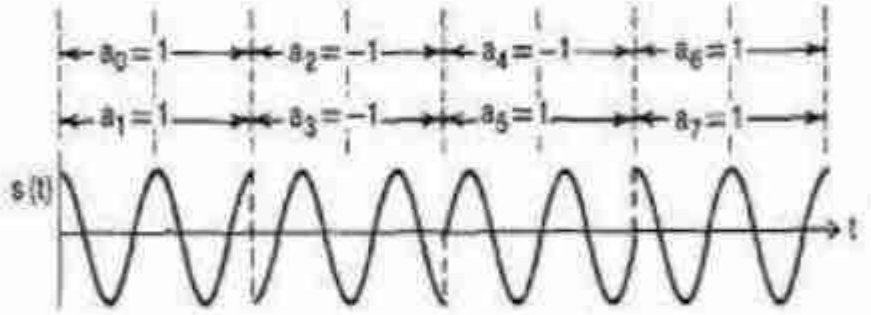
For QPSK demodulator, a coherent demodulator is taken as an example. In coherent detection technique the knowledge of the carrier frequency and phase must be known to the receiver. This can be achieved by using a PLL (phase lock loop) at the receiver. A PLL

essentially locks to the incoming carrier frequency and tracks the variations in frequency and phase. For the following simulation, a PLL is not used but instead we simple use the output of the PLL.

For demonstration purposes we simply assume that the carrier phase recovery is done and simply use the generated reference frequencies at the receiver (cos(ωt)) and (sin(ωt))

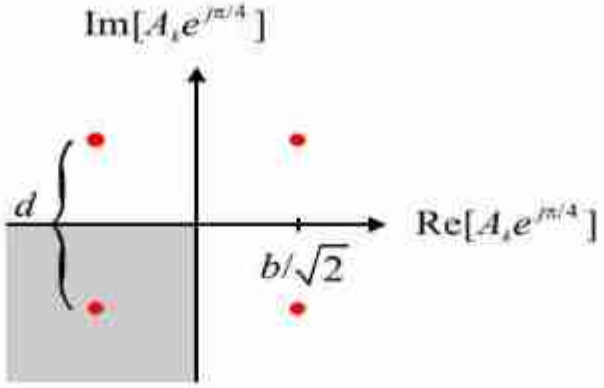
In the demodulator the received signal is multiplied by a reference frequency generators (cos(ωt)) and (sin(ωt)) on separate arms (in-phase and quadrature arms). The multiplied output on each arm is integrated over one bit period using an integrator. A threshold detector makes a decision on each integrated bit based on a threshold. Finally the bits on the in-phase arm (even bits) and on the quadrature arm (odd bits) are remapped to form detected information stream. Detector for in-phase arm is shown below. For quadrature arm the below architecture remains same but sin(ωt) basis function must be used instead.





Symbol error probability of QPSK:

Assuming optimum decision regions, rotation of the constellation doesn't affect the error probabilities in any way. Therefore, we can consider the following QPSK constellation rotated by 45°:



The noise samples are now also rotated, M Z e k k j = π/4. With the earlier assumptions, the noise statistics are not affected, so the real and imaginary parts of the noise are still independent and they have variance σ2.

In case of Ak = −b , the correct decision corresponds to the shaded region of the above figure, i.e.,

$$\begin{aligned}
&\Pr(\textit{correct decision at time } k \mid A_k = -b) \\
&= \Pr(\mathrm{Re}[M_k] < d/2,\ \mathrm{Im}[M_k] < d/2) \\
&= \Pr(\mathrm{Re}[M_k] < d/2) \cdot \Pr(\mathrm{Im}[M_k] < d/2) \\
&= \left(1 - Q\left(\frac{d}{2\sigma}\right)\right)^2 \\
&= 1 - 2Q\left(\frac{d}{2\sigma}\right) + Q^2\left(\frac{d}{2\sigma}\right)
\end{aligned}$$

All the symbols have the same error probabilities, so

$$\begin{aligned}
\Pr(\textit{symbol error at time } k) &= \Pr(\textit{symbol error at time } k \mid A_k = -b) \\
&= 1 - \Pr(\textit{correct decision at time } k \mid A_k = -b) \\
&= 2Q\left(\frac{d}{2\sigma}\right) - Q^2\left(\frac{d}{2\sigma}\right) \\
&\cong 2Q\left(\frac{d}{2\sigma}\right)
\end{aligned}$$

In practical cases, it can be assumed to be less than 0.1, so the approximation error is no more than 5 %.

M-ary PSK:

Multi-level modulation techniques permit high data rates within fixed bandwidth constraints. A convenient set of signals for M-ary PSK is

$$\phi_i(t) = A\cos(\omega_c t + \theta_i), \qquad 0 < t \leq T_s,$$

where the M phase angles are

$$\theta_i = 0, \frac{2\pi}{M}, \ldots, \frac{2(M-1)\pi}{M}.$$

For equiprobable ones and zeros the PSD for M-ary PSK is

$$S_\phi(\omega) = A^2 T_s \mathrm{Sa}^2\left[(\omega - \omega_c)\frac{T_s}{2}\right].$$

The symbols in this case are of duration Ts , so the information (or bit) rate Tb satisfies

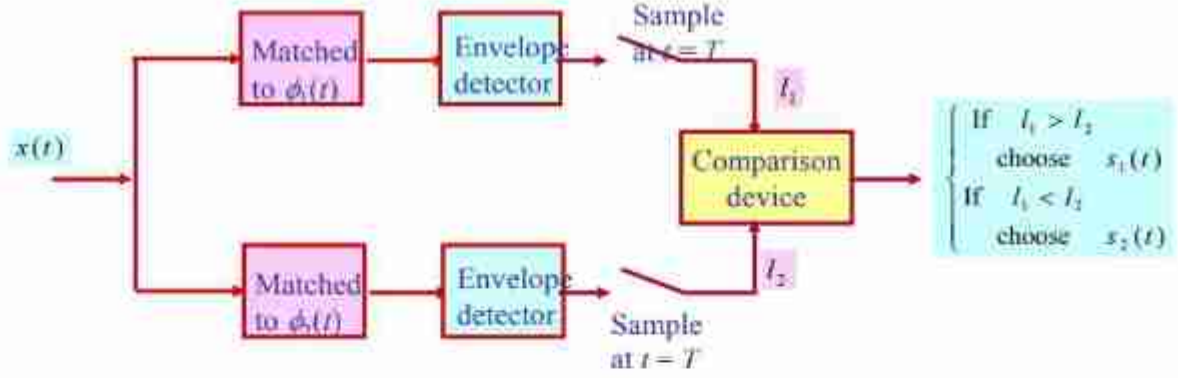
$$T_s = T_b \log_2 M.$$

The potential bandwidth efficiency of M-ary PSK can be shown to be

$$\frac{f_b}{B} = \log_2 M \text{ bps/Hz}.$$

Non-Coherent Binary FSK:

The receiver tries to discriminate between s1(t) and s2(t), regardless of the carrier phase. This goal can be achieved by the following receiver structure:



The average probability of error for the non-coherent receiver,

$$P_e = \frac{1}{2}\exp\left(-\frac{E}{2N_0}\right)$$

where E is the signal energy per symbol and N0/2 is the noise spectral density.

Differential Phase-Shift Keying:

DPSK eliminates the need for a coherent reference signal at the receiver by combining two basic operations at transmitter.

Transmitter's two operations:

(1) Differential encoding of the input binary sequence and

(2) phase-shift keying

Generation of DPSK:

For an input binary sequence {bk} , a differential encoded sequence {dk} is determined by,

$$d_k = d_{k-1} \oplus b_k \quad \text{or} \quad d_k = \overline{d}_{k-1} \oplus \overline{b}_k$$

index k:		0	1	2	3	4	5	6	7	8	9	10	
	$\{b_k\}$		1	1	0	1	0	1	1	0	0	1	
Differentially encoded sequence	$\{d_k\}$	ref. bit 1	1	1	0	0	1	1	1	0	1	1	$d_k = \bar{d}_{k-1} \oplus \bar{b}_k$
Corresponding phase shift	$\{\theta(k)\}$		π	π	0	0	π	π	π	0	π	π	

Table 1. Illustrating the generation of DPSK signal

DPSK is an another example of noncoherent orthogonal modulation, when it is considered over two bit intervals. In this case, from (1) we get the average probability of error for DPSK is

$$P_e = \frac{1}{2}\exp\left(-\frac{E_b}{N_0}\right)$$

Communication through Band Limited Channels

Introduction:

In base band data transmission, transmitted pulses tend to spread during transmission. The pulse spreading causes overlapping of pulses into adjacent time slot. This may result in an error at the receiver.

The phenomenon of pulse spreading & resultant difficulty in discriminating the symbols at the receiver is termed as Inter symbol interference ISI.

Digital PAM Transmission through Band Limited Channels:

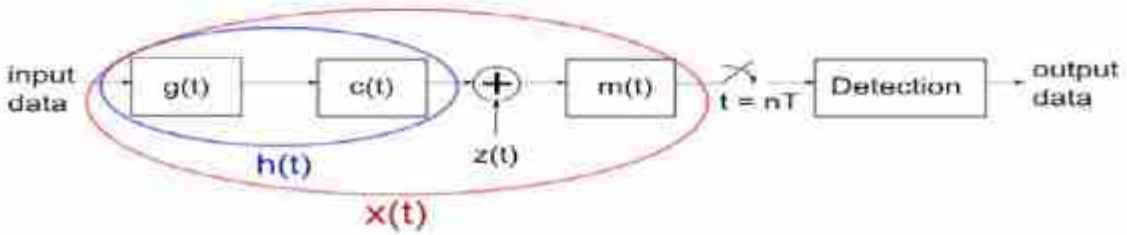
Analysis in previous chapters considered the channel bandwidth to be unbounded, All physical channels are band limited, with C(f) = 0 for |f| > W. Non-distorting (ideal) channel: |C(f)| = const. for | f | < W and is linear. All other channels are non-ideal (distort the signal in amplitude, phase or both)

The impulse modulator,

$$s^*(t) = \sum_{k=-\infty}^{\infty} a_k \delta(t - kT)$$

The band-limiting transmit filter,

$$s(t) = \sum_{k=-\infty}^{\infty} a_k g_T(t - kT)$$



The equivalent discrete-time model:

$$y_k = \sum_{n=0}^{\infty} I_n x_{k-n} + v_k, \qquad k = 0, 1, \ldots$$

This can be expressed as:

$$y_k = x_0 \left(I_k + \frac{1}{x_0} \sum_{\substack{n=0 \\ n \neq k}}^{\infty} I_n x_{k-n} \right) + v_k, \qquad k = 0, 1, \ldots$$

With X0=1:

$$y_k = I_k + \sum_{\substack{n=0 \\ n \neq k}}^{\infty} I_n x_{k-n} + v_k$$

The equivalent discrete-time model:

$$y_k = \sum_{n=0}^{\infty} I_n x_{k-n} + v_k, \qquad k = 0, 1, \ldots$$

This can be expressed as:

$$y_k = x_0 \left(I_k + \frac{1}{x_0} \sum_{\substack{n=0 \\ n \neq k}}^{\infty} I_n x_{k-n} \right) + v_k, \qquad k = 0, 1, \ldots$$

With X0=1:

$$y_k = I_k + \sum_{\substack{n=0 \\ n \neq k}}^{\infty} I_n x_{k-n} + v_k$$

ISI!

Channel is assumed to be ideal band-limited

Task: to find pulse shapes that will

- Result in zero ISI at the receiver
- Allow us to achieve maximum possible transmission rate with zero ISI (optimum utilization of the given bandwidth)

The condition for zero ISI:

$$y_k = I_k + \sum_{\substack{n=0 \\ n \neq k}}^{\infty} I_n x_{k-n} + v_k \quad \Longrightarrow \quad x(t = kT) \equiv x_k = \begin{cases} 1 & k = 0 \\ 0 & k \neq 0 \end{cases}$$

Theorem (Nyquist):

The necessary and sufficient condition for x(t) to satisfy

$$x(nT) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}$$

is that its Fourier transform X(f) satisfies

$$\sum_{m=-\infty}^{\infty} X(f + m/T) = T$$

The replicas of X(f) obtained by sampling x(t) should add up to form a flat spectrum (a delta in time domain)

The Nyquist Criterion for No ISI:

The impulse response samples are

$$g(nT) = \frac{1}{2\pi}\int_{-\infty}^{\infty} G(\omega)e^{j\omega nT}\,d\omega$$

Let ωs = 2πfs = 2π/T. Then

$$g(nT) = \sum_{k=-\infty}^{\infty}\frac{1}{\omega_s}\int_{-\frac{\omega_s}{2}-k\omega_s}^{\frac{\omega_s}{2}-k\omega_s}\frac{1}{T}G(\omega)e^{j\omega nT}\,d\omega$$

$$= \sum_{k=-\infty}^{\infty}\frac{1}{\omega_s}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}}\frac{1}{T}G(\omega - k\omega_s)e^{j(\omega-k\omega_s)nT}\,d\omega$$

Recognizing that e −jknωsT = e −kn2π = 1 and taking the sum inside the integral gives

$$g(nT) = \frac{1}{\omega_s}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} G^*(\omega)e^{j\omega nT}\,d\omega$$

Where,

$$G^*(\omega) = \frac{1}{T}\sum_{k=-\infty}^{\infty} G(\omega - k\omega_s)$$

The function G* (ω) is called the aliased or folded spectrum.

There is no ISI if and only if,

$$G^*(\omega) = 1$$

Because then the integral at the bottom of the previous slide becomes

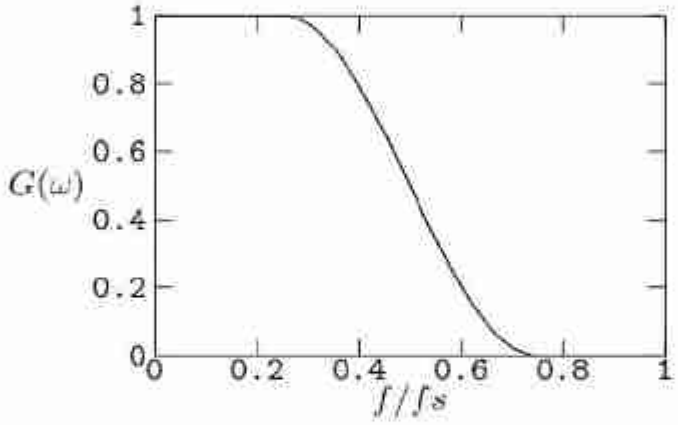
$$g(nT) = \frac{1}{\omega_s}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} e^{j\omega nT}\,d\omega = \delta[n]$$

Raised Cosine Baseband Shaping Filters:

The frequency response of a raised cosine filter is,

$$G(\omega) = \begin{cases} T & \text{for } |\omega| \leq (1-\alpha)\frac{\omega_s}{2} \\ \frac{T}{2}\left\{1 - \sin\left[\frac{T}{2\alpha}\left(|\omega| - \frac{\omega_s}{2}\right)\right]\right\} & \text{for } (1-\alpha)\frac{\omega_s}{2} \leq |\omega| \leq (1+\alpha)\frac{\omega_s}{2} \\ 0 & \text{elsewhere} \end{cases}$$

$\alpha \in [0, 1]$ is the excess bandwidth factor,



The corresponding impulse response is

$$g(t) = \frac{\sin\left(\frac{\omega_s}{2}t\right)}{\frac{\omega_s}{2}t} \frac{\cos\left(\alpha\frac{\omega_s}{2}t\right)}{1 - 4(\alpha t/T)^2}$$

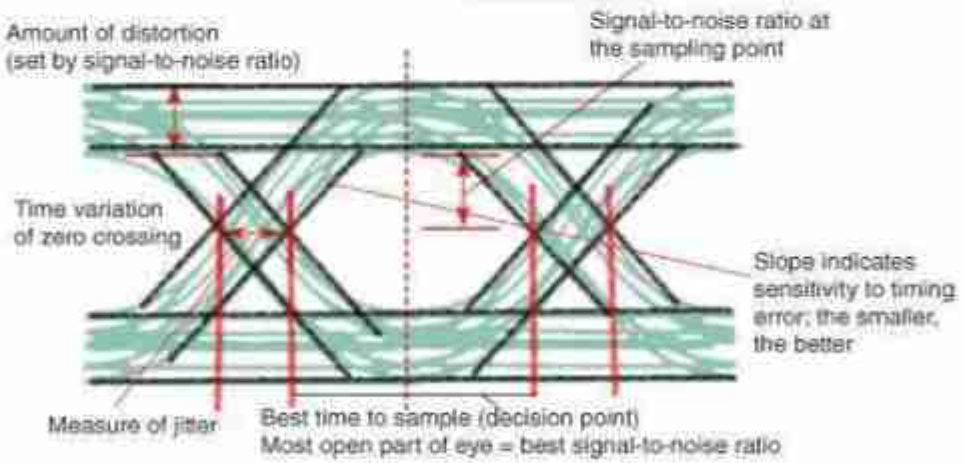
Special Case:

- If $\alpha = 0$, the raised cosine filter becomes an ideal flat low pass filter with cut-off frequency $\omega s/2$
- If $\alpha = 1$, the frequency response has no flat region and is one cycle of a cosine function raised up so it becomes 0 at the cut-off frequency of ω
- As the bandwidth is increased by making α closer to 1, the impulse response decays more rapid

Eye Diagram:

An eye diagram is a common indicator of the quality of signals in high-speed digital transmissions. An oscilloscope generates an eye diagram by overlaying sweeps of different segments of a long data stream driven by a master clock. The triggering edge may be positive or negative, but the displayed pulse that appears after a delay period may go either way; there is no way of knowing beforehand the value of an arbitrary bit. Therefore, when many such transitions have been overlaid, positive and negative pulses are superimposed on each other. Overlaying many bits produces an eye diagram, so called because the resulting image looks like the opening of an eye. In an ideal world, eye diagrams would look like rectangular boxes. In reality, communications are imperfect, so the transitions do not line perfectly on top of each other, and an eye-shaped pattern results. On an oscilloscope, the shape of an eye diagram will depend upon various types of triggering signals, such as clock triggers, divided clock triggers, and pattern triggers. Differences in timing and amplitude from bit to bit cause the eye opening to shrink.

It can indicate the best point for sampling, divulge the SNR (signal-to-noise ratio) at the sampling point, and indicate the amount of jitter and distortion. Additionally, it can show the time variation at zero crossing, which is a measure of jitter.



Eye diagrams provide instant visual data that engineers can use to check the signal integrity of a design and uncover problems early in the design process. Used in conjunction with other measurements such as bit-error rate, an eye diagram can help a designer predict performance and identify possible sources of problems.

Correlative Coding:

Correlative level coding is used to transmit a baseband signal with the signalling rate of 2Bo over the channel of bandwidth Bo. This is made physically possible by allowing ISI in the transmitted in controlled manner. This ISI is known to receiver. The correlative coding is implemented by duo-binary signalling and modified duo-binary signalling.

Duo binary signalling:

- The idea of correlative level coding is illustrated by considering the duo binary signalling
- "duo" means doubling of the transmission capacity of a straight binary system.

Encoding Process:

1) a_n = binary input bit; $a_n \in \{0,1\}$.

2) b_n = NRZ polar output of Level converter in the pre-coder and is given by,

$$b_n = \begin{cases} -d, \ if \ a_n = 0 \\ +d, \ if \ a_n = 1 \end{cases}$$

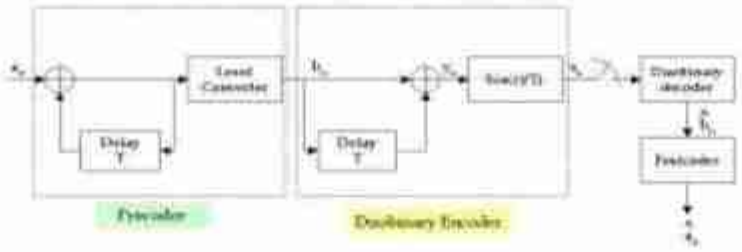
3) y_n can be represented as,

$$y_n = b_n + b_{n-1} = \begin{cases} 2d, & if \ a_n = a_{n-1} = 1 \\ 0, & if \ a_n \neq a_{n-1} \\ -2d, & if \ a_n = a_{n-1} = 0 \end{cases}$$

Note that the samples b_n are uncorrelated (i.e either +d for "1" or -d for "0" input). On the other-hand, the samples y_n are correlated (i.e. there are three possible values +2d,0,-2d depending on a_n and a_{n-1}). Meaning that the duo-binary encoding correlates present sample a_n and the previous input sample a_{n-1}.

4) From the diagram, impulse response of the duo-binary encoder is computed as,

$$h(t) = sinc\left(\frac{t}{T}\right) + sinc\left(\frac{t-T}{T}\right)$$



Decoding Process:

5) The receiver consists of a duo-binary decoder and a post-coder (inverse of pre-coder).The duo-binary decoder implements the following equation (which can be deduced from the equation given under step 3

$$\hat{b}_n = y_n - \hat{b}_{n-1}$$

This equation indicates that the decoding process is prone to error propagation as the estimate of present sample relies on the estimate of previous sample. This error propagation is avoided by using a pre-coder before duo-binary encoder at the transmitter and a post-coder after the duo-binary decoder. The pre-coder ties the present sample and previous sample (correlates these two samples) and the post-coder does the reverse process.

6) The entire process of duo-binary decoding and the post-coding can be combined together as one algorithm. The following decision rule is used for detecting the original duo-binary signal samples $\{a_n\}$ from $\{y_n\}$

$$\begin{aligned} &if\ y_n < d,\ \ then\ \hat{a}_n = 1 \\ &if\ y_n > d,\ \ then\ \hat{a}_n = 0 \\ if\ y_n = 0,\ \ &randomly\ guess\ \hat{a}_n \end{aligned}$$

Modified Duo-binary encoding:

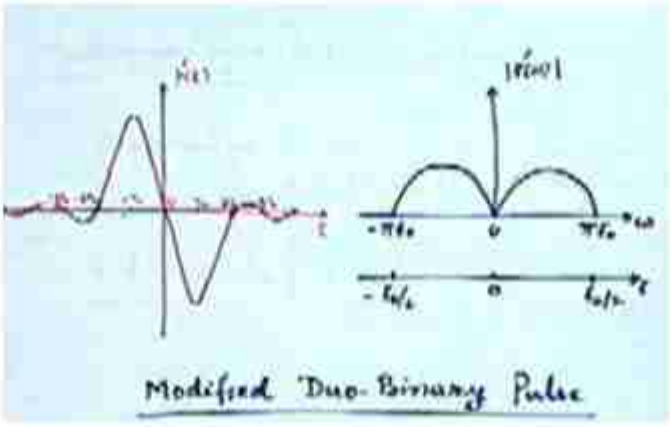
Now this is precisely the problem that Lender addressed himself to again and came up with what he called a modified duo binary system which has practically all the nice features of the duo binary scheme that we have just discussed and in addition it introduces a spectral null at omega equal to 0 right.

So basically the motivation is that we would like to have the properties of the duo binary scheme repeated plus a spectral null at DC right, which is desirable, now let me motivate it like this, if you were to look at the duo binary pulse shapes structure a bit carefully you would notice that it is really a composite function, this pulse shape p t has a composite pulse shape consisting of two nyquist pulses, two sinc pulses separated by T0 seconds.

Remember that, one of amplitude f0 or not f0 it is amplitude is some 2 f0 by pi or something right whatever the amplitude it does not matter, both of the same amplitude but displaced from each other by T0 seconds, only the sampling instants are different, chosen to be different. This in fact forms the basis of designing other pulse shapes in general right, we could select a linear combination of such time shifted nyquist pulses or sinc pulses to construct other kinds of partial response signals right.

And that is the basic idea of also that modify duo binary system as well as many other partial response systems which people suggested later right, so let us therefore construct a pulse shape p prime t which has the desirable property of having a spectral null and you also know what we should do to have spectral null, we have done that discussion before, the area under the pulse should be 0 right.

Takes two sinc pulses, one located at minus T0 other located at plus T0 and subtracts them out of course this is a pulse shape defined for which is centred at t equal to 0 for defining the pulse shape at t equal to 0 centre the T equal to 0 what is taking is the difference between a sinc pulse located at minus T0 and another at plus T0 okay, if you were to, maybe we can do that later and I can first show you what the pulse shape will look like right



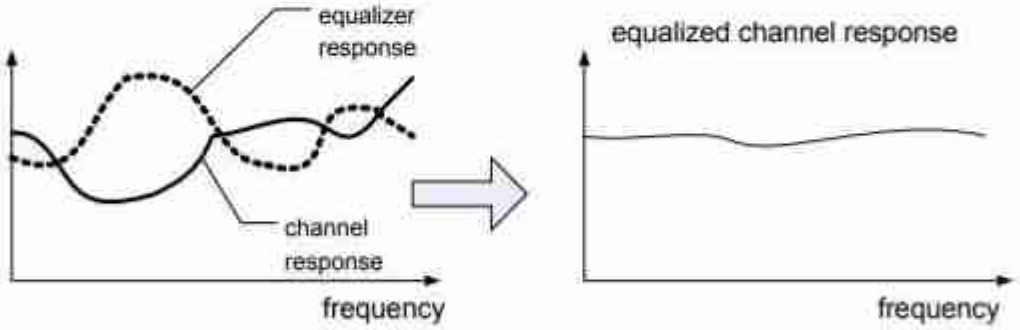
Obviously at T equal to 0 it will go through a 0 just like the Manchester coding goes to a transition in the middle of the pulse right only thing is instead of a sharp transition like that you have a gradual transition like that right because you are subtracting two sinc pulses the difference is like that right and we would like it to be like this so that the spectrum is limited, bandwidth is limited in a, when we discuss Manchester coding we were not worried about the total limitation, absolute limitation of bandwidth to within the nyquist band right.

Channel Equalization:

Using discrete time representation, the received signal is a filtered and noise-corrupted version of the transmitted sequence:

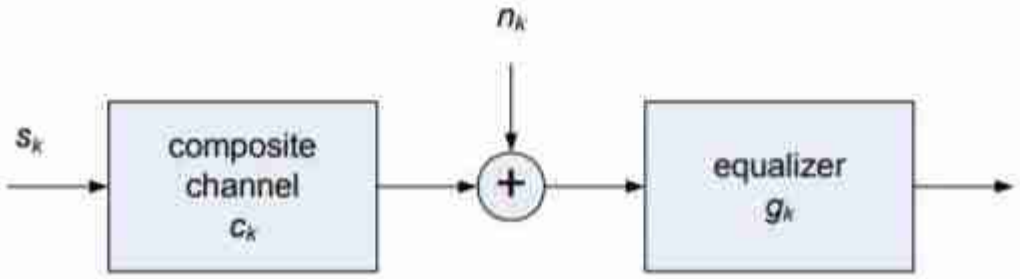
$$r_k = s_k \otimes c_k + n_k$$

- The multipath channel causes frequency selectivity and ISI
- Equalization can reduce the ISI and noise effects for better demodulation.



Linear Equalization:

A linear equalizer is a filter that can undo the channel effect



- Ideally, the output of an equalizer is a delayed version of the transmitted signal
- A fixed equalizer measures the time-invariant channel and compensates the frequency selectivity during the entire transmission of data
- An adaptive equalizer adjusts its coefficients to track a slowly time-varying channel

Mathematically, the output of an N-tap equalizer {g0k, ⋯ , gNk} is given by,

$$\widehat{s}_{k-k_o} = \sum_{n=0}^{N} g_{nk} r_{k-n}$$

- For fixed equalizers, gnk = gn (i.e., its coefficients do not change with time k).
- For adaptive equalizers, gnk is updated periodically based on the current channel characteristics.
- The equalizer coefficients can be determined from the channel response, the training sequence, or the data sequence directly

Zero-Forcing:

In zero-forcing equalization, the equalizer {gk} attempts to completely inverse the channel by forcing

$$c_k * g_k = \delta(k - k_0)$$

Using matrix representation, this is equivalent to,

$$\begin{bmatrix} c_0 & 0 & \cdots & 0 \\ c_1 & c_0 & \ddots & \vdots \\ \vdots & c_1 & \ddots & 0 \\ c_L & \vdots & \ddots & c_0 \\ 0 & c_L & \vdots & c_1 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & c_L \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_N \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

Or,

$$Cg = 1$$

While it is not always possible to find the perfect solution to the above equation

$$\mathbf{g} = \mathbf{C}^{\dagger}\mathbf{1}$$

Adaptive equalization:

When the channel is time-varying (LTV), it is necessary to update the equalizer coefficients in order to track the channel changes.

Define the input signal to the equalizer as a vector rk where

$$\mathbf{r}_k = \begin{bmatrix} r_k & r_{k-1} & r_{k-2} & \cdots & r_{k-N} \end{bmatrix}^T$$

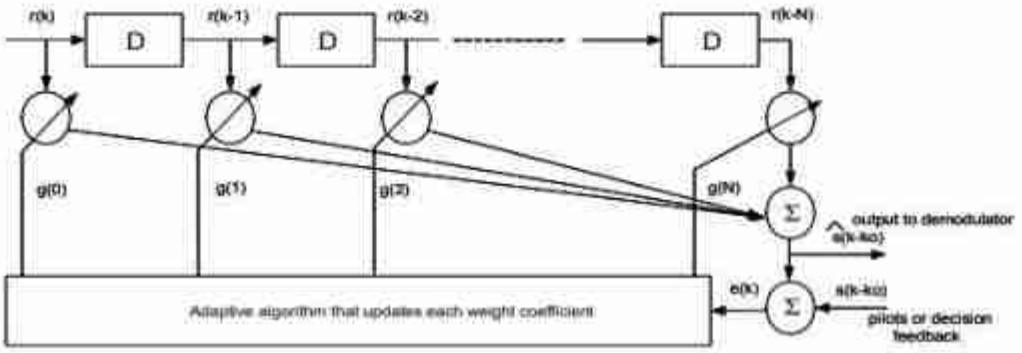
and an equalizer weight vector gk where,

$$\mathbf{g}_k = \begin{bmatrix} g_{0k} & g_{1k} & g_{2k} & \cdots & g_{Nk} \end{bmatrix}^T$$

The error signal ek is given by,

$$e_k = s_{k-k_o} - \widehat{s}_{k-k_o} = s_{k-k_o} - \mathbf{r}_k^T \mathbf{g}_k$$

An adaptive channel equalizer has the following structure.



In most cases, a training sequence (or pilot sequence) is inserted periodically during data transmission. This particular set of symbols {sk } are therefore known to the receiver and thus can be used to train the equalizer (i.e., update its coefficients).

Module - 5

Spread spectrum modulation

Nandini. S,
Assistant. Professor,
Dept. of ECE,
BVISIT.

Spread spectrum modulation is a signalling technique used for transmitting a signal to provide a secure communi-cation.

The main advantage of spread spectrum communication technique is to prevent interference. The signals modulated with these techniques are hard to interfere and never allowed to crack signals with no official access.

Defnition of spread spectrum modulation.

* "The modulated signal occupies a much more bandwidth than the minimum required bandwidth"
* The spectrum spreading at the transmitter and despreading at the receiver is obtained by special code [chipping code] which is independent of data [message signal]

Advantages of spread spectrum

1) Cross talk elimination
2) Better output with data integrity
3) Reduced effect of multipath fading
4) Better security
5) Reduction in noise
6) co-existence with other systems
7) Longest operative distances.
8) Hard to detect

9> Not easy to demodulate/decode

10> Difficult to jam the signals

Classification of spread spectrum systems

Spread spectrum modulation techniques are classified as

1> Direct sequence spread spectrum [DSSS]

2> Frequency hopped spread spectrum [FHSS]

Direct sequence spread spectrum [DSSS]

The direct sequence spread spectrum uses two stages of modulation. In the first stage, each and every bit of the incoming data is multiplied by a secret code signal [PN signal]. This transforms the narrowband incoming signal into a wideband signal i.e the spectrum of the signal is spreaded.

In the second stage the wideband signal is modulated using Phase shift keying [PSK]

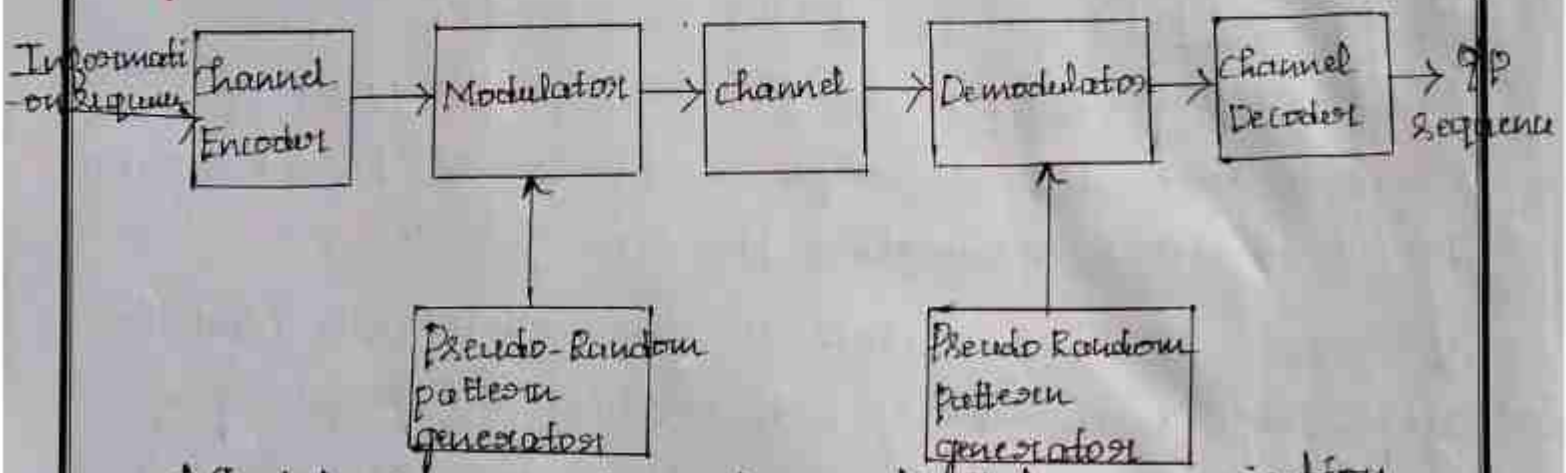
Frequency hopped spread spectrum.

In the frequency hopped spread spetrum the users are allowed to change the frequencies of usage, from one to another in a specified time interval. Hence the name frequency hopping.

The spectrum of the data modulated carrier is widened by changing the carrier frequency in pseudo random manner

For both direct sequence and frequency-hopped spread spectrum modulation techniques, a noise like spreading code called Pseudo-noise [PN] sequence is used.

Model of spread spectrum digital communication system.



Model of spread spectrum digital communication system.

* The channel encoder encodes the binary information sequence
* The channel encoded sequence is then given to the modulator. The modulator gets pseudonoise [PN] sequence from the pseudo-random pattern generator
* The PN sequence spreads the transmitted signal randomly over the wide frequency band. The signal at the output of the modulator is the spread spectrum modulated signal. This signal is transmitted over the channel.
* At the receiver, the demodulator recovers the encoded signal back from the spread spectrum signal

using the same PN sequence which was used at the transmitting end.

* Hence the pseudorandom pattern generators at the transmitter and receiver side operate in synchronisation with each other.

* The channel decoder decodes the binary information sequence back.

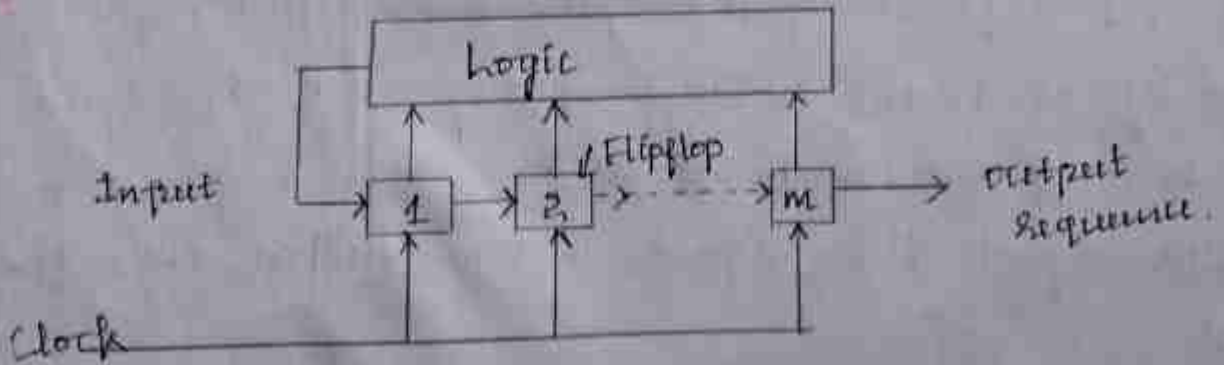
* If the PN sequence at the modulator uses phase shift keying [PSK] modulation to shift the phase of a signal, then this technique is known as Direct sequence spread spectrum modulation [DS-SS]

* When the PN sequence at the modulator is used in conjunction with M-ary FSK to shift the frequency of a signal pseudo randomly, then this technique is known as frequency hopped spread spectrum modulation [FH SS]

PN [pseudonoise sequence]

A pseudo-noise [PN] sequence is defined as a coded sequence of 1's and 0's with certain autocorrelation properties.

Generation of pseudo-noise [PN] sequence



Feedback shift register to generate PN sequence

* A pseudo noise [pseudo-random] sequence is a noise like high frequency signal
* The PN sequence is generated by a feedback shift register and the combinational logic
* The feedback shift register consists of m flip-flops. The data of one flipflop is shifted to the next flipflop whenever the clock pulse is applied. At the same time the output of flipflop is given to the logic circuit.
* Depending upon the outputs of flipflops, the output of combinational circuit decided. The combination logic circuit output is given as input for the first flipflop of the shift register.
* The PN sequence generated at the output of the last flipflop in the shift register
* The generated PN sequence is periodic and it repeats after 2^m digits. This is because the m-stage shift register have 2^m states, hence the output sequence repeats after 2^m bits.
* The combinational logic circuit used is a mod-2 adder [XOR].
* If the shift registers enters a zero state [Eg: 000], it is not possible to come out of this state [i.e $0 \oplus 0 = 0$] and output sequence generated will be only zeros.
* To prevent this problem, the zero state is not allowed. Therefore the total number of states in m-stage

feedback shift register is reduced from 2^m to 2^{m-1}. The PN sequence generated have a period of $2^m - 1$ bits.

Maximum length sequences:

The pseudonoise [PN] sequence generated by a linear feedback shift register has the length $2^m - 1$, then the sequence is known as maximum length [ML] sequence.

Properties of maximum length sequences [ML]

1) Balance property: In each period of a maximum length sequence, the number of 1's is always greater by one number than the number of 0's. This property is called balance property.

For an ML-sequence generated by m-state shift register with linear feedback

i) Period = $2^m - 1$ bits.
ii) Number of 1's = 2^{m-1}.
iii) Number of 0's = $2^{m-1} - 1$.

2) Run-property:

A Run is defined as a sequence of single type of binary [1's or 0's] digits

* The appearance of an alternate digit in a sequence starts a new run.
* The length of the run is the number of digits in the run.
* Among the runs of 1's and 0's in each period of ML sequence, it is observed that one-half (1/2) the runs of each type [0's or 1's) are length 1.

one-fourth [Y_{4i}] are of length 2, one-eighth are of length of 3 and so on.

* For a ML sequence generated by a m-stage linear feedback shift register,

The total number of runs $= \frac{L+1}{2}$ where $L = 2^m - 1$

3. Autocorrelation property:

Let $\{C_0, C_1, C_2, \ldots\ldots C_{L-1}\}$ be a ML-sequence of period $L = 2^m - 1$, generated by a m-stage linear feedback register. The normalised circular autocorrelation function [ACR] of the sequence is given by

$$R(\tau) = \frac{1}{L} \sum_{i=0}^{L-1} C_i \, C_{(i \oplus \tau)_{\text{mod-}L}} \qquad \tau = 0, 1, 2, \ldots L-1$$

where τ is the shift.

* If The symbols 1 and 0 are represented by +1 volt and −1 volt respectively.

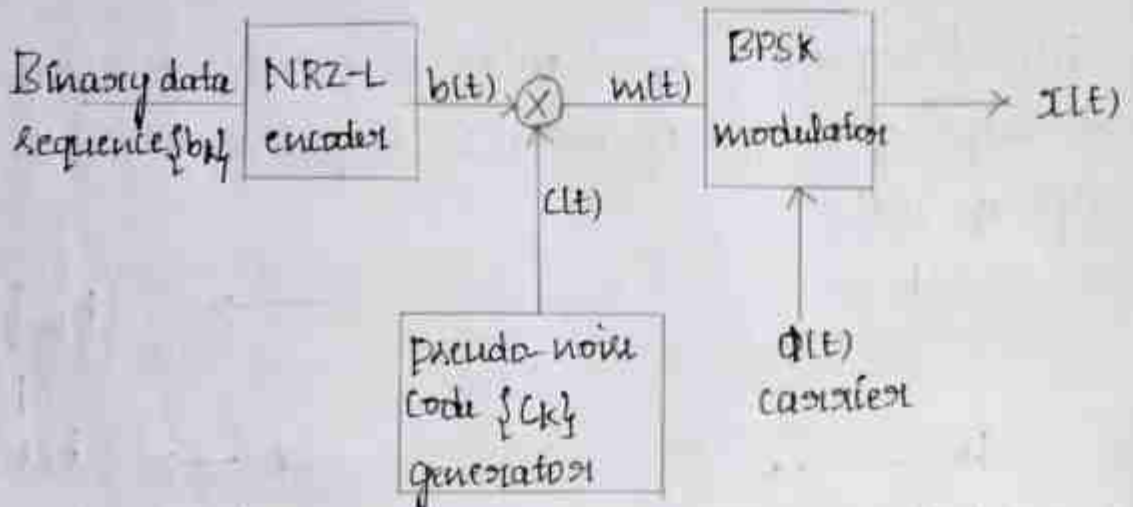
Autocorrelation function is defined as

$$R(\tau) = \begin{cases} 1 & \text{for } \tau = kL \quad k = 0, 1, 2 \ldots \\ -\frac{1}{L} & \text{for } \tau \neq kL \quad k = 0, 1, 2 \ldots \end{cases}$$

Direct sequence spread-spectrum system with BPSK [OR] DS-SS BPSK

Consider the transmission of binary information sequence modulated by binary phase shift keying [BPSK] The information rate is R bits/sec, channel bandwidth is B_c Hz, where $B_c >> R$. At the modulator, the bandwidth of the information signal is expanded to $W = B_c$ Hz by using pseudo-noise [PN] sequence.

Generation of DS-SS signal [OR] DS-SS BPSK transmitter.



DS-SS BPSK transmitter

* Let the data sequence be represented {bk}. This data sequence is converted to polar NRZ waveform b(t).

i.e when $b_k = 1$, $b(t) = +1$

when $b_k = 0$, $b(t) = -1$

* Let {ck} denote the PN sequence and c(t) denotes the polar NRZ signal generated by c_k.

i.e when $c_k = 1$, $c(t) = +1$

when $c_k = 0$, $c(t) = -1$

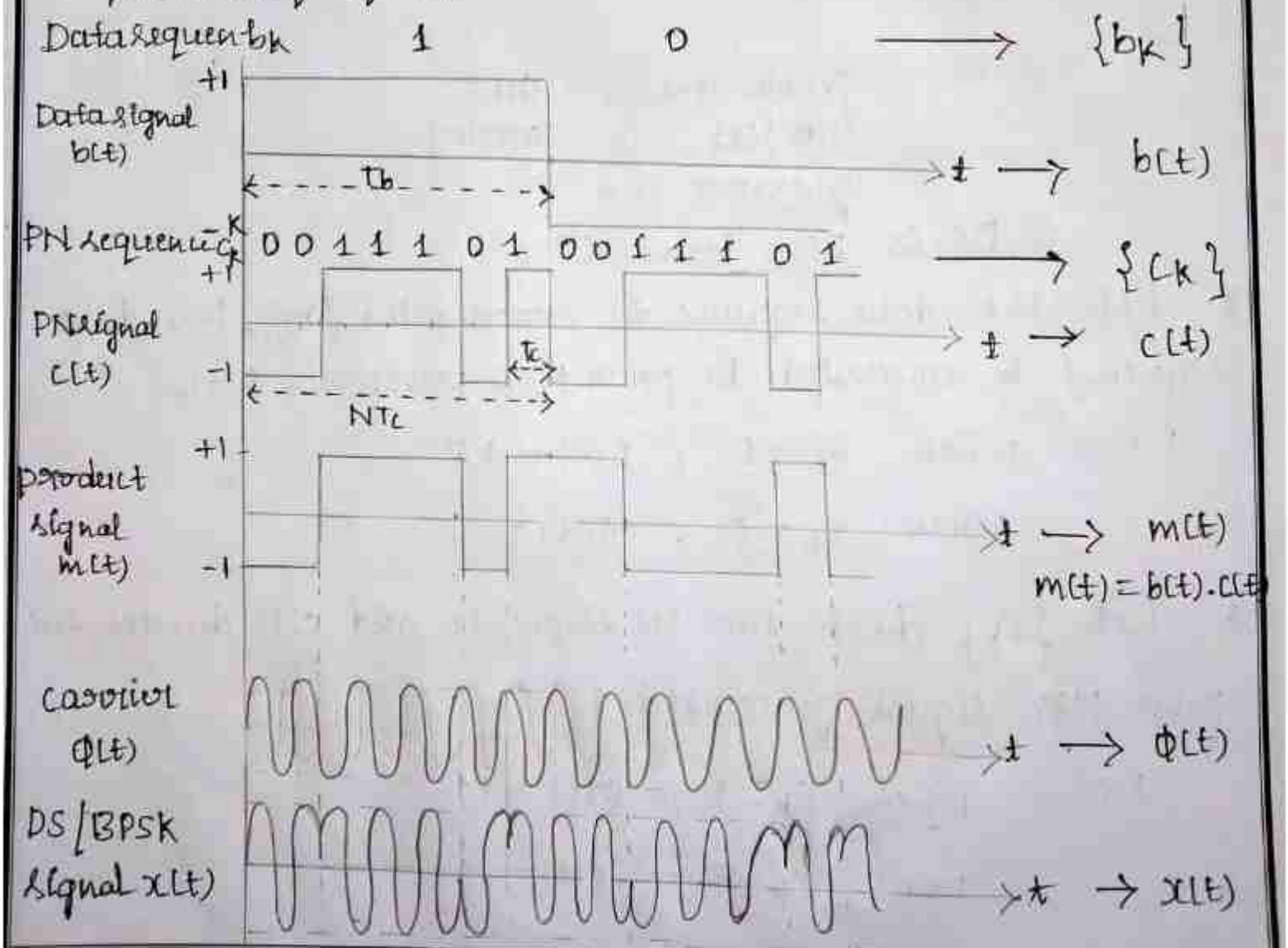
* The multiplier multiplies two signals b(t) & c(t) outputs direct sequence spread spectrum signal m(t).
* This signal is given as modulating signal to BPSK transmitter.
* The Direct sequence BPSK signal is generated at the output [x(t)].
* Let $\phi(t)$ be the carrier

$$\phi(t) = \sqrt{\frac{2}{T}} \sin 2\pi f_c t.$$

* Then transmitted signal x(t) is

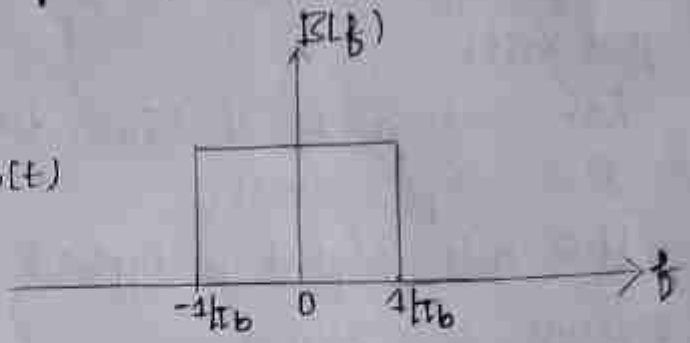
$$\boxed{x(t) = \sqrt{\frac{2}{T}} . m(t) . \sin 2\pi f_c t.}$$ where $m(t) = b(t) . c(t)$

* When Direct sequence signal m(t) is positive then there is a phase shift of 0° and if m(t) is negative, there is a phase shift of 180°

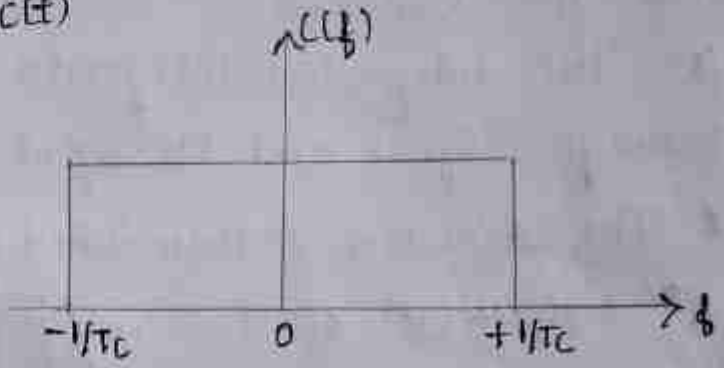


Spectrum spreading using simple rectangular spectra.

(a) Spectrum of data signal b(t)

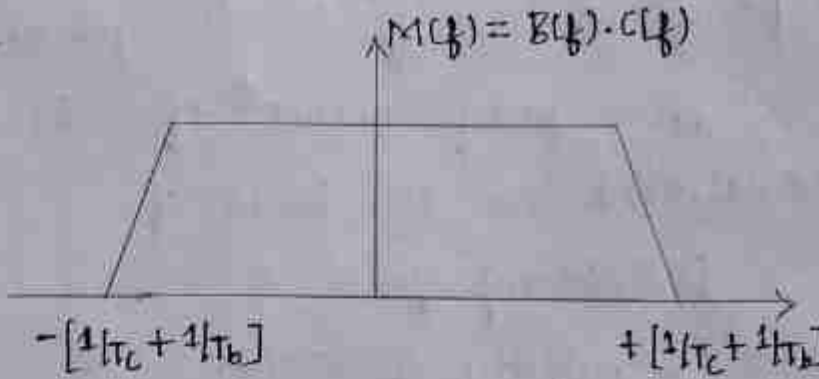


(b) Spectrum of carrier PN signal c(t)



(c) Convolution of spectra of b(t) & c(t)

$m(t) = b(t) * c(t)$



Convolution of the spectra of data signal with PN code signal.

Direct sequence spread spectrum BPSK receiver OR

DS-SS BPSK Receiver

* Fig shows the block diagram of DS-SS BPSK receiver
* there are two stages of demodulation
  - the received signal y(t) is multiplied with locally generated coherent carrier.

* The multiplied signal is passed through LPF. The Bandwidth of LPF is equal to the bandwidth of modulating signal m(t).
* This stage is a BPSK detector. The signal obtained at this stage is m(t).
(ii) The m(t) signal is applied to second demodulator which despreads the signal
* The local PN signal is exact replica of that used at the transmitter.
* The integrator integrates the product of detected message signal and PN signal over one bit period Tb.
* The decision is then taken depending on the polarity of output (V) of the integrator.

Performance of DS-SS system

The performance of DS-SS system can be evaluated on the basis of

Processing gain &

Probability of error.

▷ **Processing gain [PG]:** Processing gain is defined as the ratio of the bandwidth of spreaded message signal to the bandwidth of the unspreaded message signal.

$$PG = \frac{\text{BW of spread signal}}{\text{BW of unspreaded signal.}}$$

Bandwidth of unspreaded msg signal, $R_b = \frac{1}{T_b}$.

Bandwidth of spreaded msg signal, $W_c = \frac{1}{T_c}$

Then $PG = \frac{1/T_c}{1/T_b}$ $T_b \rightarrow$ bit period

$$\boxed{PG = \frac{T_b}{T_c}} \longrightarrow (1)$$ $T_c \rightarrow$ chip period

We know that one bit period T_b of message signal is equal to 'N' bits periods of PN code signal.

i.e $$\boxed{T_b = NT_c} \longrightarrow (2)$$

By substituting eq^n (2) in eq^n (1)

$$PG = \frac{NT_c}{T_c}$$

$$\boxed{PG = N}$$

2) Probability of error

The probability of error for DS-BPSK system is

$$P_e = \frac{1}{2} erfc\left(\sqrt{E_b/JT_c}\right) \longrightarrow (1)$$

where $J \rightarrow$ average interference noise power.

W.K.T for coherent PSK.

$$P_e = \frac{1}{2} erfc\left(\sqrt{E_b/N_0}\right) \longrightarrow (2)$$

In DS - BPSK, the Interference may be treated as wideband noise power density $N_0/2$ defined as.

From eq^n (1) and (2).

$$\frac{N_0}{2} = \frac{JT_c}{2}$$

$$N_0 = JT_c$$

* The bit energy $E_b = PT_b$

Where $P \rightarrow$ average signal power.

* The bit energy to noise density ratio is defined as

$$\frac{E_b}{N_0} = \frac{PT_b}{JT_c}$$

$$\frac{E_b}{N_0} = \frac{T_b}{T_c} \cdot \frac{P}{J}$$

$$\frac{E_b}{N_0} = PG_1 \cdot \frac{P}{J}$$

$$\boxed{\frac{J}{P} = \frac{PG_1}{E_b/N_0}}$$

The ratio of J/P is termed as Jamming margin

$$(\text{Jamming margin})_{dB} = 10 \log_{10}\left(\frac{PG_1}{E_b/N_0}\right)$$

$$= 10\log_{10}(PG_1) - 10\log_{10}(E_b/N_0)_{min}$$

$$\left[\begin{matrix}\text{Jamming} \\ \text{margin}\end{matrix}\right]_{dB} = (PG_1)_{dB} - 10\log_{10}(E_b/N_0)_{min}$$

Effect of despreading on a Narrowband interference

If the data sequence $b(t)$ is narrowband & PN sequence $c(t)$ is wideband.

Transmitted signal / spreaded signal $m(t) = c(t) \cdot b(t)$

Received signal $y(t) = m(t) + i(t)$

$$y(t) = c(t) \cdot b(t) + i(t)$$

where $i(t) \rightarrow$ interference signal.

* The received signal is demodulated by using a locally generated PN sequence ($c(t)$).

Demodulated signal is

$$z(t) = c(t) \cdot y(t) = c(t)\,[\,c(t) \cdot b(t) + i(t)\,]$$

$$z(t) = b(t)\,c^2(t) + i(t) \cdot c(t)$$

* As $c(t)$ alternates between -1 & $+1$, when it is squared $c^2(t) = 1$ for all t.

$$z(t) = \underbrace{b(t)}_{\text{Narrow band signal}} + \underbrace{c(t)\, i(t)}_{\rightarrow \text{spreading code affects only it}}$$

* the term $i(t) \cdot c(t)$ results in wideband interference with power spectral density $I_0 = P_I / W$
* the desired signal $b(t)$ is demodulated by passing $z(t)$ through matched filter that has a bandwidth R.
* the total power in the interference at the output of demodulator is

$$I_0 R = \frac{P_I}{W} \cdot R = \frac{P_I}{W/R} = \frac{P_I}{T_b/T_c} = \frac{P_I}{L_c}$$

* the power in the interfering signal is reduced by an amount equal to the bandwidth expansion factor W/R.

* the Reduction in the interference power is the basic reason for using spread-spectrum signals to transmit digital information over interfering channels.

Frequency hop spread spectrum [FH-SS]

The spread spectrum in which the carrier hops randomly from one frequency to another is called frequency hop spread spectrum technique. The modulation used is MFSK [M-ary FSK]

The FH-SS are of two types.

1) Slow FH-SS
2) Fast FH-SS.

In Slow FH-SS, the symbol rate R_s of the MFSK signal is an integer multiple of the hop rate R_h

i.e Several symbols are transmitted in each frequency hop.

hop rate $R_h < R_s$

In Fast FH-SS, the hop rate R_h is an integer multiple of the MFSK symbol rate R_s. i.e carrier frequency will change at hop several times during the transmission of one symbol. $R_h > R_s$

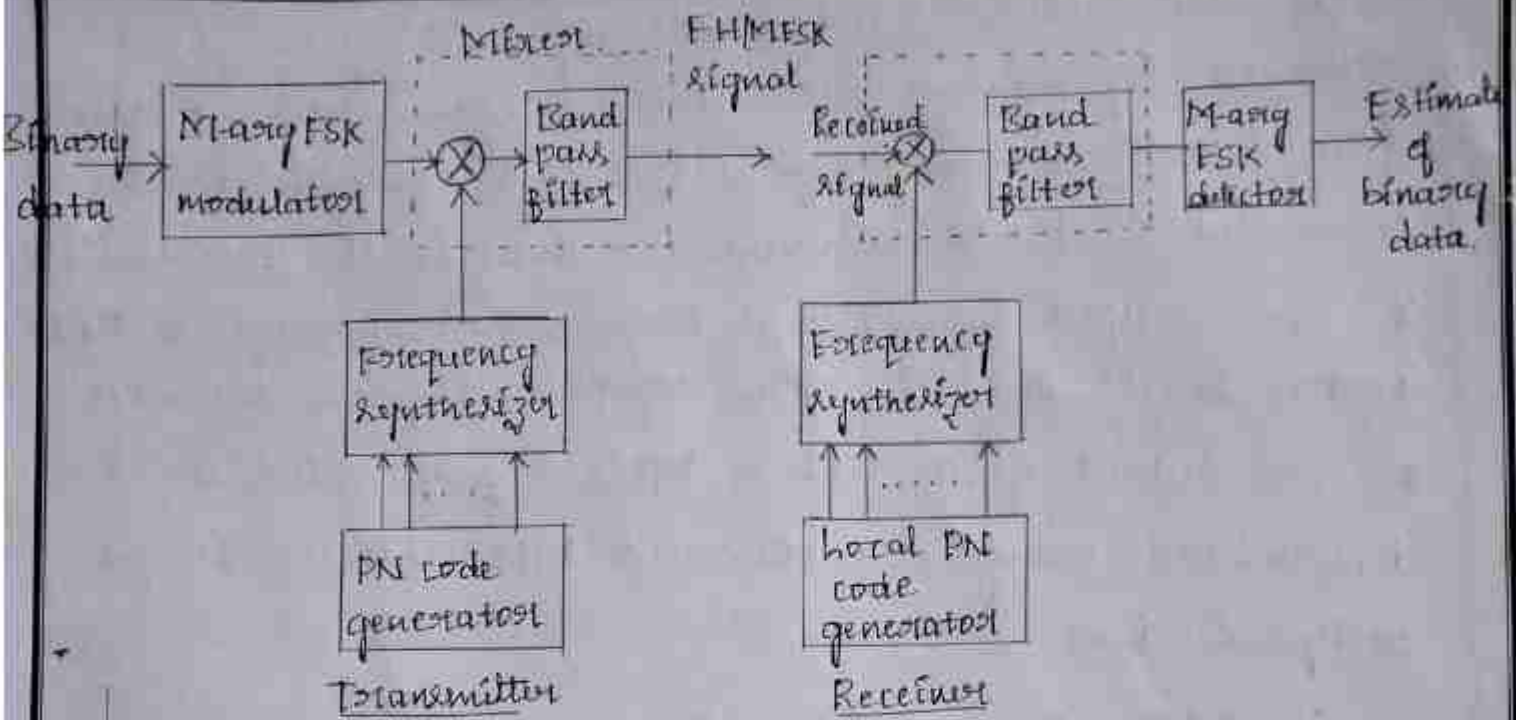


Fig shows the block diagram of a FH/MFSK transmitter and receiver

* the FH/MFSK transmitter involves frequency shift keying followed by mixer.
* the incoming data is applied to M-ary FSK system. The resulting modulated wave & output from a digital frequency synthesizer are then applied to a mixer that consists of a multiplier followed by a filter.
* the BPF is designed to select sum frequency component resulting from multiplication process.
* PN source will drive the frequency synthesizer, which enables carrier frequency over 2^k distinct values.
* on single hop, bandwidth of the transmitted signal is same as that resulting from the use of conventional MFSK format.
* For complete range of 2^k frequency hops, FH/MFSK signal occupies a much larger bandwidth.
* The Non coherent MFSK detector is used.

* At the receiver, the frequency hopping is first removed by mixing the received signal with output of a local frequency synthesizer that is synchronously controlled in the same manner as that in the transmitter.
* the output of mixer is then passed through a BPF which selects the difference component from the mixer.
* The output of a BPF is MFSK signal which is demodulated using non-coherent MFSK detector to get original data.
* In FH SS system, an FH tone of shortest duration is called 'chip'

the chip rate R_c,

$$\underline{R_c = \max(R_h, R_s)}$$

$R_h \rightarrow$ hop rate

$R_s \rightarrow$ Symbol rate

* For slow FH-SS system

$$R_c = R_s$$

$$\therefore R_c = R_s = \frac{R_b}{K} \geq R_h \qquad \text{where } K = \log_2 M$$

* For Fast FH-SS system

$$R_c = R_h$$

* Processing gain

$$PG = \frac{\text{BW of spreaded signal}}{\text{BW of unspreaded signal}} = \frac{\text{BW of FH-signal}}{\text{BW of Baseband signal}}$$

Let f_s be the symbol frequency. Out these are 2^k frequency hops generated from K-bits of PN sequence

∴ BW of the FH-signal is $2^k f_s f$

BW of the unspreaded signal is f_s

$$P_{PG} = \frac{2^k f_s}{f_s}$$

$$P_{PG} = 2^k$$

Applications of Direct sequence spread spectrum [DS-SS] systems.

1) Low detectability signal transmission.
OR
Low probability Intercept.

For this application, the signal spectral density is intensionally kept small with respect to channel noise and receiver noise so that the presence of the signal is not detected easily.

Consider the average received signal power at the intended receiver is P_{AV} and the noise power is N_{AV}. Then the signal is transmitted at low power levels. Such that $\frac{P_{AV}}{N_{AV}} < 1$. Hence the receivers which are in the vicinity of the intended receiver cannot detect the presence of the signal.

The intended receiver recovers the message signal with the help of processing gain and coding gain. Other receivers who doesn't know the PRI sequence hence they can not receive the information signal with the help of processing gain and coding gain This is called as the signal has low probability of being intercepted - [LPI]

2) Code-division multiple Access with DS-SS. [CDMA]

Many users transmit their signals on the same channel bandwidth. Each transmitter and receiver pair has a distinct PN sequence. Thus signals of particular transmitter are received by its intended receiver only. Even if many users are transmitting at the same time the signals from other users appear as additive interference which are rejected by the spread decoder. The level of interference depends on the number of users transmitting at a time.

The main advantage of CDMA is that the number of users sharing a same channel can be increased or decreased very easily.

Large number of users can transmit on the same channel if their messages are for short periods of time.

3) Wireless LAN

Spread spectrum signals have been used in the IEEE wireless LAN standards 802.11 and 802.11b which operate in the 2.4GHz frequency band.

The available channel bandwidth is subdivided into 14 overlapping 22MHz channels, although not all channels are used in all countries.

In the 802.11 standard, an 11-chip Barker sequence is modulated and transmitted at a rate of 11MHz. The 11 chip Barker sequence is {1, -1, 1, 1, -1, 1, 1, 1, -1, -1, -1}

the Barker sequence is modulated either with BPSK or QPSK. When BPSK is used with 11 chips per bit a data rate of 1 Mbps is achieved. When QPSK is used with 11-chips per symbol [2-bits] - a data rate of 2Mbps. is achieved.

In the 802.11b standard 11 MHz chip rate is maintained but the Barker sequence is replaced by a set of 8-chip waveform sequences called complementary code shift keying [CCK]. The use of CCK modulation results in a data rate of 1 Mbps.

CDMA System based on IS-95

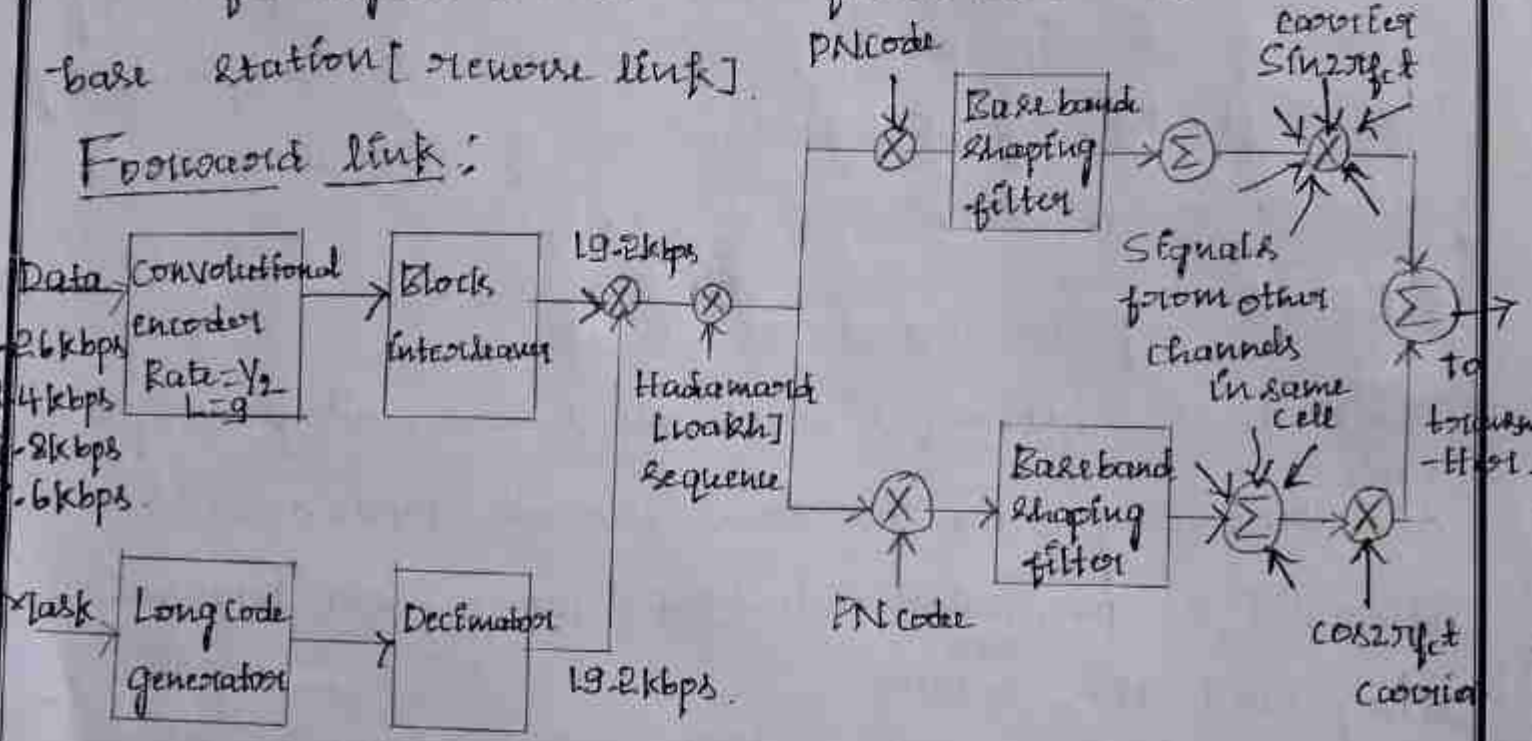
In this system, several users transmit messages simultaneously over the same channel. Each user is allocated a particular code [PN sequence] for transmitting over the common channel.

* CDMA [1st generation digital cellular communication system) was developed by Qualcomm, and it has been standardized and designated as IS-95 [Interim-standard] by Telecommunication Industry Association [TIA] for use in the 800MHz and 1800 MHz frequency bands.
* CDMA system based IS-95 is widely used in North America and some eastern countries.
* A major advantage of CDMA is that a large number of users can be accomodated., if each user transmits messages for a short period of time.

* Advantage of CDMA over other multiple access methods [TDMA & FDMA] is that the entire frequency band is available at each base station.

* The nominal bandwidth used for transmission from a base station to the mobile receivers [forward link] is 1.25MHz. A separate channel (1.25MHz bandwidth) is used for signal transmission from mobile receiver to base station [reverse link].

Forward link:



* The data from the speech coder is encoded by a rate 1/2, constraint length L=9, convolutional code. the speech coder generates data at the variable rates of 9.6kbps, 4.8 kbps, 2.4 kbps & 1.2 kbps.

* The encoded bit for each frame are passed through a block interleaver, which overcomes the effect of burst errors that may occur in transmission through the channel.

* The data bits from the block interleaver are scrambled by multiplication with the output of a long code generator running at chip rate of 1.2288×10^6 chips/sec, but whose output is decimated by a factor of 64 to 19.2 kchip/sec.

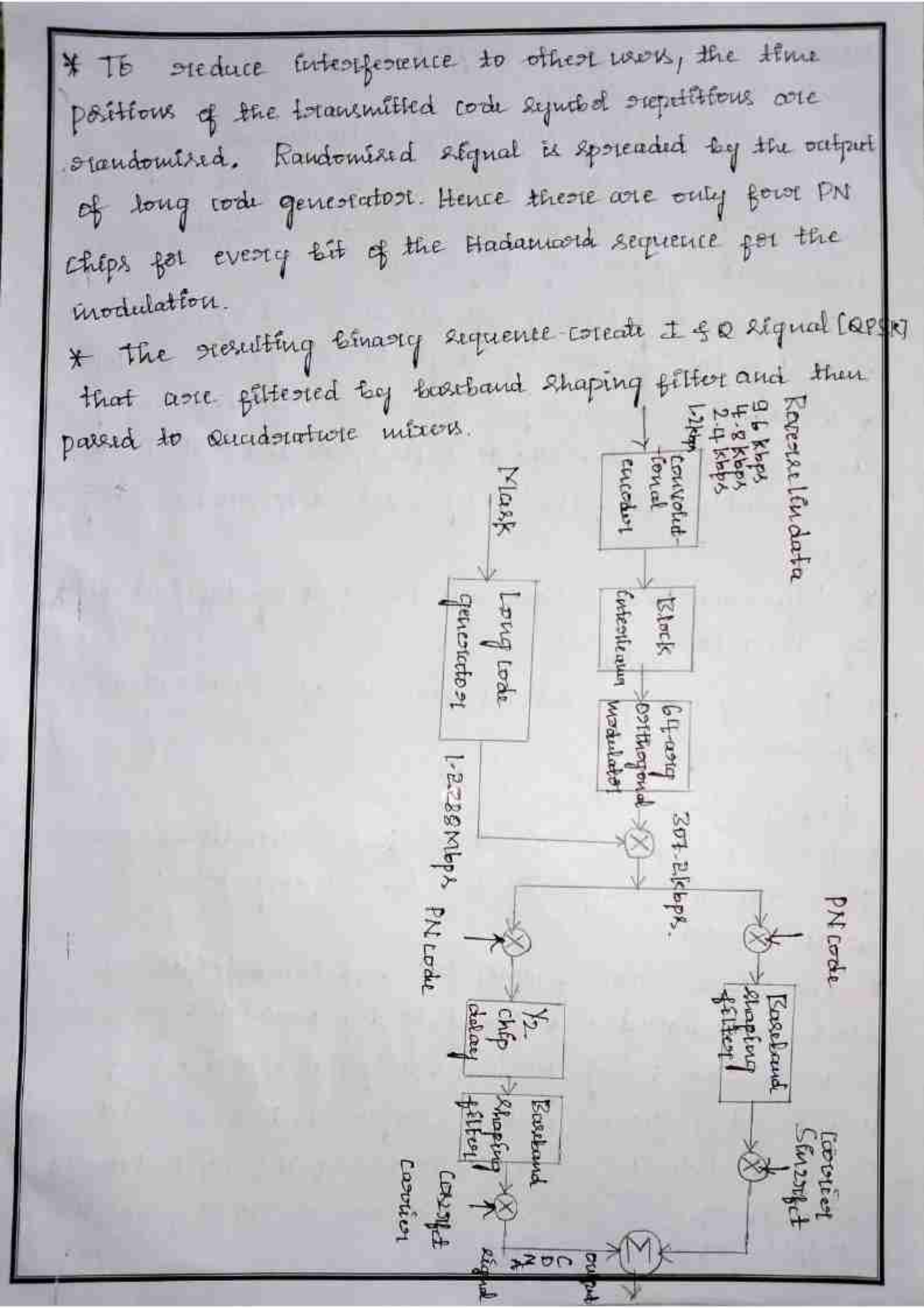
* Each channel user is assigned Hadamard [walsh] sequence of length 64.
* Each user, using the assigned Hadamard sequence, multiplies the data sequence by the assigned Hadamard sequence. Thus each encoded data bit is multiplies the Hadamard sequence of length 64
* The resulting binary sequence is spread by multiplication with two PN sequences of length $N = 2^{15}$, this creates inphase & Quadrature phase signal components. Thus the binary data signal is converted to a four-phase and both I & Q components are filtered by baseband spectral shaping filter.
* Different base stations are identified by different offset of these PN sequences.
* The signals for all 64 channels are transmitted synchronously.

Reverse link

* Reverse link uses a rate 1/3, K=9 convolution coder. This code has essentially the same coding gain in an AWGN channel.
* For each 20ms frame, the 576 encoded bits are block interleaved and passed to the modulator. The data is modulated using M=64 orthogonal signal set of Hadamard sequences each of length 64. Thus a 6-bit block of data is mapped into one of the 64-Hadamard sequences. The result is a chip rate of 307.2 kbps at the output of the modulator.

* To reduce interference to other users, the time positions of the transmitted code symbol repetitions are randomised. Randomised signal is spreaded by the output of long code generator. Hence there are only four PN chips for every bit of the Hadamard sequence for the modulation.

* The resulting binary sequence create I & Q signal [QPSK] that are filtered by baseband shaping filter and then passed to quadrature mixers.



Module-1:

Anusha.M.N
Asst. professor,
Dept. of ECE,
BGSIT

Discrete Fourier Transform(DFT)

Introduction:

In the digital signal processing, all the signals are discrete. When the analog signals such as speech, music etc are to be processed by the DSP system, then such signals are converted to discrete form by A/D Converters. After processing, the signals are converted to analog form by D/A Converters.

Many applications demand the processing of signals in frequency-domain: frequency content, periodicity, energy and power spectrum etc... can be better analyzed in frequency-domain. Hence, the signals are transformed from time-domain to frequency domain. Such transformations can be obtained with the help of Fourier transform (FT) and Discrete-Fourier transform (DFT).

For the discrete signals, DFT is used. DFT can also be used for linear filtering and Correlation.

Once, the required analysis and processing is performed in frequency-domain, the signals are transformed back in time-domain by inverse Discrete Fourier transform (IDFT).

Frequency-domain Sampling and Reconstruction of discrete-time signals :—

a) Need of frequency-domain Sampling :—

W.K.T the Fourier transform of the signal can be calculated as,

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\Omega n} \longrightarrow (1)$$

Here $x(n)$ is a discrete-time signal and

Ω is the frequency.

Note that 'Ω' is the continuous function from 0 to 2π.

Scanned with CamScanner

This means Eventhough x(n) is discrete, its spectrum X(Ω) is continuous. Such continuous function cannot be evaluated on digital pocessor. on digital processor, only discrete signals can be processed.

b) Frequency - domain sampling :-

To overcome the problem of digital processing, the spectrum X(Ω) is sampled uniformly.

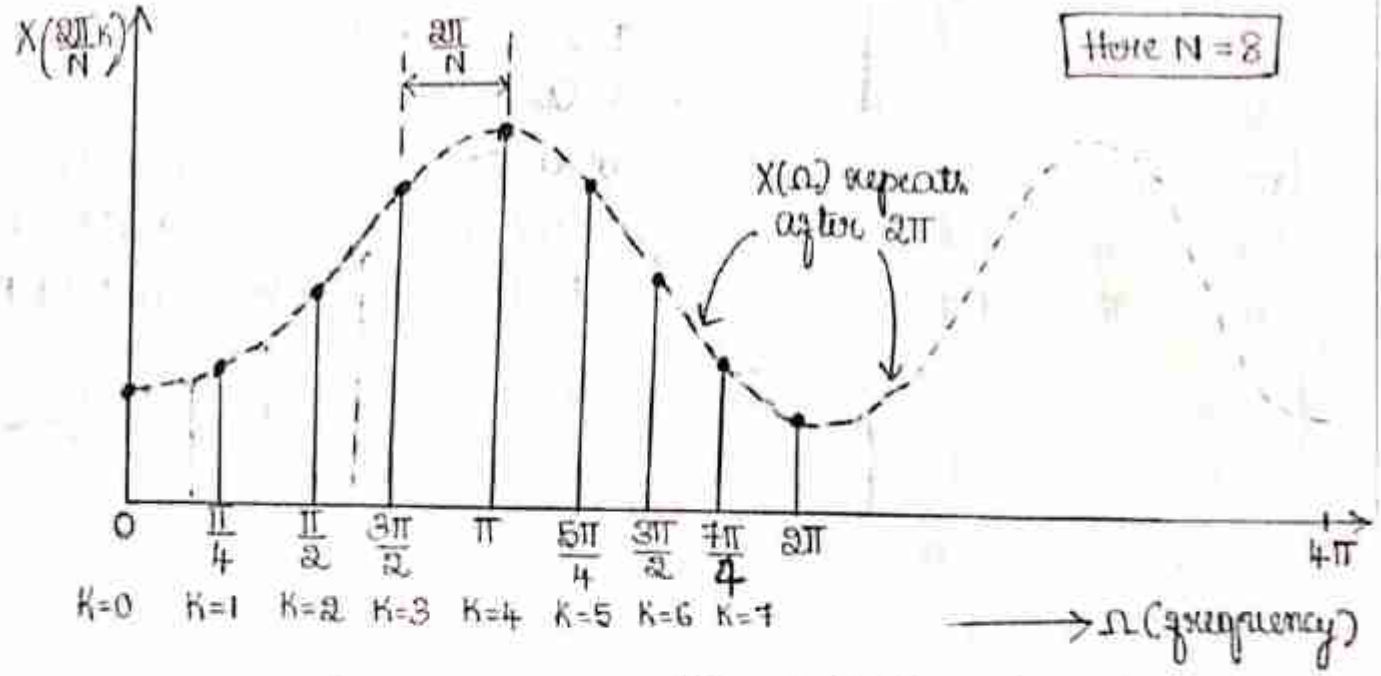
N.K.T X(Ω) is periodic with the period of 2π. Hence, the samples taken from 0 to 2π are only important for our processing.

Let 'N' samples are taken from 0 to 2π. Hence, the spacing between successive samples will be 2π/N.

Therefore in equation (1), let us put $\Omega = \frac{2\pi}{N} K$

i.e., $$X\left(\frac{2\pi}{N}K\right) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\frac{2\pi}{N}Kn} \longrightarrow (2)$$

Here K = 0, 1, 2, ... (N-1)



Thus, 'K' is an index for the samples and X(Ω) is calculated only at discrete values.

Ex:- Consider the spectrum shown in above fig, observe that N=8 samples are taken over the period of 2π. Hence, X(Ω) will be calculated at $\Omega = 0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi, \frac{5\pi}{4}, \frac{3\pi}{2}$ and $\frac{7\pi}{4}$.

Scanned with CamScanner

These samples are addressed by $K = 0, 1, 2, \ldots N$. By putting the value of 'K' in Eq(2), the sampled spectrum can be obtained.

c) To determine minimum value of 'N' :-

In Eq(2), note that 'n' varies from $-\infty$ to ∞.

Let us divide this summation into individual summation containing only 'N' samples of $x(n)$

i.e., $X\left(\frac{2\pi}{N}K\right) = \ldots\ldots + \sum + \sum + \sum + \ldots\ldots$

Each summation is carried out over 'N' samples of $x(n)$.

Therefore, above equation can be written as,

$$X\left(\frac{2\pi}{N}K\right) = \ldots\ldots + \sum_{n=-N}^{-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} + \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} + \sum_{n=N}^{2N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} + \ldots\ldots$$

The above individual summations can be represented as,

$$X\left(\frac{2\pi}{N}K\right) = \sum_{L=-\infty}^{\infty} \sum_{n=LN}^{LN+N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn}$$

Now, let us change the index of inner sum from n to $n-LN$. Hence, its limits will be $n=0$ to $n=N-1$.

$$\text{i.e., } X\left(\frac{2\pi}{N}K\right) = \sum_{L=-\infty}^{\infty} \sum_{n=0}^{N-1} x(n-LN)\, e^{-j\frac{2\pi}{N}K(n-LN)}$$

$$= \sum_{L=-\infty}^{\infty} \sum_{n=0}^{N-1} x(n-LN)\, e^{-j\frac{2\pi}{N}Kn}\, e^{j\frac{2\pi}{N}K\cdot LN}$$

Here, $e^{j2\pi KL} = 1$ always, since 'K' and 'L' are both integers. Hence, the term $2\pi KL$ will be always integer multiple of 2π.

$$\therefore X\left(\frac{2\pi}{N}K\right) = \sum_{L=-\infty}^{\infty} \sum_{n=0}^{N-1} x(n-LN)\, e^{-j\frac{2\pi}{N}Kn}$$

Let us interchange the order of summation,

$$X\left(\frac{2\pi}{N}K\right) = \sum_{n=0}^{N-1} \left[\sum_{L=-\infty}^{\infty} x(n-LN)\right] e^{-j\frac{2\pi}{N}Kn} \longrightarrow (3)$$

$$X\left(\frac{2\pi}{N}K\right) = \sum_{n=0}^{N-1} x_p(n)\, e^{-j\frac{2\pi}{N}Kn} \longrightarrow (4)$$

Here $K = 0, 1, 2, \ldots (N-1)$

Scanned with CamScanner

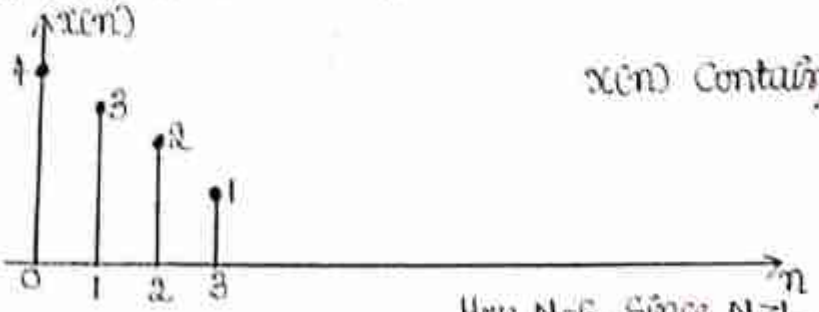
Here $x_p(n) = \sum_{l=-\infty}^{\infty} x(n-lN)$

$= \ldots + x(n+2N) + x(n+N) + x(n) + x(n-N) + x(n-2N) + \ldots$

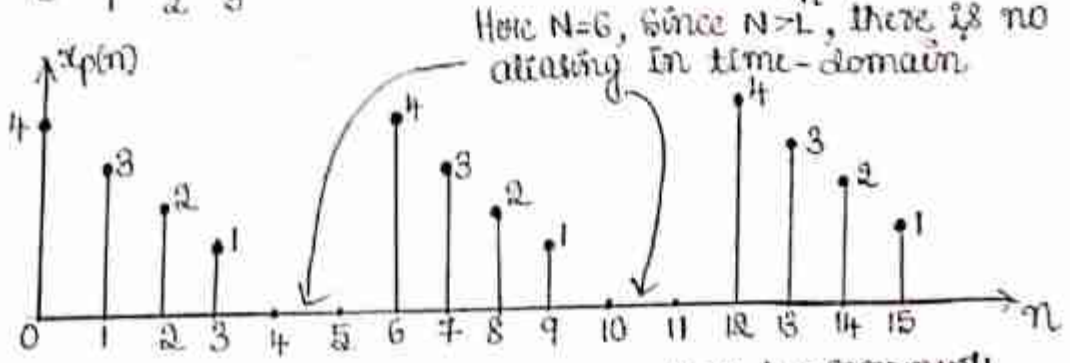
This means $x_p(n)$ is the periodic repetition of $x(n)$ with period of N-samples.

Consider some arbitrary non-periodic signal $x(n)$. It contains 'L' samples. It is shown in fig(a). In this waveforms, L=4 is taken. Note that $x(n)$ is non-periodic.

(a)



(b)



(c)

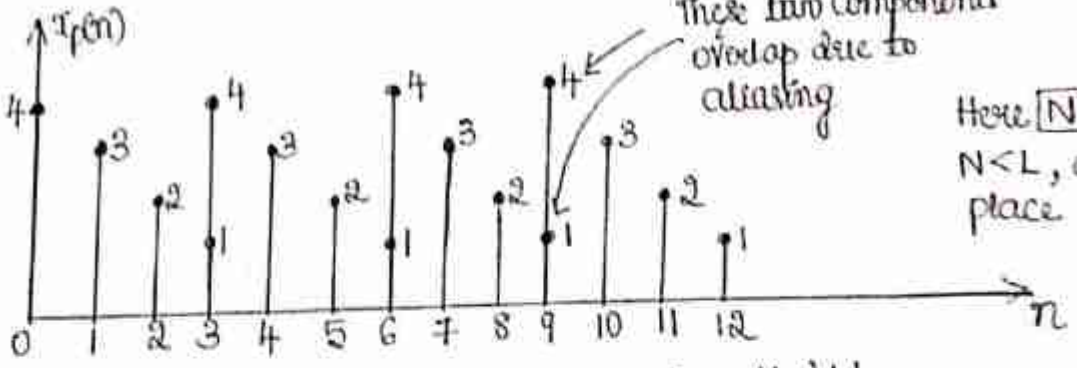


Fig (a) : An original signal $x(n)$ of length 'L'.

(b) : periodic repetition of $x(n)$ with period $N > L$.

(c) : periodic repetition of $x(n)$ with period $N < L$.

Case (i): When $N > L$, no aliasing :

Now let us prepare the signal $x_p(n)$. It is obtained by periodic repetition of $x(n)$. Let the period 'N' be greater than 'L'. For Example, N=6 in fig(b). Observe that sample values at n=4

Scanned with CamScanner

and $n=5$ are zero. This is because the signal repeats at $n=6, 12, \ldots$ etc. Here $N>L$, hence there is no aliasing.

Case(ii):- When $N<L$, Aliasing :

Now Consider the waveform of $x_p(n)$ of fig(c). In this waveform, observe that $N=3$. Since $N<L$, the two samples are overlapping at $n=3,6, \ldots$ etc. This is Called aliasing. Hence, it is possible to obtain $x(n)$ from $x_p(n)$.

Thus, to avoid aliasing in time-domain, the number of samples in the frequency spectrum must be greater than number of samples in time-domain sequence.

i.e., $\boxed{N \geq L}$ $\longrightarrow$ (5)

The above equation shows that we should take atleast 'L' samples over 0 to 2π.

Therefore, the DFT of aperiodic (or) finite-length sequence is given by,

$$\left\langle X(K) = \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} \; ; \; 0 \leq K \leq N-1 \right\rangle$$

(or)

$$\left\langle X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn} \; ; \; 0 \leq K \leq N-1 \right.$$
$$\left. \text{where } W_N^{nK} = e^{-j\frac{2\pi}{N}Kn} \right\rangle$$

Reconstruction of discrete-time signals :—

W.K.T $x_p(n)$ is periodic with period 'N'. It can be expressed by discrete Fourier series as,

$$x_p(n) = \sum_{K=0}^{N-1} C(K)\, e^{j\frac{2\pi}{N}Kn} \; ; \; 0 \leq n \leq N-1 \longrightarrow (1)$$

and the Fourier-series Co-efficients $C(K)$ are given by,

$$C(K) = \frac{1}{N} \sum_{n=0}^{N-1} x_p(n)\, e^{-j\frac{2\pi}{N}Kn} \; ; \; 0 \leq K \leq N-1 \longrightarrow (2)$$

W.K.T $\sum_{n=0}^{N-1} x_p(n)\, e^{-j\frac{2\pi}{N}Kn} = X\left(\frac{2\pi}{N}K\right)$

Scanned with CamScanner

∴ Eq(2) becomes,

$$C(K) = \frac{1}{N} X\left(\frac{2\pi}{N}K\right) \; ; \; 0 \le K \le N-1 \longrightarrow (3)$$

putting this value of C(K) in Eq(1),

$$x_p(n) = \sum_{K=0}^{N-1} \frac{1}{N} X\left(\frac{2\pi}{N}K\right) e^{\frac{j2\pi}{N}Kn} \; ; \; 0 \le n \le N-1 \longrightarrow (4)$$

This equation gives the expression for time-domain sequence $x_p(n)$ from frequency-domain samples $X(\frac{2\pi}{N}K)$. Thus, time-domain sequence can be reconstructed from frequency-domain sampling.

$W_N = e^{\frac{-j2\pi}{N}}$, which is called the "Twiddle factor".

The N-point IDFT is given by,

$$\left\langle x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) W_N^{-nK} \; ; \; 0 \le n \le N-1 \right\rangle$$

DFT as a linear Transformation (Matrix method) :-

W.K.T,

$$\text{DFT} : X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \; ; \; 0 \le K \le N-1$$

$$\text{IDFT} : x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) W_N^{-Kn} \; ; \; 0 \le n \le N-1$$

Twiddle factor : $W_N = e^{-j2\pi/N}$

Computation of each point of DFT requires: N-Complex multiplications and (N-1) Complex additions. Hence, N-point involves: N^2-Complex multiplications and N(N-1) Complex additions.

DFT and IDFT can be viewed as linear transformation on sequences x(n) and X(K) respectively.

Let x_N be a N-point vector of the discrete-time sequence $x(n)$; $0 \le n \le N-1$, X_N be N-point vector of frequency-domain signal X(K) and W_N be (N X N) matrix of Complex Exponentials then,

Scanned with CamScanner

$$x_N = \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix} \quad ; \quad X_N = \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} \quad \text{and} \quad W_N = \begin{bmatrix} W_N^{Kn} & \xrightarrow{n=0,1,\ldots(N-1)} \\ \downarrow & \\ K=0,1,\ldots(N-1) & \end{bmatrix}_{(N\times N)}$$

$$W_N = \begin{bmatrix} W_N^{00} & W_N^{01} & W_N^{02} & \cdots & W_N^{0(N-1)} \\ W_N^{10} & W_N^{11} & W_N^{12} & \cdots & W_N^{1(N-1)} \\ W_N^{20} & W_N^{21} & W_N^{22} & \cdots & W_N^{2(N-1)} \\ \vdots & & & & \vdots \\ W_N^{(N-1)0} & W_N^{(N-1)1} & W_N^{(N-1)2} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix}_{N\times N}$$

$$W_N = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & W_N^2 & \cdots & W_N^{1(N-1)} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & & \\ 1 & W_N^{1(N-1)} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix}_{N\times N}$$

Now, with all the above definitions, the N-point DFT may be expressed in matrix as:

$$\boxed{X_N = W_N x_N}$$

where, W_N is matrix of linear transformation.

Ex: W_4 matrix can be expressed as:

Here $N=4$,

$$\therefore W_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^4 & W_4^6 \\ 1 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix}_{4\times 4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^0 & W_4^2 \\ 1 & W_4^3 & W_4^2 & W_4^1 \end{bmatrix}_{4\times 4}$$

From this, we can conclude that 'W_N' is "Symmetric matrix".

Scanned with CamScanner

W.K.T $X_N = W_N \cdot x_N$

Multiplying both sides by W_N^{-1},

$$W_N^{-1} \cdot X_N = W_N \cdot x_N \cdot W_N^{-1}$$

$$W_N^{-1} \cdot X_N = x_N$$

$\therefore$ IDFT, $x_N = W_N^{-1} \cdot X_N$

But from the IDFT equation, it becomes

$$\text{IDFT}, \; x_N = \frac{1}{N} W_N^{*} \cdot X_N$$

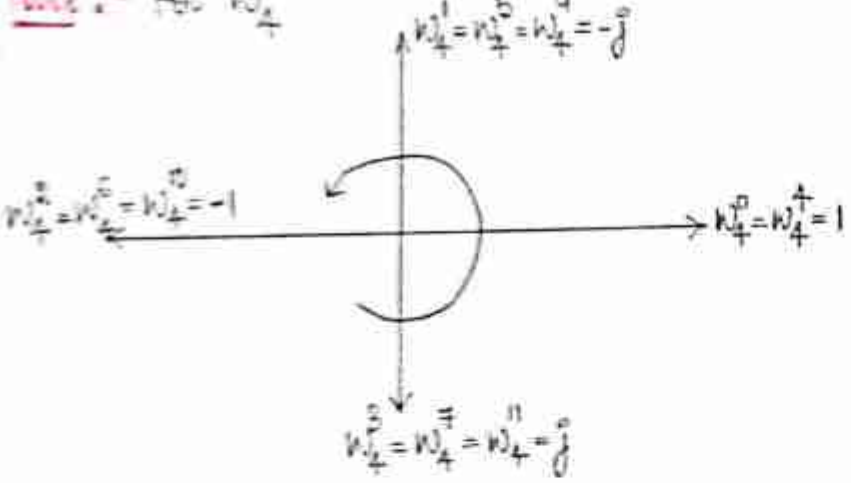
where, W_N^{*} is complex conjugate of matrix W_N.

$\therefore W_N^{-1} = \frac{1}{N} \cdot W_N^{*}$

This implies that $\boxed{W_N \cdot W_N^{*} = N I_N}$

where, $I_N \rightarrow (N \times N)$ Identity matrix.

Note :- For W_4



W_4^0	W_4^1	W_4^2	W_4^3
1	$-j$	-1	$+j$

The Twiddle factor 'W_N' satisfies the following properties :

1) Cyclic / periodicity property :-

W.K.T $W_N = e^{-j\frac{2\pi}{N}}$

prove that $W_N^{N+Kn} = W_N^{Kn}$:

Let $N=4$ and $Kn = 1$

Now, $W_4^{4+1} = W_4^1$

Scanned with CamScanner

$L.H.S = W_4^4 \cdot W_4^1 = e^{\frac{-j2\pi}{4} \cdot 4} \cdot e^{\frac{-j2\pi}{4} \cdot 1} = e^{-j2\pi} \cdot e^{-j\pi/2} = 1 \cdot e^{j\pi/2} = -j$

$R.H.S = W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = e^{\frac{-j\pi}{2}} = -j$

$\therefore \boxed{W_4^5 = W_4^1}$

It satisfies the cyclic property.

2) Symmetry property :-

W.K.T $W_N = e^{\frac{-j2\pi}{N}}$

prove that $W_N^{(N/2+Kn)} = -W_N^{Kn}$:-

Let N = 4 and Kn = 1

Now, $W_4^{(\frac{4}{2}+1)} = -W_4^1$

$W_4^{2+1} = -W_4^1$

$L.H.S = W_4^{2+1} = W_4^2 \cdot W_4^1 = e^{\frac{-j2\pi}{4} \cdot 2} \cdot e^{\frac{-j2\pi}{4} \cdot 1} = e^{-j\pi} \cdot e^{\frac{-j\pi}{2}} = (-1)(-j) = +j$

$R.H.S = -W_4^1 = -e^{\frac{-j2\pi}{4} \cdot 1} = -e^{\frac{-j\pi}{2}} = -(-j) = +j$

$\therefore \boxed{W_4^3 = -W_4^1}$

It satisfies the symmetry property.

Relationship of the DFT to other transforms :-

1) Relationship to Fourier Series Co-efficient of a periodic Sequence :-

A periodic signal $x_p(n)$ with a fundamental period 'N' can be represented in a Fourier series of the form,

$$x_p(n) = \sum_{K=0}^{N-1} C(K) e^{\frac{j2\pi}{N} Kn} \quad ; -\infty < n < \infty \longrightarrow (1)$$

Where C(K) is a Fourier-Series Co-efficient and it is computed by using,

$$C(K) = \frac{1}{N} \sum_{n=0}^{N-1} x_p(n) e^{\frac{-j2\pi}{N} Kn} \quad ; 0 \le K \le N-1$$

Scanned with CamScanner

But we know the IDFT equation,

$$x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K)\, e^{j\frac{2\pi}{N}Kn} \quad ; 0 \le n \le N-1 \longrightarrow (2)$$

on comparing Eq(1) and Eq(2),

$$C(K) = \frac{1}{N} X(K)$$

$\langle X(K) = N\,C(K) \rangle$ provided $x(n) = x_p(n)$; $0 \le n \le (N-1)$.

Hence, except for a scaling factor, the DFS and DFT relations are identical.

2) Relationship between DFT and DTFT :-

The DTFT is given by,

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\Omega n}$$

When this DTFT is sampled at 'N' equally spaced points, we get DFT X(K)

i.e., $X(K) = X(\Omega)\Big|_{\Omega = \frac{2\pi}{N}K} \quad ; 0 \le K \le N-1 \longrightarrow (1)$

$$\therefore X(K) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\frac{2\pi}{N}Kn} \quad ; 0 \le K \le N-1$$

W.K.T X(K) is also DFT of the periodic sequence $x_p(n)$ which repeats with period 'N'.

Then $x_p(n)$ and $x(n)$ are related as, $x_p(n) = \sum_{l=-\infty}^{\infty} x(n-lN)$

In otherwords, both $x_p(n)$ and $x(n)$ have same DFT if its length is 'N' samples. Here $x(n) = x_p(n)$; $0 \le n \le N-1$.

Here, the length of $x(n)$ must be less (or) equal to 'N' to hold the relationship given by Eq(1).

Scanned with CamScanner

3) Relationship between DFT and Z-transform :-

The Z-transform of a sequence $x(n)$ is given by,

$$X(z) = \sum_{n=-\infty}^{\infty} x(n)\, z^{-n} \longrightarrow (1)$$

with an ROC that includes the unit circle.

If $X(z)$ is sampled at the 'N' equally spaced points on the unit circle $z_k = e^{j2\pi k/N}$; $0 \le k \le N-1$

$$X(k) = X(z)\Big|_{z = e^{\frac{j2\pi}{N}k}}$$

$$X(k) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-\frac{j2\pi}{N}kn} \longrightarrow (2)$$

Eq(1) is identical to the Fourier transform $X(\omega)$ evaluated at the N equally spaced frequencies $\omega_k = \frac{2\pi}{N}k$; $0 \le k \le N-1$.

If the sequence $x(n)$ has a finite duration of length 'N' or less, the sequence can be recovered from its N-point DFT. Hence its Z-transform is uniquely determined by its N-point DFT. Consequently, $X(z)$ can be expressed as a function of the DFT, $X(k)$ as follows :

$$X(z) = \sum_{n=0}^{N-1} x(n)\, z^{-n}$$

$$X(z) = \sum_{n=0}^{N-1} \left[\frac{1}{N} \sum_{k=0}^{N-1} X(k)\, e^{\frac{j2\pi}{N}kn} \right] z^{-n}$$

$$X(z) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) \sum_{n=0}^{N-1} \left(e^{\frac{j2\pi}{N}k}\, z^{-1} \right)^n \longrightarrow (3)$$

$$X(z) = \frac{1 - z^{-N}}{N} \sum_{k=0}^{N-1} \frac{X(k)}{1 - e^{\frac{j2\pi}{N}k}\, z^{-1}}$$

When evaluated on the unit circle, Eq(3) yields the Fourier transform of the finite-duration sequence in terms of its DFT, in the form :

$$X(\omega) = \frac{1 - e^{-j\omega N}}{N} \sum_{k=0}^{N-1} \frac{X(k)}{1 - e^{-j(\omega - \frac{2\pi}{N}k)}} \longrightarrow (4)$$

Scanned with CamScanner

This expression for the Fourier transform is a polynomial (Lagrange) interpolation formula for $X(\omega)$ expressed in terms of the values $\{X(k)\}$ of the polynomial at a set of equally spaced discrete frequencies $\omega_k = \frac{2\pi}{N}k$; $0 \le k \le N-1$.

Important Theory Questions :-

1) Define DFT and IDFT.
2) Explain the frequency-domain sampling and reconstruction of discrete-time signals.
3) Obtain the relationship between i) DFT and DTFT ii) DFT and DFS iii) DFT and Z-transform.
4) State and prove i) periodicity / cyclic ii) Symmetry properties of twiddle factor.
5) What is Zero padding ? What are its uses ?

Scanned with CamScanner

Module-1

Anusha.M.N
Asst. professor
Dept. of ECE
BGSIT.

Discrete-Fourier Transform (DFT)

Problems on N-point DFT :

I. Compute the N-point DFT for the following sequences :

1) $x(n) = \delta(n)$

Soln:-

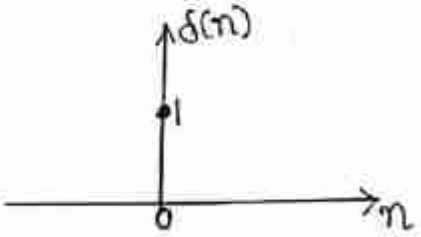
W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$= \sum_{n=0}^{N-1} \delta(n) W_N^{Kn}$

$= 1 . W_N^{Kn}\Big|_{n=0}$

$= e^{-j\frac{2\pi}{n} K \times 0}$

$\langle X(K) = 1 \rangle$



2) $x(n) = \delta(n-n_0)$

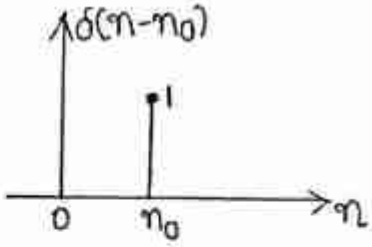
Soln:-

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$= \sum_{n=0}^{N-1} \delta(n-n_0)\, e^{-j\frac{2\pi}{N}Kn}$

$= 1 . e^{-j\frac{2\pi}{N}Kn}\Big|_{n=n_0}$

$\langle X(K) = e^{-j\frac{2\pi}{N}Kn_0} \rangle$

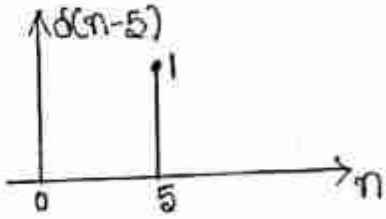


3) $x(n) = \delta(n-5)$

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$= \sum_{n=0}^{N-1} \delta(n-5)\, e^{-j\frac{2\pi}{N}Kn}$

$X(K) = 1 . e^{-j\frac{2\pi}{N}Kn}\Big|_{n=5} = e^{-j\frac{10\pi K}{N}}$



Scanned with CamScanner

4) $x(n) = u(n) - u(n-n_0)$

Soln:-

$$X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$$

$$= \sum_{n=0}^{n_0-1} 1 \cdot e^{-j\frac{2\pi}{N}Kn}$$

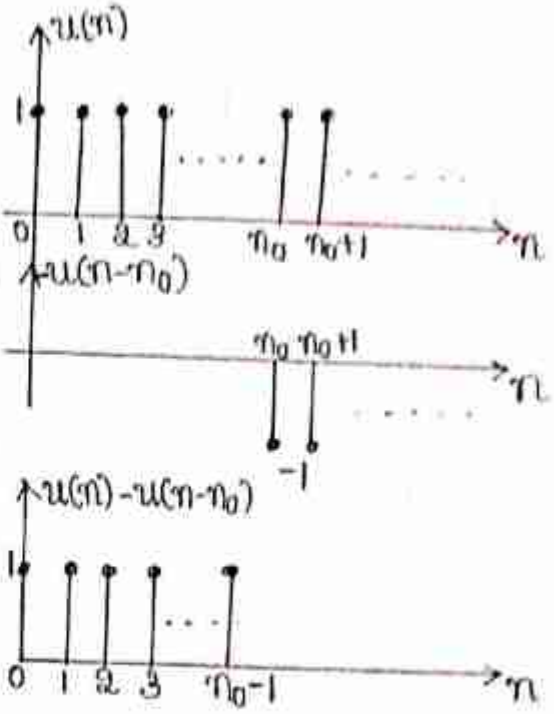
$$= \sum_{n=0}^{n_0-1} \left(e^{-j\frac{2\pi}{N}K}\right)^n$$

$$= \frac{1 - \left(e^{-j\frac{2\pi}{N}K}\right)^{n_0-1+1}}{1 - e^{-j\frac{2\pi}{N}K}}$$

$$= \frac{1 - e^{-j\frac{2\pi}{N}Kn_0}}{1 - e^{-j\frac{2\pi}{N}K}}$$

$$= \frac{e^{-j\frac{\pi}{N}Kn_0}\left(e^{j\frac{\pi}{N}Kn_0} - e^{-j\frac{\pi}{N}Kn_0}\right)}{e^{-j\frac{\pi}{N}K}\left(e^{j\frac{\pi}{N}K} - e^{-j\frac{\pi}{N}K}\right)}$$

$$X(K) = e^{-j\frac{\pi}{N}K(n_0-1)} \cdot \frac{\sin\left(\frac{\pi}{N}Kn_0\right)}{\sin\left(\frac{\pi}{N}K\right)}$$



$$\sum_{n=n_1}^{n_2} a^n = \frac{a^{n_1} - a^{n_2+1}}{1-a}$$

5) $x(n) = 1$

Soln:-

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$

$$= \sum_{n=0}^{N-1} 1 \cdot e^{-j\frac{2\pi}{N}Kn}$$

$$X(K) = \sum_{n=0}^{N-1} \left(e^{-j\frac{2\pi}{N}K}\right)^n \longrightarrow (1)$$

$$X(K) = \frac{1 - \left(e^{-j\frac{2\pi}{N}K}\right)^{N-1+1}}{1 - e^{-j\frac{2\pi}{N}K}} = 0 \; ; \; K \neq 0$$

Scanned with CamScanner

When $K=0$ in Eq(1),

$$X(K) = \sum_{n=0}^{N-1} 1 = N-1+1 = N\,\delta(K)$$

$$\sum_{n=n_1}^{n_2} 1 = n_2 - n_1 + 1$$

$$\therefore X(K) = \begin{cases} N\delta(K) & ; K=0 \\ 0 & ; K\neq 0 \end{cases}$$

6) $x(n) = a^n$ for $0<a<1$

Sol^n:-

$$X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$$

$$= \sum_{n=0}^{N-1} a^n\, e^{-j\frac{2\pi}{N}Kn}$$

$$= \sum_{n=0}^{N-1} \left(a e^{-j\frac{2\pi}{N}K}\right)^n$$

$$= \frac{1-\left(a e^{-j\frac{2\pi}{N}K}\right)^{N-1+1}}{1-a e^{-j\frac{2\pi}{N}K}}$$

$$= \frac{1-a^N e^{-j\frac{2\pi}{N}K.N}}{1-a e^{-j\frac{2\pi}{N}K}}$$

$$\boxed{X(K) = \frac{1-a^N}{1-a e^{-j\frac{2\pi}{N}K}}}$$

7) $x(n) = \cos\left(\frac{2\pi K_0 n}{N}\right)$; $0 \le n \le N-1$

(or)

$x(n) = \cos(w_0 n)$; $w_0 = \frac{2\pi}{N} K_0$; $0 \le n \le N-1$.

Sol:- $x(n) = \frac{e^{j\frac{2\pi}{N}K_0 n} + e^{-j\frac{2\pi}{N}K_0 n}}{2} = \frac{W_N^{-K_0 n} + W_N^{K_0 n}}{2}$

Scanned with CamScanner

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$$= \sum_{n=0}^{N-1} \left[\frac{W_N^{-K_0 n} + W_N^{K_0 n}}{2} \right] W_N^{Kn}$$

$$= \frac{1}{2}\left[\sum_{n=0}^{N-1} e^{\frac{j2\pi}{N} K_0 n} \cdot e^{-\frac{j2\pi}{N} Kn} + \sum_{n=0}^{N-1} e^{-\frac{j2\pi}{N} K_0 n} \cdot e^{-\frac{j2\pi}{N} Kn} \right]$$

$$= \frac{1}{2}\left\{ \sum_{n=0}^{N-1} \left[e^{-\frac{j2\pi}{N}(K-K_0)} \right]^n + \sum_{n=0}^{N-1} \left[e^{-\frac{j2\pi}{N}(K+K_0)} \right]^n \right\}$$

$$X(K) = \frac{1}{2}\left[S_1 + S_2 \right]$$

To find S_1 and S_2:

$$S_1 = \sum_{n=0}^{N-1} \left[e^{-\frac{j2\pi}{N}(K-K_0)} \right]^n \longrightarrow (1)$$

$$= \frac{1 - \left(e^{-\frac{j2\pi}{N}(K-K_0)} \right)^{N-1+1}}{1 - e^{-\frac{j2\pi}{N}(K-K_0)}}$$

$\langle S_1 = 0 \; ; \; K \neq K_0 \rangle$

When $K = K_0$ in Eq(1),

$$S_1 = \sum_{n=0}^{N-1} 1 = N - 1 - 0 + 1 = N\delta(K-K_0)$$

$$\therefore S_1 = \begin{cases} N\delta(K-K_0) & ; K = K_0 \\ 0 & ; K \neq K_0 \end{cases}$$

Similarly, $S_2 = \begin{cases} N\delta(K+K_0) & ; K = -K_0 \\ 0 & ; K \neq -K_0 \end{cases}$

Scanned with CamScanner

$$\therefore X(K) = \frac{1}{2}\left[N\delta(K-K_0) + N\delta(K+K_0)\right]$$

$$\left\langle X(K) = \frac{N}{2}\delta(K-K_0) + \frac{N}{2}\delta(K+K_0) \right\rangle$$

8) $x(n) = \sin\left(\frac{2\pi}{N}K_0 n\right)$; $0 \le n \le N-1$

(or)

$x(n) = \sin(W_0 n)$; $W_0 = \frac{2\pi}{N}K_0$.

Sol^n:- $x(n) = \dfrac{e^{j\frac{2\pi}{N}K_0 n} - e^{-j\frac{2\pi}{N}K_0 n}}{2j} = \dfrac{W_N^{-K_0 n} - W_N^{K_0 n}}{2j}$

$$X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$$

$$= \sum_{n=0}^{N-1}\left[\frac{W_N^{-K_0 n} - W_N^{K_0 n}}{2j}\right] W_N^{Kn}$$

$$= \frac{1}{2j}\left[\sum_{n=0}^{N-1} W_N^{-K_0 n}\cdot W_N^{Kn} - \sum_{n=0}^{N-1} W_N^{K_0 n}\, W_N^{Kn}\right]$$

$$= \frac{1}{2j}\left\{\sum_{n=0}^{N-1}\left[e^{-j\frac{2\pi}{N}(K-K_0)}\right]^n - \sum_{n=0}^{N-1}\left[e^{-j\frac{2\pi}{N}(K+K_0)}\right]^n\right\}$$

$$X(K) = \frac{1}{2j}\left[S_1 - S_2\right]$$

To find S_1 and S_2:

$$S_1 = \sum_{n=0}^{N-1}\left[e^{-j\frac{2\pi}{N}(K-K_0)}\right]^n = \frac{1-\left[e^{-j\frac{2\pi}{N}(K-K_0)}\right]^{N-1+1}}{1-e^{-j\frac{2\pi}{N}(K-K_0)}} = \underline{\underline{0}} \; ; \; K \neq K_0$$

When $K = K_0$,

$$S_1 = \sum_{n=0}^{N-1} 1 = N-1+1 = \underline{\underline{N\delta(K-K_0)}}$$

Scanned with CamScanner

$\therefore S_1 = \begin{cases} N\delta(K-K_0) \; ; \; K = K_0 \\ 0 \quad ; \; K \neq K_0 \end{cases}$

Similarly, $S_2 = \begin{cases} N\delta(K+K_0) \; ; \; K = -K_0 \\ 0 \quad ; \; K \neq K_0 \end{cases}$

$\therefore X(K) = \frac{1}{2j}\left[N\delta(K-K_0) - N\delta(K+K_0)\right]$

$$\boxed{X(K) = \frac{N}{2j}\delta(K-K_0) - \frac{N}{2j}\delta(K+K_0)}$$

9) $x(n) = an \; ; 0 \leq n \leq N-1$.

Sol^n:-

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} = \sum_{n=0}^{N-1} an\, W_N^{Kn} = a\sum_{n=0}^{N-1} n\, W_N^{Kn} \longrightarrow (1)$$

Let $S = \sum_{n=0}^{N-1} b^n = \frac{1-b^N}{1-b} = \frac{b^N-1}{b-1} \; ; b \neq 1$

Differentiating both sides of the above Eq^n w.r.t 'b',

$$\sum_{n=0}^{N-1} n b^{n-1} = \frac{(b-1)(Nb^{N-1}-0) - (b^N-1)(1-0)}{(b-1)^2}$$

$$\sum_{n=0}^{N-1} n b^n = \frac{b\left[Nb^N - Nb^{N-1} - b^N + 1\right]}{(b-1)^2}$$

$$= \frac{b\left[b^N(N-1) - Nb^{N-1} + 1\right]}{(b-1)^2}$$

Let $b = W_N^K$ in the above Eq^n,

$$\sum_{n=0}^{N-1} n.W_N^{Kn} = \frac{W_N^K\left[W_N^{KN}(N-1) - N\,W_N^{K(N-1)} + 1\right]}{(W_N^K - 1)^2}$$

Scanned with CamScanner

9) $$\sum_{n=0}^{N-1} n W_N^{Kn} = \frac{W_N^K \left[N-1-NW_N^{-K}+1\right]}{(W_N^K-1)^2}$$

$$= \frac{N W_N^K (1-W_N^{-K})}{(W_N^K-1)^2}$$

$$\sum_{n=0}^{N-1} n W_N^{Kn} = \frac{N(W_N^K-1)}{(W_N^K-1)^2} = \frac{N}{(W_N^K-1)} \quad ; K \neq 0$$

When $K=0$ in Eq(1),

$$X(K) = a \sum_{n=0}^{N-1} n = a\frac{N(N-1)}{2}$$

$$\therefore X(K) = \begin{cases} a\frac{N(N-1)}{2} & ; K=0 \\ \frac{aN}{(W_N^K-1)} & ; K \neq 0 \end{cases}$$

10) $x(n) = e^{j\omega mn}$, where $\omega = \frac{2\pi}{N}$

Sol^n:-

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{N-1} e^{j\frac{2\pi}{N}mn} e^{-j\frac{2\pi}{N}Kn}$$

$$= \sum_{n=0}^{N-1} \left[e^{-j\frac{2\pi}{N}(K-m)} \right]^n$$

$$= \frac{1-\left[e^{-j\frac{2\pi}{N}(K-m)}\right]^{N-1+1}}{1-e^{-j\frac{2\pi}{N}(K-m)}}$$

$X(K) = 0 \quad ; K \neq m$

Scanned with CamScanner

When $K = rN$,

$$X(K) = \sum_{n=0}^{N-1} 1 = N - 1 + 1 = N\delta(K - rN)$$

$$\therefore X(K) = \begin{cases} N\delta(K-rN) & ; K = rN \\ 0 & ; K \neq rN \end{cases}$$

11) $$x(n) = \begin{cases} \frac{1}{3} & ; 0 \le n \le 2 \\ 0 & ; \text{otherwise} \end{cases}$$

Sol^n :-

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{2} \left(\frac{1}{3}\right) W_N^{Kn}$$

$$= \frac{1}{3} \sum_{n=0}^{2} \left(e^{-j\frac{2\pi}{N}K}\right)^n$$

$$= \frac{1}{3}\left[\frac{1 - \left(e^{-j\frac{2\pi}{N}K}\right)^3}{1 - e^{-j\frac{2\pi}{N}K}}\right]$$

$$\left\langle X(K) = \frac{\frac{1}{3} - \frac{1}{3} e^{-j6\pi K/N}}{1 - e^{-j\frac{2\pi}{N}K}} \right\rangle$$

12) Find the DFT of the sequence $x(n) = 0.5^n u(n)$ for $0 \le n \le 3$ by evaluating $x(n) = a^n$ for $0 \le n \le N-1$.

Sol^n :- W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$$= \sum_{n=0}^{N-1} a^n e^{-j\frac{2\pi}{N}Kn}$$

$$= \sum_{n=0}^{N-1} \left(a e^{-j\frac{2\pi}{N}K}\right)^n$$

Scanned with CamScanner

$$X(K) = \frac{1-\left(a\,e^{-j\frac{2\pi}{N}K}\right)^{N-1+1}}{1-a\,e^{-j\frac{2\pi}{N}K}}$$

$$\left\langle X(K) = \frac{1-a^N}{1-a\,e^{-j\frac{2\pi}{N}K}} \right\rangle \longrightarrow (1)$$

To determine the DFT of $x(n) = (0.5)^n u(n)$; $n = 0,1,2,3$

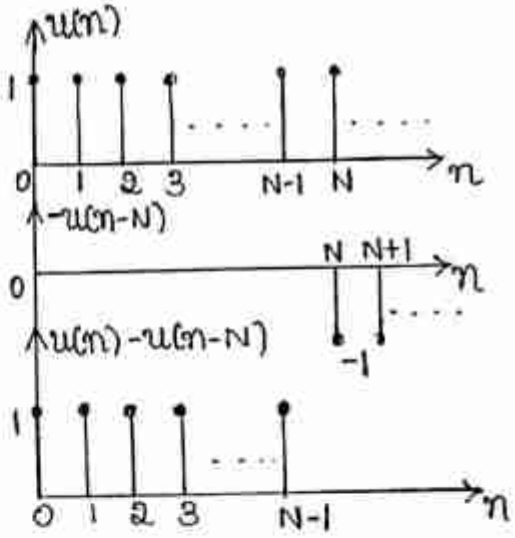
Here $N = 4$ and $a = 0.5$

Substitute these values in Eq(1),

$$X(K) = \frac{1-(0.5)^4}{1-0.5\,e^{-j\frac{2\pi}{4}K}} = \frac{0.9375}{1-0.5\,e^{-j\frac{\pi}{2}K}} \quad ; 0 \le K \le 3$$

$$\left\langle X(K) = \{1.875, 0.75-j0.375, 0.625, 0.75+j0.375\} \right\rangle$$

13) $x(n) = u(n) - u(n-N)$



Sol^n:-

$$x(n) = \begin{cases} 1 & ; 0 \le n \le N-1 \\ 0 & ; \text{else} \end{cases}$$

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$

$$= \sum_{n=0}^{N-1} 1 \cdot e^{-j\frac{2\pi}{N}Kn}$$

$$= \sum_{n=0}^{N-1} \left(e^{-j\frac{2\pi}{N}K}\right)^n$$

$$X(K) = \frac{1-\left(e^{-j\frac{2\pi}{N}K}\right)^{N-1+1}}{1-e^{-j\frac{2\pi}{N}K}} = 0 \quad ; K \neq 0$$

When $K = 0$, $X(K) = \sum_{n=0}^{N-1} 1 = N-1-0+1 = N\,\delta(K)$

$$\therefore X(K) = \begin{cases} N\delta(K) & ; K = 0 \\ 0 & ; K \neq 0 \end{cases}$$

Scanned with CamScanner

Problems on DFT as a linear Transformation (Matrix method):

Compute the DFT of the following sequences using matrix method:

1) $x(n) = \{0, 1, 2, 3\}$

Soln:- $X_N = W_N x_N$

Here $N = 4$,

$X_4 = W_4 x_4$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix}$$

$$= \begin{bmatrix} 0+1+2+3 \\ 0-j-2+j3 \\ 0-1+2-3 \\ 0+j-2-j3 \end{bmatrix} = \begin{bmatrix} 6 \\ -2+j2 \\ -2 \\ -2-j2 \end{bmatrix}$$

$\therefore X(K) = \{6, -2+j2, -2, -2-j2\}$

2) $x(n) = \{1, 0, 0, 1\}$

Soln:- $X_N = W_N x_N$

Here $N = 4$,

$X_4 = W_4 x_4$

$W_4^0 = 1$; $W_4^1 = -j$
$W_4^2 = -1$; $W_4^3 = j$

Scanned with CamScanner

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1+0+0+1 \\ 1+0+0+j \\ 1+0+0-1 \\ 1+0+0-j \end{bmatrix}$$

$$= \begin{bmatrix} 2 \\ 1+j \\ -1 \\ 1-j \end{bmatrix}$$

$\therefore X(K) = \{2, 1+j, 0, 1-j\}$

3) $x(n) = \sin^2\left(\frac{n\pi}{2} + \frac{\pi}{3}\right)$, Compute 4-point DFT using matrix method.

Sol^n:- Here $N = 4$,

@ $n=0$, $x(0) = \sin^2\left(0 \times \frac{\pi}{2} + \frac{\pi}{3}\right) = \frac{3}{4}$

@ $n=1$, $x(1) = \sin^2\left(1 \times \frac{\pi}{2} + \frac{\pi}{3}\right) = \frac{1}{4}$

@ $n=2$, $x(2) = \sin^2\left(2 \times \frac{\pi}{2} + \frac{\pi}{3}\right) = \frac{3}{4}$

@ $n=3$, $x(3) = \sin^2\left(3 \times \frac{\pi}{2} + \frac{\pi}{3}\right) = \frac{1}{4}$

$\therefore x(n) = \left\{\frac{3}{4}, \frac{1}{4}, \frac{3}{4}, \frac{1}{4}\right\}$

Scanned with CamScanner

W.K.T $X_N = W_N x_N$

Here $N=4$, $X_4 = W_4 x_4$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 3/4 \\ 1/4 \\ 3/4 \\ 1/4 \end{bmatrix}$$

$$= \begin{bmatrix} 2 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$\therefore X(K) = \{2, 0, 1, 0\}$

$W_4^0 = 1 \; ; \; W_4^1 = -j$

$W_4^2 = -1 \; ; \; W_4^3 = j$

4) $x(n) = \cos^2\left(\frac{n\pi}{2} + \frac{\pi}{8}\right)$

Sol:- $x(n) = \left\{\frac{1}{4}, \frac{3}{4}, \frac{1}{4}, \frac{3}{4}\right\}$

$X_N = W_N x_N$

Here $N=4$, $X_4 = W_4 x_4$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1/4 \\ 3/4 \\ 1/4 \\ 3/4 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ -1 \\ 0 \end{bmatrix}$$

$\therefore X(K) = \{2, 0, -1, 0\}$

$W_4^0 = 1 \; ; \; W_4^1 = -j$

$W_4^2 = -1 \; ; \; W_4^3 = j$

Scanned with CamScanner

Compute the IDFT for the following sequences using matrix method:

1) $X(K) = \{4, -j2, 0, j2\}$

Sol:- $x_N = \frac{1}{N} W_N^{*} X_N$

$W_4^{-0} = 1 \; ; \; W_4^{-2} = -1$

$W_4^{-1} = j \; ; \; W_4^{-3} = -j$

Here $N = 4$,

$$x_4 = \frac{1}{4} W_4^{*} X_4$$

$$\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} W_4^{-0} & W_4^{-0} & W_4^{-0} & W_4^{-0} \\ W_4^{-0} & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ W_4^{-0} & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ W_4^{-0} & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 4 \\ -j2 \\ 0 \\ j2 \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 4 - j2 + 0 + j2 \\ 4 + 2 + 0 + 2 \\ 4 + j2 + 0 - j2 \\ 4 - 2 + 0 - 2 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 4 \\ 8 \\ 4 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 2 \\ 1 \\ 0 \end{bmatrix}$$

$\therefore x(n) = \{1, 2, 1, 0\}$

Scanned with CamScanner

2) $X(K) = \{2, 1+j, 0, 1-j\}$

Soln:- W.K.T $x_N = \frac{1}{N} W_N^* X_N$

Here $N=4$, $x_4 = \frac{1}{4} W_4^* X_4$

$W_4^{-0} = 1$; $W_4^{-1} = j$

$W_4^{-2} = -1$; $W_4^{-3} = -j$

$$\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} W_4^{-0} & W_4^{-0} & W_4^{-0} & W_4^{-0} \\ W_4^{-0} & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ W_4^{-0} & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ W_4^{-0} & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 2 \\ 1+j \\ 0 \\ 1-j \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 2+1+j+0+1-j \\ 2+j-1+0-j-1 \\ 2-1-j+0-1+j \\ 2-j+1+0+j+1 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 4 \\ 0 \\ 0 \\ 4 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$\therefore x(n) = \{1, 0, 0, 1\}$

Scanned with CamScanner

Problems on DFT and IDFT :

1) Find the 4-point DFT of the sequence $x(n) = \{1, 2, 3, 4\}$

Sol^n:- Given $N = 4$,

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$$= x(0)W_4^0 + x(1)W_4^K + x(2)W_4^{2K} + x(3)W_4^{3K}$$

$$X(K) = 1 + 2W_4^K + 3W_4^{2K} + 4W_4^{3K} \longrightarrow (1)$$

$W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = 1$

$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = -j$

$W_4^2 = e^{\frac{-j2\pi}{4} \cdot 2} = -1$

$W_4^3 = e^{\frac{-j2\pi}{4} \cdot 3} = j$

@K=0, $X(0) = 1 + 2W_4^0 + 3W_4^0 + 4W_4^0 = 1+2+3+4 = \underline{10}$

@K=1, $X(1) = 1 + 2W_4^1 + 3W_4^2 + 4W_4^3 = 1+2(-j)+3(-1)+4(j) = \underline{-2+j2}$

@K=2, $X(2) = 1 + 2W_4^2 + 3W_4^4 + 4W_4^6 = 1+2(-1)+3(1)+4(-1) = \underline{-2}$

@K=3, $X(3) = 1 + 2W_4^3 + 3W_4^6 + 4W_4^9 = 1+2(j)+3(-1)+4(-j) = \underline{-2-j2}$

$\therefore X(K) = \{10, -2+j2, -2, -2-j2\}$

2) $x(n) = \{1, 0, 0, 1\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N = 4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$= x(0)W_4^0 + x(1)W_4^K + x(2)W_4^{2K} + x(3)W_4^{3K}$$

$$X(K) = 1 + W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$; $W_4^2 = -1$

$W_4^1 = -j$; $W_4^3 = j$

@K=0, $X(0) = 1 + W_4^0 = 1+1 = \underline{2}$

@K=1, $X(1) = 1 + W_4^3 = \underline{1+j}$

@K=2, $X(2) = 1 + W_4^6 = 1-1 = \underline{0}$

@K=3, $X(3) = 1 + W_4^9 = \underline{1-j}$

$\therefore X(K) = \{2, 1+j, 0, 1-j\}$

Scanned with CamScanner

3) $x(n) = \{1, 0, 1, 0\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = 1 + W_4^{2K} \longrightarrow (1)$$

@ $K=0$, $X(0) = 1 + W_4^0 = 1+1 = \underline{2}$

@ $K=1$, $X(1) = 1 + W_4^2 = 1-1 = \underline{0}$

@ $K=2$, $X(2) = 1 + W_4^4 = 1+1 = \underline{2}$

@ $K=3$, $X(3) = 1 + W_4^6 = 1-1 = \underline{0}$

$\therefore X(K) = \{2, 0, 2, 0\}$

$W_4^0 = 1$; $W_4^2 = -1$

$W_4^1 = -j$; $W_4^3 = j$

4) $x(n) = \{1, 1, 2, 2\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = 1 + W_4^{K} + 2W_4^{2K} + 2W_4^{3K} \longrightarrow (1)$$

@ $K=0$, $X(0) = 1 + W_4^0 + 2W_4^0 + 2W_4^0 = 1+1+2+2 = \underline{6}$

@ $K=1$, $X(1) = 1 + W_4^1 + 2W_4^2 + 2W_4^3 = 1 - j + 2(-1) + 2(j) = \underline{-1+j}$

@ $K=2$, $X(2) = 1 + W_4^2 + 2W_4^4 + 2W_4^6 = 1 - 1 + 2(1) + 2(-1) = \underline{0}$

@ $K=3$, $X(3) = 1 + W_4^3 + 2W_4^6 + 2W_4^9 = 1 + j + 2(-1) + 2(-j) = \underline{-1-j}$

$\therefore X(K) = \{6, -1+j, 0, -1-j\}$

$W_4^0 = 1$; $W_4^1 = -j$

$W_4^2 = -1$; $W_4^3 = j$

5) $x(n) = \{0, 1, 2, 3\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = W_4^{K} + 2W_4^{2K} + 3W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$; $W_4^1 = -j$

$W_4^2 = -1$; $W_4^3 = j$

Scanned with CamScanner

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$$X(K) = 6 + 7W_4^K + 6W_4^{2K} + 5W_4^{3K} \longrightarrow (1)$$

@K=0, $X(0) = 6 + 7W_4^0 + 6W_4^0 + 5W_4^0 = 6+7+6+5 = \underline{24}$

@K=1, $X(1) = 6 + 7W_4^1 + 6W_4^2 + 5W_4^3 = 6+7(-j)+6(-1)+5(j) = \underline{-j2}$

@K=2, $X(2) = 6 + 7W_4^2 + 6W_4^4 + 5W_4^6 = 6+7(-1)+6(1)+5(-1) = \underline{0}$

@K=3, $X(3) = 6 + 7W_4^3 + 6W_4^6 + 5W_4^9 = 6+7(j)+6(-1)+5(-j) = \underline{j2}$

$\therefore X(K) = \{24, -j2, 0, j2\}$

8) Compute the DFT of the sequence $x(n) = \cos\left(\frac{n\pi}{4}\right)$ for N=4, plot |X(K)| and $\underline{|X(K)}$.

Soln:- Here N=4,

@n=0, $x(0) = \cos\left(\frac{\pi}{4} \times 0\right) = \underline{1}$

@n=1, $x(1) = \cos\left(\frac{\pi}{4} \times 1\right) = \underline{0.7071}$

@n=2, $x(2) = \cos\left(\frac{\pi}{4} \times 2\right) = \underline{0}$

@n=3, $x(3) = \cos\left(\frac{\pi}{4} \times 3\right) = \underline{-0.7071}$

$\therefore x(n) = \{1, 0.7071, 0, -0.7071\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

$$X(K) = 1 + 0.7071W_4^K - 0.7071W_4^{3K} \longrightarrow (1)$$

@K=0, $X(0) = 1 + 0.7071W_4^0 - 0.7071W_4^0 = 1+0.7071-0.7071 = \underline{1}$

@K=1, $X(1) = 1 + 0.7071W_4^1 - 0.7071W_4^3 = 1+0.7071(-j)-0.7071(j)$

$= \underline{1 - j\,1.4142}$

Scanned with CamScanner

@K=0, $X(0) = W_4^0 + 2W_4^0 + 3W_4^0 = 1+2+3 = \underline{6}$

@K=1, $X(1) = W_4^1 + 2W_4^2 + 3W_4^3 = -j + 2(-1) + 3(j) = \underline{-2+j2}$

@K=2, $X(2) = W_4^2 + 2W_4^4 + 3W_4^6 = -1 + 2(1) + 3(-1) = \underline{-2}$

@K=3, $X(3) = W_4^3 + 2W_4^6 + 3W_4^9 = j + 2(-1) + 3(-j) = \underline{-2-j2}$

$\therefore X(K) = \{6, -2+j2, -2, -2-j2\}$

6) $x(n) = \{1, -2, 3, 4\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here N=4, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = 1 - 2W_4^K + 3W_4^{2K} + 4W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$; $W_4^2 = -1$

$W_4^1 = -j$; $W_4^3 = j$

@K=0, $X(0) = 1 - 2W_4^0 + 3W_4^0 + 4W_4^0 = 1-2+3+4 = \underline{6}$

@K=1, $X(1) = 1 - 2W_4^1 + 3W_4^2 + 4W_4^3 = 1 - 2(-j) + 3(-1) + 4(j) = \underline{-2+j6}$

@K=2, $X(2) = 1 - 2W_4^2 + 3W_4^4 + 4W_4^6 = 1 - 2(-1) + 3(1) + 4(-1) = \underline{2}$

@K=3, $X(3) = 1 - 2W_4^3 + 3W_4^6 + 4W_4^9 = 1 - 2(j) + 3(-1) + 4(-j) = \underline{-2-j6}$

$\therefore X(K) = \{6, -2+j6, 2, -2-j6\}$

7) Find the 4-point DFT of the sequence $x(n) = 6 + \sin\left(\frac{2\pi n}{4}\right)$; $0 \le n \le 3$.

Sol^n:- Here N=4,

@n=0, $x(0) = 6 + \sin\left(\frac{2\pi}{4}.0\right) = \underline{6}$

@n=1, $x(1) = 6 + \sin\left(\frac{2\pi}{4}.1\right) = \underline{7}$

@n=2, $x(2) = 6 + \sin\left(\frac{2\pi}{4}.2\right) = \underline{6}$

@n=3, $x(3) = 6 + \sin\left(\frac{2\pi}{4}.3\right) = \underline{5}$

$\therefore x(n) = \{6, 7, 6, 5\}$

$W_4^0 = 1$

$W_4^1 = -j$

$W_4^2 = -1$

$W_4^3 = j$

Scanned with CamScanner

@K=2, $X(2) = 1 + 0.7071 W_4^2 - 0.7071 W_4^6 = 1 + 0.7071(-1) - 0.7071(-1) = \underline{1}$

@K=3, $X(3) = 1 + 0.7071 W_4^3 - 0.7071 W_4^9 = 1 + 0.7071(j) - 0.7071(-j)$

$= 1 + j1.4142$

$\therefore X(K) = \{1, 1-j1.4142, 1, 1+j1.4142\}$

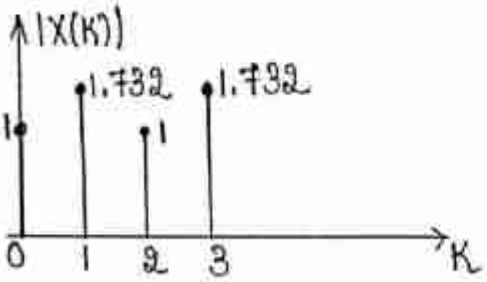
$|X(K)| = \{1, 1.732, 1, 1.732\}$

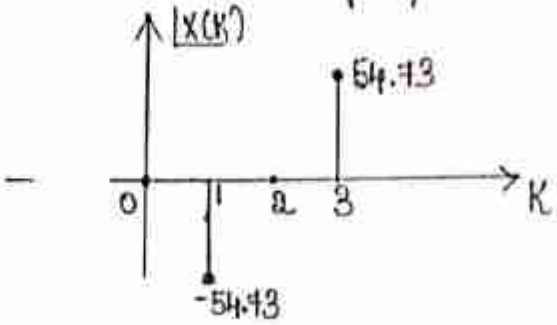
$\angle X(K) = \{0, -54.73, 0, 54.73\}$

$a+jb$

magnitude $= \sqrt{a^2+b^2}$

phase $= \tan^{-1}\left(\frac{b}{a}\right)$





9) $x(n) = \sin\left(\frac{n\pi}{2}\right)$; $0 \le n \le 3$.

Sol^n:-

@n=0, $x(0) = \sin\left(\frac{\pi}{2} \times 0\right) = 0$

@n=1, $x(1) = \sin\left(\frac{\pi}{2} \times 1\right) = 1$

@n=2, $x(2) = \sin\left(\frac{\pi}{2} \times 2\right) = 0$

@n=3, $x(3) = \sin\left(\frac{\pi}{2} \times 3\right) = -1$

$x(n) = \{0, 1, 0, -1\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$$X(K) = W_4^K - W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

@K=0, $X(0) = W_4^0 - W_4^0 = 1 - 1 = \underline{0}$

@K=1, $X(1) = W_4^1 - W_4^3 = -j - j = \underline{-j2}$

Scanned with CamScanner

@K = 2, $X(2) = W_4^2 - W_4^6 = -1-(-1) = \underline{0}$

@K = 3, $X(3) = W_4^3 - W_4^9 = j-(-j) = \underline{j2}$

$\therefore X(K) = \{0, -j2, 0, j2\}$

10) $x(n) = \cos\left(\frac{n\pi}{4}\right) + \sin\left(\frac{n\pi}{4}\right)$; $0 \le n \le 3$.

Sol^n:-

@n = 0, $x(0) = \cos\left(\frac{\pi}{4}\times 0\right) + \sin\left(\frac{\pi}{4}\times 0\right) = \underline{1}$

@n = 1, $x(1) = \cos\left(\frac{\pi}{4}\times 1\right) + \sin\left(\frac{\pi}{4}\times 1\right) = \underline{1.4142}$

@n = 2, $x(2) = \cos\left(\frac{\pi}{4}\times 2\right) + \sin\left(\frac{\pi}{4}\times 2\right) = \underline{1}$

@n = 3, $x(3) = \cos\left(\frac{\pi}{4}\times 3\right) + \sin\left(\frac{\pi}{4}\times 3\right) = \underline{0}$

$\therefore x(n) = \{1, 1.4142, 1, 0\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$X(K) = 1 + 1.4142 W_4^K + W_4^{2K} \longrightarrow (1)$$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

@K = 0, $X(0) = 1 + 1.4142 W_4^0 + W_4^0 = 1 + 1.4142 + 1 = \underline{3.4142}$

@K = 1, $X(1) = 1 + 1.4142 W_4^1 + W_4^2 = 1 + 1.4142(-j) - 1 = \underline{-j1.4142}$

@K = 2, $X(2) = 1 + 1.4142 W_4^2 + W_4^4 = 1 + 1.4142(-1) + 1 = \underline{0.586}$

@K = 3, $X(3) = 1 + 1.4142 W_4^3 + W_4^6 = 1 + 1.4142(j) + (-1) = \underline{j1.4142}$

$\therefore X(K) = \{3.4142, -j1.4142, 0.586, j1.4142\}$

11) $x(n) = \begin{cases} 1/3 & ; 0 \le n \le 3 \\ 0 & ; \text{otherwise} \end{cases}$

Sol^n:- $x(n) = \{1/3, 1/3, 1/3, 1/3\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

Here N = 4,

$$X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn} = \frac{1}{3}\left[1 + W_4^K + W_4^{2K} + W_4^{3K}\right] \longrightarrow (1)$$

Scanned with CamScanner

@K=0, $X(0) = \frac{1}{3}[1+W_4^0+W_4^0+W_4^0] = 4/3$

@K=1, $X(1) = \frac{1}{3}[1+W_4^1+W_4^2+W_4^3] = \frac{1}{3}[1-j-1+j] = 0$

@K=2, $X(2) = \frac{1}{3}[1+W_4^2+W_4^4+W_4^6] = \frac{1}{3}[1-1+1-1] = 0$

@K=3, $X(3) = \frac{1}{3}[1+W_4^3+W_4^6+W_4^9] = \frac{1}{3}[1+j-1-j] = 0$

$\therefore X(K) = \{4/3, 0, 0, 0\}$

12) $x(n) = u(n) - u(n-4)$

Sol^n:- $x(n) = \{1, 1, 1, 1\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

Here N = 4,

$$X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$$X(K) = 1 + W_4^K + W_4^{2K} + W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

@K=0, $X(0) = 1+W_4^0+W_4^0+W_4^0 = 1+1+1+1 = 4$

@K=1, $X(1) = 1+W_4^1+W_4^2+W_4^3 = 1-j-1+j = 0$

@K=2, $X(2) = 1+W_4^2+W_4^4+W_4^6 = 1-1+1-1 = 0$

@K=3, $X(3) = 1+W_4^3+W_4^6+W_4^9 = 1+j-1-j = 0$

$\therefore X(K) = \{4, 0, 0, 0\}$

13) Compute 5-point DFT of the sequence $x(n) = \{1, 2, 3, 4, 5\}$ using direct method.

Sol^n:- $$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$W_5^3 = -0.809 + j0.588$
$W_5^4 = 0.309 + j0.951$

$W_5^0 = e^{-j\frac{2\pi}{5}.0} = 1$
$W_5^1 = e^{-j\frac{2\pi}{5}.1} = 0.809 - j0.951$
$W_5^2 = -0.809 - j0.588$

Here,

$$X(K) = \sum_{n=0}^{4} x(n) W_5^{Kn} = 1 + 2W_5^K + 3W_5^{2K} + 4W_5^{3K} + 5W_5^{4K}$$

Scanned with CamScanner

@K=0, $X(0) = 1+2W_5^0+3W_5^0+4W_5^0+5W_5^0 = 1+2+3+4+5 = 15$

@K=1, $X(1) = 1+2W_5^1+3W_5^2+4W_5^3+5W_5^4 = 1+2(0.309-j0.951)+3(-0.809-j0.588)$
$+4(-0.809+j0.588)+5(0.309+j0.951)$

$X(1) = -2.5+j3.441$

@K=2, $X(2) = 1+2W_5^2+3W_5^4+4W_5^6+5W_5^8$

$= 1+2(-0.809-j0.588)+3(0.309+j0.951)+4(0.309-j0.951)+$
$5(-0.809+j0.588) = -2.5+j0.813$

@K=3, $X(3) = 1+2W_5^3+3W_5^6+4W_5^9+5W_5^{12}$

$= 1+2(-0.809+j0.588)+3(0.309-j0.951)+4(0.309+j0.951)$
$+5(-0.809-j0.588) = -2.5-j0.813$

@K=4, $X(4) = 1+2W_5^4+3W_5^8+4W_5^{12}+5W_5^{16}$

$= 1+2(0.309+j0.951)+3(-0.809+j0.588)+4(-0.809-j0.588)$
$+5(0.309-j0.951) = -2.5-j3.441$

$\therefore X(K) = \{15, -2.5+j3.441, -2.5+j0.813, -2.5-j0.813, -2.5-j3.441\}$

11) Compute a) 5-point DFT of the sequence $x_1(n) = \{1,1,1,0,0\}$ b) 3-point DFT of the sequence $x_2(n) = \{1,1,1\}$ by using direct method c) Compare the result of (a) and (b).

Soln:-

a) Given $x_1(n) = \{1,1,1,0,0\}$

$$X_1(K) = \sum_{n=0}^{N-1} x_1(n) W_N^{Kn}$$

Here N=5, $X_1(K) = \sum_{n=0}^{4} x_1(n) W_5^{Kn}$

$$X_1(K) = 1+W_5^K+W_5^{2K} \longrightarrow (1)$$

$W_5^0 = 1$

$W_5^1 = 0.309-j0.951$

$W_5^2 = -0.809-j0.588$

$W_5^3 = -0.809+j0.588$

$W_5^4 = 0.309+j0.951$

@K=0, $X_1(0) = 1+W_5^0+W_5^0 = 1+1+1 = 3$

@K=1, $X_1(1) = 1+W_5^1+W_5^2 = 1+0.309-j0.951-0.809-j0.588$

$= 0.5-j1.5387$

Scanned with CamScanner

@K=2, $X_1(2) = 1+W_5^2+W_5^4 = 1-0.809+j0.588+0.309+j0.951$

$= 0.5+j0.3633$

@K=3, $X_1(3) = 1+W_5^3+W_5^6 = 1+W_5^3+W_5^1 = 1-0.809+j0.5877+0.309-j0.951$

$= 0.5-j0.3633$

@K=4, $X_1(4) = 1+W_5^4+W_5^8 = 1+0.309+j0.951-0.809+j0.588$

$= 0.5+j1.5387$

$\therefore X_1(K) = \{3, 0.5-j1.5387, 0.5+j0.3633, 0.5-j0.3633, 0.5+j1.5387\}$

b) Given $x_2(n) = \{1, 1, 1\}$

$$X_2(K) = \sum_{n=0}^{N-1} x_2(n) W_N^{Kn}$$

$W_3^0 = e^{-j\frac{2\pi}{3}.0} = 1$

$W_3^1 = e^{-j\frac{2\pi}{3}.1} = -0.5-j0.866$

$W_3^2 = e^{-j\frac{2\pi}{3}.2} = -0.5+j0.866$

Here N=3, $X_2(K) = \sum_{n=0}^{2} x_2(n) W_3^{Kn}$

$$X_2(K) = 1+W_3^K+W_3^{2K} \longrightarrow (1)$$

@K=0, $X_2(0) = 1+W_3^0+W_3^0 = 1+1+1 = 3$

@K=1, $X_2(1) = 1+W_3^1+W_3^2 = 1-0.5-j0.866-0.5+j0.866 = 0$

@K=2, $X_2(2) = 1+W_3^2+W_3^4 = 1-0.5+j0.866-0.5-j0.866 = 0$

$\langle X_2(K) = \{3, 0, 0\} \rangle$

c) After comparing the outputs of (a) and (b), we get to know that $x_1(n) \neq x_2(n)$.

16) Compute a) 6-point DFT of the sequence $x_1(n) = \{1, 0, 1, 0, 1, 0\}$

b) 4-point DFT of the sequence $x_2(n) = \{4, 3, 1, 0\}$

Sol^n:- Given $x_1(n) = \{1, 0, 1, 0, 1, 0\}$

$$X_1(K) = \sum_{n=0}^{N-1} x_1(n) W_N^{Kn}$$

Here N=6, $X_1(K) = \sum_{n=0}^{5} x_1(n) W_6^{Kn} = 1+W_6^{2K}+W_6^{4K}$

Scanned with CamScanner

@K=0, $X_1(0) = 1+W_6^0+W_6^0 = 1+1+1 = \underline{3}$

@K=1, $X_1(1) = 1+W_6^2+W_6^4 = 1-0.5-j0.866-0.5+j0.866$
$= \underline{0}$

@K=2, $X_1(2) = 1+W_6^4+W_6^8 = 1-0.5+j0.866-0.5-j0.866$
$= \underline{0}$

@K=3, $X_1(3) = 1+W_6^6+W_6^{12} = 1+1+1 = \underline{3}$

@K=4, $X_1(4) = 1+W_6^8+W_6^{16} = 1-0.5-j0.866-0.5+j0.866$
$= \underline{0}$

@K=5, $X_1(5) = 1+W_6^{10}+W_6^{20} = 1-0.5+j0.866-0.5-j0.866 = \underline{0}$

$\langle \therefore X_1(K) = \{3, 0, 0, 3, 0, 0\} \rangle$

$W_6^0 = 1$
$W_6^1 = 0.5-j0.866$
$W_6^2 = -0.5-j0.866$
$W_6^3 = -1$
$W_6^4 = -0.5+j0.866$
$W_6^5 = 0.5+j0.866$

b) Given $x_2(n) = \{4, 3, 1, 0\}$

$$X_2(K) = \sum_{n=0}^{N-1} x_2(n) W_N^{Kn}$$

$W_4^0 = 1$; $W_4^1 = -j$
$W_4^2 = -1$; $W_4^3 = j$

Here N=4, $X_2(K) = \sum_{n=0}^{3} x_2(n) W_4^{Kn} = 4+3W_4^K+W_4^{2K} \longrightarrow (1)$

@K=0, $X_2(0) = 4+3W_4^0+W_4^0 = 4+3+1 = \underline{8}$

@K=1, $X_2(1) = 4+3W_4^1+W_4^2 = 4+3(-j)-1 = \underline{3-j3}$

@K=2, $X_2(2) = 4+3W_4^2+W_4^4 = 4+3(-1)+1 = \underline{2}$

@K=3, $X_2(3) = 4+3W_4^3+W_4^6 = 4+3(j)-1 = \underline{3+j3}$

$\langle \therefore X_2(K) = \{8, 3-j3, 2, 3+j3\} \rangle$

16) Derive the DFT of the data sequence $x(n) = \{1, 1, 2, 2, 3, 3\}$ and Compute the corresponding amplitude and phase spectrum.

Sol^n:- $$X(K) = \sum_{K=0}^{N-1} x(n) W_N^{Kn}$$

Here N=6, $X(K) = \sum_{K=0}^{5} x(n) W_6^{Kn}$

$W_6^0 = 1$; $W_6^1 = 0.5-j0.866$
$W_6^2 = -0.5-j0.866$, $W_6^3 = -1$
$W_6^4 = -0.5+j0.866$, $W_6^5 = 0.5+j0.866$

Scanned with CamScanner

$X(K) = 1 + W_6^K + 2W_6^{2K} + 2W_6^{3K} + 3W_6^{4K} + 3W_6^{5K} \rightarrow (1)$

@K=0, $X(0) = 1 + W_6^0 + 2W_6^0 + 2W_6^0 + 3W_6^0 + 3W_6^0 = 1+1+2+2+3+3$

$= 12$

@K=1, $X(1) = 1 + W_6^1 + 2W_6^2 + 2W_6^3 + 3W_6^4 + 3W_6^5$

$= 1 + 0.5 - j0.866 + 2(-0.5 - j0.866) + 2(-1) + 3(-0.5 + j0.866) +$
$3(0.5 + j0.866) = -1.5 + j2.598$

@K=2, $X(2) = 1 + W_6^2 + 2W_6^4 + 2W_6^6 + 3W_6^8 + 3W_6^{10}$

$= 1 - 0.5 - j0.866 + 2(-0.5 + j0.866) + 2(1) + 3(-0.5 - j0.866)$
$+ 3(-0.5 + j0.866) = -1.5 + j0.866$

@K=3, $X(3) = 1 + W_6^3 + 2W_6^6 + 2W_6^9 + 3W_6^{12} + 3W_6^{15}$

$= 1 - 1 + 2(1) + 2(-1) + 3(1) + 3(-1) = 0$

@K=4, $X(4) = 1 + W_6^4 + 2W_6^8 + 2W_6^{12} + 3W_6^{16} + 3W_6^{20}$

$= 1 - 0.5 + j0.866 + 2(-0.5 - j0.866) + 2(1) + 3(-0.5 + j0.866)$
$+ 3(-0.5 - j0.866) = -1.5 - j0.866$

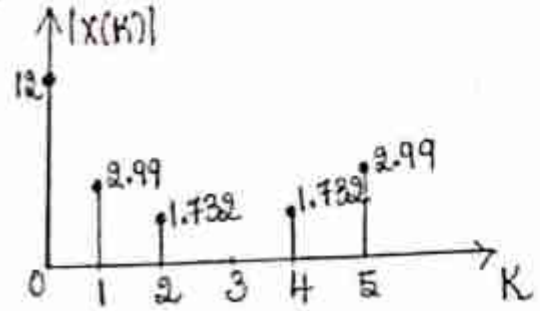
@K=5, $X(5) = 1 + W_6^5 + 2W_6^{10} + 2W_6^{15} + 3W_6^{20} + 3W_6^{25}$

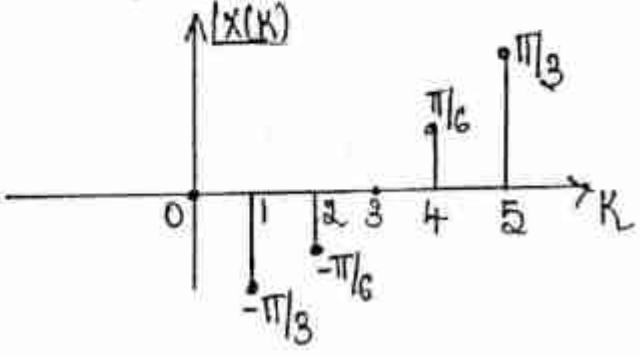
$= 1 + 0.5 + j0.866 + 2(-0.5 + j0.866) + 2(-1) + 3(-0.5 - j0.866)$
$+ 3(0.5 - j0.866) = -1.5 - j2.598$

$\therefore X(K) = \{12, -1.5 + j2.598, -1.5 + j0.866, 0, -1.5 - j0.866, -1.5 - j2.598\}$

Amplitude, $|X(K)| = \{12, 2.99, 1.732, 0, 1.732, 2.99\}$

phase, $\angle X(K) = \{0, -\pi/3, -\pi/6, 0, \pi/6, \pi/3\}$





Scanned with CamScanner

17) Compute the 6-point DFT of the sequence $x(n) = \{5, 4, 3, 2, 1, 0\}$ by using direct method.

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=6$,

$$X(K) = \sum_{n=0}^{5} x(n) W_6^{Kn}$$

$$X(K) = 5 + 4W_6^{K} + 3W_6^{2K} + 2W_6^{3K} + W_6^{4K} \longrightarrow (1)$$

$W_6^0 = 1$	$W_6^3 = -1$
$W_6^1 = 0.5 - j0.866$	$W_6^4 = -0.5 + j0.866$
$W_6^2 = -0.5 - j0.866$	$W_6^5 = 0.5 + j0.866$

@K=0, $X(0) = 5 + 4W_6^0 + 3W_6^0 + 2W_6^0 + W_6^0 = 5+4+3+2+1 = \underline{15}$

@K=1, $X(1) = 5 + 4W_6^1 + 3W_6^2 + 2W_6^3 + W_6^4$

$= 5 + 4(0.5 - j0.866) + 3(-0.5 - j0.866) + 2(-1) + (-0.5 + j0.866)$

$= \underline{3 - j5.196}$

@K=2, $X(2) = 5 + 4W_6^2 + 3W_6^4 + 2W_6^6 + W_6^8$

$= 5 + 4(-0.5 - j0.866) + 3(-0.5 + j0.866) + 2(1) - 0.5 - j0.866$

$= \underline{3 - j1.732}$

@K=3, $X(3) = 5 + 4W_6^3 + 3W_6^6 + 2W_6^9 + W_6^{12}$

$= 5 + 4(-1) + 3(1) + 2(-1) + 1 = \underline{3}$

@K=4, $X(4) = 5 + 4W_6^4 + 3W_6^8 + 2W_6^{12} + W_6^{16}$

$= 5 + 4(-0.5 + j0.866) + 3(-0.5 - j0.866) + 2(1) - 0.5 + j0.866$

$= \underline{3 + j1.732}$

@K=5, $X(5) = 5 + 4W_6^5 + 3W_6^{10} + 2W_6^{15} + W_6^{20}$

$= 5 + 4(0.5 + j0.866) + 3(-0.5 + j0.866) + 2(-1) - 0.5 - j0.866$

$= \underline{3 + j5.196}$

$\langle X(K) = \{15, 3 - j5.196, 3 - j1.732, 3, 3 + j1.732, 3 + j5.196\} \rangle$

Scanned with CamScanner

18) Determine the 8-point DFT of the sequence:
$x(n) = \{1, 1, 1, 1, 1, 1, 0, 0\}$

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N = 8$,

$$X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn}$$

$$X(K) = 1 + W_8^{K} + W_8^{2K} + W_8^{3K} + W_8^{4K} + W_8^{5K} \longrightarrow (1)$$

$W_8^0 = 1$	$W_8^4 = -1$
$W_8^1 = 0.7071 - j0.7071$	$W_8^5 = -0.7071 + j0.7071$
$W_8^2 = -j$	$W_8^6 = j$
$W_8^3 = -0.7071 - j0.7071$	$W_8^7 = 0.7071 + j0.7071$

@K=0, $X(0) = 1 + W_8^0 + W_8^0 + W_8^0 + W_8^0 + W_8^0 = 1+1+1+1+1+1 = \underline{6}$

@K=1, $X(1) = 1 + W_8^1 + W_8^2 + W_8^3 + W_8^4 + W_8^5$

$= 1 + 0.7071 - j0.7071 - j - 0.7071 - j0.7071 - 1 - 0.7071 + j0.7071$

$= \underline{-0.7071 - j1.7071}$

@K=2, $X(2) = 1 + W_8^2 + W_8^4 + W_8^6 + W_8^8 + W_8^{10}$

$= 1 - j - 1 + j + 1 - j = \underline{1 - j}$

@K=3, $X(3) = 1 + W_8^3 + W_8^6 + W_8^9 + W_8^{12} + W_8^{15} = 1 - 0.7071 - j0.7071 + j + 0.7071$
$- j0.7071 - 1 + 0.7071 + j0.7071 = \underline{0.7071 + j0.293}$

@K=4, $X(4) = 1 + W_8^4 + W_8^8 + W_8^{12} + W_8^{16} + W_8^{20}$

$= 1 - 1 + 1 - 1 + 1 - 1 = \underline{0}$

@K=5, $X(5) = 1 + W_8^5 + W_8^{10} + W_8^{15} + W_8^{20} + W_8^{25} = 1 - 0.7071 + j0.7071 - j +$
$0.7071 + j0.7071 - 1 + 0.7071 - j0.7071 = \underline{0.7071 - j0.293}$

@K=6, $X(6) = 1 + W_8^6 + W_8^{12} + W_8^{18} + W_8^{24} + W_8^{30} = 1 + j - 1 - j + 1 + j = \underline{1 + j}$

@K=7, $X(7) = 1 + W_8^7 + W_8^{14} + W_8^{21} + W_8^{28} + W_8^{35} = 1 + 0.7071 + j0.7071 + j +$
$-0.7071 + j0.7071 - 1 - 0.7071 - j0.7071 = \underline{-0.7071 + j0.7071}$

$\therefore X(K) = \{6, -0.7071 - j0.7071, 1 - j, 0.7071 + j0.293, 0, 0.7071 - j0.293,$
$1 + j, -0.7071 + j1.7071\}$

Scanned with CamScanner

19) Compute the 8-point DFT of the sequence $x(n) = \{1,1,1,1,0,0,0,0\}$ using direct method.

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=8$,

$$X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn}$$

$$X(K) = 1 + W_8^{K} + W_8^{2K} + W_8^{3K} \longrightarrow (1)$$

$W_8^0 = 1$
$W_8^1 = 0.7071 - j0.7071$
$W_8^2 = -j$
$W_8^3 = -0.7071 - j0.7071$
$W_8^4 = -1$
$W_8^5 = -0.7071 + j0.7071$
$W_8^6 = j$
$W_8^9 = 0.7071 + j0.7071$

@K=0, $X(0) = 1 + W_8^0 + W_8^0 + W_8^0 = 1+1+1+1 = \underline{4}$

@K=1, $X(1) = 1 + W_8^1 + W_8^2 + W_8^3 = 1 + 0.7071 - j0.7071 - j - 0.7071 - j0.7071$
$= \underline{1 - j2.4142}$

@K=2, $X(2) = 1 + W_8^2 + W_8^4 + W_8^6 = 1 - j - 1 + j = \underline{0}$

@K=3, $X(3) = 1 + W_8^3 + W_8^6 + W_8^9 = 1 - 0.7071 - j0.7071 + j + 0.7071 - j0.7071$
$= \underline{1 - j0.4142}$

@K=4, $X(4) = 1 + W_8^4 + W_8^8 + W_8^{12} = 1 - 1 + 1 - 1 = \underline{0}$

@K=5, $X(5) = 1 + W_8^5 + W_8^{10} + W_8^{15} = 1 - 0.7071 + j0.7071 - j + 0.7071 + j0.7071$
$= \underline{1 + j0.4142}$

@K=6, $X(6) = 1 + W_8^6 + W_8^{12} + W_8^{18} = 1 + j - 1 - j = \underline{0}$

@K=7, $X(7) = 1 + W_8^7 + W_8^{14} + W_8^{21} = 1 + 0.7071 + j0.7071 + j - 0.7071 + j0.7071$
$= \underline{1 + j2.4142}$

$\therefore X(K) = \{4, 1-j2.4142, 0, 1-j0.4142, 0, 1+j0.4142, 0, 1+j2.4142\}$

20) Compute the 8-point DFT of the sequence $x(n) = \{1, 4, 1, 0, 1, 0, 0, 0\}$ by direct method.

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn} = 1 + 4W_8^{K} + W_8^{2K} + W_8^{4K} \longrightarrow (1)$

Scanned with CamScanner

@K=0, $X(0) = 1+4W_8^0 + W_8^0 + W_8^0 = 1+4+1+1 = 7$

@K=1, $X(1) = 1+4W_8^1 + W_8^2 + W_8^4 = 1+4(0.7071-j0.7071) - j + 1$

$= 2.8284 - j2.8284$

@K=2, $X(2) = 1+4W_8^2 + W_8^4 + W_8^6 = 1+4(-j) - 1 + 1 = 1-j4$

@K=3, $X(3) = 1+4W_8^3 + W_8^6 + W_8^{12} = 1+4(-0.7071-j0.7071) + j - 1$

$= -2.8284 - j1.2284$

@K=4, $X(4) = 1+4W_8^4 + W_8^8 + W_8^{16} = 1+4(-1) + 1 + 1 = -1$

@K=5, $X(5) = 1+4W_8^5 + W_8^{10} + W_8^{20} = 1+4(-0.7071+j0.7071) - j - 1$

$= -2.8284 + j1.2284$

@K=6, $X(6) = 1+4W_8^8 + W_8^{12} + W_8^{24} = 1+4(j) - 1 + 1 = 1+j4$

@K=7, $X(7) = 1+4W_8^7 + W_8^{14} + W_8^{28} = 1+4(0.7071+j0.7071) + j - 1$

$= 2.8284 + j2.8284$

$\therefore X(K) = \{7, 2.8284 - j2.8284, 1-j4, -2.8284 - j1.2284, -1, -2.8284 + j1.2284, 1+j4, 2.8284 + j2.8284\}$

2) Determine DFT of $x(n) = \{1, 0, 1, 0, 1, 0, 1, 0\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here N = 8, $X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn}$

$W_8^0 = 1$	$W_8^4 = -1$
$W_8^1 = 0.7071 - j0.7071$	$W_8^5 = -0.7071 + j0.7071$
$W_8^2 = -j$	$W_8^6 = j$
$W_8^3 = -0.7071 - j0.7071$	$W_8^7 = 0.7071 + j0.7071$

$X(K) = 1 + W_8^{2K} + W_8^{4K} + W_8^{6K} \longrightarrow (1)$

@K=0, $X(0) = 1+W_8^0 + W_8^0 + W_8^0 = 1+1+1+1 = 4$

@K=1, $X(1) = 1+W_8^2 + W_8^4 + W_8^6 = 1-j-1+j = 0$

@K=2, $X(2) = 1+W_8^4 + W_8^8 + W_8^{12} = 1-1+1-1 = 0$

@K=3, $X(3) = 1+W_8^6 + W_8^{12} + W_8^{18} = 1+j-1-j = 0$

@K=4, $X(4) = 1+W_8^8 + W_8^{16} + W_8^{24} = 1+1+1+1 = 4$

Scanned with CamScanner

@K=5, $X(5) = 1 + W_8^{10} + W_8^{20} + W_8^{30} = 1 - j - 1 + j = \underline{0}$

@K=6, $X(6) = 1 + W_8^{12} + W_8^{24} + W_8^{36} = 1 - 1 + 1 - 1 = \underline{0}$

@K=7, $X(7) = 1 + W_8^{14} + W_8^{28} + W_8^{42} = 1 + j - 1 - j = \underline{0}$

$\therefore X(K) = \{4, 0, 0, 0, 4, 0, 0, 0\}$

22) Find the DFT of the sequence $x(n) = \begin{cases} 1 ; 0 \le n \le L \\ 0 ; \text{otherwise} \end{cases}$ for N=4. plot magnitude and phase of X(K).

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$W_4^0 = 1$	$W_4^2 = -1$
$W_4^1 = -j$	$W_4^3 = j$

Here $L = N = 4$, $x(n) = \{1, 1, 1, 1\}$

$$X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn} = 1 + W_4^K + W_4^{2K} + W_4^{3K} \longrightarrow (1)$$

@K=0, $X(0) = 1 + W_4^0 + W_4^0 + W_4^0 = 1 + 1 + 1 + 1 = \underline{4}$

@K=1, $X(1) = 1 + W_4^1 + W_4^2 + W_4^3 = 1 - j - 1 + j = \underline{0}$

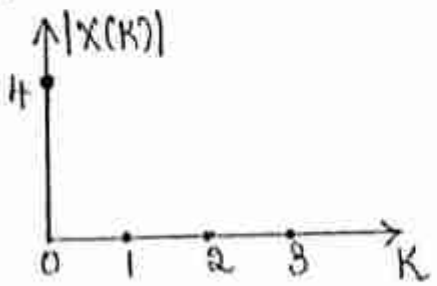
@K=2, $X(2) = 1 + W_4^2 + W_4^4 + W_4^6 = 1 - 1 + 1 - 1 = \underline{0}$

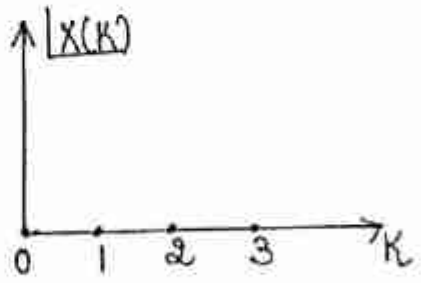
@K=3, $X(3) = 1 + W_4^3 + W_4^6 + W_4^9 = 1 + j - 1 - j = \underline{0}$

$\therefore X(K) = \{4, 0, 0, 0\}$

magnitude, $|X(K)| = \{4, 0, 0, 0\}$

phase, $\underline{|X(K)|} = \{0, 0, 0, 0\}$





Scanned with CamScanner

23) Determine DFT of the sequence $x(n) = \begin{cases} 1/3 & ; 0 \le n \le 2 \\ 0 & ; \text{otherwise} \end{cases}$

plot magnitude and phase spectrum for each case:
a) N = 4 b) N = 6 c) N = 8

Sol^n:- Given $x(n) = \frac{1}{3}$; $0 \le n \le 2$

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$; $0 \le K \le N-1$

a) When N = 4,

$x(n) = \{1/3, 1/3, 1/3, 0\}$ ← Zero-padding to increase the length to '4'.

$\therefore X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn} = \frac{1}{3} + \frac{1}{3} W_4^{K} + \frac{1}{3} W_4^{2K} \longrightarrow (1)$

@K=0, $X(0) = \frac{1}{3} + \frac{1}{3} W_4^0 + \frac{1}{3} W_4^0 = \frac{1}{3} + \frac{1}{3} + \frac{1}{3} = \underline{1}$

@K=1, $X(1) = \frac{1}{3} + \frac{1}{3} W_4^1 + \frac{1}{3} W_4^2 = \frac{1}{3} + \frac{1}{3}(-j) + \frac{1}{3}(-1) = \underline{-j\left(\frac{1}{3}\right)}$

@K=2, $X(2) = \frac{1}{3} + \frac{1}{3} W_4^2 + \frac{1}{3} W_4^4 = \frac{1}{3} + \frac{1}{3}(-1) + \frac{1}{3}(1) = \underline{\frac{1}{3}}$

@K=3, $X(3) = \frac{1}{3} + \frac{1}{3} W_4^3 + \frac{1}{3} W_4^6 = \frac{1}{3} + \frac{1}{3}(j) + \frac{1}{3}(-1) = \underline{j\left(\frac{1}{3}\right)}$

$\therefore X(K) = \{1, -j(1/3), 1/3, j(1/3)\}$

magnitude, $|X(K)| = \{1, 1/3, 1/3, 1/3\}$

phase, $\underline{/X(K)} = \{0, -\pi/2, 0, \pi/2\}$

b) When N = 6,

$x(n) = \{1/3, 1/3, 1/3, 0, 0, 0\}$ ← Zero padding

$X(K) = \sum_{n=0}^{5} x(n) W_6^{Kn} = \frac{1}{3} + \frac{1}{3} W_6^{K} + \frac{1}{3} W_6^{2K} = \frac{1}{3}\left[1 + W_6^{K} + W_6^{2K}\right] \longrightarrow (1)$

Scanned with CamScanner

@K=0, $X(0) = \frac{1}{3}(1+W_6^0+W_6^0) = \frac{1}{3}(1+1+1) = 1$

@K=1, $X(1) = \frac{1}{3}(1+W_6^1+W_6^2) = \frac{1}{3}[1+0.5-j0.866-0.5-j0.866]$

$= 0.333 - j0.5773$

@K=2, $X(2) = \frac{1}{3}(1+W_6^2+W_6^4) = \frac{1}{3}[1-0.5-j0.866-0.5+j0.866]$

$= 0$

@K=3, $X(3) = \frac{1}{3}(1+W_6^3+W_6^6) = \frac{1}{3}[1-1+1] = 1/3$

@K=4, $X(4) = \frac{1}{3}(1+W_6^4+W_6^8) = \frac{1}{3}[1-0.5+j0.866-0.5-j0.866] = 0$

@K=5, $X(5) = \frac{1}{3}(1+W_6^5+W_6^{10}) = \frac{1}{3}[1+0.5+j0.866-0.5+j0.866]$

$= 0.333+j0.5773$

$\therefore X(K) = \{1, 0.333-j0.5773, 0, 1/3, 0, 0.333+j0.5773\}$

$|X(K)| = \{1, 0.66, 0, 1/3, 0, 0.66\}$

$\angle X(K) = \{0, -60°, 0, 0, 0, 60°\}$

c) When N=8,

$X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn} = \frac{1}{3} + \frac{1}{3}W_8^K + \frac{1}{3}W_8^{2K} = \frac{1}{3}(1+W_8^K+W_8^{2K}) \longrightarrow (1)$

@K=0, $X(0) = \frac{1}{3}(1+W_8^0+W_8^0) = \frac{1}{3}(1+1+1) = 1$

@K=1, $X(1) = \frac{1}{3}(1+W_8^1+W_8^2) = \frac{1}{3}(1+0.7071-j0.7071-j) = 0.569-j0.569$

@K=2, $X(2) = \frac{1}{3}(1+W_8^2+W_8^4) = \frac{1}{3}[1-j-1] = -j\frac{1}{3}$

@K=3, $X(3) = \frac{1}{3}(1+W_8^3+W_8^6) = \frac{1}{3}[1-0.7071-j0.7071+j] = 0.0976+j0.098$

@K=4, $X(4) = \frac{1}{3}(1+W_8^4+W_8^8) = \frac{1}{3}[1-1+1] = 1/3$

@K=5, $X(5) = \frac{1}{3}(1+W_8^5+W_8^{10}) = \frac{1}{3}[1-0.7071+j0.7071-j] = 0.0976-j0.098$

Scanned with CamScanner

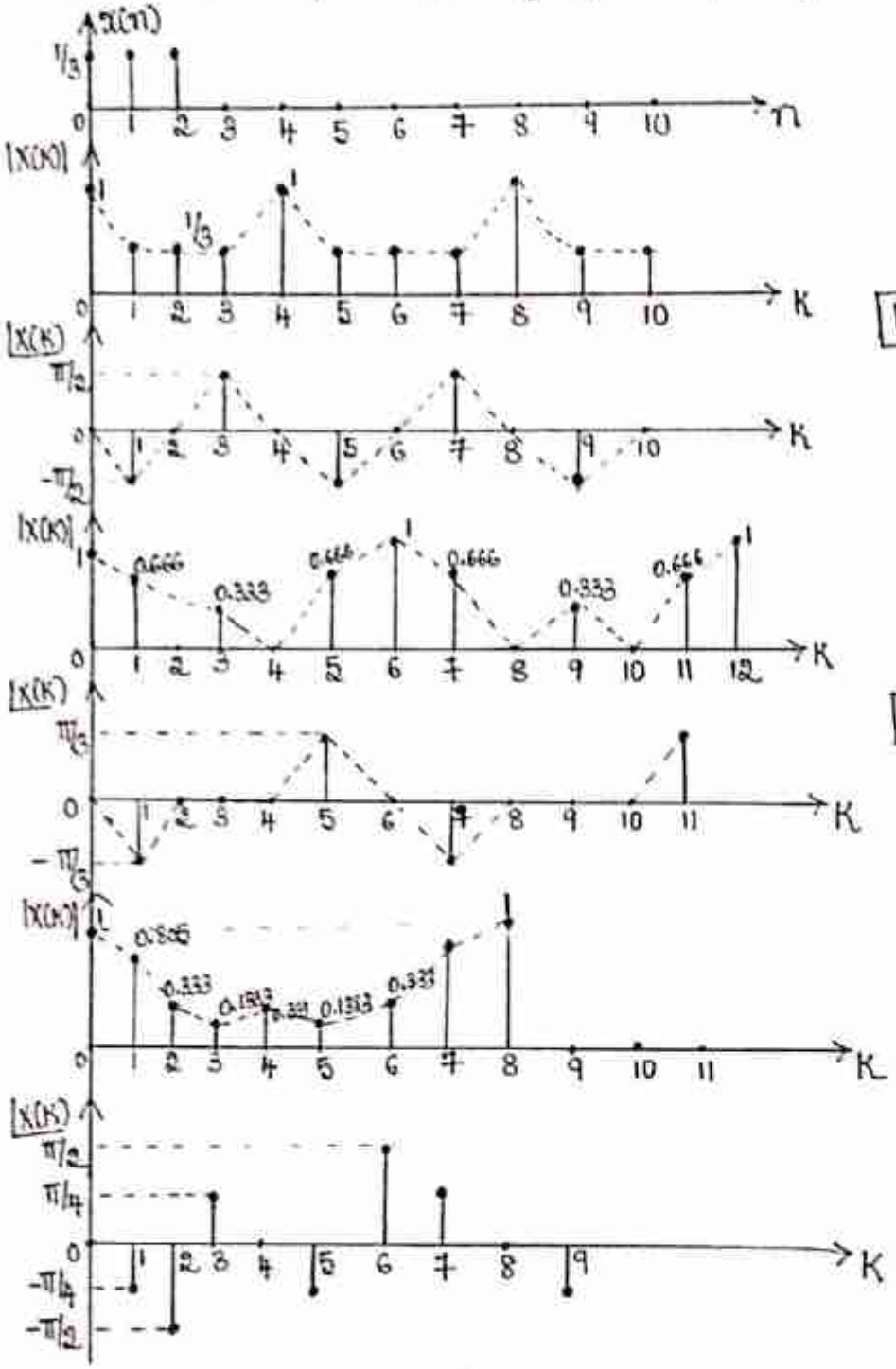
@K=6, $X(6) = \frac{1}{3}(1+W_8^6+W_8^{12}) = \frac{1}{3}[1+j-1] = \underline{j(1/3)}$

@K=7, $X(7) = \frac{1}{3}(1+W_8^7+W_8^{14}) = \frac{1}{3}[1+0.7071+j0.7071-1] = \underline{0.569+j0.569}$

$X(K) = \{1, 0.569-j0.569, -j1/3, 0.0976+j0.098, 1/3, +j1/3, 0.569+j0.569\}$

$|X(K)| = \{1, 0.805, 0.333, 0.1388, 0.333, 0.1888, 0.333, 0.805\}$

$\angle X(K) = \{0, -\pi/4, -\pi/2, \pi/4, 0, -\pi/4, \pi/2, \pi/4\}$



Scanned with CamScanner

Conclusion :-

From the zero points, it is difficult to extrapolate. Entire frequency spectrum i.e., resolution of the spectrum is very poor. In order to increase the resolution, compute DFT for more points. Therefore, the zero padding gives a high density spectrum and provides a better display.

"The procedure of adding zeros to discrete-time sequence $x(n)$ to increase the length of $x(n)$ is called as "Zero padding". Zero padding will not give any additional information about $X(w)$ of $x(n)$. However, padding $(N-L)$ zeros in sequence $x(n)$. N-point DFT length is increased for better display of $X(w)$ and frequency spacing is decreased.

2) Find the DFT of a sequence $x(n) = \begin{cases} 1 & ; 0 \le n \le 2 \\ 0 & ; \text{else} \end{cases}$ for

i) $N=4$ ii) $N=8$. plot $|X(K)|$ and $\angle X(K)$.

Soln :-

i) For $N=4$, $x(n) = \{1, 1, 1, 0\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} = \sum_{n=0}^{3} x(n) W_4^{Kn} = 1 + W_4^K + W_4^{2K} \longrightarrow (1)$$

@K=0, $X(0) = 1 + W_4^0 + W_4^0 = 1+1+1 = 3$

@K=1, $X(1) = 1 + W_4^1 + W_4^2 = 1 - j + 1 = -j$

@K=2, $X(2) = 1 + W_4^2 + W_4^4 = 1 - 1 + 1 = 1$

@K=3, $X(3) = 1 + W_4^3 + W_4^6 = 1 + j - 1 = j$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

$\therefore X(K) = \{3, -j, 1, j\}$

$|X(K)| = \{3, 1, 1, 1\}$

$\angle X(K) = \{0, -\pi/2, 0, \pi/2\}$

Scanned with CamScanner

ii) For $N=8$, $x(n)=\{1,1,1,0,0,0,0,0\}$

$$X(K)=\sum_{n=0}^{N-1} x(n)W_N^{Kn}=\sum_{n=0}^{7} x(n)W_8^{Kn}=1+W_8^{K}+W_8^{2K} \longrightarrow (1)$$

@$K=0$, $X(0)=1+W_8^0+W_8^0=1+1+1=\underline{3}$

@$K=1$, $X(1)=1+W_8^1+W_8^2=1+0.7071-j0.7071-j=\underline{1.7071-j1.7071}$

@$K=2$, $X(2)=1+W_8^2+W_8^4=1-j-1=\underline{-j}$

@$K=3$, $X(3)=1+W_8^3+W_8^6=1-0.7071-j0.7071+j=\underline{0.2929+j0.2929}$

@$K=4$, $X(4)=1+W_8^4+W_8^8=1-1+1=\underline{1}$

@$K=5$, $X(5)=1+W_8^5+W_8^{10}=1-0.7071+j0.7071-j=\underline{0.2929-j0.2929}$

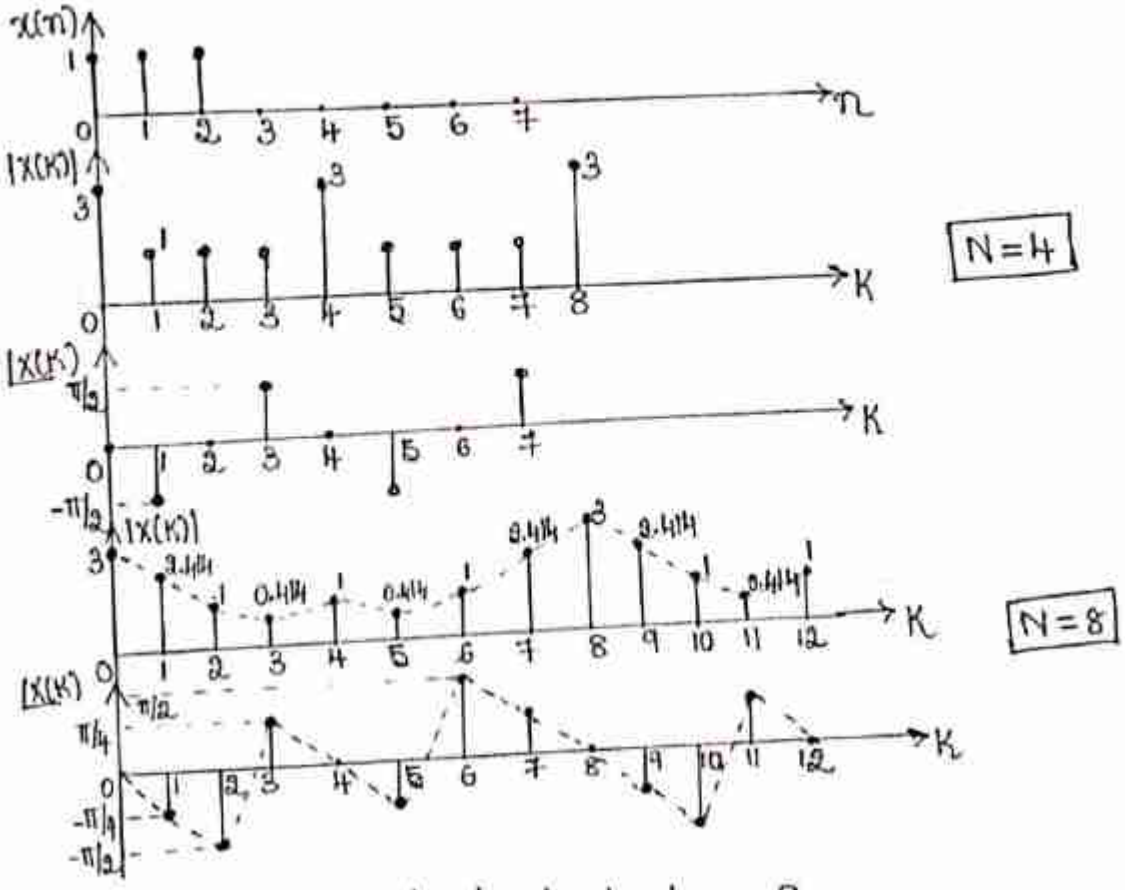
@$K=6$, $X(6)=1+W_8^6+W_8^{12}=1+j-1=\underline{j}$

@$K=7$, $X(7)=1+W_8^7+W_8^{14}=1+0.7071+j0.7071+j=\underline{1.7071+j1.7071}$

$X(K)=\{3, 1.7071-j1.7071, -j, 0.2929+j0.2929, 1, 0.2929-j0.2929, j, 1.7071+j1.7071\}$

$|X(K)|=\{3, 2.4142, 1, 0.4142, 1, 0.4142, 1, 2.4142\}$

$\angle X(K)=\{0, -\pi/4, -\pi/2, \pi/4, 0, -\pi/4, \pi/2, \pi/4\}$



Scanned with CamScanner

25) Determine the 8-point DFT of the sequence $x(n)=\{8,8,8,8,1,1,0,0\}$. Sketch its magnitude and phase spectra.

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=8$, $X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn} = 8 + 8W_8^K + 8W_8^{2K} + 8W_8^{3K} + W_8^{4K} + W_8^{5K} \longrightarrow (1)$

@$K=0$, $X(0) = 8 + 8W_8^0 + 8W_8^0 + 8W_8^0 + W_8^0 + W_8^0 = 8+8+8+8+1+1 = \underline{34}$

@$K=1$, $X(1) = 8 + 8W_8^1 + 8W_8^2 + 8W_8^3 + W_8^4 + W_8^5$

$= 8 + 8(0.7071 - j0.7071) + 8(-j) + 8(-0.7071 - j0.7071) - 1 + (-0.7071 + j0.7071) = \underline{6.293 - j18.61}$

@$K=2$, $X(2) = 8 + 8W_8^2 + 8W_8^4 + 8W_8^6 + W_8^8 + W_8^{10}$

$= 8 + 8(-j) + 8(-1) + 8(j) + 1 - j = \underline{1 - j}$

@$K=3$, $X(3) = 8 + 8W_8^3 + 8W_8^6 + 8W_8^9 + W_8^{12} + W_8^{15}$

$= 8 + 8(-0.7071 - j0.7071) + 8(j) + 8(0.7071 - j0.7071) + (-1) + (0.7071 + j0.7071) = \underline{7.71 - j2.61}$

@$K=4$, $X(4) = 8 + 8W_8^4 + 8W_8^8 + 8W_8^{12} + W_8^{16} + W_8^{20}$

$= 8 + 8(-1) + 8(1) + 8(-1) + 1 - 1 = \underline{0}$

@$K=5$, $X(5) = 8 + 8W_8^5 + 8W_8^{10} + 8W_8^{15} + W_8^{20} + W_8^{25}$

$= 8 + 8(-0.7071 + j0.7071) + 8(-j) + 8(0.7071 + j0.7071) + (-1) + (0.7071 - j0.7071) = \underline{7.71 + j2.61}$

@$K=6$, $X(6) = 8 + 8W_8^6 + 8W_8^{12} + 8W_8^{18} + W_8^{24} + W_8^{30} = \underline{1 + j}$

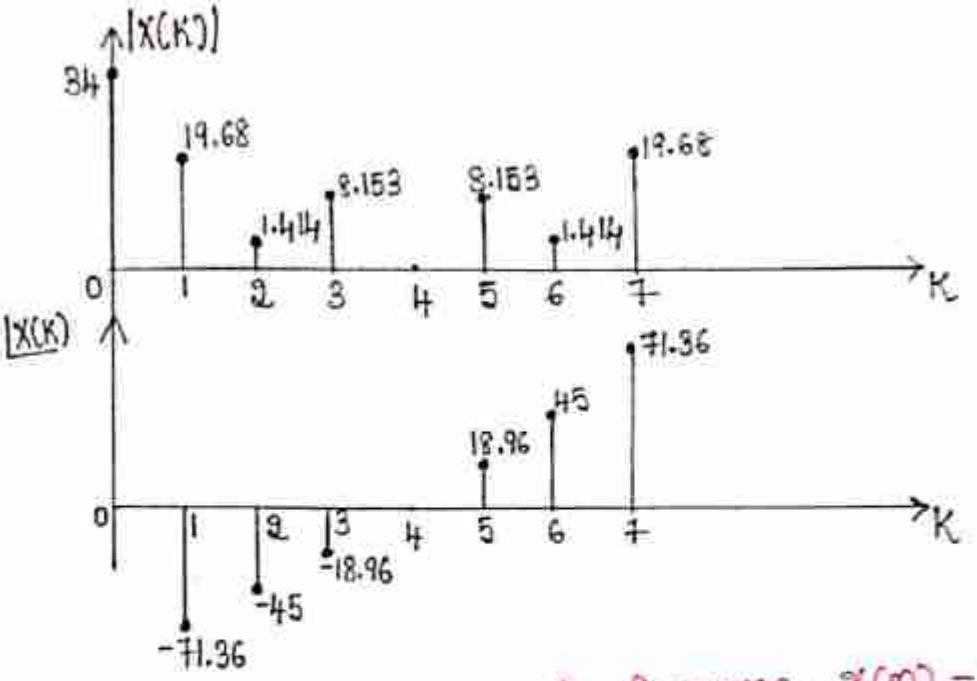
@$K=7$, $X(7) = 8 + 8W_8^7 + 8W_8^{14} + 8W_8^{21} + W_8^{28} + W_8^{35} = \underline{6.293 + j18.61}$

$\therefore X(K) = \{34, 6.293 - j18.61, 1 - j, 7.71 - j2.61, 0, 7.71 + j2.61, 1 + j, 6.293 + j18.61\}$

$|X(K)| = \{34, 19.68, 1.414, 8.153, 0, 8.153, 1.414, 19.68\}$

$\underline{|X(K)|} = \{0, -71.36, -45, -18.96, 0, 18.96, 45, 71.36\}$

Scanned with CamScanner



26) Compute 8-point DFT of the sequence $x(n) = (-1)^{n+1}$; $0 \le n \le 7$.
Also plot the magnitude of DFT.

Sol^n:- $x(n) = \{-1, 1, -1, 1, -1, 1, -1, 1\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

Here $N = 8$,

$$X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn} = -1 + W_8^{K} - W_8^{2K} + W_8^{3K} - W_8^{4K} + W_8^{5K} - W_8^{6K} + W_8^{7K} \longrightarrow (1)$$

@K=0, $X(0) = -1 + W_8^0 - W_8^0 + W_8^0 - W_8^0 + W_8^0 - W_8^0 + W_8^0 = -1 + 1 - 1 + 1 - 1 + 1 - 1 + 1$
$= 0$

@K=1, $X(1) = -1 + W_8^1 - W_8^2 + W_8^3 - W_8^4 + W_8^5 - W_8^6 + W_8^7$
$= -1 + 0.7071 - j0.7071 + j - 0.7071 - j0.7071 + 1 - 0.7071 + j0.7071 - j$
$+ 0.7071 + j0.7071 = 0$

@K=2, $X(2) = -1 + W_8^2 - W_8^4 + W_8^6 - W_8^8 + W_8^{10} - W_8^{12} + W_8^{14}$
$= -1 - j + 1 + j - 1 - j + 1 + j = 0$

@K=3, $X(3) = -1 + W_8^3 - W_8^6 + W_8^9 - W_8^{12} + W_8^{15} - W_8^{18} + W_8^{21}$
$= -1 - 0.7071 - j0.7071 - j + 0.7071 - j0.7071 + 1 + 0.7071 + j0.7071$
$+ j - 0.7071 + j0.7071 = 0$

@K=4, $X(4) = -1 + W_8^4 - W_8^8 + W_8^{12} - W_8^{16} + W_8^{20} - W_8^{24} + W_8^{28}$
$= -1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 = -8$

Scanned with CamScanner

@K=5, $X(5) = -1 + W_8^5 - W_8^{10} + W_8^{15} - W_8^{20} + W_8^{25} - W_8^{30} + W_8^{35}$

$= -1 - 0.707 + j0.707 + j + 0.707 + j0.707 + 1 + 0.707 - j0.707 - j$

$-0.707 - j0.707 = \underline{\underline{0}}$

@K=6, $X(6) = -1 + W_8^6 - W_8^{12} + W_8^{18} - W_8^{24} + W_8^{30} - W_8^{36} + W_8^{42}$

$= -1 + j + 1 - j - 1 + j + 1 - j = \underline{\underline{0}}$

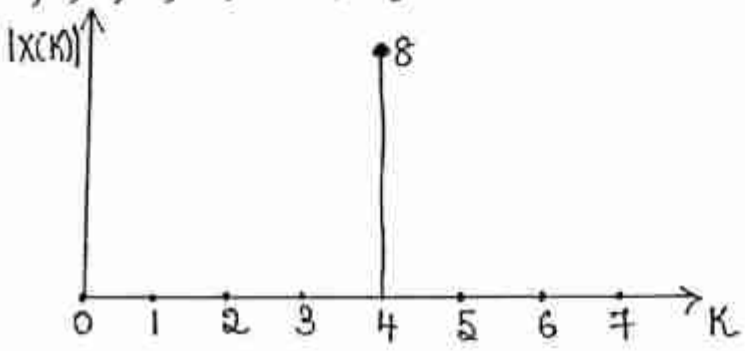
@K=7, $X(7) = -1 + W_8^7 - W_8^{14} + W_8^{21} - W_8^{28} + W_8^{35} - W_8^{42} + W_8^{49}$

$= -1 + 0.707 + j0.707 - j - 0.707 + j0.707 + 1 - 0.707 - j0.707 + j$

$+0.707 - j0.707 = \underline{\underline{0}}$

$X(K) = \{0, 0, 0, 0, -8, 0, 0, 0\}$

$|X(K)| = \{0, 0, 0, 0, 8, 0, 0, 0\}$



Scanned with CamScanner

1) Determine IDFT of the sequence : $X(K) = \{10, -2+j2, -2, -2-j2\}$.

Sol^n:- $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

Here $N=4$, $x(n) = \frac{1}{4}\sum_{K=0}^{3} X(K) W_4^{-Kn}$

$W_4^{-0} = e^{\frac{j2\pi}{4}.0} = 1$
$W_4^{-1} = e^{\frac{j2\pi}{4}.1} = j$
$W_4^{-2} = -1$; $W_4^{-3} = -j$

$= \frac{1}{4}\left[X(0)W_4^{-0} + X(1)W_4^{-n} + X(2)W_4^{-2n} + X(3)W_4^{-3n}\right]$

$x(n) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-n} - 2W_4^{-2n} + (-2-j2)W_4^{-3n}\right] \longrightarrow (1)$

@$n=0$, $x(0) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-0} - 2W_4^{-0} + (-2-j2)W_4^{-0}\right] = \frac{1}{4}[10-2+j2-2-2-j2]$

$= \underline{1}$

@$n=1$, $x(1) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-1} - 2W_4^{-2} + (-2-j2)W_4^{-3}\right]$

$= \frac{1}{4}\left[10 + (-2+j2)(j) - 2(-1) + (-2-j2)(-j)\right] = \underline{2}$

@$n=2$, $x(2) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-2} - 2W_4^{-4} + (-2-j2)W_4^{-6}\right]$

$= \frac{1}{4}\left[10 + (-2+j2)(-1) - 2(1) + (-2-j2)(-1)\right] = \underline{3}$

@$n=3$, $x(3) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-3} - 2W_4^{-6} + (-2-j2)W_4^{-9}\right]$

$= \frac{1}{4}\left[10 + (-2+j2)(-j) + (-2) + (-2-j2)(j)\right] = \underline{4}$

$\therefore x(n) = \{1, 2, 3, 4\}$

2) Determine IDFT of the sequence $X(K) = \{4, -j2, 0, j2\}$

Sol^n:- $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

$W_4^{-0} = 1$; $W_4^{-1} = j$
$W_4^{-2} = -1$; $W_4^{-3} = -j$

Here $N=4$, $x(n) = \frac{1}{4}\sum_{K=0}^{3} X(K) W_4^{-Kn}$

$= \frac{1}{4}\left[4 - j2.W_4^{-n} + j2.W_4^{-3n}\right]$

Scanned with CamScanner

@n=0, $x(0) = \frac{1}{4}[4 + (-j2)W_4^{-0} + (j2)W_4^{-0}] = \frac{1}{4}[4 - j2 + j2] = \underline{1}$

@n=1, $x(1) = \frac{1}{4}[4 + (-j2)W_4^{-1} + (j2)W_4^{-3}] = \frac{1}{4}[4 - j2(j) + j2(-j)] = \underline{2}$

@n=2, $x(2) = \frac{1}{4}[4 - j2W_4^{-2} + j2W_4^{-6}] = \frac{1}{4}[4 - j2(-1) + j2(-1)] = \underline{1}$

@n=3, $x(3) = \frac{1}{4}[4 - j2(W_4^{-3}) + j2W_4^{-9}] = \frac{1}{4}[4 - j2(-j) + j2(j)] = \underline{0}$

$\therefore x(n) = \{1, 2, 1, 0\}$

3) Determine IDFT of the sequence $X(K) = \{2, 1+j, 0, 1-j\}$

Sol^n:- $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K)W_N^{-Kn}$

$W_4^0 = 1$; $W_4^{-1} = j$

$W_4^{-2} = -1$; $W_4^{-3} = -j$

Here N=4, $x(n) = \frac{1}{4}\sum_{K=0}^{3} X(K)W_4^{-Kn}$

$x(n) = \frac{1}{4}[2 + (1+j)W_4^{-n} + (1-j)W_4^{-3n}] \longrightarrow (1)$

@n=0, $x(0) = \frac{1}{4}[2 + (1+j)W_4^{-0} + (1-j)W_4^{-0}] = \frac{1}{4}[2+1+j+1-j] = \underline{1}$

@n=1, $x(1) = \frac{1}{4}[2 + (1+j)W_4^{-1} + (1-j)W_4^{-3}] = \frac{1}{4}[2 + (1+j)(j) + (1-j)(-j)]$

$x(1) = 0$

@n=2, $x(2) = \frac{1}{4}[2 + (1+j)W_4^{-2} + (1-j)W_4^{-6}] = \frac{1}{4}[2 + (1+j)(-1) + (1-j)(-1)]$

$x(2) = 0$

@n=3, $x(3) = \frac{1}{4}[2 + (1+j)W_4^{-3} + (1-j)W_4^{-9}] = \frac{1}{4}[2 + (1+j)(-j) + (1-j)(j)]$

$x(3) = 1$

$\therefore x(n) = \{1, 0, 0, 1\}$

Scanned with CamScanner

4) Find 8-point IDFT for the sequence:

$X(K) = \{5, 0, 1-j, 0, 1, 0, 1+j, 0\}$.

Soln:- $x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

Here $N=8$, $x(n) = \frac{1}{8} \sum_{K=0}^{7} X(K) W_8^{-Kn}$

$$x(n) = \frac{1}{8}\left[5 + (1-j)W_8^{-2n} + W_8^{-4n} + (1+j)W_8^{-6n}\right] \longrightarrow (1)$$

@ $n=0$, $x(0) = \frac{1}{8}[5+1-j+1+1+j] = \underline{1}$

@ $n=1$, $x(1) = \frac{1}{8}\left[5 + (1-j)W_8^{-2} + W_8^{-4} + (1+j)W_8^{-6}\right]$

$= \frac{1}{8}[5 + (1-j)(j) + (-1) + (1+j)(-j)] = \underline{3/4}$

@ $n=2$, $x(2) = \frac{1}{8}\left[5 + (1-j)W_8^{-4} + W_8^{-8} + (1+j)W_8^{-12}\right]$

$= \frac{1}{8}[5 + (1-j)(-1) + 1 + (1+j)(-1)] = \underline{1/2}$

@ $n=3$, $x(3) = \frac{1}{8}\left[5 + (1-j)W_8^{-6} + W_8^{-12} + (1+j)W_8^{-18}\right]$

$= \frac{1}{8}[5 + (1-j)(-j) + (-1) + (1+j)(j)] = \underline{1/4}$

@ $n=4$, $x(4) = \frac{1}{8}\left[5 + (1-j)W_8^{-8} + W_8^{-16} + (1+j)W_8^{-24}\right]$

$= \frac{1}{8}[5 + (1-j)(1) + 1 + (1+j)(1)] = \underline{1}$

@ $n=5$, $x(5) = \frac{1}{8}\left[5 + (1-j)W_8^{-10} + W_8^{-20} + (1+j)W_8^{-30}\right]$

$= \frac{1}{8}[5 + (1-j)(j) + (-1) + (1+j)(-j)] = \underline{3/4}$

@ $n=6$, $x(6) = \frac{1}{8}\left[5 + (1-j)W_8^{-12} + W_8^{-24} + (1+j)W_8^{-36}\right]$

$= \frac{1}{8}[5 + (1-j)(-1) + 1 + (1+j)(-1)] = \underline{1/2}$

Scanned with CamScanner

@n=7,

$$x(n) = \frac{1}{8}\left[5 + (1-j)W_8^{-14} + W_8^{-28} + (1+j)W_8^{-42}\right]$$

$$= \frac{1}{8}\left[5 + (1-j)(-j) + (-1) + (1+j)(j)\right] = \underline{1/4}$$

$\therefore x(n) = \{1, 3/4, 1/2, 1/4, 1, 3/4, 1/2, 1/4\}$

5) Find the IDFT of $X(K) = \{255, 48.63+j166.05, -51+j102, -78.63+j46.05, -85, -78.63-j46.05, -51-j102, 48.63-j166.05\}$

Sol^n:- $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

Here N=8,

$$x(n) = \frac{1}{8}\sum_{K=0}^{7} X(K) W_8^{-Kn}$$

$$= \frac{1}{8}\Big[255 + (48.63+j166.05)W_8^{-n} + (-51+j102)W_8^{-2n} + (-78.63+j46.05)W_8^{-3n} + (-85)W_8^{-4n} + (-78.63-j46.05)W_8^{-5n} + (-51-j102)W_8^{-6n} + (48.63-j166.05)W_8^{-7n}\Big] \longrightarrow (1)$$

(W_8^{-n}: 45°; W_8^{-2n}: 90°; W_8^{-3n}: 135°; W_8^{-4n}: 180°; W_8^{-5n}: 225°; W_8^{-6n}: 270°; W_8^{-7n}: 315°)

@n=0, $x(0) = \frac{1}{8}[252.63 + j268.05 - 242.26 - 2.37 - j268.05] = \underline{1}$

@n=1, $x(1) = \frac{1}{8}[69.97 + j100.8 + 131.075 - 185.03 - j100.8] = \underline{2}$

@n=2, $x(2) = \frac{1}{8}[139.95 - j53.37 + 7.1 - 115.05 + j53.37] = \underline{4}$

@n=3, $x(3) = \frac{1}{8}[205.19 - j32.03 - 91.32 - 49.8 + j32.03] = \underline{8}$

@n=4, $x(4) = \frac{1}{8}[155.37 - j64.05 + 72.26 - 99.63 + j64.05] = \underline{16}$

@n=5, $x(5) = \frac{1}{8}[236.03 - j202.8 - 138.925 - 18.97 + j202.8] = \underline{22}$

@n=6, $x(6) = \frac{1}{8}[472.05 - j150.63 - 177.1 + 217.05 + j150.63] = \underline{64}$

@n=7, $x(7) = \frac{1}{8}[508.8 + j134.03 + 261.32 + 253.8 - j134.03] = \underline{128}$

$\therefore x(n) = \{1, 2, 4, 8, 16, 22, 64, 128\}$

Scanned with CamScanner

1) Consider, a signal of length equal to 4 defined by $x(n) = \{1, 2, 3, 1\}$
a) Compute 4-point DFT by solving explicitly 4-by-4 system of linear equations defined by the inverse DFT formula.
b) Verify the result of part (a) by finding the $X(K)$ using the defined equation by DFT.

Soln:-

a) IDFT : $x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

$$\sum_{K=0}^{N-1} X(K) W_N^{-Kn} = N\, x(n)$$

Here $N=4$, $\sum_{K=0}^{3} X(K) W_4^{-Kn} = 4\, x(n)$; $n = 0, 1, 2, 3$

@ $n=0$, $X(0)W_4^{-0} + X(1)W_4^{-0} + X(2)W_4^{-0} + X(3)W_4^{-0} = 4x(0) = 4 \times 1 = \underline{4}$

@ $n=1$, $X(0)W_4^{-0} + X(1)W_4^{-1} + X(2)W_4^{-2} + X(3)W_4^{-3} = 4x(1) = 4 \times 2 = \underline{8}$

@ $n=2$, $X(0)W_4^{-0} + X(1)W_4^{-2} + X(2)W_4^{-4} + X(3)W_4^{-6} = 4x(2) = 4 \times 3 = \underline{12}$

@ $n=3$, $X(0)W_4^{-0} + X(1)W_4^{-3} + X(2)W_4^{-6} + X(3)W_4^{-9} = 4x(3) = 4 \times 1 = \underline{4}$

The above equations are linear equations,

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} \begin{bmatrix} W_4^{-0} & W_4^{-0} & W_4^{-0} & W_4^{-0} \\ W_4^{-0} & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ W_4^{-0} & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ W_4^{-0} & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix} = \begin{bmatrix} 4 \\ 8 \\ 12 \\ 4 \end{bmatrix}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^{-0} & W_4^{-0} & W_4^{-0} & W_4^{-0} \\ W_4^{-0} & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ W_4^{-0} & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ W_4^{-0} & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix}^{-1} \begin{bmatrix} 4 \\ 8 \\ 12 \\ 4 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}^{-1} \begin{bmatrix} 4 \\ 8 \\ 12 \\ 4 \end{bmatrix}$$

Scanned with CamScanner

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}\begin{bmatrix} 4 \\ 8 \\ 12 \\ 4 \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} 4+8+12+4 \\ 4-j8-12+j4 \\ 4-8+12-4 \\ 4+j8-12-j4 \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} 28 \\ -8-j4 \\ 4 \\ -8+j4 \end{bmatrix} = \begin{bmatrix} 7 \\ -2-j1 \\ 1 \\ -2+j1 \end{bmatrix}$$

$\therefore X(K) = \{7, -2-j1, 1, -2+j1\}$

b) DFT : $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = x(0)W_4^0 + x(1)W_4^K + x(2)W_4^{2K} + x(3)W_4^{3K}$$

$$X(K) = 1 + 2W_4^K + 3W_4^{2K} + W_4^{3K} \longrightarrow (1)$$

@K=0, $X(0) = 1 + 2W_4^0 + 3W_4^0 + W_4^0 = 1+2+3+1 = 7$

@K=1, $X(1) = 1 + 2W_4^1 + 3W_4^2 + W_4^3 = 1+2(-j)+3(-1)+j = -2-j1$

@K=2, $X(2) = 1 + 2W_4^2 + 3W_4^4 + W_4^6 = 1+2(-1)+3(1)+(-1) = 1$

@K=3, $X(3) = 1 + 2W_4^3 + 3W_4^6 + W_4^9 = 1+2(j)+3(-1)-j = -2+j1$

$\therefore X(K) = \{7, -2-j1, 1, -2+j1\}$

Scanned with CamScanner

2) Consider two periodic sequences $x(n)$ and $y(n)$ $x(n)$ has a period 'N' and $y(n)$ has a period 'M'. The sequence $w(n)$ is defined as $w(n) = x(n) + y(n)$.

a) Show that $w(n)$ is periodic with period 'MN'.

b) Also show that $W(K)$ represents 'MN' point DFT of an 'MN' point sequence $w(n)$

Sol^n:-

a) To prove that $w(n) = w(n+MN)$:

The sequence $w(n)$ is periodic with period MN. Then,

$w(n+MN) = x(n+MN) + y(n+MN)$

$x(n) = x(n+N) = x(n+2N) = \ldots\ldots = x(n+MN)$ i.e., for all integer multiples of 'N'.

$y(n) = y(n+N) = y(n+2N) = \ldots\ldots = y(n+MN)$ i.e., for all integer multiples of 'M'.

$\therefore w(n+MN) = x(n) + y(n)$

$= w(n)$

Thus, $w(n)$ is periodic with period 'MN'.

b) DFT of $w(n)$:

The N-point DFT of the sequence of length 'N' is given by,

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \; ; \; 0 \le K \le (N-1).$$

Hence, DFT of $w(n)$ will be of length 'MN' which will be,

$$W(K) = \sum_{n=0}^{MN-1} w(n) W_N^{Kn} \; ; \; 0 \le K \le MN-1$$

Thus, $W(K)$ is MN-point DFT of $w(n)$.

Scanned with CamScanner

3) a) An analog signal is sampled at 10KHz and the DFT of 512 samples is computed. Determine the frequency spacing between spectral samples of the DFT.
b) Determine the frequency spacing if 1024-point DFT is calculated.

Sol^n:-

a) Given $N = 512$, $F_S = 10KHz$

Frequency resolution $\Delta F = \frac{1}{N} \cdot F_S$

$$= \frac{1}{512} \times 10{,}000$$

$$\langle \Delta F = 19.53125\ Hz \rangle$$

This means the frequency spacing between the samples of DFT will be 19.53125 Hz.

b) For 1024-point DFT, $N = 1024$ and the frequency spacing will be:

$$\Delta F = \frac{1}{N} F_S = \frac{1}{1024} \times 10{,}000 = 9.7656\ Hz$$

4) Find the Z-transform of the sequence $x(n) = \{0.5, 0, 0.5, 0\}$ (↑ at 0.5) using Z-transform, find its DFT.

Sol^n:- Here $N = 4$

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

$$X(z) = \sum_{n=0}^{3} x(n) z^{-n}$$

$$\langle X(z) = 0.5 + 0.5 z^{-2} \rangle$$

$$X(K) = X(z)\Big|_{z = e^{\frac{j2\pi}{N}K}} = e^{\frac{j2\pi}{4}K}$$

$$X(K) = 0.5 + 0.5\, e^{\frac{-j2\pi}{4}K(-2)} \quad ; 0 \le K \le 3$$

$$\langle X(K) = \{1, 0, 1, 0\} \rangle$$

Scanned with CamScanner

Module-1:

Anusha.M.N
Asst. professor,
Dept. of ECE,
BGSIT

Discrete Fourier Transform(DFT)

Introduction:

In the digital signal processing, all the signals are discrete. When the analog signals such as speech, music etc are to be processed by the DSP system, then such signals are converted to discrete form by A/D Converters. After processing, the signals are converted to analog form by D/A Converters.

Many applications demand the processing of signals in frequency-domain: frequency content, periodicity, energy and power spectrum etc... can be better analyzed in frequency-domain. Hence, the signals are transformed from time-domain to frequency domain. Such transformations can be obtained with the help of Fourier transform (FT) and Discrete-Fourier transform (DFT).

For the discrete signals, DFT is used. DFT can also be used for linear filtering and Correlation.

Once, the required analysis and processing is performed in frequency-domain, the signals are transformed back in time-domain by inverse Discrete Fourier transform (IDFT).

Frequency-domain Sampling and Reconstruction of discrete-time signals :—

a) Need of frequency-domain Sampling :—

W.K.T the Fourier transform of the signal can be calculated as,

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\Omega n} \longrightarrow (1)$$

Here $x(n)$ is a discrete-time signal and

Ω is the frequency.

Note that 'Ω' is the continuous function from 0 to 2π.

Scanned with CamScanner

This means Eventhough x(n) is discrete, its spectrum X(Ω) is continuous. Such continuous function cannot be evaluated on digital pocessor. on digital processor, only discrete signals can be processed.

b) Frequency - domain sampling :-

To overcome the problem of digital processing, the spectrum X(Ω) is sampled uniformly.

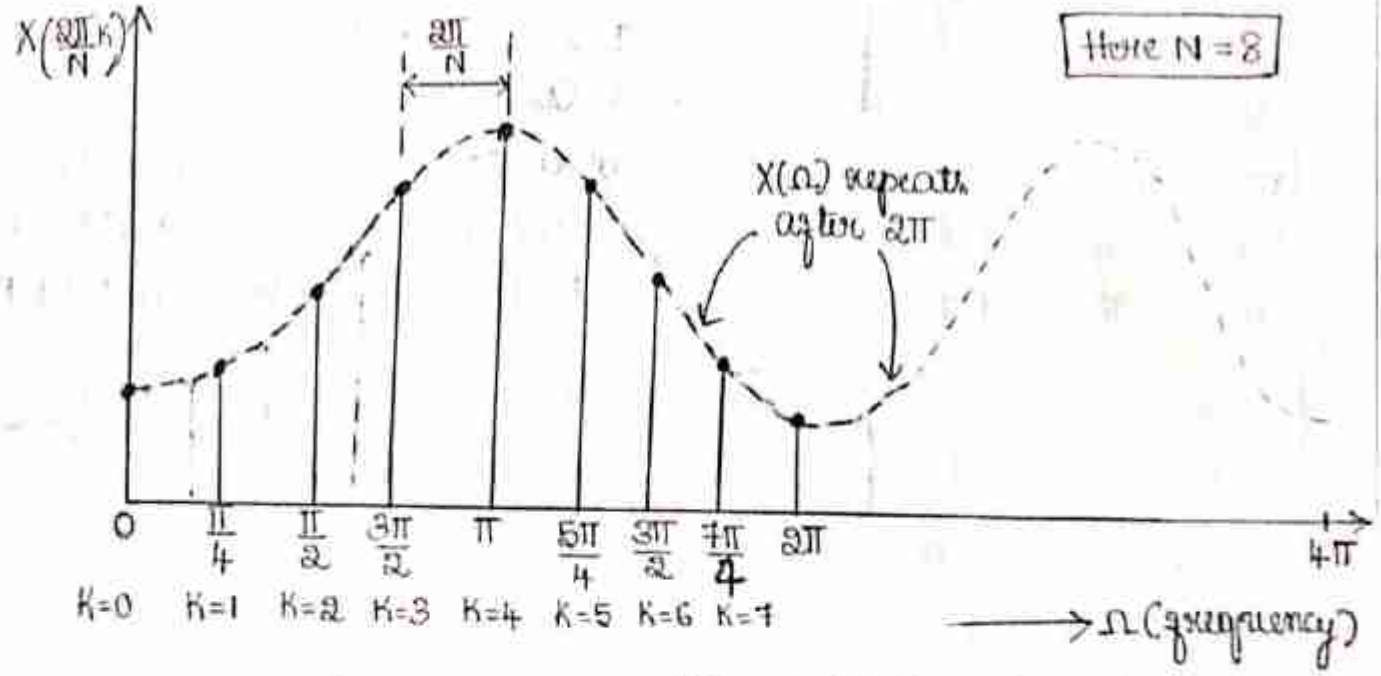
N.K.T X(Ω) is periodic with the period of 2π. Hence, the samples taken from 0 to 2π are only important for our processing.

Let 'N' samples are taken from 0 to 2π. Hence, the spacing between successive samples will be 2π/N.

Therefore in equation (1), let us put $\Omega = \frac{2\pi}{N} K$

i.e., $$X\left(\frac{2\pi}{N}K\right) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\frac{2\pi}{N}Kn} \longrightarrow (2)$$

Here K = 0, 1, 2, ... (N-1)



Thus, 'K' is an index for the samples and X(Ω) is calculated only at discrete values.

Ex:- Consider the spectrum shown in above fig, observe that N=8 samples are taken over the period of 2π. Hence, X(Ω) will be calculated at $\Omega = 0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi, \frac{5\pi}{4}, \frac{3\pi}{2}$ and $\frac{7\pi}{4}$.

Scanned with CamScanner

These samples are addressed by $K = 0, 1, 2, \ldots N$. By putting the value of 'K' in Eq(2), the sampled spectrum can be obtained.

c) To determine minimum value of 'N' :-

In Eq(2), note that 'n' varies from $-\infty$ to ∞.

Let us divide this summation into individual summation containing only 'N' samples of $x(n)$

i.e., $X\left(\frac{2\pi}{N}K\right) = \ldots\ldots + \sum + \sum + \sum + \ldots\ldots$

Each summation is carried out over 'N' samples of $x(n)$.

Therefore, above equation can be written as,

$$X\left(\frac{2\pi}{N}K\right) = \ldots\ldots + \sum_{n=-N}^{-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} + \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} + \sum_{n=N}^{2N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} + \ldots\ldots$$

The above individual summations can be represented as,

$$X\left(\frac{2\pi}{N}K\right) = \sum_{L=-\infty}^{\infty} \sum_{n=LN}^{LN+N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn}$$

Now, let us change the index of inner sum from n to $n-LN$. Hence, its limits will be $n=0$ to $n=N-1$.

$$\text{i.e., } X\left(\frac{2\pi}{N}K\right) = \sum_{L=-\infty}^{\infty} \sum_{n=0}^{N-1} x(n-LN)\, e^{-j\frac{2\pi}{N}K(n-LN)}$$

$$= \sum_{L=-\infty}^{\infty} \sum_{n=0}^{N-1} x(n-LN)\, e^{-j\frac{2\pi}{N}Kn}\, e^{j\frac{2\pi}{N}K\cdot LN}$$

Here, $e^{j2\pi KL} = 1$ always, since 'K' and 'L' are both integers. Hence, the term $2\pi KL$ will be always integer multiple of 2π.

$$\therefore X\left(\frac{2\pi}{N}K\right) = \sum_{L=-\infty}^{\infty} \sum_{n=0}^{N-1} x(n-LN)\, e^{-j\frac{2\pi}{N}Kn}$$

Let us interchange the order of summation,

$$X\left(\frac{2\pi}{N}K\right) = \sum_{n=0}^{N-1} \left[\sum_{L=-\infty}^{\infty} x(n-LN)\right] e^{-j\frac{2\pi}{N}Kn} \longrightarrow (3)$$

$$X\left(\frac{2\pi}{N}K\right) = \sum_{n=0}^{N-1} x_p(n)\, e^{-j\frac{2\pi}{N}Kn} \longrightarrow (4)$$

Here $K = 0, 1, 2, \ldots (N-1)$

Scanned with CamScanner

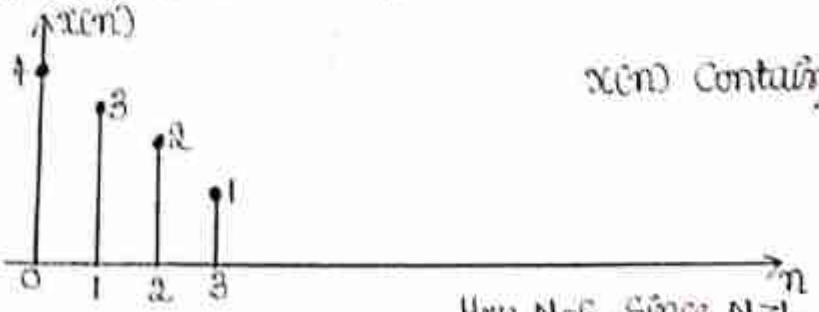
Here $x_p(n) = \sum_{l=-\infty}^{\infty} x(n-lN)$

$= \ldots + x(n+2N) + x(n+N) + x(n) + x(n-N) + x(n-2N) + \ldots$

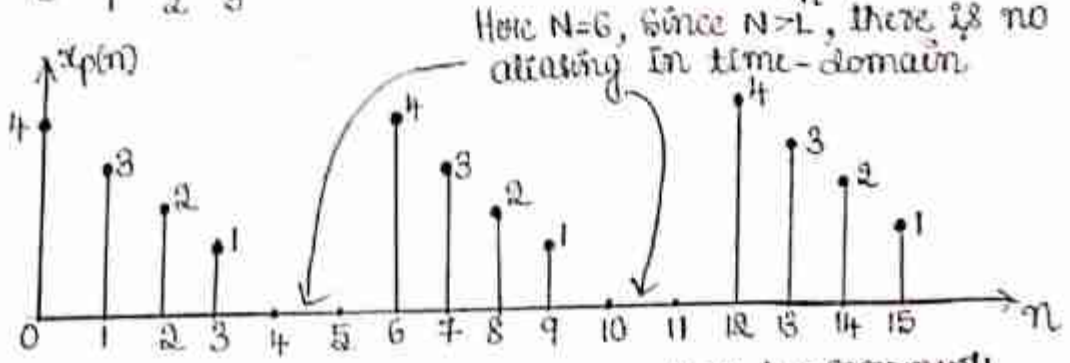
This means $x_p(n)$ is the periodic repetition of $x(n)$ with period of N-samples.

Consider some arbitrary non-periodic signal $x(n)$. It contains 'L' samples. It is shown in fig(a). In this waveforms, L=4 is taken. Note that $x(n)$ is non-periodic.

(a)



(b)



(c)

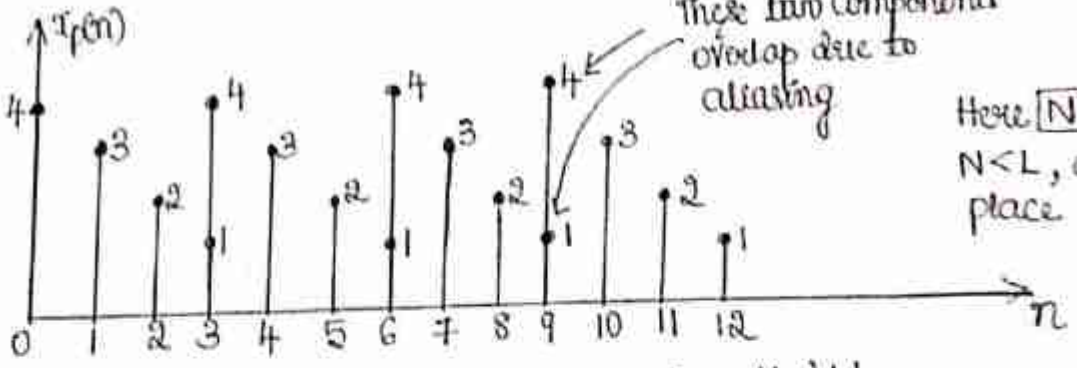


Fig (a) : An original signal $x(n)$ of length 'L'.

(b) : periodic repetition of $x(n)$ with period $N > L$.

(c) : periodic repetition of $x(n)$ with period $N < L$.

Case (i): When $N > L$, no aliasing :

Now let us prepare the signal $x_p(n)$. It is obtained by periodic repetition of $x(n)$. Let the period 'N' be greater than 'L'. For Example, N=6 in fig(b). Observe that sample values at n=4

Scanned with CamScanner

and $n=5$ are zero. This is because the signal repeats at $n=6, 12, \ldots$ etc. Here $N>L$, hence there is no aliasing.

Case(ii):- When $N<L$, Aliasing :

Now Consider the waveform of $x_p(n)$ of fig(c). In this waveform, observe that $N=3$. Since $N<L$, the two samples are overlapping at $n=3,6, \ldots$ etc. This is Called aliasing. Hence, it is possible to obtain $x(n)$ from $x_p(n)$.

Thus, to avoid aliasing in time-domain, the number of samples in the frequency spectrum must be greater than number of samples in time-domain sequence.

i.e., $\boxed{N \geq L}$ $\longrightarrow$ (5)

The above equation shows that we should take atleast 'L' samples over 0 to 2π.

Therefore, the DFT of aperiodic (or) finite-length sequence is given by,

$$\left\langle X(K) = \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} \; ; \; 0 \leq K \leq N-1 \right\rangle$$

(or)

$$\left\langle X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn} \; ; \; 0 \leq K \leq N-1 \right.$$
$$\left. \text{where } W_N^{nK} = e^{-j\frac{2\pi}{N}Kn} \right\rangle$$

Reconstruction of discrete-time signals :—

W.K.T $x_p(n)$ is periodic with period 'N'. It can be expressed by discrete Fourier series as,

$$x_p(n) = \sum_{K=0}^{N-1} C(K)\, e^{j\frac{2\pi}{N}Kn} \; ; \; 0 \leq n \leq N-1 \longrightarrow (1)$$

and the Fourier-series Co-efficients $C(K)$ are given by,

$$C(K) = \frac{1}{N} \sum_{n=0}^{N-1} x_p(n)\, e^{-j\frac{2\pi}{N}Kn} \; ; \; 0 \leq K \leq N-1 \longrightarrow (2)$$

W.K.T $\sum_{n=0}^{N-1} x_p(n)\, e^{-j\frac{2\pi}{N}Kn} = X\left(\frac{2\pi}{N}K\right)$

Scanned with CamScanner

∴ Eq(2) becomes,

$$C(K) = \frac{1}{N} X\left(\frac{2\pi}{N}K\right) \; ; \; 0 \le K \le N-1 \longrightarrow (3)$$

putting this value of C(K) in Eq(1),

$$x_p(n) = \sum_{K=0}^{N-1} \frac{1}{N} X\left(\frac{2\pi}{N}K\right) e^{\frac{j2\pi}{N}Kn} \; ; \; 0 \le n \le N-1 \longrightarrow (4)$$

This equation gives the expression for time-domain sequence $x_p(n)$ from frequency-domain samples $X(\frac{2\pi}{N}K)$. Thus, time-domain sequence can be reconstructed from frequency-domain sampling.

$W_N = e^{\frac{-j2\pi}{N}}$, which is called the "Twiddle factor".

The N-point IDFT is given by,

$$\left\langle x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) W_N^{-nK} \; ; \; 0 \le n \le N-1 \right\rangle$$

DFT as a linear Transformation (Matrix method) :-

W.K.T,

$$\text{DFT} : X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \; ; \; 0 \le K \le N-1$$

$$\text{IDFT} : x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) W_N^{-Kn} \; ; \; 0 \le n \le N-1$$

Twiddle factor : $W_N = e^{-j2\pi/N}$

Computation of each point of DFT requires: N-Complex multiplications and (N-1) Complex additions. Hence, N-point involves: N^2-Complex multiplications and N(N-1) Complex additions.

DFT and IDFT can be viewed as linear transformation on sequences x(n) and X(K) respectively.

Let x_N be a N-point vector of the discrete-time sequence $x(n)$; $0 \le n \le N-1$, X_N be N-point vector of frequency-domain signal X(K) and W_N be (N X N) matrix of Complex Exponentials then,

Scanned with CamScanner

$$x_N = \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix} \quad ; \quad X_N = \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} \quad \text{and} \quad W_N = \begin{bmatrix} W_N^{Kn} & \xrightarrow{n=0,1,\ldots(N-1)} \\ \downarrow & \\ K=0,1,\ldots(N-1) & \end{bmatrix}_{(N\times N)}$$

$$W_N = \begin{bmatrix} W_N^{00} & W_N^{01} & W_N^{02} & \cdots & W_N^{0(N-1)} \\ W_N^{10} & W_N^{11} & W_N^{12} & \cdots & W_N^{1(N-1)} \\ W_N^{20} & W_N^{21} & W_N^{22} & \cdots & W_N^{2(N-1)} \\ \vdots & & & & \vdots \\ W_N^{(N-1)0} & W_N^{(N-1)1} & W_N^{(N-1)2} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix}_{N\times N}$$

$$W_N = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & W_N^2 & \cdots & W_N^{1(N-1)} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & & \\ 1 & W_N^{1(N-1)} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix}_{N\times N}$$

Now, with all the above definitions, the N-point DFT may be expressed in matrix as:

$$\boxed{X_N = W_N x_N}$$

where, W_N is matrix of linear transformation.

Ex: W_4 matrix can be expressed as:

Here $N=4$,

$$\therefore W_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^4 & W_4^6 \\ 1 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix}_{4\times 4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^0 & W_4^2 \\ 1 & W_4^3 & W_4^2 & W_4^1 \end{bmatrix}_{4\times 4}$$

From this, we can conclude that 'W_N' is "Symmetric matrix".

Scanned with CamScanner

W.K.T $X_N = W_N \cdot x_N$

Multiplying both sides by W_N^{-1},

$$W_N^{-1} \cdot X_N = W_N \cdot x_N \cdot W_N^{-1}$$

$$W_N^{-1} \cdot X_N = x_N$$

$\therefore$ IDFT, $x_N = W_N^{-1} \cdot X_N$

But from the IDFT equation, it becomes

$$\text{IDFT}, \; x_N = \frac{1}{N} W_N^{*} \cdot X_N$$

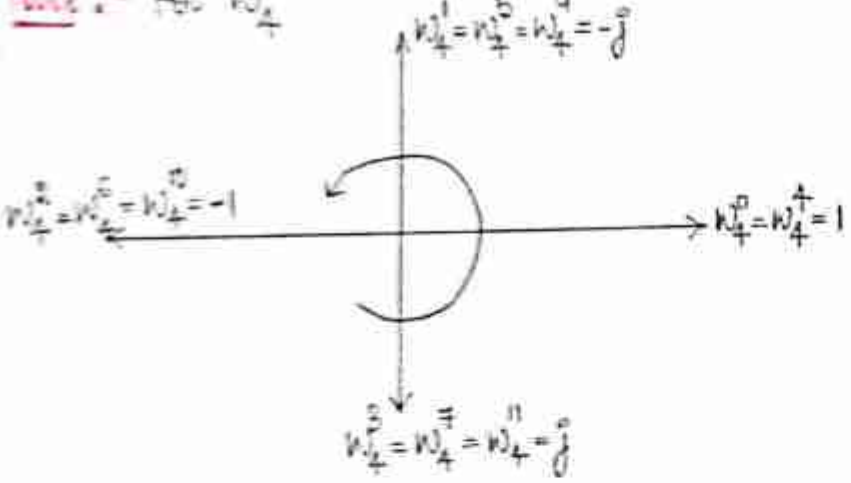
where, W_N^{*} is complex conjugate of matrix W_N.

$\therefore W_N^{-1} = \frac{1}{N} \cdot W_N^{*}$

This implies that $\boxed{W_N \cdot W_N^{*} = N I_N}$

where, $I_N \rightarrow (N \times N)$ Identity matrix.

Note :- For W_4



W_4^0	W_4^1	W_4^2	W_4^3
1	$-j$	-1	$+j$

The Twiddle factor 'W_N' satisfies the following properties :

1) Cyclic / periodicity property :-

W.K.T $W_N = e^{-j\frac{2\pi}{N}}$

prove that $W_N^{N+Kn} = W_N^{Kn}$:

Let $N=4$ and $Kn = 1$

Now, $W_4^{4+1} = W_4^1$

Scanned with CamScanner

$L.H.S = W_4^4 \cdot W_4^1 = e^{\frac{-j2\pi}{4} \cdot 4} \cdot e^{\frac{-j2\pi}{4} \cdot 1} = e^{-j2\pi} \cdot e^{-j\pi/2} = 1 \cdot e^{j\pi/2} = -j$

$R.H.S = W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = e^{\frac{-j\pi}{2}} = -j$

$\therefore \boxed{W_4^5 = W_4^1}$

It satisfies the cyclic property.

2) Symmetry property :-

W.K.T $W_N = e^{\frac{-j2\pi}{N}}$

prove that $W_N^{(N/2+Kn)} = -W_N^{Kn}$:-

Let N = 4 and Kn = 1

Now, $W_4^{(\frac{4}{2}+1)} = -W_4^1$

$W_4^{2+1} = -W_4^1$

$L.H.S = W_4^{2+1} = W_4^2 \cdot W_4^1 = e^{\frac{-j2\pi}{4} \cdot 2} \cdot e^{\frac{-j2\pi}{4} \cdot 1} = e^{-j\pi} \cdot e^{\frac{-j\pi}{2}} = (-1)(-j) = +j$

$R.H.S = -W_4^1 = -e^{\frac{-j2\pi}{4} \cdot 1} = -e^{\frac{-j\pi}{2}} = -(-j) = +j$

$\therefore \boxed{W_4^3 = -W_4^1}$

It satisfies the symmetry property.

Relationship of the DFT to other transforms :-

1) Relationship to Fourier Series Co-efficient of a periodic Sequence :-

A periodic signal $x_p(n)$ with a fundamental period 'N' can be represented in a Fourier series of the form,

$$x_p(n) = \sum_{K=0}^{N-1} C(K) e^{\frac{j2\pi}{N} Kn} \quad ; -\infty < n < \infty \longrightarrow (1)$$

Where C(K) is a Fourier-Series Co-efficient and it is computed by using,

$$C(K) = \frac{1}{N} \sum_{n=0}^{N-1} x_p(n) e^{\frac{-j2\pi}{N} Kn} \quad ; 0 \le K \le N-1$$

Scanned with CamScanner

But we know the IDFT equation,

$$x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K)\, e^{j\frac{2\pi}{N}Kn} \quad ; 0 \le n \le N-1 \longrightarrow (2)$$

on comparing Eq(1) and Eq(2),

$$C(K) = \frac{1}{N} X(K)$$

$\langle X(K) = N\,C(K) \rangle$ provided $x(n) = x_p(n)$; $0 \le n \le (N-1)$.

Hence, except for a scaling factor, the DFS and DFT relations are identical.

2) Relationship between DFT and DTFT :-

The DTFT is given by,

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\Omega n}$$

When this DTFT is sampled at 'N' equally spaced points, we get DFT X(K)

i.e., $X(K) = X(\Omega)\Big|_{\Omega = \frac{2\pi}{N}K} \quad ; 0 \le K \le N-1 \longrightarrow (1)$

$$\therefore X(K) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\frac{2\pi}{N}Kn} \quad ; 0 \le K \le N-1$$

W.K.T X(K) is also DFT of the periodic sequence $x_p(n)$ which repeats with period 'N'.

Then $x_p(n)$ and $x(n)$ are related as, $x_p(n) = \sum_{l=-\infty}^{\infty} x(n-lN)$

In otherwords, both $x_p(n)$ and $x(n)$ have same DFT if its length is 'N' samples. Here $x(n) = x_p(n)$; $0 \le n \le N-1$.

Here, the length of $x(n)$ must be less (or) equal to 'N' to hold the relationship given by Eq(1).

Scanned with CamScanner

3) Relationship between DFT and Z-transform :-

The Z-transform of a sequence $x(n)$ is given by,

$$X(z) = \sum_{n=-\infty}^{\infty} x(n)\, z^{-n} \longrightarrow (1)$$

with an ROC that includes the unit circle.

If $X(z)$ is sampled at the 'N' equally spaced points on the unit circle $z_k = e^{j2\pi k/N}$; $0 \le k \le N-1$

$$X(k) = X(z)\Big|_{z = e^{\frac{j2\pi}{N}k}}$$

$$X(k) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-\frac{j2\pi}{N}kn} \longrightarrow (2)$$

Eq(1) is identical to the Fourier transform $X(\omega)$ evaluated at the N equally spaced frequencies $\omega_k = \frac{2\pi}{N}k$; $0 \le k \le N-1$.

If the sequence $x(n)$ has a finite duration of length 'N' or less, the sequence can be recovered from its N-point DFT. Hence its Z-transform is uniquely determined by its N-point DFT. Consequently, $X(z)$ can be expressed as a function of the DFT, $X(k)$ as follows :

$$X(z) = \sum_{n=0}^{N-1} x(n)\, z^{-n}$$

$$X(z) = \sum_{n=0}^{N-1} \left[\frac{1}{N} \sum_{k=0}^{N-1} X(k)\, e^{\frac{j2\pi}{N}kn} \right] z^{-n}$$

$$X(z) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) \sum_{n=0}^{N-1} \left(e^{\frac{j2\pi}{N}k}\, z^{-1} \right)^n \longrightarrow (3)$$

$$X(z) = \frac{1 - z^{-N}}{N} \sum_{k=0}^{N-1} \frac{X(k)}{1 - e^{\frac{j2\pi}{N}k}\, z^{-1}}$$

When evaluated on the unit circle, Eq(3) yields the Fourier transform of the finite-duration sequence in terms of its DFT, in the form :

$$X(\omega) = \frac{1 - e^{-j\omega N}}{N} \sum_{k=0}^{N-1} \frac{X(k)}{1 - e^{-j(\omega - \frac{2\pi}{N}k)}} \longrightarrow (4)$$

Scanned with CamScanner

This expression for the Fourier transform is a polynomial (Lagrange) interpolation formula for $X(\omega)$ expressed in terms of the values $\{X(k)\}$ of the polynomial at a set of equally spaced discrete frequencies $\omega_k = \frac{2\pi}{N}k$; $0 \le k \le N-1$.

Important Theory Questions :-

1) Define DFT and IDFT.
2) Explain the frequency-domain sampling and reconstruction of discrete-time signals.
3) Obtain the relationship between i) DFT and DTFT ii) DFT and DFS iii) DFT and Z-transform.
4) State and prove i) periodicity / cyclic ii) Symmetry properties of twiddle factor.
5) What is Zero padding ? What are its uses ?

Scanned with CamScanner

Module-1

Anusha.M.N
Asst. professor
Dept. of ECE
BGSIT.

Discrete-Fourier Transform (DFT)

Problems on N-point DFT :

I. Compute the N-point DFT for the following sequences :

1) $x(n) = \delta(n)$

Soln:-

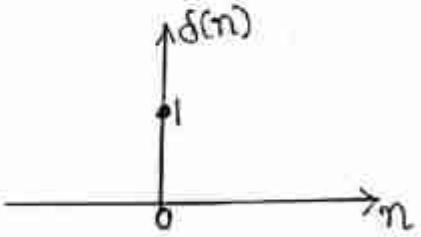
W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$= \sum_{n=0}^{N-1} \delta(n) W_N^{Kn}$

$= 1 . W_N^{Kn}\Big|_{n=0}$

$= e^{-j\frac{2\pi}{n} K \times 0}$

$\langle X(K) = 1 \rangle$



2) $x(n) = \delta(n-n_0)$

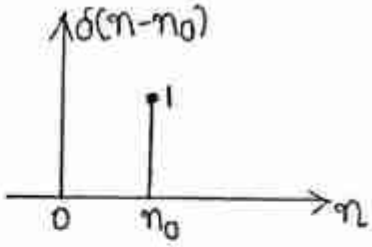
Soln:-

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$= \sum_{n=0}^{N-1} \delta(n-n_0)\, e^{-j\frac{2\pi}{N}Kn}$

$= 1 . e^{-j\frac{2\pi}{N}Kn}\Big|_{n=n_0}$

$\langle X(K) = e^{-j\frac{2\pi}{N}Kn_0} \rangle$

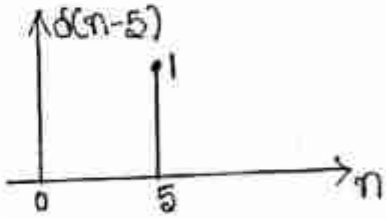


3) $x(n) = \delta(n-5)$

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$= \sum_{n=0}^{N-1} \delta(n-5)\, e^{-j\frac{2\pi}{N}Kn}$

$X(K) = 1 . e^{-j\frac{2\pi}{N}Kn}\Big|_{n=5} = e^{-j\frac{10\pi K}{N}}$



Scanned with CamScanner

4) $x(n) = u(n) - u(n-n_0)$

Soln:-

$$X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$$

$$= \sum_{n=0}^{n_0-1} 1 \cdot e^{-j\frac{2\pi}{N}Kn}$$

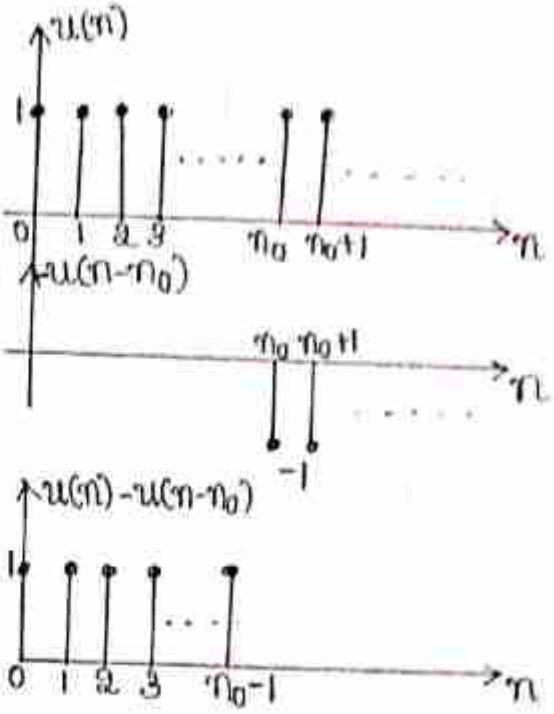
$$= \sum_{n=0}^{n_0-1} \left(e^{-j\frac{2\pi}{N}K}\right)^n$$

$$= \frac{1 - \left(e^{-j\frac{2\pi}{N}K}\right)^{n_0-1+1}}{1 - e^{-j\frac{2\pi}{N}K}}$$

$$= \frac{1 - e^{-j\frac{2\pi}{N}Kn_0}}{1 - e^{-j\frac{2\pi}{N}K}}$$

$$= \frac{e^{-j\frac{\pi}{N}Kn_0}\left(e^{j\frac{\pi}{N}Kn_0} - e^{-j\frac{\pi}{N}Kn_0}\right)}{e^{-j\frac{\pi}{N}K}\left(e^{j\frac{\pi}{N}K} - e^{-j\frac{\pi}{N}K}\right)}$$

$$X(K) = e^{-j\frac{\pi}{N}K(n_0-1)} \cdot \frac{\sin\left(\frac{\pi}{N}Kn_0\right)}{\sin\left(\frac{\pi}{N}K\right)}$$



$$\sum_{n=n_1}^{n_2} a^n = \frac{a^{n_1} - a^{n_2+1}}{1-a}$$

5) $x(n) = 1$

Soln:-

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$

$$= \sum_{n=0}^{N-1} 1 \cdot e^{-j\frac{2\pi}{N}Kn}$$

$$X(K) = \sum_{n=0}^{N-1} \left(e^{-j\frac{2\pi}{N}K}\right)^n \longrightarrow (1)$$

$$X(K) = \frac{1 - \left(e^{-j\frac{2\pi}{N}K}\right)^{N-1+1}}{1 - e^{-j\frac{2\pi}{N}K}} = 0 \; ; \; K \neq 0$$

Scanned with CamScanner

When $K=0$ in Eq(1),

$$X(K) = \sum_{n=0}^{N-1} 1 = N-1+1 = N\,\delta(K)$$

$$\sum_{n=n_1}^{n_2} 1 = n_2 - n_1 + 1$$

$$\therefore X(K) = \begin{cases} N\delta(K) & ; K=0 \\ 0 & ; K\neq 0 \end{cases}$$

6) $x(n) = a^n$ for $0<a<1$

Sol^n:-

$$X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$$

$$= \sum_{n=0}^{N-1} a^n\, e^{-j\frac{2\pi}{N}Kn}$$

$$= \sum_{n=0}^{N-1} \left(a e^{-j\frac{2\pi}{N}K}\right)^n$$

$$= \frac{1-\left(a e^{-j\frac{2\pi}{N}K}\right)^{N-1+1}}{1-a e^{-j\frac{2\pi}{N}K}}$$

$$= \frac{1-a^N e^{-j\frac{2\pi}{N}K.N}}{1-a e^{-j\frac{2\pi}{N}K}}$$

$$\boxed{X(K) = \frac{1-a^N}{1-a e^{-j\frac{2\pi}{N}K}}}$$

7) $x(n) = \cos\left(\frac{2\pi K_0 n}{N}\right)$; $0 \le n \le N-1$

(or)

$x(n) = \cos(w_0 n)$; $w_0 = \frac{2\pi}{N} K_0$; $0 \le n \le N-1$.

Sol:- $x(n) = \frac{e^{j\frac{2\pi}{N}K_0 n} + e^{-j\frac{2\pi}{N}K_0 n}}{2} = \frac{W_N^{-K_0 n} + W_N^{K_0 n}}{2}$

Scanned with CamScanner

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$$= \sum_{n=0}^{N-1} \left[\frac{W_N^{-K_0 n} + W_N^{K_0 n}}{2} \right] W_N^{Kn}$$

$$= \frac{1}{2}\left[\sum_{n=0}^{N-1} e^{\frac{j2\pi}{N} K_0 n} \cdot e^{-\frac{j2\pi}{N} Kn} + \sum_{n=0}^{N-1} e^{-\frac{j2\pi}{N} K_0 n} \cdot e^{-\frac{j2\pi}{N} Kn} \right]$$

$$= \frac{1}{2}\left\{ \sum_{n=0}^{N-1} \left[e^{-\frac{j2\pi}{N}(K-K_0)} \right]^n + \sum_{n=0}^{N-1} \left[e^{-\frac{j2\pi}{N}(K+K_0)} \right]^n \right\}$$

$$X(K) = \frac{1}{2}\left[S_1 + S_2 \right]$$

To find S_1 and S_2:

$$S_1 = \sum_{n=0}^{N-1} \left[e^{-\frac{j2\pi}{N}(K-K_0)} \right]^n \longrightarrow (1)$$

$$= \frac{1 - \left(e^{-\frac{j2\pi}{N}(K-K_0)} \right)^{N-1+1}}{1 - e^{-\frac{j2\pi}{N}(K-K_0)}}$$

$\langle S_1 = 0 \; ; \; K \neq K_0 \rangle$

When $K = K_0$ in Eq(1),

$$S_1 = \sum_{n=0}^{N-1} 1 = N - 1 - 0 + 1 = N\delta(K-K_0)$$

$$\therefore S_1 = \begin{cases} N\delta(K-K_0) & ; K = K_0 \\ 0 & ; K \neq K_0 \end{cases}$$

Similarly, $S_2 = \begin{cases} N\delta(K+K_0) & ; K = -K_0 \\ 0 & ; K \neq -K_0 \end{cases}$

Scanned with CamScanner

$$\therefore X(K) = \frac{1}{2}\left[N\delta(K-K_0) + N\delta(K+K_0)\right]$$

$$\left\langle X(K) = \frac{N}{2}\delta(K-K_0) + \frac{N}{2}\delta(K+K_0) \right\rangle$$

8) $x(n) = \sin\left(\frac{2\pi}{N}K_0 n\right)$; $0 \le n \le N-1$

(or)

$x(n) = \sin(W_0 n)$; $W_0 = \frac{2\pi}{N}K_0$.

Sol^n:- $x(n) = \dfrac{e^{j\frac{2\pi}{N}K_0 n} - e^{-j\frac{2\pi}{N}K_0 n}}{2j} = \dfrac{W_N^{-K_0 n} - W_N^{K_0 n}}{2j}$

$$X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$$

$$= \sum_{n=0}^{N-1}\left[\frac{W_N^{-K_0 n} - W_N^{K_0 n}}{2j}\right] W_N^{Kn}$$

$$= \frac{1}{2j}\left[\sum_{n=0}^{N-1} W_N^{-K_0 n}\cdot W_N^{Kn} - \sum_{n=0}^{N-1} W_N^{K_0 n}\, W_N^{Kn}\right]$$

$$= \frac{1}{2j}\left\{\sum_{n=0}^{N-1}\left[e^{-j\frac{2\pi}{N}(K-K_0)}\right]^n - \sum_{n=0}^{N-1}\left[e^{-j\frac{2\pi}{N}(K+K_0)}\right]^n\right\}$$

$$X(K) = \frac{1}{2j}\left[S_1 - S_2\right]$$

To find S_1 and S_2:

$$S_1 = \sum_{n=0}^{N-1}\left[e^{-j\frac{2\pi}{N}(K-K_0)}\right]^n = \frac{1-\left[e^{-j\frac{2\pi}{N}(K-K_0)}\right]^{N-1+1}}{1-e^{-j\frac{2\pi}{N}(K-K_0)}} = \underline{\underline{0}} \; ; \; K \neq K_0$$

When $K = K_0$,

$$S_1 = \sum_{n=0}^{N-1} 1 = N-1+1 = \underline{\underline{N\delta(K-K_0)}}$$

Scanned with CamScanner

$\therefore S_1 = \begin{cases} N\delta(K-K_0) \; ; \; K = K_0 \\ 0 \quad ; \; K \neq K_0 \end{cases}$

Similarly, $S_2 = \begin{cases} N\delta(K+K_0) \; ; \; K = -K_0 \\ 0 \quad ; \; K \neq K_0 \end{cases}$

$\therefore X(K) = \frac{1}{2j}\left[N\delta(K-K_0) - N\delta(K+K_0)\right]$

$$\boxed{X(K) = \frac{N}{2j}\delta(K-K_0) - \frac{N}{2j}\delta(K+K_0)}$$

9) $x(n) = an \; ; 0 \leq n \leq N-1$.

Sol^n:-

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} = \sum_{n=0}^{N-1} an\, W_N^{Kn} = a\sum_{n=0}^{N-1} n\, W_N^{Kn} \longrightarrow (1)$$

Let $S = \sum_{n=0}^{N-1} b^n = \frac{1-b^N}{1-b} = \frac{b^N-1}{b-1} \; ; b \neq 1$

Differentiating both sides of the above Eq^n w.r.t 'b',

$$\sum_{n=0}^{N-1} n b^{n-1} = \frac{(b-1)(Nb^{N-1}-0) - (b^N-1)(1-0)}{(b-1)^2}$$

$$\sum_{n=0}^{N-1} n b^n = \frac{b\left[Nb^N - Nb^{N-1} - b^N + 1\right]}{(b-1)^2}$$

$$= \frac{b\left[b^N(N-1) - Nb^{N-1} + 1\right]}{(b-1)^2}$$

Let $b = W_N^K$ in the above Eq^n,

$$\sum_{n=0}^{N-1} n.W_N^{Kn} = \frac{W_N^K\left[W_N^{KN}(N-1) - N\,W_N^{K(N-1)} + 1\right]}{(W_N^K - 1)^2}$$

Scanned with CamScanner

9) $$\sum_{n=0}^{N-1} n W_N^{Kn} = \frac{W_N^K \left[N-1-NW_N^{-K}+1\right]}{(W_N^K-1)^2}$$

$$= \frac{N W_N^K (1-W_N^{-K})}{(W_N^K-1)^2}$$

$$\sum_{n=0}^{N-1} n W_N^{Kn} = \frac{N(W_N^K-1)}{(W_N^K-1)^2} = \frac{N}{(W_N^K-1)} \quad ; K \neq 0$$

When $K=0$ in Eq(1),

$$X(K) = a \sum_{n=0}^{N-1} n = a\frac{N(N-1)}{2}$$

$$\therefore X(K) = \begin{cases} a\frac{N(N-1)}{2} & ; K=0 \\ \frac{aN}{(W_N^K-1)} & ; K \neq 0 \end{cases}$$

10) $x(n) = e^{j\omega mn}$, where $\omega = \frac{2\pi}{N}$

Sol^n:-

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{N-1} e^{j\frac{2\pi}{N}mn} e^{-j\frac{2\pi}{N}Kn}$$

$$= \sum_{n=0}^{N-1} \left[e^{-j\frac{2\pi}{N}(K-m)} \right]^n$$

$$= \frac{1-\left[e^{-j\frac{2\pi}{N}(K-m)}\right]^{N-1+1}}{1-e^{-j\frac{2\pi}{N}(K-m)}}$$

$X(K) = 0 \quad ; K \neq m$

Scanned with CamScanner

When $K = rN$,

$$X(K) = \sum_{n=0}^{N-1} 1 = N - 1 + 1 = N\delta(K - rN)$$

$$\therefore X(K) = \begin{cases} N\delta(K-rN) & ; K = rN \\ 0 & ; K \neq rN \end{cases}$$

11) $$x(n) = \begin{cases} \frac{1}{3} & ; 0 \le n \le 2 \\ 0 & ; \text{otherwise} \end{cases}$$

Sol^n :-

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{2} \left(\frac{1}{3}\right) W_N^{Kn}$$

$$= \frac{1}{3} \sum_{n=0}^{2} \left(e^{-j\frac{2\pi}{N}K}\right)^n$$

$$= \frac{1}{3}\left[\frac{1 - \left(e^{-j\frac{2\pi}{N}K}\right)^3}{1 - e^{-j\frac{2\pi}{N}K}}\right]$$

$$\left\langle X(K) = \frac{\frac{1}{3} - \frac{1}{3} e^{-j6\pi K/N}}{1 - e^{-j\frac{2\pi}{N}K}} \right\rangle$$

12) Find the DFT of the sequence $x(n) = 0.5^n u(n)$ for $0 \le n \le 3$ by evaluating $x(n) = a^n$ for $0 \le n \le N-1$.

Sol^n :- W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$$= \sum_{n=0}^{N-1} a^n e^{-j\frac{2\pi}{N}Kn}$$

$$= \sum_{n=0}^{N-1} \left(a e^{-j\frac{2\pi}{N}K}\right)^n$$

Scanned with CamScanner

$$X(K) = \frac{1-\left(a\,e^{-j\frac{2\pi}{N}K}\right)^{N-1+1}}{1-a\,e^{-j\frac{2\pi}{N}K}}$$

$$\left\langle X(K) = \frac{1-a^N}{1-a\,e^{-j\frac{2\pi}{N}K}} \right\rangle \longrightarrow (1)$$

To determine the DFT of $x(n) = (0.5)^n u(n)$; $n = 0,1,2,3$

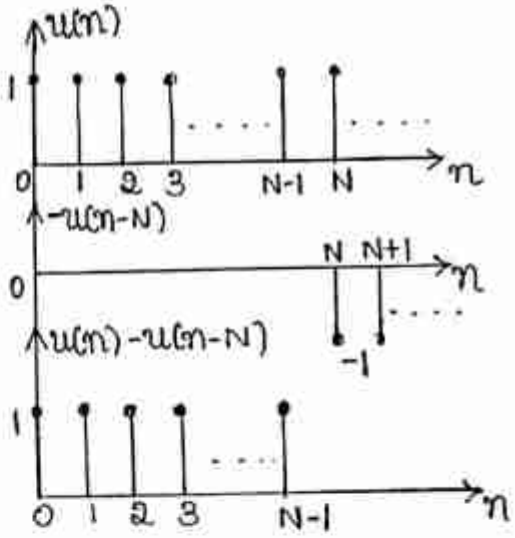
Here $N = 4$ and $a = 0.5$

Substitute these values in Eq(1),

$$X(K) = \frac{1-(0.5)^4}{1-0.5\,e^{-j\frac{2\pi}{4}K}} = \frac{0.9375}{1-0.5\,e^{-j\frac{\pi}{2}K}} \quad ; 0 \le K \le 3$$

$$\left\langle X(K) = \{1.875, 0.75-j0.375, 0.625, 0.75+j0.375\} \right\rangle$$

13) $x(n) = u(n) - u(n-N)$



Sol^n:-

$$x(n) = \begin{cases} 1 & ; 0 \le n \le N-1 \\ 0 & ; \text{else} \end{cases}$$

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{Kn}$

$$= \sum_{n=0}^{N-1} 1 \cdot e^{-j\frac{2\pi}{N}Kn}$$

$$= \sum_{n=0}^{N-1} \left(e^{-j\frac{2\pi}{N}K}\right)^n$$

$$X(K) = \frac{1-\left(e^{-j\frac{2\pi}{N}K}\right)^{N-1+1}}{1-e^{-j\frac{2\pi}{N}K}} = 0 \quad ; K \neq 0$$

When $K = 0$, $X(K) = \sum_{n=0}^{N-1} 1 = N-1-0+1 = N\,\delta(K)$

$$\therefore X(K) = \begin{cases} N\delta(K) & ; K = 0 \\ 0 & ; K \neq 0 \end{cases}$$

Scanned with CamScanner

Problems on DFT as a linear Transformation (Matrix method):

Compute the DFT of the following sequences using matrix method:

1) $x(n) = \{0, 1, 2, 3\}$

Soln:- $X_N = W_N x_N$

Here $N = 4$,

$X_4 = W_4 x_4$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix}$$

$$= \begin{bmatrix} 0+1+2+3 \\ 0-j-2+j3 \\ 0-1+2-3 \\ 0+j-2-j3 \end{bmatrix} = \begin{bmatrix} 6 \\ -2+j2 \\ -2 \\ -2-j2 \end{bmatrix}$$

$\therefore X(K) = \{6, -2+j2, -2, -2-j2\}$

2) $x(n) = \{1, 0, 0, 1\}$

Soln:- $X_N = W_N x_N$

Here $N = 4$,

$X_4 = W_4 x_4$

$W_4^0 = 1$; $W_4^1 = -j$
$W_4^2 = -1$; $W_4^3 = j$

Scanned with CamScanner

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1+0+0+1 \\ 1+0+0+j \\ 1+0+0-1 \\ 1+0+0-j \end{bmatrix}$$

$$= \begin{bmatrix} 2 \\ 1+j \\ -1 \\ 1-j \end{bmatrix}$$

$\therefore X(K) = \{2, 1+j, 0, 1-j\}$

3) $x(n) = \sin^2\left(\frac{n\pi}{2} + \frac{\pi}{3}\right)$, Compute 4-point DFT using matrix method.

Sol^n:- Here $N = 4$,

@ $n=0$, $x(0) = \sin^2\left(0 \times \frac{\pi}{2} + \frac{\pi}{3}\right) = \frac{3}{4}$

@ $n=1$, $x(1) = \sin^2\left(1 \times \frac{\pi}{2} + \frac{\pi}{3}\right) = \frac{1}{4}$

@ $n=2$, $x(2) = \sin^2\left(2 \times \frac{\pi}{2} + \frac{\pi}{3}\right) = \frac{3}{4}$

@ $n=3$, $x(3) = \sin^2\left(3 \times \frac{\pi}{2} + \frac{\pi}{3}\right) = \frac{1}{4}$

$\therefore x(n) = \left\{\frac{3}{4}, \frac{1}{4}, \frac{3}{4}, \frac{1}{4}\right\}$

Scanned with CamScanner

W.K.T $X_N = W_N x_N$

Here $N=4$, $X_4 = W_4 x_4$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 3/4 \\ 1/4 \\ 3/4 \\ 1/4 \end{bmatrix}$$

$$= \begin{bmatrix} 2 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$\therefore X(K) = \{2, 0, 1, 0\}$

$W_4^0 = 1 \; ; \; W_4^1 = -j$

$W_4^2 = -1 \; ; \; W_4^3 = j$

4) $x(n) = \cos^2\left(\frac{n\pi}{2} + \frac{\pi}{8}\right)$

Sol:- $x(n) = \left\{\frac{1}{4}, \frac{3}{4}, \frac{1}{4}, \frac{3}{4}\right\}$

$X_N = W_N x_N$

Here $N=4$, $X_4 = W_4 x_4$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1/4 \\ 3/4 \\ 1/4 \\ 3/4 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ -1 \\ 0 \end{bmatrix}$$

$\therefore X(K) = \{2, 0, -1, 0\}$

$W_4^0 = 1 \; ; \; W_4^1 = -j$

$W_4^2 = -1 \; ; \; W_4^3 = j$

Scanned with CamScanner

Compute the IDFT for the following sequences using matrix method:

1) $X(K) = \{4, -j2, 0, j2\}$

Sol:- $x_N = \frac{1}{N} W_N^{*} X_N$

$W_4^{-0} = 1 \; ; \; W_4^{-2} = -1$

$W_4^{-1} = j \; ; \; W_4^{-3} = -j$

Here $N = 4$,

$$x_4 = \frac{1}{4} W_4^{*} X_4$$

$$\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} W_4^{-0} & W_4^{-0} & W_4^{-0} & W_4^{-0} \\ W_4^{-0} & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ W_4^{-0} & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ W_4^{-0} & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 4 \\ -j2 \\ 0 \\ j2 \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 4 - j2 + 0 + j2 \\ 4 + 2 + 0 + 2 \\ 4 + j2 + 0 - j2 \\ 4 - 2 + 0 - 2 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 4 \\ 8 \\ 4 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 2 \\ 1 \\ 0 \end{bmatrix}$$

$\therefore x(n) = \{1, 2, 1, 0\}$

Scanned with CamScanner

2) $X(K) = \{2, 1+j, 0, 1-j\}$

Soln:- W.K.T $x_N = \frac{1}{N} W_N^* X_N$

Here $N=4$, $x_4 = \frac{1}{4} W_4^* X_4$

$W_4^{-0} = 1$; $W_4^{-1} = j$

$W_4^{-2} = -1$; $W_4^{-3} = -j$

$$\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} W_4^{-0} & W_4^{-0} & W_4^{-0} & W_4^{-0} \\ W_4^{-0} & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ W_4^{-0} & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ W_4^{-0} & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 2 \\ 1+j \\ 0 \\ 1-j \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 2+1+j+0+1-j \\ 2+j-1+0-j-1 \\ 2-1-j+0-1+j \\ 2-j+1+0+j+1 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 4 \\ 0 \\ 0 \\ 4 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$\therefore x(n) = \{1, 0, 0, 1\}$

Scanned with CamScanner

Problems on DFT and IDFT :

1) Find the 4-point DFT of the sequence $x(n) = \{1, 2, 3, 4\}$

Sol^n:- Given $N = 4$,

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$$= x(0)W_4^0 + x(1)W_4^K + x(2)W_4^{2K} + x(3)W_4^{3K}$$

$$X(K) = 1 + 2W_4^K + 3W_4^{2K} + 4W_4^{3K} \longrightarrow (1)$$

$W_4^0 = e^{\frac{-j2\pi}{4} \cdot 0} = 1$

$W_4^1 = e^{\frac{-j2\pi}{4} \cdot 1} = -j$

$W_4^2 = e^{\frac{-j2\pi}{4} \cdot 2} = -1$

$W_4^3 = e^{\frac{-j2\pi}{4} \cdot 3} = j$

@K=0, $X(0) = 1 + 2W_4^0 + 3W_4^0 + 4W_4^0 = 1+2+3+4 = \underline{10}$

@K=1, $X(1) = 1 + 2W_4^1 + 3W_4^2 + 4W_4^3 = 1+2(-j)+3(-1)+4(j) = \underline{-2+j2}$

@K=2, $X(2) = 1 + 2W_4^2 + 3W_4^4 + 4W_4^6 = 1+2(-1)+3(1)+4(-1) = \underline{-2}$

@K=3, $X(3) = 1 + 2W_4^3 + 3W_4^6 + 4W_4^9 = 1+2(j)+3(-1)+4(-j) = \underline{-2-j2}$

$\therefore X(K) = \{10, -2+j2, -2, -2-j2\}$

2) $x(n) = \{1, 0, 0, 1\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N = 4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$= x(0)W_4^0 + x(1)W_4^K + x(2)W_4^{2K} + x(3)W_4^{3K}$$

$$X(K) = 1 + W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$; $W_4^2 = -1$

$W_4^1 = -j$; $W_4^3 = j$

@K=0, $X(0) = 1 + W_4^0 = 1+1 = \underline{2}$

@K=1, $X(1) = 1 + W_4^3 = \underline{1+j}$

@K=2, $X(2) = 1 + W_4^6 = 1-1 = \underline{0}$

@K=3, $X(3) = 1 + W_4^9 = \underline{1-j}$

$\therefore X(K) = \{2, 1+j, 0, 1-j\}$

Scanned with CamScanner

3) $x(n) = \{1, 0, 1, 0\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = 1 + W_4^{2K} \longrightarrow (1)$$

@ $K=0$, $X(0) = 1 + W_4^0 = 1+1 = \underline{2}$

@ $K=1$, $X(1) = 1 + W_4^2 = 1-1 = \underline{0}$

@ $K=2$, $X(2) = 1 + W_4^4 = 1+1 = \underline{2}$

@ $K=3$, $X(3) = 1 + W_4^6 = 1-1 = \underline{0}$

$\therefore X(K) = \{2, 0, 2, 0\}$

$W_4^0 = 1$; $W_4^2 = -1$

$W_4^1 = -j$; $W_4^3 = j$

4) $x(n) = \{1, 1, 2, 2\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = 1 + W_4^{K} + 2W_4^{2K} + 2W_4^{3K} \longrightarrow (1)$$

@ $K=0$, $X(0) = 1 + W_4^0 + 2W_4^0 + 2W_4^0 = 1+1+2+2 = \underline{6}$

@ $K=1$, $X(1) = 1 + W_4^1 + 2W_4^2 + 2W_4^3 = 1 - j + 2(-1) + 2(j) = \underline{-1+j}$

@ $K=2$, $X(2) = 1 + W_4^2 + 2W_4^4 + 2W_4^6 = 1 - 1 + 2(1) + 2(-1) = \underline{0}$

@ $K=3$, $X(3) = 1 + W_4^3 + 2W_4^6 + 2W_4^9 = 1 + j + 2(-1) + 2(-j) = \underline{-1-j}$

$\therefore X(K) = \{6, -1+j, 0, -1-j\}$

$W_4^0 = 1$; $W_4^1 = -j$

$W_4^2 = -1$; $W_4^3 = j$

5) $x(n) = \{0, 1, 2, 3\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = W_4^{K} + 2W_4^{2K} + 3W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$; $W_4^1 = -j$

$W_4^2 = -1$; $W_4^3 = j$

Scanned with CamScanner

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$$X(K) = 6 + 7W_4^K + 6W_4^{2K} + 5W_4^{3K} \longrightarrow (1)$$

@K=0, $X(0) = 6 + 7W_4^0 + 6W_4^0 + 5W_4^0 = 6+7+6+5 = \underline{24}$

@K=1, $X(1) = 6 + 7W_4^1 + 6W_4^2 + 5W_4^3 = 6+7(-j)+6(-1)+5(j) = \underline{-j2}$

@K=2, $X(2) = 6 + 7W_4^2 + 6W_4^4 + 5W_4^6 = 6+7(-1)+6(1)+5(-1) = \underline{0}$

@K=3, $X(3) = 6 + 7W_4^3 + 6W_4^6 + 5W_4^9 = 6+7(j)+6(-1)+5(-j) = \underline{j2}$

$\therefore X(K) = \{24, -j2, 0, j2\}$

8) Compute the DFT of the sequence $x(n) = \cos\left(\frac{n\pi}{4}\right)$ for N=4, plot |X(K)| and $\underline{|X(K)}$.

Soln:- Here N=4,

@n=0, $x(0) = \cos\left(\frac{\pi}{4} \times 0\right) = \underline{1}$

@n=1, $x(1) = \cos\left(\frac{\pi}{4} \times 1\right) = \underline{0.7071}$

@n=2, $x(2) = \cos\left(\frac{\pi}{4} \times 2\right) = \underline{0}$

@n=3, $x(3) = \cos\left(\frac{\pi}{4} \times 3\right) = \underline{-0.7071}$

$\therefore x(n) = \{1, 0.7071, 0, -0.7071\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

$$X(K) = 1 + 0.7071W_4^K - 0.7071W_4^{3K} \longrightarrow (1)$$

@K=0, $X(0) = 1 + 0.7071W_4^0 - 0.7071W_4^0 = 1+0.7071-0.7071 = \underline{1}$

@K=1, $X(1) = 1 + 0.7071W_4^1 - 0.7071W_4^3 = 1+0.7071(-j)-0.7071(j)$

$= \underline{1 - j\,1.4142}$

Scanned with CamScanner

@K=0, $X(0) = W_4^0 + 2W_4^0 + 3W_4^0 = 1+2+3 = \underline{6}$

@K=1, $X(1) = W_4^1 + 2W_4^2 + 3W_4^3 = -j + 2(-1) + 3(j) = \underline{-2+j2}$

@K=2, $X(2) = W_4^2 + 2W_4^4 + 3W_4^6 = -1 + 2(1) + 3(-1) = \underline{-2}$

@K=3, $X(3) = W_4^3 + 2W_4^6 + 3W_4^9 = j + 2(-1) + 3(-j) = \underline{-2-j2}$

$\therefore X(K) = \{6, -2+j2, -2, -2-j2\}$

6) $x(n) = \{1, -2, 3, 4\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$W_4^0 = 1$; $W_4^2 = -1$

$W_4^1 = -j$; $W_4^3 = j$

Here N=4, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = 1 - 2W_4^K + 3W_4^{2K} + 4W_4^{3K} \longrightarrow (1)$$

@K=0, $X(0) = 1 - 2W_4^0 + 3W_4^0 + 4W_4^0 = 1-2+3+4 = \underline{6}$

@K=1, $X(1) = 1 - 2W_4^1 + 3W_4^2 + 4W_4^3 = 1 - 2(-j) + 3(-1) + 4(j) = \underline{-2+j6}$

@K=2, $X(2) = 1 - 2W_4^2 + 3W_4^4 + 4W_4^6 = 1 - 2(-1) + 3(1) + 4(-1) = \underline{2}$

@K=3, $X(3) = 1 - 2W_4^3 + 3W_4^6 + 4W_4^9 = 1 - 2(j) + 3(-1) + 4(-j) = \underline{-2-j6}$

$\therefore X(K) = \{6, -2+j6, 2, -2-j6\}$

7) Find the 4-point DFT of the sequence $x(n) = 6 + \sin\left(\frac{2\pi n}{4}\right)$; $0 \le n \le 3$.

Sol^n:- Here N=4,

$W_4^0 = 1$

$W_4^1 = -j$

$W_4^2 = -1$

$W_4^3 = j$

@n=0, $x(0) = 6 + \sin\left(\frac{2\pi}{4}.0\right) = \underline{6}$

@n=1, $x(1) = 6 + \sin\left(\frac{2\pi}{4}.1\right) = \underline{7}$

@n=2, $x(2) = 6 + \sin\left(\frac{2\pi}{4}.2\right) = \underline{6}$

@n=3, $x(3) = 6 + \sin\left(\frac{2\pi}{4}.3\right) = \underline{5}$

$\therefore x(n) = \{6, 7, 6, 5\}$

Scanned with CamScanner

@K=2, $X(2) = 1 + 0.7071 W_4^2 - 0.7071 W_4^6 = 1 + 0.7071(-1) - 0.7071(-1) = \underline{1}$

@K=3, $X(3) = 1 + 0.7071 W_4^3 - 0.7071 W_4^9 = 1 + 0.7071(j) - 0.7071(-j)$

$= 1 + j1.4142$

$\therefore X(K) = \{1, 1-j1.4142, 1, 1+j1.4142\}$

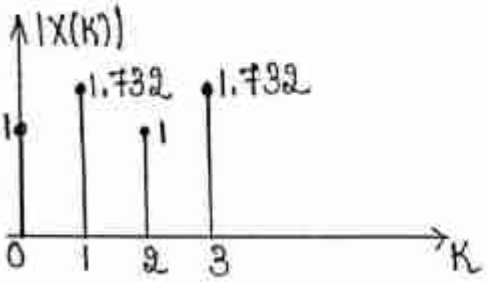
$|X(K)| = \{1, 1.732, 1, 1.732\}$

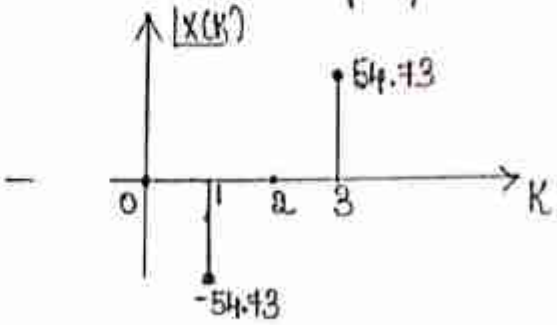
$\angle X(K) = \{0, -54.73, 0, 54.73\}$

$a+jb$

magnitude $= \sqrt{a^2+b^2}$

phase $= \tan^{-1}\left(\frac{b}{a}\right)$





9) $x(n) = \sin\left(\frac{n\pi}{2}\right)$; $0 \le n \le 3$.

Sol^n:-

@n=0, $x(0) = \sin\left(\frac{\pi}{2} \times 0\right) = 0$

@n=1, $x(1) = \sin\left(\frac{\pi}{2} \times 1\right) = 1$

@n=2, $x(2) = \sin\left(\frac{\pi}{2} \times 2\right) = 0$

@n=3, $x(3) = \sin\left(\frac{\pi}{2} \times 3\right) = -1$

$x(n) = \{0, 1, 0, -1\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$= \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$$X(K) = W_4^K - W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

@K=0, $X(0) = W_4^0 - W_4^0 = 1 - 1 = \underline{0}$

@K=1, $X(1) = W_4^1 - W_4^3 = -j - j = \underline{-j2}$

Scanned with CamScanner

@K = 2, $X(2) = W_4^2 - W_4^6 = -1-(-1) = \underline{0}$

@K = 3, $X(3) = W_4^3 - W_4^9 = j-(-j) = \underline{j2}$

$\therefore X(K) = \{0, -j2, 0, j2\}$

10) $x(n) = \cos\left(\frac{n\pi}{4}\right) + \sin\left(\frac{n\pi}{4}\right)$; $0 \le n \le 3$.

Sol^n:-

@n = 0, $x(0) = \cos\left(\frac{\pi}{4}\times 0\right) + \sin\left(\frac{\pi}{4}\times 0\right) = \underline{1}$

@n = 1, $x(1) = \cos\left(\frac{\pi}{4}\times 1\right) + \sin\left(\frac{\pi}{4}\times 1\right) = \underline{1.4142}$

@n = 2, $x(2) = \cos\left(\frac{\pi}{4}\times 2\right) + \sin\left(\frac{\pi}{4}\times 2\right) = \underline{1}$

@n = 3, $x(3) = \cos\left(\frac{\pi}{4}\times 3\right) + \sin\left(\frac{\pi}{4}\times 3\right) = \underline{0}$

$\therefore x(n) = \{1, 1.4142, 1, 0\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$$X(K) = 1 + 1.4142 W_4^K + W_4^{2K} \longrightarrow (1)$$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

@K = 0, $X(0) = 1 + 1.4142 W_4^0 + W_4^0 = 1 + 1.4142 + 1 = \underline{3.4142}$

@K = 1, $X(1) = 1 + 1.4142 W_4^1 + W_4^2 = 1 + 1.4142(-j) - 1 = \underline{-j1.4142}$

@K = 2, $X(2) = 1 + 1.4142 W_4^2 + W_4^4 = 1 + 1.4142(-1) + 1 = \underline{0.586}$

@K = 3, $X(3) = 1 + 1.4142 W_4^3 + W_4^6 = 1 + 1.4142(j) + (-1) = \underline{j1.4142}$

$\therefore X(K) = \{3.4142, -j1.4142, 0.586, j1.4142\}$

11) $x(n) = \begin{cases} 1/3 & ; 0 \le n \le 3 \\ 0 & ; \text{otherwise} \end{cases}$

Sol^n:- $x(n) = \{1/3, 1/3, 1/3, 1/3\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

Here N = 4,

$$X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn} = \frac{1}{3}\left[1 + W_4^K + W_4^{2K} + W_4^{3K}\right] \longrightarrow (1)$$

Scanned with CamScanner

@K=0, $X(0) = \frac{1}{3}[1+W_4^0+W_4^0+W_4^0] = 4/3$

@K=1, $X(1) = \frac{1}{3}[1+W_4^1+W_4^2+W_4^3] = \frac{1}{3}[1-j-1+j] = 0$

@K=2, $X(2) = \frac{1}{3}[1+W_4^2+W_4^4+W_4^6] = \frac{1}{3}[1-1+1-1] = 0$

@K=3, $X(3) = \frac{1}{3}[1+W_4^3+W_4^6+W_4^9] = \frac{1}{3}[1+j-1-j] = 0$

$\therefore X(K) = \{4/3, 0, 0, 0\}$

12) $x(n) = u(n) - u(n-4)$

Sol^n:- $x(n) = \{1, 1, 1, 1\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

Here N = 4,

$$X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$$

$$X(K) = 1 + W_4^K + W_4^{2K} + W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

@K=0, $X(0) = 1+W_4^0+W_4^0+W_4^0 = 1+1+1+1 = 4$

@K=1, $X(1) = 1+W_4^1+W_4^2+W_4^3 = 1-j-1+j = 0$

@K=2, $X(2) = 1+W_4^2+W_4^4+W_4^6 = 1-1+1-1 = 0$

@K=3, $X(3) = 1+W_4^3+W_4^6+W_4^9 = 1+j-1-j = 0$

$\therefore X(K) = \{4, 0, 0, 0\}$

13) Compute 5-point DFT of the sequence $x(n) = \{1, 2, 3, 4, 5\}$ using direct method.

Sol^n:- $$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

$W_5^3 = -0.809 + j0.588$
$W_5^4 = 0.309 + j0.951$

$W_5^0 = e^{-j\frac{2\pi}{5}.0} = 1$
$W_5^1 = e^{-j\frac{2\pi}{5}.1} = 0.809 - j0.951$
$W_5^2 = -0.809 - j0.588$

Here,

$$X(K) = \sum_{n=0}^{4} x(n) W_5^{Kn} = 1 + 2W_5^K + 3W_5^{2K} + 4W_5^{3K} + 5W_5^{4K}$$

Scanned with CamScanner

@K=0, $X(0) = 1+2W_5^0+3W_5^0+4W_5^0+5W_5^0 = 1+2+3+4+5 = 15$

@K=1, $X(1) = 1+2W_5^1+3W_5^2+4W_5^3+5W_5^4 = 1+2(0.309-j0.951)+3(-0.809-j0.588)$
$+4(-0.809+j0.588)+5(0.309+j0.951)$

$X(1) = -2.5+j3.441$

@K=2, $X(2) = 1+2W_5^2+3W_5^4+4W_5^6+5W_5^8$

$= 1+2(-0.809-j0.588)+3(0.309+j0.951)+4(0.309-j0.951)+$
$5(-0.809+j0.588) = -2.5+j0.813$

@K=3, $X(3) = 1+2W_5^3+3W_5^6+4W_5^9+5W_5^{12}$

$= 1+2(-0.809+j0.588)+3(0.309-j0.951)+4(0.309+j0.951)$
$+5(-0.809-j0.588) = -2.5-j0.813$

@K=4, $X(4) = 1+2W_5^4+3W_5^8+4W_5^{12}+5W_5^{16}$

$= 1+2(0.309+j0.951)+3(-0.809+j0.588)+4(-0.809-j0.588)$
$+5(0.309-j0.951) = -2.5-j3.441$

$\therefore X(K) = \{15, -2.5+j3.441, -2.5+j0.813, -2.5-j0.813, -2.5-j3.441\}$

11) Compute a) 5-point DFT of the sequence $x_1(n) = \{1,1,1,0,0\}$ b) 3-point DFT of the sequence $x_2(n) = \{1,1,1\}$ by using direct method c) Compare the result of (a) and (b).

Soln:-

a) Given $x_1(n) = \{1,1,1,0,0\}$

$$X_1(K) = \sum_{n=0}^{N-1} x_1(n) W_N^{Kn}$$

Here N=5, $X_1(K) = \sum_{n=0}^{4} x_1(n) W_5^{Kn}$

$$X_1(K) = 1+W_5^K+W_5^{2K} \longrightarrow (1)$$

$W_5^0 = 1$

$W_5^1 = 0.309-j0.951$

$W_5^2 = -0.809-j0.588$

$W_5^3 = -0.809+j0.588$

$W_5^4 = 0.309+j0.951$

@K=0, $X_1(0) = 1+W_5^0+W_5^0 = 1+1+1 = 3$

@K=1, $X_1(1) = 1+W_5^1+W_5^2 = 1+0.309-j0.951-0.809-j0.588$

$= 0.5-j1.5387$

Scanned with CamScanner

@K=2, $X_1(2) = 1+W_5^2+W_5^4 = 1-0.809+j0.588+0.309+j0.951$

$= 0.5+j0.3633$

@K=3, $X_1(3) = 1+W_5^3+W_5^6 = 1+W_5^3+W_5^1 = 1-0.809+j0.5877+0.309-j0.951$

$= 0.5-j0.3633$

@K=4, $X_1(4) = 1+W_5^4+W_5^8 = 1+0.309+j0.951-0.809+j0.588$

$= 0.5+j1.5387$

$\therefore X_1(K) = \{3, 0.5-j1.5387, 0.5+j0.3633, 0.5-j0.3633, 0.5+j1.5387\}$

b) Given $x_2(n) = \{1, 1, 1\}$

$$X_2(K) = \sum_{n=0}^{N-1} x_2(n) W_N^{Kn}$$

$W_3^0 = e^{-j\frac{2\pi}{3}.0} = 1$

$W_3^1 = e^{-j\frac{2\pi}{3}.1} = -0.5-j0.866$

$W_3^2 = e^{-j\frac{2\pi}{3}.2} = -0.5+j0.866$

Here N=3, $X_2(K) = \sum_{n=0}^{2} x_2(n) W_3^{Kn}$

$$X_2(K) = 1+W_3^K+W_3^{2K} \longrightarrow (1)$$

@K=0, $X_2(0) = 1+W_3^0+W_3^0 = 1+1+1 = 3$

@K=1, $X_2(1) = 1+W_3^1+W_3^2 = 1-0.5-j0.866-0.5+j0.866 = 0$

@K=2, $X_2(2) = 1+W_3^2+W_3^4 = 1-0.5+j0.866-0.5-j0.866 = 0$

$\langle X_2(K) = \{3, 0, 0\} \rangle$

c) After comparing the outputs of (a) and (b), we get to know that $x_1(n) \neq x_2(n)$.

16) Compute a) 6-point DFT of the sequence $x_1(n) = \{1, 0, 1, 0, 1, 0\}$

b) 4-point DFT of the sequence $x_2(n) = \{4, 3, 1, 0\}$

Sol^n:- Given $x_1(n) = \{1, 0, 1, 0, 1, 0\}$

$$X_1(K) = \sum_{n=0}^{N-1} x_1(n) W_N^{Kn}$$

Here N=6, $X_1(K) = \sum_{n=0}^{5} x_1(n) W_6^{Kn} = 1+W_6^{2K}+W_6^{4K}$

Scanned with CamScanner

@K=0, $X_1(0) = 1+W_6^0+W_6^0 = 1+1+1 = \underline{3}$

@K=1, $X_1(1) = 1+W_6^2+W_6^4 = 1-0.5-j0.866-0.5+j0.866$
$= \underline{0}$

@K=2, $X_1(2) = 1+W_6^4+W_6^8 = 1-0.5+j0.866-0.5-j0.866$
$= \underline{0}$

@K=3, $X_1(3) = 1+W_6^6+W_6^{12} = 1+1+1 = \underline{3}$

@K=4, $X_1(4) = 1+W_6^8+W_6^{16} = 1-0.5-j0.866-0.5+j0.866$
$= \underline{0}$

@K=5, $X_1(5) = 1+W_6^{10}+W_6^{20} = 1-0.5+j0.866-0.5-j0.866 = \underline{0}$

$\langle \therefore X_1(K) = \{3, 0, 0, 3, 0, 0\} \rangle$

$W_6^0 = 1$
$W_6^1 = 0.5-j0.866$
$W_6^2 = -0.5-j0.866$
$W_6^3 = -1$
$W_6^4 = -0.5+j0.866$
$W_6^5 = 0.5+j0.866$

b) Given $x_2(n) = \{4, 3, 1, 0\}$

$$X_2(K) = \sum_{n=0}^{N-1} x_2(n) W_N^{Kn}$$

$W_4^0 = 1$; $W_4^1 = -j$
$W_4^2 = -1$; $W_4^3 = j$

Here N=4, $X_2(K) = \sum_{n=0}^{3} x_2(n) W_4^{Kn} = 4+3W_4^K+W_4^{2K} \longrightarrow (1)$

@K=0, $X_2(0) = 4+3W_4^0+W_4^0 = 4+3+1 = \underline{8}$

@K=1, $X_2(1) = 4+3W_4^1+W_4^2 = 4+3(-j)-1 = \underline{3-j3}$

@K=2, $X_2(2) = 4+3W_4^2+W_4^4 = 4+3(-1)+1 = \underline{2}$

@K=3, $X_2(3) = 4+3W_4^3+W_4^6 = 4+3(j)-1 = \underline{3+j3}$

$\langle \therefore X_2(K) = \{8, 3-j3, 2, 3+j3\} \rangle$

16) Derive the DFT of the data sequence $x(n) = \{1, 1, 2, 2, 3, 3\}$ and Compute the corresponding amplitude and phase spectrum.

Sol^n:- $$X(K) = \sum_{K=0}^{N-1} x(n) W_N^{Kn}$$

Here N=6, $X(K) = \sum_{K=0}^{5} x(n) W_6^{Kn}$

$W_6^0 = 1$; $W_6^1 = 0.5-j0.866$
$W_6^2 = -0.5-j0.866$, $W_6^3 = -1$
$W_6^4 = -0.5+j0.866$, $W_6^5 = 0.5+j0.866$

Scanned with CamScanner

$X(K) = 1 + W_6^K + 2W_6^{2K} + 2W_6^{3K} + 3W_6^{4K} + 3W_6^{5K} \rightarrow (1)$

@K=0, $X(0) = 1 + W_6^0 + 2W_6^0 + 2W_6^0 + 3W_6^0 + 3W_6^0 = 1+1+2+2+3+3$

$= 12$

@K=1, $X(1) = 1 + W_6^1 + 2W_6^2 + 2W_6^3 + 3W_6^4 + 3W_6^5$

$= 1 + 0.5 - j0.866 + 2(-0.5 - j0.866) + 2(-1) + 3(-0.5 + j0.866) +$
$3(0.5 + j0.866) = -1.5 + j2.598$

@K=2, $X(2) = 1 + W_6^2 + 2W_6^4 + 2W_6^6 + 3W_6^8 + 3W_6^{10}$

$= 1 - 0.5 - j0.866 + 2(-0.5 + j0.866) + 2(1) + 3(-0.5 - j0.866)$
$+ 3(-0.5 + j0.866) = -1.5 + j0.866$

@K=3, $X(3) = 1 + W_6^3 + 2W_6^6 + 2W_6^9 + 3W_6^{12} + 3W_6^{15}$

$= 1 - 1 + 2(1) + 2(-1) + 3(1) + 3(-1) = 0$

@K=4, $X(4) = 1 + W_6^4 + 2W_6^8 + 2W_6^{12} + 3W_6^{16} + 3W_6^{20}$

$= 1 - 0.5 + j0.866 + 2(-0.5 - j0.866) + 2(1) + 3(-0.5 + j0.866)$
$+ 3(-0.5 - j0.866) = -1.5 - j0.866$

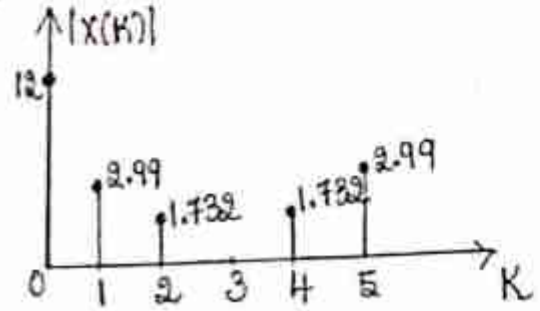
@K=5, $X(5) = 1 + W_6^5 + 2W_6^{10} + 2W_6^{15} + 3W_6^{20} + 3W_6^{25}$

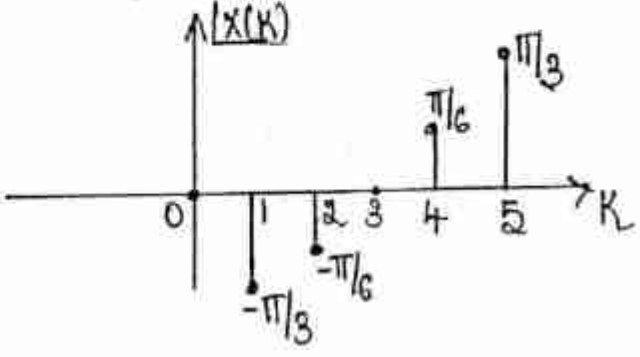
$= 1 + 0.5 + j0.866 + 2(-0.5 + j0.866) + 2(-1) + 3(-0.5 - j0.866)$
$+ 3(0.5 - j0.866) = -1.5 - j2.598$

$\therefore X(K) = \{12, -1.5 + j2.598, -1.5 + j0.866, 0, -1.5 - j0.866, -1.5 - j2.598\}$

Amplitude, $|X(K)| = \{12, 2.99, 1.732, 0, 1.732, 2.99\}$

phase, $\angle X(K) = \{0, -\pi/3, -\pi/6, 0, \pi/6, \pi/3\}$





Scanned with CamScanner

17) Compute the 6-point DFT of the sequence $x(n) = \{5, 4, 3, 2, 1, 0\}$ by using direct method.

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=6$,

$$X(K) = \sum_{n=0}^{5} x(n) W_6^{Kn}$$

$$X(K) = 5 + 4W_6^{K} + 3W_6^{2K} + 2W_6^{3K} + W_6^{4K} \longrightarrow (1)$$

$W_6^0 = 1$	$W_6^3 = -1$
$W_6^1 = 0.5 - j0.866$	$W_6^4 = -0.5 + j0.866$
$W_6^2 = -0.5 - j0.866$	$W_6^5 = 0.5 + j0.866$

@K=0, $X(0) = 5 + 4W_6^0 + 3W_6^0 + 2W_6^0 + W_6^0 = 5+4+3+2+1 = \underline{15}$

@K=1, $X(1) = 5 + 4W_6^1 + 3W_6^2 + 2W_6^3 + W_6^4$

$= 5 + 4(0.5 - j0.866) + 3(-0.5 - j0.866) + 2(-1) + (-0.5 + j0.866)$

$= \underline{3 - j5.196}$

@K=2, $X(2) = 5 + 4W_6^2 + 3W_6^4 + 2W_6^6 + W_6^8$

$= 5 + 4(-0.5 - j0.866) + 3(-0.5 + j0.866) + 2(1) - 0.5 - j0.866$

$= \underline{3 - j1.732}$

@K=3, $X(3) = 5 + 4W_6^3 + 3W_6^6 + 2W_6^9 + W_6^{12}$

$= 5 + 4(-1) + 3(1) + 2(-1) + 1 = \underline{3}$

@K=4, $X(4) = 5 + 4W_6^4 + 3W_6^8 + 2W_6^{12} + W_6^{16}$

$= 5 + 4(-0.5 + j0.866) + 3(-0.5 - j0.866) + 2(1) - 0.5 + j0.866$

$= \underline{3 + j1.732}$

@K=5, $X(5) = 5 + 4W_6^5 + 3W_6^{10} + 2W_6^{15} + W_6^{20}$

$= 5 + 4(0.5 + j0.866) + 3(-0.5 + j0.866) + 2(-1) - 0.5 - j0.866$

$= \underline{3 + j5.196}$

$\langle X(K) = \{15, 3 - j5.196, 3 - j1.732, 3, 3 + j1.732, 3 + j5.196\} \rangle$

Scanned with CamScanner

18) Determine the 8-point DFT of the sequence:
$x(n) = \{1, 1, 1, 1, 1, 1, 0, 0\}$

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N = 8$,

$$X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn}$$

$$X(K) = 1 + W_8^{K} + W_8^{2K} + W_8^{3K} + W_8^{4K} + W_8^{5K} \longrightarrow (1)$$

$W_8^0 = 1$	$W_8^4 = -1$
$W_8^1 = 0.7071 - j0.7071$	$W_8^5 = -0.7071 + j0.7071$
$W_8^2 = -j$	$W_8^6 = j$
$W_8^3 = -0.7071 - j0.7071$	$W_8^7 = 0.7071 + j0.7071$

@K=0, $X(0) = 1 + W_8^0 + W_8^0 + W_8^0 + W_8^0 + W_8^0 = 1+1+1+1+1+1 = \underline{6}$

@K=1, $X(1) = 1 + W_8^1 + W_8^2 + W_8^3 + W_8^4 + W_8^5$

$= 1 + 0.7071 - j0.7071 - j - 0.7071 - j0.7071 - 1 - 0.7071 + j0.7071$

$= \underline{-0.7071 - j1.7071}$

@K=2, $X(2) = 1 + W_8^2 + W_8^4 + W_8^6 + W_8^8 + W_8^{10}$

$= 1 - j - 1 + j + 1 - j = \underline{1 - j}$

@K=3, $X(3) = 1 + W_8^3 + W_8^6 + W_8^9 + W_8^{12} + W_8^{15} = 1 - 0.7071 - j0.7071 + j + 0.7071$
$- j0.7071 - 1 + 0.7071 + j0.7071 = \underline{0.7071 + j0.293}$

@K=4, $X(4) = 1 + W_8^4 + W_8^8 + W_8^{12} + W_8^{16} + W_8^{20}$

$= 1 - 1 + 1 - 1 + 1 - 1 = \underline{0}$

@K=5, $X(5) = 1 + W_8^5 + W_8^{10} + W_8^{15} + W_8^{20} + W_8^{25} = 1 - 0.7071 + j0.7071 - j +$
$0.7071 + j0.7071 - 1 + 0.7071 - j0.7071 = \underline{0.7071 - j0.293}$

@K=6, $X(6) = 1 + W_8^6 + W_8^{12} + W_8^{18} + W_8^{24} + W_8^{30} = 1 + j - 1 - j + 1 + j = \underline{1 + j}$

@K=7, $X(7) = 1 + W_8^7 + W_8^{14} + W_8^{21} + W_8^{28} + W_8^{35} = 1 + 0.7071 + j0.7071 + j +$
$-0.7071 + j0.7071 - 1 - 0.7071 - j0.7071 = \underline{-0.7071 + j0.7071}$

$\therefore X(K) = \{6, -0.7071 - j0.7071, 1 - j, 0.7071 + j0.293, 0, 0.7071 - j0.293,$
$1 + j, -0.7071 + j1.7071\}$

Scanned with CamScanner

19) Compute the 8-point DFT of the sequence $x(n) = \{1,1,1,1,0,0,0,0\}$ using direct method.

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=8$,

$$X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn}$$

$$X(K) = 1 + W_8^{K} + W_8^{2K} + W_8^{3K} \longrightarrow (1)$$

$W_8^0 = 1$
$W_8^1 = 0.7071 - j0.7071$
$W_8^2 = -j$
$W_8^3 = -0.7071 - j0.7071$
$W_8^4 = -1$
$W_8^5 = -0.7071 + j0.7071$
$W_8^6 = j$
$W_8^9 = 0.7071 + j0.7071$

@K=0, $X(0) = 1 + W_8^0 + W_8^0 + W_8^0 = 1+1+1+1 = \underline{4}$

@K=1, $X(1) = 1 + W_8^1 + W_8^2 + W_8^3 = 1 + 0.7071 - j0.7071 - j - 0.7071 - j0.7071$
$= \underline{1 - j2.4142}$

@K=2, $X(2) = 1 + W_8^2 + W_8^4 + W_8^6 = 1 - j - 1 + j = \underline{0}$

@K=3, $X(3) = 1 + W_8^3 + W_8^6 + W_8^9 = 1 - 0.7071 - j0.7071 + j + 0.7071 - j0.7071$
$= \underline{1 - j0.4142}$

@K=4, $X(4) = 1 + W_8^4 + W_8^8 + W_8^{12} = 1 - 1 + 1 - 1 = \underline{0}$

@K=5, $X(5) = 1 + W_8^5 + W_8^{10} + W_8^{15} = 1 - 0.7071 + j0.7071 - j + 0.7071 + j0.7071$
$= \underline{1 + j0.4142}$

@K=6, $X(6) = 1 + W_8^6 + W_8^{12} + W_8^{18} = 1 + j - 1 - j = \underline{0}$

@K=7, $X(7) = 1 + W_8^7 + W_8^{14} + W_8^{21} = 1 + 0.7071 + j0.7071 + j - 0.7071 + j0.7071$
$= \underline{1 + j2.4142}$

$\therefore X(K) = \{4, 1-j2.4142, 0, 1-j0.4142, 0, 1+j0.4142, 0, 1+j2.4142\}$

20) Compute the 8-point DFT of the sequence $x(n) = \{1, 4, 1, 0, 1, 0, 0, 0\}$ by direct method.

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn} = 1 + 4W_8^{K} + W_8^{2K} + W_8^{4K} \longrightarrow (1)$

Scanned with CamScanner

@K=0, $X(0) = 1+4W_8^0 + W_8^0 + W_8^0 = 1+4+1+1 = 7$

@K=1, $X(1) = 1+4W_8^1 + W_8^2 + W_8^4 = 1+4(0.7071-j0.7071) - j + 1$

$= 2.8284 - j2.8284$

@K=2, $X(2) = 1+4W_8^2 + W_8^4 + W_8^6 = 1+4(-j) - 1 + 1 = 1-j4$

@K=3, $X(3) = 1+4W_8^3 + W_8^6 + W_8^{12} = 1+4(-0.7071-j0.7071) + j - 1$

$= -2.8284 - j1.2284$

@K=4, $X(4) = 1+4W_8^4 + W_8^8 + W_8^{16} = 1+4(-1) + 1 + 1 = -1$

@K=5, $X(5) = 1+4W_8^5 + W_8^{10} + W_8^{20} = 1+4(-0.7071+j0.7071) - j - 1$

$= -2.8284 + j1.2284$

@K=6, $X(6) = 1+4W_8^8 + W_8^{12} + W_8^{24} = 1+4(j) - 1 + 1 = 1+j4$

@K=7, $X(7) = 1+4W_8^7 + W_8^{14} + W_8^{28} = 1+4(0.7071+j0.7071) + j - 1$

$= 2.8284 + j2.8284$

$\therefore X(K) = \{7, 2.8284 - j2.8284, 1-j4, -2.8284 - j1.2284, -1, -2.8284 + j1.2284, 1+j4, 2.8284 + j2.8284\}$

2) Determine DFT of $x(n) = \{1, 0, 1, 0, 1, 0, 1, 0\}$

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here N = 8, $X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn}$

$W_8^0 = 1$	$W_8^4 = -1$
$W_8^1 = 0.7071 - j0.7071$	$W_8^5 = -0.7071 + j0.7071$
$W_8^2 = -j$	$W_8^6 = j$
$W_8^3 = -0.7071 - j0.7071$	$W_8^7 = 0.7071 + j0.7071$

$X(K) = 1 + W_8^{2K} + W_8^{4K} + W_8^{6K} \longrightarrow (1)$

@K=0, $X(0) = 1+W_8^0 + W_8^0 + W_8^0 = 1+1+1+1 = 4$

@K=1, $X(1) = 1+W_8^2 + W_8^4 + W_8^6 = 1-j-1+j = 0$

@K=2, $X(2) = 1+W_8^4 + W_8^8 + W_8^{12} = 1-1+1-1 = 0$

@K=3, $X(3) = 1+W_8^6 + W_8^{12} + W_8^{18} = 1+j-1-j = 0$

@K=4, $X(4) = 1+W_8^8 + W_8^{16} + W_8^{24} = 1+1+1+1 = 4$

Scanned with CamScanner

@K=5, $X(5) = 1 + W_8^{10} + W_8^{20} + W_8^{30} = 1 - j - 1 + j = \underline{0}$

@K=6, $X(6) = 1 + W_8^{12} + W_8^{24} + W_8^{36} = 1 - 1 + 1 - 1 = \underline{0}$

@K=7, $X(7) = 1 + W_8^{14} + W_8^{28} + W_8^{42} = 1 + j - 1 - j = \underline{0}$

$\therefore X(K) = \{4, 0, 0, 0, 4, 0, 0, 0\}$

22) Find the DFT of the sequence $x(n) = \begin{cases} 1 ; 0 \le n \le L \\ 0 ; \text{otherwise} \end{cases}$ for N=4. plot magnitude and phase of X(K).

Soln:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

$W_4^0 = 1$	$W_4^2 = -1$
$W_4^1 = -j$	$W_4^3 = j$

Here $L = N = 4$, $x(n) = \{1, 1, 1, 1\}$

$$X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn} = 1 + W_4^K + W_4^{2K} + W_4^{3K} \longrightarrow (1)$$

@K=0, $X(0) = 1 + W_4^0 + W_4^0 + W_4^0 = 1 + 1 + 1 + 1 = \underline{4}$

@K=1, $X(1) = 1 + W_4^1 + W_4^2 + W_4^3 = 1 - j - 1 + j = \underline{0}$

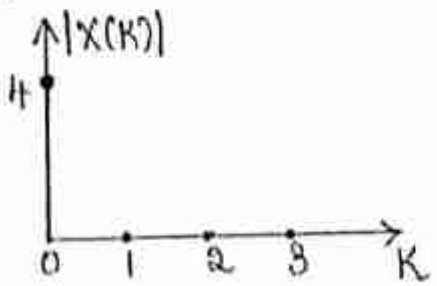
@K=2, $X(2) = 1 + W_4^2 + W_4^4 + W_4^6 = 1 - 1 + 1 - 1 = \underline{0}$

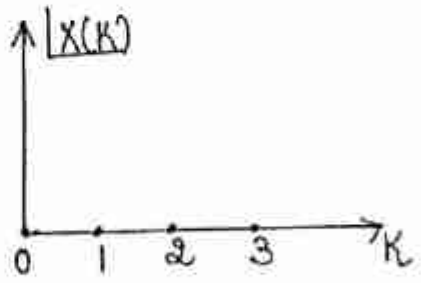
@K=3, $X(3) = 1 + W_4^3 + W_4^6 + W_4^9 = 1 + j - 1 - j = \underline{0}$

$\therefore X(K) = \{4, 0, 0, 0\}$

magnitude, $|X(K)| = \{4, 0, 0, 0\}$

phase, $\underline{|X(K)|} = \{0, 0, 0, 0\}$





Scanned with CamScanner

23) Determine DFT of the sequence $x(n) = \begin{cases} 1/3 & ; 0 \le n \le 2 \\ 0 & ; \text{otherwise} \end{cases}$

plot magnitude and phase spectrum for each case:
a) N = 4 b) N = 6 c) N = 8

Sol^n:- Given $x(n) = \frac{1}{3}$; $0 \le n \le 2$

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$; $0 \le K \le N-1$

a) When N = 4,

$x(n) = \{1/3, 1/3, 1/3, 0\}$ ← Zero-padding to increase the length to '4'.

$\therefore X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn} = \frac{1}{3} + \frac{1}{3} W_4^{K} + \frac{1}{3} W_4^{2K} \longrightarrow (1)$

@K=0, $X(0) = \frac{1}{3} + \frac{1}{3} W_4^0 + \frac{1}{3} W_4^0 = \frac{1}{3} + \frac{1}{3} + \frac{1}{3} = \underline{1}$

@K=1, $X(1) = \frac{1}{3} + \frac{1}{3} W_4^1 + \frac{1}{3} W_4^2 = \frac{1}{3} + \frac{1}{3}(-j) + \frac{1}{3}(-1) = \underline{-j\left(\frac{1}{3}\right)}$

@K=2, $X(2) = \frac{1}{3} + \frac{1}{3} W_4^2 + \frac{1}{3} W_4^4 = \frac{1}{3} + \frac{1}{3}(-1) + \frac{1}{3}(1) = \underline{\frac{1}{3}}$

@K=3, $X(3) = \frac{1}{3} + \frac{1}{3} W_4^3 + \frac{1}{3} W_4^6 = \frac{1}{3} + \frac{1}{3}(j) + \frac{1}{3}(-1) = \underline{j\left(\frac{1}{3}\right)}$

$\therefore X(K) = \{1, -j(1/3), 1/3, j(1/3)\}$

magnitude, $|X(K)| = \{1, 1/3, 1/3, 1/3\}$

phase, $\underline{/X(K)} = \{0, -\pi/2, 0, \pi/2\}$

b) When N = 6,

$x(n) = \{1/3, 1/3, 1/3, 0, 0, 0\}$ ← Zero padding

$X(K) = \sum_{n=0}^{5} x(n) W_6^{Kn} = \frac{1}{3} + \frac{1}{3} W_6^{K} + \frac{1}{3} W_6^{2K} = \frac{1}{3}\left[1 + W_6^{K} + W_6^{2K}\right] \longrightarrow (1)$

Scanned with CamScanner

@K=0, $X(0) = \frac{1}{3}(1+W_6^0+W_6^0) = \frac{1}{3}(1+1+1) = 1$

@K=1, $X(1) = \frac{1}{3}(1+W_6^1+W_6^2) = \frac{1}{3}[1+0.5-j0.866-0.5-j0.866]$

$= 0.333 - j0.5773$

@K=2, $X(2) = \frac{1}{3}(1+W_6^2+W_6^4) = \frac{1}{3}[1-0.5-j0.866-0.5+j0.866]$

$= 0$

@K=3, $X(3) = \frac{1}{3}(1+W_6^3+W_6^6) = \frac{1}{3}[1-1+1] = 1/3$

@K=4, $X(4) = \frac{1}{3}(1+W_6^4+W_6^8) = \frac{1}{3}[1-0.5+j0.866-0.5-j0.866] = 0$

@K=5, $X(5) = \frac{1}{3}(1+W_6^5+W_6^{10}) = \frac{1}{3}[1+0.5+j0.866-0.5+j0.866]$

$= 0.333+j0.5773$

$\therefore X(K) = \{1, 0.333-j0.5773, 0, 1/3, 0, 0.333+j0.5773\}$

$|X(K)| = \{1, 0.66, 0, 1/3, 0, 0.66\}$

$\angle X(K) = \{0, -60°, 0, 0, 0, 60°\}$

c) When N=8,

$X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn} = \frac{1}{3} + \frac{1}{3}W_8^K + \frac{1}{3}W_8^{2K} = \frac{1}{3}(1+W_8^K+W_8^{2K}) \longrightarrow (1)$

@K=0, $X(0) = \frac{1}{3}(1+W_8^0+W_8^0) = \frac{1}{3}(1+1+1) = 1$

@K=1, $X(1) = \frac{1}{3}(1+W_8^1+W_8^2) = \frac{1}{3}(1+0.7071-j0.7071-j) = 0.569-j0.569$

@K=2, $X(2) = \frac{1}{3}(1+W_8^2+W_8^4) = \frac{1}{3}[1-j-1] = -j\frac{1}{3}$

@K=3, $X(3) = \frac{1}{3}(1+W_8^3+W_8^6) = \frac{1}{3}[1-0.7071-j0.7071+j] = 0.0976+j0.098$

@K=4, $X(4) = \frac{1}{3}(1+W_8^4+W_8^8) = \frac{1}{3}[1-1+1] = 1/3$

@K=5, $X(5) = \frac{1}{3}(1+W_8^5+W_8^{10}) = \frac{1}{3}[1-0.7071+j0.7071-j] = 0.0976-j0.098$

Scanned with CamScanner

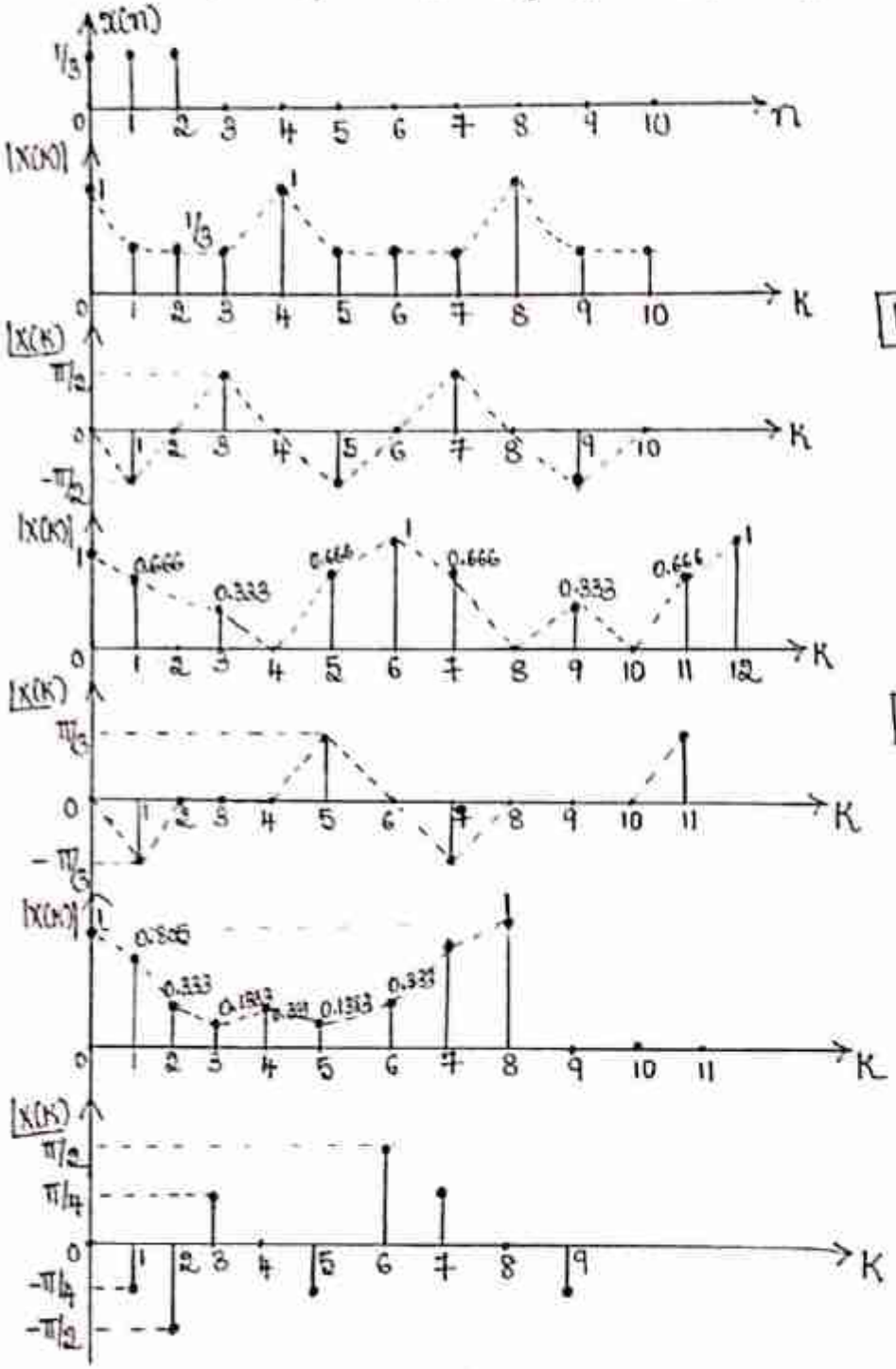
@K=6, $X(6) = \frac{1}{3}(1+W_8^6+W_8^{12}) = \frac{1}{3}[1+j-1] = \underline{j(1/3)}$

@K=7, $X(7) = \frac{1}{3}(1+W_8^7+W_8^{14}) = \frac{1}{3}[1+0.7071+j0.7071-1] = \underline{0.569+j0.569}$

$X(K) = \{1, 0.569-j0.569, -j1/3, 0.0976+j0.098, 1/3, +j1/3, 0.569+j0.569\}$

$|X(K)| = \{1, 0.805, 0.333, 0.1388, 0.333, 0.1888, 0.333, 0.805\}$

$\angle X(K) = \{0, -\pi/4, -\pi/2, \pi/4, 0, -\pi/4, \pi/2, \pi/4\}$



Scanned with CamScanner

Conclusion :-

From the zero points, it is difficult to extrapolate. Entire frequency spectrum i.e., resolution of the spectrum is very poor. In order to increase the resolution, compute DFT for more points. Therefore, the zero padding gives a high density spectrum and provides a better display.

"The procedure of adding zeros to discrete-time sequence $x(n)$ to increase the length of $x(n)$ is called as "Zero padding". Zero padding will not give any additional information about $X(w)$ of $x(n)$. However, padding $(N-L)$ zeros in sequence $x(n)$. N-point DFT length is increased for better display of $X(w)$ and frequency spacing is decreased.

2) Find the DFT of a sequence $x(n) = \begin{cases} 1 & ; 0 \le n \le 2 \\ 0 & ; \text{else} \end{cases}$ for

i) $N=4$ ii) $N=8$. plot $|X(K)|$ and $\angle X(K)$.

Soln :-

i) For $N=4$, $x(n) = \{1, 1, 1, 0\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} = \sum_{n=0}^{3} x(n) W_4^{Kn} = 1 + W_4^K + W_4^{2K} \longrightarrow (1)$$

@K=0, $X(0) = 1 + W_4^0 + W_4^0 = 1+1+1 = 3$

@K=1, $X(1) = 1 + W_4^1 + W_4^2 = 1 - j + 1 = -j$

@K=2, $X(2) = 1 + W_4^2 + W_4^4 = 1 - 1 + 1 = 1$

@K=3, $X(3) = 1 + W_4^3 + W_4^6 = 1 + j - 1 = j$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

$\therefore X(K) = \{3, -j, 1, j\}$

$|X(K)| = \{3, 1, 1, 1\}$

$\angle X(K) = \{0, -\pi/2, 0, \pi/2\}$

Scanned with CamScanner

ii) For $N=8$, $x(n)=\{1,1,1,0,0,0,0,0\}$

$$X(K)=\sum_{n=0}^{N-1} x(n)W_N^{Kn}=\sum_{n=0}^{7} x(n)W_8^{Kn}=1+W_8^{K}+W_8^{2K} \longrightarrow (1)$$

@$K=0$, $X(0)=1+W_8^0+W_8^0=1+1+1=\underline{3}$

@$K=1$, $X(1)=1+W_8^1+W_8^2=1+0.7071-j0.7071-j=\underline{1.7071-j1.7071}$

@$K=2$, $X(2)=1+W_8^2+W_8^4=1-j-1=\underline{-j}$

@$K=3$, $X(3)=1+W_8^3+W_8^6=1-0.7071-j0.7071+j=\underline{0.2929+j0.2929}$

@$K=4$, $X(4)=1+W_8^4+W_8^8=1-1+1=\underline{1}$

@$K=5$, $X(5)=1+W_8^5+W_8^{10}=1-0.7071+j0.7071-j=\underline{0.2929-j0.2929}$

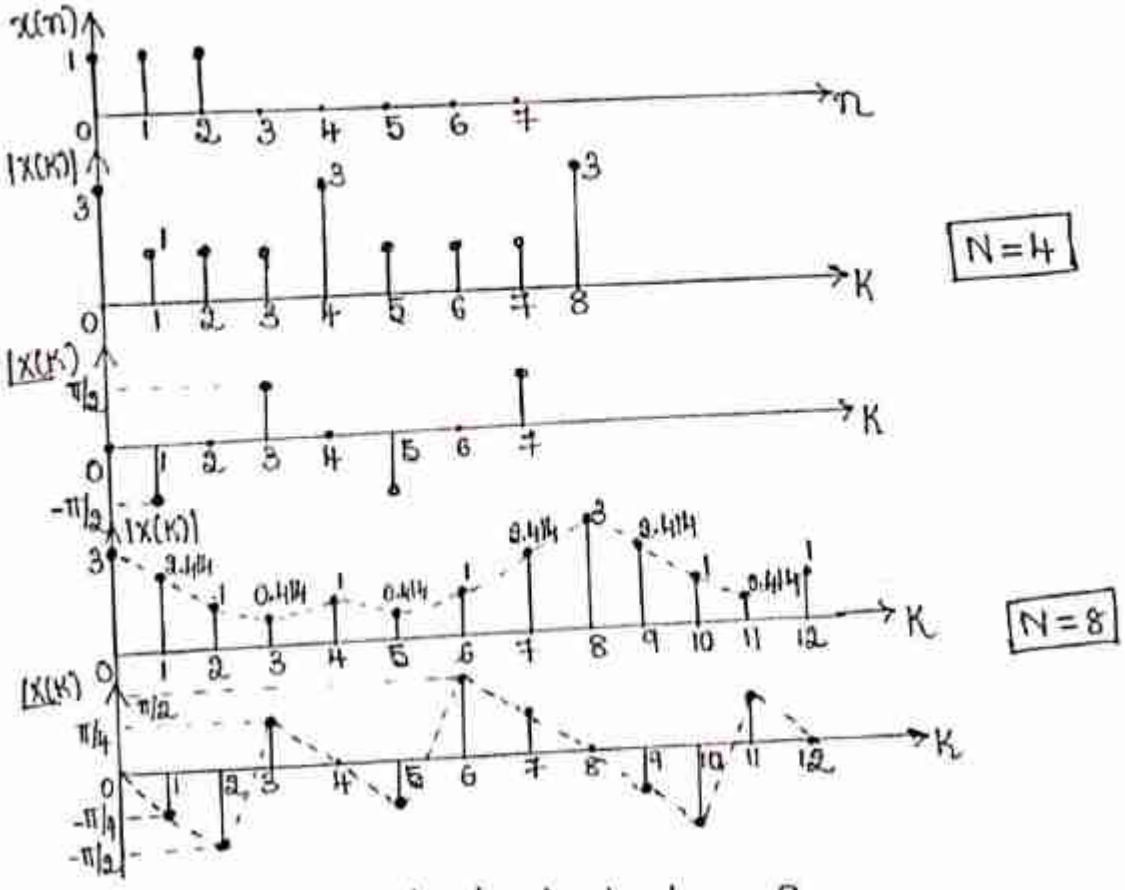
@$K=6$, $X(6)=1+W_8^6+W_8^{12}=1+j-1=\underline{j}$

@$K=7$, $X(7)=1+W_8^7+W_8^{14}=1+0.7071+j0.7071+j=\underline{1.7071+j1.7071}$

$X(K)=\{3, 1.7071-j1.7071, -j, 0.2929+j0.2929, 1, 0.2929-j0.2929, j, 1.7071+j1.7071\}$

$|X(K)|=\{3, 2.4142, 1, 0.4142, 1, 0.4142, 1, 2.4142\}$

$\angle X(K)=\{0, -\pi/4, -\pi/2, \pi/4, 0, -\pi/4, \pi/2, \pi/4\}$



Scanned with CamScanner

25) Determine the 8-point DFT of the sequence $x(n)=\{8,8,8,8,1,1,0,0\}$. Sketch its magnitude and phase spectra.

Sol^n:- $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=8$, $X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn} = 8 + 8W_8^K + 8W_8^{2K} + 8W_8^{3K} + W_8^{4K} + W_8^{5K} \longrightarrow (1)$

@$K=0$, $X(0) = 8 + 8W_8^0 + 8W_8^0 + 8W_8^0 + W_8^0 + W_8^0 = 8+8+8+8+1+1 = \underline{34}$

@$K=1$, $X(1) = 8 + 8W_8^1 + 8W_8^2 + 8W_8^3 + W_8^4 + W_8^5$

$= 8 + 8(0.7071 - j0.7071) + 8(-j) + 8(-0.7071 - j0.7071) - 1 + (-0.7071 + j0.7071) = \underline{6.293 - j18.61}$

@$K=2$, $X(2) = 8 + 8W_8^2 + 8W_8^4 + 8W_8^6 + W_8^8 + W_8^{10}$

$= 8 + 8(-j) + 8(-1) + 8(j) + 1 - j = \underline{1 - j}$

@$K=3$, $X(3) = 8 + 8W_8^3 + 8W_8^6 + 8W_8^9 + W_8^{12} + W_8^{15}$

$= 8 + 8(-0.7071 - j0.7071) + 8(j) + 8(0.7071 - j0.7071) + (-1) + (0.7071 + j0.7071) = \underline{7.71 - j2.61}$

@$K=4$, $X(4) = 8 + 8W_8^4 + 8W_8^8 + 8W_8^{12} + W_8^{16} + W_8^{20}$

$= 8 + 8(-1) + 8(1) + 8(-1) + 1 - 1 = \underline{0}$

@$K=5$, $X(5) = 8 + 8W_8^5 + 8W_8^{10} + 8W_8^{15} + W_8^{20} + W_8^{25}$

$= 8 + 8(-0.7071 + j0.7071) + 8(-j) + 8(0.7071 + j0.7071) + (-1) + (0.7071 - j0.7071) = \underline{7.71 + j2.61}$

@$K=6$, $X(6) = 8 + 8W_8^6 + 8W_8^{12} + 8W_8^{18} + W_8^{24} + W_8^{30} = \underline{1 + j}$

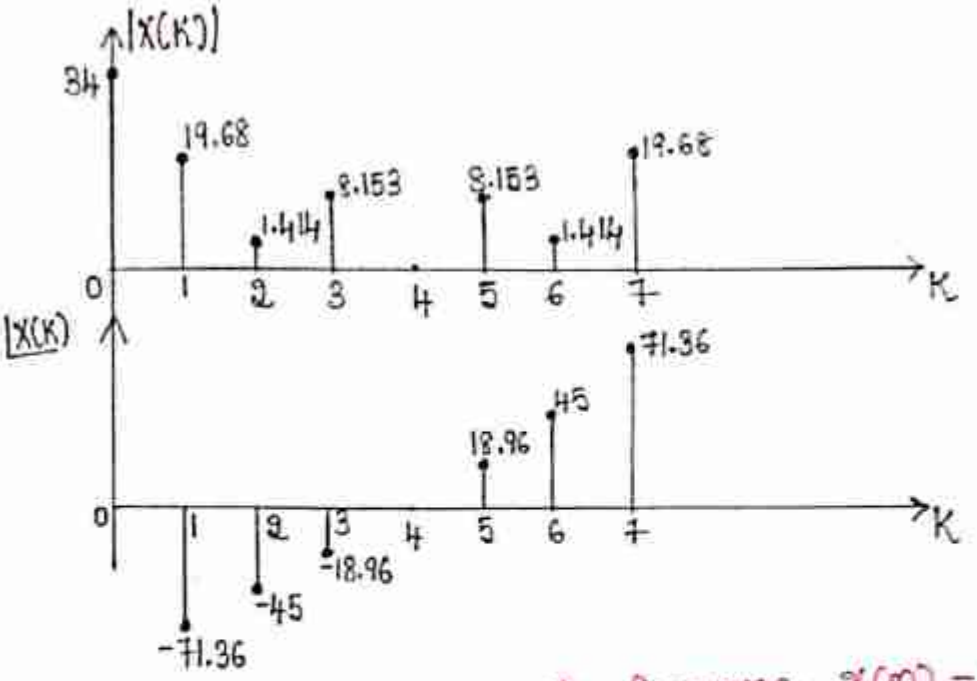
@$K=7$, $X(7) = 8 + 8W_8^7 + 8W_8^{14} + 8W_8^{21} + W_8^{28} + W_8^{35} = \underline{6.293 + j18.61}$

$\therefore X(K) = \{34, 6.293 - j18.61, 1 - j, 7.71 - j2.61, 0, 7.71 + j2.61, 1 + j, 6.293 + j18.61\}$

$|X(K)| = \{34, 19.68, 1.414, 8.153, 0, 8.153, 1.414, 19.68\}$

$\underline{|X(K)|} = \{0, -71.36, -45, -18.96, 0, 18.96, 45, 71.36\}$

Scanned with CamScanner



26) Compute 8-point DFT of the sequence $x(n) = (-1)^{n+1}$; $0 \le n \le 7$.
Also plot the magnitude of DFT.

Sol^n:- $x(n) = \{-1, 1, -1, 1, -1, 1, -1, 1\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

Here $N = 8$,

$$X(K) = \sum_{n=0}^{7} x(n) W_8^{Kn} = -1 + W_8^{K} - W_8^{2K} + W_8^{3K} - W_8^{4K} + W_8^{5K} - W_8^{6K} + W_8^{7K} \longrightarrow (1)$$

@K=0, $X(0) = -1 + W_8^0 - W_8^0 + W_8^0 - W_8^0 + W_8^0 - W_8^0 + W_8^0 = -1 + 1 - 1 + 1 - 1 + 1 - 1 + 1$
$= 0$

@K=1, $X(1) = -1 + W_8^1 - W_8^2 + W_8^3 - W_8^4 + W_8^5 - W_8^6 + W_8^7$
$= -1 + 0.7071 - j0.7071 + j - 0.7071 - j0.7071 + 1 - 0.7071 + j0.7071 - j$
$+ 0.7071 + j0.7071 = 0$

@K=2, $X(2) = -1 + W_8^2 - W_8^4 + W_8^6 - W_8^8 + W_8^{10} - W_8^{12} + W_8^{14}$
$= -1 - j + 1 + j - 1 - j + 1 + j = 0$

@K=3, $X(3) = -1 + W_8^3 - W_8^6 + W_8^9 - W_8^{12} + W_8^{15} - W_8^{18} + W_8^{21}$
$= -1 - 0.7071 - j0.7071 - j + 0.7071 - j0.7071 + 1 + 0.7071 + j0.7071$
$+ j - 0.7071 + j0.7071 = 0$

@K=4, $X(4) = -1 + W_8^4 - W_8^8 + W_8^{12} - W_8^{16} + W_8^{20} - W_8^{24} + W_8^{28}$
$= -1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 = -8$

Scanned with CamScanner

@K=5, $X(5) = -1 + W_8^5 - W_8^{10} + W_8^{15} - W_8^{20} + W_8^{25} - W_8^{30} + W_8^{35}$

$= -1 - 0.707 + j0.707 + j + 0.707 + j0.707 + 1 + 0.707 - j0.707 - j$

$-0.707 - j0.707 = \underline{\underline{0}}$

@K=6, $X(6) = -1 + W_8^6 - W_8^{12} + W_8^{18} - W_8^{24} + W_8^{30} - W_8^{36} + W_8^{42}$

$= -1 + j + 1 - j - 1 + j + 1 - j = \underline{\underline{0}}$

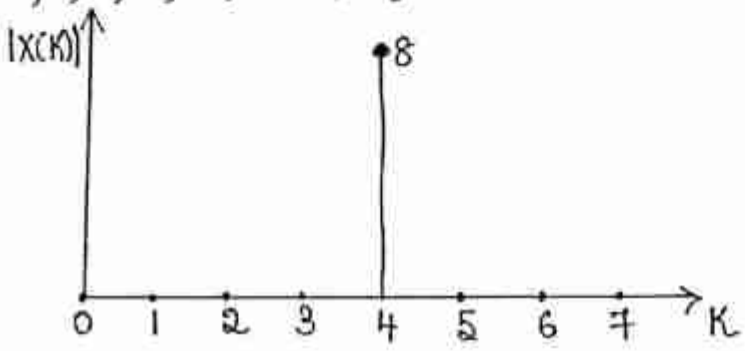
@K=7, $X(7) = -1 + W_8^7 - W_8^{14} + W_8^{21} - W_8^{28} + W_8^{35} - W_8^{42} + W_8^{49}$

$= -1 + 0.707 + j0.707 - j - 0.707 + j0.707 + 1 - 0.707 - j0.707 + j$

$+0.707 - j0.707 = \underline{\underline{0}}$

$X(K) = \{0, 0, 0, 0, -8, 0, 0, 0\}$

$|X(K)| = \{0, 0, 0, 0, 8, 0, 0, 0\}$



Scanned with CamScanner

1) Determine IDFT of the sequence : $X(K) = \{10, -2+j2, -2, -2-j2\}$.

Sol^n:- $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

Here $N=4$, $x(n) = \frac{1}{4}\sum_{K=0}^{3} X(K) W_4^{-Kn}$

$W_4^{-0} = e^{\frac{j2\pi}{4}.0} = 1$
$W_4^{-1} = e^{\frac{j2\pi}{4}.1} = j$
$W_4^{-2} = -1$; $W_4^{-3} = -j$

$= \frac{1}{4}\left[X(0)W_4^{-0} + X(1)W_4^{-n} + X(2)W_4^{-2n} + X(3)W_4^{-3n}\right]$

$x(n) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-n} - 2W_4^{-2n} + (-2-j2)W_4^{-3n}\right] \longrightarrow (1)$

@$n=0$, $x(0) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-0} - 2W_4^{-0} + (-2-j2)W_4^{-0}\right] = \frac{1}{4}[10-2+j2-2-2-j2]$

$= \underline{1}$

@$n=1$, $x(1) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-1} - 2W_4^{-2} + (-2-j2)W_4^{-3}\right]$

$= \frac{1}{4}\left[10 + (-2+j2)(j) - 2(-1) + (-2-j2)(-j)\right] = \underline{2}$

@$n=2$, $x(2) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-2} - 2W_4^{-4} + (-2-j2)W_4^{-6}\right]$

$= \frac{1}{4}\left[10 + (-2+j2)(-1) - 2(1) + (-2-j2)(-1)\right] = \underline{3}$

@$n=3$, $x(3) = \frac{1}{4}\left[10 + (-2+j2)W_4^{-3} - 2W_4^{-6} + (-2-j2)W_4^{-9}\right]$

$= \frac{1}{4}\left[10 + (-2+j2)(-j) + (-2) + (-2-j2)(j)\right] = \underline{4}$

$\therefore x(n) = \{1, 2, 3, 4\}$

2) Determine IDFT of the sequence $X(K) = \{4, -j2, 0, j2\}$

Sol^n:- $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

$W_4^{-0} = 1$; $W_4^{-1} = j$
$W_4^{-2} = -1$; $W_4^{-3} = -j$

Here $N=4$, $x(n) = \frac{1}{4}\sum_{K=0}^{3} X(K) W_4^{-Kn}$

$= \frac{1}{4}\left[4 - j2.W_4^{-n} + j2.W_4^{-3n}\right]$

Scanned with CamScanner

@n=0, $x(0) = \frac{1}{4}[4 + (-j2)W_4^{-0} + (j2)W_4^{-0}] = \frac{1}{4}[4 - j2 + j2] = \underline{1}$

@n=1, $x(1) = \frac{1}{4}[4 + (-j2)W_4^{-1} + (j2)W_4^{-3}] = \frac{1}{4}[4 - j2(j) + j2(-j)] = \underline{2}$

@n=2, $x(2) = \frac{1}{4}[4 - j2W_4^{-2} + j2W_4^{-6}] = \frac{1}{4}[4 - j2(-1) + j2(-1)] = \underline{1}$

@n=3, $x(3) = \frac{1}{4}[4 - j2(W_4^{-3}) + j2W_4^{-9}] = \frac{1}{4}[4 - j2(-j) + j2(j)] = \underline{0}$

$\therefore x(n) = \{1, 2, 1, 0\}$

3) Determine IDFT of the sequence $X(K) = \{2, 1+j, 0, 1-j\}$

Sol^n:- $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K)W_N^{-Kn}$

$W_4^0 = 1$; $W_4^{-1} = j$

$W_4^{-2} = -1$; $W_4^{-3} = -j$

Here N=4, $x(n) = \frac{1}{4}\sum_{K=0}^{3} X(K)W_4^{-Kn}$

$x(n) = \frac{1}{4}[2 + (1+j)W_4^{-n} + (1-j)W_4^{-3n}] \longrightarrow (1)$

@n=0, $x(0) = \frac{1}{4}[2 + (1+j)W_4^{-0} + (1-j)W_4^{-0}] = \frac{1}{4}[2+1+j+1-j] = \underline{1}$

@n=1, $x(1) = \frac{1}{4}[2 + (1+j)W_4^{-1} + (1-j)W_4^{-3}] = \frac{1}{4}[2 + (1+j)(j) + (1-j)(-j)]$

$x(1) = 0$

@n=2, $x(2) = \frac{1}{4}[2 + (1+j)W_4^{-2} + (1-j)W_4^{-6}] = \frac{1}{4}[2 + (1+j)(-1) + (1-j)(-1)]$

$x(2) = 0$

@n=3, $x(3) = \frac{1}{4}[2 + (1+j)W_4^{-3} + (1-j)W_4^{-9}] = \frac{1}{4}[2 + (1+j)(-j) + (1-j)(j)]$

$x(3) = 1$

$\therefore x(n) = \{1, 0, 0, 1\}$

Scanned with CamScanner

4) Find 8-point IDFT for the sequence:

$X(K) = \{5, 0, 1-j, 0, 1, 0, 1+j, 0\}$.

Soln:- $x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

Here $N=8$, $x(n) = \frac{1}{8} \sum_{K=0}^{7} X(K) W_8^{-Kn}$

$$x(n) = \frac{1}{8}\left[5 + (1-j)W_8^{-2n} + W_8^{-4n} + (1+j)W_8^{-6n}\right] \longrightarrow (1)$$

@ $n=0$, $x(0) = \frac{1}{8}[5+1-j+1+1+j] = \underline{1}$

@ $n=1$, $x(1) = \frac{1}{8}\left[5 + (1-j)W_8^{-2} + W_8^{-4} + (1+j)W_8^{-6}\right]$

$= \frac{1}{8}[5 + (1-j)(j) + (-1) + (1+j)(-j)] = \underline{3/4}$

@ $n=2$, $x(2) = \frac{1}{8}\left[5 + (1-j)W_8^{-4} + W_8^{-8} + (1+j)W_8^{-12}\right]$

$= \frac{1}{8}[5 + (1-j)(-1) + 1 + (1+j)(-1)] = \underline{1/2}$

@ $n=3$, $x(3) = \frac{1}{8}\left[5 + (1-j)W_8^{-6} + W_8^{-12} + (1+j)W_8^{-18}\right]$

$= \frac{1}{8}[5 + (1-j)(-j) + (-1) + (1+j)(j)] = \underline{1/4}$

@ $n=4$, $x(4) = \frac{1}{8}\left[5 + (1-j)W_8^{-8} + W_8^{-16} + (1+j)W_8^{-24}\right]$

$= \frac{1}{8}[5 + (1-j)(1) + 1 + (1+j)(1)] = \underline{1}$

@ $n=5$, $x(5) = \frac{1}{8}\left[5 + (1-j)W_8^{-10} + W_8^{-20} + (1+j)W_8^{-30}\right]$

$= \frac{1}{8}[5 + (1-j)(j) + (-1) + (1+j)(-j)] = \underline{3/4}$

@ $n=6$, $x(6) = \frac{1}{8}\left[5 + (1-j)W_8^{-12} + W_8^{-24} + (1+j)W_8^{-36}\right]$

$= \frac{1}{8}[5 + (1-j)(-1) + 1 + (1+j)(-1)] = \underline{1/2}$

Scanned with CamScanner

@n=7,

$$x(n) = \frac{1}{8}\left[5 + (1-j)W_8^{-14} + W_8^{-28} + (1+j)W_8^{-42}\right]$$

$$= \frac{1}{8}\left[5 + (1-j)(-j) + (-1) + (1+j)(j)\right] = \underline{1/4}$$

$\therefore x(n) = \{1, 3/4, 1/2, 1/4, 1, 3/4, 1/2, 1/4\}$

5) Find the IDFT of $X(K) = \{255, 48.63+j166.05, -51+j102, -78.63+j46.05, -85, -78.63-j46.05, -51-j102, 48.63-j166.05\}$

Sol^n:- $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

Here N=8,

$$x(n) = \frac{1}{8}\sum_{K=0}^{7} X(K) W_8^{-Kn}$$

$$= \frac{1}{8}\Big[255 + (48.63+j166.05)W_8^{-n} + (-51+j102)W_8^{-2n} + (-78.63+j46.05)W_8^{-3n} + (-85)W_8^{-4n} + (-78.63-j46.05)W_8^{-5n} + (-51-j102)W_8^{-6n} + (48.63-j166.05)W_8^{-7n}\Big] \longrightarrow (1)$$

(W_8^{-n}: 45°; W_8^{-2n}: 90°; W_8^{-3n}: 135°; W_8^{-4n}: 180°; W_8^{-5n}: 225°; W_8^{-6n}: 270°; W_8^{-7n}: 315°)

@n=0, $x(0) = \frac{1}{8}[252.63 + j268.05 - 242.26 - 2.37 - j268.05] = \underline{1}$

@n=1, $x(1) = \frac{1}{8}[69.97 + j100.8 + 131.075 - 185.03 - j100.8] = \underline{2}$

@n=2, $x(2) = \frac{1}{8}[139.95 - j53.37 + 7.1 - 115.05 + j53.37] = \underline{4}$

@n=3, $x(3) = \frac{1}{8}[205.19 - j32.03 - 91.32 - 49.8 + j32.03] = \underline{8}$

@n=4, $x(4) = \frac{1}{8}[155.37 - j64.05 + 72.26 - 99.63 + j64.05] = \underline{16}$

@n=5, $x(5) = \frac{1}{8}[236.03 - j202.8 - 138.925 - 18.97 + j202.8] = \underline{22}$

@n=6, $x(6) = \frac{1}{8}[472.05 - j150.63 - 177.1 + 217.05 + j150.63] = \underline{64}$

@n=7, $x(7) = \frac{1}{8}[508.8 + j134.03 + 261.32 + 253.8 - j134.03] = \underline{128}$

$\therefore x(n) = \{1, 2, 4, 8, 16, 22, 64, 128\}$

Scanned with CamScanner

1) Consider, a signal of length equal to 4 defined by $x(n) = \{1, 2, 3, 1\}$
a) Compute 4-point DFT by solving explicitly 4-by-4 system of linear equations defined by the inverse DFT formula.
b) Verify the result of part (a) by finding the $X(K)$ using the defined equation by DFT.

Soln:-

a) IDFT : $x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

$$\sum_{K=0}^{N-1} X(K) W_N^{-Kn} = N\, x(n)$$

Here $N=4$, $\sum_{K=0}^{3} X(K) W_4^{-Kn} = 4\, x(n)$; $n = 0, 1, 2, 3$

@ $n=0$, $X(0)W_4^{-0} + X(1)W_4^{-0} + X(2)W_4^{-0} + X(3)W_4^{-0} = 4x(0) = 4 \times 1 = \underline{4}$

@ $n=1$, $X(0)W_4^{-0} + X(1)W_4^{-1} + X(2)W_4^{-2} + X(3)W_4^{-3} = 4x(1) = 4 \times 2 = \underline{8}$

@ $n=2$, $X(0)W_4^{-0} + X(1)W_4^{-2} + X(2)W_4^{-4} + X(3)W_4^{-6} = 4x(2) = 4 \times 3 = \underline{12}$

@ $n=3$, $X(0)W_4^{-0} + X(1)W_4^{-3} + X(2)W_4^{-6} + X(3)W_4^{-9} = 4x(3) = 4 \times 1 = \underline{4}$

The above equations are linear equations,

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} \begin{bmatrix} W_4^{-0} & W_4^{-0} & W_4^{-0} & W_4^{-0} \\ W_4^{-0} & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ W_4^{-0} & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ W_4^{-0} & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix} = \begin{bmatrix} 4 \\ 8 \\ 12 \\ 4 \end{bmatrix}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^{-0} & W_4^{-0} & W_4^{-0} & W_4^{-0} \\ W_4^{-0} & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ W_4^{-0} & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ W_4^{-0} & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix}^{-1} \begin{bmatrix} 4 \\ 8 \\ 12 \\ 4 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}^{-1} \begin{bmatrix} 4 \\ 8 \\ 12 \\ 4 \end{bmatrix}$$

Scanned with CamScanner

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}\begin{bmatrix} 4 \\ 8 \\ 12 \\ 4 \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} 4+8+12+4 \\ 4-j8-12+j4 \\ 4-8+12-4 \\ 4+j8-12-j4 \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} 28 \\ -8-j4 \\ 4 \\ -8+j4 \end{bmatrix} = \begin{bmatrix} 7 \\ -2-j1 \\ 1 \\ -2+j1 \end{bmatrix}$$

$\therefore X(K) = \{7, -2-j1, 1, -2+j1\}$

b) DFT : $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

Here $N=4$, $X(K) = \sum_{n=0}^{3} x(n) W_4^{Kn}$

$$X(K) = x(0)W_4^0 + x(1)W_4^K + x(2)W_4^{2K} + x(3)W_4^{3K}$$

$$X(K) = 1 + 2W_4^K + 3W_4^{2K} + W_4^{3K} \longrightarrow (1)$$

@K=0, $X(0) = 1 + 2W_4^0 + 3W_4^0 + W_4^0 = 1+2+3+1 = 7$

@K=1, $X(1) = 1 + 2W_4^1 + 3W_4^2 + W_4^3 = 1+2(-j)+3(-1)+j = -2-j1$

@K=2, $X(2) = 1 + 2W_4^2 + 3W_4^4 + W_4^6 = 1+2(-1)+3(1)+(-1) = 1$

@K=3, $X(3) = 1 + 2W_4^3 + 3W_4^6 + W_4^9 = 1+2(j)+3(-1)-j = -2+j1$

$\therefore X(K) = \{7, -2-j1, 1, -2+j1\}$

Scanned with CamScanner

2) Consider two periodic sequences $x(n)$ and $y(n)$ $x(n)$ has a period 'N' and $y(n)$ has a period 'M'. The sequence $w(n)$ is defined as $w(n) = x(n) + y(n)$.

a) Show that $w(n)$ is periodic with period 'MN'.

b) Also show that $W(K)$ represents 'MN' point DFT of an 'MN' point sequence $w(n)$

Sol^n:-

a) To prove that $w(n) = w(n+MN)$:

The sequence $w(n)$ is periodic with period MN. Then,

$w(n+MN) = x(n+MN) + y(n+MN)$

$x(n) = x(n+N) = x(n+2N) = \ldots\ldots = x(n+MN)$ i.e., for all integer multiples of 'N'.

$y(n) = y(n+N) = y(n+2N) = \ldots\ldots = y(n+MN)$ i.e., for all integer multiples of 'M'.

$\therefore w(n+MN) = x(n) + y(n)$

$= w(n)$

Thus, $w(n)$ is periodic with period 'MN'.

b) DFT of $w(n)$:

The N-point DFT of the sequence of length 'N' is given by,

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \; ; \; 0 \le K \le (N-1).$$

Hence, DFT of $w(n)$ will be of length 'MN' which will be,

$$W(K) = \sum_{n=0}^{MN-1} w(n) W_N^{Kn} \; ; \; 0 \le K \le MN-1$$

Thus, $W(K)$ is MN-point DFT of $w(n)$.

Scanned with CamScanner

3) a) An analog signal is sampled at 10KHz and the DFT of 512 samples is computed. Determine the frequency spacing between spectral samples of the DFT.
b) Determine the frequency spacing if 1024-point DFT is calculated.

Sol^n:-

a) Given $N = 512$, $F_S = 10KHz$

Frequency resolution $\Delta F = \frac{1}{N} \cdot F_S$

$$= \frac{1}{512} \times 10{,}000$$

$$\langle \Delta F = 19.53125\ Hz \rangle$$

This means the frequency spacing between the samples of DFT will be 19.53125 Hz.

b) For 1024-point DFT, $N = 1024$ and the frequency spacing will be:

$$\Delta F = \frac{1}{N} F_S = \frac{1}{1024} \times 10{,}000 = 9.7656\ Hz$$

4) Find the Z-transform of the sequence $x(n) = \{0.5, 0, 0.5, 0\}$ (↑ at 0.5) using Z-transform, find its DFT.

Sol^n:- Here $N = 4$

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

$$X(z) = \sum_{n=0}^{3} x(n) z^{-n}$$

$$\langle X(z) = 0.5 + 0.5 z^{-2} \rangle$$

$$X(K) = X(z)\Big|_{z = e^{\frac{j2\pi}{N}K}} = e^{\frac{j2\pi}{4}K}$$

$$X(K) = 0.5 + 0.5\, e^{\frac{-j2\pi}{4}K(-2)} \quad ; 0 \le K \le 3$$

$$\langle X(K) = \{1, 0, 1, 0\} \rangle$$

Scanned with CamScanner

Module-1

Anusha.M.N
Asst. professor
Dept. of ECE
BGSIT, B.G.Nagar

Properties of DFT

Properties of DFT :-

The DFT satisfies various properties that are useful in signal processing applications. A list of the DFT properties are given below:

1) Periodicity
2) Linearity
3) Circular time shift
4) Circular time reversal
5) Circular frequency shift
6) Circular Convolution
7) Modulation
8) Symmetry properties
9) Circular Correlation
10) parseval's theorem
11) Complex Conjugate property

1) periodicity :-

If $x(n) \xleftrightarrow[\text{DFT}]{N} X(K)$

then $x(n+N) = x(n)$; for all 'n'

and $X(K+N) = X(K)$; for all 'K'.

proof :-

W.K.T $x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K)\, e^{\frac{j2\pi}{N}Kn}$; $0 \le n \le N-1$

$\therefore \quad x(n+N) = \frac{1}{N} \sum_{K=0}^{N-1} X(K)\, e^{\frac{j2\pi}{N}K(n+N)}$

$= \frac{1}{N} \sum_{K=0}^{N-1} X(K)\, e^{\frac{j2\pi}{N}Kn} \cdot \underbrace{e^{\frac{j2\pi}{N}KN}}_{1}$

$= \frac{1}{N} \sum_{K=0}^{N-1} X(K)\, e^{\frac{j2\pi}{N}Kn} \qquad \left[\because e^{j2\pi K} = 1\right]$

$\therefore x(n+N) = x(n)$

Scanned with CamScanner

Similarly, $X(K) = \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn}$; $0 \le K \le (N-1)$

$$\therefore X(K+N) = \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}(K+N)n}$$

$$= \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} \cdot e^{-j\frac{2\pi Nn}{N}}$$

$$X(K+N) = \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} \quad \left[\because e^{-j2\pi n} = 1\right]$$

$$\langle \therefore X(K+N) = X(K) \rangle$$

2) Linearity :-

If $x_1(n) \xleftrightarrow[\text{DFT}]{N} X_1(K)$

and $x_2(n) \xleftrightarrow[\text{DFT}]{N} X_2(K)$

then for any real-valued (or) complex-valued constants 'a' and 'b',

$$y(n) = a\,x_1(n) + b\,x_2(n) \xleftrightarrow[\text{DFT}]{N} Y(K) = a\,X_1(K) + b\,X_2(K)$$

proof :-

$$Y(K) = \sum_{n=0}^{N-1} y(n)\, e^{-j\frac{2\pi}{N}Kn} \quad ; 0 \le K \le (N-1)$$

$$= \sum_{n=0}^{N-1} [a\,x_1(n) + b\,x_2(n)]\, e^{-j\frac{2\pi}{N}Kn}$$

$$= a\sum_{n=0}^{N-1} x_1(n)\, e^{-j\frac{2\pi}{N}Kn} + b\sum_{n=0}^{N-1} x_2(n)\, e^{-j\frac{2\pi}{N}Kn}$$

$$\langle \therefore Y(K) = a\,X_1(K) + b\,X_2(K) \rangle$$

3) Circular time shift :-

If $x(n) \xleftrightarrow[\text{DFT}]{N} X(K)$

then $x((n-l))_N \xleftrightarrow[\text{DFT}]{N} X(K)\, e^{-j\frac{2\pi}{N}Kl} = W_N^{Kl}\, X(K)$

Scanned with CamScanner

Proof :-

From the definition of DFT,

$$\text{DFT}\{x((n-l))_N\} = \sum_{n=0}^{N-1} x((n-l))_N e^{-j\frac{2\pi}{N}Kn}$$

$$= \underbrace{\sum_{n=0}^{l-1} x((n-l))_N e^{-j\frac{2\pi}{N}Kn}}_{\text{I-term}} + \underbrace{\sum_{n=l}^{N-1} x(n-l) e^{-j\frac{2\pi}{N}Kn}}_{\text{II-term}}$$

W.K.T $x((n-l))_N = x(N-l+n)$

$$\therefore \text{I term} \longrightarrow \sum_{n=0}^{l-1} x((n-l))_N e^{-j\frac{2\pi}{N}Kn} = \sum_{n=0}^{l-1} x(N-l+n) e^{-j\frac{2\pi}{N}Kn}$$

put $m = N-l+n$; @$n=0$, $m = N-l+0 = N-l$

@$n=l-1$, $m = N-l+l-1 = N-1$

$$= \sum_{m=N-l}^{N-1} x(m) e^{-j\frac{2\pi}{N}K(m+l-N)}$$

$$= \sum_{m=N-l}^{N-1} x(m) e^{-j\frac{2\pi}{N}K(m+l)} \underbrace{e^{j\frac{2\pi}{N}KN}}_{1}$$

$$= \sum_{m=N-l}^{N-1} x(m) e^{-j\frac{2\pi}{N}K(m+l)} \quad \left[\because e^{j2\pi K} = 1\right]$$

$$\text{II-term} \longrightarrow \sum_{n=l}^{N-1} x(n-l) e^{-j\frac{2\pi}{N}Kn}$$

put $m = n-l$; @$n=l$, $m = l-l = \underline{0}$

@$n = N-1$, $m = \underline{N-1-l}$

$$= \sum_{m=0}^{N-l-1} x(m) e^{-j\frac{2\pi}{N}K(m+l)}$$

$$\therefore \text{DFT}\{x((n-l))_N\} = \text{II-term} + \text{I-term}$$

$$= \sum_{m=0}^{N-l-1} x(m) e^{-j\frac{2\pi}{N}K(m+l)} + \sum_{m=N-l}^{N-1} x(m) e^{-j\frac{2\pi}{N}K(m+l)}$$

Scanned with CamScanner

$$\text{DFT}\{x((n-l))_N\} = \sum_{m=0}^{N-1} x(m)\, e^{-j\frac{2\pi}{N}km}\, e^{-j\frac{2\pi}{N}kl}$$

$$= X(K)\, e^{-j\frac{2\pi}{N}kl}$$

$$\left\langle \text{DFT}\{x((n-l))_N\} = W_N^{kl}\, X(K) \right\rangle$$

4) Time Reversal property :-

If $x(n) \xleftrightarrow[\text{DFT}]{N} X(K)$

then $y(n) = x((-n))_N = x(N-n) \xleftrightarrow[\text{DFT}]{N} Y(K) = X((-K))_N = X(N-K)$

proof :-

$$Y(K) = \text{DFT}\{y(n)\} = \text{DFT}\{x((-n))_N\}$$

$$= \text{DFT}\{x(N-n)\}$$

$$= \sum_{n=0}^{N-1} x(N-n)\, e^{-j\frac{2\pi}{N}kn}$$

put $m = N-n$,

@ $n=0$, $m = N-0 = \underline{N}$

@ $n=N-1$, $m = N-N+1 = \underline{1}$

$$Y(K) = \sum_{m=1}^{N} x(m)\, e^{-j\frac{2\pi}{N}k(N-m)}$$

$$= \sum_{m=1}^{N} x(m)\, e^{j\frac{2\pi}{N}km}\, \underbrace{e^{-j\frac{2\pi}{N}kN}}_{1}$$

$$= \sum_{m=0}^{N-1} x(m)\, e^{j\frac{2\pi}{N}km}\, e^{-j\frac{2\pi}{N}mN}$$ [from periodicity property of DFT]

$$= \sum_{m=0}^{N-1} x(m)\, e^{-j\frac{2\pi}{N}m(N-K)}$$

$$= X(N-K)$$

$$\left\langle Y(K) = X((-K))_N \right\rangle$$

Scanned with CamScanner

5) Circular frequency Shift property :-

If $x(n) \xleftrightarrow[\text{DFT}]{N} X(K)$

then $y(n) = x(n)\, e^{j\frac{2\pi}{N}ln} \xleftrightarrow[\text{DFT}]{N} X((K-l))_N$

proof :-

$$y(n) = \text{IDFT}\{X((K-l))_N\} = \frac{1}{N}\left[\sum_{K=0}^{N-1} X((K-l))_N\, e^{j\frac{2\pi}{N}Kn}\right]$$

$$= \frac{1}{N}\left[\sum_{K=0}^{l-1} X((K-l))_N\, e^{j\frac{2\pi}{N}Kn} + \sum_{K=l}^{N-1} X(K-l)\, e^{j\frac{2\pi}{N}Kn}\right]$$

$$= \frac{1}{N}\left[\underbrace{\sum_{K=0}^{l-1} X(N-l+K)\, e^{j\frac{2\pi}{N}Kn}}_{\text{I-term}} + \underbrace{\sum_{K=l}^{N-1} X(K-l)\, e^{j\frac{2\pi}{N}Kn}}_{\text{II-term}}\right]$$

$$\text{I-term} \rightarrow \sum_{K=0}^{l-1} X(N-l+K)\, e^{j\frac{2\pi}{N}Kn}$$

put $m = N-l+K$

@ $K=0$, $m = N-l+0 = \underline{N-l}$

@ $K=l-1$, $m = N-l+l-1 = \underline{N-1}$

$$= \sum_{m=N-l}^{N-1} X(m)\, e^{j\frac{2\pi}{N}n(m-N+l)}$$

$$= \sum_{m=N-l}^{N-1} X(m)\, e^{j\frac{2\pi}{N}n(m+l)}\, e^{-j2\pi n N}$$

$e^{-j2\pi n N} = 1$

$$= \sum_{m=N-l}^{N-1} X(m)\, e^{j\frac{2\pi}{N}n(m+l)}$$

$$\text{II term} \rightarrow \sum_{K=l}^{N-1} X(K-l)\, e^{j\frac{2\pi}{N}Kn}$$

put $m = K-l$,

@ $K=l$, $m = l-l = \underline{0}$

@ $K=N-1$, $m = \underline{N-1-l}$

Scanned with CamScanner

$$= \sum_{m=0}^{N-l-1} X(m)\, e^{\frac{j2\pi}{N}n(m+l)}$$

$$\therefore y(n) = \text{IDFT}\{X((K-l))_N\} = \frac{1}{N}\left[\text{II term} + \text{I term}\right]$$

$$= \frac{1}{N}\left[\sum_{m=0}^{N-l-1} X(m)\, e^{\frac{j2\pi}{N}n(m+l)} + \sum_{m=N-l}^{N-1} X(m)\, e^{\frac{j2\pi}{N}n(m+l)}\right]$$

$$= \frac{1}{N}\sum_{m=0}^{N-1} X(m)\, e^{\frac{j2\pi}{N}n(m+l)}$$

$$= \frac{1}{N}\sum_{m=0}^{N-1} X(m)\, e^{\frac{j2\pi}{N}mn}\, e^{\frac{j2\pi}{N}ln}$$

$$\boxed{y(n) = x(n)\, e^{\frac{j2\pi}{N}ln}}$$

6) Circular Convolution :-

If $x_1(n) \xleftrightarrow[\text{IDFT}]{N} X_1(K)$

and $x_2(n) \xleftrightarrow[\text{IDFT}]{N} X_2(K)$

then $y(n) = x_1(n) \textcircled{N} x_2(n) \xleftrightarrow[\text{IDFT}]{N} Y(K) = X_1(K)\, X_2(K)$

Proof :-

Consider two length 'N' sequences $x_1(n)$ and $x_2(n)$. Their N-point DFT is given by,

$$X_1(K) = \sum_{n=0}^{N-1} x_1(n)\, e^{-\frac{j2\pi}{N}Kn} \quad ; 0 \le K \le (N-1) \longrightarrow (1)$$

$$X_2(K) = \sum_{n=0}^{N-1} x_2(n)\, e^{-\frac{j2\pi}{N}Kn} \quad ; 0 \le K \le (N-1) \longrightarrow (2)$$

Let $Y(K) = X_1(K)\, X_2(K) \quad ; 0 \le K \le (N-1) \longrightarrow (3)$

Then IDFT of $Y(K)$ is given by,

$$y(n) = \frac{1}{N}\sum_{K=0}^{N-1} Y(K)\, e^{\frac{j2\pi}{N}Kn} \quad ; 0 \le n \le (N-1) \longrightarrow (4)$$

Substituting Eq(3) in Eq(4),

$$y(n) = \frac{1}{N}\sum_{K=0}^{N-1} X_1(K)\, X_2(K)\, e^{\frac{j2\pi}{N}Kn} \longrightarrow (5)$$

Scanned with CamScanner

Replacing 'n' by 'm' and 'l' in Eq(1) and Eq(2) respectively and substituting in Eq(5),

$$y(n) = \frac{1}{N}\sum_{K=0}^{N-1}\left[\sum_{m=0}^{N-1} x_1(m)\, e^{-j\frac{2\pi}{N}Km}\right]\left[\sum_{l=0}^{N-1} x_2(l)\, e^{-j\frac{2\pi}{N}Kl}\right] e^{j\frac{2\pi}{N}Kn} \longrightarrow (6)$$

Changing the order of summation in Eq(6),

$$y(n) = \frac{1}{N}\sum_{m=0}^{N-1} x_1(m) \sum_{l=0}^{N-1} x_2(l)\left[\sum_{K=0}^{N-1} e^{j\frac{2\pi}{N}K(n-m-l)}\right] \longrightarrow (7)$$

The sum in the bracket in Eq(7) has the form,

$$\sum_{K=0}^{N-1}\alpha^K = \frac{1-\alpha^N}{1-\alpha} \quad ; \alpha \neq 1$$

$$= N \quad ; \alpha = 1$$

In Eq(7), $\alpha = 1$ when $(n-m-l)$ is an integer multiple of N.

i.e., $\sum_{K=0}^{N-1}\alpha^K = N$; $n-m-l = pN$, where 'p' is an integer.

i.e., $l = n-m-pN = ((n-m))_N$

$= 0$; otherwise

∴ From Eq(7),

$$y(n) = \sum_{m=0}^{N-1} x_1(m)\, x_2((n-m))_N \; ; 0 \le n \le (N-1) \longrightarrow (8)$$

$$\langle \therefore\ y(n) = x_1(n) \textcircled{N} x_2(n) \rangle \longrightarrow (9)$$

In Eq(9), $y(n)$ is the N-point Circular Convolution of $x_1(n)$ and $x_2(n)$. Therefore, the Circular Convolution of two length-N sequences can also be computed by multiplying their N-point DFT's and then taking an N-point IDFT. This method is known as Stockham's method.

7) Modulation (Multiplication of two sequences) :-

If $x_1(n) \xleftrightarrow[\text{DFT}]{N} X_1(K)$

and $x_2(n) \xleftrightarrow[\text{DFT}]{N} X_2(K)$

then $y(n) = x_1(n)\, x_2(n) \xleftrightarrow[\text{DFT}]{N} Y(K) = \frac{1}{N}\left[X_1(K) \textcircled{N} X_2(K)\right]$

Scanned with CamScanner

Proof :-

Consider two sequences $x_1(n)$ and $x_2(n)$ which has N-point DFT $X_1(K)$ and $X_2(K)$ respectively -

$$\therefore x_1(n) = \frac{1}{N}\sum_{K=0}^{N-1} X_1(K)\, e^{\frac{j2\pi Kn}{N}} \quad ; 0 \le n \le (N-1) \longrightarrow (1)$$

$$\therefore x_2(n) = \frac{1}{N}\sum_{K=0}^{N-1} X_2(K)\, e^{\frac{j2\pi Kn}{N}} \quad ; 0 \le n \le (N-1) \longrightarrow (2)$$

Let $y(n) = x_1(n)\, x_2(n) \longrightarrow (3)$

The N-point DFT of $y(n)$ is given by,

$$Y(K) = \sum_{n=0}^{N-1} y(n)\, e^{-\frac{j2\pi}{N}Kn} \quad ; 0 \le K \le (N-1) \longrightarrow (4)$$

Substituting Eq(3) in Eq(4),

$$Y(K) = \sum_{n=0}^{N-1} x_1(n)\, x_2(n)\, e^{-\frac{j2\pi}{N}Kn} \longrightarrow (5)$$

Replacing 'K' by 'm' and 'l' in Eq(1) and Eq(2) respectively and substituting in Eq(5),

$$Y(K) = \sum_{n=0}^{N-1} \left[\frac{1}{N}\sum_{m=0}^{N-1} X_1(m)\, e^{\frac{j2\pi mn}{N}}\right]\left[\frac{1}{N}\sum_{l=0}^{N-1} X_2(l)\, e^{+\frac{j2\pi ln}{N}}\right] e^{-\frac{j2\pi Kn}{N}} \longrightarrow (6)$$

Changing the order of summation in Eq(6),

$$Y(K) = \frac{1}{N^2}\sum_{m=0}^{N-1} X_1(m) \sum_{l=0}^{N-1} X_2(l) \left[\sum_{n=0}^{N-1} e^{-\frac{j2\pi n(K-m-l)}{N}}\right] \longrightarrow (7)$$

The sum in the bracket in Eq(7) has the form,

$$\sum_{n=0}^{N-1} \alpha^n = \frac{1-\alpha^N}{1-\alpha} \quad ; \alpha \neq 1$$

$$= N \quad ; \alpha = 1$$

where $\alpha = e^{-\frac{j2\pi}{N}(K-m-l)} \longrightarrow (8)$

In Eq(8), $\alpha = 1$ when $(K-m-l)$ is an integer multiple of N.

Scanned with CamScanner

i.e., $\sum_{n=0}^{N-1} \alpha^n = N$; $(K-m-l) = pN$, where 'p' is an integer.

(i.e., $l = K-m-pN = ((K-m))_N$

$= 0$; otherwise.

∴ From eq(7)

$$Y(K) = \frac{1}{N} \sum_{m=0}^{N-1} X_1(m)\, X_2((K-m))_N$$

$$\therefore Y(K) = \frac{1}{N}\left[X_1(K) \textcircled{N} X_2(K)\right]$$

Thus $Y(K)$ is $\frac{1}{N}$ times Circular Convolution of $X_1(K)$ and $X_2(K)$.

8) Circular Correlation :-

For Complex-valued sequence $x(n)$ and $y(n)$ if,

$$x(n) \xleftrightarrow[\text{DFT}]{N} X(K)$$

and $y(n) \xleftrightarrow[\text{DFT}]{N} Y(K)$

then $\tilde{\gamma}_{xy}(l) \xleftrightarrow[\text{DFT}]{N} \tilde{R}_{xy}(K) = X(K)Y^*(K)$

where $\tilde{\gamma}_{xy}(l)$ is the Circular Correlation sequence defined as,

$$\tilde{\gamma}_{xy}(l) = \sum_{n=0}^{N-1} x(n)\, y^*((n-l))_N \longrightarrow (1)$$

Proof :-

From eq(1),

$$\tilde{\gamma}_{xy}(l) = \sum_{n=0}^{N-1} x(n)\, y^*((n-l))_N$$

Eq(1) for $\tilde{\gamma}_{xy}(l)$ can be written as the N-point Circular Convolution of $x(n)$ with $y^*(-n)$.

i.e., $\tilde{\gamma}_{xy}(l) = x(l) \textcircled{N} y^*(-l) \longrightarrow (2)$

Now taking DFT on both the sides of Eq(2),

$$\tilde{R}_{xy}(K) = X(K)\, Y^*(K) \longrightarrow (3)$$

Scanned with CamScanner

If $x(n) = y(n)$, then $\tilde{r}_{xx}(l)$ is known as Circular auto-correlation of $x(n)$.

From Eq(8),

$$\tilde{r}_{xx}(l) \xleftrightarrow[\text{DFT}]{N} \tilde{R}_{xx}(K) = X(K)X^*(K) = |X(K)|^2$$

9) Parseval's theorem :-

For Complex-Valued Sequence $x(n)$ and $y(n)$,

if $x(n) \xleftrightarrow[\text{DFT}]{N} X(K)$

and $y(n) \xleftrightarrow[\text{DFT}]{N} Y(K)$

then $\sum_{n=0}^{N-1} x(n)y^*(n) \xleftrightarrow[\text{DFT}]{N} \frac{1}{N}\sum_{K=0}^{N-1} X(K)Y^*(K)$

Proof :-

W.K.T $\tilde{r}_{xy}(l) = \sum_{n=0}^{N-1} x(n)y^*((n-l))_N$ ——— (1)

Substituting $l=0$ in Eq(1),

$$\tilde{r}_{xy}(0) = \sum_{n=0}^{N-1} x(n)y^*(n) \quad \text{——— (2)}$$

If $\tilde{r}_{xy}(l) \xleftrightarrow[\text{DFT}]{N} \tilde{R}_{xy}(K)$

then $\tilde{r}_{xy}(l) = \frac{1}{N}\sum_{K=0}^{N-1} \tilde{R}_{xy}(K)\, e^{\frac{j2\pi Kl}{N}}$ ——— (3)

From Circular Correlation property,

$$\tilde{r}_{xy}(l) = \frac{1}{N}\sum_{K=0}^{N-1} X(K)Y^*(K)\, e^{\frac{j2\pi Kl}{N}} \quad \text{——— (4)}$$

Substituting $l=0$ in Eq(4),

$$\tilde{r}_{xy}(0) = \frac{1}{N}\sum_{K=0}^{N-1} X(K)Y^*(K) \quad \text{——— (5)}$$

Comparing Eq(2) and Eq(5),

$$\sum_{n=0}^{N-1} x(n)y^*(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K)Y^*(K) \quad \text{——— (6)}$$

Scanned with CamScanner

Comparing real and imaginary parts in Eq(5),

$$X_R(K) = \sum_{n=0}^{N-1} \left[x_R(n) \cos\left(\frac{2\pi}{N} Kn\right) + x_I(n) \sin\left(\frac{2\pi}{N} Kn\right) \right] \longrightarrow (a)$$

and $$X_I(K) = -\sum_{n=0}^{N-1} \left[x_R(n) \sin\left(\frac{2\pi}{N} Kn\right) - x_I(n) \cos\left(\frac{2\pi}{N} Kn\right) \right] \longrightarrow (b)$$

Similarly by the difinition of IDFT,

$$x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K)\, e^{j\frac{2\pi}{N}Kn} \quad ; 0 \le n \le (N-1) \longrightarrow (6)$$

Substituting Eq(1) and Eq(2) in Eq(6),

$$x_R(n) + j x_I(n) = \frac{1}{N} \sum_{K=0}^{N-1} \left[X_R(K) + j X_I(K) \right] e^{j\frac{2\pi}{N}Kn}$$

$$= \frac{1}{N} \sum_{K=0}^{N-1} \left[X_R(K) + j X_I(K) \right] \left[\cos\left(\frac{2\pi Kn}{N}\right) + j \sin\left(\frac{2\pi Kn}{N}\right) \right]$$

$$= \frac{1}{N} \sum_{K=0}^{N-1} \left[X_R(K) \cos\left(\frac{2\pi Kn}{N}\right) + j X_R(K) \sin\left(\frac{2\pi Kn}{N}\right) + j X_I(K) \cos\left(\frac{2\pi Kn}{N}\right) - X_I(K) \sin\left(\frac{2\pi Kn}{N}\right) \right] \longrightarrow (7)$$

Grouping real and imaginary parts in Eq(7),

$$x_R(n) + j x_I(n) = \frac{1}{N} \sum_{K=0}^{N-1} \left\{ \left[X_R(K) \cos\left(\frac{2\pi Kn}{N}\right) - X_I(K) \sin\left(\frac{2\pi Kn}{N}\right) \right] + j \left[X_R(K) \sin\left(\frac{2\pi Kn}{N}\right) + X_I(K) \cos\left(\frac{2\pi}{N} Kn\right) \right] \right. \longrightarrow (8)$$

Comparing real and imaginary parts in Eq(8),

$$x_R(n) = \frac{1}{N} \sum_{K=0}^{N-1} \left[X_R(K) \cos\left(\frac{2\pi}{N} Kn\right) - X_I(K) \sin\left(\frac{2\pi Kn}{N}\right) \right] \text{ and } \longrightarrow (c)$$

$$x_I(n) = \frac{1}{N} \sum_{K=0}^{N-1} \left[X_R(K) \sin\left(\frac{2\pi}{N} Kn\right) + X_I(K) \cos\left(\frac{2\pi}{N} Kn\right) \right] \longrightarrow (d)$$

i) Real and Even Sequence :-

Consider that length-N sequence $x(n)$ is real-valued [i.e., $x_I(n) = 0 \;\therefore x(n) = x_R(n)$] and Even.

i.e., $x(n) = x(N-n) = x((-n))_N \quad ; 0 \le n \le (N-1) \longrightarrow (9)$

Scanned with CamScanner

Eq(6) is the general form of parseval's theorem.
As a special form, if $x(n) = y(n)$, then Eq(6) becomes,

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{K=0}^{N-1} |X(K)|^2 \longrightarrow (7)$$

Eq(7) gives the energy of finite-duration sequence $x(n)$ in terms of its DFT.

10) Symmetry properties :-

Let us assume that a length-N sequence $x(n)$ and its N-point DFT $X(K)$ are both Complex Valued. Then they can be expressed as,

$$x(n) = x_R(n) + j x_I(n) \quad ; 0 \le n \le (N-1) \longrightarrow (1)$$

$$X(K) = X_R(K) + j X_I(K) \quad ; 0 \le K \le (N-1) \longrightarrow (2)$$

Where $x_R(n)$ and $x_I(n)$ are real and imaginary parts of $x(n)$ respectively. Similarly, $X_R(K)$ and $X_I(K)$ are real and imaginary parts of $X(K)$ respectively.

By definition of DFT,

$$X(K) = \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn} \quad ; 0 \le K \le (N-1) \longrightarrow (3)$$

Substituting Eq(1) and Eq(2) in Eq(3),

$$X_R(K) + j X_I(K) = \sum_{n=0}^{N-1} \left[x_R(n) + j x_I(n)\right] e^{-j\frac{2\pi}{N}Kn}$$

$$= \sum_{n=0}^{N-1} \left[x_R(n) + j x_I(n)\right] \left[\cos\left(\frac{2\pi}{N}Kn\right) - j \sin\left(\frac{2\pi}{N}Kn\right)\right]$$

$$= \sum_{n=0}^{N-1} \left[x_R(n) \cos\left(\frac{2\pi}{N}Kn\right) - j x_R(n) \sin\left(\frac{2\pi}{N}Kn\right) + j x_I(n) \cos\left(\frac{2\pi}{N}Kn\right) + x_I(n) \sin\left(\frac{2\pi}{N}Kn\right)\right] \longrightarrow (4)$$

Grouping real and imaginary parts in Eq(4),

$$X_R(K) + j x_I(K) = \sum_{n=0}^{N-1} \left\{\left[x_R(n) \cos\left(\frac{2\pi}{N}Kn\right) + x_I(n) \sin\left(\frac{2\pi}{N}Kn\right)\right] - j\left[x_R(n) \sin\left(\frac{2\pi}{N}Kn\right) - x_I(n) \cos\left(\frac{2\pi}{N}Kn\right)\right]\right\} \longrightarrow (5)$$

Scanned with CamScanner

W.K.T $X(K) = X_R(K) + j X_I(K)$ ⟶ (10)

Substituting Eq(a) and (b) in Eq(10),

$$X(K) = \sum_{n=0}^{N-1} \left[x_R(n) \cos\left(\frac{2\pi}{N}Kn\right) + x_I(n) \sin\left(\frac{2\pi}{N}Kn\right)\right] - j \sum_{n=0}^{N-1} \left[x_R(n) \sin\left(\frac{2\pi}{N}Kn\right) - x_I(n) \cos\left(\frac{2\pi}{N}Kn\right)\right]$$

$$X(K) = \sum_{n=0}^{N-1} x(n) \cos\left(\frac{2\pi}{N}Kn\right) - j \sum_{n=0}^{N-1} x(n) \sin\left(\frac{2\pi}{N}Kn\right) \quad [\because x(n) = x_R(n)] \longrightarrow (11)$$

In Eq(11), the second term is zero because the sequence $x(n)$ is Even and $\sin\left(\frac{2\pi}{N}Kn\right)$ is odd. The multiplication of Even and odd sequences will result in an odd sequence. The summation of the samples of odd sequence is zero.

$$\therefore X(K) = \sum_{n=0}^{N-1} x(n) \cos\left(\frac{2\pi}{N}Kn\right) \; ; 0 \le K \le (N-1)$$

ii) Real and odd sequence :-

Consider that length-N sequence $x(n)$ is real-valued [i.e., $x_I(n) = 0 \;\therefore x(n) = x_R(n)$] and odd.

i.e., $x(n) = -x(N-n) = -x((-n))_N \; ; 0 \le n \le (N-1)$

W.K.T $X(K) = X_R(K) + j X_I(K)$ ⟶ (11)

Substituting Eq(a) and (b) in Eq(11),

$$X(K) = \sum_{n=0}^{N-1} \left[x_R(n) \cos\left(\frac{2\pi}{N}Kn\right) + x_I(n) \sin\left(\frac{2\pi}{N}Kn\right)\right] - j \sum_{n=0}^{N-1} \left[x_R(n) \sin\left(\frac{2\pi}{N}Kn\right) - x_I(n) \cos\left(\frac{2\pi}{N}Kn\right)\right]$$

$$= \sum_{n=0}^{N-1} x(n) \cos\left(\frac{2\pi}{N}Kn\right) - j \sum_{n=0}^{N-1} x(n) \sin\left(\frac{2\pi}{N}Kn\right) \quad [\because x_I(n) = 0 \;\&\; x(n) = x_R(n)] \longrightarrow (12)$$

In Eq(12), the first term is zero because the sequence $x(n)$ is odd and $\cos\left(\frac{2\pi}{N}Kn\right)$ is Even. The multiplication of odd and Even sequences results in odd sequence. The summation of odd sequence is zero.

Scanned with CamScanner

$$\therefore X(K) = -j \sum_{n=0}^{N-1} x(n) \sin\left(\frac{2\pi Kn}{N}\right) ; \quad 0 \le K \le (N-1)$$

iii) purely imaginary sequence :—

Consider that length-N sequence $x(n)$ is purely imaginary,

[i.e., $x_R(n) = 0 \therefore x(n) = j x_I(n)$]

W.K.T $X(K) = X_R(K) + j X_I(K)$ ——— (13)

Substituting Eq(a) and Eq(b) in Eq(13),

$$X(K) = \sum_{n=0}^{N-1} \left[x_R(n) \cancel{\cos\left(\frac{2\pi Kn}{N}\right)}^{0} + x_I(n) \sin\left(\frac{2\pi Kn}{N}\right) \right] - j \sum_{n=0}^{N-1} \left[x_R(n) \cancel{\sin\left(\frac{2\pi}{N} Kn\right)}^{0} - x_I(n) \cos\left(\frac{2\pi}{N} Kn\right) \right]$$

$$X_R(K) + j X_I(K) = \sum_{n=0}^{N-1} x(n) \sin\left(\frac{2\pi}{N} Kn\right) + j \sum_{n=0}^{N-1} x(n) \cos\left(\frac{2\pi}{N} Kn\right)$$

Comparing real and imaginary parts, [$\because x_R(n) = 0$ & $x(n) = x_I(n)$]

$$X_R(K) = \sum_{n=0}^{N-1} x(n) \sin\left(\frac{2\pi}{N} Kn\right) ; \quad 0 \le K \le (N-1)$$

& $$X_I(K) = \sum_{n=0}^{N-1} x(n) \cos\left(\frac{2\pi}{N} Kn\right) ; \quad 0 \le K \le (N-1)$$

Scanned with CamScanner

Problems on Circular time shifting property :—

1) A length-6 sequence is given by,
$x(n) = \delta(n) + 3\delta(n-1) + 2\delta(n-2) + 5\delta(n-3) - 2\delta(n-4) + 7\delta(n-5)$.
obtain $y(n) = x((n-4))_6$

Soln :-

$x(n) = \{1, 3, 2, 5, -2, 7\}$

$y(n) = x((n-4))_6$

@ $n=0$, $y(0) = x((0-4))_6 = x(6-4) = x(2) = \underline{2}$

@ $n=1$, $y(1) = x((1-4))_6 = x(6-3) = x(3) = \underline{5}$

@ $n=2$, $y(2) = x((2-4))_6 = x(6-2) = x(4) = \underline{-2}$

@ $n=3$, $y(3) = x((3-4))_6 = x(6-1) = x(5) = \underline{7}$

@ $n=4$, $y(4) = x((4-4))_6 = x(0) = \underline{1}$

@ $n=5$, $y(5) = x((5-4))_6 = x(1) = \underline{3}$

$\therefore y(n) = x((n-4))_6 = \{2, 5, -2, 7, 1, 3\}$

2) A length-10 sequence is given by,
$S(n) = \{0, 3, 4, -1, 4, 2, 8, 9, 2, 3\}$. obtain
a) $x(n) = S((n-7))_{10}$ b) $y(n) = S((n+3))_{10}$ c) Compare the results of (a) and (b).

Soln :-

a) $x(n) = S((n-7))_{10}$

$x(0) = S((-7))_{10} = S(10-7) = S(3) = \underline{-1}$

$x(1) = S((1-7))_{10} = S(4) = \underline{4}$

$x(2) = S((2-7))_{10} = S(5) = \underline{2}$

$x(3) = S((3-7))_{10} = S(6) = \underline{8}$

$x(4) = S((4-7))_{10} = S(7) = \underline{9}$

$x(5) = S((5-7))_{10} = S(8) = \underline{2}$

$x(6) = S((6-7))_{10} = S(9) = \underline{3}$

$x(7) = S((7-7))_{10} = S(0) = \underline{0}$

$x(8) = S((8-7))_{10} = S(1) = \underline{3}$

$x(9) = S((9-7))_{10} = S(2) = \underline{4}$

$\therefore x(n) = S((n-7))_{10} = \{-1, 4, 2, 8, 9, 2, 3, 0, 3, 4\}$

Scanned with CamScanner

b) $y(n) = S((n+3))_{10}$

$y(0) = S((0+3))_{10} = S(8) = -1$ $\quad$ $y(6) = S((6+3))_{10} = S(9) = 3$

$y(1) = S((1+3))_{10} = S(4) = 4$ $\quad$ $y(7) = S((7+3))_{10} = S(10-10) = 0$

$y(2) = S((2+3))_{10} = S(5) = 2$ $\quad$ $y(8) = S((8+3))_{10} = S(1) = 3$

$y(3) = S((3+3))_{10} = S(6) = 8$ $\quad$ $y(9) = S((9+3))_{10} = S(2) = 4$

$y(4) = S((4+3))_{10} = S(7) = 9$

$y(5) = S((5+3))_{10} = S(8) = 2$

$\therefore y(n) = \{-1, 4, 2, 8, 9, 2, 3, 0, 3, 4\}$

c) From the results obtained in (a) and (b), $x(n) = y(n)$

i.e., $S((n-7))_{10} = S((n+3))_{10}$.

3) Compute the 4-point DFT of the sequence, $x_1(n) = \{1, 2, 3, 2\}$ and thereby compute the 4-point DFT of the sequence $x_2(n)$ using appropriate property of DFT, where $x_2(n) = \{3, 2, 1, 2\}$

Soln: a) $x_1(n) = \{1, 2, 3, 2\}$

$$X_1(K) = \sum_{n=0}^{N-1} x_1(n) W_N^{Kn}$$

Here $N = 4$, $X_1(K) = 1 + 2W_4^{K} + 3W_4^{2K} + 2W_4^{3K} \longrightarrow (1)$

@ $K = 0$, $X_1(0) = 1 + 2 + 3 + 2 = 8$

@ $K = 1$, $X_1(1) = 1 + 2(-j) + 3(-1) + 2(j) = -2$

@ $K = 2$, $X_1(2) = 1 + 2(-1) + 3(1) + 2(-1) = 0$

@ $K = 3$, $X_1(3) = 1 + 2(j) + 3(-1) + 2(-j) = -2$

$\therefore X_1(K) = \{8, -2, 0, -2\}$

b) $x_2(n) = \{3, 2, 1, 2\}$

It is observed that $x_2(n) = x_1((n-2))_4$

Taking DFT on both the sides using Circular time shift property, $X_2(K) = W_4^{2K} X_1(K) = e^{-j2\pi \times 2K} X_1(K) = (-1)^K X_1(K)$

Scanned with CamScanner

$\therefore X_2(0) = (-1)^0 X_1(0) = (1)(8) = \underline{8}$

$X_2(1) = (-1)^1 X_1(1) = -1(-2) = \underline{2}$

$X_2(2) = (-1)^2 X_1(2) = 1(0) = \underline{0}$

$X_2(3) = (-1)^3 X_1(3) = -1(-2) = \underline{2}$

$\therefore X_2(K) = \{8, 2, 0, 2\}$

4) The 8-point DFT of the following sequence is given by X(K).

$$x(n) = \begin{cases} 1 & ; 0 \le n \le 3 \\ 0 & ; 4 \le n \le 7 \end{cases}$$

Compute the DFT of the following sequences using circular time shift property in terms of X(K).

a) $x_1(n) = \{1, 0, 0, 0, 0, 1, 1, 1\}$ b) $x_2(n) = \{0, 0, 1, 1, 1, 1, 0, 0\}$

Sol^n:- Given $x(n) = \{1, 1, 1, 1, 0, 0, 0, 0\}$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \; ; \text{ Here } N = 8$$

$$= 1 + W_8^K + W_8^{2K} + W_8^{3K}$$

$$= 1 + e^{\frac{-j2\pi \cdot K}{8}} + e^{\frac{-j2\pi \cdot 2K}{8}} + e^{\frac{-j2\pi (3K)}{8}}$$

45° 90° 135°

$X(K) = \{4, 1-j2.4142, 0, 1-j0.4142, 0, 1+j0.4142, 0, 1+j2.4142\}$

a) $x_1(n) = \{1, 0, 0, 0, 0, 1, 1, 1\}$

It is observed that,

$$x_1(n) = x((n+3))_8 = x((n-5))_8$$

Using Circular time shifting property,

$$X_1(K) = e^{\frac{-j2\pi K(-3)}{8}} X(K) = e^{\frac{-j2\pi K(5)}{8}} X(K)$$

(find $X_1(K)$ by substituting $K = 0, 1, 2 \ldots 7$)

Scanned with CamScanner

b) $x_2(n) = \{0,0,1,1,1,1,0,0\}$

$x_2(n) = x((n-2))_8 = x((n+6))_8$

using Circular time shifting property,

$X_2(K) = e^{-j\frac{2\pi}{8}K(2)} X(K) = e^{-j\frac{2\pi}{8}K(-6)} X(K)$

(find $X_2(K)$ by substituting $K = 0,1,2 \ldots 7$)

5) Consider a finite duration sequence, $x(n) = \{0,1,2,3,4,0\}$. Find the sequence $y(n)$ with 6-point DFT given by,

$Y(K) = W_2^K X(K)$; $0 \le K \le 5$.

Sol^n:- Given $Y(K) = W_2^K X(K)$

$= e^{-j\frac{2\pi}{2}K} X(K)$

$Y(K) = e^{-j\frac{2\pi}{6}\cdot 3K} X(K)$ [$\because N = 6$]

By IDFT and using Circular time shifting property,

$y(n) = x((n-3))_6$

$\langle y(n) = \{3,4,0,0,1,2\} \rangle$

6) A length-8 sequence $x(n) = \{-4,5,2,-3,0,-2,3,4\}$ with 8-point DFT given by $X(K)$. Determine the sequence $y(n)$ whose 8-point DFT is given by $Y(K) = W_4^{3K} X(K)$.

Sol^n:- $Y(K) = W_4^{3K} X(K)$

$= e^{-j\frac{2\pi}{4\cdot 2}\cdot 3K} X(K)$

$Y(K) = e^{-j\frac{2\pi}{8}\cdot 6K} X(K)$

By IDFT and using Circular time shifting property,

$y(n) = x((n-6))_8$

$\langle \therefore y(n) = \{2,-3,0,-2,3,4,-4,5\} \rangle$

Scanned with CamScanner

7) Consider a length-12 sequence defined for $0 \le n \le 11$, $x(n) = \{8, 4, 7, -1, 2, 0, -2, -4, -5, 1, 4, 3\}$ with 12-point DFT $X(K)$; $0 \le K \le 11$. Evaluate the following function without computing the DFT, $\sum_{K=0}^{11} e^{-j\frac{4\pi}{6}K} X(K)$.

Soln :- From IDFT equation,

$$x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K)\, e^{\frac{j2\pi}{N}Kn}$$

$$x((n-4))_{12} = \frac{1}{12} \sum_{K=0}^{11} \left[e^{\frac{-j2\pi}{12}4K} X(K)\right] e^{\frac{j2\pi}{N}Kn}$$

put $n=0$,

$$x((0-4))_{12} = \frac{1}{12} \sum_{K=0}^{11} \left[e^{\frac{-j2\pi}{12}4K} X(K)\right] \cdot 1$$

$$\sum_{K=0}^{11} e^{\frac{-j4\pi}{6}K} X(K) = 12\, x((-4))_{12}$$

$$= 12\, x(12-4)$$

$$= 12\, x(8)$$

$$= 12(-5)$$

$$= -60$$

$$\sum_{K=0}^{11} e^{\frac{-j4\pi}{6}K} X(K) = \sum_{K=0}^{2} e^{\frac{-j4\pi}{3}K} X(K)$$

$$= \sum_{K=0}^{11} e^{\frac{-j2\pi}{12}\times 4K} X(K)$$

8) Let $X(K)$ be a 7-point DFT of a length-7 sequence $x(n)$ given by, $X(K) = \{1+j2, -2+j3, -1-j2, 0, 8+j4, -3+j, 2+j5\}$. If $y(n) = x((n-3))_7$, find $Y(K)$ without computing the DFT.

Soln :- $y(n) = x((n-3))_7$

By DFT, $Y(K) = W_7^{3K} X(K)$

$$Y(K) = e^{\frac{-j2\pi}{7}(3K)} X(K)$$

@ $K=0$, $Y(0) = X(0) = 1+j2$

@ $K=1$, $Y(1) = e^{\frac{-j2\pi}{7}(3)} X(1) = (-1)(-2+j3) = 2-j3$

$Y(2) = (0.623)X(2)$ [find $Y(K)$ by substituting $K = 0, 1, \ldots 6$]

$Y(3) = (-0.222)X(3) = 0$

$Y(4) = (-0.223)X(4) =$

$Y(5) = (0.624)X(5) =$

$Y(6) = (-0.9012)X(6) = -1.8024 - j4.506$

Scanned with CamScanner

9) Given that X(K) and Y(K) are the 6-point DFT of sequences x(n) and y(n) respectively. The $X(K) = \{1+j, -2.1+j3.2, -1.2-j2.4, 0, 0.9+j0.31, -0.3+j1.1\}$. The sequence x(n) and y(n) are related by, $y(n) = x((n-4))_6$. Determine Y(K) without computing the DFT.

Soln:- Given $y(n) = x((n-4))_6$

By DFT, $Y(K) = W_6^{4K} X(K)$ [using circular time shifting property]

$= e^{\frac{-j2\pi}{6}(4K)} X(K)$

$Y(0) = X(0) = 1+j$

$Y(1) = (-0.5 + j0.866)(-2.1+j3.2) = -1.738 - j3.4$

Similarly, $Y(2) = -1.4778 + j2.238$

$Y(3) = 0$

$Y(4) = -3.13 - j0.77$

$Y(5) = 1.1 - j0.29$

$Y(K) = \{1+j, -1.738 - j3.4, -1.4778 + j2.238, 0, -3.13 - j0.77, 1.1 - j0.29\}$

10) Consider the sequence, $x(n) = 4\delta(n) + 3\delta(n-1) + 2\delta(n-2) + \delta(n-3)$. Let X(K) be the six point DFT of x(n), find the finite length sequence y(n) that has six-point DFT, $Y(K) = W_6^{4K} X(K)$.

Soln:- $x(n) = \{4, 3, 2, 1\}$; Here N=6, $x(n) = \{4, 3, 2, 1, 0, 0\}$

Given $Y(K) = W_6^{4K} X(K)$

Taking IDFT using circular time shifting property,

$y(n) = x((n-4))_6$

$y(n) = \{2, 1, 0, 0, 4, 3\}$

Scanned with CamScanner

problem on circular frequency-shifting property:-

1) Consider two length-6 sequences $g(n)$ and $h(n)$ which has 6-point DFT $G(K)$ and $H(K)$ respectively. Let $g(n) = \{4.1, 3.5, 1.2, 5, 2, 3.3\}$. The DFT $G(K)$ and $H(K)$ are related by $H(K) = G((K-3))_6$. Determine $h(n)$ without computing IDFT.

Soln:- Given $H(K) = G((K-3))_6$

Taking 6-point IDFT on both the sides,

$$h(n) = e^{j\frac{2\pi}{6}n(3)} g(n) \quad \text{[using circular frequency shift property]}$$

$$= e^{j\pi n} g(n)$$

$$h(n) = (-1)^n g(n)$$

@ $n=0$, $h(0) = (-1)^0 g(0) = \underline{4.1}$

@ $n=1$, $h(1) = (-1)^1 g(1) = \underline{-3.5}$

@ $n=2$, $h(2) = (-1)^2 g(2) = \underline{1.2}$

@ $n=3$, $h(3) = (-1)^3 g(3) = \underline{-5}$

@ $n=4$, $h(4) = (-1)^4 g(4) = \underline{2}$

@ $n=5$, $h(5) = (-1)^5 g(5) = \underline{-3.3}$

$\therefore h(n) = \{4.1, -3.5, 1.2, -5, 2, -3.3\}$

2) Let $X(K)$ be a 7-point DFT of a length-7 sequence $x(n)$ is given by, $x(n) = \{-3.1, 2.4, 4.5, -6, 1, -3, 7\}$. If $Y(K) = X((K-4))_7$: find $y(n)$ without computing the IDFT.

Soln:- $y(n) = \text{IDFT}\{Y(K)\} = \text{IDFT}\{X((K-4))_7\}$

$$= e^{j\frac{2\pi}{7}.4n} x(n) \quad \text{[using circular frequency shifting property]}$$

$$= e^{j\frac{8\pi}{7}n} x(n)$$

Scanned with CamScanner

3) The 8-point DFT of a length-8 sequence $x(n)$ is given by, $X(K) = \{2, 1+j, 2-j4, 3+j2, 1, 3-j2, 2+j4, 1-j\}$. Evaluate the function, $\sum_{n=0}^{7} e^{j\pi n} x(n)$

Sol^n:- $\sum_{n=0}^{7} e^{j\pi n} x(n) = \sum_{n=0}^{7} e^{j\frac{2\pi}{8}4n} x(n)$

$DFT\{e^{j\frac{2\pi}{8}4n} x(n)\} = X((K-4))_8$

i.e., $X((K-4))_8 = \sum_{n=0}^{7} \left[e^{j\frac{2\pi}{8}4n} x(n)\right] e^{-j\frac{2\pi}{8}Kn}$

put $K=0$,

$$X((-4))_8 = \sum_{n=0}^{7} e^{j\pi n} x(n)$$

$\therefore \sum_{n=0}^{7} e^{j\pi n} x(n) = X(8-4) = X(4) = \underline{1}$

4) $g(n)$ and $h(n)$ are the two sequences of length-6 with 6-point DFT's $G(K)$ and $H(K)$ respectively. The sequence $g(n) = \{4,3,1,5,2,6\}$ The DFT's are related by Circular frequency shift of $H(K) = G((K-3))_6$. Determine $h(n)$ without Computing DFT and IDFT.

Sol^n:- Given $H(K) = G((K-3))_6$

Taking IDFT on both the sides using Circular frequency shift property,

$$h(n) = e^{-j\frac{2\pi}{6}3n} g(n)$$
$$= e^{j\pi n} g(n)$$
$$= (-1)^n g(n)$$

$\langle h(n) = \{4, -3, 1, -5, 2, -6\} \rangle$

Scanned with CamScanner

5) For $x(n) = \{7, 0, 8, 0\}$, find $y(n)$ if $Y(K) = X((K-2))_4$.

Soln:- Given $Y(K) = X((K-2))_4$

Taking IDFT using circular frequency shift property,

$$y(n) = e^{\frac{-j2\pi}{4} \times 2n} x(n)$$

$$= e^{j\pi n} x(n)$$

$$= (-1)^n x(n)$$

$$\langle y(n) = \{7, 0, 8, 0\} \rangle$$

Scanned with CamScanner

Problems on Circular time reversal property :—

1) Consider, a length-8 sequence $x(n) = \{1, 2, 3, 0, 6, -7, 8, -1\}$. Obtain $y(n) = x((-n))_8$.

Soln :- $x(n) = \{1, 2, 3, 0, 6, -7, 8, -1\}$

$y(n) = x((-n))_8$

$y(0) = x((-0))_8 = x(0) = \underline{1}$

$y(1) = x((-1))_8 = x(7) = \underline{-1}$

$y(2) = x((-2))_8 = x(6) = \underline{8}$

$y(3) = x((-3))_8 = x(5) = \underline{-7}$

$y(4) = x((-4))_8 = x(4) = \underline{6}$

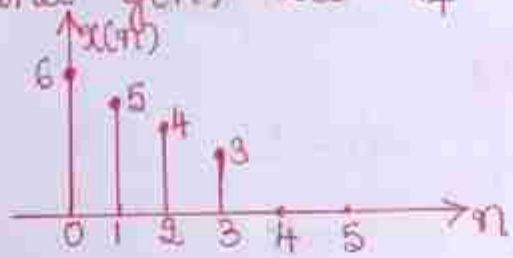
$y(5) = x((-5))_8 = x(3) = \underline{0}$

$y(6) = x((-6))_8 = x(2) = \underline{3}$

$y(7) = x((-7))_8 = x(1) = \underline{2}$

$\therefore y(n) = \{1, -1, 8, -7, 6, 0, 3, 2\}$

2) Fig below shows a length-4 sequence $x(n)$. Sketch the sequence $y(n) = x((-n))_4$



Soln :- Given $x(n) = \{6, 5, 4, 3\}$

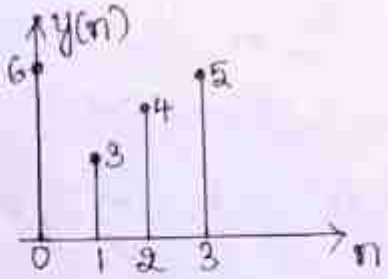
$y(0) = x((-0))_4 = x(0) = \underline{6}$

$y(1) = x((-1))_4 = x(3) = \underline{3}$

$y(2) = x((-2))_4 = x(2) = \underline{4}$

$y(3) = x((-3))_4 = x(1) = \underline{5}$

$\therefore y(n) = \{6, 3, 4, 5\}$



3) obtain the 4-point DFT of the sequence $x_1(n) = \{1, 1, 2, 3\}$ and thereby find the 4-point of the sequence $x_2(n) = \{1, 3, 2, 1\}$ using time reversal property.

Soln :-

a) $x_1(n) = \{1, 1, 2, 3\}$

Scanned with CamScanner

$X_1(K) = \sum_{n=0}^{N-1} x_1(n) W_N^{Kn}$; Here $N = 4$

$X_1(K) = 1 + W_4^K + 2W_4^{2K} + 3W_4^{3K} \longrightarrow (1)$

put $K=0$, $X_1(0) = 1+1+2+3 = \underline{7}$

@ $K=1$, $X_1(1) = 1 + W_4^1 + 2W_4^2 + 3W_4^3 = 1 - j + 2(-1) + 3(j) = \underline{-1+j2}$

@ $K=2$, $X_1(2) = 1 + W_4^2 + 2W_4^4 + 3W_4^6 = 1 - 1 + 2 - 3 = \underline{-1}$

@ $K=3$, $X_1(3) = 1 + W_4^3 + 2W_4^6 + 3W_4^9 = 1 + j - 2 - j3 = \underline{-1-j2}$

$\therefore X_1(K) = \{7, -1+j2, -1, -1-j2\}$

b) $x_2(n) = \{1, 3, 2, 1\}$

It is observed that, $x_2(n) = x_1((-n))_4$

Taking DFT on both the sides using time reversal property,

$X_2(K) = X_1((-K))_4$

$X_2(0) = X_1((-0))_4 = X_1(0) = \underline{7}$

$X_2(1) = X_1((-1))_4 = X(3) = \underline{-1-j2}$

$X_2(2) = X_1((-2))_4 = X(2) = \underline{-1}$

$X_2(3) = X_1((-3))_4 = X(1) = \underline{-1+j2}$

$\therefore X_2(K) = \{7, -1-j2, -1, -1+j2\}$

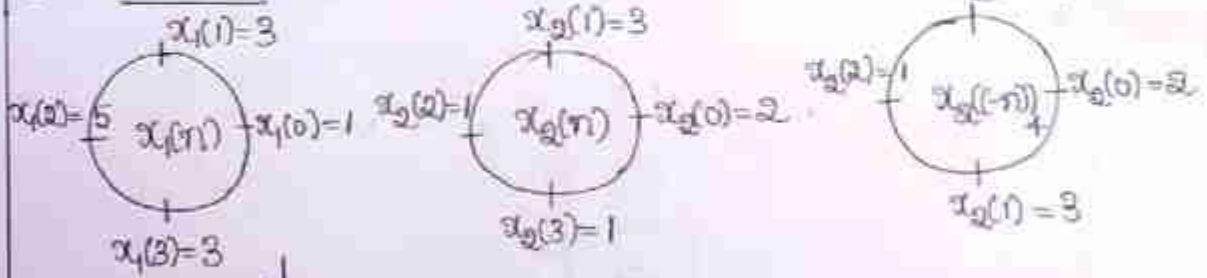
Scanned with CamScanner

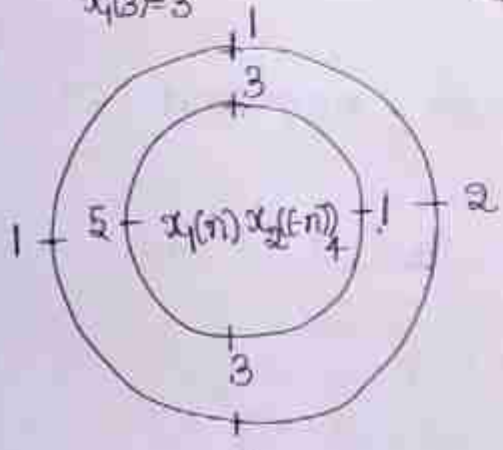
Problems on Circular Convolution :—

1) Determine the 4-point Circular Convolution $y(n)$ of the sequences $x_1(n) = \{1, 3, 5, 3\}$ and $x_2(n) = \{2, 3, 1, 1\}$. Use i) Concentric Circle method ii) Matrix method.

Soln :—

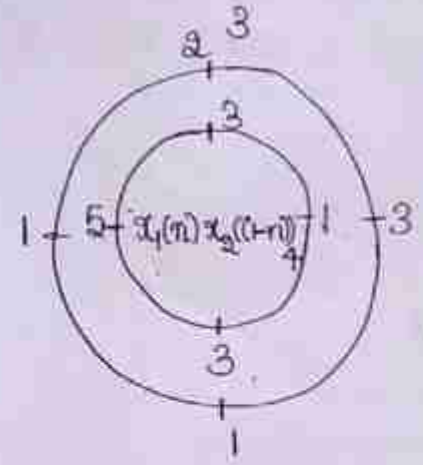
i) Concentric Circle method :—





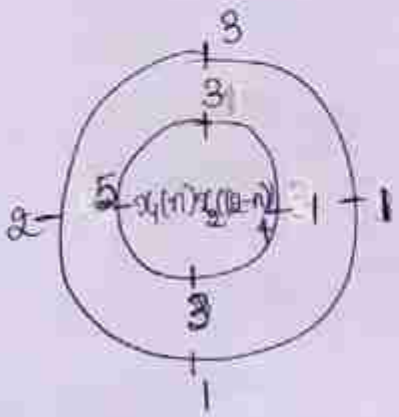
$y(0) = 1\times2 + 3\times1 + 5\times1 + 3\times3$

$\langle y(0) = 19 \rangle$



$y(1) = 1\times3 + 3\times2 + 5\times1 + 3\times1$

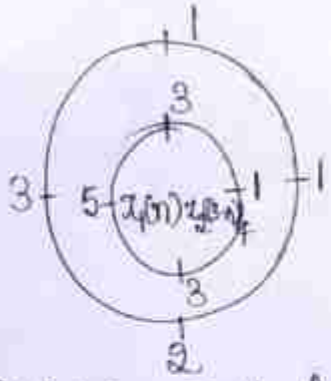
$\langle y(1) = 17 \rangle$



$y(2) = 1\times1 + 3\times3 + 5\times2 + 3\times1$

$\langle y(2) = 23 \rangle$

Scanned with CamScanner



$y(8) = 1\times1 + 3\times1 + 5\times3 + 3\times2$

$\langle y(3) = 25 \rangle$

ii) Matrix method :—

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 1 & 3 & 5 & 3 \\ 3 & 1 & 3 & 5 \\ 5 & 3 & 1 & 3 \\ 3 & 5 & 3 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 19 \\ 17 \\ 23 \\ 25 \end{bmatrix}$$

Note:— Circular and linear Convolution using DFT and IDFT is known as "Stockham's method".

2) Given a finite length sequence, $x(n) = \delta(n) + 2\delta(n-5)$. Determine a) the 10-point DFT of $x(n)$. b) the sequence $y(n)$ that has DFT $Y(K) = e^{j\frac{2\pi}{5}K} X(K)$. c) the length-10 sequence $y(n)$ that has DFT, $Y(K) = X(K)\,W(K)$, where $X(K)$ is the 10-point DFT of $x(n)$ and $W(K)$ is the 10-point DFT of $w(n)$ given by,

$$w(n) = \begin{cases} 1 \; ; & 0 \le n \le 6 \\ 0 \; ; & \text{otherwise.} \end{cases}$$

Sol^n:- $x(n) = \delta(n) + 2\delta(n-5)$

i.e., $x(n) = \{1, 0, 0, 0, 0, 2, 0, 0, 0, 0\}$; $N = 10$

i) The 10-point DFT of $x(n)$

$$X(K) = \sum_{n=0}^{9} x(n)\, e^{-j\frac{2\pi}{10}Kn} \quad ; 0 \le K \le 9$$

$$X(K) = 1 + 2e^{-j\pi K}$$

$$\langle X(K) = \{3, -1, 3, -1, 3, -1, 3, -1, 3, -1\} \rangle$$

Scanned with CamScanner

ii) Given $Y(K) = e^{\frac{j2\pi}{5}} X(K) = e^{-\frac{j2\pi(-2)}{10}} X(K)$

By IDFT,

$y(n) = x((n+2))_{10}$ [using Circular time Shifting property]

$\langle y(n) = \{0,0,0,2,0,0,0,0,1,0\} \rangle$

iii) $Y(K) = X(K)\,W(K)$

By IDFT, $y(n) = x(n) \circledast_{10} w(n)$ [using Circular Convolution property]

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 2 \\ 2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 3 \\ 3 \\ 1 \\ 1 \\ 1 \\ 3 \\ 3 \\ 2 \\ 2 \\ 2 \end{bmatrix}$$

Scanned with CamScanner

3) Consider the sequences $x_1(n) = \{0,1,0,0,0\}$ and $x_2(n) = \{0,1,2,3,4\}$. Determine the sequence $y(n)$ such that $Y(K) = X_1(K) X_2(K)$.

Soln:- Given $Y(K) = X_1(K) X_2(K)$

By IDFT, $y(n) = x_1(n) \textcircled{5} x_2(n)$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \\ 1 \\ 2 \\ 3 \end{bmatrix}$$

Scanned with CamScanner

Problems on parseval's theorem :—

1) Consider a length-6 sequence $x(n) = \{1, 3, -2, 1, -3, 4\}$ with 6-point DFT given by $X(K)$. Evaluate the function given by,

$$\sum_{K=0}^{5} |X(K)|^2$$

Sol^n :- From parseval's theorem,

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{K=0}^{N-1} |X(K)|^2$$

Here $N = 6$,

$$\sum_{n=0}^{5} |x(n)|^2 = \frac{1}{6} \sum_{K=0}^{5} |X(K)|^2$$

$$\sum_{K=0}^{5} |X(K)|^2 = 6 \sum_{n=0}^{5} |x(n)|^2$$

$$= 6[1 + 9 + 4 + 1 + 9 + 16]$$

$$= 240$$

2) A length-12 sequence is given by, $x(n) = \{3, -1, 2, 4, -3, -2, 0, 1, -4, 6, 2, 5\}$ with 12-point DFT given by $X(K)$. Evaluate the following functions of $X(K)$ without computing the DFT.

i) $X(0)$ ii) $X(6)$ iii) $\sum_{K=0}^{11} X(K)$ iv) $\sum_{K=0}^{11} e^{-j\frac{4\pi}{6}K} X(K)$ v) $\sum_{K=0}^{11} |X(K)|^2$

Sol^n :- $X(K) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}Kn} \longrightarrow (1)$

i) Given $N = 12$

put $K = 0$ in Eq(1),

$$X(0) = \sum_{n=0}^{11} x(n)$$

$$= [x(0) + x(1) + x(2) + \ldots\ldots + x(11)]$$

$$= (3 - 1 + 2 + 4 - 3 - 2 + 0 + 1 - 4 + 6 + 2 + 5)$$

$$= 13$$

Scanned with CamScanner

ii) put $K=6$ in Eq(1),

$$X(6) = \sum_{n=0}^{11} x(n)\, e^{-j\frac{2\pi}{12}.6n}$$

$$= \sum_{n=0}^{11} (-1)^n x(n)$$

$$= [3+1+2-4-3+2+0-1-4-6+2-5]$$

$$= -13$$

iii) $$x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K)\, e^{j\frac{2\pi}{N}Kn}$$

Given $N=12$,

$$x(n) = \frac{1}{12} \sum_{K=0}^{11} X(K)\, e^{j\frac{2\pi}{12}Kn}$$

put $n=0$, $x(0) = \frac{1}{12} \sum_{K=0}^{11} X(K)$

$$\therefore \sum_{K=0}^{11} X(K) = 12.\, x(0) = 12 \times 3 = 36$$

iv) $e^{-j\frac{4\pi}{6}K} X(K) = e^{-j\frac{2\pi}{12}\times 4K} \xleftrightarrow{\text{IDFT}} x((n-4))_{12}$

$$x((n-4))_{12} = \frac{1}{12} \sum_{K=0}^{11} \left[e^{-j\frac{4\pi}{6}K} X(K)\right] e^{j\frac{2\pi}{12}Kn}$$

put $n=0$,

$$x((-4))_{12} = \frac{1}{12} \sum_{K=0}^{11} e^{-j\frac{4\pi}{6}K} X(K)$$

$$\therefore \sum_{K=0}^{11} e^{-j\frac{4\pi}{6}K} X(K) = 12\, x((-4))_{12} = 12\, x(12-4)$$

$$= 12\, x(8)$$

$$= 12(-4)$$

$$= -48$$

v) From parseval's theorem,

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{K=0}^{N-1} |X(K)|^2$$

Scanned with CamScanner

Here $N=12$, $\sum_{n=0}^{11}|x(n)|^2 = \frac{1}{12}\sum_{K=0}^{11}|X(K)|^2$

$\therefore \sum_{K=0}^{11}|X(K)|^2 = 12\sum_{n=0}^{11}|x(n)|^2$

$= 12[9+1+4+16+9+4+0+1+16+36+4+25]$

$= 1500$

3) Let $X(K)$ be a 14-point DFT of a length-14 real-valued sequence $x(n)$. The first 8 samples of $X(K)$ are given by,

$X(0)=12$; $X(1)=-1+j3$; $X(2)=3+j4$; $X(3)=1-j5$; $X(4)=-2+j2$;

$X(5)=6+j3$; $X(6)=-2-j3$; $X(7)=10$.

Determine the remaining samples of $X(K)$. Evaluate the following functions of $x(n)$ without computing the IDFT of $X(K)$.

i) $x(0)$ ii) $x(7)$ iii) $\sum_{n=0}^{13}x(n)$ iv) $\sum_{n=0}^{13}e^{\frac{j4\pi}{7}n}x(n)$ v) $\sum_{n=0}^{13}|x(n)|^2$

Sol^n:-

$X(K) = X^*(N-K)$

$X(8)=-2+j3$; $X(9)=6-j3$; $X(10)=-2-j2$; $X(11)=1+j5$; $X(12)=3-j4$

$X(13)=-1-j3$.

i) W.K.T $x(n) = \frac{1}{N}\sum_{K=0}^{N-1}X(K)\,e^{\frac{j2\pi}{N}Kn}$; $0\le n\le(N-1)$

Here $N=14$, $x(n) = \frac{1}{14}\sum_{K=0}^{13}X(K)\,e^{\frac{j\pi}{7}Kn}$

put $n=0$; $x(0) = \frac{1}{14}\sum_{K=0}^{13}X(K)$

$= \frac{1}{14}[X(0)+X(1)+X(2)+\cdots\cdots+X(13)]$

$= \frac{32}{14} = 2.2857$

ii) W.K.T $x(n) = \frac{1}{14}\sum_{K=0}^{13}X(K)\,e^{\frac{j2\pi}{14}Kn}$

put $n=7$; $x(7) = \frac{1}{14}\sum_{K=0}^{13}(-1)^K X(K) = \frac{-12}{14} = -0.8571$

Scanned with CamScanner

iii) $X(K) = \sum_{n=0}^{13} x(n) e^{-j\frac{2\pi}{14}Kn} = \sum_{n=0}^{13} x(n) e^{-j\frac{\pi}{7}Kn}$

put $K=0$, $X(0) = \sum_{n=0}^{13} x(n)$

$$\sum_{n=0}^{13} x(n) = X(0) = \underline{12}$$

iv) DFT $\{ e^{j\frac{4\pi}{7}n} x(n) \} = e^{j\frac{2\pi}{14}.4n} x(n)$ is $X((K-4))_{14}$

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}Kn}$

$\therefore X((K-4))_{14} = \sum_{n=0}^{13} e^{j\frac{4\pi}{7}n} x(n) e^{-j\frac{2\pi}{14}Kn}$

put $K=0$, $X((-4))_{14} = \sum_{n=0}^{13} e^{j\frac{4\pi}{7}n} x(n)$

$$\sum_{n=0}^{13} e^{j\frac{4\pi}{7}n} x(n) = X(14-4) = X(10) = \underline{-2-j2}$$

v) From parseval's theorem,

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{K=0}^{N-1} |X(K)|^2$$

Here $N=14$,

$$\sum_{n=0}^{13} |x(n)|^2 = \frac{1}{14} \sum_{K=0}^{13} |X(K)|^2$$

$$= \frac{1}{14} [|X(0)|^2 + |X(1)|^2 + \ldots + |X(13)|^2]$$

$$= \frac{498}{14} = 35.5714$$

4) Consider a length-12 sequence defined for $0 \le n \le 11$, $x(n) = \{8, 4, 7, -1, 2, 0, -2, -4, -5, 1, 4, 3\}$ with 12-point DFT given by $X(K)$, $0 \le K \le 11$, Evaluate the following function without computing DFT, $\sum_{K=0}^{11} e^{-j\frac{4\pi K}{6}} X(K)$

Scanned with CamScanner

Sol^n:- W.K.T $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K)\, e^{\frac{j2\pi}{N}Kn}$

Here $N=12$, $x(n) = \frac{1}{12}\sum_{K=0}^{11} X(K)\, e^{\frac{j2\pi}{12}Kn}$

$DFT\left\{e^{-\frac{j2\pi}{12}\times 4K} X(K)\right\} = x((n-4))_{12}$

$x((n-4))_{12} = \frac{1}{12}\sum_{K=0}^{11}\left\{e^{-\frac{j4\pi K}{6}} X(K)\right\} e^{\frac{j2\pi}{12}Kn}$

put $n=0$,

$x((-4))_{12} = \frac{1}{12} \times \sum_{K=0}^{11} e^{-\frac{j4\pi K}{6}} X(K)$

$\sum_{K=0}^{11} e^{-\frac{j4\pi K}{6}} X(K) = 12\, x((-4))_{12} = 12\, x(8) = 12(-5) = -60$

5) Consider a sequence $x(n) = \{8, 3, 4, 1, -5, -4, -2, 0, 2, -1, 7, 4\}$. Evaluate the following without explicitly computing $X(K)$.

i) $DFT[DFT[DFT[DFT[x(n)]]]]$

ii) $\sum_{K=0}^{11} X(K)$ iii) $\sum_{K=0}^{11} e^{-\frac{j\pi K}{6}} X(K)$

Sol^n:-

i) W.K.T $DFT[DFT[DFT[DFT[x(n)]]]] = N^2 x(n)$

$= (12)^2 x(n)$

ii) $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K)\, e^{-\frac{j2\pi}{N}Kn}$

Here $N=12$,

$x(n) = \frac{1}{12}\sum_{K=0}^{11} X(K)\, e^{-\frac{j2\pi}{12}Kn}$

put $n=0$;

$x(0) = \frac{1}{12}\sum_{K=0}^{11} X(K)$

$\therefore \sum_{K=0}^{11} X(K) = 12\, x(0) = 12(8) = 96$

Scanned with CamScanner

iii) $x(n) = \frac{1}{12}\sum_{K=0}^{11} X(K)\, e^{+j\frac{2\pi}{12}Kn}$

put $n=-1$,

$$x(-1) = \frac{1}{12}\sum_{K=0}^{11} X(K)\, e^{-j\frac{\pi}{6}K} = x((-1))_{12} = x(11) = 4$$

$$\therefore \sum_{K=0}^{11} X(K)\, e^{-j\frac{\pi}{6}K} = 12 \times 4 = \underline{48}$$

6) For $x(n) = \{1, -2, 3, -4, 5, -6\}$, without computing its DFT, find the following i) $X(0)$ ii) $\sum_{K=0}^{5} X(K)$ iii) $X(3)$ iv) $\sum_{K=0}^{5} |X(K)|^2$ v) $\sum_{K=0}^{5} (-1)^K X(K)$

Soln :- i) $X(K) = \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn}$

Here $N=6$,

$$X(K) = \sum_{n=0}^{5} x(n)\, e^{-j\frac{2\pi}{6}Kn} \longrightarrow (1)$$

put $K=0$,

$$X(0) = \sum_{n=0}^{5} x(n) = 1-2+3-4+5-6 = \underline{-3}$$

ii) $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K)\, e^{j\frac{2\pi}{N}Kn}$

Here $N=6$, $x(n) = \frac{1}{6}\sum_{K=0}^{5} X(K)\, e^{j\frac{2\pi}{6}Kn} \longrightarrow (2)$

put $n=0$,

$$x(0) = \frac{1}{6}\sum_{K=0}^{5} X(K)$$

$$\therefore \sum_{K=0}^{5} X(K) = 6.x(0) = 6(1) = \underline{6}$$

iii) put $K=3$ in Eq(1),

$$X(3) = \sum_{n=0}^{5} x(n)\, e^{-j\frac{\pi}{3}\times 3n} = \sum_{n=0}^{5} x(n)(-1)^n = 1+2+3+4+5+6$$

$$= \underline{21}$$

Scanned with CamScanner

iv) From parseval's theorem,

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{K=0}^{N-1} |X(K)|^2$$

Here $N=6$,

$$\sum_{n=0}^{5} |x(n)|^2 = \frac{1}{6} \sum_{K=0}^{5} |X(K)|^2$$

$$\therefore \sum_{K=0}^{5} |X(K)|^2 = 6 \sum_{n=0}^{5} |x(n)|^2$$

$$= 6[1+4+9+16+25+36]$$

$$= 546$$

7) Let $x(n) = \{1, 2, 0, 3, -2, 4, 7, 5\}$. Evaluate the following:

i) $X(0)$ ii) $X(4)$ iii) $\sum_{K=0}^{4} X(K)$ iv) $\sum_{K=0}^{7} |X(K)|^2$

Sol^n:-

$$X(K) = \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}Kn}$$

Here $N=8$,

$$X(K) = \sum_{n=0}^{7} x(n)\, e^{-j\frac{2\pi}{8}Kn} \longrightarrow (1)$$

i) put $K=0$ in Eq(1),

$$X(0) = \sum_{n=0}^{7} x(n)$$

$$= 1+2+0+3-2+4+7+5$$

$$= 20$$

ii) put $K=4$ in Eq(1),

$$X(4) = \sum_{n=0}^{7} x(n)\, e^{-j\frac{\pi}{4}\times 4n} = \sum_{n=0}^{7} (-1)^n x(n)$$

$$= 1-2+0-3-2-4+7-5$$

$$= -8$$

Scanned with CamScanner

iii) $x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) e^{j\frac{2\pi}{N}Kn}$

Here $N = 8$,

$$x(n) = \frac{1}{8} \sum_{K=0}^{7} X(K) e^{j\frac{2\pi}{8}Kn}$$

@ $n=0$,

$$x(0) = \frac{1}{8} \sum_{K=0}^{7} X(K)$$

$$\sum_{K=0}^{7} X(K) = 8\, x(0) = 8(1) = 8$$

iv) From parseval's theorem,

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{K=0}^{N-1} |X(K)|^2$$

Here $N=8$,

$$\sum_{n=0}^{7} |x(n)|^2 = \frac{1}{8} \sum_{K=0}^{7} |X(K)|^2$$

$$\therefore \sum_{K=0}^{7} |X(K)|^2 = 8 \sum_{n=0}^{7} |x(n)|^2 = 8 \times 108 = 864$$

Scanned with CamScanner

Module-1:

Anusha.M.N
Asst. professor
Dept. of ECE
BGSIT, B.G. Nagar

Properties of DFT

Problems on Linearity property:-

1) obtain the 4-point DFT of the following length 4-sequences.
a) $x_1(n) = \{1, 2, 0, 3\}$ b) $x_2(n) = \{0.75, 1.5, 0, 2.25\}$

Soln:-

a) $x_1(n) = \{1, 2, 0, 3\}$; $N = 4$

W.K.T $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$; $0 \le K \le (N-1)$

$$X_1(K) = 1 + 2W_4^{K} + 3W_4^{3K} \longrightarrow (1)$$

$W_4^0 = 1$
$W_4^1 = -j$
$W_4^2 = -1$
$W_4^3 = j$

@ $K = 0$, $X_1(0) = 1 + 2W_4^0 + 3W_4^0 = 1 + 2 + 3 = \underline{6}$

@ $K = 1$, $X_1(1) = 1 + 2W_4^1 + 3W_4^3 = 1 + 2(-j) + 3(+j) = \underline{1 + j1}$

@ $K = 2$, $X_1(2) = 1 + 2W_4^2 + 3W_4^6 = 1 + 2(-1) + 3(-1) = \underline{-4}$

@ $K = 3$, $X_1(3) = 1 + 2W_4^3 + 3W_4^9 = 1 + 2(j) + 3(-j) = \underline{1 - j1}$

$\therefore X_1(K) = \{6, 1+j1, -4, 1-j1\}$

b) $x_2(n) = \{0.75, 1.5, 0, 2.25\}$

By inspection,

$x_2(n) = 0.75\, x_1(n)$

From linearity property,

$X_2(K) = 0.75\, X_1(K)$

$X_2(K) = \{4.5, 0.75+j1.75, -3, 0.75-j1.75\}$

2) obtain the 4-point DFT of the following sequences:
a) $x_1(n) = \{1, 1, 2, 2\}$ b) $x_2(n) = \{1, 0, 0, 0\}$ c) $x_3(n) = \{0.5, 0.5, 1, 1\}$

Scanned with CamScanner

Sol^n:-

a) $x_1(n) = \{1, 1, 2, 2\}$

$$X_1(K) = \sum_{n=0}^{N-1} x_1(n) W_N^{Kn}$$

Here $N = 4$,

$$X_1(K) = 1 + W_4^{K} + 2W_4^{2K} + 2W_4^{3K} \longrightarrow (1)$$

@$K=0$, $X_1(0) = 4$; @$K=1$, $X_1(1) = -1+j$

@$K=2$, $X_1(2) = 0$; @$K=3$, $X_1(3) = -1-j$

$\langle \therefore X_1(K) = \{4, -1+j, 0, -1-j\} \rangle$

b) $x_2(n) = \{1, 0, 0, 0\}$

$$X_2(K) = \sum_{n=0}^{N-1} x_2(n) W_N^{Kn}$$

$X_2(K) = 1$

$\langle \therefore X_2(K) = \{1, 1, 1, 1\} \rangle$

c) $x_3(n) = \{0.5, 0.5, 1, 1\}$

By inspection,

$x_3(n) = 0.5\, x_1(n) + 5\, x_2(n)$

From linearity property,

$X_3(K) = 0.5\, X_1(K) + 5\, X_2(K)$

$= \{3, -0.5 + j0.5, 0, -0.5 - j0.5\} + \{5, 5, 5, 5\}$

$\langle X_3(K) = \{8, 4.5 + j0.5, 5, 4.5 - j0.5\} \rangle$

8) Find the 4-point DFT of the sequence : $x(n) = \cos\left(\frac{\pi}{4}n\right) + \sin\left(\frac{\pi}{4}n\right)$ Use linearity property.

Sol^n:- Let $x_1(n) = \cos\left(\frac{\pi}{4}n\right)$ and $x_2(n) = \sin\left(\frac{\pi}{4}n\right)$

$x_1(n) = \{1, 0.7071, 0, -0.7071\}$ $x_2(n) = \{0, 0.7071, 1, 0.7071\}$

Scanned with CamScanner

$X_1(K) = \sum_{n=0}^{N-1} x_1(n) W_N^{Kn}$

$X_1(K) = 1 + 0.707 W_8^{K} - 0.707 W_8^{3K} \longrightarrow (1)$

$\langle X_1(K) = \{1, 1-j1.4142, 1, 1+j1.4142\} \rangle$

$X_2(K) = \sum_{n=0}^{N-1} x_2(n) W_N^{Kn}$

$X_2(K) = 0.707 W_8^{K} + W_8^{2K} + 0.707 W_8^{3K}$ $\longrightarrow (2)$

$\langle X_2(K) = \{2.4142, -1, -0.4142, -1\} \rangle$

using linearity property,

$X(K) = X_1(K) + X_2(K)$

$\langle X(K) = \{3.4142, -j1.4142, 0.586, j1.4142\} \rangle$

Scanned with CamScanner

1) Develop a decimation-in-time algorithm for evaluating the DFT for N=6.

Soln:-

Here N=6, which is not the power of 2. Hence, it is Composite number.

Let us Consider that the 6-point sequence is given by,

$x(n) = \{x(0), x(1), x(2), x(3), x(4), x(5)\}$.

Let us represent N by its factors i.e., $N = 3 \times 2$.

Here $N = p_1 \cdot q_1$, hence $p_1 = 3$ and $q_1 = 2$. This means $x(n)$ will be divided into 3 sequences with each sequence having 2 samples.

$f_1(n) = x(3n) = \{x(0), x(3)\}$

$f_2(n) = x(3n+1) = \{x(1), x(4)\}$

$f_3(n) = x(3n+2) = \{x(2), x(5)\}$

Let the DFT's of the above sequences be,

$f_1(n) \xleftarrow[q_1]{DFT} F_1(K)$, $f_2(n) \xleftarrow[q_1]{DFT} F_2(K)$, $f_3(n) \xleftarrow[q_1]{DFT} F_3(K)$

These q_1-point DFT's are Combined as follows:

$$X(K) = F_1(K) + W_N^K F_2(K) + W_N^{2K} F_3(K) \; ; \; 0 \le K \le 5$$

@K = 0, $X(0) = F_1(0) + W_6^0 F_2(0) + W_6^0 F_3(0)$

@K = 1, $X(1) = F_1(1) + W_6^1 F_2(1) + W_6^2 F_3(1)$

@K = 2, $X(2) = F_1(2) + W_6^2 F_2(2) + W_6^4 F_3(2)$

@K = 3, $X(3) = F_1(3) + W_6^3 F_2(3) + W_6^6 F_3(3)$

@K = 4, $X(4) = F_1(4) + W_6^4 F_2(4) + W_6^8 F_3(4)$

@K = 5, $X(5) = F_1(5) + W_6^5 F_2(5) + W_6^{10} F_3(5)$

W.K.T $F_1(K)$, $F_2(K)$ and $F_3(K)$ are q_1-point DFT's. Here $q_1 = 2$, Hence, they are 2-point DFT's.

Scanned with CamScanner

Hence,

$$F_1(K) = F_1(K+2) = F_1(K+4) = \cdots$$
$$F_2(K) = F_2(K+2) = F_2(K+4) = \cdots$$
$$F_3(K) = F_3(K+2) = F_3(K+4) = \cdots$$

} By periodicity property

$\therefore$ $F_1(0) = F_1(2) = F_1(4)$ $\quad$ $F_2(0) = F_2(2) = F_2(4)$

$F_1(1) = F_1(3) = F_1(5)$ $\quad$ $F_2(1) = F_2(3) = F_2(5)$

and $F_3(0) = F_3(2) = F_3(4)$

$F_3(1) = F_3(3) = F_3(5)$

$\therefore$ $X(0) = F_1(0) + F_2(0) + F_3(0)$

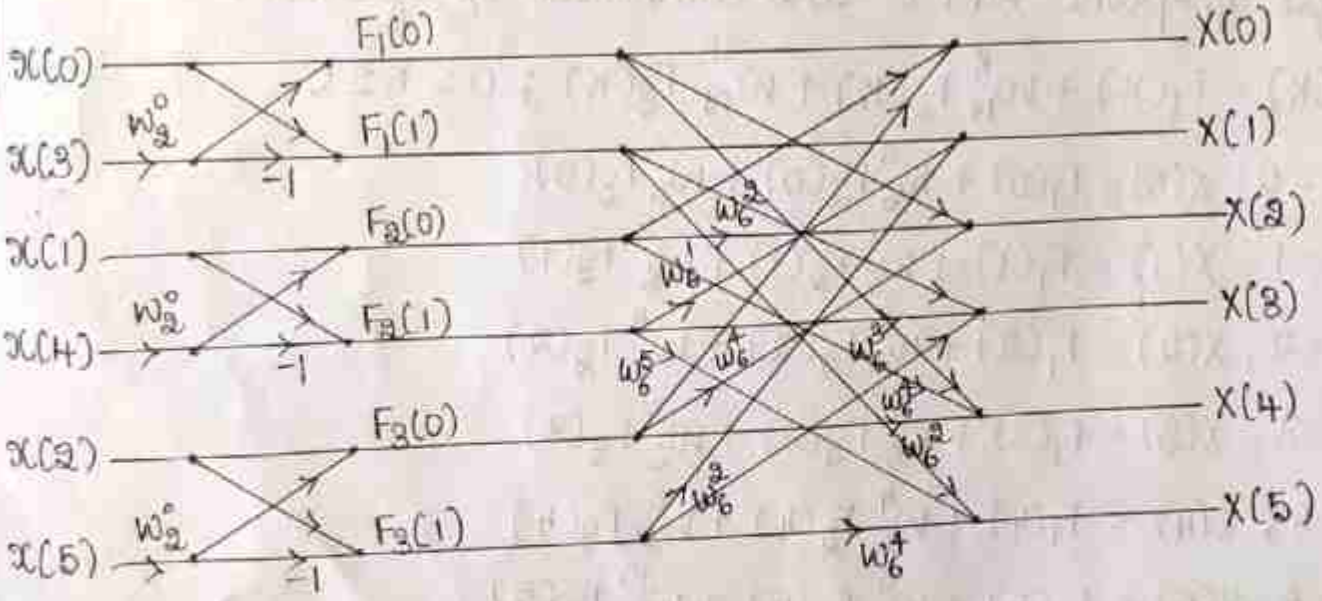
$X(1) = F_1(1) + W_6^1 F_2(1) + W_6^2 F_3(1)$

$X(2) = F_1(0) + W_6^2 F_2(0) + W_6^4 F_3(0)$

$X(3) = F_1(1) + W_6^3 F_2(1) + F_3(1)$

$X(4) = F_1(0) + W_6^4 F_2(0) + W_6^2 F_3(0)$

$X(5) = F_1(1) + W_6^5 F_2(1) + W_6^4 F_3(1)$



Scanned with CamScanner

2) Develop DIT-FFT algorithm for $N = 9 = 3 \times 3$ and draw the Complete signal flowgraph.
A nine point real-valued sequence is given by $x(n) = \{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1, \frac{3}{4}, \frac{1}{2}, \frac{1}{4}, 0\}$. Find DFT of the sequence using DIT-FFT algorithm, clearly indicating the signal flowgraph.

Sol^n:-

Here, the length of the sequence is $N = 9$. It can be factored as, $N = 3 \times 3$.

Hence, $N = p_1 \cdot q_1$ i.e., $p_1 = 3$ and $q_1 = 3$. This means the sequence $x(n)$ will be divided into 3 sequences with each sequence having 3 samples.

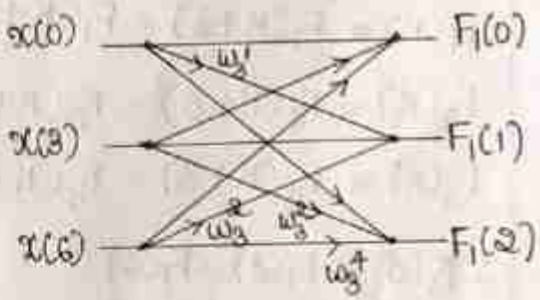
i.e., $f_1(n) = x(3n) = \{x(0), x(3), x(6)\}$

$f_2(n) = x(3n+1) = \{x(1), x(4), x(7)\}$

$f_3(n) = x(3n+2) = \{x(2), x(5), x(8)\}$

Calculation of $F_1(K)$, $F_2(K)$ and $F_3(K)$:—

$F_1(K) = x(0) + W_3^K x(3) + W_3^{2K} x(6)$



@K=0, $F_1(0) = x(0) + x(3) + x(6)$

@K=1, $F_1(1) = x(0) + W_3^1 x(3) + W_3^2 x(6)$

@K=2, $F_1(2) = x(0) + W_3^2 x(3) + W_3^4 x(6)$

$F_2(K) = x(1) + W_3^K x(4) + W_3^{2K} x(7)$

@K=0, $F_2(0) = x(1) + W_3^0 x(4) + W_3^0 x(7)$

@K=1, $F_2(1) = x(1) + W_3^1 x(4) + W_3^2 x(7)$

@K=2, $F_2(2) = x(1) + W_3^2 x(4) + W_3^4 x(7)$

$F_3(K) = x(2) + W_3^K x(5) + W_3^{2K} x(8)$

@K=0, $F_3(0) = x(2) + x(5) + x(8)$

@K=1, $F_3(1) = x(2) + W_3^1 x(5) + W_3^2 x(8)$

@K=2, $F_3(2) = x(2) + W_3^2 x(5) + W_3^4 x(8)$

Scanned with CamScanner

Combining three 3-point DFT's :-

$$X(K) = F_1(K) + W_9^K F_2(K) + W_9^{2K} F_3(K) \; ; \; 0 \le K \le 8$$

@K=0, $X(0) = F_1(0) + F_2(0) + F_3(0)$

@K=1, $X(1) = F_1(1) + W_9^1 F_2(1) + W_9^2 F_3(1)$

@K=2, $X(2) = F_1(2) + W_9^2 F_2(2) + W_9^4 F_3(2)$

@K=3, $X(3) = F_1(3) + W_9^3 F_2(3) + W_9^6 F_3(3)$

@K=4, $X(4) = F_1(4) + W_9^4 F_2(4) + W_9^8 F_3(4)$

@K=5, $X(5) = F_1(5) + W_9^5 F_2(5) + W_9^{10} F_3(5)$

@K=6, $X(6) = F_1(6) + W_9^6 F_2(6) + W_9^{12} F_3(6)$

@K=7, $X(7) = F_1(7) + W_9^7 F_2(7) + W_9^{14} F_3(7)$

@K=8, $X(8) = F_1(8) + W_9^8 F_2(8) + W_9^{16} F_3(8)$

$F_1(K)$, $F_2(K)$ and $F_3(K)$ are 8-point DFT's. Hence, by periodicity property :

$F_1(K) = F_1(K+3) = F_1(K+6) = \cdots$

$F_2(K) = F_2(K+3) = F_2(K+6) = \cdots$

$F_3(K) = F_3(K+3) = F_3(K+6) = \cdots$

$\therefore F_1(0) = F_1(3) = F_1(6)$ $\quad F_2(0) = F_2(3) = F_2(6)$

$F_1(1) = F_1(4) = F_1(7)$ $\quad F_2(1) = F_2(4) = F_2(7)$

$F_1(2) = F_1(5) = F_1(8)$ $\quad F_2(2) = F_2(5) = F_2(8)$

and

$F_3(0) = F_3(3) = F_3(6)$

$F_3(1) = F_3(4) = F_3(7)$

$F_3(2) = F_3(5) = F_3(8)$

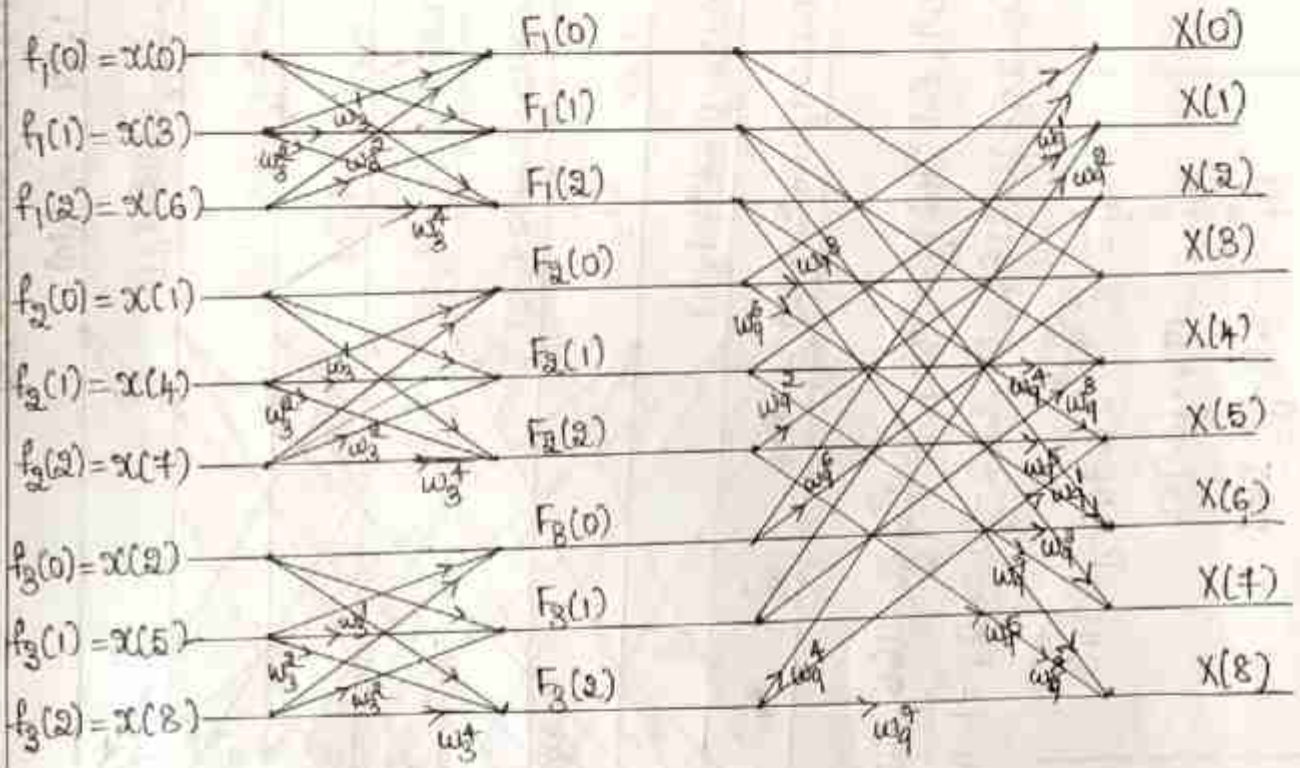
$\therefore X(0) = F_1(0) + F_2(0) + F_3(0)$

$X(1) = F_1(1) + W_9^1 F_2(1) + W_9^2 F_3(1)$

$X(2) = F_1(2) + W_9^2 F_2(2) + W_9^4 F_3(2)$

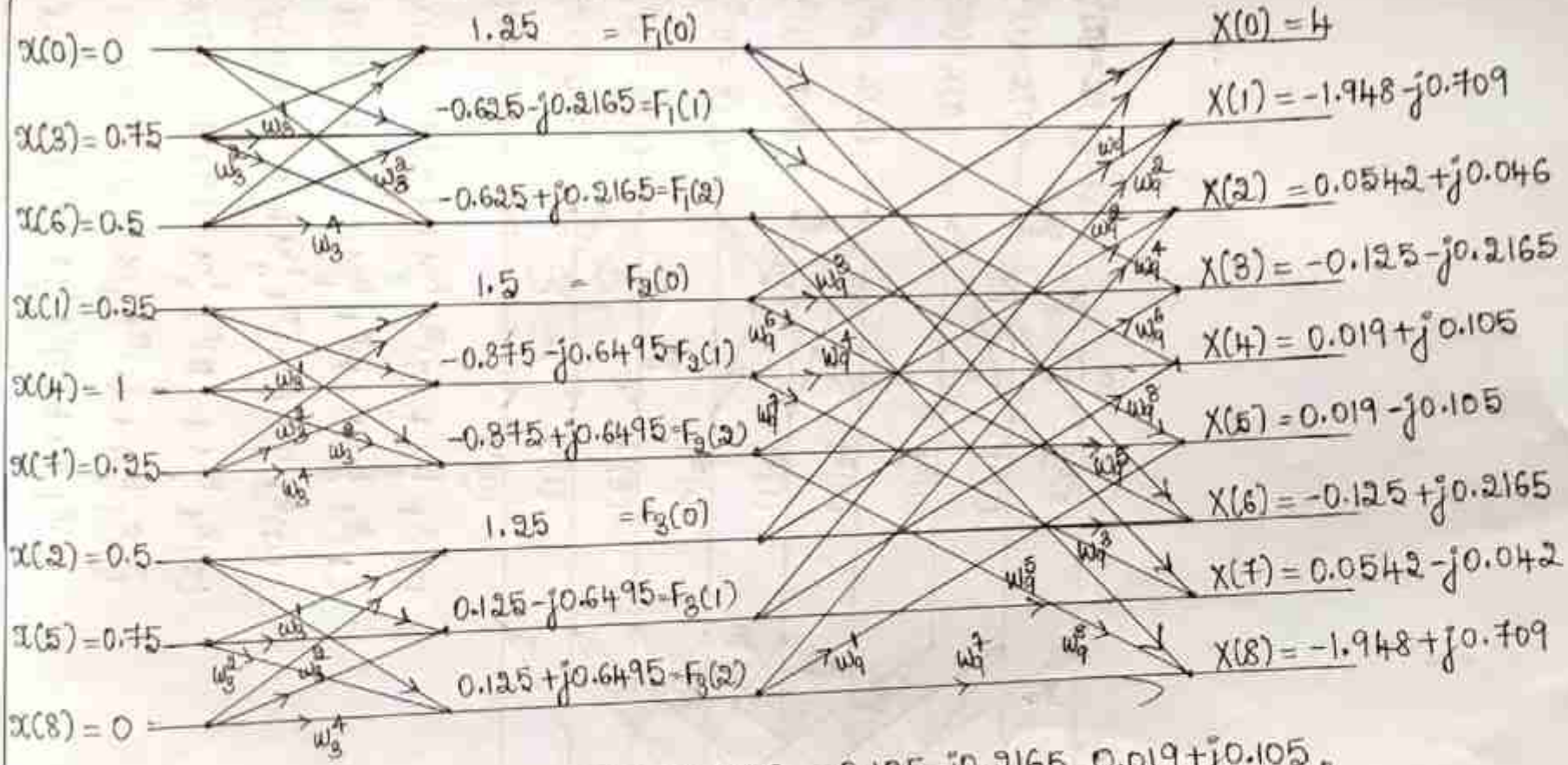
Scanned with CamScanner

$$X(3) = F_1(0) + W_9^3 F_2(0) + W_9^6 F_3(0)$$
$$X(4) = F_1(1) + W_9^4 F_2(1) + W_9^8 F_3(1)$$
$$X(5) = F_1(2) + W_9^5 F_2(2) + W_9^1 F_3(2)$$
$$X(6) = F_1(0) + W_9^6 F_2(0) + W_9^3 F_2(0)$$
$$X(7) = F_1(1) + W_9^7 F_2(1) + W_9^5 F_3(1)$$
$$X(8) = F_1(2) + W_9^8 F_2(2) + W_9^7 F_3(2)$$



Scanned with CamScanner

$x(n) = \{0, 0.25, 0.5, 0.75, 1, 0.75, 0.5, 0.25, 0\}$



$\langle X(K) = \{4, -1.948 - j0.709, 0.0542 + j0.046, -0.125 - j0.2165, 0.019 + j0.105, 0.019 - j0.105, -0.125 + j0.2165, 0.0542 - j0.042, -1.948 + j0.709\} \rangle$

$W_3^1 = e^{\frac{-j2\pi}{3}.1} = -0.5 - j0.866$

$W_3^2 = -0.5 + j0.866$

$W_3^4 = -0.5 - j0.866$

$W_9^1 = e^{\frac{-j2\pi}{9}.1} = 0.766 - j0.643$

$W_9^2 = 0.174 - j0.985$

$W_9^3 = -0.5 - j0.866$

$W_9^4 = -0.9397 - j0.342$

$W_9^5 = -0.9397 + j0.342$

$W_9^6 = -0.5 + j0.866$

$W_9^7 = 0.174 + j0.985$

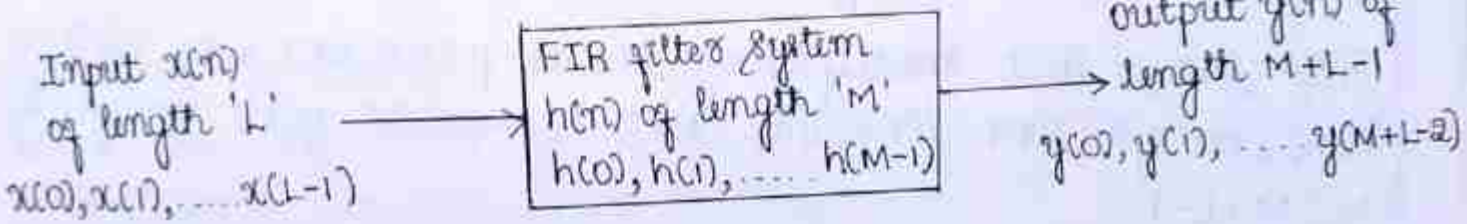
$W_9^8 = 0.766 + j0.643$

Scanned with CamScanner

Applications of DFT :-

① Use of DFT in linear filtering :-

Linear filtering operation is implemented with the help of linear convolution. The output $y(n)$ is obtained by convolving impulse response $h(n)$ with input $x(n)$.
Linear convolution can be computed efficiently with the help of DFT.
Let the unit sample response of the LTI system be $h(n)$ of length 'M' i.e., $h(0), h(1), \ldots, h(M-1)$.
Let the input to this LTI system be $x(n)$ of length 'L' i.e., $x(0), x(1), \ldots x(L-1)$. Such system is shown in fig below:



The output of the LTI system as shown in above fig is $y(n)$. This output $y(n)$ is the linear convolution of input $x(n)$ and unit sample response $h(n)$.

$$\text{i.e., } y(n) = \sum_{K=-\infty}^{\infty} h(K)\, x(n-K) \longrightarrow (1)$$

Since, the length of $h(n)$ is 'M' and that of $x(n)$ is 'L', the length of $y(n)$ will be $M+L-1$.
Consider the fourier transform of eq(1),

$$Y(w) = F\left\{ \sum_{K=-\infty}^{\infty} h(K)\, x(n-K) \right\} \longrightarrow (2)$$

Convolution property of Fourier transform states that,

$$F\{x_1(n) * x_2(n)\} = X_1(w)\, X_2(w)$$

Hence, R.H.S of eq(2) can be written as,

$$Y(w) = H(w)\, X(w) \longrightarrow (3)$$

Scanned with CamScanner

where $H(w)$ is fourier transform of $h(n)$ and $X(w)$ is fourier transform of $x(n)$.

W.K.T $y(n)$ has $N+L-1$ samples.

If DFT of $y(n)$ has $M+L-1$ samples, then only it will represents $y(n)$ uniquely. Thus, $Y(K)$, the DFT of size $N \geq M+L-1$ represents $y(n)$ uniquely in time-domain.

W.K.T $Y(K)$ can be obtained from $Y(w)$ as,

$$Y(K) = Y(w)\Big|_{w=\frac{2\pi}{N}K} ; \quad 0 \leq K \leq N-1$$

$$\therefore Y(K) = X(w)H(w)\Big|_{w=\frac{2\pi}{N}K} ; \quad 0 \leq K \leq N-1 \longrightarrow (4)$$

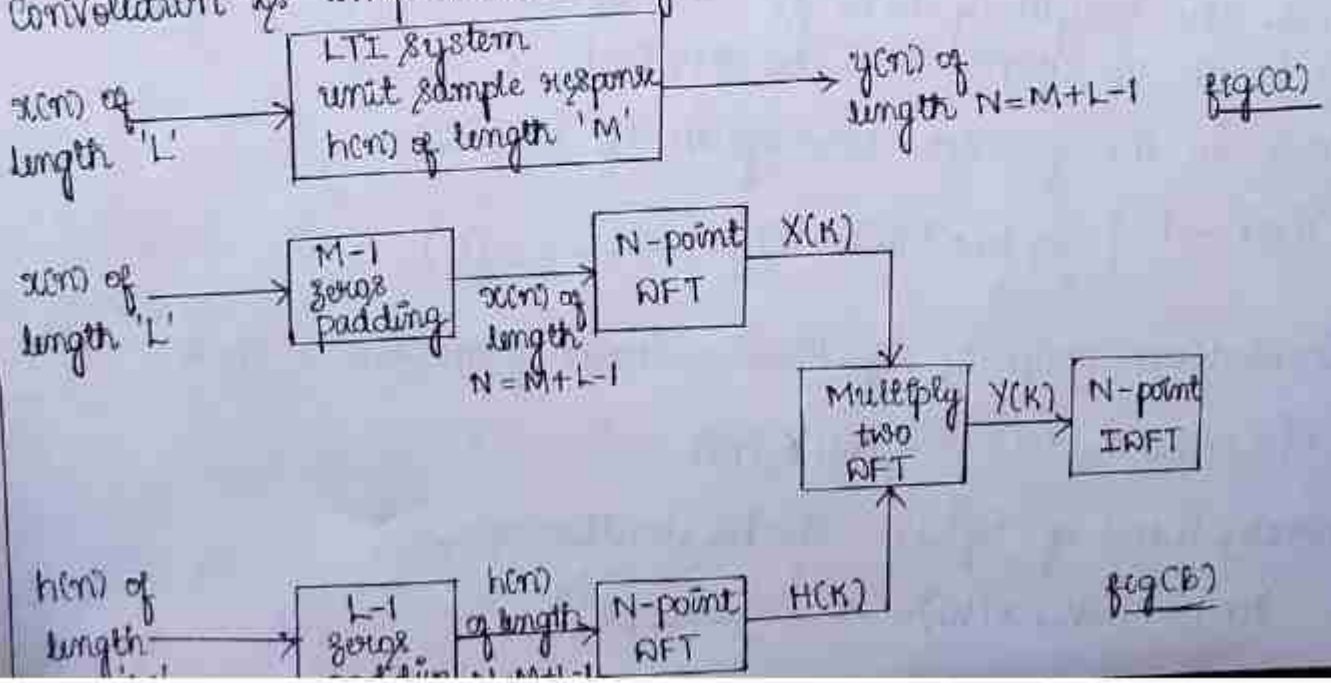
This shows that multiplying the N-point DFT's of $x(n)$ and $h(n)$, we get DFT $Y(K)$. This DFT represents $y(n)$ uniquely if $N \geq M+L-1$.

Hence, $y(n)$ can be obtained by taking IDFT of $Y(K)$

i.e., $y(n) = IDFT\{Y(K)\}$

$$= IDFT\{X(K).H(K)\} ; \quad 0 \leq K \leq N-1 \longrightarrow (5)$$

$\therefore y(n)$ can be obtained with the help of DFT. Thus linear convolution is implemented by DFT.



Scanned with CamScanner

Note :-

1) Linear Convolution operation can be implemented by DFT and IDFT. These calculations are carried out in frequency-domain.
2) Linear Convolution operation can be implemented by Circular Convolution. The sequences are appended by zeros to the length 'N'. This avoids aliasing Effect in output sequence y(n). The calculations of circular convolution are carried out in time-domain.

b) DFT for linear filtering of long-duration sequences :-

In practical applications, we find that the input x(n) is a very long data sequences and we require linear filtering of this sequence. This is required in some real-time signal processing applications such as ECG, speech etc.
Due to the limitations of memory size of digital computers, a long input sequence must be segmented to fixed-size blocks prior to processing. The DFT computation is done only on block of data.

As the filtering is linear, successive blocks can be processed one after another through DFT and then processed output blocks are placed systematically to form total output sequence. Filtering operation of long-data sequences can be done block-by-block basis using DFT.

There are two methods for this operation :

1) overlap-add method
2) overlap-save method.

For the both methods, it is assumed that;

1) The FIR filter has duration 'M' and
2) The input sequence is segmented into blocks of 'L' points by assuming $L >> M$.

Scanned with CamScanner

Fig(a) shows straight forward linear convolution of x(n) and h(n). Fig(b) shows how linear convolution can be implemented with the help of DFT and IDFT. The DFT should be of length $N = M+L-1$. Hence, the lengths of x(n) and h(n) are first appended to 'N' by zero padding as shown in fig(b). This zero padding doesnot change the meaning of the sequence. The N-point DFT's X(K) and H(K) are then computed. The two DFT's X(K) and H(K) are multiplied to give Y(K). The N-point IDFT of Y(K) then gives y(n) as shown in fig(b).

To implement linear convolution using Circular convolution:-

Consider, $Y(K) = X(K)H(K)$; $0 \le K \le N-1$ ⟶ (1)

From the Circular property of DFT,

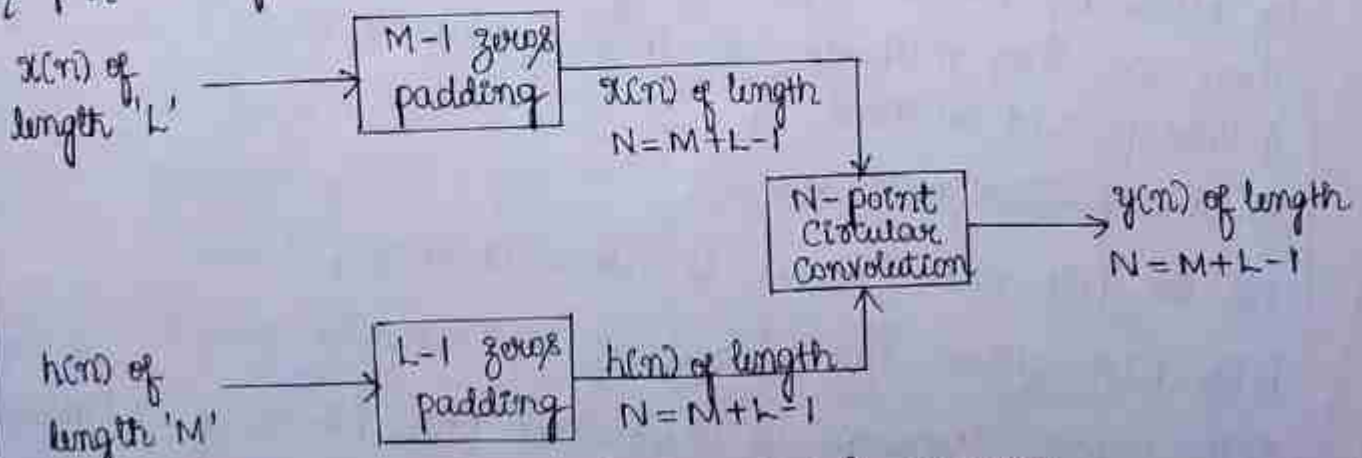
$$x_1(n) \textcircled{N} x_2(n) \xleftrightarrow[N\text{-point}]{\text{DFT}} X_1(K)\, X_2(K)$$

This means multiplication of two DFT's is equivalent to Circular Convolution of Corresponding time-domain sequences.

∴ Eq(1) becomes,

$$y(n) = x(n) \textcircled{N} h(n) \; ; \; 0 \le n \le N-1 \longrightarrow (2)$$

Thus, N-point Circular Convolution of x(n) and h(n) gives sequence y(n). Thus, linear convolution by Circular Convolution is possible if $N = L+M-1$. This operation is shown in fig below:



Linear convolution implemented by Circular convolution

Scanned with CamScanner

Note :-

1) Filtering of long-duration sequence is also called as Fast-Convolution (or) Sectioned Convolution (or) linear Convolution - one finite and one infinite sequence.

2) It involves : i) Taking DFT of block of data.
ii) Multiplying the result by DFT of unit impulse response.
iii) obtaining IDFT.
iv) process is repeated on subsequent blocks of data.

Overlap-add method for linear filtering :-

Let the unit sample response $h(n)$ be of length 'M' and input sequence $x(n)$ be segmented into blocks of 'L' samples. The following steps are followed by overlap-add method to filter the sequence :

Step 1 : Select the value of $N = 2^M$, so that radix-2 FFT can be used.

Step 2 : The length of $h(n)$ is made 'N' by padding L-1 zeros i.e., $N = M + L - 1$.

Step 3 : The sequence $x(n)$ is sectioned into sub-sequences of length 'N' as follows :

$x_1(n) = \{x(0), x(1), \ldots x(L-1), 0, 0, \ldots 0\}$

$x_2(n) = \{x(L), x(L+1), \ldots x(2L-1), 0, 0, \ldots 0\}$

$x_3(n) = \{x(2L), x(2L+1), \ldots x(3L-1), 0, 0, \ldots 0\}$

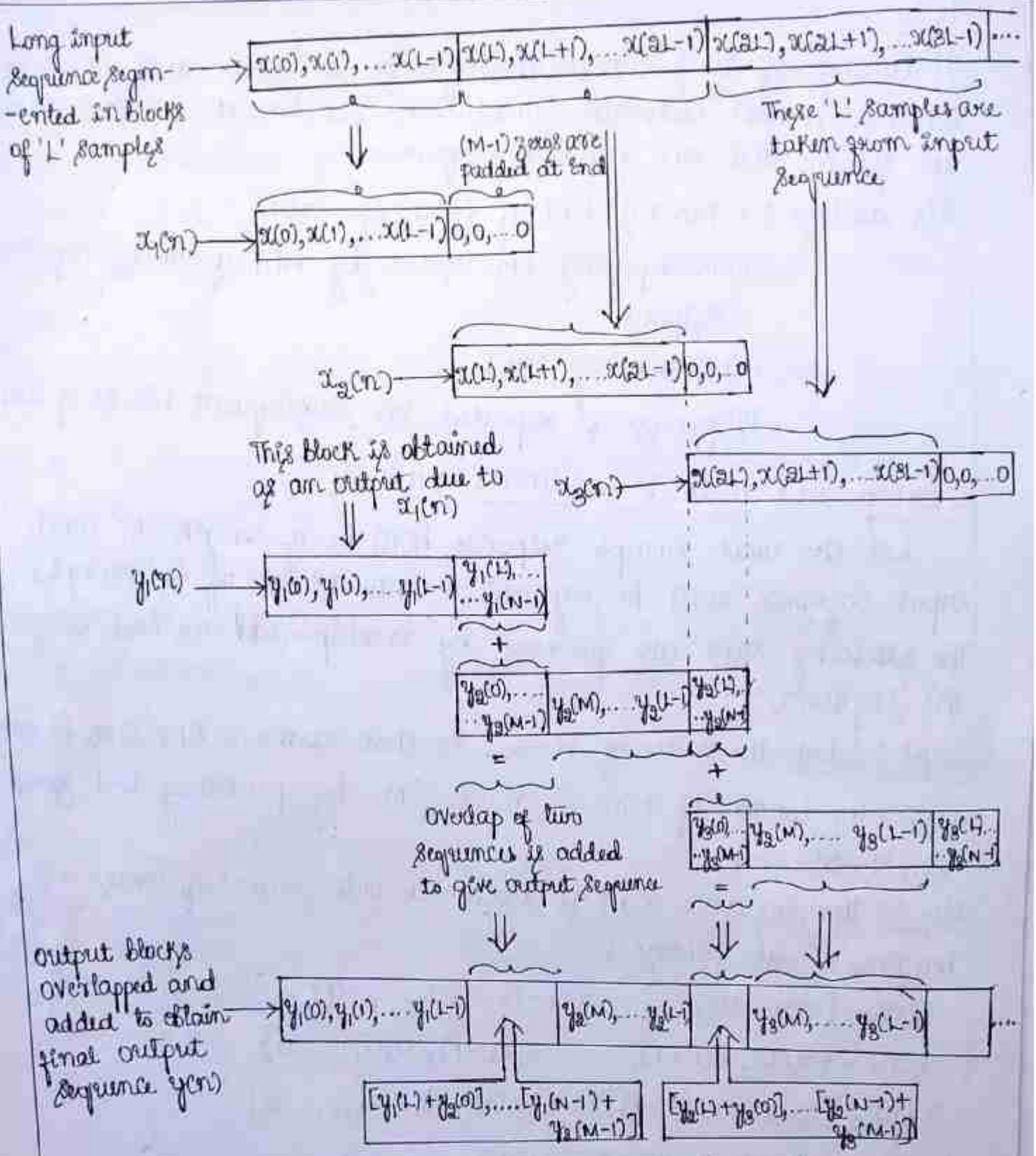
Step 4 : Calculate $Y_1(K) = X_1(K)\,H(K)$ (or) $y_1(n) = x_1(n) \textcircled{N} h(n)$

Calculate $y_1(n) = \text{IDFT}\{Y_1(K)\}$

Step 5 : Repeat Step 4 to obtain $y_2(n), y_3(n), \ldots$ so on.

Step 6 : Add last (M-1) samples of each output sequence to first (M-1) samples of succeeding output sequence. Such sequences are fitted one after another to get final sequence.

Scanned with CamScanner



overlap-add method for filtering of long data sequences using DFT

Scanned with CamScanner

Overlap-Save method for linear filtering :—

Let the unit sample response $h(n)$ be of length 'M' and the input sequence $x(n)$ be segmented into blocks of 'L' samples. Overlap-Save method follows the following steps to filter the sequence.

Step 1 :- Select the value of $N = 2^M$, so that radix-2 FFT can be used.

Step 2 :- The length of $h(n)$ is made 'N' by padding $(L-1)$ zeros i.e., $N = M+L-1$.

Step 3 :- The sequence $x(n)$ is sectioned into sub-sequences of length 'N' as follows :

$x_1(n) = \{0, 0, \ldots 0, x(0), x(1), \ldots x(L-1)\}$

$x_2(n) = \{x(L-M+1), \ldots x(L-1), x(L), x(L+1), \ldots x(2L-1)\}$

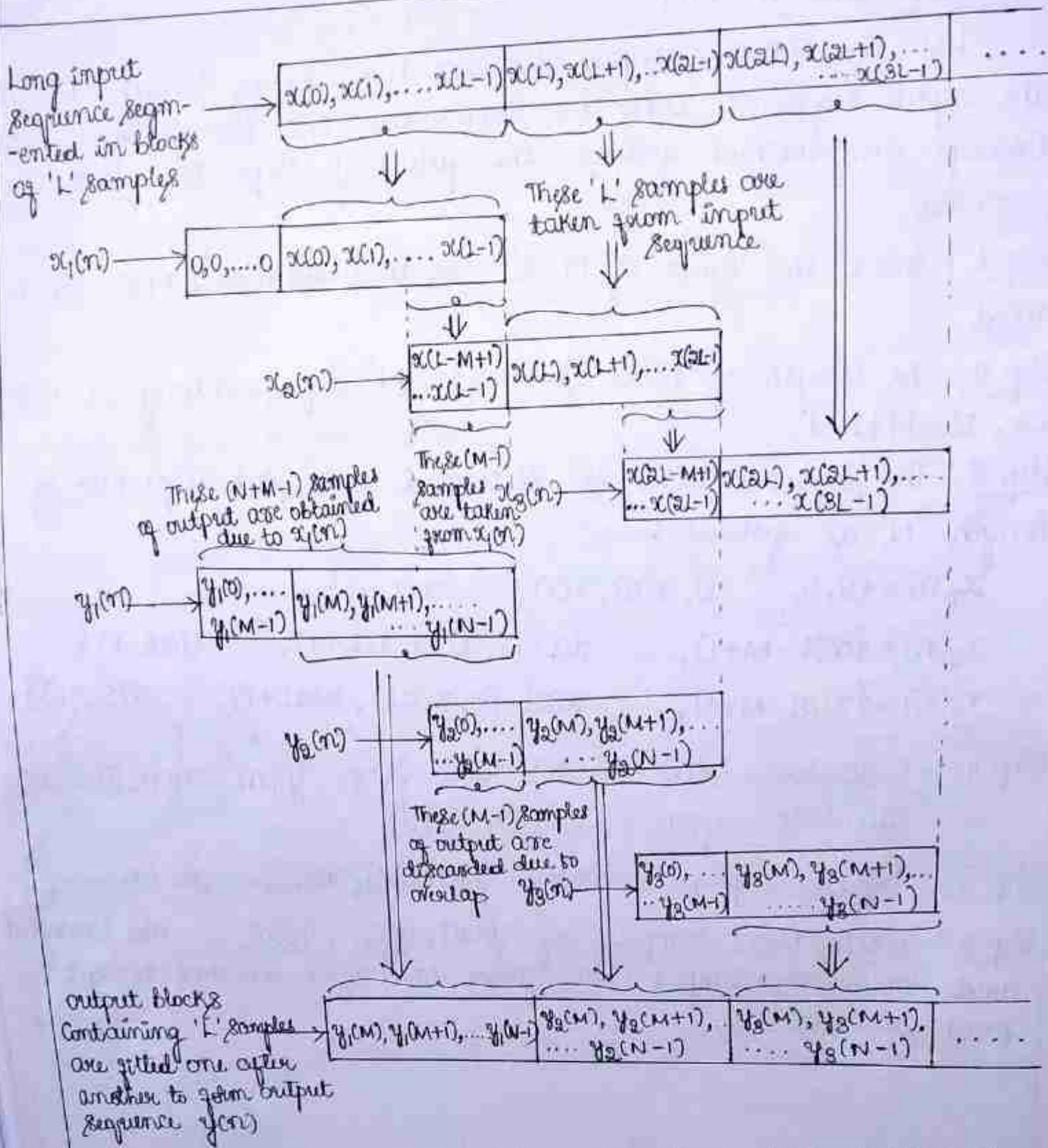
$x_3(n) = \{x(2L-M+1), \ldots x(2L-1), x(2L), x(2L+1), \ldots x(3L-1)\}$

Step 4 :- Calculate $Y_1(K) = X_1(K) \cdot H(K)$ } (or) $y_1(n) = x_1(n) \textcircled{N} h(n)$
Calculate $y_1(n) = \text{IDFT}\{Y_1(K)\}$ }

Step 5 :- Repeat Step-4 to obtain $y_2(n), y_3(n), \ldots$ so on.

Step 6 :- First $(M-1)$ samples of $y_1(n), y_2(n), y_3(n), \ldots$ are discarded and remaining samples are gitted one after another to get final sequence.

Scanned with CamScanner



Overlap-Save method for filtering of long data sequences using DFT

Scanned with CamScanner

Comparison between Linear and Circular Convolution :—

Linear Convolution	Circular Convolution
1) Convolution operation is perfor-med linearly.	1) Convolution operation is performed circularly.
2) The length of the two sequences can be unequal.	2) The length of the two sequences must be equal.
3) The length of the resulting sequence = length of first sequence + length of second sequence − 1.	3) The length of the resulting sequence is same as that of the sequences.
4) Linear Convolution result can be obtained from Circular convolution by appending appropriate zeros to the sequences.	4) Circular Convolution result cannot be obtained from linear convolution.
5) $y(n) = \sum_{K=-\infty}^{\infty} x(K)\, h(n-K)$	5) $y(n) = \sum_{K=-\infty}^{\infty} x(K)\, h((n-K))_N$; $n = 0, 1, \ldots (N-1)$
6) $x(n) * h(n) \xleftrightarrow{FT} X(\omega)\, H(\omega)$	6) $x(n) \textcircled{N} h(n) \xleftrightarrow[N\text{-point}]{DFT} X(K)\, H(K)$
7) Range of time index : $-\infty \le n \le \infty$	7) Range of time index : $0 \le n \le (N-1)$

Scanned with CamScanner

Problems on overlap-add method :-

1) A sequence $x(n) = \{1, 2, -1, 3, 2, -3, -2, -1, 2\}$ is filtered through a filter having impulse response $h(n) = \{3, 2\}$. Find the filter output using overlap-add method.

Sol^n :- Given $h(n) = \{3, 2\}$

$x(n) = \{1, 2, -1, 3, 2, -3, -2, -1, 2\}$

$\therefore M = 2$, we choose the transform length for DFT and IDFT computation as $N = 2^M = 2^2 = 4$

Since $N = M + L - 1$

$4 = 2 + L - 1$

$\langle L = 3 \rangle$

Step 1 :- To find $y_1(n)$

$x_1(n) = \{1, 2, -1, \boxed{0}\}$ ← (M-1) zeros

$h(n) = \{3, 2, \boxed{0, 0}\}$ ← (L-1) zeros

W.K.T $y_1(n) = x_1(n) \circledast_4 h(n)$

By DFT, $Y_1(K) = X_1(K) H(K)$

$= (1 + 2W_4^K - W_4^{2K})(3 + 2W_4^K)$

$= 3 + 6W_4^K - 3W_4^{2K}$

$2W_4^K + 4W_4^{2K} - 2W_4^{3K}$

$Y_1(K) = 3 + 8W_4^K + W_4^{2K} - 2W_4^{3K}$

W.K.T $\delta(n - n_0) = W_N^{n_0 K}$

By IDFT, $y_1(n) = 3\delta(n) + 8\delta(n-1) + \delta(n-2) - 2\delta(n-3)$

$\langle \therefore y_1(n) = \{3, 8, 1, -2\} \rangle$

Step 2 : To find $y_2(n)$

$x_2(n) = \{3, 2, -3, \boxed{0}\}$ ← (M-1) zeros

$h(n) = \{3, 2, \boxed{0, 0}\}$ ← (L-1) zeros

Scanned with CamScanner

$y_2(n) = x_2(n) \circledast_4 h(n)$

By DFT, $Y_2(K) = X_2(K)\, H(K)$

$= (3 + 2W_4^K - 3W_4^{2K})(3 + 2W_4^K)$

$= 9 + 6W_4^K - 9W_4^{2K}$

$6W_4^K + 4W_4^{2K} - 6W_4^{3K}$

$Y_2(K) = 9 + 12W_4^K - 5W_4^{2K} - 6W_4^{3K}$

W.K.T $\delta(n-n_0) = W_N^{Kn_0}$

By IDFT, $y_2(n) = 9\delta(n) + 12\delta(n-1) - 5\delta(n-2) - 6\delta(n-3)$

$\langle \therefore y_2(n) = \{9, 12, -5, -6\} \rangle$

↑

Step 3 :- To find $y_3(n)$

$x_3(n) = \{-2, -1, 2, \boxed{0}\}$ ↑ (M-1) zeros

$h(n) = \{3, 2, \boxed{0, 0}\}$ ↑ (L-1) zeros

$y_3(n) = x_3(n) \circledast_4 h(n)$

By DFT, $Y_3(K) = X_3(K)\, H(K)$

$= (-2 - W_4^K + 2W_4^{2K})(3 + 2W_4^K)$

$= -6 - 3W_4^K + 6W_4^{2K}$

$-4W_4^K - 2W_4^{2K} + 4W_4^{3K}$

$Y_3(K) = -6 - 7W_4^K + 4W_4^{2K} + 4W_4^{3K}$

By IDFT, $y_3(n) = -6\delta(n) - 7\delta(n-1) + 4\delta(n-2) + 4\delta(n-3)$

$\langle \therefore y_3(n) = \{-6, -7, 4, 4\} \rangle$

↑

Step 4 :- $y(n) = y_1(n) + y_2(n-L) + y_3(n-2L) + \ldots$

Here $L = 3$,

$y(n) = y_1(n) + y_2(n-3) + y_3(n-6)$

Scanned with CamScanner

$n \longrightarrow$	0	1	2	3	4	5	6	7	8	9
$y_1(n) \longrightarrow$	3	8	1	-2						
$y_2(n-3) \longrightarrow$				9	12	-5	-6			
$y_3(n-6) \longrightarrow$							-6	-7	4	4
$y(n) \longrightarrow$	3	8	1	7	12	-5	-12	-7	4	(4)

↑ Discard (M-1) points

$\therefore y(n) = \{3, 8, 1, 7, 12, -5, -12, -7, 4\}$

Note:- The last one points of $y_3(n-6)$ must be overlapped with $y_4(n-9)$ and added. Since $y_4(n)$ is not calculated, the samples beyond $n=8$ cannot be computed.

Verification:-

	$x(n) \longrightarrow$	1	2	-1	3	2	-3	-2	-1	2
$h(n)$ ↓	3	3	6	-3	9	6	-9	-6	-3	6
	2	2	4	-2	6	4	-6	-4	-2	4

$y(n) = \{3, 8, 1, 7, 12, -5, -12, -7, 4, 4\}$

2) Using DFT properties, which relates linear convolution to Circular convolution, obtain the output of a linear filter given the impulse response $h(n) = \{1, 1, 1\}$ and an input to be a long sequence $x(n) = \{1, 2, 0, -3, 4, 2, -1, 1, -2, 3, 2, 1, -3\}$

Sol^n:- Since $M=3$, we choose transform length $N = 2^M = 2^3 = 8$.

Since $N = M+L-1$

$8 = 3+L-1$

$L = 6$

$x_1(n) = \{1, 2, 0, -3, 4, 2, 0, 0\}$

$x_2(n) = \{-1, 1, -2, 3, 2, 1, 0, 0\}$

$x_3(n) = \{-3, 0, 0, 0, 0, 0, 0, 0\}$

$h(n) = \{1, 1, 1, 0, 0, 0, 0, 0\}$

Step 1:- To find $y_1(n) = x_1(n) \circledast_8 h(n)$

By DFT, $Y_1(K) = X_1(K) H(K)$

$= (1 + 2W_8^K - 3W_8^{3K} + 4W_8^{4K} + 2W_8^{5K})(1 + W_8^K + W_8^{2K})$

$= 1 + 2W_8^K - 3W_8^{3K} + 4W_8^{4K} + 2W_8^{5K}$
$W_8^K + 2W_8^{2K} - 3W_8^{4K} + 4W_8^{5K} + 2W_8^{6K}$
$W_8^{2K} + 2W_8^{3K} - 3W_8^{5K} + 4W_8^{6K} + 2W_8^{7K}$

Scanned with CamScanner

$Y_1(K) = 1 + 3W_8^K + 3W_8^{2K} - W_8^{3K} + W_8^{4K} + 3W_8^{5K} + 6W_8^{6K} + 2W_8^{7K}$

By IDFT,

$y_1(n) = \delta(n) + 3\delta(n-1) + 3\delta(n-2) - \delta(n-3) + \delta(n-4) + 3\delta(n-5) + 6\delta(n-6) + 2\delta(n-7)$

$\therefore y_1(n) = \{1, 3, 3, -1, 1, 3, 6, 2\}$

Step 2 :- To find $y_2(n) = x_2(n) \circledast_8 h(n)$

By DFT, $Y_2(K) = X_2(K) H(K)$

$= (-1 + W_8^K - 2W_8^{2K} + 3W_8^{3K} + 2W_8^{4K} + W_8^{5K})(1 + W_8^K + W_8^{2K})$

$= -1 + W_8^K - 2W_8^{2K} + 3W_8^{3K} + 2W_8^{4K} + W_8^{5K}$
$- W_8^K + W_8^{2K} - 2W_8^{3K} + 3W_8^{4K} + 2W_8^{5K} + W_8^{6K}$
$- W_8^{2K} + W_8^{3K} - 2W_8^{4K} + 3W_8^{5K} + 2W_8^{6K} + W_8^{7K}$

$Y_2(K) = -1 - 2W_8^{2K} + 2W_8^{3K} + 3W_8^{4K} + 6W_8^{5K} + 3W_8^{6K} + W_8^{7K}$

By IDFT, $y_2(n) = -\delta(n) - 2\delta(n-2) + 2\delta(n-3) + 3\delta(n-4) + 6\delta(n-5) + 3\delta(n-6) + \delta(n-7)$

$\therefore y_2(n) = \{-1, 0, -2, 2, 3, 6, 3, 1\}$

Step 3 :- To find $y_3(n) = x_3(n) \circledast_8 h(n)$

By DFT, $Y_3(K) = X_3(K) H(K)$

$= (-3)(1 + W_8^K + W_8^{2K})$

$Y_3(K) = -3 - 3W_8^K - 3W_8^{2K}$

By IDFT, $y_3(n) = -3\delta(n) - 3\delta(n-1) - 3\delta(n-2)$

$\therefore y_3(n) = \{-3, -3, -3, 0, 0, 0, 0, 0\}$

Step 4 :- $y(n) = y_1(n) + y_2(n-L) + y_3(n-2L)$

Since $L = 6$, $y(n) = y_1(n) + y_2(n-6) + y_3(n-12)$

Scanned with CamScanner

$n \rightarrow$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$y_1(n) \rightarrow$	1	3	3	-1	1	3	6	2												
$y_2(n-6) \rightarrow$							-1	0	-2	2	3	6	3	1						
$y_3(n-12) \rightarrow$													-3	-3	-3	0	0	0	0	0

$y(n) \rightarrow$	1	3	3	-1	1	3	5	2	-2	2	3	6	0	-2	-3	0	0	0	0	0

Wipcard (M-1) points

$\therefore y(n) = \{1, 3, 3, -1, 1, 3, 5, 2, -2, 2, 3, 6, 0, -2, 3, 0, 0, 0\}$

Note :- The last two points of $y_3(n-12)$ must be overlapped with $y_4(n-18)$ and added. Since $y_4(n)$ is not calculated, the samples beyond $n=17$ cannot be computed.

Verification :-

$x(n) \rightarrow$

	1	2	0	-3	4	2	-1	1	-2	3	2	1	-3
$h(n)$ 1	1	2	0	-3	4	2	-1	1	-2	3	2	1	-3
↓ 1		1	2	0	-3	4	2	-1	1	-2	3	2	1	-3
1			1	2	0	-3	4	2	-1	1	-2	3	2	1	-3

$y(n) = \{1, 3, 3, -1, 1, 8, 5, 2, -2, 2, 3, 6, 0, -2, -3\}$

3) Consider, a FIR filter with impulse response $h(n) = \{3, 2, 1, 1\}$. If the input is $x(n) = \{1, 2, 3, 3, 2, 1, -1, -2, -3, 5, 6, -1, 2, 0, 2, 1\}$. Find the output $y(n)$ using overlap-add method and the 7-point circular convolution (or) assuming the length of block is 7.

Soln :- Since $M = 4$

Given $N = 7$

Since $N = M + L - 1 \Rightarrow L = N - M + 1$

$= 7 - 4 + 1$

$\langle L = 4 \rangle$

$x_1(n) = \{1, 2, 3, 3, 0, 0, 0\}$

$x_2(n) = \{2, 1, -1, -2, 0, 0, 0\}$

$x_3(n) = \{-3, 5, 6, -1, 0, 0, 0\}$

$x_4(n) = \{2, 0, 2, 1, 0, 0, 0\}$

Scanned with CamScanner

and $h(n) = \{3, 2, 1, 1, 0, 0, 0\}$

Step 1: To find $y_1(n) = x_1(n) \circledast_7 h(n)$

By DFT, $Y_1(K) = X_1(K) H(K)$

$$= (1 + 2W_7^K + 3W_7^{2K} + 3W_7^{3K})(3 + 2W_7^K + W_7^{2K} + W_7^{3K})$$

$$= 3 + 6W_7^K + 9W_7^{2K} + 9W_7^{3K}$$
$$2W_7^K + 4W_7^{2K} + 6W_7^{3K} + 6W_7^{4K}$$
$$+ W_7^{2K} + 2W_7^{3K} + 3W_7^{4K} + 3W_7^{5K}$$
$$+ W_7^{3K} + 2W_7^{4K} + 3W_7^{5K} + 3W_7^{6K}$$

$$Y_1(K) = 3 + 8W_7^K + 14W_7^{2K} + 18W_7^{3K} + 11W_7^{4K} + 6W_7^{5K} + 3W_7^{6K}$$

By IDFT,

$$y_1(n) = 3\delta(n) + 8\delta(n-1) + 14\delta(n-2) + 18\delta(n-3) + 11\delta(n-4) + 6\delta(n-5) + 3\delta(n-6)$$

$\therefore y_1(n) = \{3, 8, 14, 18, 11, 6, 3\}$ (↑ at 3)

Step 2 :- To find $y_2(n) = x_2(n) \circledast_7 h(n)$

By DFT, $Y_2(K) = X_2(K) H(K)$

$$= (2 + W_7^K - W_7^{2K} - 2W_7^{3K})(3 + 2W_7^K + W_7^{2K} + W_7^{3K})$$

$$= 6 + 3W_7^K - 3W_7^{2K} - 6W_7^{3K}$$
$$4W_7^K + 2W_7^{2K} - 2W_7^{3K} - 4W_7^{4K}$$
$$2W_7^{2K} + W_7^{3K} - W_7^{4K} - 2W_7^{5K}$$
$$2W_7^{3K} + W_7^{4K} - W_7^{5K} - 2W_7^{6K}$$

$$Y_2(K) = 6 + 7W_7^K + W_7^{2K} - 5W_7^{3K} - 4W_7^{4K} - 3W_7^{5K} - 2W_7^{6K}$$

By IDFT,

$$y_2(n) = 6\delta(n) + 7\delta(n-1) + \delta(n-2) - 5\delta(n-3) - 4\delta(n-4) - 3\delta(n-5) - 2\delta(n-6)$$

$\therefore y_2(n) = \{6, 7, 1, -5, -4, -3, -2\}$ (↑ at 6)

Scanned with CamScanner

Step 3 :- To find $y_3(n) = x_3(n) \circledast_4 h(n)$

By DFT, $Y_3(K) = X_3(K)\, H(K)$

$$= (-3 + 5W_4^K + 6W_4^{2K} - W_4^{3K})(3 + 2W_4^K + W_4^{2K} + W_4^{3K})$$

$$= -9 + 15W_4^K + 18W_4^{2K} - 3W_4^{3K}$$
$$- 6W_4^K + 10W_4^{2K} + 12W_4^{3K} - 2W_4^{4K}$$
$$- 3W_4^{2K} + 5W_4^{3K} + 6W_4^{4K} - W_4^{5K}$$
$$- 3W_4^{3K} + 5W_4^{4K} + 6W_4^{5K} - W_4^{6K}$$

$$Y_3(K) = -9 + 9W_4^K + 25W_4^{2K} + 11W_4^{3K} + 9W_4^{4K} + 5W_4^{5K} - W_4^{6K}$$

By IDFT,

$$y_3(n) = -9\delta(n) + 9\delta(n-1) + 25\delta(n-2) + 11\delta(n-3) + 9\delta(n-4) + 5\delta(n-5) - \delta(n-6)$$

$\therefore y_3(n) = \{-9, 9, 25, 11, 9, 5, -1\}$

Step 4 :- To find $y_4(n) = x_4(n) \circledast_4 h(n)$

By DFT, $Y_4(K) = X_4(K)\, H(K)$

$$= (2 + 2W_4^{2K} + W_4^{3K})(3 + 2W_4^K + W_4^{2K} + W_4^{3K})$$

$$= 6 + 4W_4^K + 2W_4^{2K} + 2W_4^{3K}$$
$$+ 6W_4^{2K} + 4W_4^{3K} + 2W_4^{4K} + 2W_4^{5K}$$
$$3W_4^{3K} + 2W_4^{4K} + W_4^{5K} + W_4^{6K}$$

$$Y_4(K) = 6 + 4W_4^K + 8W_4^{2K} + 9W_4^{3K} + 4W_4^{4K} + 3W_4^{5K} + W_4^{6K}$$

By IDFT,

$$y_4(n) = 6\delta(n) + 4\delta(n-1) + 8\delta(n-2) + 9\delta(n-3) + 4\delta(n-4) + 3\delta(n-5) + \delta(n-6)$$

$\therefore y_4(n) = \{6, 4, 8, 9, 4, 3, 1\}$

Scanned with CamScanner

Step 5:- $y(n) = y_1(n) + y_2(n-L) + y_3(n-2L) + y_4(n-3L)$

Here $L=4$, $y(n) = y_1(n) + y_2(n-4) + y_3(n-8) + y_4(n-12)$

$y_1(n) \rightarrow$ 3 8 14 18 11 6 3

$y_2(n-4) \rightarrow$ 6 7 1 -5 -4 -3 -2

$y_3(n-8) \rightarrow$ -9 9 25 11 9 5 -1

$y_4(n-12) \rightarrow$ 6 4 8 9 4 3 1

$y(n) \rightarrow$ 3 8 14 18 17 13 4 -5 -13 6 23 11 15 9 7 9 | 4 3 1

Discard (M-1) points

$\therefore y(n) = \{3, 8, 14, 18, 17, 13, 4, -5, -13, 6, 23, 11, 15, 9, 7, 9\}$

Verification:-

		1	2	3	3	2	1	-1	-2	-3	5	6	-1	2	0	2	1
	$x(n) \rightarrow$																
$h(n)$																	
↓	3	3	6	9	9	6	3	-2	-6	-9	15	18	-3	6	0	6	3
	2	2	4	6	6	4	2	-2	-4	-6	10	12	-2	4	0	4	2
	1	1	2	3	3	2	1	-1	-2	-3	5	6	-1	2	0	2	1
	1	1	2	3	3	2	1	-1	-2	-3	5	6	-1	2	0	2	1

$y(n) = \{3, 8, 14, 18, 17, 13, 4, -5, -13, 6, 23, 11, 15, 9, 7, 9, 4, 3, 1\}$

4) A long sequence $x(n)$ is filtered through a filter whose impulse response $h(n)$ to yield the output $y(n)$. If $x(n) = \{1, 4, 3, 0, 7, 4, -7, -7, -1, 3, 4, 3\}$ and $h(n) = \{1, 2\}$. Compute $y(n)$ using overlap-add technique. Use only a 5-point Circular Convolution in your approach.

Soln:- Here $M=2$ and

Given $N=5$

Since $N = M+L-1 \Rightarrow L = N-M+1$

$= 5-2+1$

$L = 4$

$x_1(n) = \{1, 4, 3, 0, 0\}$

$x_2(n) = \{7, 4, -7, -7, 0\}$ and $h(n) = \{1, 2, 0, 0\}$

$x_3(n) = \{-1, 3, 4, 3, 0\}$

5)

Scanned with CamScanner

Step 1 :- To find $y_1(n) = x_1(n) \circledast_5 h(n)$

By DFT, $Y_1(K) = X_1(K)\,H(K)$

$= (1 + 4W_5^K + 3W_5^{2K})(1 + 2W_5^K)$

$= 1 + 4W_5^K + 3W_5^{2K} + 2W_5^K + 8W_5^{2K} + 6W_5^{3K}$

$Y_1(K) = 1 + 6W_5^K + 11W_5^{2K} + 6W_5^{3K}$

By IDFT, $y_2(n) = \delta(n) + 6\delta(n-1) + 11\delta(n-2) + 6\delta(n-3)$

$\therefore y_2(n) = \{\underset{\uparrow}{1}, 6, 11, 6, 0\}$

Step 2 :- To find $y_2(n) = x_2(n) \circledast_5 h(n)$

By DFT, $Y_2(K) = X_2(K)\,H(K)$

$= (7 + 4W_5^K - 7W_5^{2K} - 7W_5^{3K})(1 + 2W_5^K)$

$= 7 + 14W_5^K - 7W_5^{2K} - 14W_5^{3K} - 7W_5^{3K} + 8W_5^{2K} - 14W_5^{4K}$

$Y_2(K) = 7 + 18W_5^K + W_8^{2K} - 21W_5^{3K} - 14W_5^{4K}$

By IDFT, $y_2(n) = 7\delta(n) + 18\delta(n-1) + \delta(n-2) - 21\delta(n-3) - 14\delta(n-4)$

$\therefore y_2(n) = \{\underset{\uparrow}{7}, 18, 1, -21, -14\}$

Step 3 :- To find $y_3(n) = x_3(n) \circledast_5 h(n)$

By DFT, $Y_3(K) = X_3(K)\,H(K)$

$= (-1 + 3W_5^K + 4W_5^{2K} + 3W_5^{3K})(1 + 2W_5^K)$

$= -1 + 3W_5^K + 4W_5^{2K} + 3W_5^{3K} - 2W_5^K + 6W_5^{2K} + 8W_5^{3K} + 6W_5^{4K}$

$Y_3(K) = -1 + W_5^K + 10W_5^{2K} + 10W_5^{3K} + 6W_5^{4K}$

By IDFT, $y_3(n) = -\delta(n) + \delta(n-1) + 10\delta(n-2) + 11\delta(n-3) + 6\delta(n-4)$

$\therefore y_3(n) = \{\underset{\uparrow}{-1}, 1, 10, 11, 6\}$

Scanned with CamScanner

Step 4 :- $y(n) = y_1(n) + y_2(n-L) + y_3(n-2L) + \cdots$

Here $L = 4$, $y(n) = y_1(n) + y_2(n-4) + y_3(n-8)$

$y_1(n) \longrightarrow$ 1 6 11 6 0

$y_2(n-4) \rightarrow$ 7 18 1 -21 -14

$y_3(n-8) \rightarrow$ -1 1 10 11 6

$y(n) \longrightarrow$ 1 6 11 6 7 18 1 -21 -15 1 10 11 [6]

Discard (M-1) points

$\langle y(n) = \{1, 6, 11, 6, 7, 18, 1, -21, -15, 1, 10, 11\} \rangle$

Verification :-

$x(n) \longrightarrow$ 1 4 3 0 7 4 -7 -7 -1 3 4 3

$h(n)$ ↓	1	1	4	3	0	7	4	-7	-7	-1	3	4	3
	2	2	8	6	0	14	8	-14	-14	-2	6	8	6

$\langle y(n) = \{1, 6, 11, 6, 7, 18, 1, -21, -15, 1, 10, 11, 6\} \rangle$

5) Determine the response of an LTI system with $h(n) = \{1, -1, 2\}$ for an input $x(n) = \{1, 0, 1, -2, 1, 2, 3, -1, 0, 2\}$. Employ overlap-add method with block length of 4.

Soln :- Given $N = 4$,

Here $M = 3$

Since $N = M + L - 1 \Rightarrow L = N - M + 1$

$= 4 - 3 + 1$

$\langle L = 2 \rangle$

$x_1(n) = \{1, 0, 0, 0\}$

$x_2(n) = \{1, -2, 0, 0\}$

$x_3(n) = \{1, 2, 0, 0\}$

$x_4(n) = \{3, -1, 0, 0\}$

$x_5(n) = \{0, 2, 0, 0\}$

$h(n) = \{1, -1, 2, 0\}$

Step 1 :- To find $y_1(n) = x_1(n) \circledast_4 h(n)$

By DFT, $Y_1(K) = X_1(K) H(K)$

$= (1)(1 - W_4^K + 2W_4^{2K})$

$Y_1(K) = 1 - W_4^K + 2W_4^{2K}$

By IDFT, $y_1(n) = \delta(n) - \delta(n-1) + 2\delta(n-2)$

$\langle \therefore y_1(n) = \{1, -1, 2, 0\} \rangle$

Scanned with CamScanner

Step 2 :- To find $y_2(n) = x_2(n) \circledast_4 h(n)$

By DFT, $Y_2(K) = X_2(K)\, H(K)$

$= (1 - 2W_4^K)(1 - W_4^K + 2W_4^{2K})$

$= 1 - W_4^K + 2W_4^{2K} - 2W_4^K + 2W_4^{2K} - 4W_4^{3K}$

$Y_2(K) = 1 - 3W_4^K + 4W_4^{2K} - 4W_4^{3K}$

By IDFT, $y_2(n) = \delta(n) - 3\delta(n-1) + 4\delta(n-2) - 4\delta(n-3)$

$\therefore y_2(n) = \{\underset{\uparrow}{1}, -3, 4, -4\}$

Step 3 :- To find $y_3(n) = x_3(n) \circledast_4 h(n)$

By DFT, $Y_3(K) = X_3(K)\, H(K)$

$= (1 + 2W_4^K)(1 - W_4^K + 2W_4^{2K})$

$= 1 - W_4^K + 2W_4^{2K} + 2W_4^K - 2W_4^{2K} + 4W_4^{3K}$

$Y_3(K) = 1 + W_4^K + 4W_4^{3K}$

By IDFT, $y_3(n) = \delta(n) + \delta(n-1) + 4\delta(n-3)$

$\therefore y_3(n) = \{\underset{\uparrow}{1}, 1, 0, 4\}$

Step 4 :- To find $y_4(n) = x_4(n) \circledast_4 h(n)$

By DFT, $Y_4(K) = X_4(K)\, H(K)$

$= (3 - W_4^K)(1 - W_4^K + 2W_4^{2K})$

$= 3 - 3W_4^K + 6W_4^{2K} - W_4^K + W_4^{2K} - 2W_4^{3K}$

$Y_4(K) = 3 - 4W_4^K + 7W_4^{2K} - 2W_4^{3K}$

By IDFT, $y_4(n) = 3\delta(n) - 4\delta(n-1) + 7\delta(n-2) - 2\delta(n-3)$

$\therefore y_4(n) = \{\underset{\uparrow}{3}, -4, 7, -2\}$

Scanned with CamScanner

Step 5 :- To find $y_5(n) = x_5(n) \circledast_4 h(n)$

By DFT, $Y_5(K) = X_5(K)\,H(K)$

$= (2W_4^K)(1 - W_4^K + 2W_4^{2K})$

$Y_5(K) = 2W_4^K - 2W_4^{3K} + 4W_4^{3K}$

By IDFT, $y_5(n) = 2\delta(n-1) - 2\delta(n-2) + 4\delta(n-3)$

$\therefore y_5(n) = \{0, +2, -2, 4\}$ (0 underlined)

Step 6 :- $y(n) = y_1(n) + y_2(n-2L) + y_3(n-2L) + \ldots$

Here $L = 2$,

$y(n) = y_1(n) + y_2(n-2) + y_3(n-4) + y_4(n-6) + y_5(n-8)$

$y_1(n) \rightarrow 1 \quad -1 \quad 2 \quad 0$

$y_2(n-2) \rightarrow 1 \quad -3 \quad 4 \quad -4$

$y_3(n-4) \quad 1 \quad 1 \quad 0 \quad 4$

$y_4(n-6) \rightarrow 3 \quad -4 \quad 7 \quad -2$

$y_5(n-8) \rightarrow 0 \quad 2 \quad -2 \quad 4$

$y(n) \rightarrow 1 \quad -1 \quad 3 \quad -3 \quad 5 \quad -3 \quad 3 \quad 0 \quad 7 \quad 0 \quad \boxed{-2 \quad 4}$

↑ Discard (M-1) points

$\therefore y(n) = \{1, -1, 3, -3, 5, -3, 3, 0, 7, 0\}$

Verification :-

	x(n) → 1	0	1	-2	1	2	3	-1	0	2
h(n) 1	1	0	1	-2	1	2	3	-1	0	2
↓ -1	-1	0	-1	2	-1	-2	-3	1	0	-2
2	2	0	2	-4	2	4	6	-2	0	4

$y(n) = \{1, -1, 3, -3, 5, -3, 3, 0, 7, 0, -2, 4\}$

6) Find the output $y(n)$ of a filter whose impulse response is $h(n) = \{1, -2, 1\}$ and input signal $x(n) = \{3, 1, -2, 1, -1, 2, 4, 3, 6\}$. Use a 8-point Circular Convolution using overlap-add method.

Scanned with CamScanner

Sol^n:-

Here $M=3$, $N=2^M=2^3=8$

Since $N=M+L-1 \Rightarrow L=N-M+1$

$=8-3+1$

$\langle L=6 \rangle$

$x_1(n)=\{3,1,-2,1,-1,2,0,0\}$

$x_2(n)=\{4,3,6,0,0,0,0,0\}$

$h(n)=\{1,-2,1,0,0,0,0,0\}$

Step 1 :- To find $y_1(n)=x_1(n) \circledast_8 h(n)$

By DFT, $Y_1(K)=X_1(K)H(K)$

$=(3+W_8^K-2W_8^{2K}+W_8^{3K}-W_8^{4K}+2W_8^{5K})(1-2W_8^K+W_8^{2K})$

$=3+W_8^K-2W_8^{2K}+W_8^{3K}-W_8^{4K}+2W_8^{5K}$

$-6W_8^K-2W_8^{2K}+4W_8^{3K}-2W_8^{4K}+2W_8^{5K}-4W_8^{6K}$

$3W_8^{2K}+W_8^{3K}-2W_8^{4K}+W_8^{5K}-W_8^{6K}+2W_8^{7K}$

$Y_1(K)=3-5W_8^K-W_8^{2K}+6W_8^{3K}-5W_8^{4K}+5W_8^{5K}-5W_8^{6K}+2W_8^{7K}$

By IDFT, $y_2(n)=3\delta(n)-5\delta(n-1)-\delta(n-2)+6\delta(n-3)-5\delta(n-4)+5\delta(n-5)-5\delta(n-6)+2\delta(n-7)$

$\langle \therefore y_2(n)=\{\underset{\uparrow}{3},-5,-1,6,-5,5,-5,2\} \rangle$

Step 2 :- To find $y_2(n)=x_2(n) \circledast_8 h(n)$

By DFT, $Y_3(K)=X_3(K)H(K)$

$=(4+3W_8^K+6W_8^{2K})(1-2W_8^K+W_8^{2K})$

$=4+3W_8^K+6W_8^{2K}$

$-8W_8^K-6W_8^{2K}-12W_8^{3K}$

$+4W_8^{2K}+3W_8^{3K}+6W_8^{4K}$

$Y_3(K)=4-5W_8^K+4W_8^{2K}-9W_8^{3K}+6W_8^{4K}$

By IDFT, $y_3(n)=4\delta(n)-5\delta(n-1)+4\delta(n-2)-9\delta(n-3)+6\delta(n-4)$

$\langle \therefore y_3(n)=\{\underset{\uparrow}{4},-5,4,-9,6,0,0,0\} \rangle$

Scanned with CamScanner

Step 3 :- $y(n) = y_1(n) + y_2(n-L) + y_3(n-2L) + \cdots$

Here $L = 6$, $y(n) = y_1(n) + y_2(n-6)$

$y_1(n) \rightarrow$ 3 -5 -1 6 -5 5 -5 2

$y_2(n-6) \rightarrow$ 4 -5 4 -9 6 0 0 0

$y(n) \rightarrow$ 3 -5 -1 6 -5 5 -1 -3 4 -9 6 0 | 0 0 |

discard (M-1) points

$y(n) = \{3, -5, -1, 6, -5, 5, -1, -3, 4, -9, 6, 0\}$

Verification :-

$x(n) \rightarrow$ 3 1 -2 1 -1 2 4 3 6

$h(n)$	3	1	-2	1	-1	2	4	3	6
1	3	1	-2	1	-1	2	4	3	6
-2	-6	-2	4	-2	2	-4	-8	-6	-12
1	3	1	-2	1	-1	2	4	3	6

$y(n) = \{3, -5, -1, 6, -5, 5, -1, -3, 4, -9, 6\}$

7) An FIR filter has the unit impulse response $h(n) = \{1, 2\}$. Determine the output sequence in response to the input sequence $x(n) = \{1, -1, 2, 1, 2, -1, 1, 3\}$ using overlap-add technique. Use 5-point Circular Convolution.

Sol^n :- Given $N = 5$, $M = 2$

Since $N = M + L - 1 \Rightarrow L = N - M + 1$

$= 5 - 2 + 1$

$L = 4$

$x_1(n) = \{1, -1, 2, 1, 0\}$

$x_2(n) = \{2, -1, 1, 3, 0\}$

$h(n) = \{1, 2, 0, 0, 0\}$

Step 1 :- To find $y_1(n) = x_1(n) \circledast_5 h(n)$

By DFT, $Y_1(K) = X_1(K) H(K)$

$= (1 - W_5^K + 2W_5^{2K} + W_5^{3K})(1 + 2W_5^K)$

$= 1 - W_5^K + 2W_5^{2K} + W_5^{3K}$

$2W_5^K - 2W_5^{2K} + 4W_5^{3K} + 2W_5^{4K}$

$Y_1(K) = 1 + W_5^K + 5W_5^{3K} + 2W_5^{4K}$

Scanned with CamScanner

By IDFT, $y_1(n) = \delta(n) + \delta(n-1) + 5\delta(n-3) + 2\delta(n-4)$

$\therefore y_1(n) = \{1, 1, 0, 5, 2\}$

Step 2 :- To find $y_2(n) = x_2(n) \circledast_5 h(n)$

By DFT, $Y_2(K) = X_2(K)\, H(K)$

$$= (2 - W_5^K + W_5^{2K} + 3W_5^{3K})(1 + 2W_5^K)$$

$$= 2 - W_5^K + W_5^{2K} + 3W_5^{3K}$$
$$4W_5^K - 2W_5^{2K} + 2W_5^{3K} + 6W_5^{4K}$$

$$Y_2(K) = 2 + 3W_5^K - W_5^{2K} + 5W_5^{3K} + 6W_5^{4K}$$

By IDFT, $y_2(n) = 2\delta(n) + 3\delta(n-1) - \delta(n-2) + 5\delta(n-3) + 6\delta(n-4)$

$\therefore y_2(n) = \{2, 3, -1, 5, 6\}$

Step 3 :- $y(n) = y_1(n) + y_2(n-L) + y_3(n-2L) + \ldots$

Here $L = 4$, $y(n) = y_1(n) + y_2(n-4)$

$y_1(n) \longrightarrow$	1	1	0	5	2			
$y_2(n-4) \longrightarrow$					2	3	-1	5	6
$y(n) \longrightarrow$	1	1	0	5	4	3	-1	5	[6]

↑ Discard (M-1) points

$\therefore y(n) = \{1, 1, 0, 5, 4, 3, -1, 5\}$

Verification :—

		$x(n) \longrightarrow$ 1	-1	2	1	2	-1	1	3
$h(n)$	1	1	-1	2	1	2	-1	1	3
↓	2	2	-2	4	2	4	-2	2	6

$y(n) = \{1, 1, 0, 5, 4, 3, -1, 5, 6\}$

Scanned with CamScanner

Problems on overlap-save method :-

For the given sequence $x(n) = \{1, 2, -1, 3, 2, -3, -2, -1, 2\}$ and $h(n) = \{3, 2\}$. Find the linear convolution between the two sequences using overlap-save method

Sol^n :- $M = 2$

$N = 2^M = 2^2 = 4$

Since $N = M + L - 1$

$L = N - M + 1 = 4 - 2 + 1$

$\langle L = 3 \rangle$ — $(N-1)$ zeros

$x_1(n) = \{0, 0, 1, 2, -1\}$

$x_2(n) = \{-1, 3, 2, -3\}$ $\qquad h(n) = \{3, 2, 0, 0\}$

$x_3(n) = \{-3, -2, -1, 2\}$

$x_4(n) = \{2, 0, 0, 0\}$

Step 1 :- To find $y_1(n) = x_1(n) \circledast_4 h(n)$

By DFT, $Y_1(K) = X_1(K)\, H(K)$

$= (W_4^K + 2W_4^{2K} - W_4^{3K})(3 + 2W_4^K)$

$= 3W_4^K + 6W_4^{2K} - 3W_4^{3K}$

$+ 2W_4^{2K} + 4W_4^{3K} - 2W_4^{4K}$

$W_4^{0.K}$

$Y_1(K) = -2 + 3W_4^K + 8W_4^{2K} + W_4^{3K}$

By IDFT, $y_1(n) = -2\delta(n) + 3\delta(n-1) + 8\delta(n-2) + \delta(n-3)$

$\therefore y_1(n) = \{-2, 3, 8, 1\} \Rightarrow \langle y_1(n) = \{3, 8, 1\} \rangle$

Step 2 :- To find $y_2(n) = x_2(n) \circledast_4 h(n)$

By DFT, $Y_2(K) = X_2(K)\, H(K)$

$= (-1 + 3W_4^K + 2W_4^{2K} - 3W_4^{3K})(3 + 2W_4^K)$

$= -3 + 9W_4^K + 6W_4^{2K} - 9W_4^{3K}$

$- 2W_4^K + 6W_4^{2K} + 4W_4^{3K} - 6W_4^{4K}$

$Y_2(K) = -9 + 7W_4^K + 12W_4^{2K} - 5W_4^{3K}$

By IDFT, $y_2(n) = -9\delta(n) + 7\delta(n-1) + 12\delta(n-2) - 5\delta(n-3)$

$\therefore y_2(n) = \{\underset{\uparrow}{\boxed{-9}}, 7, 12, -5\} \Rightarrow \langle y_2(n) = \{7, 12, -5\} \rangle$

Discard (M-1) point

Step 3 :- To find $y_3(n) = x_3(n) \circledast_4 h(n)$

By DFT, $Y_3(K) = X_3(K) H(K)$

$= (-3 - 2W_4^K - W_4^{2K} + 2W_4^{3K})(3 + 2W_4^K)$

$= -9 - 6W_4^K - 3W_4^{2K} + 6W_4^{3K}$
$\quad - 6W_4^K - 4W_4^{2K} - 2W_4^{3K} + 4W_4^{4K}$

$W_4^{4K} \rightarrow W_4^{0 \cdot K}$

$Y_3(K) = -5 - 12W_4^K - 7W_4^{2K} + 4W_4^{3K}$

By IDFT, $y_3(n) = -5\delta(n) - 12\delta(n-1) - 7\delta(n-2) + 4\delta(n-3)$

$\therefore y_3(n) = \{\underset{\uparrow}{\boxed{-5}}, -12, -7, 4\} \Rightarrow \langle y_3(n) = \{-12, -7, 4\} \rangle$

Discard (M-1) point

Step 4 :- To find $y_4(n) = x_4(n) \circledast_4 h(n)$

By DFT, $Y_4(K) = X_4(K) H(K)$

$Y_4(K) = (2)(3 + 2W_4^K)$

$Y_4(K) = 6 + 4W_4^K$

By IDFT, $y_4(n) = 6\delta(n) + 4\delta(n-1)$

$\therefore y_4(n) = \{\underset{\uparrow}{\boxed{6}}, 4, 0, 0\} \Rightarrow \langle y_4(n) = \{4, 0, 0\} \rangle$

Discard (M-1) point

Step 5 :- $y(n) = \{y_1(n), y_2(n), y_3(n), y_4(n)\}$

$\langle y(n) = \{3, 8, 1, 7, 12, -5, -12, -7, 4, 4, 0, 0\} \rangle$

Verification :-

	$x(n) \rightarrow$	2	-1	3	2	-3	-2	-1	2	
$h(n)$	3	6	-3	9	6	-9	-6	-3	6	$y(n) = \{3, 8, 1, 7, 12,$
↓	2	3	4	-2	6	4	-6	-4	-2	4	$-5, -12, -7, 4, 4\}$

2) A long sequence $x(n)$ is filtered through a filter with unit impulse response $h(n)$ to give an output $y(n)$. If $x(n)=\{1,1,1,1,1,3,1,1,4,2,1,1,3,1\}$ and $h(n)=\{1,-1\}$, compute $y(n)$ using overlap-save method. Use only 5-point circular convolution in your approach.

Sol^n:- Given $N=5$

Since $M=2$

W.K.T $N = M+L-1$

$\Rightarrow L = N-M+1$

$= 5-2+1$

$\langle L=4 \rangle$

$x_1(n)=\{0,1,1,1,1\}$

$x_2(n)=\{1,1,3,1,1\}$

$x_3(n)=\{1,4,2,1,1\}$

$x_4(n)=\{1,3,1,0,0\}$

& $h(n)=\{1,-1,0,0,0\}$

Step 1 :- To find $y_1(n) = x_1(n) \circledast_5 h(n)$

By DFT, $Y_1(K) = X_1(K)H(K)$

$$= (W_5^K + W_5^{2K} + W_5^{3K} + W_5^{4K})(1-W_5^K)$$

$$= W_5^K + W_5^{2K} + W_5^{3K} + W_5^{4K} - W_5^{2K} - W_5^{3K} - W_5^{4K} - \underbrace{W_5^{5K}}_{W_5^{0.K}}$$

$$Y_1(K) = -1 + W_5^K$$

By IDFT, $y_1(n) = -\delta(n) + \delta(n-1)$

$\therefore y_1(n) = \{\boxed{-1}, 1, 0, 0, 0\} \Rightarrow \langle y_1(n)=\{1,0,0,0\} \rangle$

↑ Discard (M-1) points

Step 2 :- To find $y_2(n) = x_2(n) \circledast_5 h(n)$

By DFT, $Y_2(K) = X_2(K)H(K)$

$$= (1+W_5^K + 3W_5^{2K} + W_5^{3K} + W_5^{4K})(1-W_5^K)$$

$$= 1 + W_5^K + 3W_5^{2K} + W_5^{3K} + W_5^{4K} - W_5^K - W_5^{2K} - 3W_5^{3K} - W_5^{4K} - \underbrace{W_5^{5K}}_{W_5^{0.K}}$$

$$Y_2(K) = 2W_5^{2K} - 2W_5^{3K}$$

By IDFT, $y_2(n) = 2\delta(n-2) - 2\delta(n-3)$

$\therefore y_2(n) = \{\underset{\uparrow}{0}, 0, 2, -2, 0\} \Rightarrow \langle y_2(n) = \{0, 2, -2, 0\} \rangle$

Discard (M-1) points

Step 3 :- To find $y_3(n) = x_3(n) \circledast_5 h(n)$

By DFT, $Y_3(K) = X_3(K)\, H(K)$

$= (1 + 4W_5^K + 2W_5^{2K} + W_5^{3K} + W_5^{4K})(1 - W_5^K)$

$= 1 + 4W_5^K + 2W_5^{2K} + W_5^{3K} + W_5^{4K}$
$- W_5^K - 4W_5^{2K} - 2W_5^{3K} - W_5^{4K} - W_5^{5K}$ ($\because W_5^{5K} = 1$)

$Y_3(K) = 3W_5^K - 2W_5^{2K} - W_5^{3K}$

By IDFT, $y_3(n) = 3\delta(n-1) - 2\delta(n-2) - \delta(n-3)$

$\therefore y_3(n) = \{\underset{\uparrow}{0}, 3, -2, -1, 0\} \Rightarrow \langle y_3(n) = \{3, -2, -1, 0\} \rangle$

Discard (M-1) points

Step 4 :- To find $y_4(n) = x_4(n) \circledast_5 h(n)$

By DFT, $Y_4(K) = X_4(K)\, H(K)$

$= (1 + 3W_5^K + W_5^{2K})(1 - W_5^K)$

$= 1 + 3W_5^K + W_5^{2K}$
$- W_5^K - 3W_5^{2K} - W_5^{3K}$

$Y_4(K) = 1 + 2W_5^K - 2W_5^{2K} - W_5^{3K}$

By IDFT, $y_4(n) = \delta(n) + 2\delta(n-1) - 2\delta(n-2) - \delta(n-3)$

$\therefore y_4(n) = \{\underset{\uparrow}{1}, 2, -2, -1, 0\} \Rightarrow \langle y_4(n) = \{2, -2, -1, 0\} \rangle$

Discard (M-1) points

Step 5 :- $y(n) = \{y_1(n), y_2(n), y_3(n), y_4(n)\}$

$y(n) = \{1, 0, 0, 0, 0, 2, -2, 0, 3, -2, -1, 0, 2, -2, -1, 0\}$

Verification :-

$x(n) \rightarrow$	1	1	1	1	1	3	1	1	4	2	1	1	3	1
$x(n)$ ↓ 1	1	1	1	1	1	3	1	1	4	2	1	1	3	1
-1	-1	-1	-1	-1	-1	-3	-1	-1	-4	-2	-1	-1	-3	-1

$y(n) = \{1, 0, 0, 0, 0, 2, -2, 0, 3, -2, -1, 0, 2, -2, -1, 0\}$

3) Using Overlap-save method, Compute $y(n)$ of a FIR filter with impulse response $h(n)=\{3,2,1\}$ to an input $x(n)=\{2,1,-1,-2,-3,5,6,-1,2,0,2,1\}$. Use only 8-point Circular Convolution.

Sol^n:- Given $N=8$

Since $M=3$

$N=M+L-1 \Rightarrow L=N-M+1$

$L=8-3+1$

$\langle L=6 \rangle$

$x_1(n)=\{0,0,2,1,-1,-2,-3,5\}$

$x_2(n)=\{-3,5,6,-1,2,0,2,1\}$

$x_3(n)=\{2,1,0,0,0,0,0,0\}$

$h(n)=\{3,2,1,0,0,0,0,0\}$

Step 1 :- To find $y_1(n)=x_1(n) \circledast_8 h(n)$

By DFT, $Y_1(K)=X_1(K)H(K)$

$$=(2W_8^{2K}+W_8^{3K}-W_8^{4K}-2W_8^{5K}-3W_8^{6K}+5W_8^{7K})(3+2W_8^{K}+W_8^{2K})$$

$$=6W_8^{2K}+3W_8^{3K}-3W_8^{4K}-6W_8^{5K}-9W_8^{6K}+15W_8^{7K}$$
$$+4W_8^{3K}+2W_8^{4K}-2W_8^{5K}-4W_8^{6K}-6W_8^{7K}+10W_8^{8K}$$
$$+2W_8^{4K}+W_8^{5K}-W_8^{6K}-2W_8^{7K}-3W_8^{8K}+5W_8^{9K}$$

$$Y_1(K)=7+5W_8^{K}+6W_8^{2K}+7W_8^{3K}+W_8^{4K}-7W_8^{5K}-14W_8^{6K}+7W_8^{7K}$$

By IDFT,

$y_1(n)=7\delta(n)+5\delta(n-1)+6\delta(n-2)+7\delta(n-3)+\delta(n-4)-7\delta(n-5)-14\delta(n-6)+7\delta(n-7)$

$\therefore y_1(n)=\{\boxed{7,5},6,7,1,-7,-14,7\} \Rightarrow \langle y_1(n)=\{6,7,1,-7,-14,7\} \rangle$

↑ Discard (M-1) points

Step 2 :- To find $y_2(n)=x_2(n) \circledast_8 h(n)$

By DFT, $Y_2(K)=X_2(K)H(K)$

$$=(-3+5W_8^{K}+6W_8^{2K}-W_8^{3K}+2W_8^{4K}+2W_8^{6K}+W_8^{7K})(3+2W_8^{K}+W_8^{2K})$$

$$=-9+15W_8^{K}+18W_8^{2K}-3W_8^{3K}+6W_8^{4K}+6W_8^{6K}+8W_8^{7K}$$
$$-6W_8^{K}+10W_8^{2K}+12W_8^{3K}-2W_8^{4K}+4W_8^{5K}+4W_8^{7K}+2W_8^{8K}$$
$$-3W_8^{2K}+5W_8^{3K}+6W_8^{4K}-W_8^{5K}+2W_8^{6K}+2W_8^{8K}+W_8^{9K}$$

$$Y_2(K)=-5+10W_8^{K}+25W_8^{2K}+14W_8^{3K}+10W_8^{4K}+3W_8^{5K}+8W_8^{6K}+7W_8^{7K}$$

By IDFT,

$y_2(n) = -5\delta(n) + 10\delta(n-1) + 25\delta(n-2) + 14\delta(n-3) + 10\delta(n-4) + 3\delta(n-5) + 8\delta(n-6) + 7\delta(n-7)$

$\therefore y_2(n) = \{\underbrace{-5, 10}_{\text{Discard (M-1) points}}, 25, 14, 10, 3, 8, 7\} \Rightarrow \langle y_2(n) = \{25, 14, 10, 3, 8, 7\} \rangle$

Step 3:- To find $y_3(n) = x_3(n) \circledast_8 h(n)$

By DFT, $Y_3(K) = X_3(K) H(K)$

$= (2 + W_8^K)(3 + 2W_8^K + W_8^{2K})$

$= 6 + 4W_8^K + 2W_8^{2K}$
$\quad 3W_8^K + 2W_8^{2K} + W_8^{3K}$

$Y_3(K) = 6 + 7W_8^K + 4W_8^{2K} + W_8^{3K}$

By IDFT, $y_3(n) = 6\delta(n) + 7\delta(n-1) + 4\delta(n-2) + \delta(n-3)$

$\therefore y_3(n) = \{\underbrace{6, 7}_{\text{Discard (M-1) points}}, 4, 1, 0, 0, 0, 0\} \Rightarrow \langle y_3(n) = \{4, 1, 0, 0, 0, 0\} \rangle$

Step 4:- $y(n) = \{y_1(n), y_2(n), y_3(n)\}$

$\langle \therefore y(n) = \{6, 7, 1, -7, -14, 7, 25, 14, 10, 3, 8, 7, 4, 1, 0, 0, 0, 0\} \rangle$

Verification :-

$x(n) \rightarrow$

$h(n)$ ↓	2	1	-1	-2	-3	5	6	-1	2	0	2	1
3	6	3	-3	-6	-9	15	18	-3	6	0	6	3
2	4	2	-2	-4	-6	10	12	-2	4	0	4	2
1	2	1	-1	-2	-3	5	6	-1	2	0	2	1

$y(n) = \{6, 7, 1, -7, -14, 7, 25, 14, 10, 3, 8, 7, 4, 1\}$

4) Find the output $y(n)$ of a filter whose impulse response is $h(n) = \{1, 2\}$ and the input signal to the filter is $x(n) = \{1, 2, -1, 2, 3, -2, -3, -1, 1, 2, -1\}$ using overlap-save method

Soln:- Since $M = 2$, $N = 2^M = 2^2 = 4$

$N = M + L - 1 \Rightarrow L = N - M + 1 = 4 - 2 + 1 = 3$

$x_1(n) = \{0, 1, 2, -1\}$

$x_2(n) = \{-1, 2, 3, -2\}$ and $h(n) = \{1, 2, 0, 0\}$

$x_3(n) = \{-2, -3, -1, 1\}$

$x_4(n) = \{1, 2, -1, 0\}$

Step 1:- To find $y_1(n) = x_1(n) \circledast_4 h(n)$

By DFT, $Y_1(K) = X_1(K) H(K)$

$$= (W_4^K + 2W_4^{2K} - W_4^{3K})(1 + 2W_4^K)$$

$$= W_4^K + 2W_4^{2K} - W_4^{3K} + 2W_4^{2K} + 4W_4^{3K} - 2W_4^{4K}$$

$W_4^{0.K}$

$$Y_1(K) = -2 + W_4^K + 4W_4^{2K} + 3W_4^{3K}$$

By IDFT, $y_1(n) = -2\delta(n) + \delta(n-1) + 4\delta(n-2) + 3\delta(n-3)$

$\therefore y_1(n) = \{\boxed{-2}, 1, 4, 3\} \Rightarrow \langle y_1(n) = \{1, 4, 3\} \rangle$

↑ Discard (M-1) points

Step 2:- To find $y_2(n) = x_2(n) \circledast_4 h(n)$

By DFT, $Y_2(K) = X_2(K) H(K)$

$$= (-1 + 2W_4^K + 3W_4^{2K} - 2W_4^{3K})(1 + 2W_4^K)$$

$$= -1 + 2W_4^K + 3W_4^{2K} - 2W_4^{3K} - 2W_4^K + 4W_4^{2K} + 6W_4^{3K} - 4W_4^{4K}$$

$W_4^{0.K}$

$$Y_2(K) = -5 + 7W_4^{2K} + 4W_4^{3K}$$

By IDFT, $y_2(n) = -5\delta(n) + 7\delta(n-2) + 4\delta(n-3)$

$\therefore y_2(n) = \{\boxed{-5}, 0, 7, 4\} \Rightarrow \langle y_2(n) = \{0, 7, 4\} \rangle$

↑ Discard (M-1) points

Step 3:- To find $y_3(n) = x_3(n) \circledast_4 h(n)$

By DFT, $Y_3(K) = X_3(K) H(K)$

$$= (-2 - 3W_4^K - W_4^{2K} + W_4^{3K})(1 + 2W_4^K)$$

$$= -2 - 3W_4^K - W_4^{2K} + W_4^{3K} - 4W_4^K - 6W_4^{2K} - 2W_4^{3K} + 2W_4^{4K}$$

$Y_3(K) = -7W_4^K - 7W_4^{2K} - W_4^{3K}$

By IDFT,

$y_3(n) = -7\delta(n-1) - 7\delta(n-2) - \delta(n-3)$

$\therefore y_3(n) = \{\boxed{0}, -7, -7, -1\} \Rightarrow \langle y_3(n) = \{-7, -7, -1\} \rangle$

↑ Discard (M-1) points

Step 4 :- To find $y_4(n) = x_4(n) \circledast_4 h(n)$

By DFT, $Y_4(K) = X_4(K) H(K)$

$= (1 + 2W_4^K - W_4^{2K})(1 + 2W_4^K)$

$= 1 + 2W_4^K - W_4^{2K}$

$2W_4^K + 4W_4^{2K} - 2W_4^{3K}$

$Y_4(K) = 1 + 4W_4^K + 3W_4^{2K} - 2W_4^{3K}$

By IDFT, $y_4(n) = \delta(n) + 4\delta(n-1) + 3\delta(n-2) - 2\delta(n-3)$

$\therefore y_4(n) = \{\boxed{1}, 4, 3, -2\} \Rightarrow \langle y_4(n) = \{4, 3, -2\} \rangle$

↑ Discard (M-1) points

Step 5 :- $y(n) = \{y_1(n), y_2(n), y_3(n), y_4(n)\}$

$\langle \therefore y(n) = \{1, 4, 3, 0, -7, 4, -7, -7, -1, 4, 3, -2\} \rangle$

Verification :-

$x(n) \rightarrow$

		1	2	-1	2	3	-2	-3	-1	1	2	-1
$h(n)$	1	1	2	-1	2	3	-2	-3	-1	1	2	-1
↓	2	2	4	-2	4	6	-4	-6	-2	2	4	-2

$\langle y(n) = \{1, 4, 3, 0, -7, 4, -7, -7, -1, 4, 3, -2\} \rangle$

5) Find the output $y(n)$ of a filter whose impulse response is $h(n) = \{1, 1, 1\}$ and the input signal $x(n) = \{3, -1, 0, 1, 3, 2, 0, 1, 2, 1\}$ using overlap-save method.

Sol^n :- Since $M = 3$, $N = 2^M = 2^3 = 8$

$N = M + L - 1 \Rightarrow L = N - M + 1 = 8 - 3 + 1 = 6$

$x_1(n) = \{0, 0, 3, -1, 0, 1, 3, 2\}$ and $h(n) = \{1, 1, 1, 0, 0, 0, 0, 0\}$

$x_2(n) = \{3, 2, 0, 1, 2, 1, 0, 0\}$

Step 1 :- To find $y_1(n) = x_1(n) \circledast_8 h(n)$

By DFT, $Y_1(K) = X_1(K)\, H(K)$

$$= (3W_8^{2K} - W_8^{3K} + W_8^{5K} + 3W_8^{6K} + 2W_8^{7K})(1 + W_8^{K} + W_8^{2K})$$

$$= 3W_8^{2K} - W_8^{3K} + W_8^{5K} + 3W_8^{6K} + 2W_8^{7K}$$
$$3W_8^{3K} - W_8^{4K} + W_8^{6K} + 3W_8^{7K} + 2W_8^{8K}$$
$$3W_8^{4K} - W_8^{5K} + W_8^{7K} + 3W_8^{8K} + 2W_8^{9K}$$

$$Y_1(K) = 5 + 2W_8^{K} + 3W_8^{2K} + 2W_8^{3K} + 2W_8^{4K} + 4W_8^{6K} + 6W_8^{7K}$$

By IDFT,

$$y_1(n) = 5\delta(n) + 2\delta(n-1) + 3\delta(n-2) + 2\delta(n-3) + 2\delta(n-4) + 4\delta(n-6) + 6\delta(n-7)$$

$\therefore y_1(n) = \{\boxed{5, 2}, 3, 2, 2, 0, 4, 6\} \Rightarrow \langle y_1(n) = \{3, 2, 2, 0, 4, 6\} \rangle$

↑ Discard (M-1) points

Step 2 :- To find $y_2(n) = x_2(n) \circledast_8 h(n)$

By DFT, $Y_2(K) = X_2(K)\, H(K)$

$$= (3 + 2W_8^{K} + W_8^{3K} + 2W_8^{4K} + W_8^{5K})(1 + W_8^{K} + W_8^{2K})$$

$$= 3 + 2W_8^{K} + W_8^{3K} + 2W_8^{4K} + W_8^{5K}$$
$$3W_8^{K} + 2W_8^{2K} + W_8^{4K} + 2W_8^{5K} + W_8^{6K}$$
$$3W_8^{2K} + 2W_8^{3K} + W_8^{5K} + 2W_8^{6K} + W_8^{7K}$$

$$Y_2(K) = 3 + 5W_8^{K} + 5W_8^{2K} + 3W_8^{3K} + 3W_8^{4K} + 4W_8^{5K} + 3W_8^{6K} + W_8^{7K}$$

By IDFT,

$$y_2(n) = 3\delta(n) + 5\delta(n-1) + 5\delta(n-2) + 3\delta(n-3) + 3\delta(n-4) + 4\delta(n-5) + 3\delta(n-6) + \delta(n-7)$$

$\therefore y_2(n) = \{\boxed{3, 5}, 5, 3, 3, 4, 3, 1\} \Rightarrow \langle y_2(n) = \{5, 3, 3, 4, 3, 1\} \rangle$

↑ Discard (M-1) points

Step 3: $y(n) = \{y_1(n), y_2(n)\}$

$\langle y(n) = \{3, 2, 2, 0, 4, 6, 5, 3, 3, 4, 3, 1\} \rangle$

Verification :—

$x(n) \longrightarrow$

$h(n)$ ↓	3	-1	0	1	3	2	0	1	2	1
1	3	-1	0	1	3	2	0	1	2	1
1	3	-1	0	1	3	2	0	1	2	1
1	3	-1	0	1	3	2	0	1	2	1

$y(n) = \{3, 2, 2, 0, 4, 6, 5, 3, 3, 4, 3, 1\}$

6) Consider FIR filter with impulse response $h(n) = \{3, 2, 1, 1\}$, if the input is $x(n) = \{1, 2, 3, 3, 2, 1, -1, -2, -3, 5, 6, -1, 2, 1\}$. using overlap-save method and 8-point circular convolution, find $y(n)$.

Soln :- Given $N = 8$

Since $M = 4$

$N = M + L - 1 \Rightarrow L = N - M + 1$

$= 8 - 4 + 1$

$\langle L = 5 \rangle$

$x_1(n) = \{0, 0, 0, 1, 2, 3, 3, 2\}$

$x_2(n) = \{3, 3, 2, 1, -1, -2, -3, 5\}$ and $h(n) = \{3, 2, 1, 1, 0, 0, 0, 0\}$

$x_3(n) = \{-2, -3, 5, 6, -1, 2, 1, 0\}$

$x_4(n) = \{2, 1, 0, 0, 0, 0, 0, 0\}$

Step 1 : To find $y_1(n) = x_1(n) \circledast_8 h(n)$

By DFT, $Y_1(K) = X_1(K) H(K)$

$= (W_8^{3K} + 2W_8^{4K} + 3W_8^{5K} + 3W_8^{6K} + 2W_8^{7K})(3 + 2W_8^{K} + W_8^{2K} + W_8^{3K})$

$= 3W_8^{3K} + 6W_8^{4K} + 9W_8^{5K} + 9W_8^{6K} + 6W_8^{7K}$

$+ 2W_8^{4K} + 4W_8^{5K} + 6W_8^{6K} + 6W_8^{7K} + 4W_8^{8K}$

$+ W_8^{5K} + 2W_8^{6K} + 3W_8^{7K} + 3W_8^{8K} + 2W_8^{9K}$

$+ W_8^{6K} + 2W_8^{7K} + 3W_8^{8K} + 3W_8^{9K} + 2W_8^{10K}$

$$Y_1(K) = 10 + 5W_8^K + 2W_8^{2K} + 3W_8^{3K} + 8W_8^{4K} + 14W_8^{5K} + 18W_8^{6K} + 17W_8^{7K}$$

By IDFT,

$$y_1(n) = 10\delta(n) + 5\delta(n-1) + 2\delta(n-2) + 3\delta(n-3) + 8\delta(n-4) + 14\delta(n-5) + 18\delta(n-6) + 17\delta(n-7)$$

$\therefore y_1(n) = \{\boxed{10, 5, 2}, 3, 8, 14, 18, 17\} \Rightarrow \langle y_1(n) = \{3, 8, 14, 18, 17\} \rangle$

↑ Discard (M-1) points

Step 2 :- To find $y_2(n) = x_2(n) \circledast_8 h(n)$

By DFT, $Y_2(K) = X_2(K)\,H(K)$

$$= (3 + 3W_8^K + 2W_8^{2K} + W_8^{3K} - W_8^{4K} - 2W_8^{5K} - 3W_8^{6K} + 5W_8^{7K})(3 + 2W_8^K + W_8^{2K} + W_8^{3K}$$

$$= 9 + 9W_8^K + 6W_8^{2K} + 3W_8^{3K} - 3W_8^{4K} - 6W_8^{5K} - 9W_8^{6K} + 15W_8^{7K}$$
$$+ 6W_8^K + 6W_8^{2K} + 4W_8^{3K} + 2W_8^{4K} - 2W_8^{5K} - 4W_8^{6K} - 6W_8^{7K} + 10W_8^{8K}$$
$$+ 3W_8^{2K} + 3W_8^{3K} + 2W_8^{4K} + W_8^{5K} - W_8^{6K} - 2W_8^{7K} - 3W_8^{8K} + 5W_8^{9K}$$
$$+ 3W_8^{3K} + 3W_8^{4K} + 2W_8^{5K} + W_8^{6K} - W_8^{7K} - 2W_8^{8K} - 3W_8^{9K} + 5W_8^{10K}$$

$$Y_2(K) = 14 + 17W_8^K + 20W_8^{2K} + 13W_8^{3K} + 4W_8^{4K} - 5W_8^{5K} - 13W_8^{6K} + 6W_8^{7K}$$

By IDFT,

$$y_2(n) = 14\delta(n) + 17\delta(n-1) + 20\delta(n-2) + 13\delta(n-3) + 4\delta(n-4) - 5\delta(n-5) - 13\delta(n-6) + 6\delta(n-7)$$

$\therefore y_2(n) = \{\boxed{14, 17, 20}, 13, 4, -5, -13, 6\} \Rightarrow \langle y_2(n) = \{13, 4, -5, -13, 6\} \rangle$

↑ Discard (M-1) points

Step 3 :- To find $y_3(n) = x_3(n) \circledast_8 h(n)$

By DFT, $Y_3(K) = X_3(K)\,H(K)$

$$= (-2 - 3W_8^K + 5W_8^{2K} + 6W_8^{3K} - W_8^{4K} + 2W_8^{5K} + W_8^{6K})(3 + 2W_8^K + W_8^{2K} + W_8^{3K}$$

$$= -6 - 9W_8^K + 15W_8^{2K} + 18W_8^{3K} - 3W_8^{4K} + 6W_8^{5K} + 3W_8^{6K}$$
$$- 4W_8^K - 6W_8^{2K} + 10W_8^{3K} + 12W_8^{4K} - 2W_8^{5K} + 4W_8^{6K} + 2W_8^{7K}$$
$$- 2W_8^{2K} - 3W_8^{3K} + 5W_8^{4K} + 6W_8^{5K} - W_8^{6K} + 2W_8^{7K} + W_8^{8K}$$
$$- 2W_8^{3K} - 3W_8^{4K} + 5W_8^{5K} + 6W_8^{6K} - W_8^{7K} + 2W_8^{8K} + W_8^{9K}$$

$Y_3(K) = -3 - 12W_8^K + 7W_8^{2K} + 23W_8^{3K} + 11W_8^{4K} + 15W_8^{5K} + 12W_8^{6K} + 3W_8^{7K}$

By IDFT,

$y_3(n) = -3\delta(n) - 12\delta(n-1) + 7\delta(n-2) + 23\delta(n-3) + 11\delta(n-4) + 15\delta(n-5)$
$+ 12\delta(n-6) + 3\delta(n-7)$

$\therefore y_3(n) = \{\boxed{-3, -12, 7}, 23, 11, 15, 12, 3\} \Rightarrow \langle y_3(n) = \{23, 11, 15, 12, 3\} \rangle$

↑ Discard (M-1) points

Step 4 :- To find $y_4(n) = x_4(n) \circledast_8 h(n)$

By DFT, $Y_4(K) = X_4(K)\, H(K)$

$= (2 + W_8^K)(3 + 2W_8^K + W_8^{2K} + W_8^{3K})$

$= 6 + 4W_8^K + 2W_8^{2K} + 2W_8^{3K}$
$+ 3W_8^K + 2W_8^{2K} + W_8^{3K} + W_8^{4K}$

$Y_4(K) = 6 + 7W_8^K + 4W_8^{2K} + 3W_8^{3K} + W_8^{4K}$

By IDFT, $y_4(n) = 6\delta(n) + 7\delta(n-1) + 4\delta(n-2) + 3\delta(n-3) + \delta(n-4)$

$\therefore y_4(n) = \{\boxed{6, 7, 4}, 3, 1, 0, 0, 0\} \Rightarrow \langle y_4(n) = \{3, 1, 0, 0, 0\} \rangle$

↑ Discard (M-1) points

Step 5 :- $y(n) = \{y_1(n), y_2(n), y_3(n), y_4(n)\}$

$\langle \therefore y(n) = \{3, 8, 14, 18, 17, 13, 4, -5, -13, 6, 23, 11, 15, 12, 3, 3, 1\} \rangle$

Verification :-

$x(n) \rightarrow$

	1	2	3	3	2	1	-1	-2	-3	5	6	-1	2	1
h(n) 3	3	6	9	9	6	3	-3	-6	-9	15	18	-3	6	3
↓ 2	2	4	6	6	4	2	-2	-4	-6	10	12	-2	4	2
1	1	2	3	3	2	1	-1	-2	-3	5	6	-1	2	1
1	1	2	3	3	2	1	-1	-2	-3	5	6	-1	2	1

$y(n) = \{3, 8, 14, 18, 17, 13, 4, -5, -13, 6, 23, 11, 15, 12, 3, 3, 1\}$

Module-3

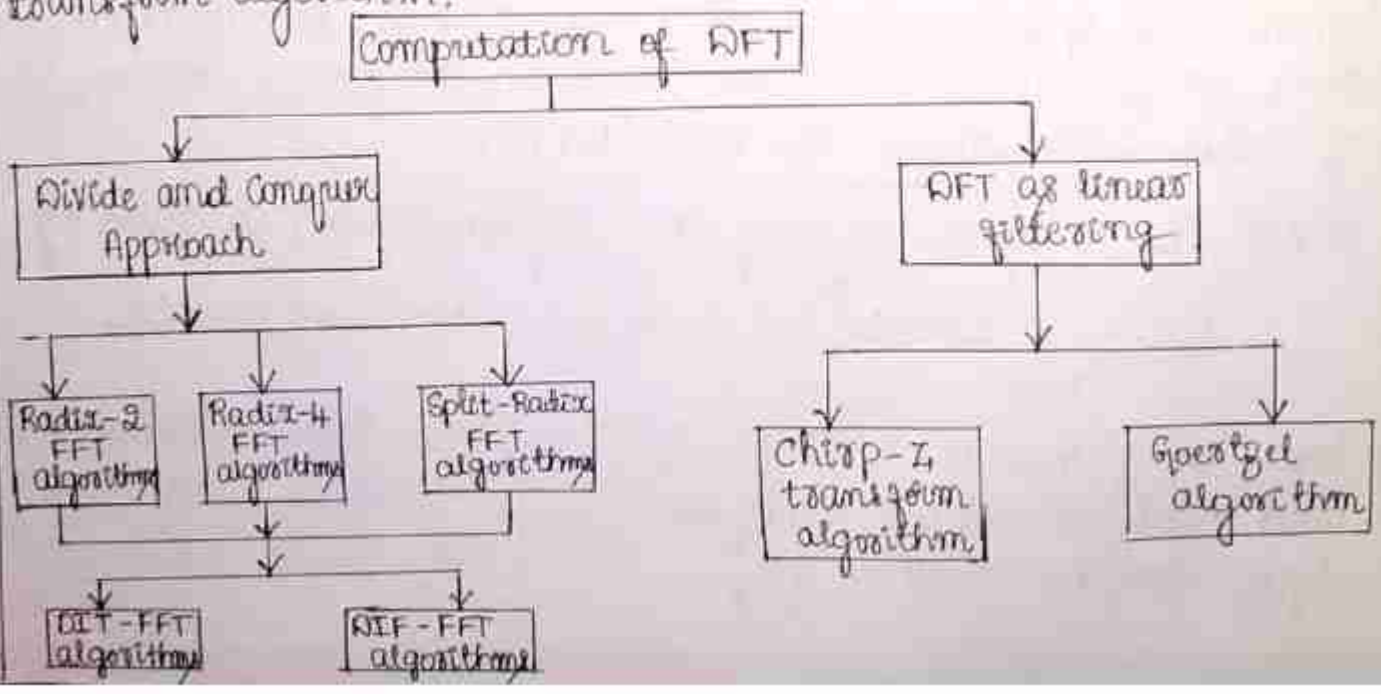
Anusha. M.N
Asst. professor
Dept. of ECE
BGSIT, B.G. Nagar

Radix-2 FFT Algorithms and Linear Filtering

Fast-Fourier transform (FFT) is most important technique, popularized by Cooley and Tukey is based on decomposing (or) breaking the transform into smaller transforms and combining them to give the total transform. As Cooley and Tukey have shown that, this decomposition process can be done in both the time and frequency-domains.

Classification of FFT algorithms :-

FFT algorithms are based on two basic methods. The first one is divide and conquer approach. In this method, the 'N' point DFT is divided successively to 2-point DFT's to reduce calculations.
In this method, radix-2, radix-4 decimation in time and decimation in frequency etc. type of FFT algorithms are developed.
The second one is based on linear filtering. Based on this method, there are two algorithms : Goertzel algorithm and chirp-Z transform algorithm.



Scanned with CamScanner

W.K.T

Radix-2 FFT algorithms :-

The radix-2 FFT algorithms are based on divide and conquer approach. In this approach, the N-point DFT is successively decomposed into smaller DFT's. Because of this decomposition, the number of computations are reduced.
Let value of 'N' be selected such that $N = 2^v$. This N-point DFT is decomposed successively such that smallest DFT will be of size N=2. Hence, this type of algorithms are called as radix-2 (or) radix of these algorithm is '2'.

1) Radix-2 decimation-in-time (DIT) FFT algorithm :-

First decomposition on x(n) :-

Let the N-point data sequence x(n) be splitted into two $\frac{N}{2}$-point data sequences $f_1(n)$ and $f_2(n)$.
Let $f_1(n)$ contain even numbered samples of x(n) and $f_2(n)$ contain odd numbered samples of x(n).

Thus, $$\left.\begin{aligned} f_1(n) &= x(2n) \quad ; n = 0, 1, \ldots \frac{N}{2}-1 \\ f_2(n) &= x(2n+1) \; ; n = 0, 1, \ldots \frac{N}{2}-1 \end{aligned}\right\} \longrightarrow (1)$$

Here, time-domain sequence x(n) is splitted into two sequences. This splitting operation is called decimation. Since, it is done on time-domain sequence. Hence, it is called Decimation-in-time (DIT).

W.K.T the N-point DFT of the sequence x(n) is given by,

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \; ; K = 0, 1, \ldots (N-1) \longrightarrow (2)$$

Since, the sequence x(n) is splitted into even numbered and odd numbered samples.
Eq(2) can be written as,

Scanned with CamScanner

$$X(K) = \sum_{n \text{ even}} x(n) W_N^{Kn} + \sum_{n \text{ odd}} x(n) W_N^{Kn}$$

$$X(K) = \sum_{n=0}^{\frac{N}{2}-1} x(2n) W_N^{2nK} + \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) W_N^{(2n+1)K}$$

From Eq(1), $f_1(n) = x(2n)$ and

$f_2(n) = x(2n+1)$

$$\therefore X(K) = \sum_{n=0}^{\frac{N}{2}-1} f_1(n) (W_N^2)^{Kn} + \sum_{n=0}^{\frac{N}{2}-1} f_2(n) (W_N^2)^{Kn} \cdot W_N^K$$

$$X(K) = \sum_{n=0}^{\frac{N}{2}-1} f_1(n) W_{N/2}^{Kn} + W_N^K \sum_{n=0}^{\frac{N}{2}-1} f_2(n) W_{N/2}^{Kn} \longrightarrow (3)$$

Compare Eq(3) with the definition of DFT of Eq(2), the first summation represents $\frac{N}{2}$-point DFT of $f_1(n)$ and second summation represents $\frac{N}{2}$-point DFT of $f_2(n)$.

Thus, $F_1(K)$ is $\frac{N}{2}$-point DFT of $f_1(n)$ and $F_2(K)$ is $\frac{N}{2}$-point DFT of $f_2(n)$.

Thus $F_1(K)$ is $\frac{N}{2}$-point DFT of $f_1(n)$ and $F_2(K)$ is $\frac{N}{2}$-point DFT of $f_2(n)$.

i.e., $X(K) = F_1(K) + W_N^K F_2(K)$; $K = 0, 1, \ldots (N-1) \longrightarrow (4)$

Since $F_1(K)$ and $F_2(K)$ are $\frac{N}{2}$-point DFT's, they are periodic with period $\frac{N}{2}$.

i.e., $F_1\left(K + \frac{N}{2}\right) = F_1(K)$ and

$F_2\left(K + \frac{N}{2}\right) = F_2(K)$ $\Bigg\} \longrightarrow (5)$

Hence, replacing 'K' by $K + \frac{N}{2}$ in Eq(4),

$$X\left(K + \frac{N}{2}\right) = F_1\left(K + \frac{N}{2}\right) + W_N^{K+\frac{N}{2}} F_2\left(K + \frac{N}{2}\right)$$

Scanned with CamScanner

$\therefore X\left(K+\frac{N}{2}\right) = F_1(K) - W_N^K F_2(K)$, Since $W_N^{K+\frac{N}{2}} = -W_N^K$ ⟶ (6)

Where X(K) is N-point DFT. By taking $K = 0, 1, \ldots \frac{N}{2}-1$, $F_1(K)$ and $F_2(K)$ can be calculated since they are $\frac{N}{2}$-point DFT's. The N-point DFT X(K) can be combinely obtained from Eq(4) and Eq(6) by taking $K = 0$ to $\frac{N}{2}-1$.

i.e., $X(K) = F_1(K) + W_N^K F_2(K)$; $K = 0, 1, \ldots \frac{N}{2}-1$ ⟶ (7)

$$X\left(K+\frac{N}{2}\right) = F_1(K) - W_N^K F_2(K) \; ; \; K = 0, 1, \ldots \frac{N}{2}-1 \longrightarrow (8)$$

The above two equations shows that N-point DFT can be obtained by two $\frac{N}{2}$-point DFT's.

An Example of 8-point DIT-FFT :-

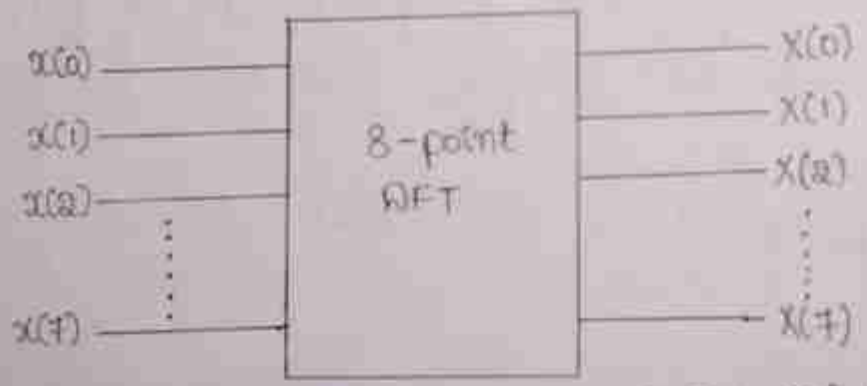


fig 1 : Direct Computation of 8-point DFT

The 8-point DFT can be Computed directly and hence no reduction in Computation. It is shown symbolically as a single-block which Computes 8-point DFT directly.

According to Eq(7) and Eq(8), X(K) can be obtained from $F_1(K)$ and $F_2(K)$. Here $F_1(K)$ and $F_2(K)$ are two 4-point DFT's.

Scanned with CamScanner

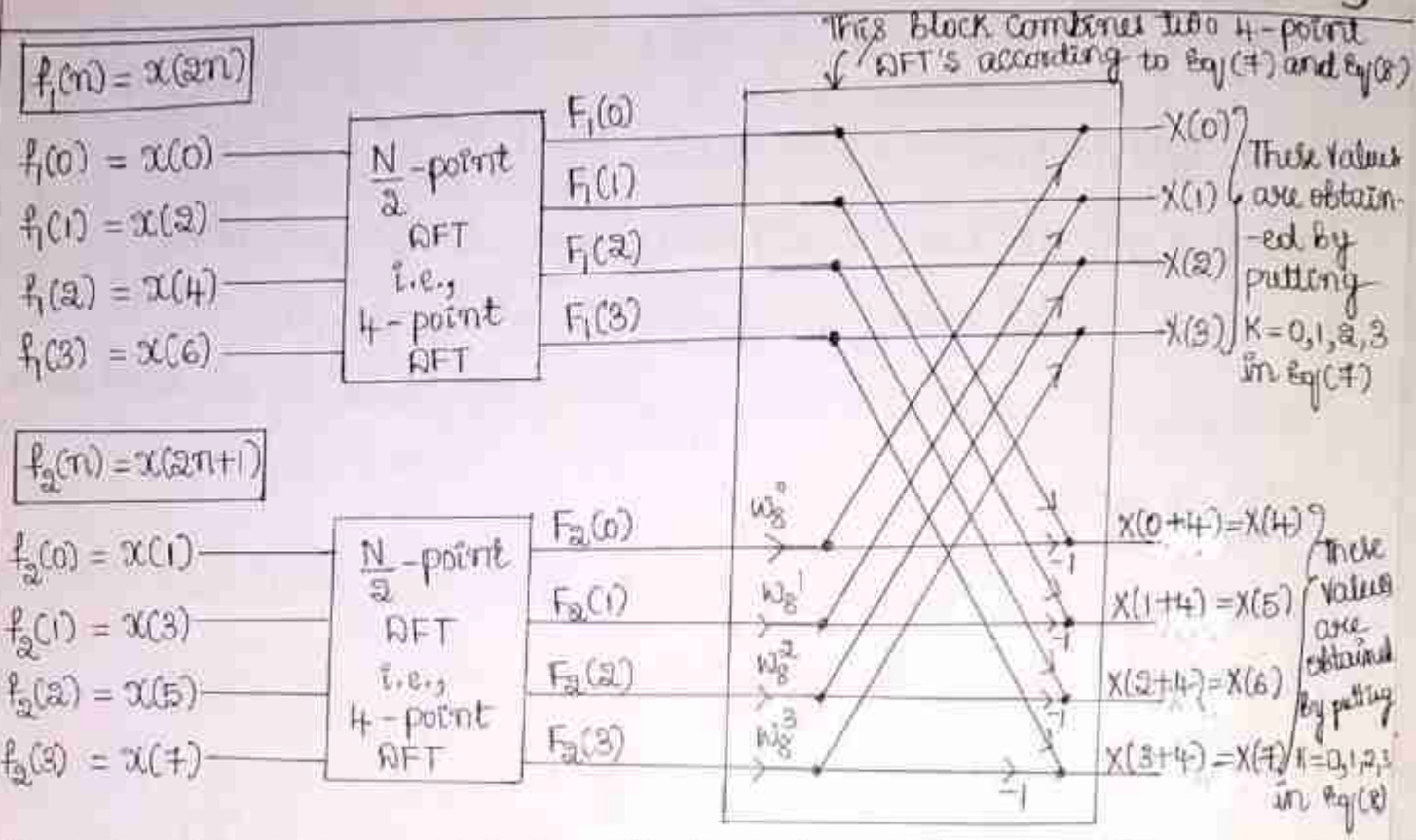


Fig 3: 8-point DFT of X(K) is obtained by combining two 4-point DFT's $F_1(K)$ and $F_2(K)$

In fig 3, observe that for 8-point x(n), the sequences $f_1(n)$ and $f_2(n)$ are given as:

$$f_1(n) = x(2n) = \{x(0), x(2), x(4), x(6)\}$$
$$f_2(n) = x(2n+1) = \{x(1), x(3), x(5), x(7)\}$$
$\Big\}$ $\frac{N}{2}$ point sequences $\rightarrow$ (9)

As shown in fig 3, two $\frac{N}{2}$-point DFT's are to be computed separately and then they are combined as per Eq(7) and Eq(8) to get N-point DFT.

Second stage of decimation on $f_1(n)$ and $f_2(n)$:-

Each of $\frac{N}{2}$-point sequences can be decimated further into two sequences of lengths $\frac{N}{4}$. W.K.T $f_1(n)$ and $f_2(n)$ are $\frac{N}{2}$-point sequences.

Let $f_1(n)$ be splitted into even-numbered samples and odd-numbered samples as follows:

Scanned with CamScanner

$$V_{11}(n) = f_1(2n) \; ; \; n = 0, 1, \ldots \frac{N}{4} - 1$$
$$\text{and } V_{12}(n) = f_1(2n+1) \; ; \; n = 0, 1, \ldots \frac{N}{4} - 1 \quad \Bigg\} \longrightarrow (10)$$

Where, $V_{11}(n)$ is Even-numbered Sequence of $f_1(n)$ and
$V_{12}(n)$ is odd-numbered Sequence of $f_1(n)$.

Since, $f_1(n)$ Contain $\left(\frac{N}{2}\right)$ Samples, $V_{11}(n)$ and $V_{12}(n)$ Contains $\left(\frac{N}{4}\right)$ samples.

Similarly, $f_2(n)$ Can be Splitted into Even and odd numbered Sequence of $f_2(n)$ as follows:

$$V_{21}(n) = f_2(2n) \; ; \; n = 0, 1, \ldots \frac{N}{4} - 1$$
$$\text{and } V_{22}(n) = f_2(2n+1) \; ; \; n = 0, 1, \ldots \frac{N}{4} - 1 \quad \Bigg\} \longrightarrow (11)$$

Where,
$V_{21}(n)$ is Even numbered Sequence of $f_2(n)$ and
$V_{22}(n)$ is odd numbered Sequence of $f_2(n)$.

Both $V_{21}(n)$ and $V_{22}(n)$ Contain $\frac{N}{4}$-samples Each.

We obtained $X(K)$ and $X\left(K+\frac{N}{2}\right)$ from $F_1(K)$ and $F_2(K)$ with $f_1(n)$ and $f_2(n)$ as decimated Sequences. The length of DFT was $\frac{N}{2}$. By using the Similar method, we can obtain $F_1(K)$ and $F_1\left(K+\frac{N}{4}\right)$ from $V_{11}(K)$ and $V_{12}(K)$.

Here, $V_{11}(K)$ is $\frac{N}{4}$-point DFT of $V_{11}(n)$ and $V_{12}(K)$ is $\frac{N}{4}$-point DFT of $V_{12}(n)$ of Eq(10).

i.e., $F_1(K) = V_{11}(K) + W_{N/2}^{K} V_{12}(K) \; ; \; K = 0, 1, \ldots \frac{N}{4} - 1 \longrightarrow (12)$

$F_1\left(K+\frac{N}{4}\right) = V_{11}(K) - W_{N/2}^{K} V_{12}(K) \; ; \; K = 0, 1, \ldots \frac{N}{4} - 1 \longrightarrow (13)$

Scanned with CamScanner

In Eq(7) and Eq(8), N-point DFT is obtained from two $\frac{N}{2}$-point DFT's. In Eq(12) and Eq(13), $\frac{N}{2}$-point DFT (i.e., $F_1(K)$) is obtained from two $\frac{N}{4}$-point DFT's.

Similarly,

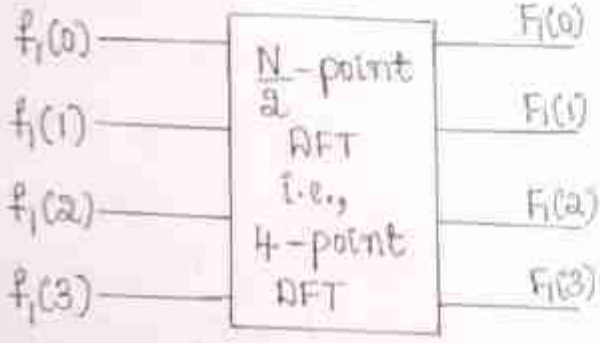
$$F_2(K) = V_{21}(K) + W_{N/2}^{K} V_{22}(K) \; ; \; K = 0, 1, \ldots, \frac{N}{4} - 1 \longrightarrow (14)$$

$$F_2\left(K + \frac{N}{4}\right) = V_{21}(K) - W_{N/2}^{K} V_{22}(K) \; ; \; K = 0, 1, \ldots, \frac{N}{4} - 1 \longrightarrow (15)$$

Where, $V_{21}(K)$ is $\frac{N}{4}$-point DFT of $V_{21}(n)$ and

$V_{22}(K)$ is $\frac{N}{4}$-point DFT of $V_{22}(n)$.

Ex:- 8-point DIT-FFT



a) A Symbolic representation of 4-point DFT $F_1(K)$ and direct Computation



b) A Symbolic representation of 4-point DFT $F_2(K)$ using direct Computation

As shown in fig, $F_1(K)$ and $F_2(K)$ are two 4-point DFT's Computed directly. These two DFT's are represented by single-blocks Symbolically as shown in fig.

$F_1(K)$ can be obtained from $V_{11}(K)$ and $V_{12}(K)$ as per Eq(12) and Eq(13).

For $N = 8$, $V_{11}(K)$ and $V_{12}(K)$ will be $\frac{N}{4}$ i.e., 2-point DFT's.

Scanned with CamScanner

$V_{11}(n) = f_1(2n) = x(4n)$

$V_{11}(0) = f_1(0) = x(0)$

$V_{11}(1) = f_1(2) = x(4)$

$\frac{N}{4}$-point DFT i.e., 2-point DFT

$V_{11}(0)$, $V_{11}(1)$

$F_1(0)$, $F_1(1)$ — These values are obtained by putting $K=0,1$ in Eq(12)

$V_{12}(n) = f_1(2n+1) = x(4n+2)$

$V_{12}(0) = f_1(1) = x(2)$

$V_{12}(1) = f_1(3) = x(6)$

$\frac{N}{4}$-point DFT i.e., 2-point DFT

$V_{12}(0)$, $V_{12}(1)$; W_4^0, W_4^1; -1, -1

$F_1(0+2) = F_1(2)$, $F_1(1+2) = F_1(3)$ — These values are obtained by putting $K=0,1$ in Eq(13)

This block combines two 2-point DFT's according to Eq(12) and Eq(13)

fig(a): 4-point DFT of $F_1(K)$ is obtained by combining two 2-point DFT's of $V_{11}(K)$ and $V_{12}(K)$

Fig(a) shows the symbolic diagram for 2-point DFT's. For 4-point $f_1(n)$, the sequences $V_{11}(n)$ and $V_{12}(n)$ are given as:

$$V_{11}(n) = f_1(2n) = x(4n) = \{x(0), x(4)\}\ ; n = 0, 1$$

$$V_{12}(n) = f_1(2n+1) = x(4n+2) = \{x(2), x(6)\}\ ; n = 0, 1$$

} $\frac{N}{4}$-point sequences ⟶ (16)

Thus, the two $\frac{N}{4}$-point DFT's are to be computed separately and then they are combined as per Eq(12) and Eq(13) to get $\frac{N}{2}$-point DFT of $F_1(K)$.

$V_{21}(n) = f_2(2n) = x(4n+1)$

$V_{21}(0) = f_2(0) = x(1)$

$V_{21}(1) = f_2(2) = x(5)$

$\frac{N}{4}$-point DFT i.e., 2-point DFT

$V_{21}(0)$, $V_{21}(1)$

$F_2(0)$, $F_2(1)$ — These values are obtained by putting $K=0,1$ in Eq(14)

$V_{22}(n) = f_2(2n+1) = x(4n+3)$

$V_{22}(0) = f_2(1) = x(3)$

$V_{22}(1) = f_2(3) = x(7)$

$\frac{N}{4}$-point DFT i.e., 2-point DFT

$V_{22}(0)$, $V_{22}(1)$; W_4^0, W_4^1; -1, -1

$F_2(0+2) = F_2(2)$, $F_2(1+2) = F_2(3)$ — These values are obtained by putting $K=0,1$ in Eq(15)

This block combines two 2-point DFT's according to Eq(14) and Eq(15).

Fig(b): 4-point DFT of $f_2(K)$ is obtained by combining two 2-point DFT's of $V_{21}(K)$ and $V_{22}(K)$

Scanned with CamScanner

Fig(6) shows the computation of $F_2(K)$ as per Eq(14) and Eq(15). In this figure, the two $\frac{N}{4}$-point DFT's are combined. The $\frac{N}{4}$-point sequences $V_{21}(n)$ and $V_{22}(n)$ sequences are given as :

$$V_{21}(n) = f_2(2n) = x(4n+1) = \{x(1), x(5)\}\ ; n = 0,1$$
$$V_{22}(n) = f_2(2n+1) = x(4n+3) = \{x(3), x(7)\}\ ; n = 0,1$$
$\left.\begin{matrix}\\ \\ \end{matrix}\right\}$ $-\frac{N}{4}$ point sequences $\longrightarrow$ (17)

Computation of 2-point DFT's :-

The $\frac{N}{4}$-point sequences of Eq(10) and Eq(11) are further splitted in their even and odd parts. Hence, we get next stage of decimation and the sequences will be of length $\frac{N}{8}$.

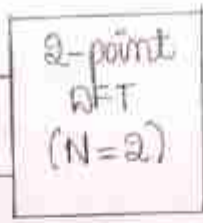
The procedure discussed till now can be repeated further to compute the DFT's, till we reach to 2-point DFT. Since, $N = 2^V$, we reach to 2-point DFT after $(V-1)$ decimations.

Ex :- For $N=8$, we performed $(3-1) = 2$ (Since $V=3$) decimations to reach 2-point DFT's.

$v_{11}(n)\ ; n = 0,1$

$v_{11}(0) = f_1(0) = x(0)$

$v_{11}(1) = f_1(2) = x(4)$



$V_{11}(0)$

$V_{11}(1)$

$V_{11}(K)\ ; K = 0,1$

Computation of 2-point DFT $V_{11}(K)$ of fig(a)

From the fig,

$$V_{11}(K) = \sum_{n=0}^{1} v_{11}(n) W_2^{Kn}\ ; K = 0,1 \longrightarrow (18)$$

For $K=0$, $V_{11}(0) = \sum_{n=0}^{1} v_{11}(n) W_2^0$

$= \sum_{n=0}^{1} v_{11}(n)$, since $W_2^0 = 1$

Scanned with CamScanner

$$V_{11}(0) = \vartheta_{11}(0) + \vartheta_{11}(1) \longrightarrow (19)$$

For $K=1$, $V_{11}(1) = \sum_{n=0}^{1} \vartheta_{11}(n) W_2^n$

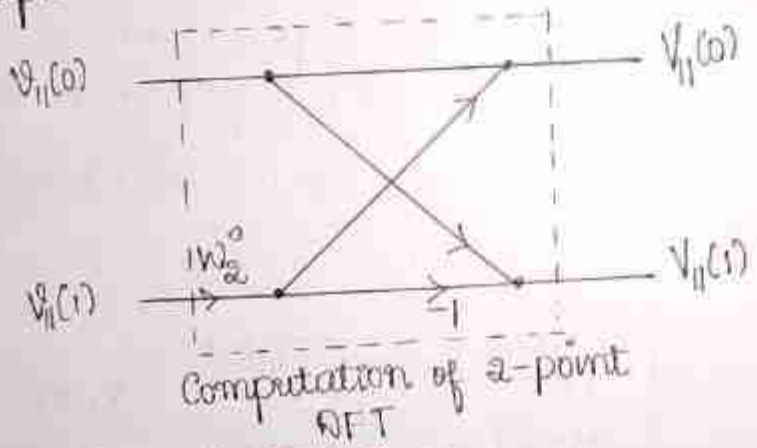
$$= \vartheta_{11}(0) W_2^0 + \vartheta_{11}(1) W_2^1$$

$$V_{11}(1) = \vartheta_{11}(0) - \vartheta_{11}(1) \longrightarrow (20)$$ Since $W_2^0 = 1$, $W_2^1 = -1$

W.K.T $W_2^0 = 1$. Let us multiply $\vartheta_{11}(1)$ by W_2^0 in eq(19) and eq(20). Note that this does not make any difference.

$$\therefore V_{11}(0) = \vartheta_{11}(0) + W_2^0 \vartheta_{11}(1)$$
$$V_{11}(1) = \vartheta_{11}(0) - W_2^0 \vartheta_{11}(1) \Big\} \longrightarrow (21)$$

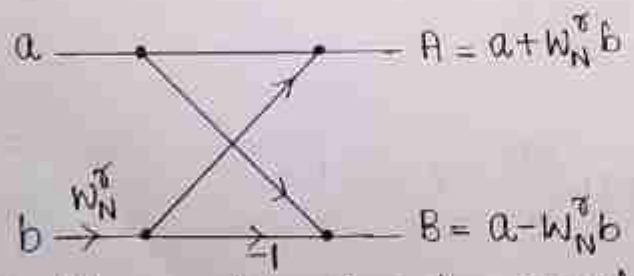
This computation is illustrated below with the help of signal flowgraph.



Computation of 2-point DFT

Butterfly Computation :-

The below figure shows the structure of butterfly used for DIT.

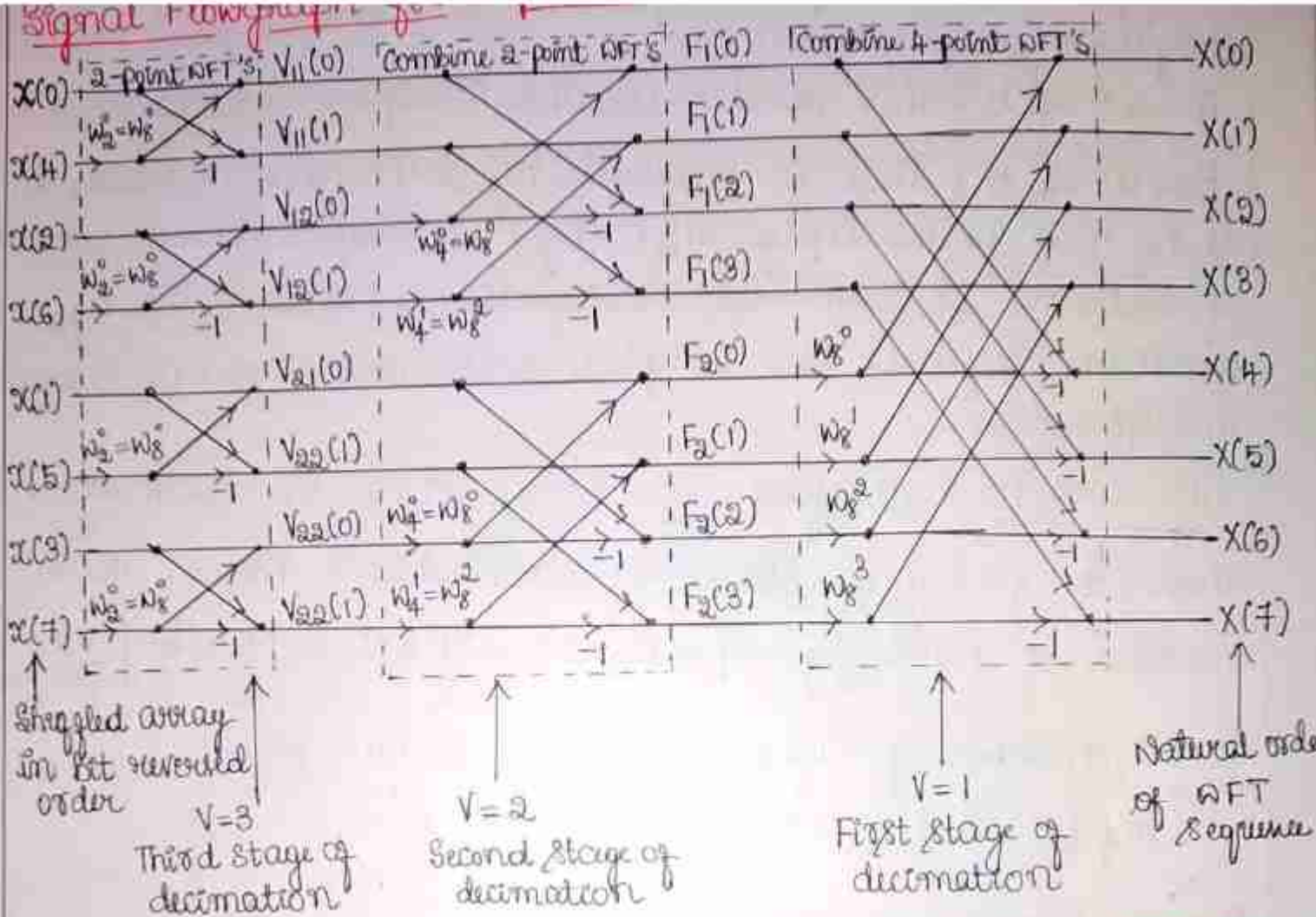


Here 'a' and 'b' are two Complex numbers. By arithmetic manipulation from this butterfly, two new Complex numbers A and B are generated.

The above figure which is called signal flowgraph resembles a butterfly. hence it is called butterfly.

Scanned with CamScanner

Signal Flowgraph of 8-point DFT



Computational Complexity Compared to direct Computation:-

a) Computational Complexity of FFT algorithm :-

FFT algorithms reduces the number of computations. The computations involved in the butterfly operation are shown in fig below.

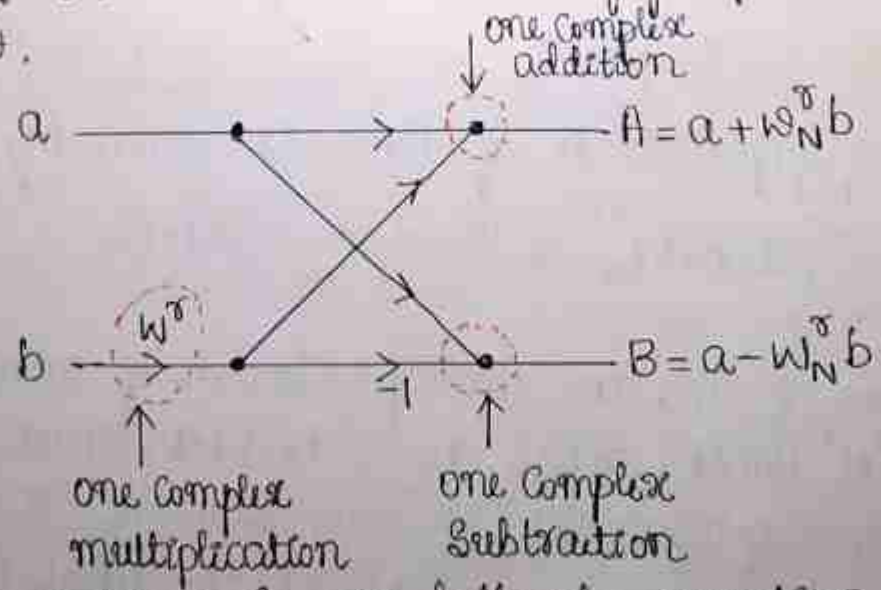


Fig : Computations in one butterfly operation. Observe that 'Subtraction' operation avoids multiplications by '-1'

Scanned with CamScanner

In fig, 'b' is multiplied by 'W_N^r' which results in one Complex multiplication.

$A = a + W_N^r b$; This results in one Complex addition.

$B = a - W_N^r b$; This is obtained by subtraction. Hence, there is no need to multiply $W_N^r b$ by -1. Computational Complexity of addition is same as subtraction, hence butterfly operation requires two Complex additions and one Complex multiplication.

There are $\frac{N}{2}$ butterflies in Every stage of decimation.

There are $V = \log_2 N$ stages of decimation. Hence, total number of butterfly operations $= \frac{N}{2} \times V = \frac{N}{2} \times \log_2 N$

Hence, number of Complex multiplications will be,

$$\text{Complex multiplications} = \frac{N}{2} \log_2 N$$

This is because one butterfly needs one multiplication and two additions are required for one butterfly.

$\therefore$ Complex additions $= 2 \times \frac{N}{2} \log_2 N$

$$\text{Complex additions} = N \log_2 N$$

b) Computational Complexity of direct Computation :-

The direct Computation of DFT is given by,

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \; ; \; K = 0, 1, \ldots (N-1)$$

For one Value of 'K', the multiplication of $x(n)$ and W_N^{Kn} is done for 'N' times, Since, $n = 0$ to $N-1$. That is there are 'N' Complex multiplications for one Value of 'K'. Since 'K' also has N values (Since $K = 0, 1, \ldots (N-1)$), the total Complex multiplications for Computation of $X(K)$ Will be,

Scanned with CamScanner

<Complex multiplications = $N \times N = N^2$>

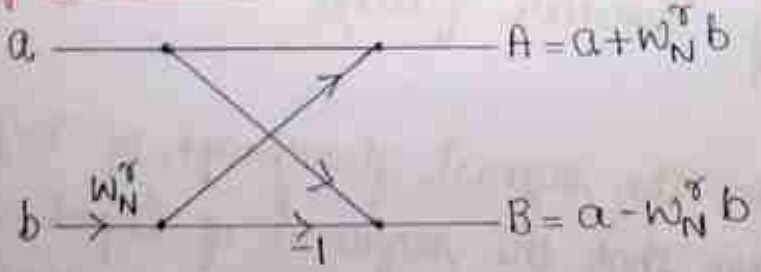
Similarly, for one value of 'K', $\sum_{n=0}^{N-1}$ is performed on $x(n) W_N^{Kn}$. Thus, to add total 'N' (0 to N-1) times, 'N-1' complex additions are required. Since 'K' also has 'N' values, the total complex additions for computations of X(K) will be,

<Complex additions = $N(N-1) = N^2 - N$>

Number of points 'N'	Direct computation		DIT-FFT algorithm		Improvement in processing speed for $*^{ns}$, $\frac{N^2}{\frac{N}{2}\log_2 N}$
	Complex $*^{ns}$, N^2	Complex $+^{ns}$, N^2-N	Complex $*^{ns}$, $\frac{N}{2}\log_2 N$	Complex $+^{ns}$, $N \log_2 N$	
8	64	56	12	24	5.3 times
16	256	240	32	64	8 times
256	65536	65280	1024	2048	6.4 times
1024	1048576	1047552	5120	10240	204.8 times

Computational Complexity of DIT-FFT algorithm Compared to direct method

Memory requirement and Inplace computations :—



The above figure shows the butterfly computation used in FFT. From values 'a' and 'b', new values 'A' and 'B' are computed. Once 'A' and 'B' are computed, there is no need to store 'a' and 'b'. Thus, same memory locations can be used to store A, B where a, b were stored. Since A, B (or) a, b are complex numbers, they need two memory locations each. Thus, for

Scanned with CamScanner

Computation of one butterfly, four memory locations are required, i.e., two for 'a' (or) 'A' and two for 'B' and 'B'.

Memory locations for one butterfly $= 2 \times 2 = \underline{4}$

In such computation, 'A' is stored in place of 'a' and 'B' is stored in place of 'b'. This is called inplace computation.

The advantage of inplace computation is that it reduces memory requirement.

Normally, the computations are performed stagewise. In every stage, there are $\frac{N}{2}$ (i.e., 4 for N=8) butterflies. Hence, number of memory locations required for one stage of will be,

Memory locations for one stage $= 4 \times \frac{N}{2} = 2N$

Thus '2N' locations are required for one stage. Since, stages are computed successively, these locations can be shared. Hence, total 2N locations are required for computation of N-point DFT. In every stage, there are $\frac{N}{2}$ (i.e., 4 for N=8) twiddle factors required.

∴ memory requirement of N-point DFT $= \underline{2N}$

Maximum memory requirement of N-point DFT including twiddle factors $= \left(2N + \frac{N}{2}\right)$

Bit Reversal :-

Consider again the signal flowgraph of 8-point DIT-FFT algorithm. observe that the sequence of input data is shuffled as x(0), x(4), x(2), x(6), x(1), x(5), x(3), x(7) and the DFT sequence X(K) at the output is in proper order i.e., X(0), X(1), X(2), X(3), X(4), X(5), X(6), X(7). The shuffling of the input sequence has well-defined format.

Scanned with CamScanner

Memory address of x(n) in decimal i.e., 'n'	Memory address of x(n) in binary n_2 n_1 n_0	Memory address in bit reversed order n_0 n_1 n_2	New memory address of x(n) according to reversed order of bits 'm'
0	0 0 0	0 0 0	0
1	0 0 1	1 0 0	4
2	0 1 0	0 1 0	2
3	0 1 1	1 1 0	6
4	1 0 0	0 0 1	1
5	1 0 1	1 0 1	5
6	1 1 0	0 1 1	3
7	1 1 1	1 1 1	7

This is the order in which data is to be stored and applied as an input

In the table, observe that the data point $x(1) \equiv x(001)$ is to be placed at $m=4$ i.e., $(100)^{th}$ position in the decimated array. The last column of above table shows the order in which the data is required.
Thus, the input data should be stored in the bit-reversed order then the DFT will be obtained in natural sequence.
In the FFT algorithms, bit reversal is always one of the important parts.

(i) Radix-2 DIF-FFT algorithms :-

First stage of decomposition :-

The N-point DFT is given by,

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \; ; \; 0 \le K \le N-1 \longrightarrow (1)$$

Let us split this formula into two summations. The first summation will be for first $\frac{N}{2}$ data points and second summation will be for remaining $\frac{N}{2}$ data points as shown below :

$$X(K) = \sum_{n=0}^{\frac{N}{2}-1} x(n) W_N^{Kn} + \sum_{n=\frac{N}{2}}^{N-1} x(n) W_N^{Kn} \longrightarrow (2)$$

Scanned with CamScanner

Let us rearrange the second summation of Eq(2) as follows:

$$X(K) = \sum_{n=0}^{\frac{N}{2}-1} x(n) W_N^{Kn} + \sum_{n=0}^{\frac{N}{2}-1} x\left(n+\frac{N}{2}\right) W_N^{K\left(n+\frac{N}{2}\right)} \longrightarrow (3)$$

$$= \sum_{n=0}^{\frac{N}{2}-1} x(n) W_N^{Kn} + W_N^{KN/2} \sum_{n=0}^{\frac{N}{2}-1} x\left(n+\frac{N}{2}\right) W_N^{Kn} \longrightarrow (4)$$

In the above Eq^n, $W_N^{KN/2}$ can be Expressed as :

$$W_N^{KN/2} = e^{\frac{-j2\pi}{N}\cdot\frac{KN}{2}} = e^{-j\pi K} = (\cos\pi - j\sin\pi)^K = (-1-j(0))^K = (-1)^K$$

$\therefore$ Eq(4) can be written as,

$$X(K) = \sum_{n=0}^{\frac{N}{2}-1} x(n) W_N^{Kn} + (-1)^K \sum_{n=0}^{\frac{N}{2}-1} x\left(n+\frac{N}{2}\right) W_N^{Kn}$$

$$= \sum_{n=0}^{\frac{N}{2}-1} \left[x(n) + (-1)^K x\left(n+\frac{N}{2}\right)\right] W_N^{Kn} \longrightarrow (5)$$

Now, let us split X(K) into Even and odd numbered samples:

$$X(2K) = \sum_{n=0}^{\frac{N}{2}-1} \left[x(n) + (-1)^{2K} x\left(n+\frac{N}{2}\right)\right] W_N^{2Kn} \longrightarrow (6)$$

; $K = 0, 1, \ldots \frac{N}{2}-1$

and

$$X(2K+1) = \sum_{n=0}^{\frac{N}{2}-1} \left[x(n) + (-1)^{2K+1} x\left(n+\frac{N}{2}\right)\right] W_N^{(2K+1)n} \quad ; K = 0, 1, \ldots \frac{N}{2}-1$$

$\hookrightarrow (7)$

In Eq(6) above, $(-1)^{2K} = 1$ always and W_N^{2Kn} will be,

$$W_N^{2Kn} = (W_N^2)^{Kn} = W_{N/2}^{Kn}$$

$\therefore$ Eq(6) becomes,

$$X(2K) = \sum_{n=0}^{\frac{N}{2}-1} \left[x(n) + x\left(n+\frac{N}{2}\right)\right] W_{N/2}^{Kn} \; ; K = 0, 1, \ldots \frac{N}{2}-1 \longrightarrow (8)$$

Scanned with CamScanner

Now Consider Eq(7), $(-1)^{2K+1} = (-1)^{2K}\cdot(-1) = -1$ always and

$W_N^{(2K+1)n} = W_N^{2Kn+n} = W_N^{2Kn}\cdot W_N^{n} = W_{N/2}^{Kn}\cdot W_N^{n}$

$\therefore$ Eq(7) becomes,

$$X(2K+1) = \sum_{n=0}^{\frac{N}{2}-1} \left\{\left[x(n) - x\left(n+\frac{N}{2}\right)\right] W_N^n\right\} W_{N/2}^{Kn} \longrightarrow (9)$$

Eq(8) and Eq(9) combinely gives DFT X(K). Let us define the two $\frac{N}{2}$ point sequences $g_1(n)$ and $g_2(n)$ as,

$$g_1(n) = x(n) + x\left(n+\frac{N}{2}\right) \; ; n = 0, 1, \ldots, \frac{N}{2}-1 \longrightarrow (a)$$

and $$g_2(n) = \left[x(n) - x\left(n+\frac{N}{2}\right)\right] W_N^n \; ; n = 0, 1, \ldots, \frac{N}{2}-1 \longrightarrow (b)$$

Then, Eq(8) and Eq(9) can be written as :

$$X(2K) = \sum_{n=0}^{\frac{N}{2}-1} g_1(n)\, W_{N/2}^{Kn} \; ; K = 0, 1, \ldots, \frac{N}{2}-1 \longrightarrow (10)$$

and

$$X(2K+1) = \sum_{n=0}^{\frac{N}{2}-1} g_2(n)\, W_{N/2}^{Kn} \; ; K = 0, 1, \ldots, \frac{N}{2}-1 \longrightarrow (11)$$

Here, N-point DFT is splitted into two $\frac{N}{2}$-points DFT's. Since X(K) is decimated with 'K' Even and 'K' odd, this is Decimation-in-frequency (DIF) FFT.

'K' represents frequency Components in the range of 0 to 2π.

An Example of N=8 point DFT:

The Values of $g_1(n)$ and $g_2(n)$ of Eq(a) and Eq(b) becomes,

$g_1(n) = x(n) + x(n+4)$ from Eq(a) with N=8

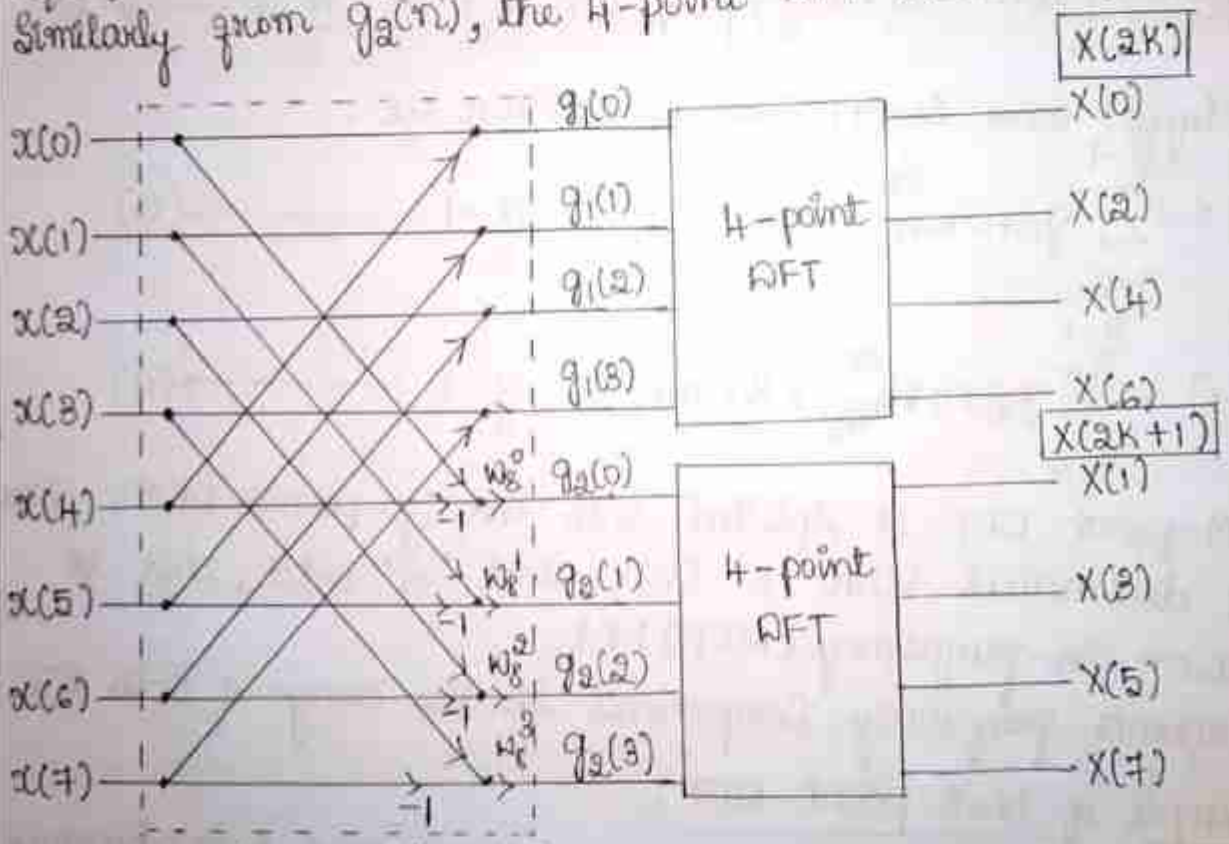
$$\left.\begin{aligned} g_1(0) &= x(0) + x(4) \\ g_1(1) &= x(1) + x(5) \\ g_1(2) &= x(2) + x(6) \\ g_1(3) &= x(3) + x(7) \end{aligned}\right\} \; ; n = 0, 1, 2, 3 \longrightarrow (12)$$

Scanned with CamScanner

and $g_2(n) = [x(n) - x(n+4)] W_8^n$ from Eq(6) with N = 8

$$\left.\begin{aligned} g_2(0) &= [x(0) - x(4)] W_8^0 \\ g_2(1) &= [x(1) - x(5)] W_8^1 \\ g_2(2) &= [x(2) - x(6)] W_8^2 \\ g_2(3) &= [x(3) - x(7)] W_8^3 \end{aligned}\right\} \; ; n = 0, 1, 2, 3 \qquad \longrightarrow (13)$$

Figure below shows the partial signal flowgraph for computation of $g_1(n)$ and $g_2(n)$ according to Eq(12) and Eq(13). This figure also shows that from $g_1(n)$, the 4-point DFT X(2K) is obtained. Similarly from $g_2(n)$, the 4-point DFT X(2K+1) is obtained.

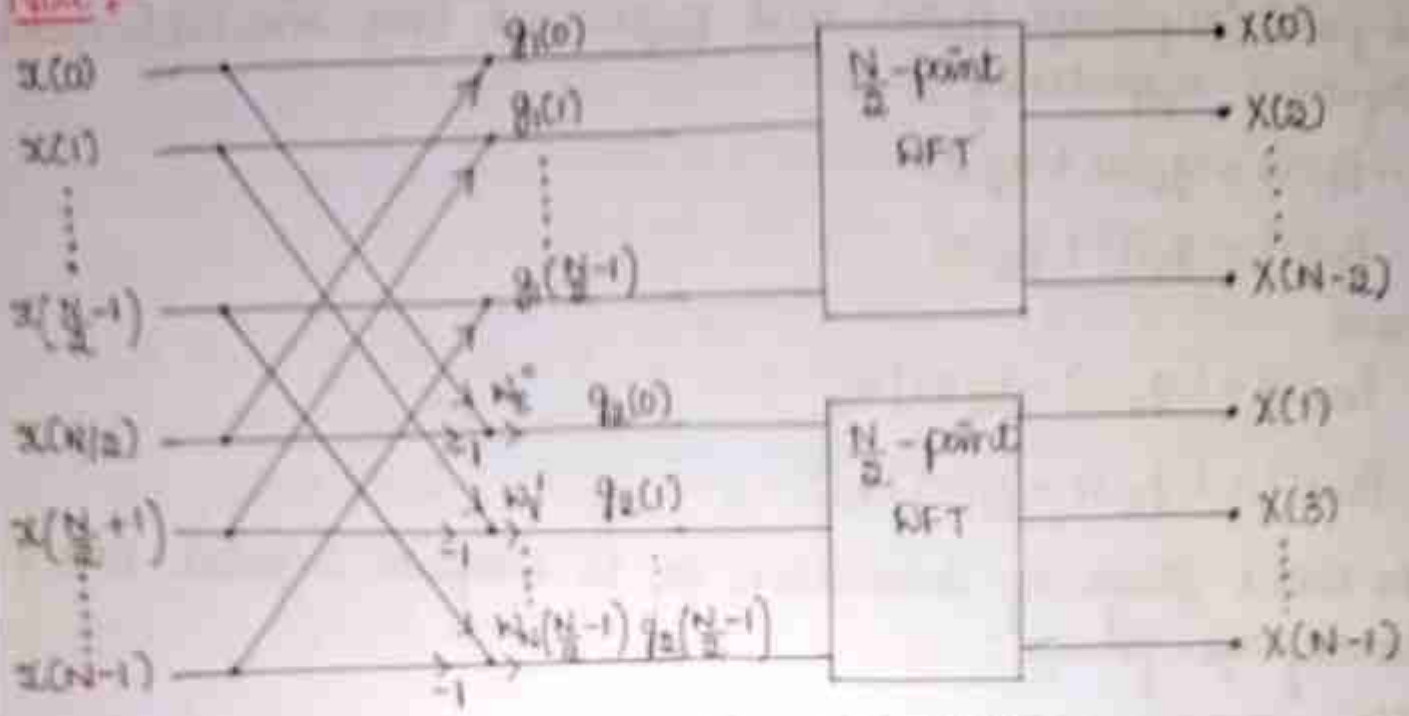


This block splits x(n) into $g_1(n)$ and $g_2(n)$ according to Eq(a) and Eq(b)

fig : First stage of decimation of DIF-FFT algorithm for N = 8

Scanned with CamScanner

Note :-



Generally, we can write the diagram as :

$$g_1(n) = x(n) + x\left(n + \frac{N}{2}\right) \quad ; \quad 0 \le n \le \frac{N}{2} - 1$$

$$g_2(n) = \left[x(n) - x\left(n + \frac{N}{2}\right)\right] W_N^n$$

b) Second stage of decimation :-

For $N = 8$, we know that :

$g_1(n) = \{g_1(0), g_1(1), g_1(2), g_1(3)\}$ and

$g_2(n) = \{g_2(0), g_2(1), g_2(2), g_2(3)\}$.

The 4-point sequence $g_1(n)$ can be splitted into 2-point sequences $P_{11}(n)$ and $P_{12}(n)$ of even and odd numbered sequences respectively.

$\therefore P_{11}(0) = g_1(0) + g_1(2)$

$P_{11}(1) = g_1(1) + g_1(3)$

and

$P_{12}(0) = [g_1(0) - g_1(2)] W_8^0$

$P_{12}(1) = [g_1(1) - g_1(3)] W_8^2$

Scanned with CamScanner

Similarly, the 4-point sequence $g_2(n)$ can be splitted into 2-point sequences $p_{21}(n)$ and $p_{22}(n)$ as Even and odd-numbered sequences respectively.

$\therefore p_{21}(0) = g_2(0) + g_2(2)$

$p_{21}(1) = g_2(1) + g_2(3)$

and

$p_{22}(0) = [g_2(0) - g_2(2)]\, W_8^0$

$p_{22}(1) = [g_2(1) - g_2(3)]\, W_8^2$

The third stage of decimation is to compute 2-point DFT's directly as follows:

$X(0) = p_{11}(0) + p_{11}(1)$

$X(4) = [p_{11}(0) - p_{11}(1)]\, W_8^0$

$X(2) = p_{12}(0) + p_{12}(1)$

$X(6) = [p_{12}(0) - p_{12}(1)]\, W_8^0$

$X(1) = p_{21}(0) + p_{21}(1)$

$X(5) = [p_{21}(0) - p_{21}(1)]\, W_8^0$

$X(3) = p_{22}(0) + p_{22}(1)$

$X(7) = [p_{22}(0) - p_{22}(1)]\, W_8^0$

2-point butterfly for DIF-FFT :-

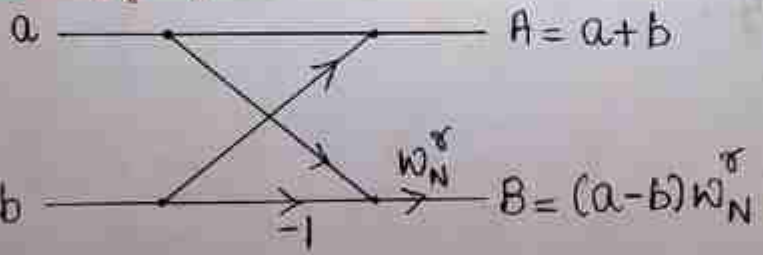


Fig : Basic butterfly for DIF-FFT

Note :- The Computational complexity, in-place Computation and the memory requirements in DIF-FFT is same as that of DIT-FFT.

Scanned with CamScanner

* For the decimation-in-frequency (DIF) algorithm, it is seen that:
1) The input is in normal order.
2) The output is in bit-reversed order.
3) Computations are performed in place.

* The DIF-FFT flowgraph is similar to the flowgraph of DIT-FFT, but is in opposite direction.

* The Computational Complexity for N-point radix-2 DIF-FFT algorithm:

Number of complex multiplications $= \frac{N}{2} \log_2 N$

Number of complex additions $= N \log_2 N$

Fig: Signal flow-graph and stages of Computation of Radix-2 DIF-FFT algorithm for N = 8

Comparison between Radix-2 DIT-FFT and DIF-FFT algorithms:

Radix-2 DIT-FFT algorithm	Radix-2 DIF-FFT algorithm
1) The time-domain sequence $x(n)$ is decimated.	1) The frequency-domain sequence $X(K)$ is decimated.
2) The input is in bit-reversed order whereas the output is in normal order.	2) The input is in normal order whereas the output is in bit reversed order.
3) In each stage of computation, the phase factors W_N are multiplied before addition.	3) In each stage of computation, the phase factors W_N are multiplied after addition.
4) If $N = 2^v$, then the algorithm consists of 'v' stages of computations.	4) If $N = 2^v$, then the algorithm consists of 'v' stages of computations.
5) Total number of arithmetic operations required are, $\frac{N}{2}\log_2 N$ Complex multiplications and $N\log_2 N$ complex additions.	5) Total number of arithmetic operations required are, $\frac{N}{2}\log_2 N$ Complex multiplications and $N\log_2 N$ Complex additions.

Scanned with CamScanner

Problems on Efficient Computations of the DFT :-

1) How many Complex multiplications and additions are required for computing 256-point DFT using direct Computation of DFT and using FFT Computation. which one is evaluate DFT at a faster rate?

Sol^n :- Given $N = 256$

Using direct Computation :-

a) Complex multiplications $= N^2 = (256)^2 = 65,536$

b) Complex additions $= N(N-1) = 256(256-1) = 65,280$

Using FFT algorithm :-

a) Complex multiplications $= \frac{N}{2} \log_2 N = \frac{256}{2} \log_2 256 = 1024$

b) Complex additions $= N \log_2 N = 256 \log_2 256 = 2048$

Speed improvement factor :-

$$\text{Speed Improvement factor} = \frac{N^2}{\frac{N}{2} \log_2 N} = \frac{(256)^2}{\frac{256}{2} \log_2 256} = 64 \text{ times}$$

FFT is 64-times faster than direct Computation in evaluation of DFT for $N = 256$.

2) What is the total number of Complex additions and multiplications required for $N = 512$ points, If DFT is Computed directly and if FFT is used and also find number of stages required and its memory requirements

Sol^n :- Given $N = 512$

Using direct Computation :-

a) Complex additions $= N(N-1) = 512(512-1) = 2,61,632$

b) Complex multiplications $= N^2 = (512)^2 = 2,62,144$

Scanned with CamScanner

Using FFT algorithm :-

a) Complex multiplications $= \frac{N}{2}\log_2 N = \frac{512}{2}\log_2 512 = 2304$

b) Complex additions $= N\log_2 N = 512\log_2 512 = 4608$

Number of Stages required, $r = \log_2 N = \log_2 512 = 9$ Stages

Memory requirements :-

Real registers + Complex registers $= 2N + \frac{N}{2} = 2\times512 + \frac{512}{2} = 1280$

∴ 1280 registers are used.

3) How many multiplications and additions are needed for 64-point sequence in calculations of DFT using FFT algorithm and using direct DFT computation. Specify number of real registers needed to perform these computation. Also find real multiplications and real additions are involved in DFT computations

Sol^n :- Given $N = 64$

Using direct Computation :-

a) Complex multiplications $= N^2 = (64)^2 = 4096$

b) Complex additions $= N(N-1) = 64(64-1) = 4032$

Using FFT algorithm :-

a) Complex multiplications $= \frac{N}{2}\log_2 N = \frac{64}{2}\log_2 64 = 192$

b) Complex additions $= N\log_2 N = 64\log_2 64 = 384$

Memory requirements :-

Real registers + Complex registers $= 2N + \frac{N}{2} = 2\times64 + \frac{64}{2}$

$= 160$ registers

Scanned with CamScanner

Real multiplications and Real additions :—

a) Real multiplications $= 4N^2 = 4(64)^2 = 16384$

b) Real additions $= 4N^2 - 2N = 4(64)^2 - 2(64) = 16256$

4) Determine the following for a 128-point FFT computation
number of : a) Stages b) Butterflies in Each Stage c) Butterflies
needed for Entire Computation d) total number of Complex
multiplications e) total number of Complex additions f) Speed
improvement factor Compared with direct Computation.

Sol^n :-

a) Number of Stages, $x = \log_2 N = \log_2 128 = 7$ stages

b) Butterflies in Each Stage $= \frac{N}{2} = \frac{128}{2} = 64$ butterflies

c) Butterflies needed for Entire Computation $= \frac{N}{2} \log_2 N$

$= \frac{128}{2} \log_2 128 = 448$ Butterflies

d) Total number of Complex multiplications $= \frac{N}{2} \log_2 N$

$= \frac{128}{2} \log_2 128 = 448$

e) Total number of Complex additions $= N \log_2 N = 128 \log_2 128$

$= 896$

f) Speed improvement factor $= \frac{N^2}{\frac{N}{2} \log_2 N} = \frac{(128)^2}{\frac{128}{2} \log_2 128} = 36.57$ times

FFT is 36.57 times faster than direct Computation of DFT for $N = 128$.

Scanned with CamScanner

Module-3

Arushi M.
Asst. Professor
Dept. of ECE,
BGSIT, B.G.

Radix-2 FFT Algorithms and Linear Filtering

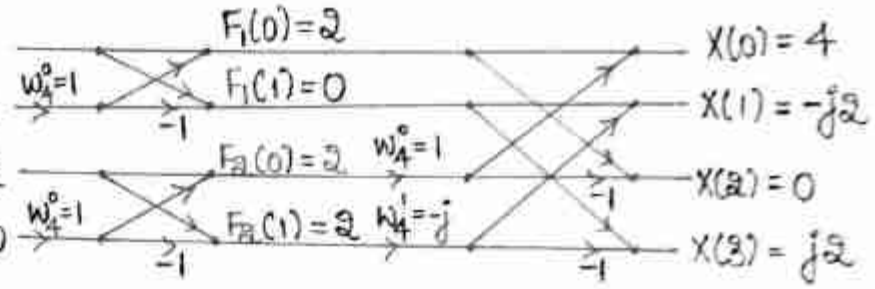
Problems on DIT-FFT :-

1) For the given sequence, $x(n)=\{1,2,1,0\}$. Find DFT using DIT-FFT algorithm.

Soln :-

$W_4^0 = 1 \; ; \; W_4^1 = -j$

Bit Reversal		
00	00	$x(0)=1$
01	10	$x(2)=1$
10	01	$x(1)=2$
11	11	$x(3)=0$



$\langle X(K) = \{4, -j2, 0, j2\} \rangle$

1st Stage o/p's :-

$F_1(0) = x(0) + W_4^0 x(2) = 1 + 1(1) = \underline{2}$

$F_1(1) = x(0) - W_4^0 x(2) = 1 - 1(1) = \underline{0}$

$F_2(0) = x(1) + W_4^0 x(3) = 2 + 1(0) = \underline{2}$

$F_2(1) = x(1) - W_4^0 x(3) = 2 - 1(0) = \underline{2}$

2nd Stage o/p's :-

$X(0) = F_1(0) + W_4^0 F_2(0) = 2 + 1(2) = \underline{4}$

$X(1) = F_1(1) + W_4^1 F_2(1) = 0 + (-j)2 = \underline{-j2}$

$X(2) = F_1(0) - W_4^0 F_2(0) = 2 - 1(2) = \underline{0}$

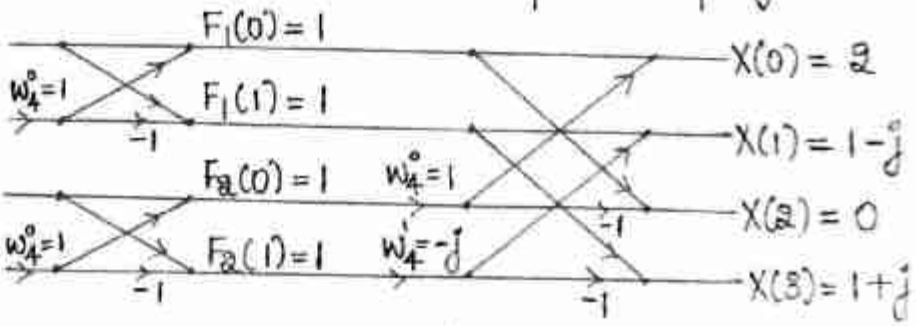
$X(3) = F_1(1) - W_4^1 F_2(1) = 0 - (-j)2 = \underline{j2}$

2) Find the DFT for the sequence $x(n)=\{1,1,0,0\}$ using DIT-FFT.

Soln :-

$W_4^0 = 1 \; ; \; W_4^1 = -j$

Bit Reversal		
00	00	$x(0)=1$
01	10	$x(2)=0$
10	01	$x(1)=1$
11	11	$x(3)=0$

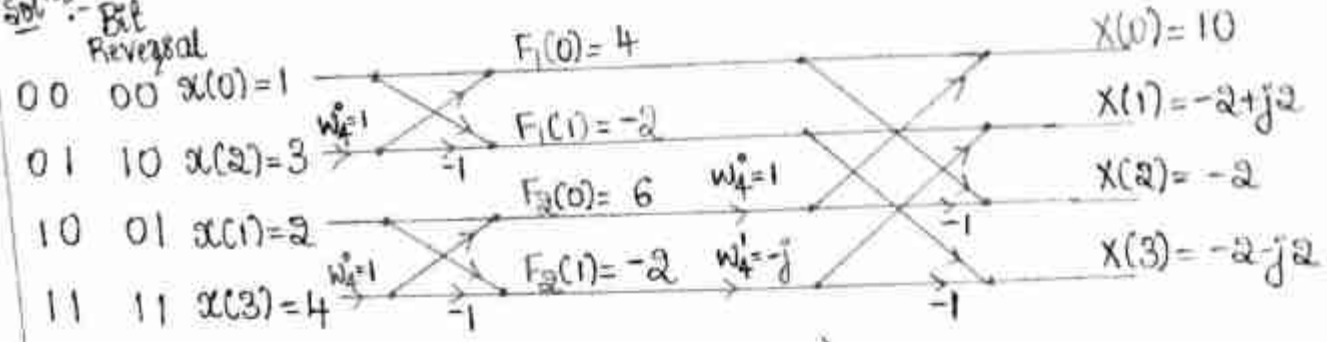


$\langle \therefore X(K) = \{2, 1-j, 0, 1+j\} \rangle$

$x(n) = \{1, 2, 3, 4\}$

$W_4^0 = 1 \; ; W_4^1 = -j$

Soln:- Bit Reversal



$\therefore X(K) = \{10, -2+j2, -2, -2-j2\}$

4) $x(n) = (-1)^n \; ; \; 0 \le n \le 3$

Soln:-

@ $n=0$, $x(0) = (-1)^0 = 1$

@ $n=1$, $x(1) = (-1)^1 = -1$

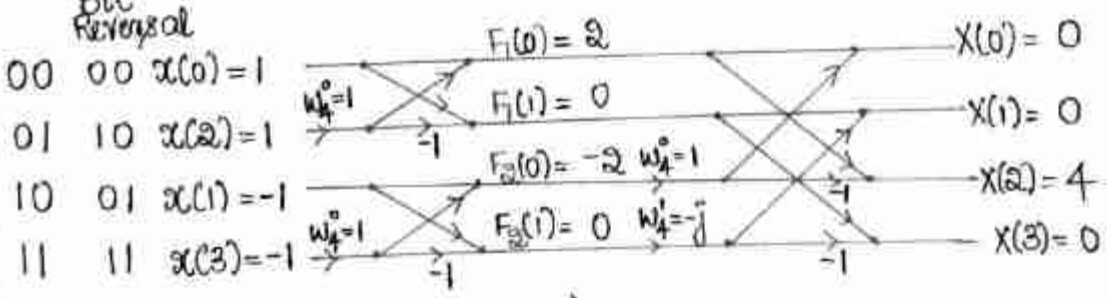
@ $n=2$, $x(2) = (-1)^2 = 1$

@ $n=3$, $x(3) = (-1)^3 = -1$

$\therefore x(n) = \{1, -1, 1, -1\}$

$W_4^0 = 1 \; ; W_4^1 = -j$

Bit Reversal



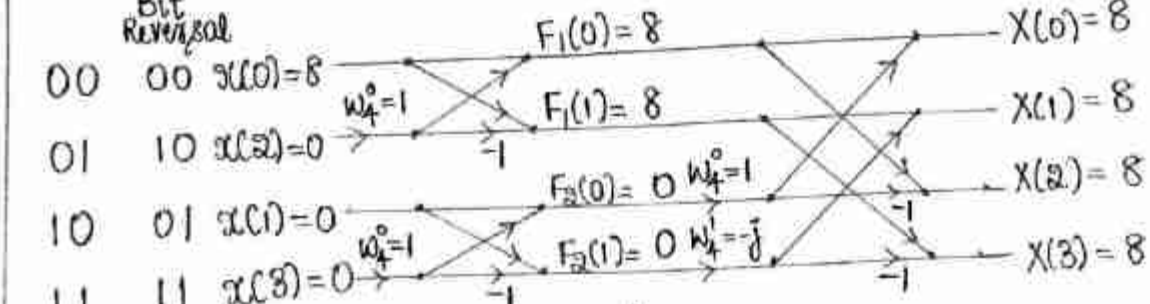
$\therefore X(K) = \{0, 0, 4, 0\}$

5) Find the 4-point DFT for the sequence $x(n) = \{8\}$ using DIT-FFT algorithm.

Soln:- Since $N=4$, $\therefore x(n) = \{8, 0, 0, 0\}$

$W_4^0 = 1 \; ; W_4^1 = -j$

Bit Reversal



$\therefore X(K) = \{8, 8, 8, 8\}$

6) Find the DFT of the sequence: $x(n) = \cos\left(\frac{n\pi}{2}\right)$; $0 \le n \le 3$ using DIT-FFT algorithm.

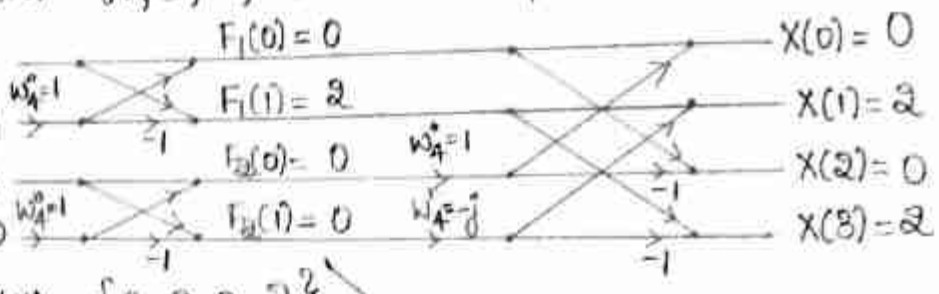
Sol^n:- @$n=0$, $x(0) = \cos\left(\frac{\pi}{2} \times 0\right) = 1$ @$n=2$, $x(2) = \cos\left(\frac{\pi}{2} \times 2\right) = -1$

@$n=1$, $x(1) = \cos\left(\frac{\pi}{2} \times 1\right) = 0$ @$n=3$, $x(3) = \cos\left(\frac{\pi}{2} \times 3\right) = 0$

$\therefore x(n) = \{1, 0, -1, 0\}$

$W_4^0 = 1$; $W_4^1 = -j$

Bit Reversal	
00	00 $x(0)=1$
01	10 $x(2)=-1$
10	01 $x(1)=0$
11	11 $x(3)=0$



$\therefore X(K) = \{0, 2, 0, 2\}$

7) For the given sequence $x(t) = 2t$, find 4-point DFT using DIT-FFT algorithm. If $x(t)$ is sampled at 10 samples/sec

Sol^n:- Given $x(t) = 2t$ and $f_s = 10$ samples/sec

$T_s = \frac{1}{f_s} = \frac{1}{10} = 0.1$ sec

$x(n) = x(t)\big|_{t = nT_s}$

$\therefore x(n) = 2 \times nT_s$

$x(n) = 2 \times 0.1 \times n = 0.2n$

$x(n) = \{0, 0.2, 0.4, 0.6\}$

@$n=0$, $x(0) = 0.2 \times 0 = 0$

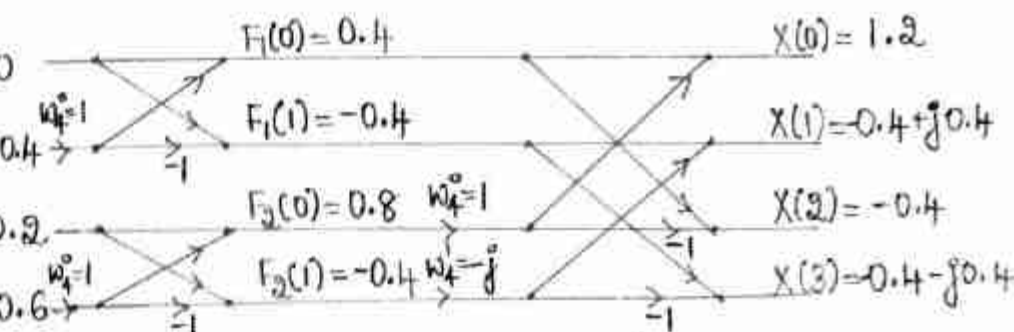
@$n=1$, $x(1) = 0.2 \times 1 = 0.2$

@$n=2$, $x(2) = 0.2 \times 2 = 0.4$

@$n=3$, $x(3) = 0.2 \times 3 = 0.6$

$W_4^0 = 1$; $W_4^1 = -j$

Bit Reversal	
00	00 $x(0)=0$
01	10 $x(2)=0.4$
10	01 $x(1)=0.2$
11	11 $x(3)=0.6$

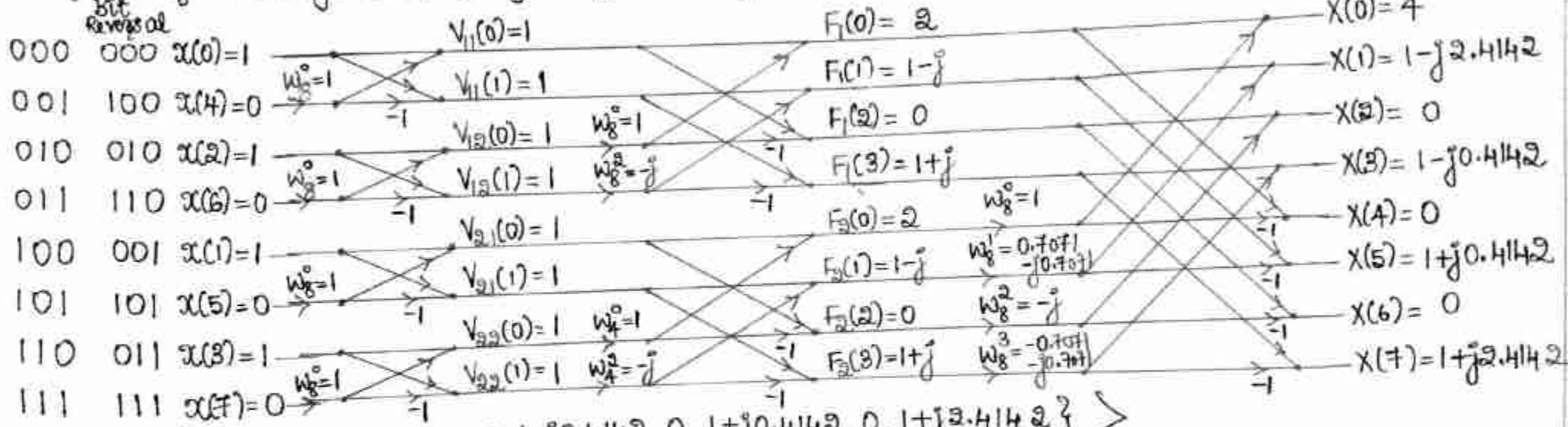


$\therefore X(K) = \{1.2, -0.4 + j0.4, -0.4, -0.4 - j0.4\}$

3) For the given sequence, $x(n) = \{1, 1, 1, 1\}$, find 8-point DFT using DIT-FFT algorithm. Indicate stage outputs.

Sol^n :- Since $N = 8$, $x(n) = \{1, 1, 1, 1, 0, 0, 0, 0\}$

$W_8^0 = 1$; $W_8^1 = 0.7071 - j0.7071$; $W_8^2 = -j$; $W_8^3 = -0.7071 - j0.7071$



$X(K) = \{4, 1-j2.4142, 0, 1-j0.4142, 0, 1+j0.4142, 0, 1+j2.4142\}$

1st stage o/p's :-

$V_{11}(0) = x(0) + W_8^0 x(4) = 1$
$V_{11}(1) = x(0) - W_8^0 x(4) = 1$
$V_{12}(0) = x(2) + W_8^0 x(6) = 1$
$V_{12}(1) = x(2) - W_8^0 x(6) = 1$
$V_{21}(0) = x(1) + W_8^0 x(5) = 1$
$V_{21}(1) = x(1) - W_8^0 x(5) = 1$
$V_{22}(0) = x(3) + W_8^0 x(7) = 1$
$V_{22}(1) = x(3) - W_8^0 x(7) = 1$

2nd stage o/p's :-

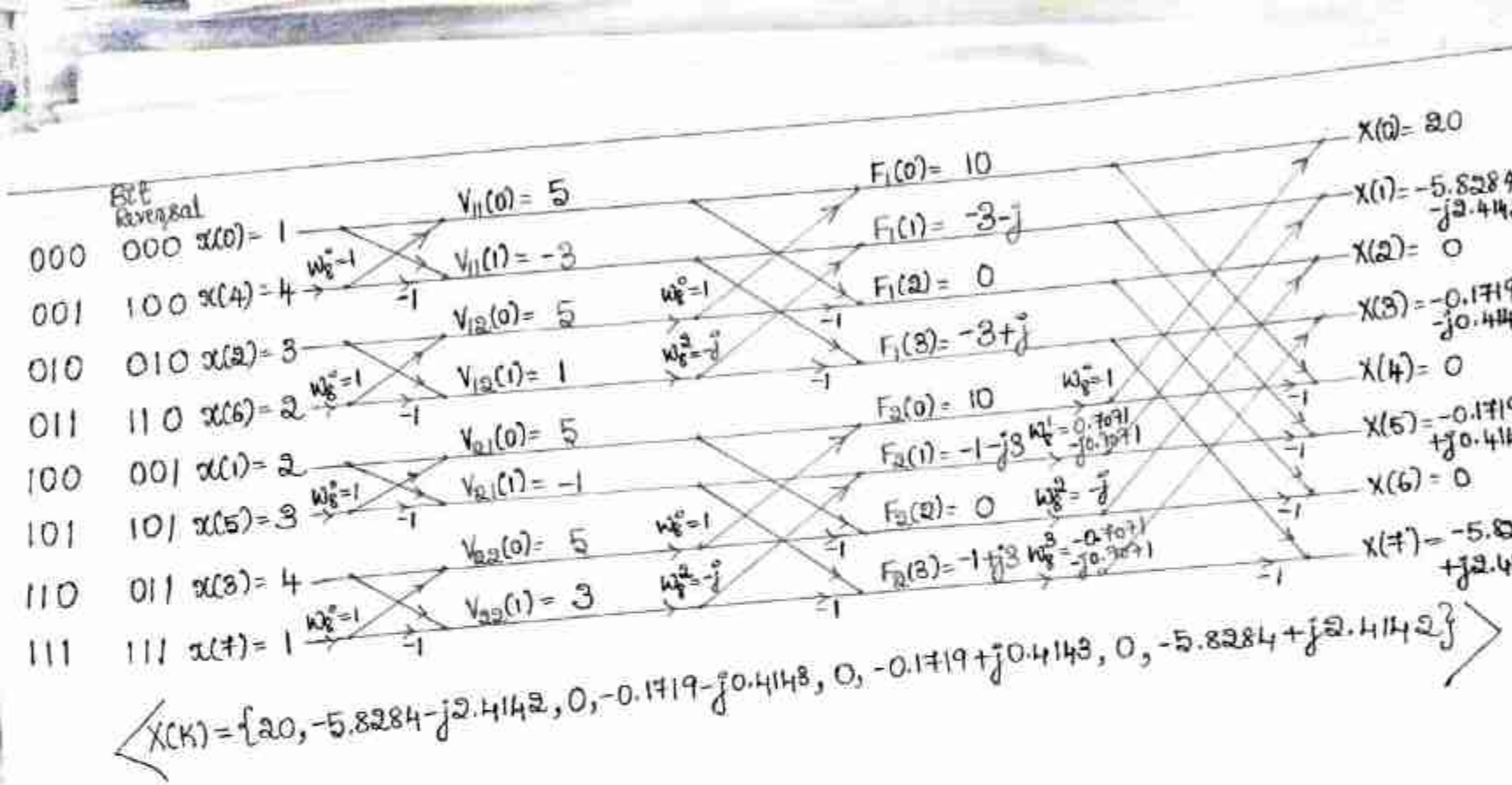
$F_1(0) = V_{11}(0) + W_8^0 V_{12}(0) = 2$
$F_1(1) = V_{11}(1) + W_8^2 V_{12}(1) = 1 - j$
$F_1(2) = V_{11}(0) - W_8^0 V_{12}(0) = 0$
$F_1(3) = V_{11}(1) - W_8^2 V_{12}(1) = 1 + j$
$F_2(0) = V_{21}(0) + W_8^0 V_{22}(0) = 2$
$F_2(1) = V_{21}(1) + W_8^2 V_{22}(1) = 1 - j$
$F_2(2) = V_{21}(0) - W_8^0 V_{22}(0) = 0$
$F_2(3) = V_{21}(1) - W_8^2 V_{22}(1) = 1 + j$

Final stage o/p's :-

$X(0) = F_1(0) + W_8^0 F_2(0) = 4$
$X(1) = F_1(1) + W_8^1 F_2(1) = 1 - j2.4142$
$X(2) = F_1(2) + W_8^2 F_2(2) = 0$
$X(3) = F_1(3) + W_8^3 F_2(3) = 1 - j0.4142$
$X(4) = F_1(0) - W_8^0 F_2(0) = 0$
$X(5) = F_1(1) - W_8^1 F_2(1) = 1 + j0.4142$
$X(6) = F_1(2) - W_8^2 F_2(2) = 0$
$X(7) = F_1(3) - W_8^3 F_2(3) = 1 + j2.4142$

3) Find the 8-point DFT for the sequence $x(n)=\{1,2,3,4,4,3,2,1\}$ using DIT-FFT algorithm.

Soln: $W_8^0=1$; $W_8^1=0.7071-j0.7071$; $W_8^2=-j$; $W_8^3=-0.7071-j0.7071$

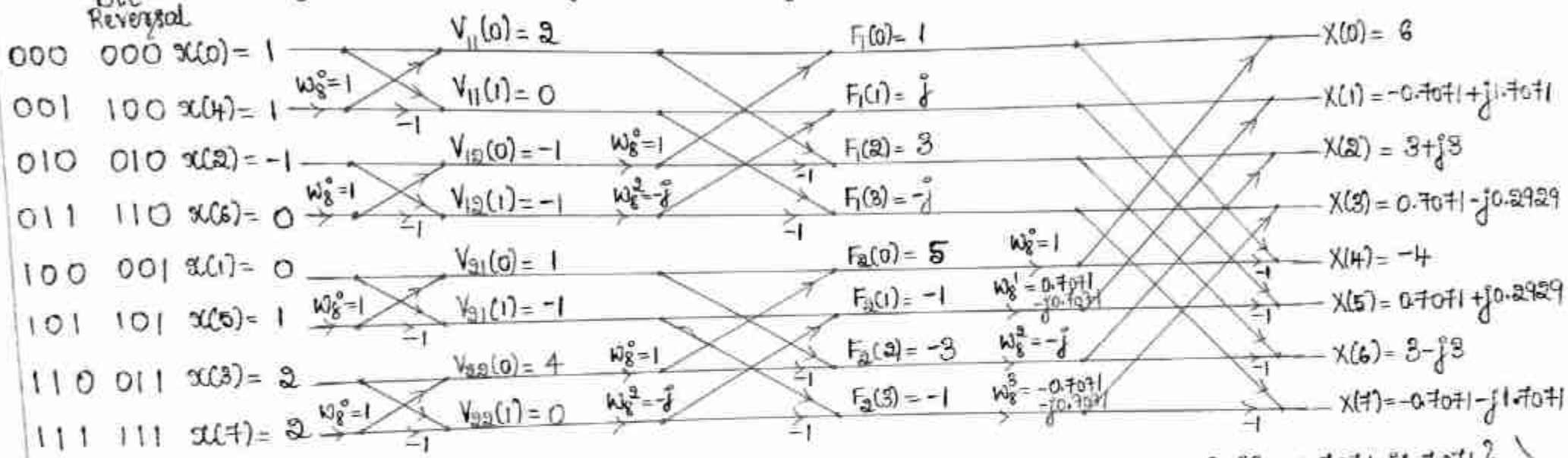


$X(K)=\{20,-5.8284-j2.4142,0,-0.1719-j0.4148,0,-0.1719+j0.4143,0,-5.8284+j2.4142\}$

Compute 8-point DFT for the sequence $x(n) = \{1, 0, -1, 2, 1, 1, 0, 2\}$ using DIT-FFT algorithm.

Sol^n:-

$W_8^0 = 1$; $W_8^1 = 0.7071 - j0.7071$; $W_8^2 = -j$; $W_8^3 = -0.7071 - j0.7071$



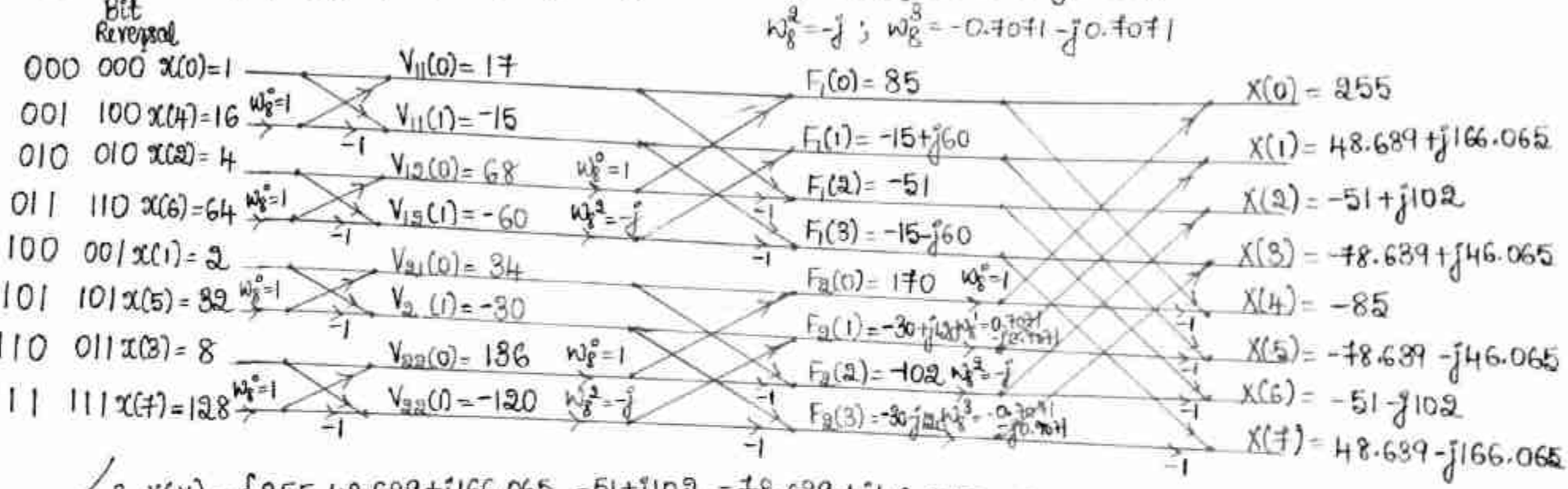
$X(K) = \{6, -0.7071 + j1.7071, 3 + j3, 0.7071 - j0.2929, -4, 0.7071 + j0.2929, 3 - j3, -0.7071 - j1.7071\}$

11) $x(n) = (2)^n$, find 8-point DFT using DIT-FFT algorithm.

Soln:- $x(n) = \{1, 2, 4, 8, 16, 32, 64, 128\}$

$W_8^0 = 1$; $W_8^1 = 0.7071 - j0.7071$

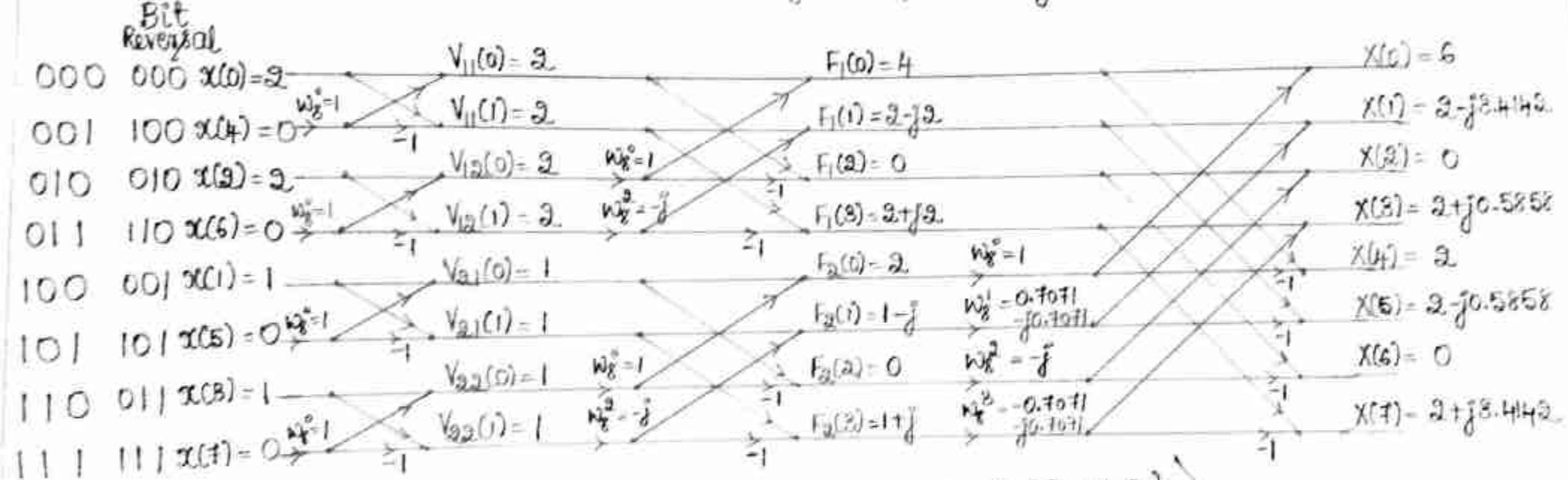
$W_8^2 = -j$; $W_8^3 = -0.7071 - j0.7071$



$\therefore X(K) = \{255, 48.639+j166.065, -51+j102, -78.639+j46.065, -85, -78.639-j46.065, -51-j102, 48.639-j166.065\}$

12) Obtain 8-point FFT of the sequence $x(n) = \{2, 1, 2, 1\}$ using Radix-2 DIT-FFT algorithm.

Sol^n:- $x(n) = \{2, 1, 2, 1, 0, 0, 0, 0\}$ $\quad W_8^0 = 1 ;\ W_8^1 = 0.7071 - j0.7071 ;\ W_8^2 = -j ;\ W_8^3 = -0.7071 - j0.7071$

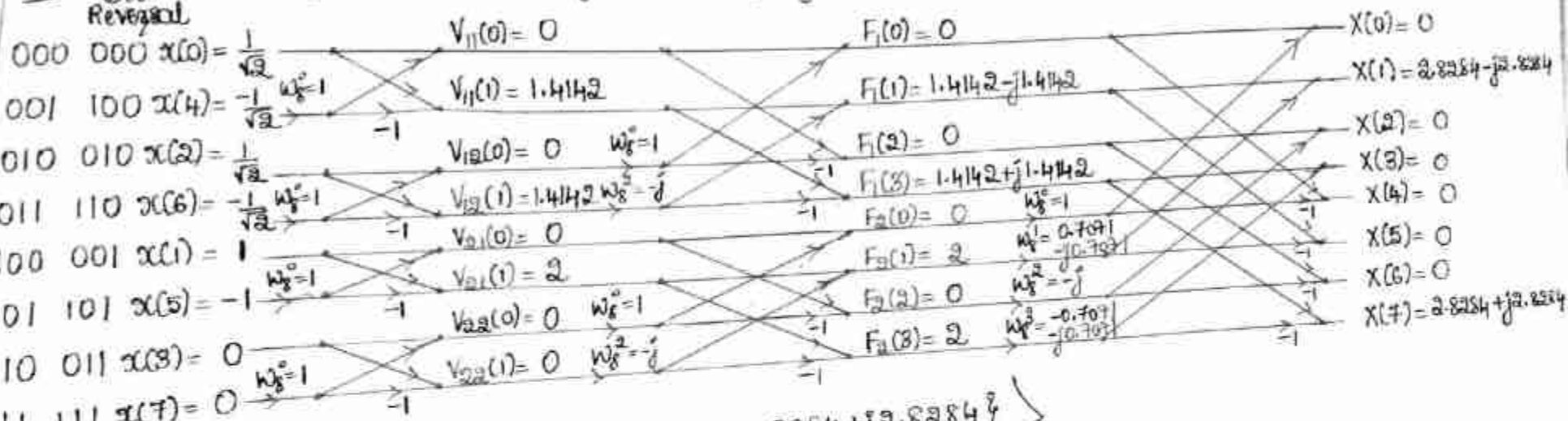


$\therefore X(K) = \{6, 2-j2.4142, 0, 2+j0.5858, 2, 2-j0.5858, 0, 2+j2.4142\}$

(or) 13) Let $x(n)$ be the following sequence: $x(n) = \{\frac{1}{\sqrt{2}}, 1, \frac{1}{\sqrt{2}}, 0, \frac{-1}{\sqrt{2}}, -1, \frac{-1}{\sqrt{2}}, 0\}$. Use DIT-FFT algorithm to compute DFT of the above sequence.

Soln:

$W_8^0 = 1$; $W_8^1 = 0.7071 - j0.7071$; $W_8^2 = -j$; $W_8^3 = -0.7071 - j0.7071$



$\therefore X(K) = \{0, 2.8284 - j2.8284, 0, 0, 0, 0, 0, 2.8284 + j2.8284\}$

14) Determine DFT of the following 8-point sequence, $x(n) = \left(\frac{1}{4}\right)^n$; $0 \le n \le 7$ using DIT-FFT algorithm. Show all the intermediate results.

Soln:- $x(n) = \left\{1, \frac{1}{4}, \frac{1}{16}, \frac{1}{64}, \frac{1}{256}, \frac{1}{1024}, \frac{1}{4096}, \frac{1}{16384}\right\}$

$W_8^0 = 1$; $W_8^1 = 0.7071 - j0.7071$

$W_8^2 = -j$; $W_8^3 = -0.7071 - j0.7071$

Bit Reversal

000 000 $x(0) = 1$

001 100 $x(4) = \frac{1}{256}$

010 010 $x(2) = \frac{1}{16}$

011 110 $x(6) = \frac{1}{4096}$

100 001 $x(1) = \frac{1}{4}$

101 101 $x(5) = \frac{1}{1024}$

110 011 $x(3) = \frac{1}{64}$

111 111 $x(7) = \frac{1}{16384}$

$V_{11}(0) = \frac{257}{256}$; $V_{11}(1) = \frac{255}{256}$

$V_{12}(0) = \frac{257}{4096}$; $V_{12}(1) = \frac{255}{4096}$

$V_{21}(0) = \frac{257}{1024}$; $V_{22}(1) = \frac{255}{1024}$

$V_{22}(0) = \frac{257}{16384}$; $V_{22}(1) = \frac{255}{16384}$

$F_1(0) = \frac{4369}{4096}$

$F_1(1) = \frac{255}{256} - j\frac{255}{4096}$

$F_1(2) = \frac{8855}{4096}$

$F_1(3) = \frac{255}{256} + j\frac{255}{4096}$

$F_2(0) = \frac{4369}{16384}$

$F_2(1) = \frac{255}{1024} - j\frac{255}{16384}$

$F_2(2) = \frac{8855}{16384}$

$F_2(3) = \frac{255}{1024} + j\frac{255}{16384}$

$X(0) = 1.333$

$X(1) = 1.161 - j0.249$

$X(2) = 0.941 - j0.235$

$X(3) = 0.831 - j0.12$

$X(4) = 0.8$

$X(5) = 0.831 + j0.12$

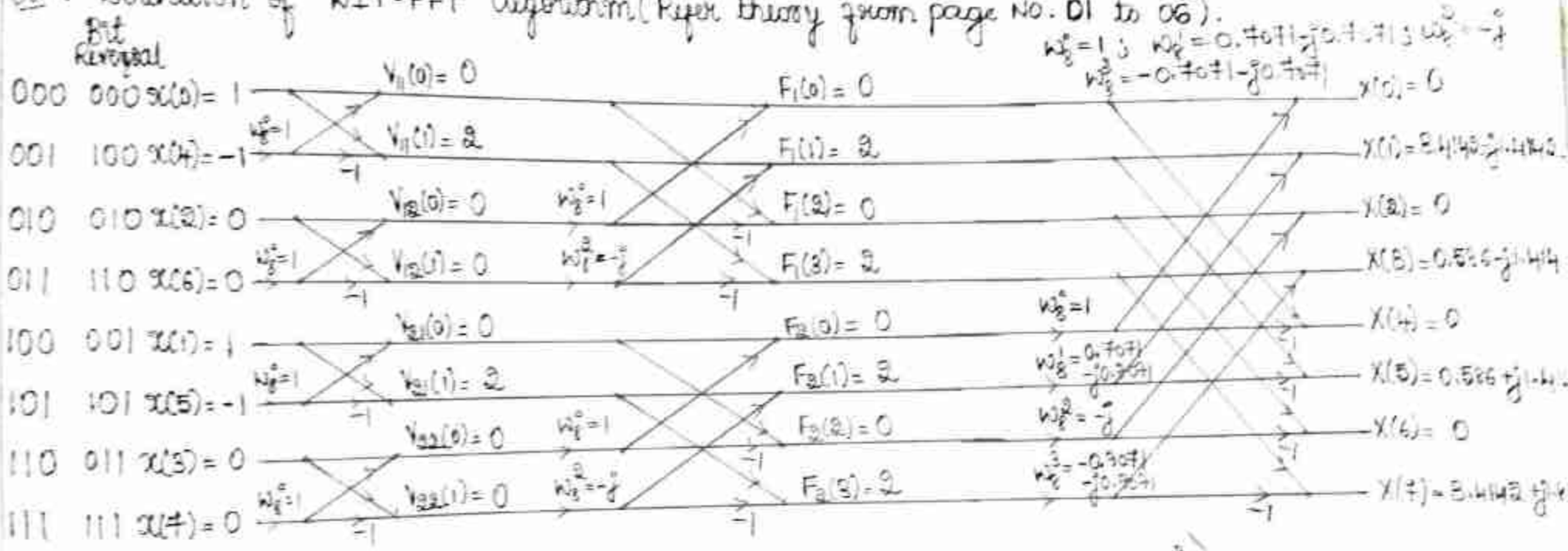
$X(6) = 0.941 + j0.235$

$X(7) = 1.161 + j0.24$

$X(K) = \{1.333, 1.161 - j0.249, 0.941 - j0.235, 0.831 - j0.124, 0.8, 0.831 + j0.124, 0.941 + j0.235, 1.161 + j0.249\}$

5) Derive the DIT-FFT algorithm for N=8 and using the resulting signal flowgraph, compute the 8-point DFT of an 8-point sequence: $x(n) = \{1, 1, 0, 0, -1, -1, 0, 0\}$

Sol^n:- Derivation of DIT-FFT algorithm (Paper theory from page No. 01 to 06).

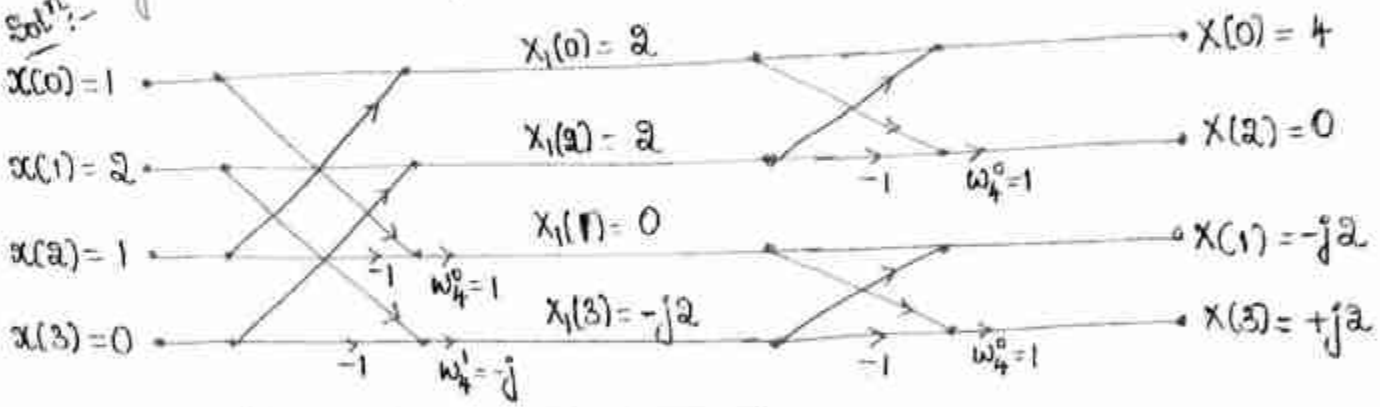


$\therefore X(K) = \{0, 3.4142 - j1.4142, 0, 0.586 - j1.414, 0, 0.586 + j1.414, 0, 3.4142 + j1.4142\}$

Problems on DIF-FFT :-

16) For the given sequence $x(n) = \{1, 2, 1, 0\}$. find 4-point DFT using DIF-FFT algorithm.

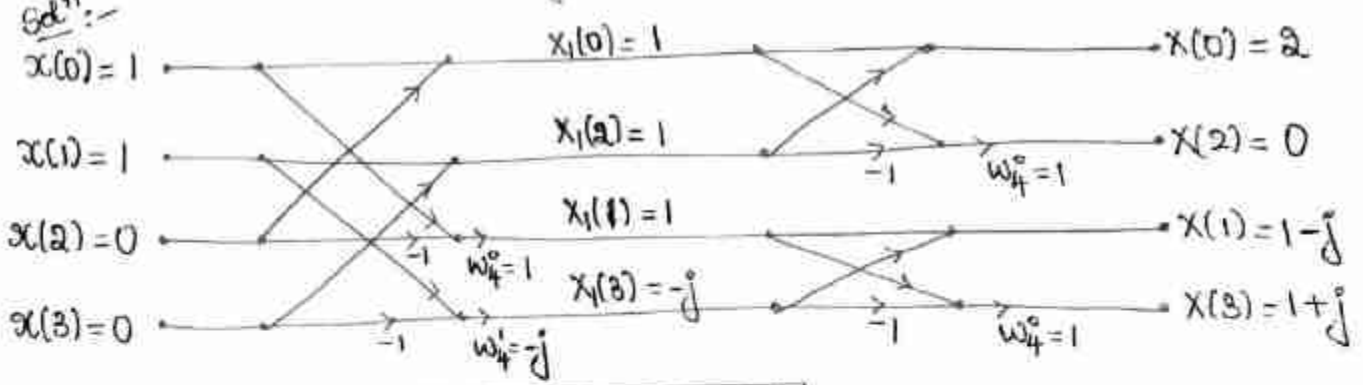
Sol^n:-



$\therefore \boxed{X(K) = \{4, -j2, 0, j2\}}$

17) Find 4-point DFT for the sequence $x(n) = \{1, 1, 0, 0\}$ using radix-2 DIF-FFT algorithm.

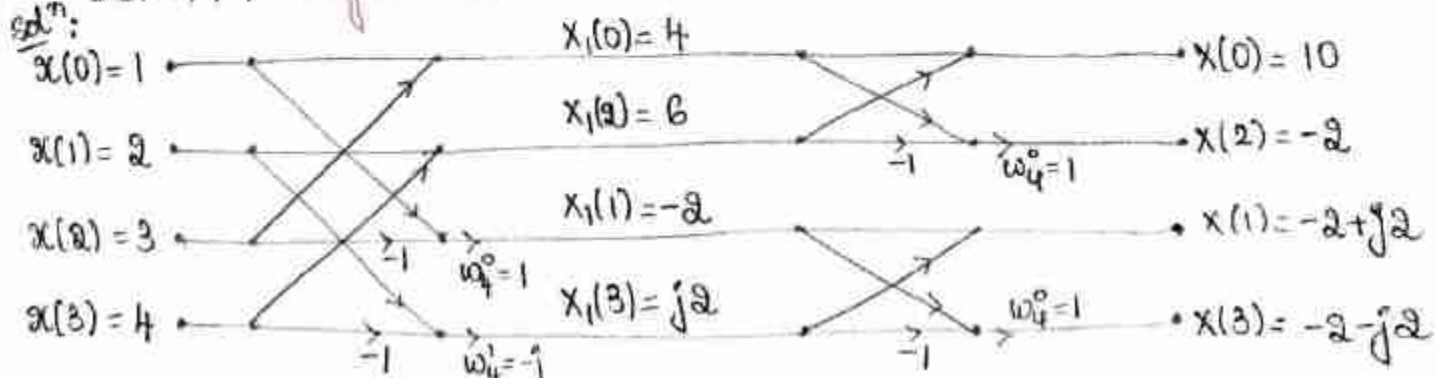
Sol^n:-



$\therefore \boxed{X(K) = \{2, 1-j, 0, 1+j\}}$

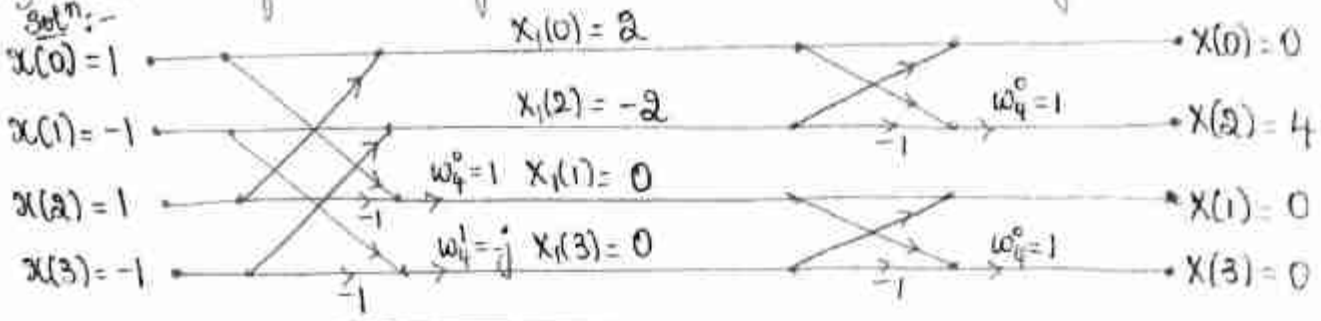
18) Find 4-point DFT for the sequence $x(n) = \{1, 2, 3, 4\}$ using DIF-FFT algorithm.

Sol^n:



19) Find DFT for the sequence $x(n)=\{1,-1,1,-1\}$ using DIF-FFT

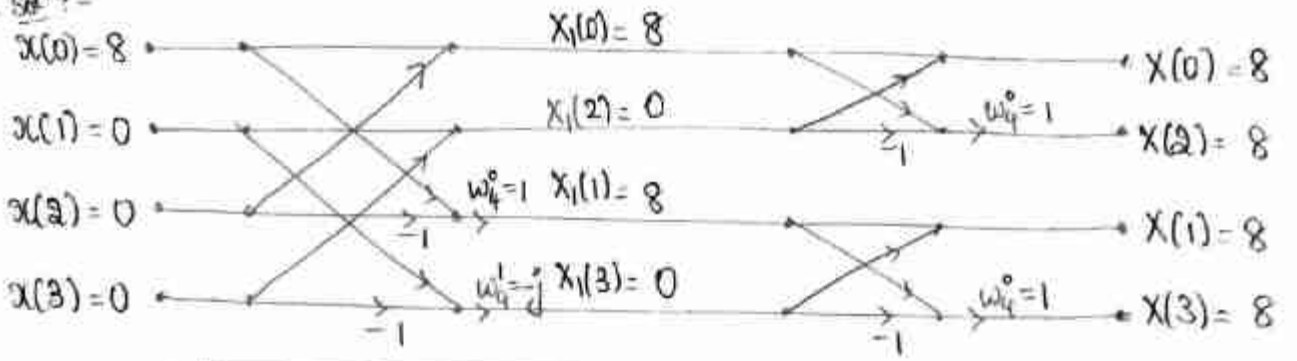
Soln:-



$\therefore \boxed{X(K)=\{0,0,4,0\}}$

20) Find 4-point DFT for the sequence $x(n)=\{8,0,0,0\}$ using DIF-FFT.

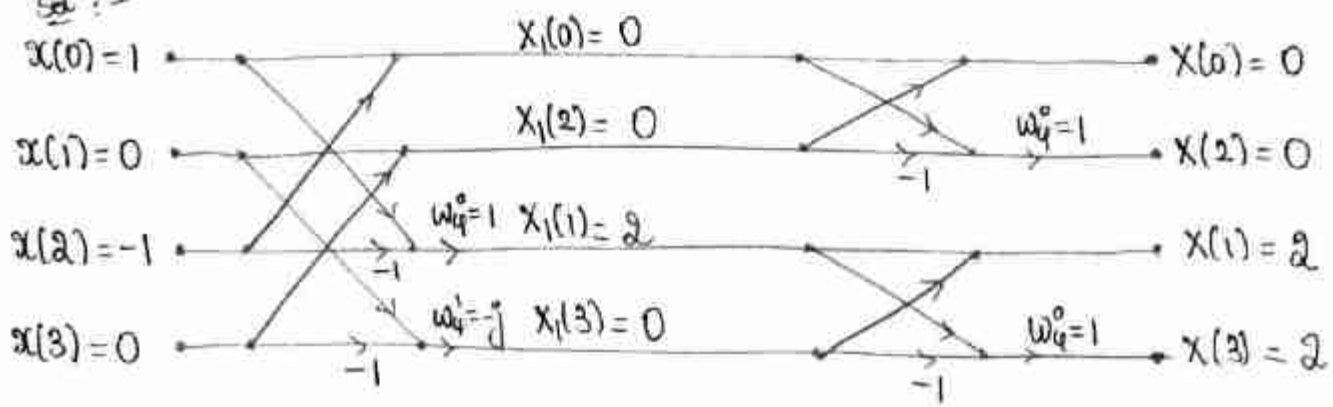
Soln:-



$\therefore \boxed{X(K)=\{8,8,8,8\}}$

21) Find 4-point DFT for the sequence $x(n)=\{1,0,-1,0\}$ using DIF-FFT

Soln:-

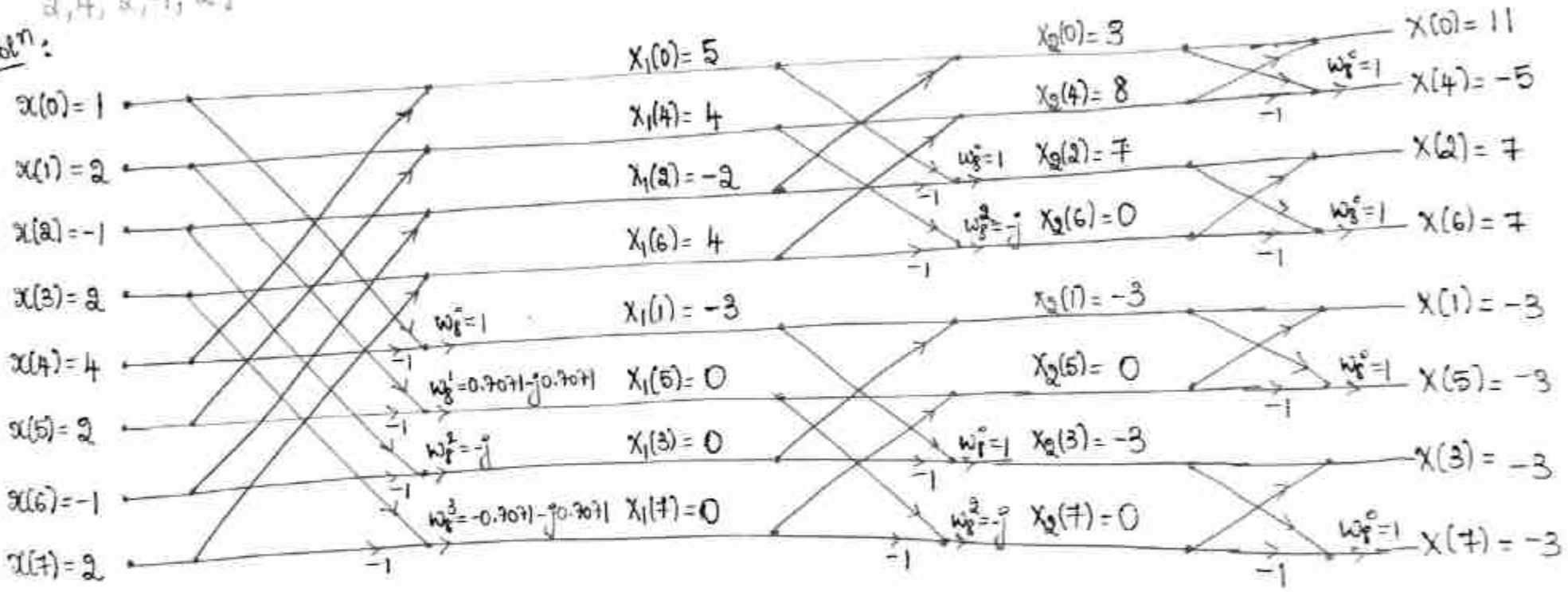


$\therefore \boxed{X(K)=\{0,2,0,2\}}$

(7)

a) Using DIF-FFT algorithm, Compute 8-point DFT for the sequence $x(n) = \{1, 2, -1, 2, 3, 4, 3, -1, 2\}$

Soln:

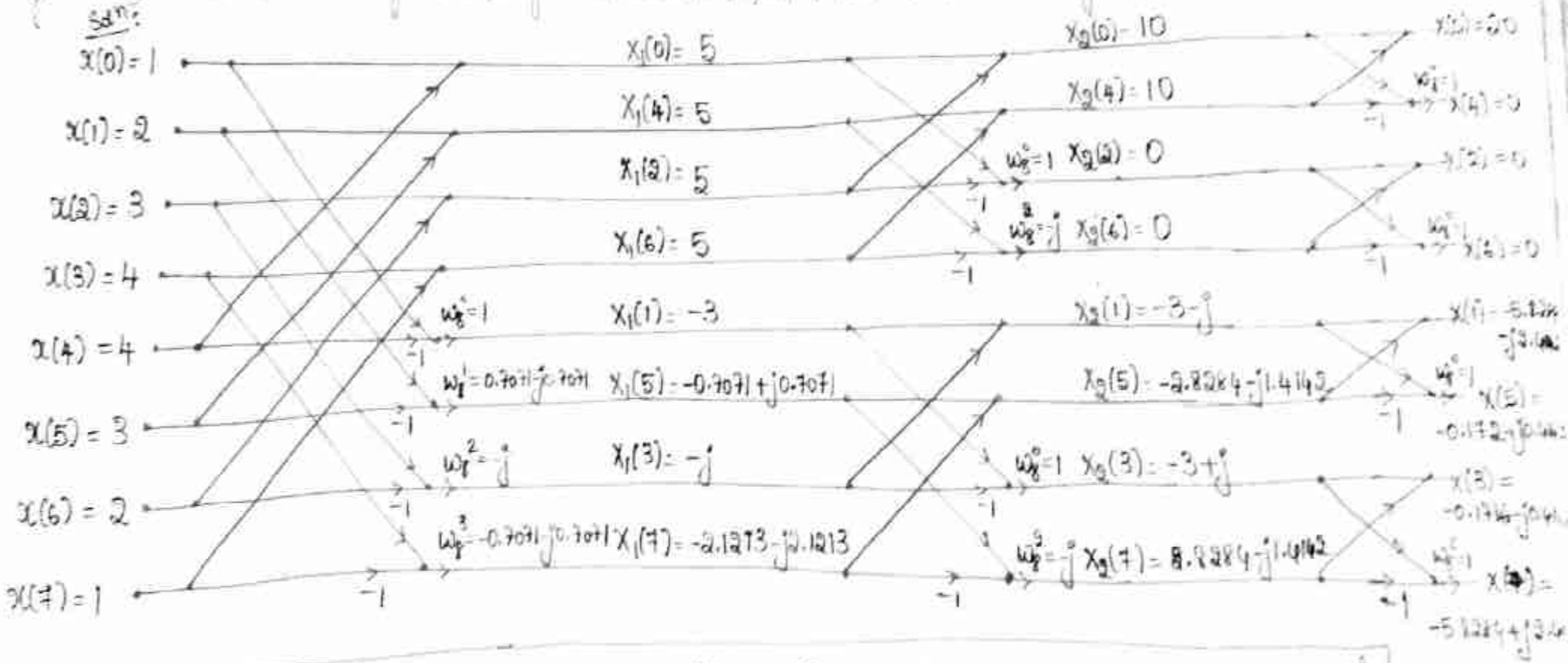


$\therefore X(K) = \{11, -3, 7, -3, -5, -3, 7, -3\}$

(8)

(3) Find the DFT of the sequence $x(n)=\{1,2,3,4,4,3,2,1\}$ using DIT-FFT algorithm.

Soln:

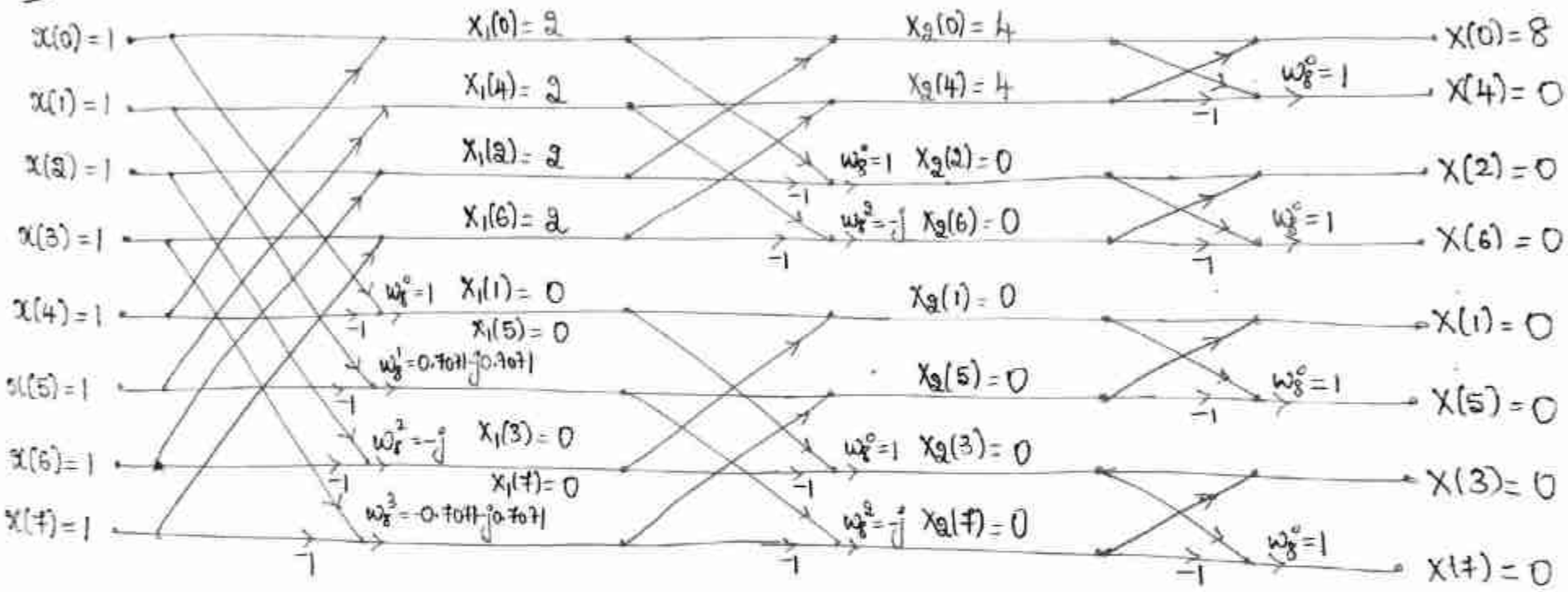


$$\therefore \boxed{X(K) = \{20, -5.8284 - j2.4142, 0, -0.1716 - j0.4142, 0, -0.172 + j0.4142, 0, -5.8284 + j2.4142\}}$$

(11)

3) Compute 8-point DFT for the sequence $x(n) = \{1, 1, 1, 1, 1, 1, 1, 1\}$ using DIF-FFT

Soln:

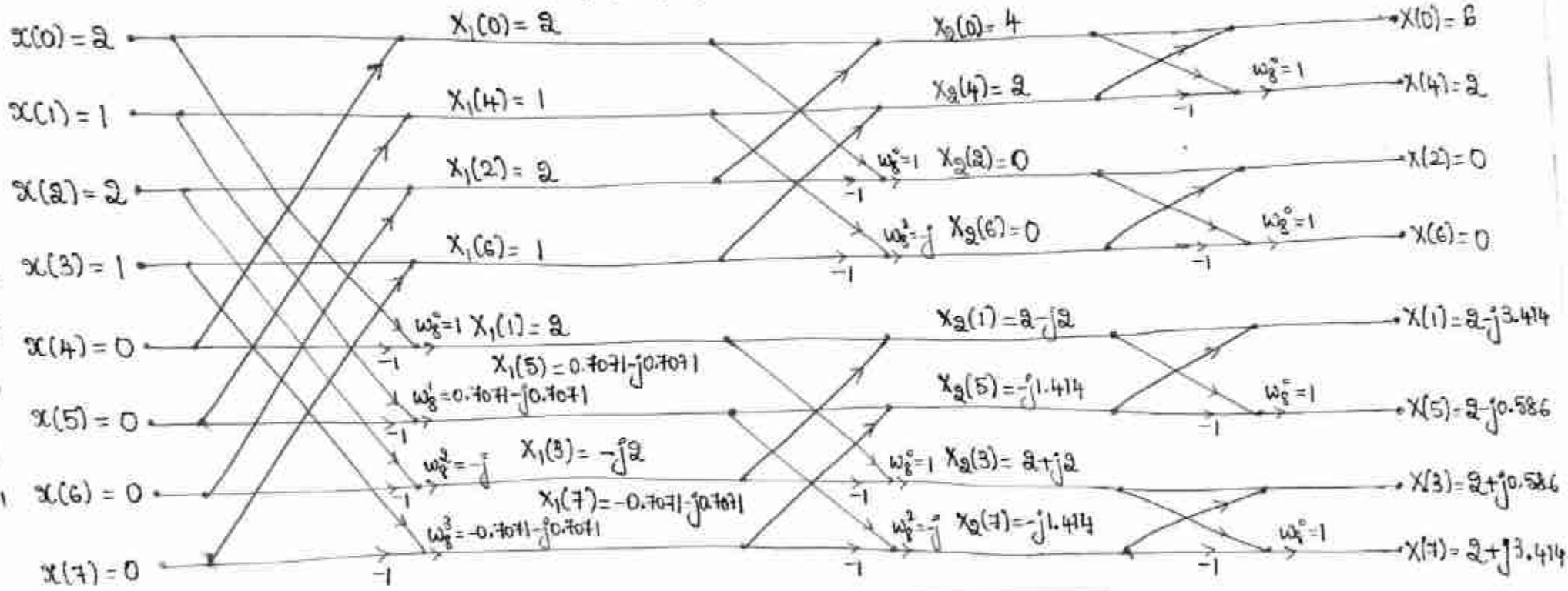


$$\therefore X(K) = \{8, 0, 0, 0, 0, 0, 0, 0\}$$

(11)

25) Find 8-point DFT for the sequence $x(n)=\{2,1,2,1\}$ using DIF-FFT.

Sol^n: Since $N=8$, $x(n)=\{2,1,2,1,0,0,0,0\}$

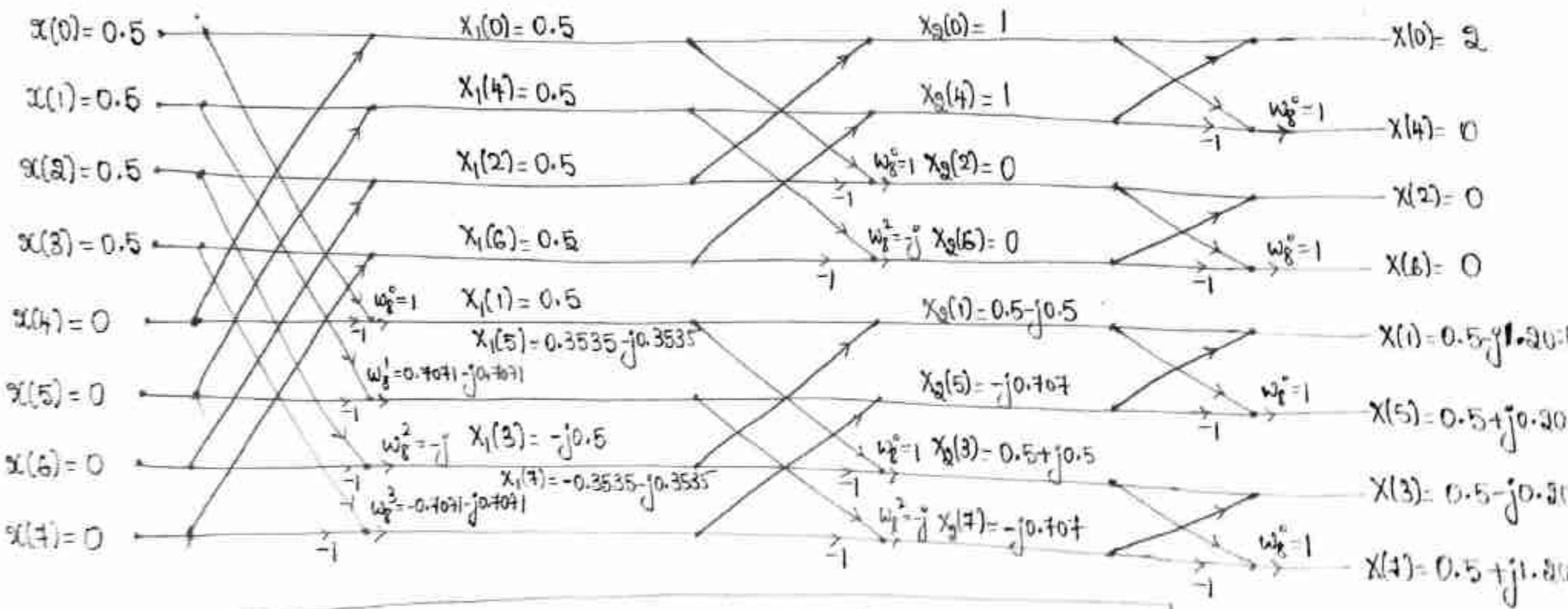


$\therefore$ $X(K)=\{6, 2-j3.414, 0, 2+j0.586, 2, 2-j0.586, 0, 2+j3.414\}$

(18)

26) Find 8-point DFT of $x(n) = \{0.5, 0.5, 0.5, 0.5, 0, 0, 0, 0\}$ using DIF-FFT.

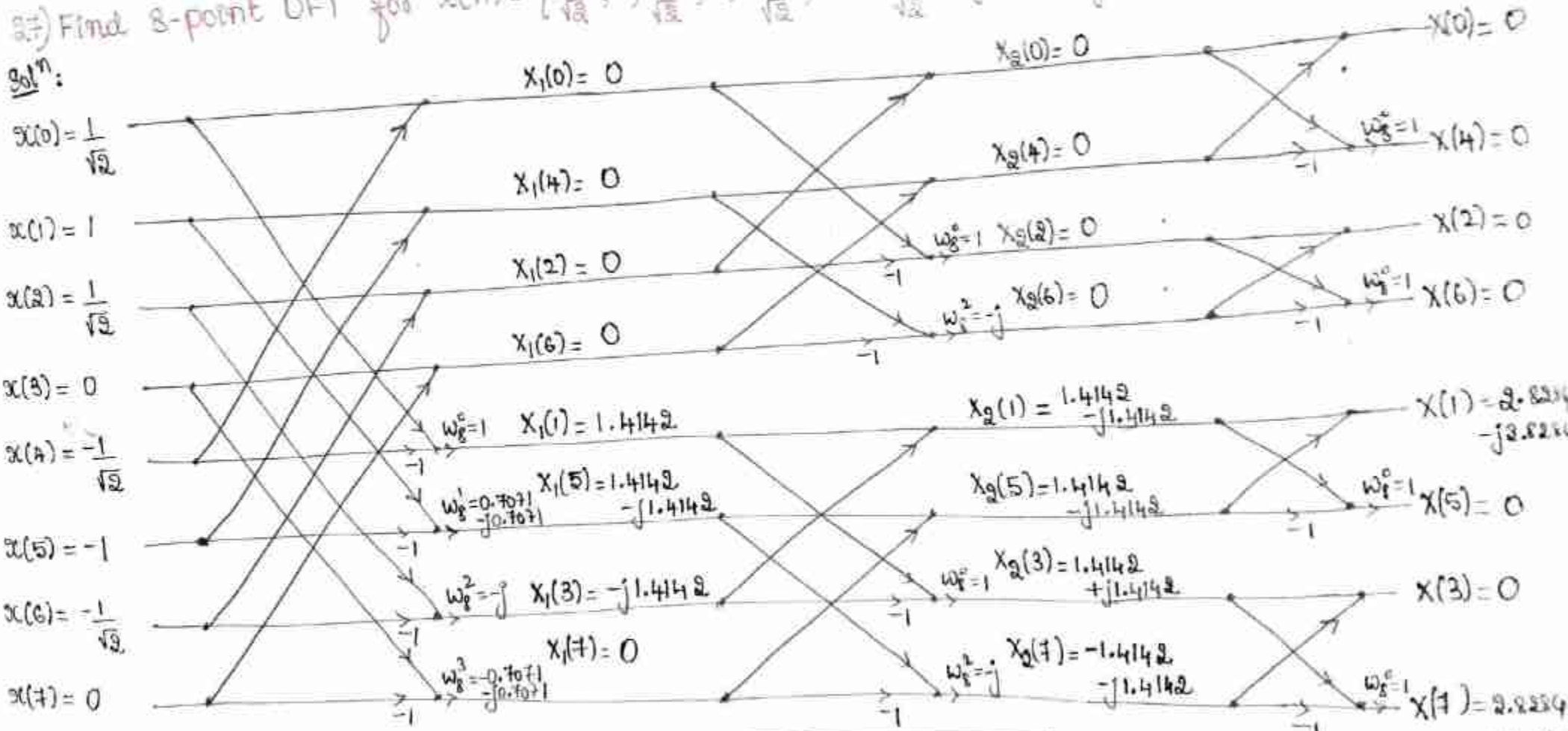
Soln:



$\therefore X(k) = \{2, 0.5 - j1.207, 0, 0.5 - j0.207, 0, 0.5 + j0.207, 0, 0.5 + j1.207\}$

8.7) Find 8-point DFT for $x(n) = \{\frac{1}{\sqrt{2}}, 1, \frac{1}{\sqrt{2}}, 0, -\frac{1}{\sqrt{2}}, -1, -\frac{1}{\sqrt{2}}, 0\}$ using DIF-FFT

Soln:

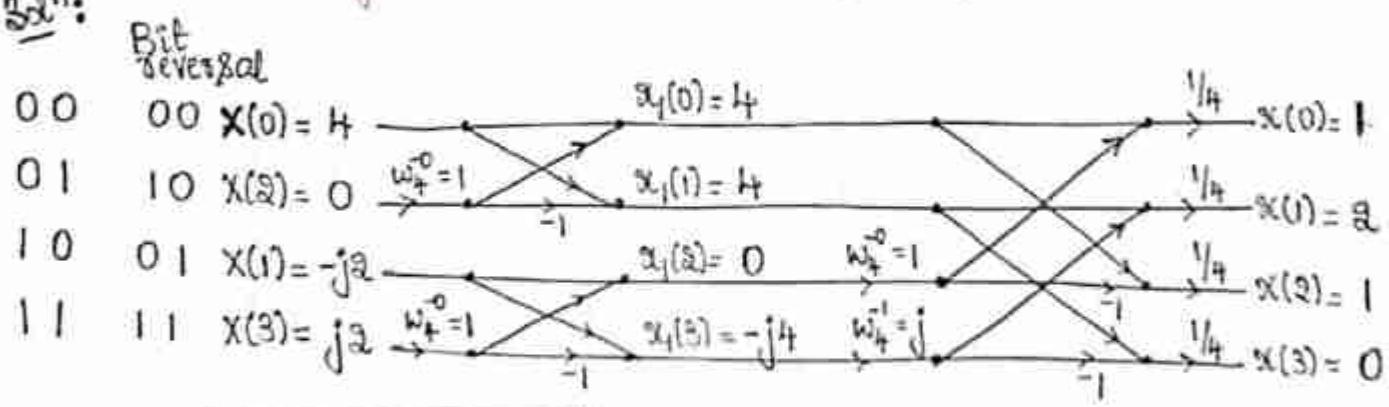


$\therefore X(K) = \{0, 2.8284 - j2.8284, 0, 0, 0, 0, 0, 2.8284 + j2.8284\}$

Problems on Inverse Radix-2 DIF-FFT algorithm :-

36) For the given DFT points, find the sequence $x(n)$ using DIF-FFT algorithm: $X(K) = \{4, -j2, 0, j2\}$.

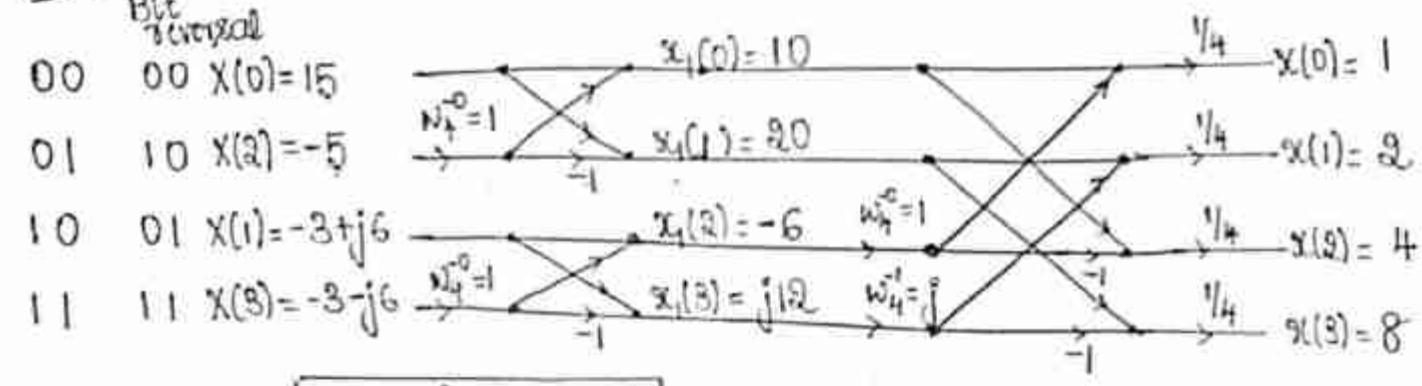
Soln:



$\therefore \boxed{x(n) = \{1, 2, 1, 0\}}$

37) Find 4-point IDFT for $X(K) = \{15, -3+j6, -5, -3-j6\}$.

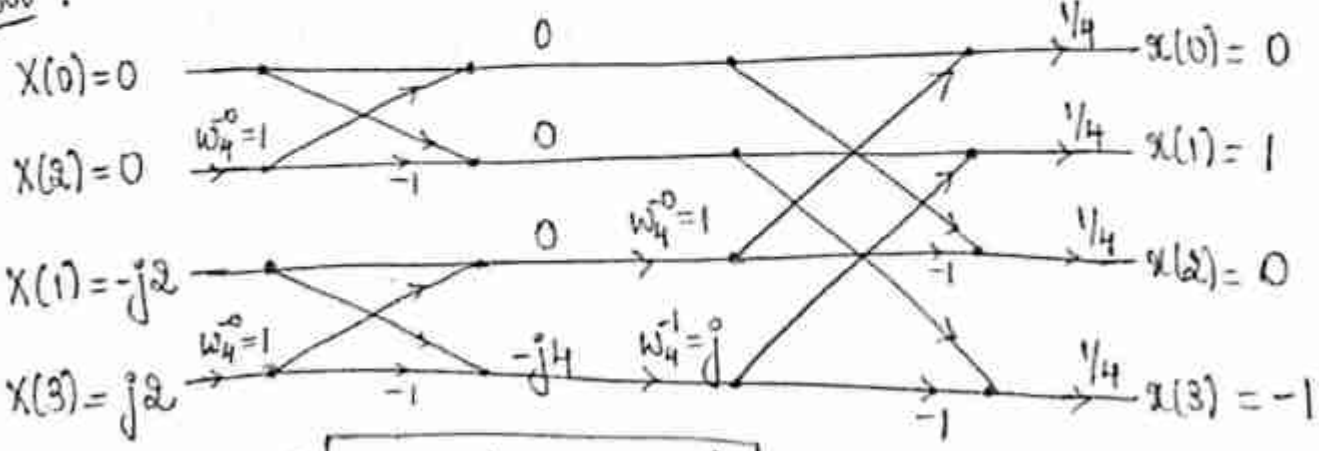
Soln:



$\therefore \boxed{x(n) = \{1, 2, 4, 8\}}$

38) $X(K) = \{0, -j2, 0, j2\}$

Soln:



$\therefore \boxed{x(n) = \{0, 1, 0, -1\}}$

Scanned with CamScanner

(17)

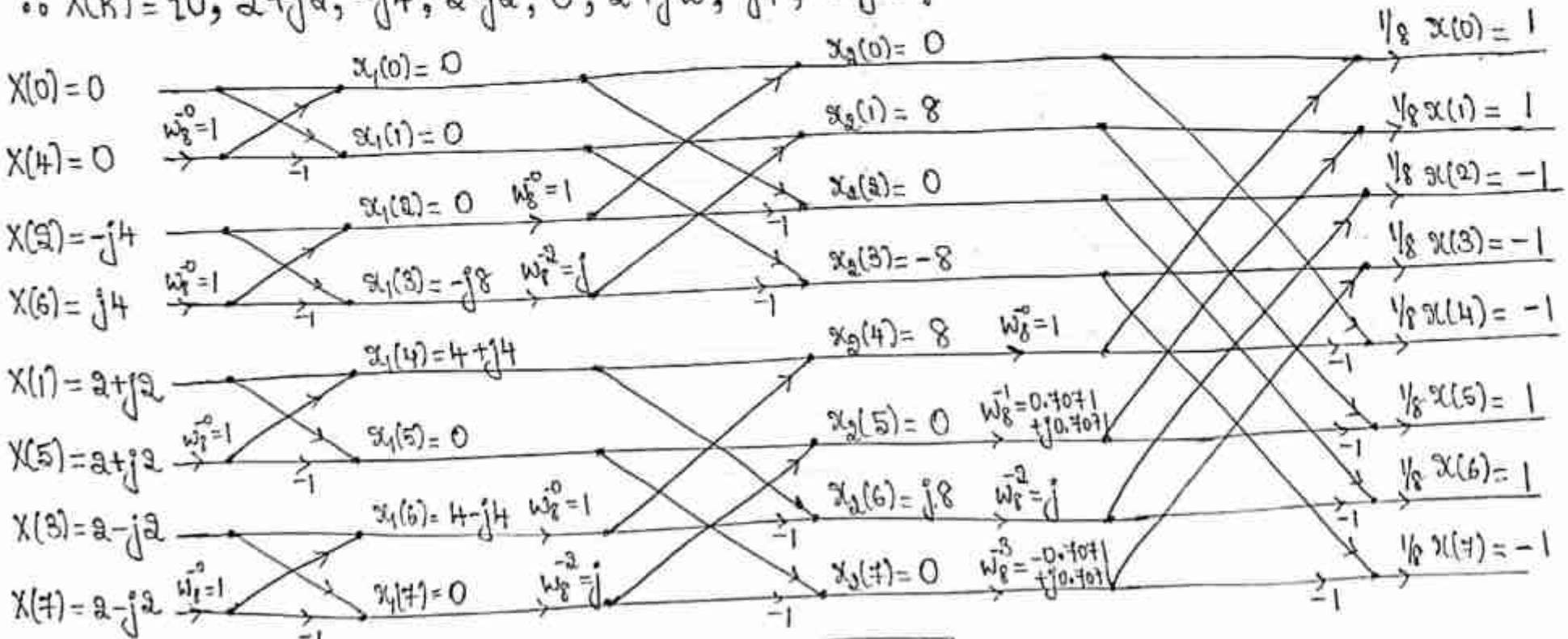
39) First five points of the Eight point DFT of a real valued sequence is given by: $X(K) = \{0, 2+j2, -j4, 2-j2, 0\}$. Determine the remaining points. Hence find the original sequence $x(n)$ using DIF-FFT algorithm.

Soln :- $X(K) = X^*(N-K)$

Here $N=8$, $K=5,6,7$

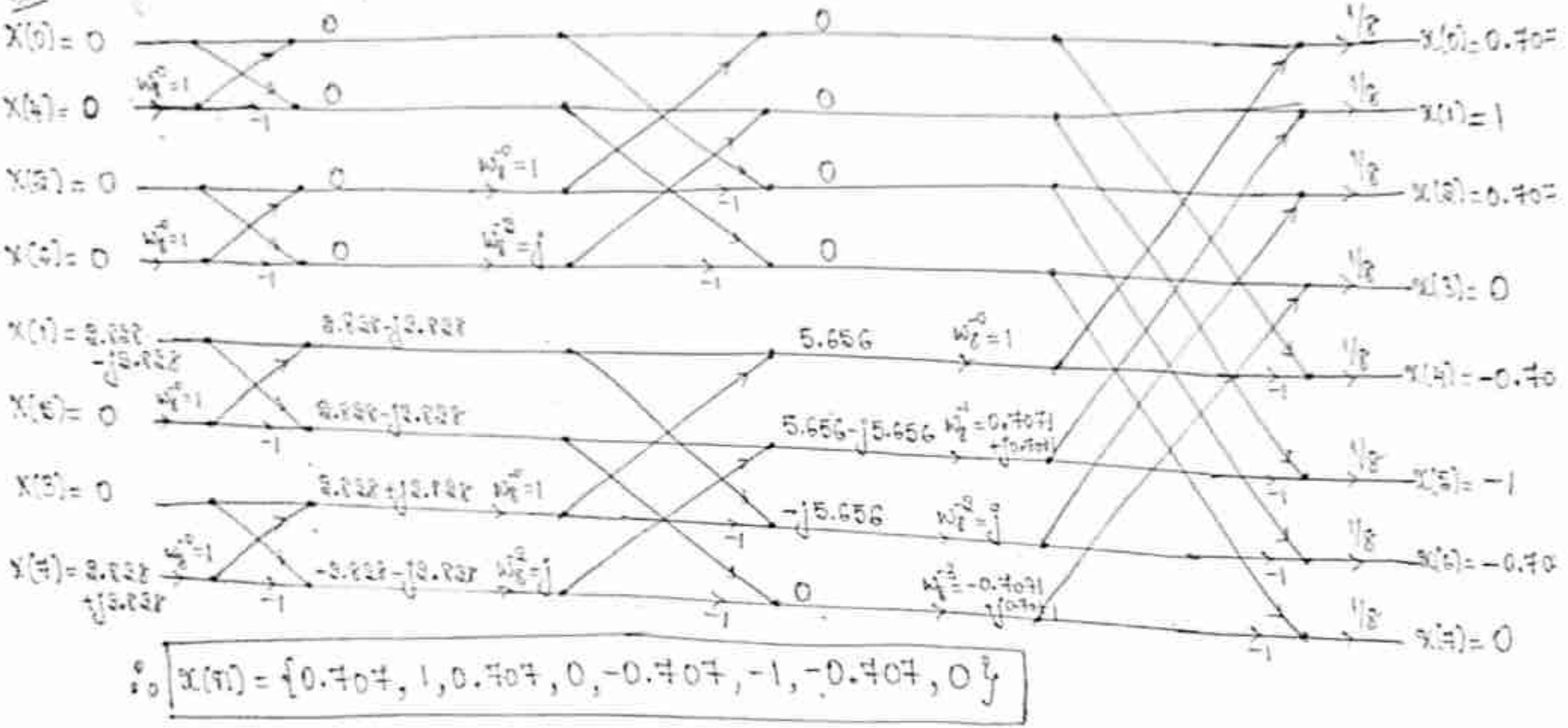
$X(5) = X^*(8-5) = X^*(3) = \underline{2+j2}$; $X(6) = X^*(8-6) = X^*(2) = \underline{j4}$; $X(7) = X^*(8-7) = X^*(1) = \underline{2-j2}$

$\therefore X(K) = \{0, 2+j2, -j4, 2-j2, 0, 2+j2, j4, 2-j2\}$



$\therefore \boxed{x(n) = \{1, 1, -1, -1, -1, 1, 1, -1\}}$

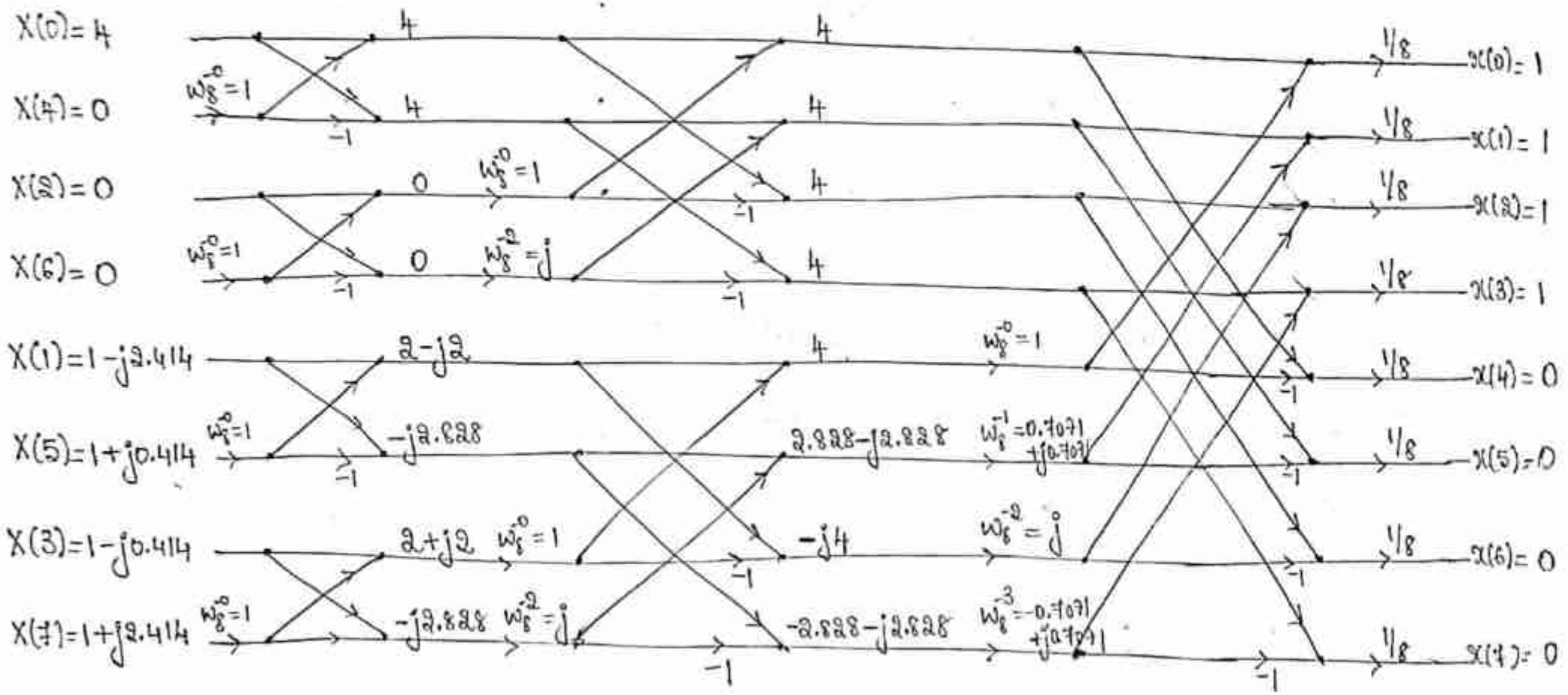
Scanned with CamScanner



$\therefore \boxed{x(n) = \{0.707, 1, 0.707, 0, -0.707, -1, -0.707, 0\}}$

(12.) a) For the given DFT points $X(K) = \{4, 1-j2.414, 0, 1-j0.414, 0, 1+j0.414, 0, 1+j2.414\}$. Find the sequence $x(n)$ corresponding to 8-point DFT using DIF-FFT.

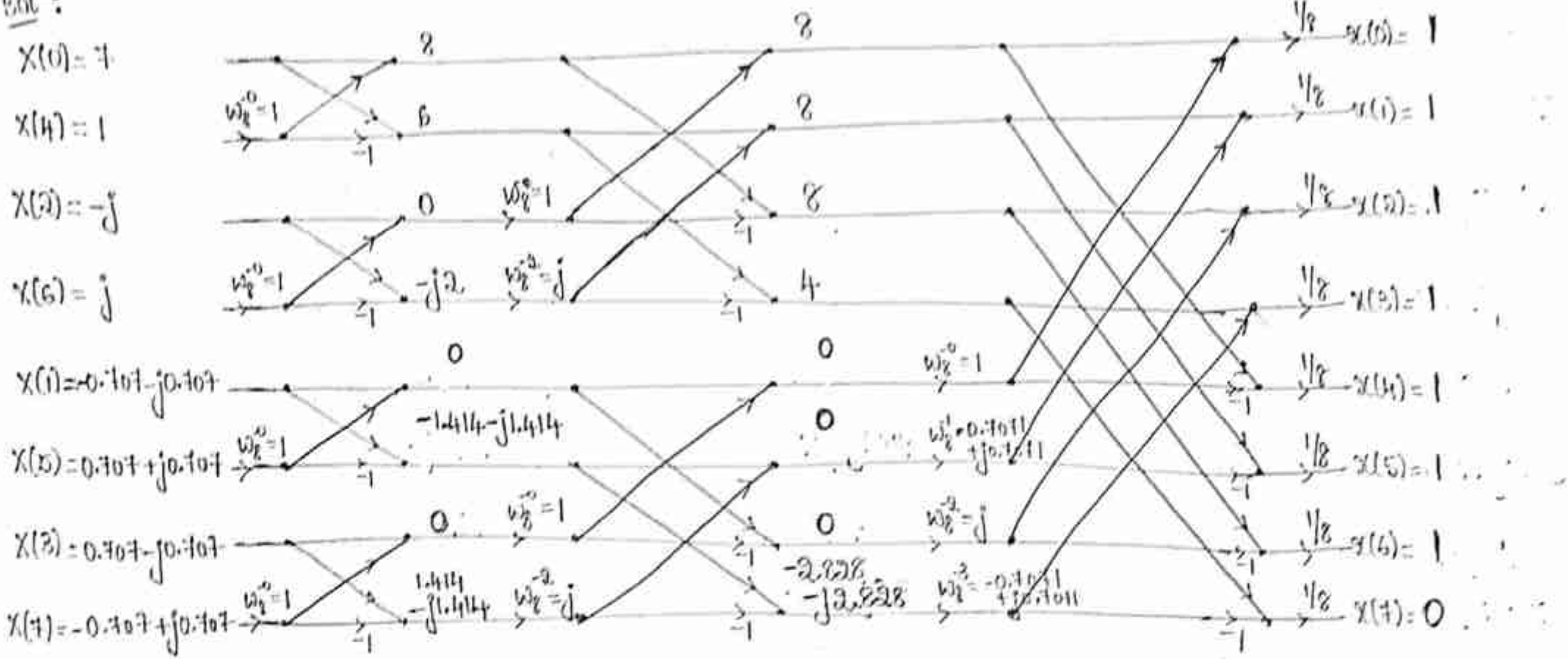
Soln:



$\therefore$ $x(n) = \{1, 1, 1, 1, 0, 0, 0, 0\}$

b2) $X(n) = \{1, 0.707 - j0.707, -j, -0.707 - j0.707, 1, 0.707 + j0.707, j, -0.707 + j0.707\}$

Soln:



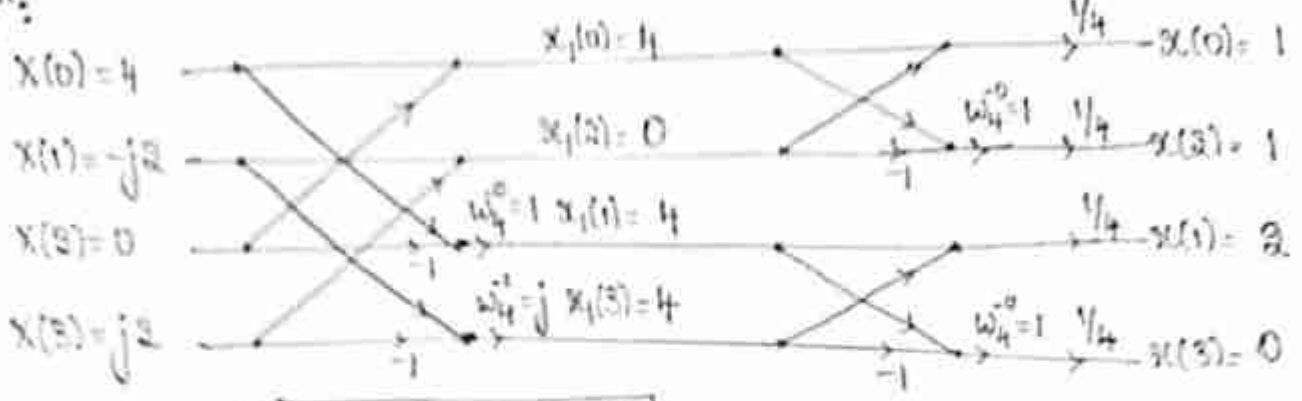
$\therefore \boxed{x(n) = \{1, 1, 1, 1, 1, 1, 1, 0\}}$

Scanned with CamScanner

Problems on Inverse Radix-2 DIT-FFT :—

43) For the given DFT pairs $x(k) = \{4, -j2, 0, j2\}$. Find the sequence $x(n)$ using DIT-FFT.

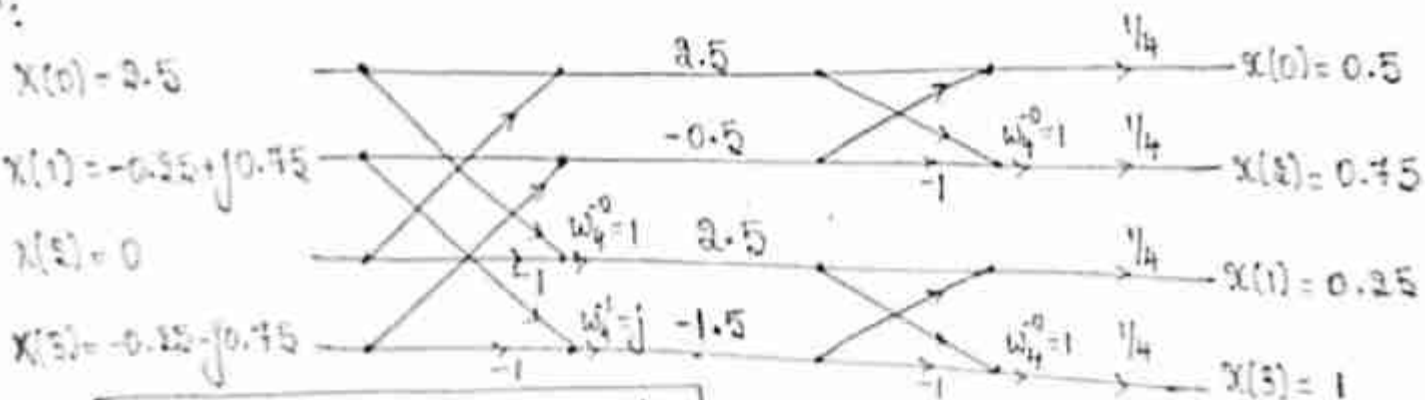
Soln:



$\therefore \boxed{x(n) = \{1, 2, 1, 0\}}$

44) $x(k) = \{2.5, -0.25 + j0.75, 0, -0.25 - j0.75\}$.

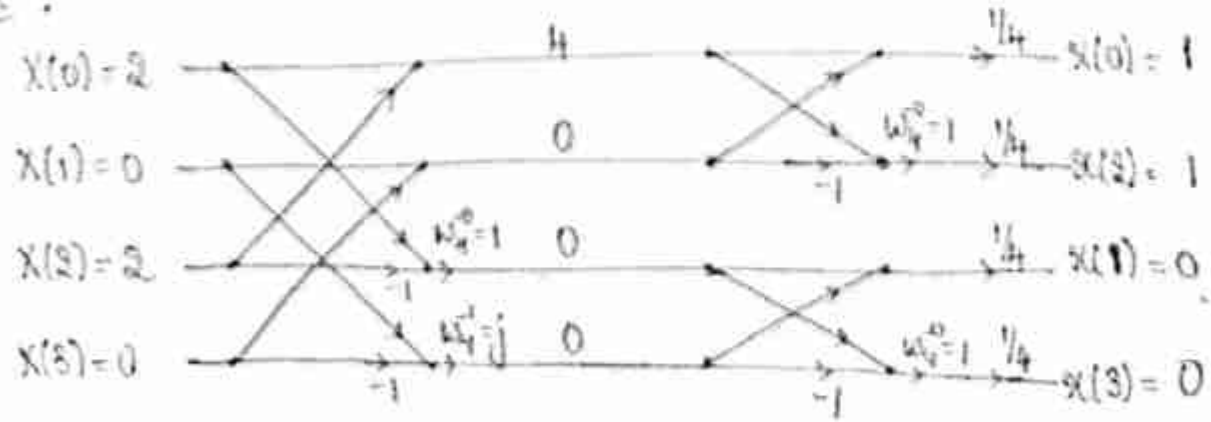
Soln:



$\boxed{x(n) = \{0.5, 0.25, 0.75, 1\}}$

45) $x(k) = \{2, 0, 2, 0\}$

Soln:

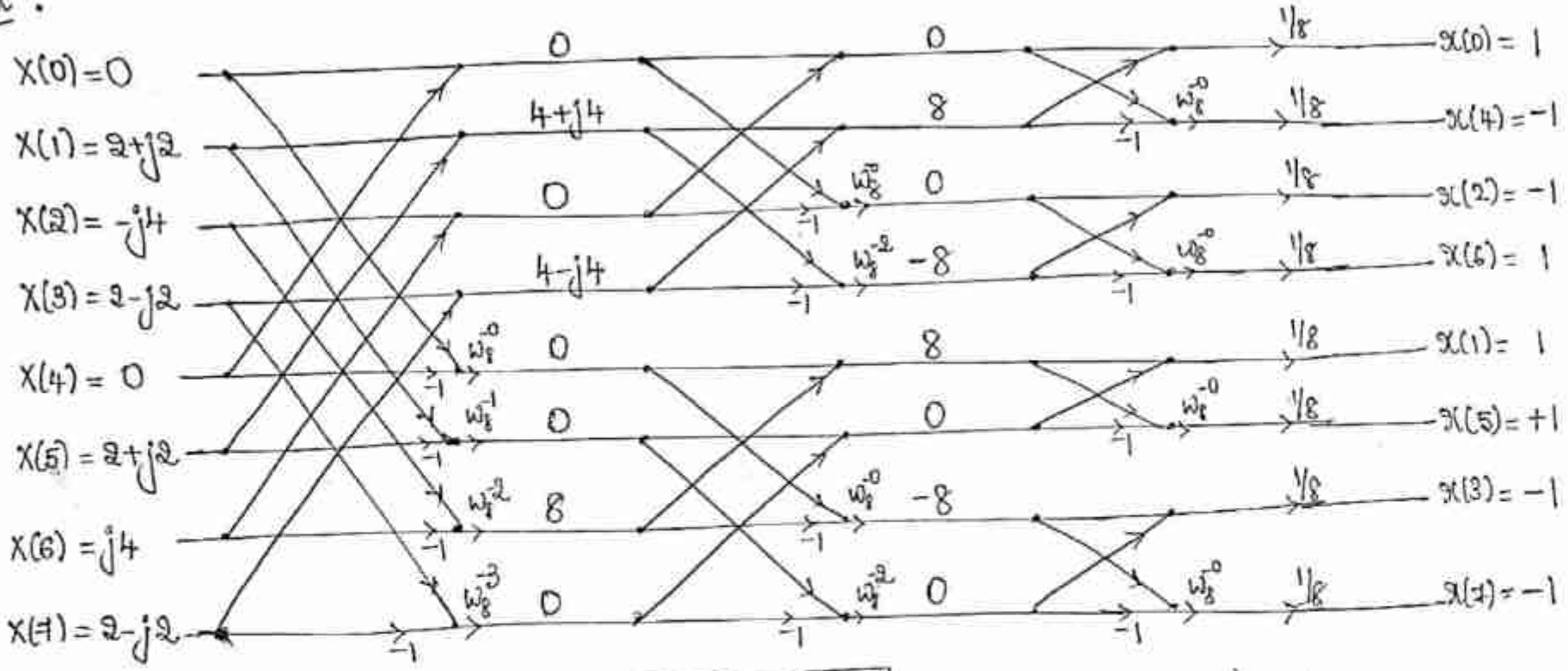


$\boxed{x(n) = \{1, 0, 1, 0\}}$

Scanned with CamScanner

4b) $X(K) = \{0, 2+j2, -j4, 2-j2, 0, 2+j2, j4, 2-j2\}$.

Soln:

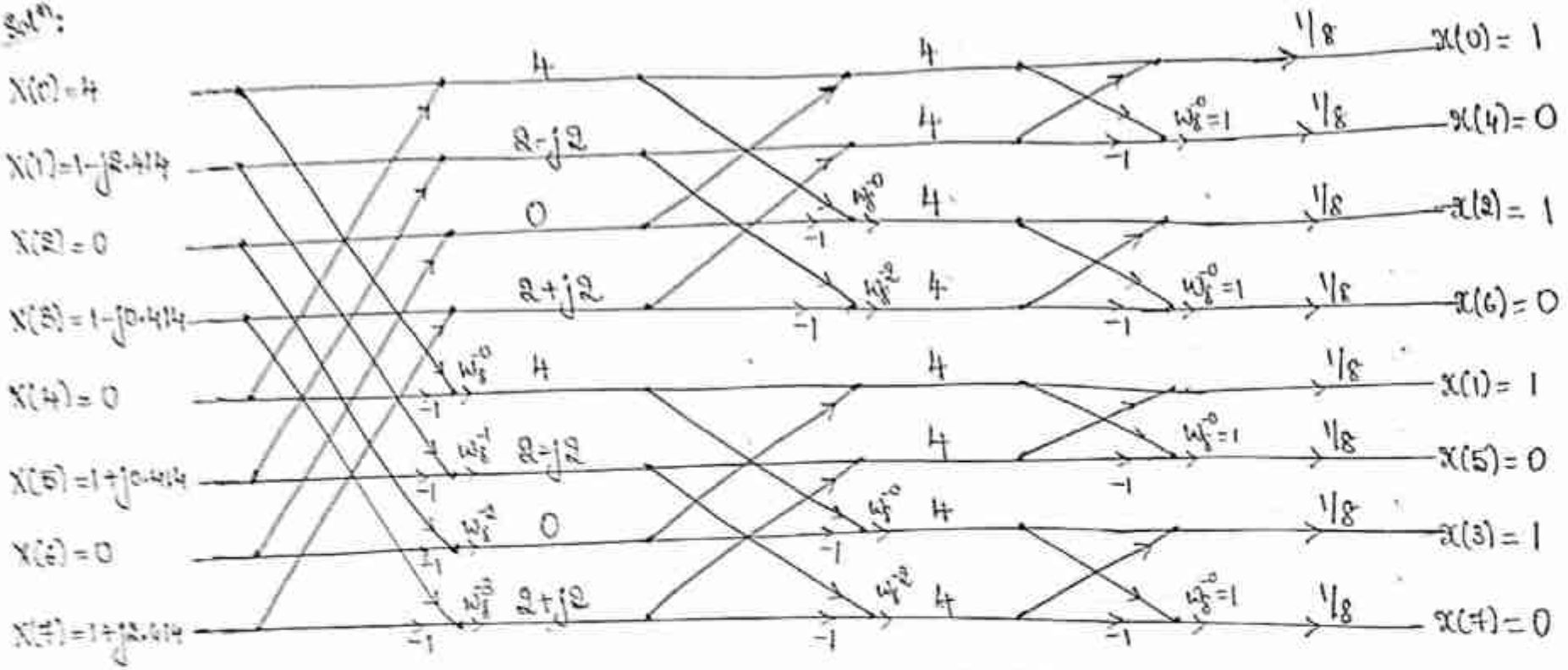


$\therefore \boxed{x(n) = \{1, 1, -1, -1, -1, +1, 1, -1\}}$

Scanned with CamScanner

(5) $X(k) = \{4, 1-j2.414, 0, 1-j0.414, 0, 1+j0.414, 0, 1+j2.414\}$

Soln:



$\therefore \boxed{x(n) = \{1, 1, 1, 1, 0, 0, 0, 0\}}$

Scanned with CamScanner

48) $X(K) = \{0, 2.8284 - j2.8284, 0, 0, 0, 0, 0, 2.8284 + j2.8284\}$

Soln:

$X(0) = 0$

$X(1) = 2.8284 - j2.8284$

$X(2) = 0$

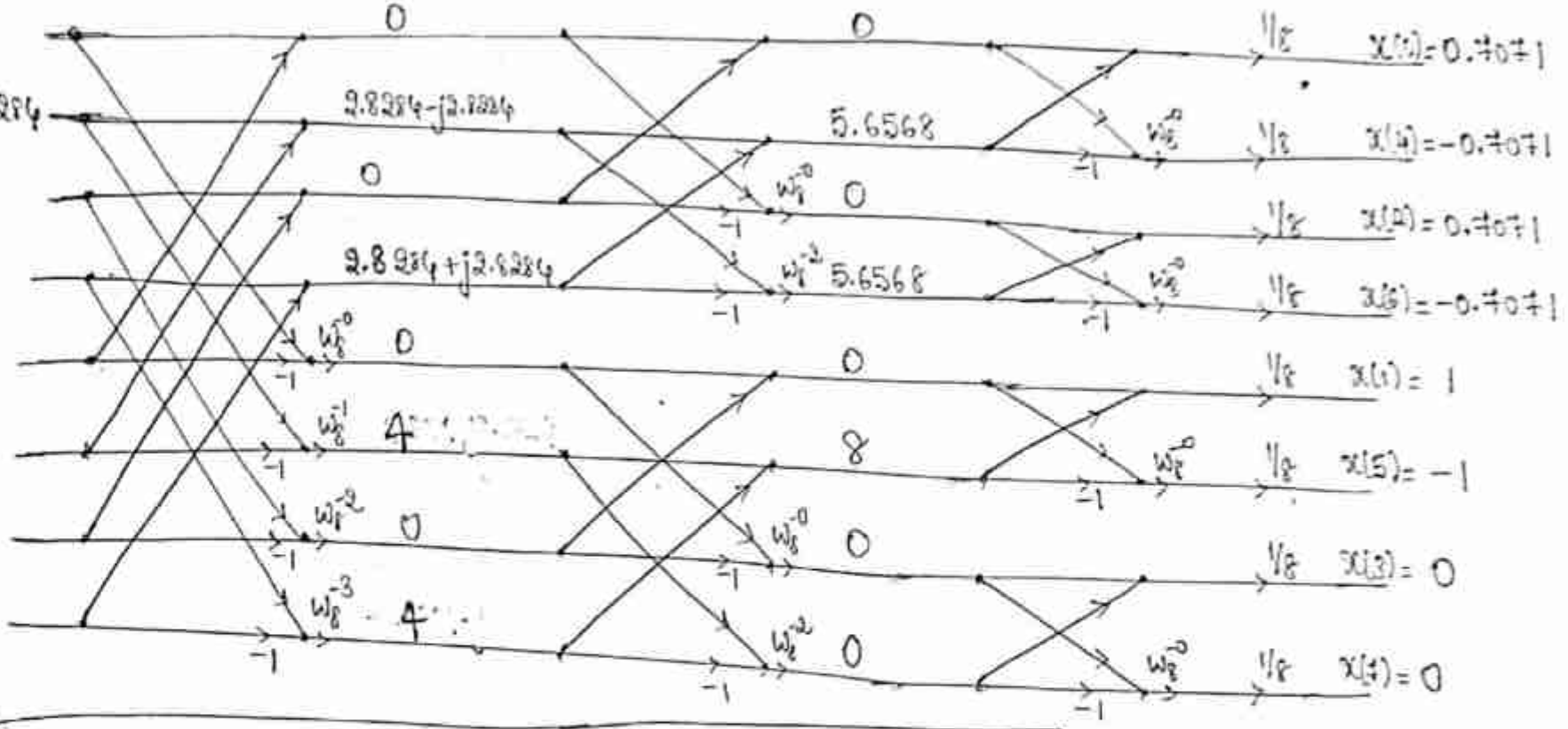
$X(3) = 0$

$X(4) = 0$

$X(5) = 0$

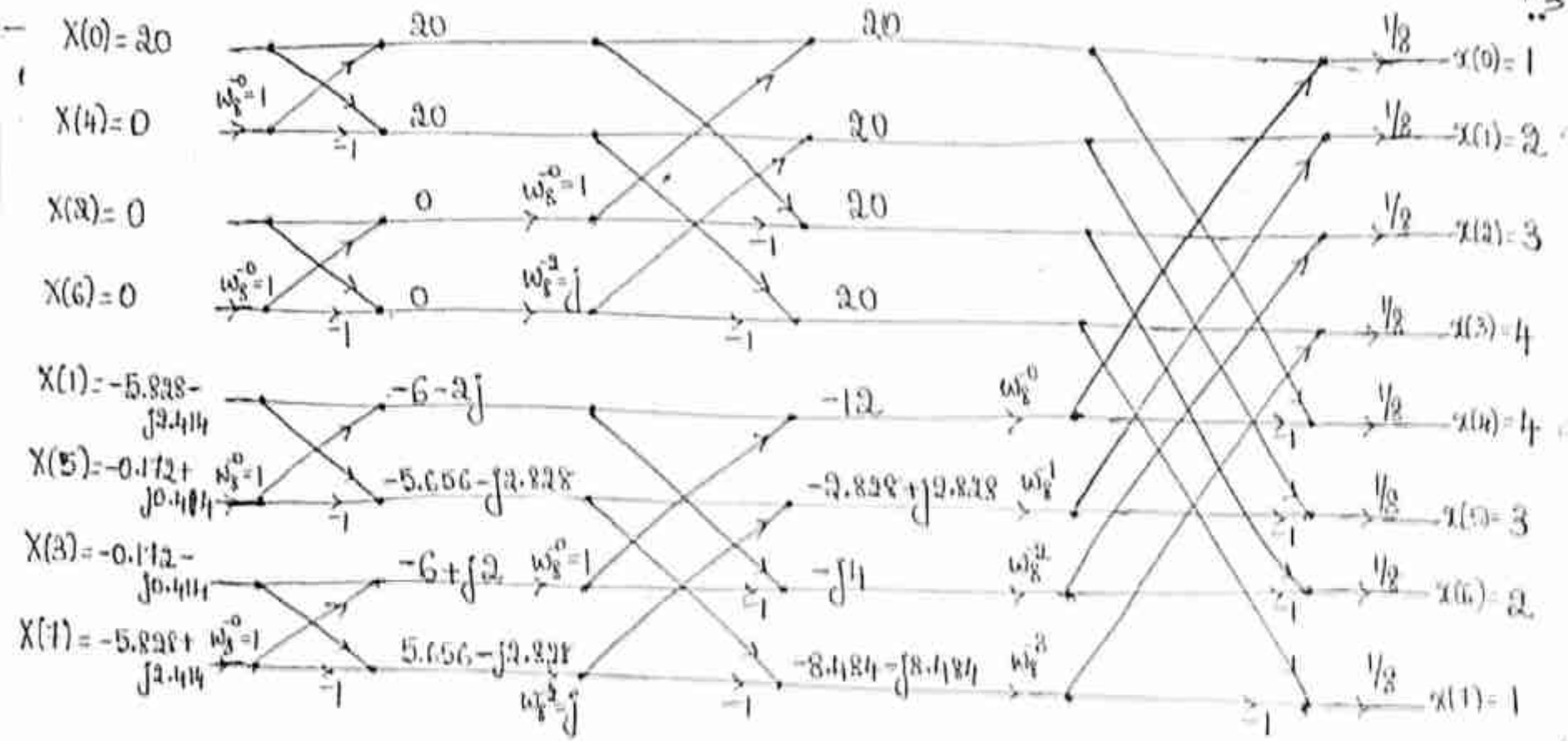
$X(6) = 0$

$X(7) = 2.8284 + j2.8284$



$\therefore x(n) = \{0.7071, 1, 0.7071, 0, -0.7071, -1, -0.7071, 0\}$

Scanned with CamScanner

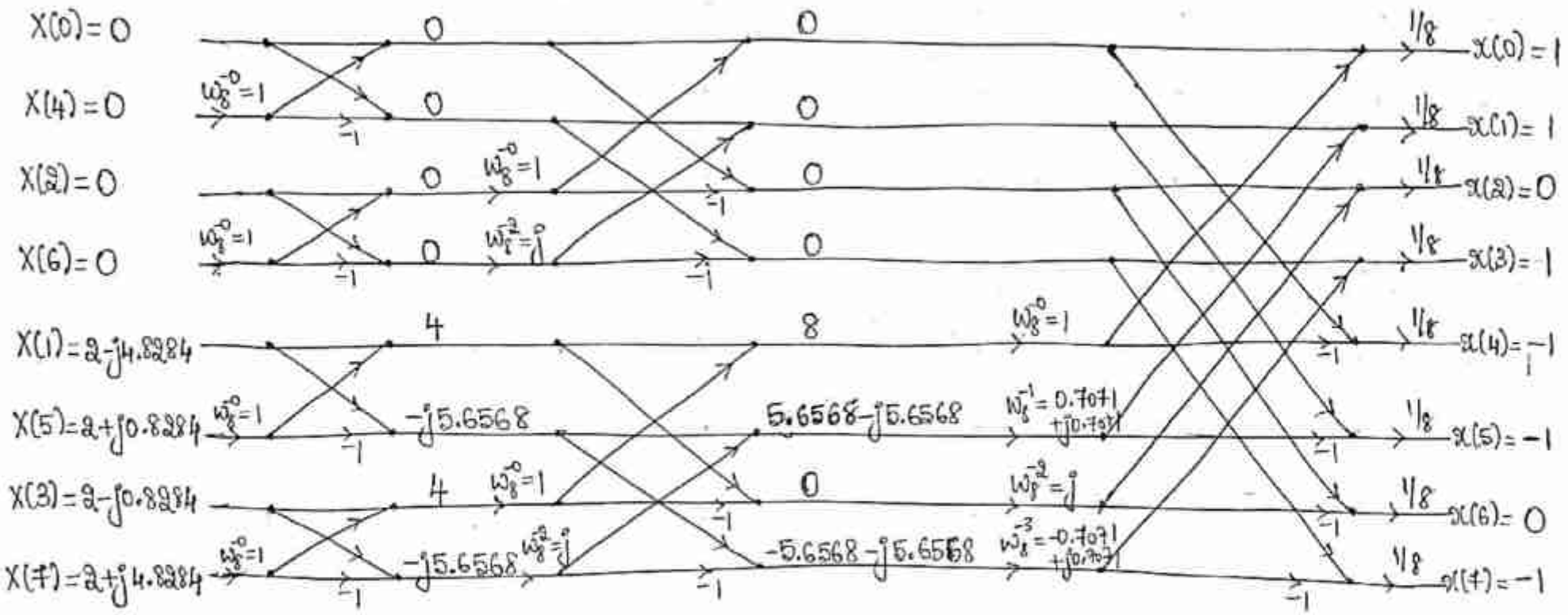


$$\therefore x(n) = \{1, 2, 3, 4, 4, 3, 2, 1\}$$

Scanned with CamScanner

5a) Find 8-point DFT of a sequence $x(n)$ is given by $X(K) = \{0, 2-j4.8284, 0, 2-j0.8284, 0, 2+j0.8284, 0, 2+j4.8284\}$. Determine $x(n)$ using DIF-FFT algorithm.

Soln:-

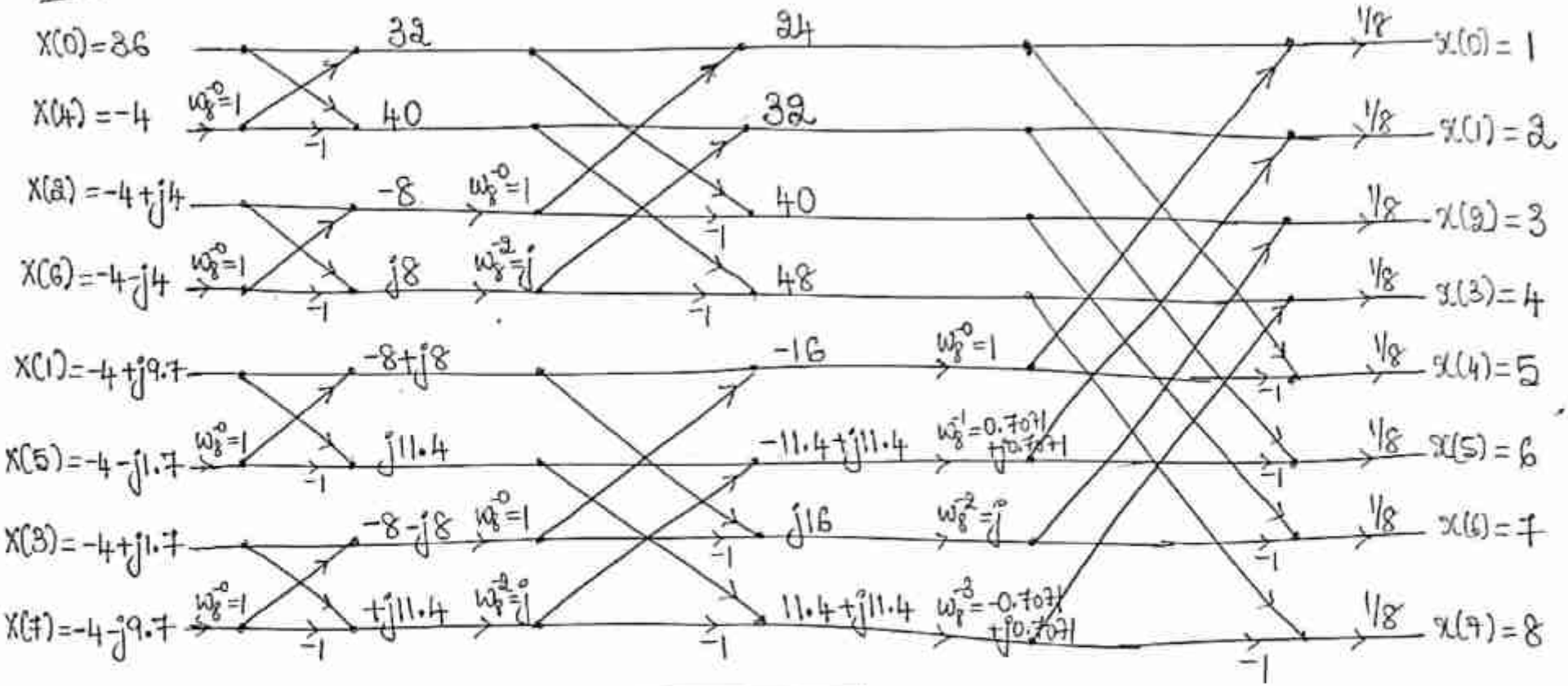


$\therefore \boxed{x(n) = \{1, 1, 0, 1, -1, -1, 0, -1\}}$

Scanned with CamScanner

– 5) Find IDFT of $X(K) = \{36, -4+j9.7, -4+j4, -4+j1.7, -4, -4-j1.7, -4-j4, -4-j9.7\}$. Using DIF-FFT algorithm. Show clearly all the intermediate results.

Sol^n:



$\therefore \boxed{x(n) = \{1, 2, 3, 4, 5, 6, 7, 8\}}$

Scanned with CamScanner

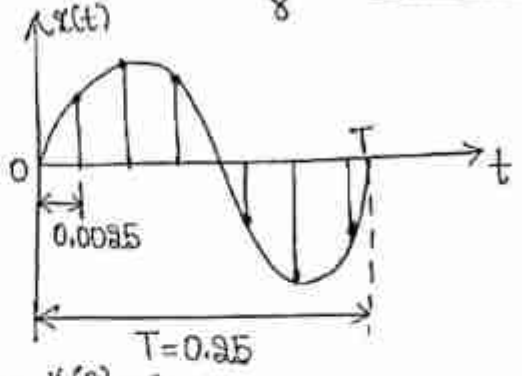
(52) Determine 8-point DFT of a Continuous time Signal $x(t) = \sin(2\pi f t)$ with $f = 50Hz$. Use DIF-FFT algorithm.

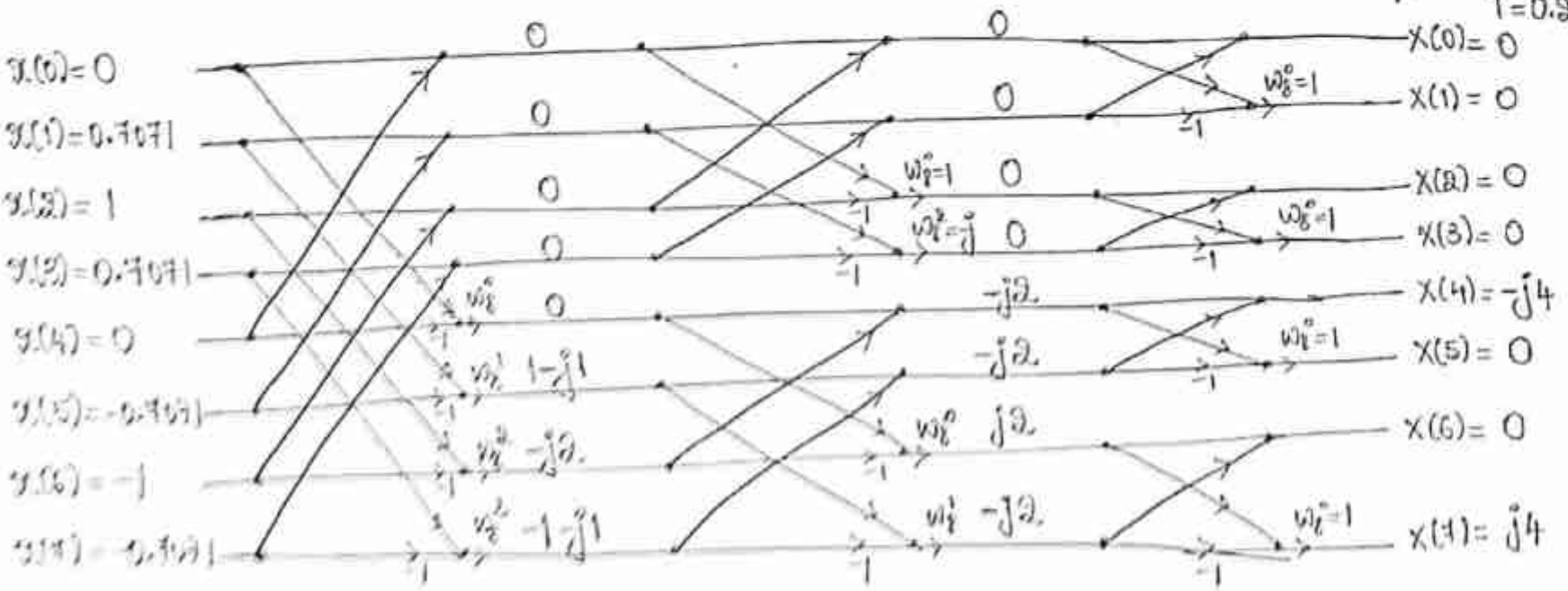
Soln: $x(n) = x(t)|_{t=nT_s}$ Given $f = 50Hz$

$$T = \frac{1}{f} = \frac{1}{50} = 0.02 \text{ sec}$$; Take the interval as $\frac{0.02}{8} = 0.0025$ sec

$x(n) = \sin(2\pi f \times nT_s) = \sin(2 \times 180 \times 50 \times n \times 0.0025)$

$\therefore x(n) = \{0, 0.7071, 1, 0.7071, 0, -0.7071, -1, -0.7071\}$



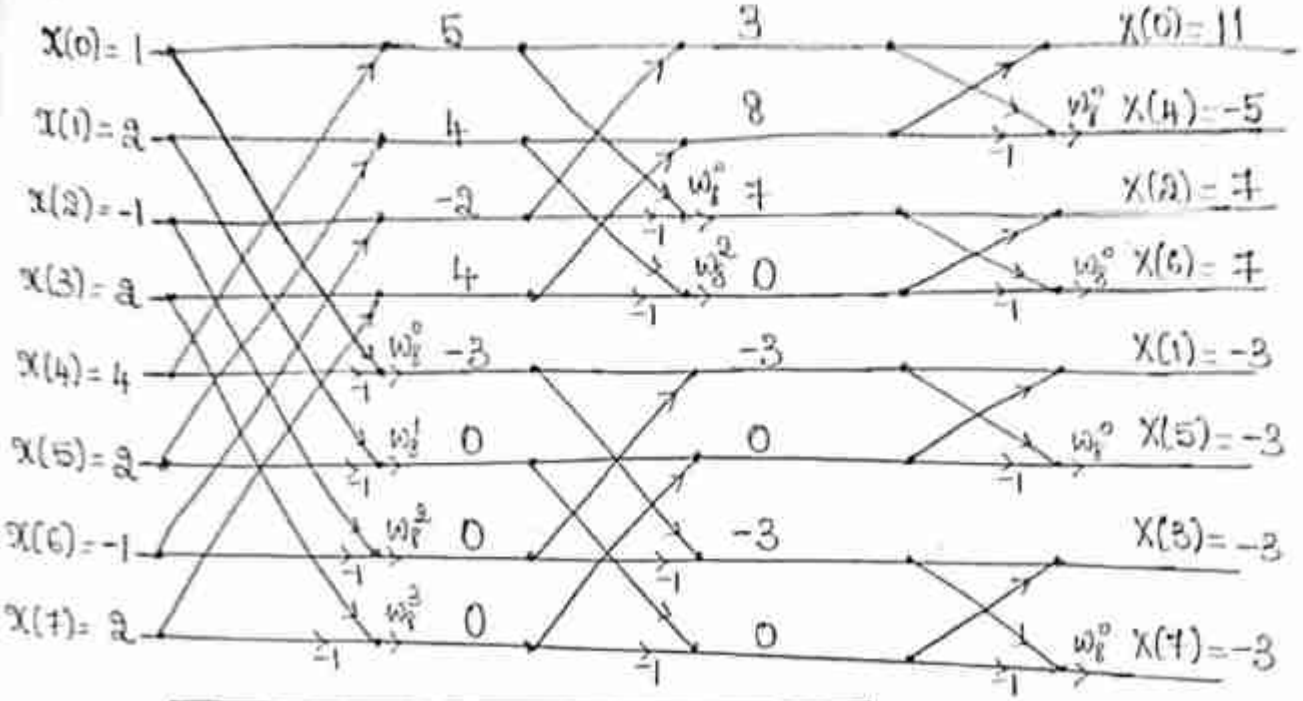


$$\therefore \boxed{X(k) = \{0, 0, 0, 0, -j4, 0, 0, j4\}}$$

Scanned with CamScanner

53) Using DIF-FFT algorithm, compute DFT of the sequence, $x(n) = \{1, 2, -1, 2, 4, 2, -1, 2\}$. If $x_1(n) = x((n-4))_8$, compute $X_1(K)$ without invoking FFT algorithm.

Soln:



$\therefore \boxed{X(K) = \{11, -3, 7, -3, -5, 3, 7, -3\}}$

From Circular time shifting property,

W.K.T. $x((n-m))_N \xleftrightarrow{DFT} X(K)\, W_N^{mK}$

Here $m = 4$,

$x_1(n) \xleftrightarrow{DFT} X_1(K)$

$x((n-4))_8 \xleftrightarrow{DFT} X(K)\, W_8^{4K} = X_1(K)$

$e^{-j\frac{2\pi}{8}\times 4} = \cos\pi - j\sin\pi$

$X_1(K) = X(K)\, W_8^{4K}$

@ $K=0$, $X_1(0) = X(0) W_8^0 = \underline{11}$

@ $K=1$, $X_1(1) = X(1) W_8^4 = -3 \times (-1) = \underline{3}$

@ $K=2$, $X_1(2) = X(2) W_8^8 = 7(1) = \underline{7}$

@ $K=3$, $X_1(3) = X(3) W_8^{12} = -3(-1) = \underline{3}$

@ $K=4$, $X_1(4) = X(4) W_8^{16} = (-5)(1) = \underline{-5}$

@ $K=5$, $X_1(5) = X(5) W_8^{20} = (-3)(-1) = \underline{3}$

@ $K=6$, $X_1(6) = X(6) W_8^{24} = (7)(1) = \underline{7}$

@ $K=7$, $X_1(7) = X(7) W_8^{28} = (-3)(-1) = \underline{3}$

$\therefore \boxed{X_1(K) = \{11, 3, 7, 3, -5, 3, 7, 3\}}$

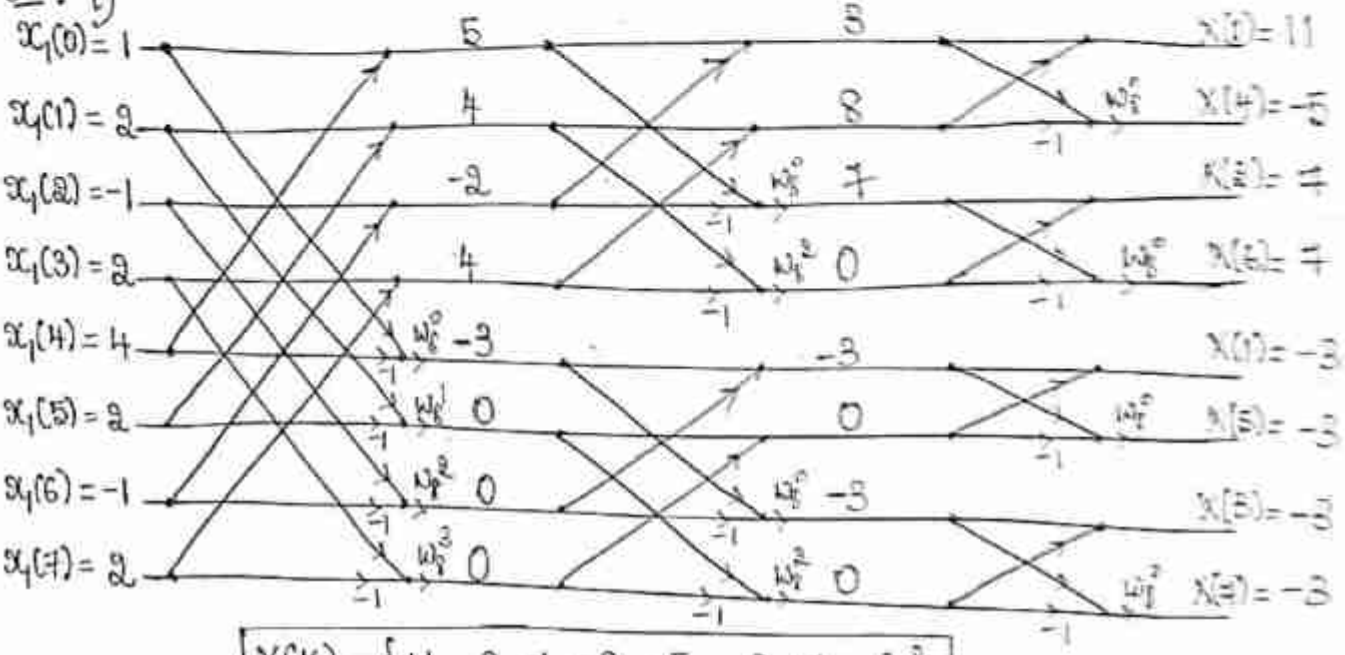
Scanned with CamScanner

54) Develop DIF-FFT algorithm for N = 8,

(i) Using resulting signal flow graph, Compute 8-point DFT of the sequence, $x_1(n) = \{1, 2, -1, 2, 4, 2, -1, 2\}$. Show all intermediate results.

(ii) If $x_2(n) = x_1((-n))_8$ without performing FFT find $x_2(k)$ using $X_1(k)$.

Sol:



$$X(K) = \{11, -3, 7, -3, -5, -3, 7, -3\}$$

From circular time reversal (folding) property

(ii) W.K.T. $x((-n))_N = x(N-n) \xleftrightarrow[N]{\text{DFT}} X((-K))_N = X(N-K)$

$$x((-n))_8 = x(8-n) \xleftrightarrow[8\text{-pt}]{\text{DFT}} X((-K))_8 = X(8-K)$$

@K=0, $X((-0))_8 = X(0) = 11$

@K=1, $X((-1))_8 = X(8-1) = X(7) = -3$

@K=2, $X((-2))_8 = X(8-2) = X(6) = 7$

@K=3, $X((-3))_8 = X(8-3) = X(5) = -3$

@K=4, $X((-4))_8 = X(8-4) = X(4) = -5$

@K=5, $X((-5))_8 = X(8-5) = X(3) = -3$

@K=6, $X((-6))_8 = X(8-6) = X(2) = 7$

@K=7, $X((-7))_8 = X(8-7) = X(1) = -3$

$$\therefore X_2(K) = \{11, -3, 7, -3, -5, -3, 7, -3\}$$

Scanned with CamScanner

55) Let $X(K), 0 \le K \le N-1$ to be discrete fourier transform of a sequence $x(n)$. How would you use an FFT algorithm to compute the inverse DFT (i.e., obtain $x(n)$ from $X(K)$)? Hence determine the sequence $x(n)$ whose 8-point DFT is given by the sequence $X(K) = \{0, 4-j4, 0, 0, 0, 0, 0, 4+j4\}$. Use the decimation-in-frequency FFT algorithm to arrive at your answer. provide the values at the intermediate stages.

i) IDFT using FFT algorithms:

The DFT is given as, $X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$

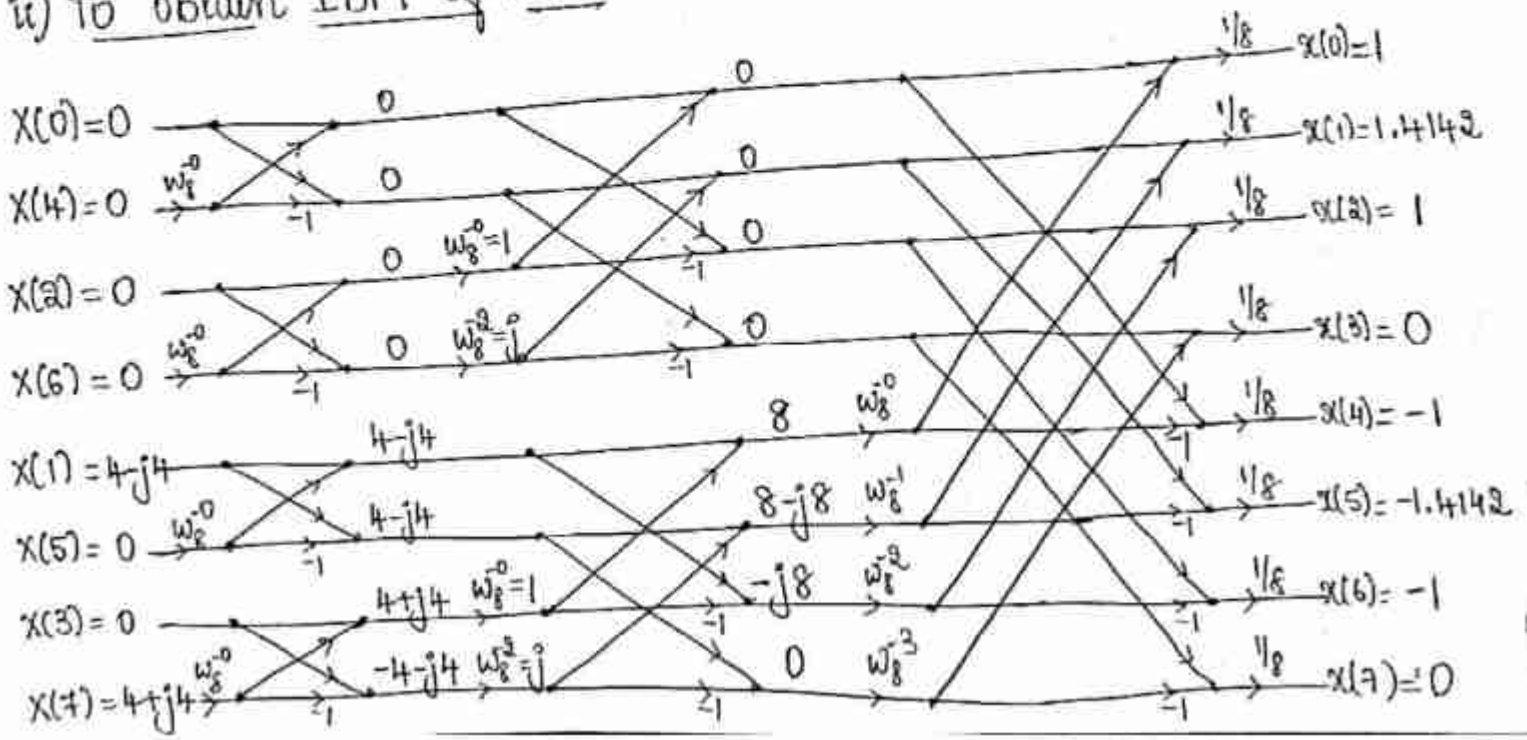
The IDFT is given as, $x(n) = \frac{1}{N} \sum_{K=0}^{N-1} X(K) W_N^{-Kn}$

On comparing above two equations, we find that FFT algorithms can be used to calculate IDFT with minor modifications i.e.,

i) The sign of exponent of twiddle factor, W_N is made negative.

ii) The overall value after summation is divided by N.

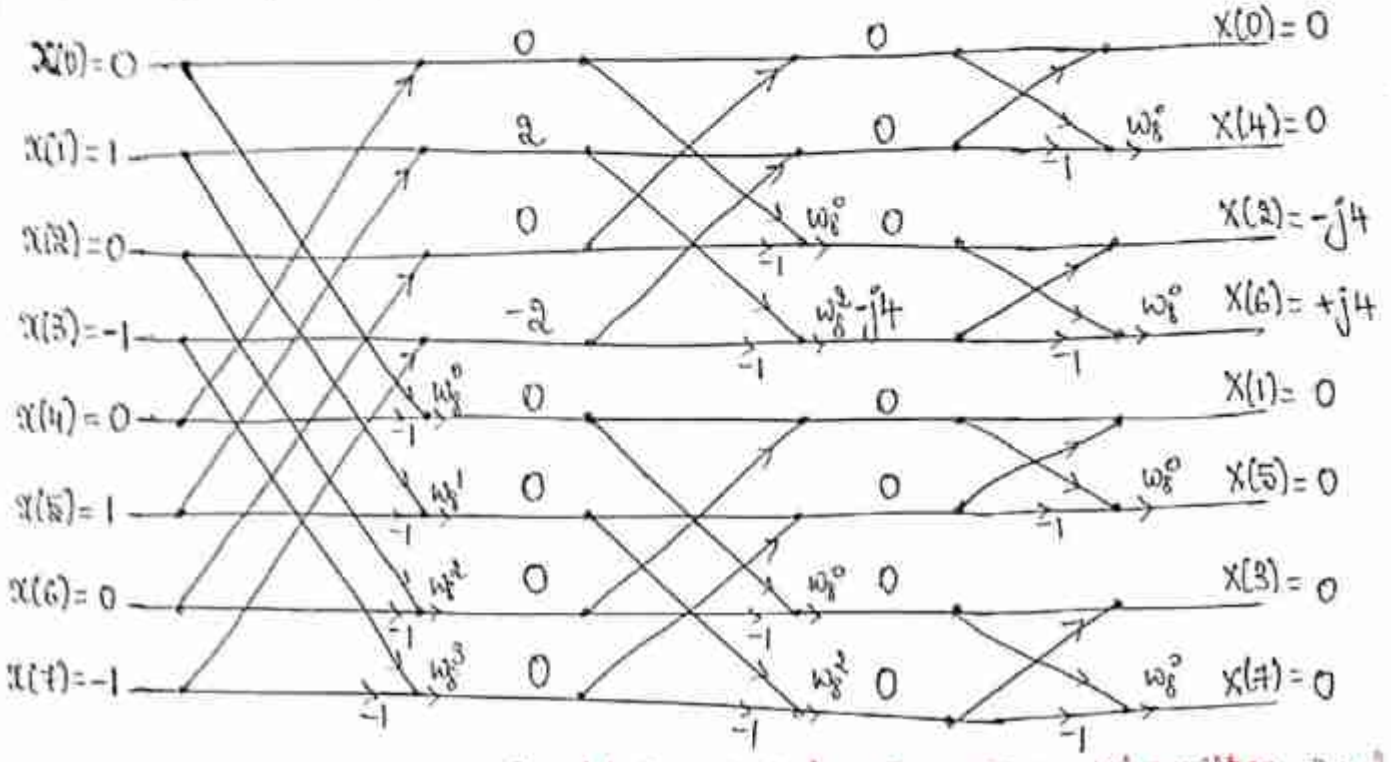
ii) To obtain IDFT of $X(K)$:-



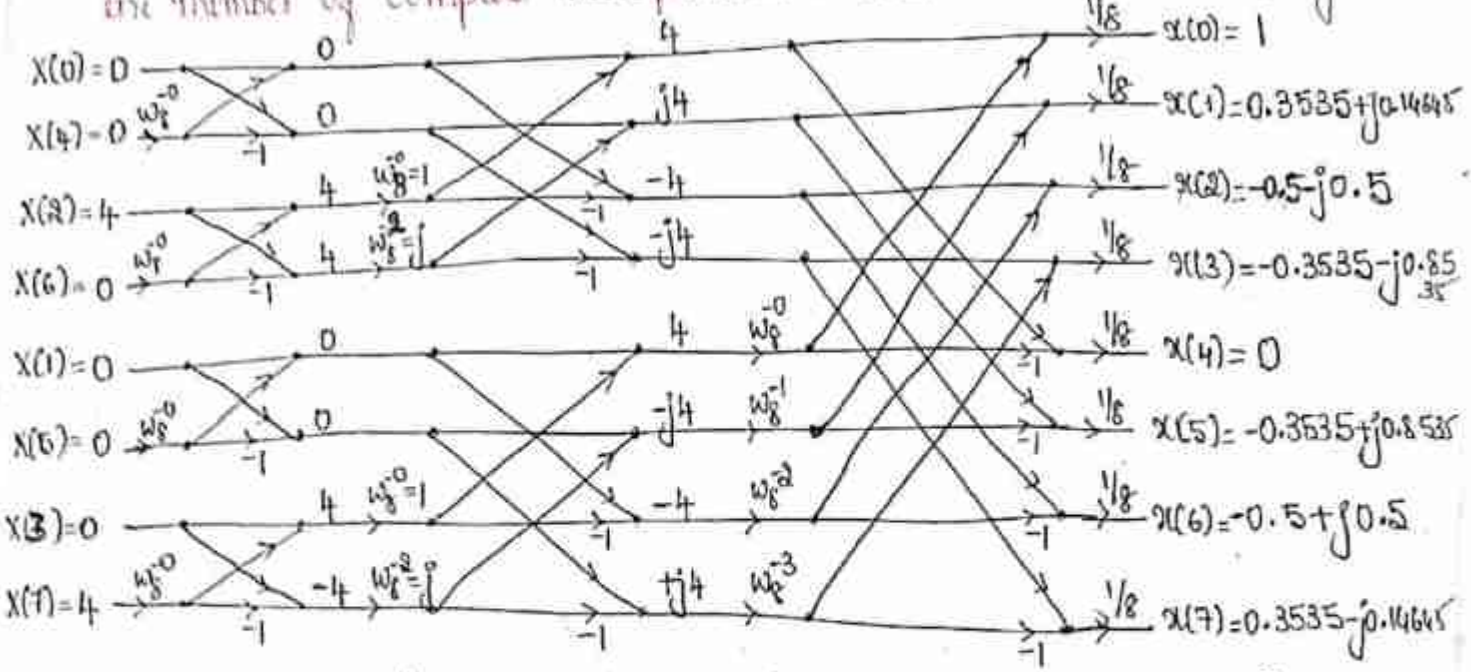
Scanned with CamScanner

(b) Develop the DIF-FFT algorithm for N=8. Using the resulting signal flowgraph, compute the 8-point DFT of the sequence $x(n) = \sin\left(\frac{\pi}{2}n\right)$; $0 \le n \le 7$.

Soln: $x(n) = \{0, 1, 0, -1, 0, 1, 0, -1\}$



→ X(K) = {0, 0, 4, 0, 0, 0, 0, 4}. obtain x(n) by DIF-IFFT algorithm. Find the number of complex multiplications and additions in the algorithm



Number of complex multiplications $= \frac{N}{2}\log_2 N = \frac{8}{2}\log_2 8 = \underline{12}$

Number of complex additions $= N \log_2 N = 8 \log_2 8 = \underline{24}$

Scanned with CamScanner

problems on linear and Circular convolution using DIT and DIF

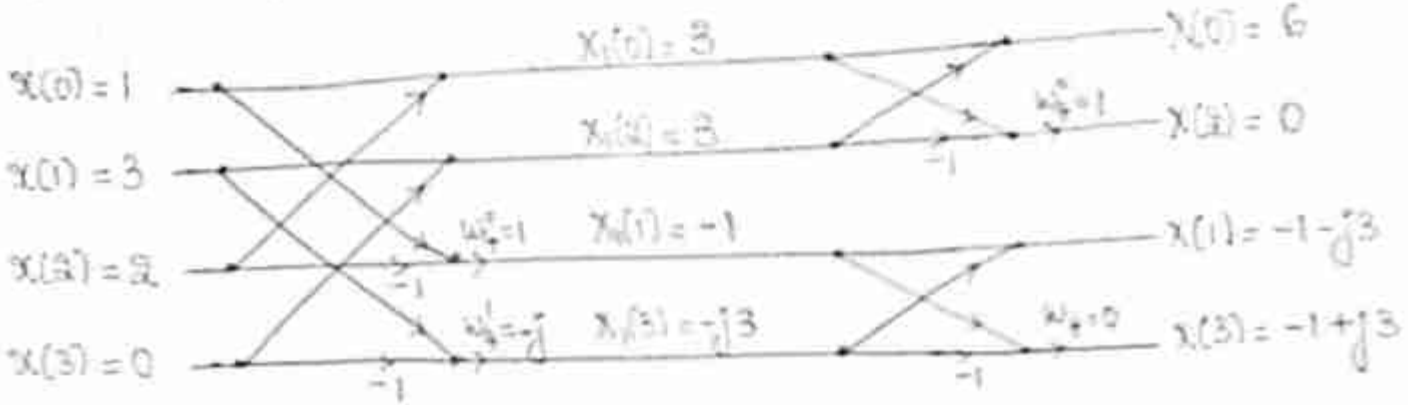
57) For the given sequence $x(n) = \{1, 3, 2\}$ and $h(n) = \{1, 4, 3, 2\}$. Find the Circular convolution using DIF-FFT algorithm.

Sol^n: $N = \max(\text{length}(x(n)), \text{length}(h(n)))$

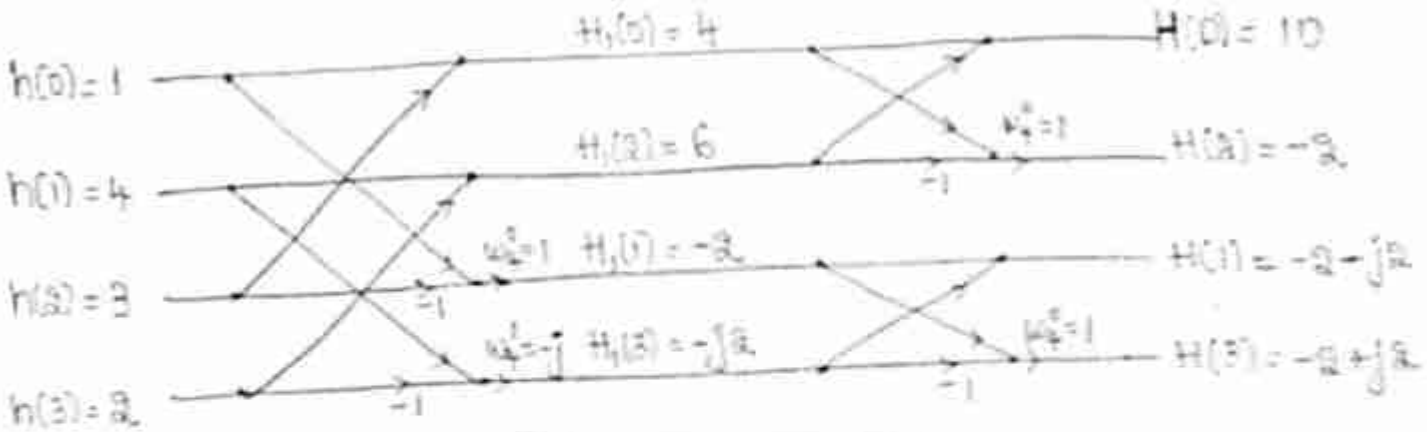
$N = \max(3, 4)$

$\boxed{N = 4}$

$\therefore x(n) = \{1, 3, 2, 0\}$ and $h(n) = \{1, 4, 3, 2\}$



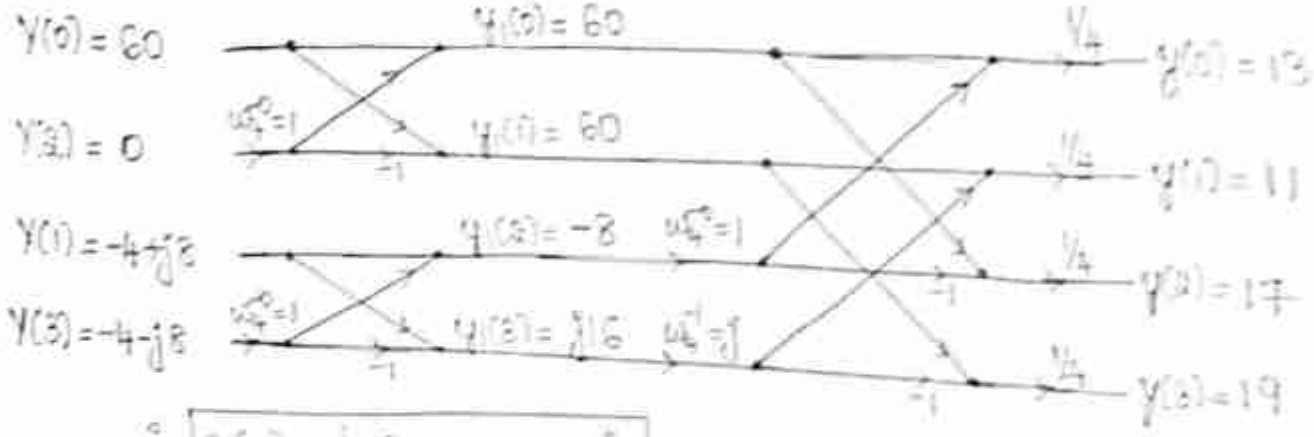
$\therefore \boxed{X(K) = \{6, -1-j3, 0, -1+j3\}}$



$\therefore \boxed{H(K) = \{10, -2-j2, -2, -2+j2\}}$

W.K.T $Y(K) = X(K) \cdot H(K) = \{60, -4+j8, 0, -4-j8\}$

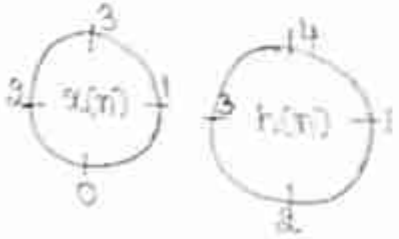
Scanned with CamScanner

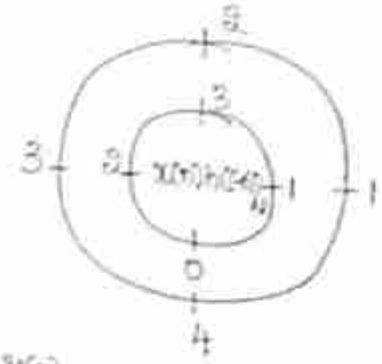


$\therefore \boxed{y(n) = \{13, 11, 17, 19\}}$

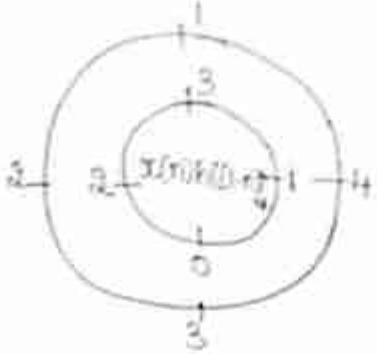
Verification :-

(OR) $x(n) \circledast_4 h(n) = \begin{bmatrix} 1 & 0 & 2 & 3 \\ 3 & 1 & 0 & 2 \\ 2 & 3 & 1 & 0 \\ 0 & 2 & 3 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \\ 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 13 \\ 11 \\ 17 \\ 19 \end{bmatrix}$

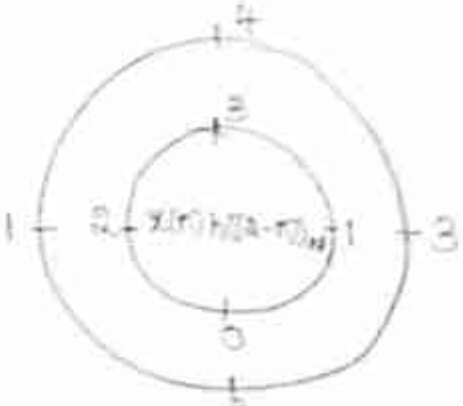




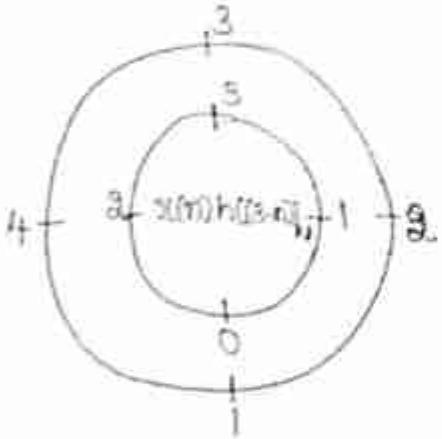
$y(0) = 1+6+6+0 = \underline{13}$



$y(1) = 4+3+4+0 = \underline{11}$



$y(2) = 3+12+2+0$
$= \underline{17}$



$y(4) = 3+9+8+0 = \underline{19}$

$\therefore y(n) = \{13, 11, 17, 19\}$

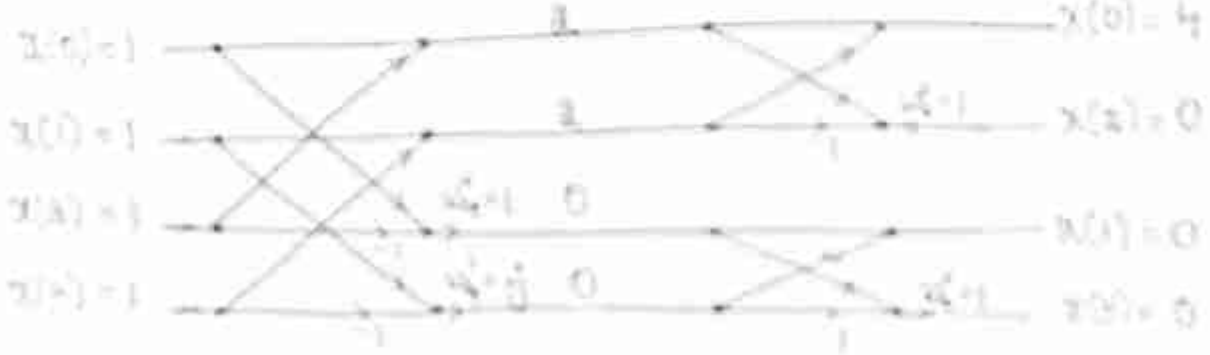
Scanned with CamScanner

5@) For the given sequence $x(n) = \{1, 1, 1, 1\}$ and $h(n) = \{1, 0, 1, 0\}$. Find the circular convolution $y(n) = x(n) \otimes h(n)$ using DIF-FFT radix-2

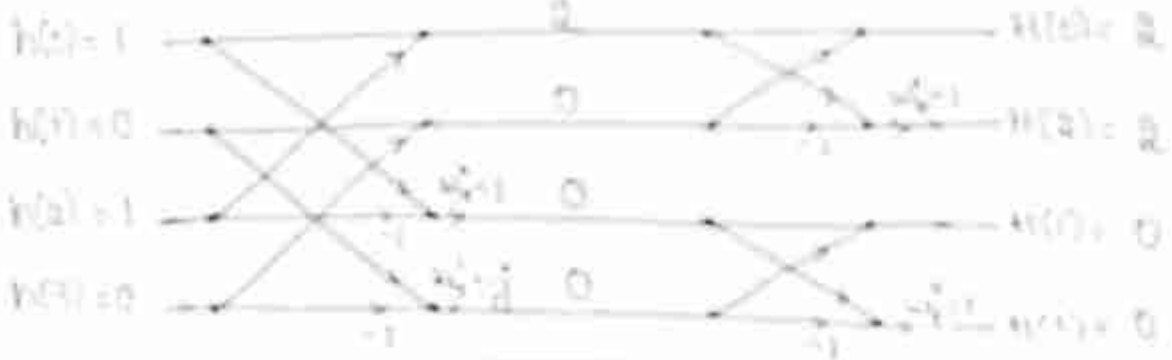
Sol^n: $N = \max(\text{length}(x(n)), \text{length}(h(n)))$

$= \max(4, 4)$

$\boxed{N = 4}$



$\therefore \boxed{x(k) = \{4, 0, 0, 0\}}$



$\therefore \boxed{H(k) = \{2, 0, 2, 0\}}$

W.K.T $Y(k) = X(k)H(k) = \{8, 0, 0, 0\}$



$\therefore \boxed{y(n) = \{2, 2, 2, 2\}}$

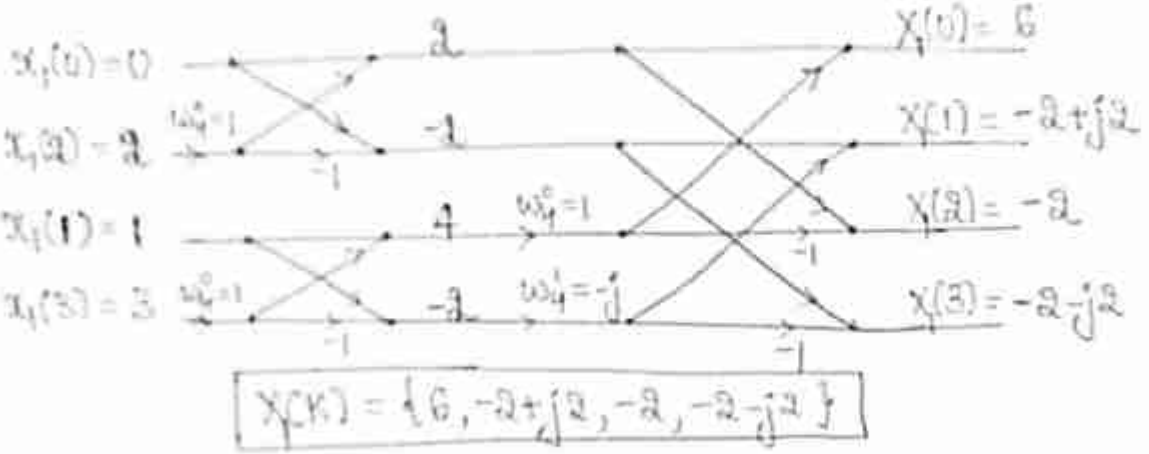
Scanned with CamScanner

5) Compute circular convolution between 2 sequences:

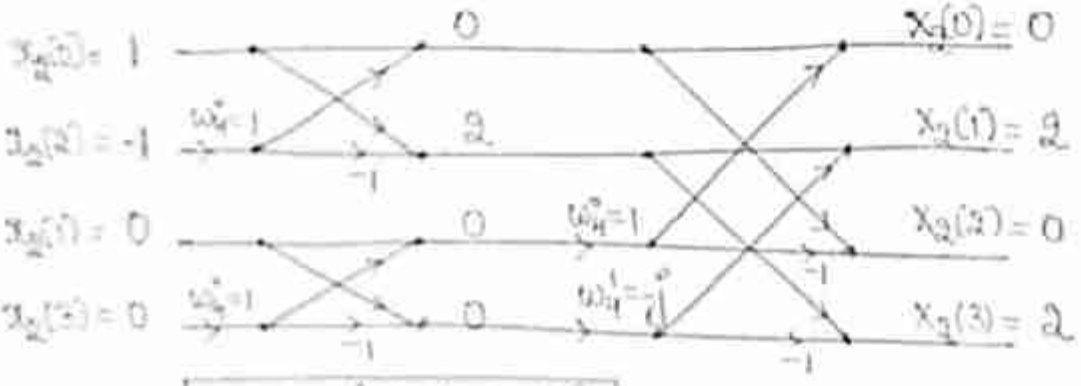
$x_1(n) = \begin{cases} n\,; & 0 \le n \le 3 \\ 0\,; & \text{else} \end{cases}$ and $x_2(n) = \begin{cases} \cos\left(\frac{n\pi}{2}\right)\,; & 0 \le n \le 3 \\ 0\,; & \text{else} \end{cases}$

Using DIT-FFT.

Soln: $x_1(n) = \{0, 1, 2, 3\}$ and $x_2(n) = \{1, 0, -1, 0\}$; Given $N = 4$.

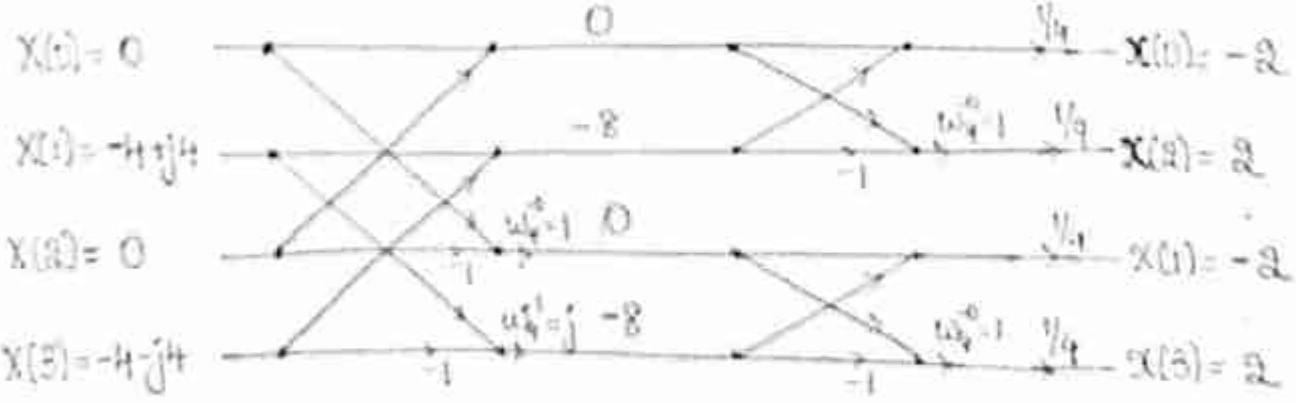


$\boxed{X_1(K) = \{6, -2+j2, -2, -2-j2\}}$



$\boxed{X_2(K) = \{0, 2, 0, 2\}}$

W.K.T. $X(K) = X_1(K)\,X_2(K) = \{0, -4+j4, 0, -4-j4\}$



$\therefore \boxed{x(n) = \{-2, -2, 2, 2\}}$

Scanned with CamScanner

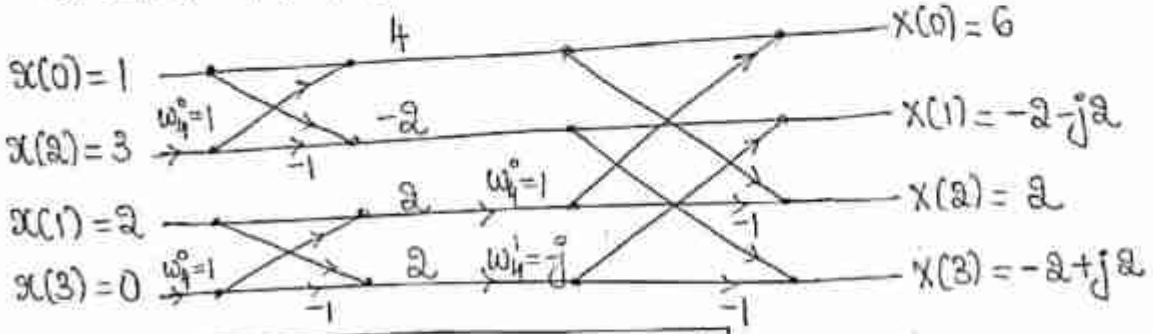
60) If $x(n) = \{1, 2, 3\}$ and $h(n) = \{1, 2\}$. Find the linear convolution b/w two sequence using DIT-FFT.

Sol^n: N = length (x(n)) + length (h(n)) − 1

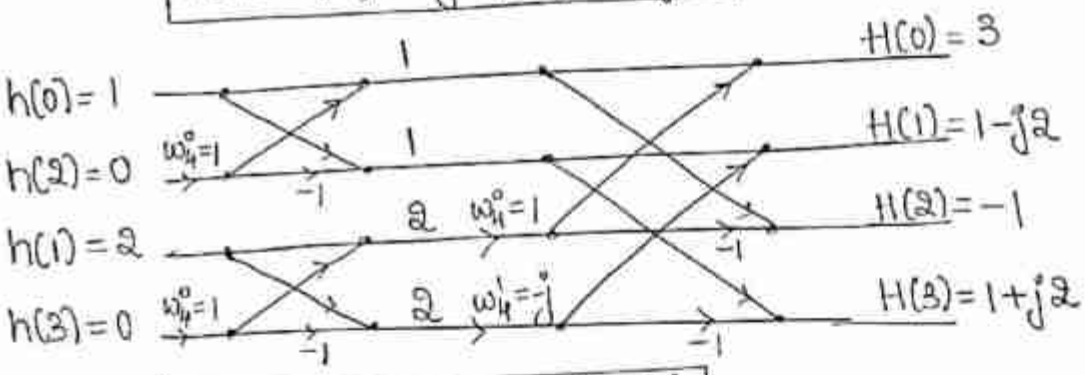
$N = L + M - 1$

$N = 3 + 2 - 1 \Rightarrow \boxed{N = 4}$

$\therefore x(n) = \{1, 2, 3, 0\}$ and $h(n) = \{1, 2, 0, 0\}$

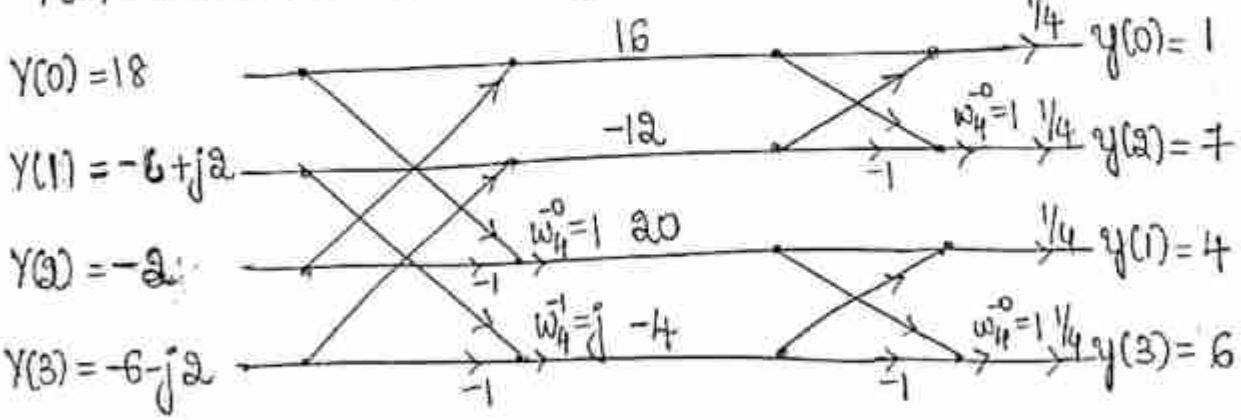


$\boxed{X(K) = \{6, -2-j2, 2, -2+j2\}}$



$\boxed{H(K) = \{3, 1-j2, -1, 1+j2\}}$

$Y(K) = X(K)\,H(K) = \{18, -6+j2, -2, -6-j2\}$



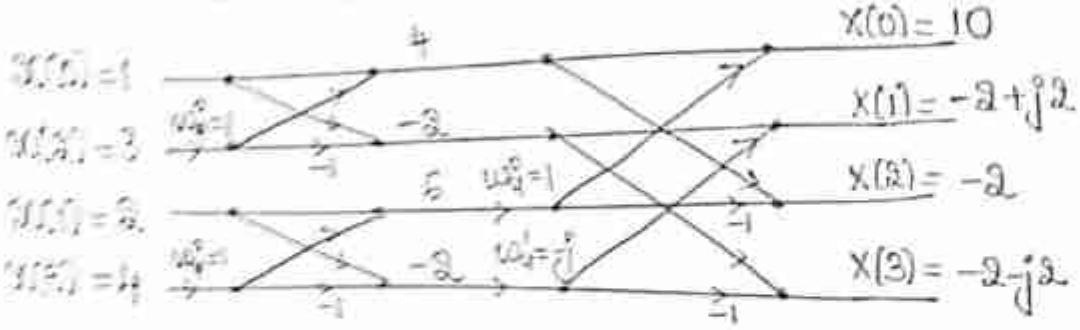
$\therefore \boxed{y(n) = \{1, 4, 7, 6\}}$

Scanned with CamScanner

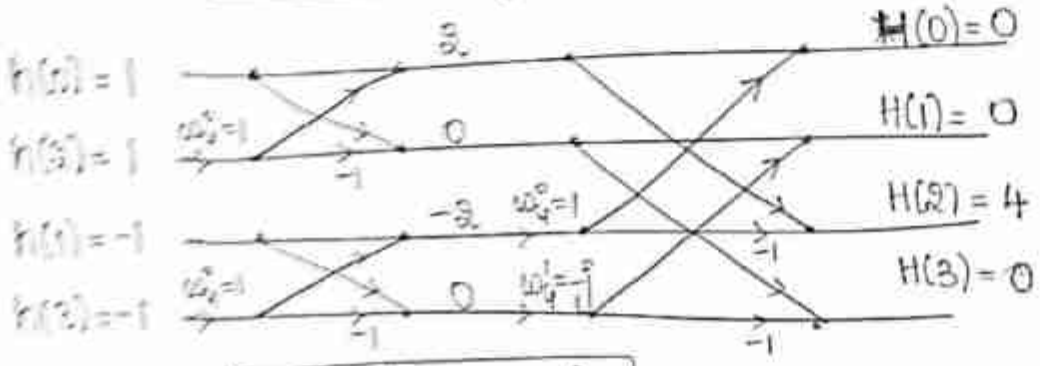
8) a) If $x(n) = n+1$ and $h(n) = (-1)^n$. Find Circular Convolution.
b) Use the two sequence for N=4 using DIT-FFT.

Sol:- Given N = 4,

$x(n) = \{1, 2, 3, 4\}$; $h(n) = \{1, -1, 1, -1\}$.

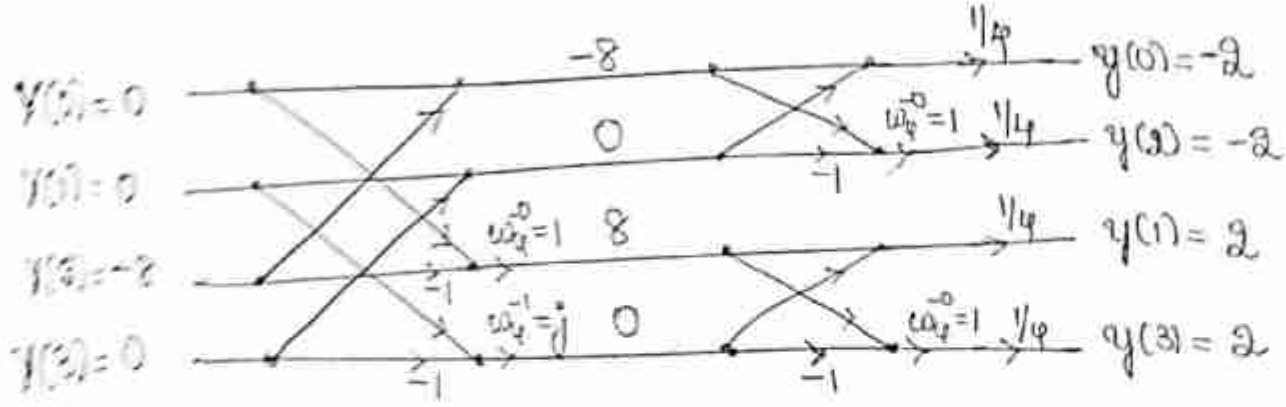


$\therefore X(K) = \{10, -2+j2, -2, -2-j2\}$



$\therefore H(K) = \{0, 0, 4, 0\}$

$Y(K) = X(K)\,H(K) = \{0, 0, -8, 0\}$

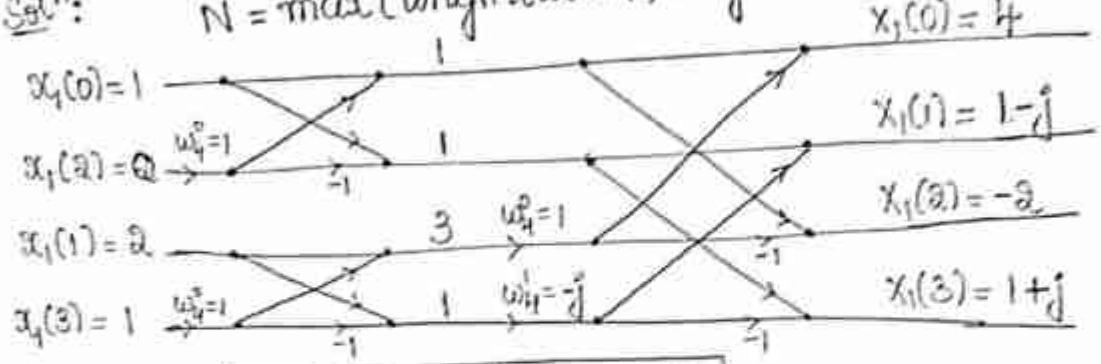


$\therefore y(n) = \{-2, 2, -2, 2\}$

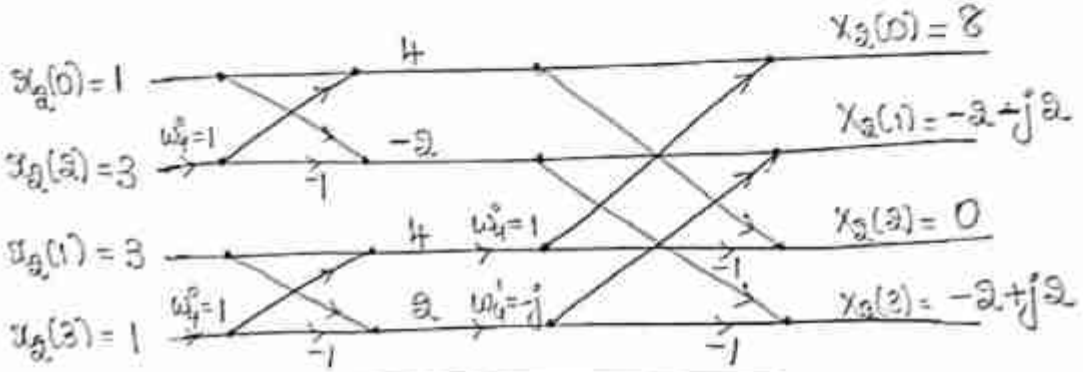
Scanned with CamScanner

(32) If $x_1(n) = \{1, 2, 0, 1\}$ and $x_2(n) = \{1, 3, 3, 1\}$, obtain $x_1(n)$ (*) $x_2(n)$ by using DIT-FFT algorithm.

Soln: $N = \max(\text{length}(x(n)), \text{length}(h(n))) = \max(4, 4) = 4$

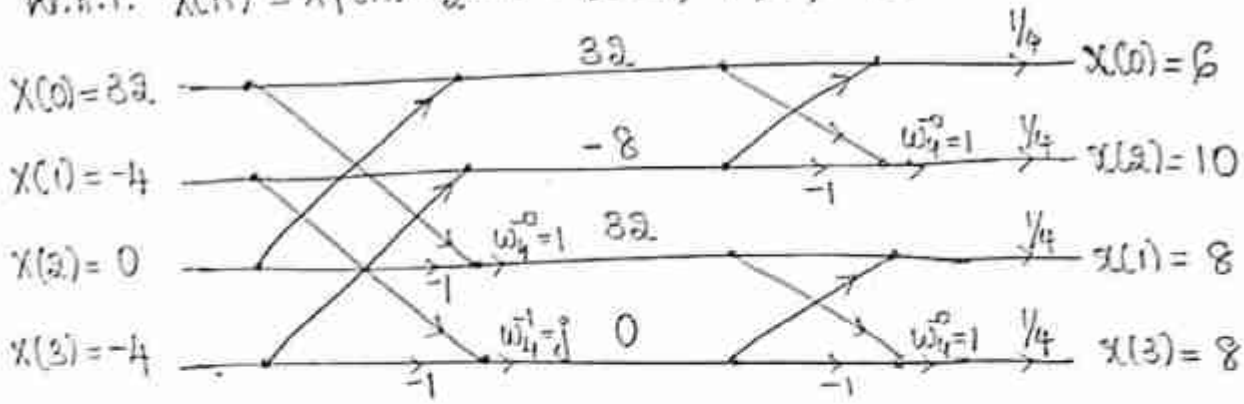


$\therefore X_1(K) = \{4, 1-j, -2, 1+j\}$



$\therefore X_2(K) = \{8, -2-j2, 0, -2+j2\}$

W.K.T. $X(K) = X_1(K) \cdot X_2(K) = \{32, -4, 0, -4\}$



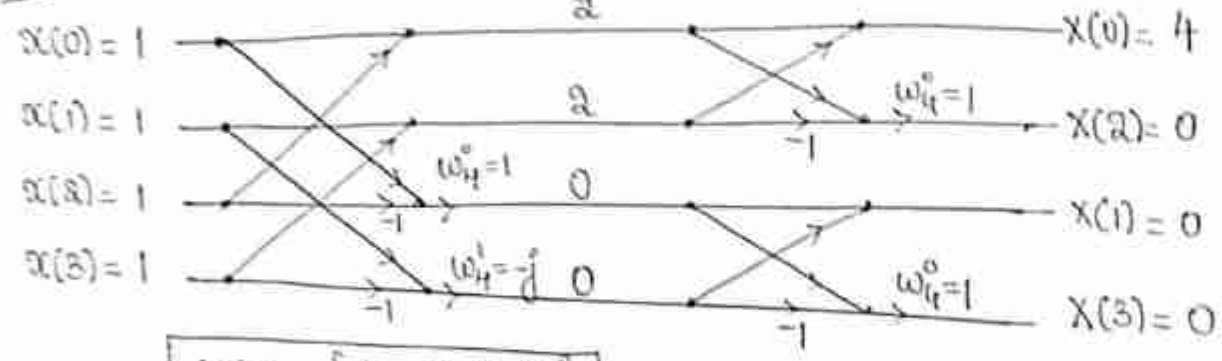
$\therefore x(n) = \{6, 8, 10, 8\}$

Scanned with CamScanner

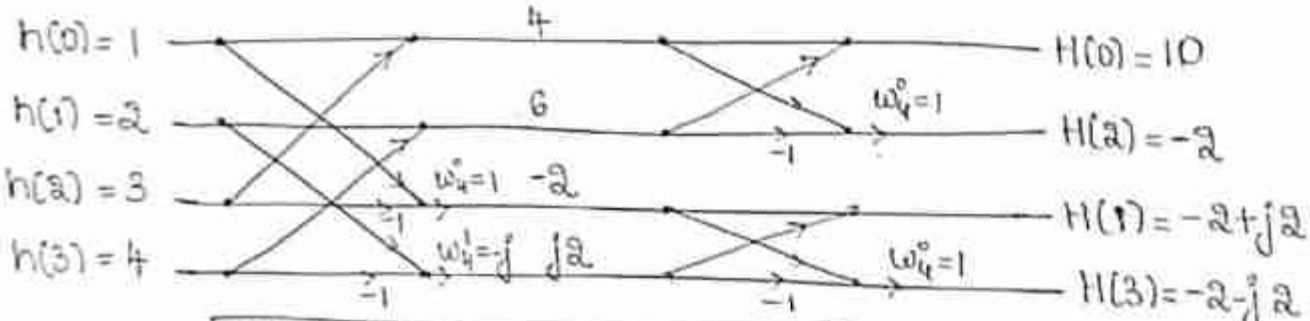
23) Find Circular convolution of $x(n) = \{1,1,1,1\}$ with $h(n) = \{1,2,3,4\}$ using radix-2 decimation in frequency post fourier transform for DFTs and radix-2 decimation in time post fourier transform to find IDFT.

Soln:

$$N = \max(4,4) = 4$$

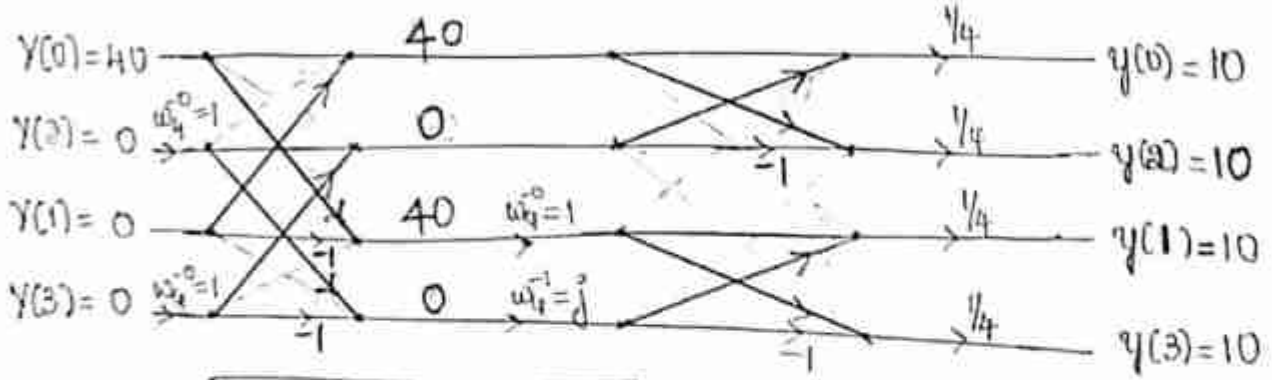


$$X(K) = \{4, 0, 0, 0\}$$



$$H(K) = \{10, -2+j2, -2, -2-j2\}$$

W.K.T. $Y(K) = X(K)\,H(K) = \{40, 0, 0, 0\}$



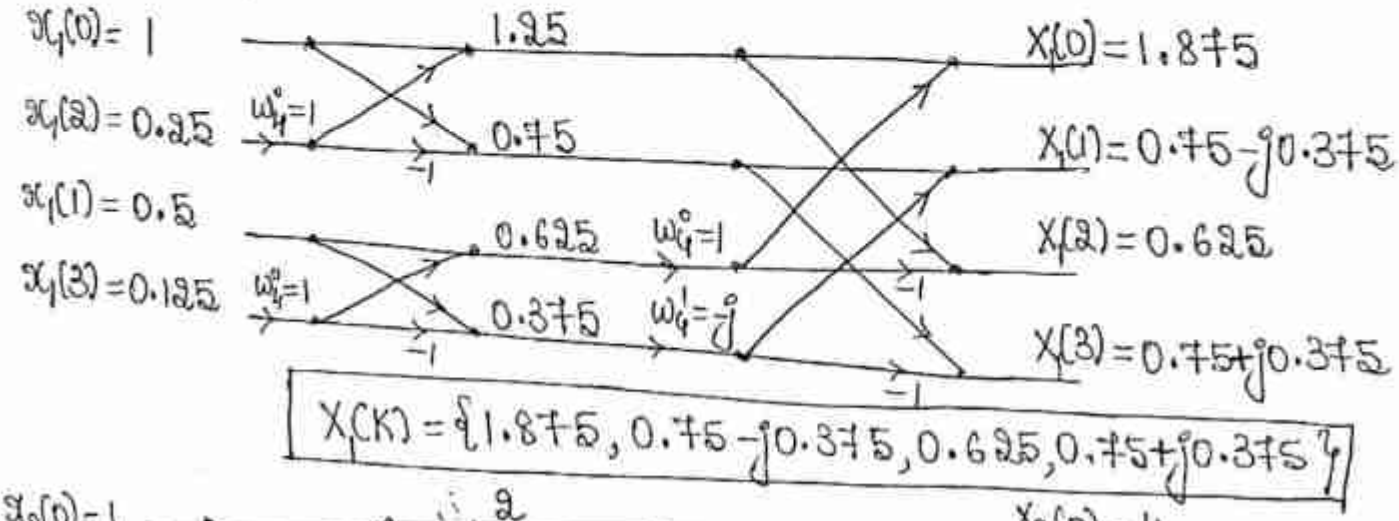
$$\therefore y(n) = \{10, 10, 10, 10\}$$

Scanned with CamScanner

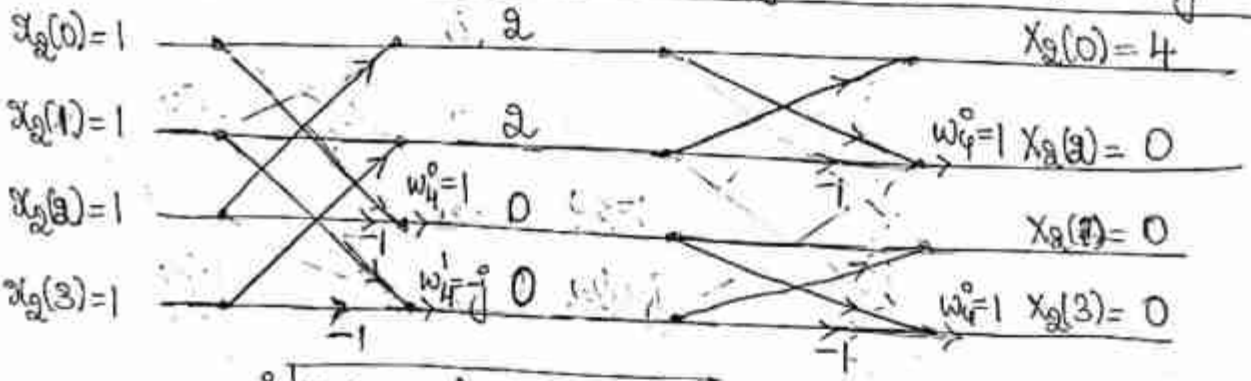
64) Consider $x_1(n) = \{1, 0.5, 0.25, 0.125\}$, $x_2(n) = \{1, 1, 1, 1\}$. Determine $X_1(K)$ using decimation-in time FFT algorithm and $X_2(K)$ using decimation in frequency FFT algorithm. hence find Circular Convolution of $x_1(n)$ (N) $x_2(n)$.

Soln:

Given N = 4

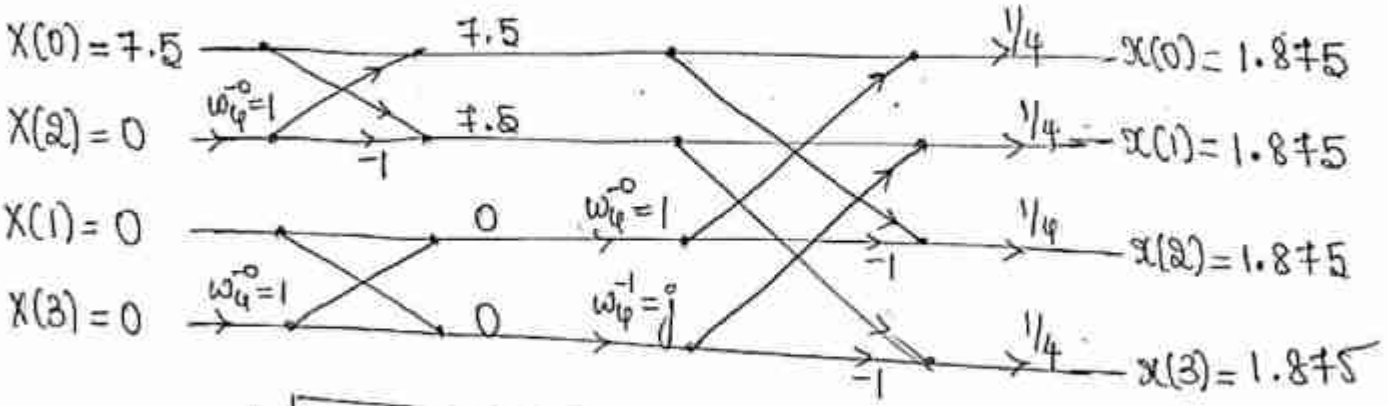


$$X_1(K) = \{1.875, 0.75 - j0.375, 0.625, 0.75 + j0.375\}$$



$$\therefore X_2(K) = \{4, 0, 0, 0\}$$

W.K.T. $X(K) = X_1(K) \cdot X_2(K) = \{7.5, 0, 0, 0\}$



$$\therefore x(n) = \{1.875, 1.875, 1.875, 1.875\}$$

Scanned with CamScanner

* $x(t) = \frac{(t+1)}{8}$ for $0 \le t \le \frac{8}{5}$ Sec. obtain X(K) using 8-point DIT-FFT algorithm. Assume x(t) is sampled at 5 samples/sec. Draw the magnitude spectrum.

Soln: Given $x(t) = \frac{(t+1)}{8}$ for $0 \le t \le \frac{8}{5}$

$= \frac{1}{8}(t+1)$ for $0 \le t \le 1.6$

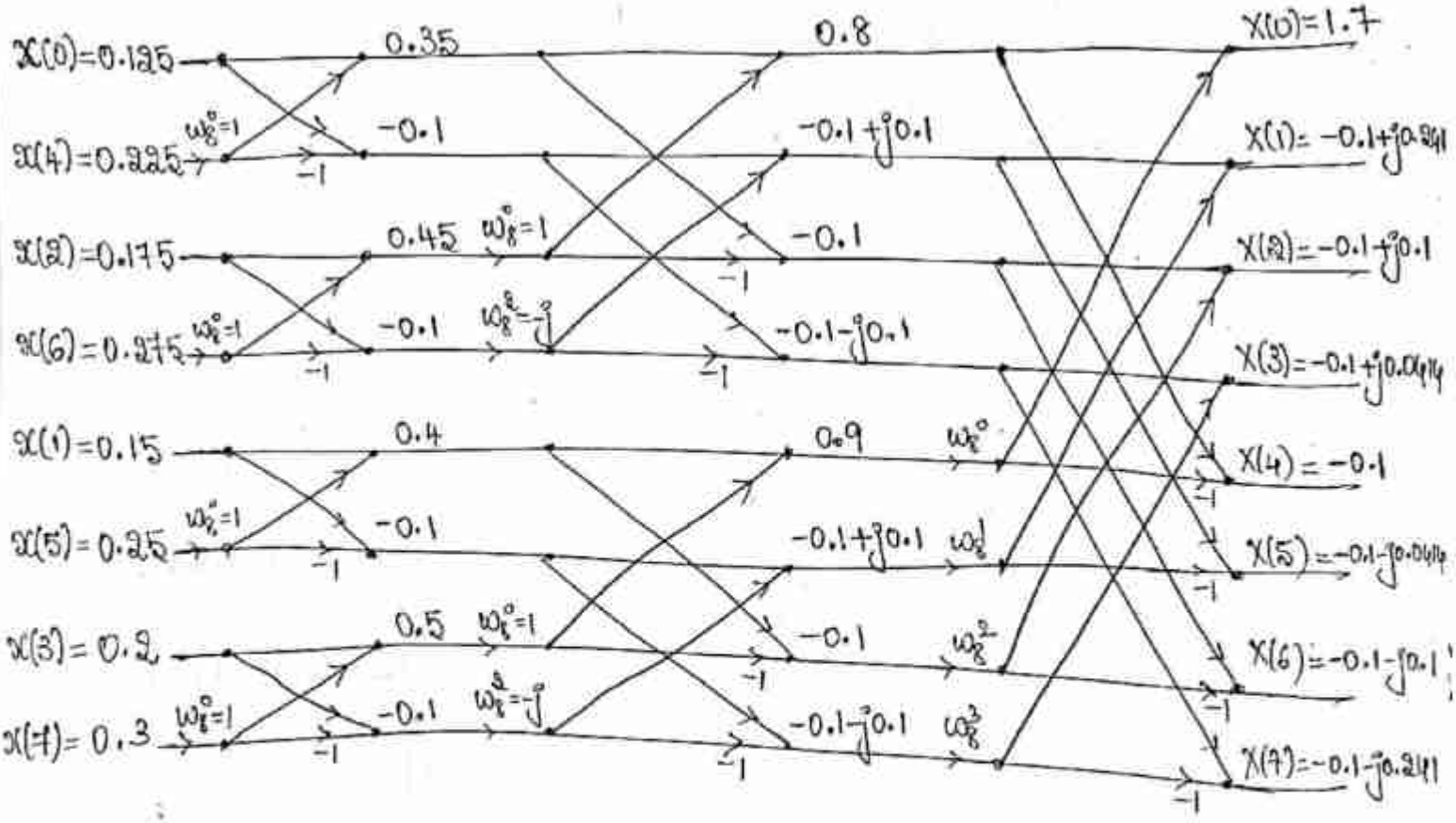
Here $F_S = 5Hz$, hence $T_S = \frac{1}{F_S} = \frac{1}{5} = 0.2$ sec

The sampled version of x(t) will be,

$x(n) = x(t)|_{t=nT_S}$; $n = 0, 1, \ldots 7 = \frac{nT_S + 1}{8} = \frac{(n \times 0.2) + 1}{8}$

$x(n) = \{x(t=0), x(t=0.2), x(t=0.4), x(0.6), x(0.8), x(1), x(1.2), x(1.4)\}$

$x(n) = \{0.125, 0.15, 0.175, 0.2, 0.225, 0.25, 0.275, 0.3\}$



Scanned with CamScanner

65) A designer has a number of 8-point FFT chips. Show explicitly how he should interconnect four such chips in order to compute a 32-point DFT.

C_1	C_2	C_3	C_4	C_1	C_2	C_3	C_4
0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

Soln:

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

Here $N = 32$,

$$X(K) = \sum_{n=0}^{31} x(n) W_{32}^{Kn}$$

For,

Chip 1 : 0 4 8 12 16 20 24 28 $\longrightarrow 4n$

Chip 2 : 1 5 9 13 17 21 25 29 $\longrightarrow 4n+1$

Chip 3 : 2 6 10 14 18 22 26 30 $\longrightarrow 4n+2$

Chip 4 : 3 7 11 15 19 23 27 31 $\longrightarrow 4n+3$

$$X(K) = \underbrace{\sum_{n=0}^{28} x(n) W_N^{Kn}}_{4n} + \underbrace{\sum_{n=1}^{29} x(n) W_N^{Kn}}_{4n+1} + \underbrace{\sum_{n=2}^{30} x(n) W_N^{Kn}}_{4n+2} + \underbrace{\sum_{n=3}^{31} x(n) W_N^{Kn}}_{4n+3}$$

$$= \sum_{n=0}^{7} x(4n) W_N^{K(4n)} + \sum_{n=0}^{7} x(4n+1) W_N^{K(4n+1)} + \sum_{n=0}^{7} x(4+2) W_N^{K(4n+2)} + \sum_{n=0}^{7} x(4n+3) W_N^{K(4n+3)}$$

$$X(K) = \sum_{n=0}^{7} f_1(n) W_{N/4}^{Kn} + W_N^{K} \sum_{n=0}^{7} f_2(n) W_{N/4}^{Kn} + W_N^{2K} \sum_{n=0}^{7} f_3(n) W_{N/4}^{Kn} + W_N^{3K} \sum_{n=0}^{7} f_4(n) W_{N/4}^{Kn}$$

$$X(K) = F_1(K) + W_N^K F_2(K) + W_N^{2K} F_3(K) + W_N^{3K} F_4(K) \; ; 0 \le K \le \frac{N}{4} - 1$$

Scanned with CamScanner

66) A designer has a number of 8-point FFT chips. Show explicitly how he should interconnect 3-chips in order to compute 24-point DFT.

C_1	C_2	C_3	C_1	C_2	C_3	C_1	C_2	C_3
0	1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16	17
18	19	20	21	22	23			

Soln:

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn}$$

Here $N = 24$.

$$X(K) = \sum_{n=0}^{23} x(n) W_{24}^{Kn}$$

For,

Chip 1 : 0 3 6 9 12 15 18 21 $\longrightarrow 3n$

Chip 2 : 1 4 7 10 13 16 19 22 $\longrightarrow 3n+1$

Chip 3 : 2 5 8 11 14 17 20 23 $\longrightarrow 3n+2$

$$X(K) = \underbrace{\sum_{n=0}^{21} x(n) W_N^{Kn}}_{3n} + \underbrace{\sum_{n=1}^{22} x(n) W_N^{Kn}}_{3n+1} + \underbrace{\sum_{n=2}^{23} x(n) W_N^{Kn}}_{3n+2}$$

$$= \sum_{n=0}^{7} x(3n) W_N^{K(3n)} + \sum_{n=0}^{7} x(3n+1) W_N^{K(3n+1)} + \sum_{n=0}^{7} x(3n+2) W_N^{K(3n+2)}$$

$$= \sum_{n=0}^{7} f_1(n) W_{N/3}^{Kn} + W_N^{K} \sum_{n=0}^{7} f_2(n) W_{N/3}^{Kn} + W_N^{2K} \sum_{n=0}^{7} f_3(n) W_{N/3}^{Kn}$$

$$X(K) = F_1(K) + W_N^K F_2(K) + W_N^{2K} F_3(K) \quad ; 0 \le K \le \frac{N}{3} - 1$$

67) A designer has a number of 4-point FFT chips. Show explicitly how he should interconnect 4-chips in order to complete 16-point DFT.

C_1	C_2	C_3	C_4	C_1	C_2	C_3	C_4
0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15

Soln:-

$$X(K) = \sum_{n=0}^{15} x(n) W_{16}^{Kn}$$

Scanned with CamScanner

For,

Chip 1 : 0 4 8 12 $\longrightarrow 4n$

Chip 2 : 1 5 9 13 $\longrightarrow 4n+1$

Chip 3 : 2 6 10 14 $\longrightarrow 4n+2$

Chip 4 : 3 7 11 15 $\longrightarrow 4n+3$

$$X(K) = \underbrace{\sum_{n=0}^{12} x(n) W_N^{Kn}}_{4n} + \underbrace{\sum_{n=1}^{13} x(n) W_N^{Kn}}_{4n+1} + \underbrace{\sum_{n=2}^{14} x(n) W_N^{Kn}}_{4n+2} + \underbrace{\sum_{n=3}^{15} x(n) W_N^{Kn}}_{4n+3}$$

$$= \sum_{n=0}^{3} x(4n) W_N^{K(4n)} + \sum_{n=0}^{3} x(4n+1) W_N^{K(4n+1)} + \sum_{n=0}^{3} x(4n+2) W_N^{K(4n+2)} + \sum_{n=0}^{3} x(4n+3) W_N^{K(4n+3)}$$

$$= \sum_{n=0}^{3} f_1(n) W_{N/4}^{Kn} + W_N^K \sum_{n=0}^{3} f_2(n) W_{N/4}^{Kn} + W_N^{2K} \sum_{n=0}^{3} f_3(n) W_{N/4}^{Kn} + W_N^{3K} \sum_{n=0}^{3} f_4(n) W_{N/4}^{Kn}$$

$$X(K) = F_1(K) + W_N^K F_2(K) + W_N^{2K} F_3(K) + W_N^{3K} F_4(K) \quad ; \; 0 \le K \le \frac{N}{4} - 1$$

68) It is required to take DFT of a data stream of length 8192. However, the analyzer has only a fixed hardware implementation for a 2048 point DFT, Assuming other storage is available along with ways of adding and multiplying, how could the desired 8192-point transform be obtained?

Soln:

$$\text{No. of Chips} = \frac{\text{DFT points}}{\text{FFT points}} = \frac{8192}{2048} = 4$$

Scanned with CamScanner

Decimate the i/p sequence as shown below,

C_1	C_2	C_3	C_4	C_1	C_2	C_3	C_4
$x(0)$	$x(1)$	$x(2)$	$x(3)$	$x(4)$	$x(5)$	$x(6)$	$x(7)$
8	9	10	11	12	13	14	15
⋮				⋮			
8184	8185	8186	8187	8188	8189	8190	8191

For

Chip 1 : 0 4 8 12 ..... $8188 \rightarrow 4n$

Chip 2 : 1 5 9 13 ..... $8189 \rightarrow 4n+1$

Chip 3 : 2 6 10 14 ..... $8190 \rightarrow 4n+2$

Chip 4 : 3 7 11 15 ..... $8191 \rightarrow 4n+3$

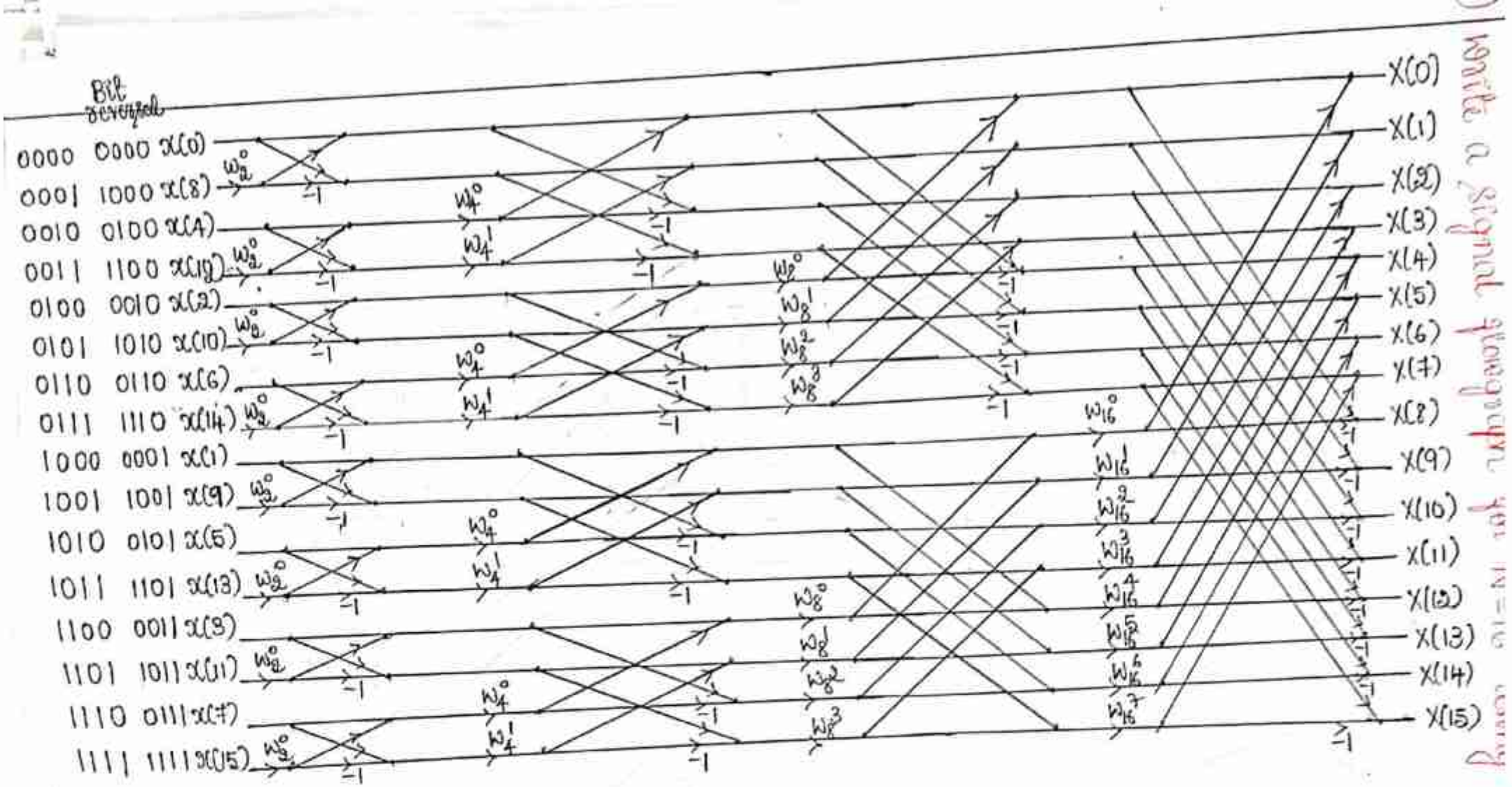
$$X(K) = \underbrace{\sum_{n=0}^{8088} x(n) W_N^{Kn}}_{4n} + \underbrace{\sum_{n=0}^{8089} x(n) W_N^{Kn}}_{4n+1} + \underbrace{\sum_{n=0}^{8090} x(n) W_N^{Kn}}_{4n+2} + \underbrace{\sum_{n=0}^{8091} x(n) W_N^{Kn}}_{4n+3}$$

$$= \sum_{n=0}^{2047} x(4n) W_N^{K(4n)} + \sum_{n=0}^{2047} x(4n+1) W_N^{K(4n+1)} + \sum_{n=0}^{2047} x(4n+2) W_N^{K(4n+2)} + \sum_{n=0}^{2047} x(4n+3) W_N^{K(4n+3)}$$

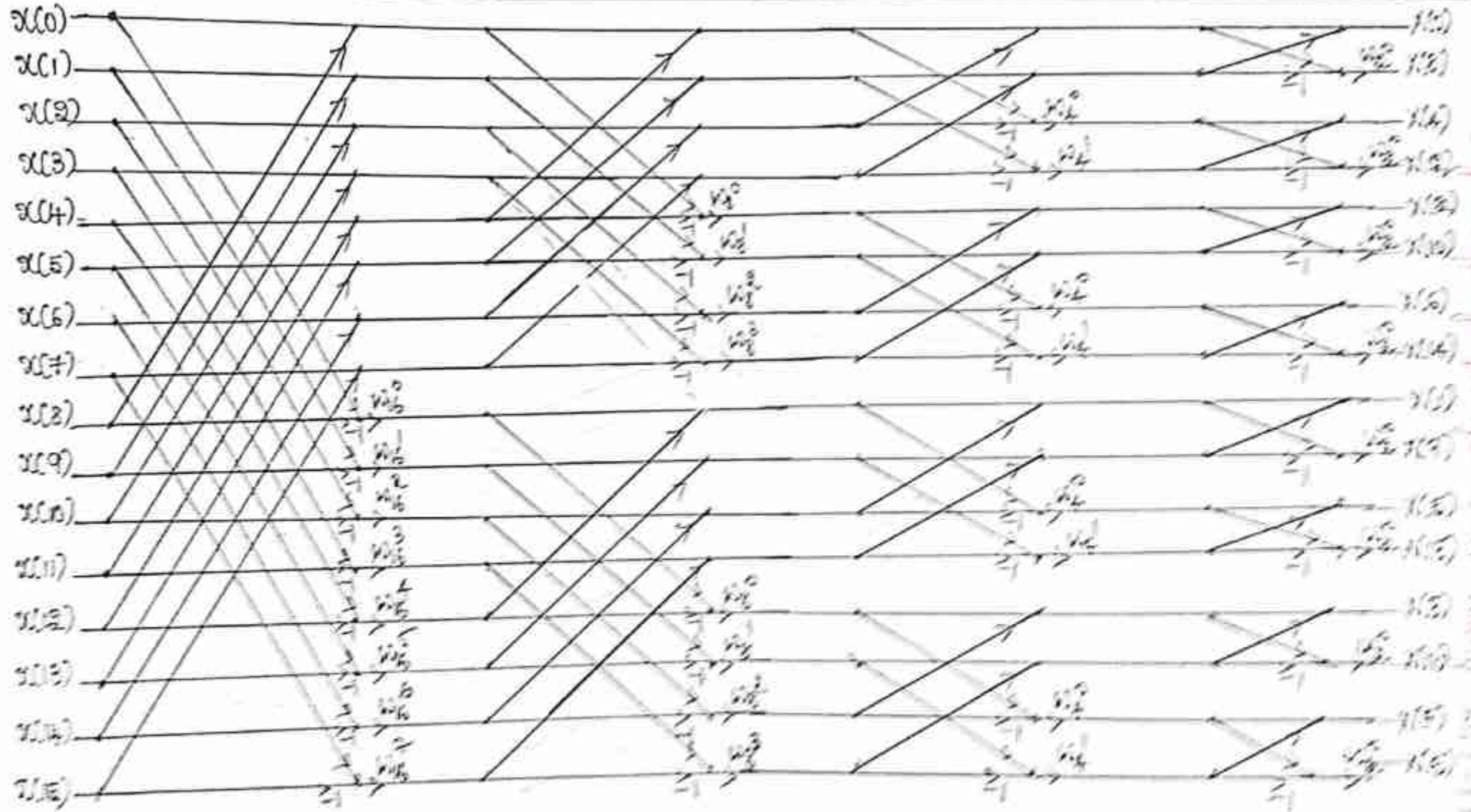
$$= \sum_{n=0}^{2047} f_1(n) W_{N/4}^{Kn} + W_N^{K} \sum_{n=0}^{2047} f_2(n) W_{N/4}^{Kn} + W_N^{2K} \sum_{n=0}^{2047} f_3(n) W_{N/4}^{Kn} + W_N^{3K} \sum_{n=0}^{2047} f_4(n) W_{N/4}^{Kn}$$

$$X(K) = F_1(K) + W_N^K F_2(K) + W_N^{2K} F_3(K) + W_N^{3K} F_4(K) \; ; \; 0 \le K \le \frac{N}{4} - 1$$

Scanned with CamScanner



Scanned with CamScanner



Scanned with CamScanner

Goertzel Algorithm :-

W.K.T the FFT algorithms take N-data points of the sequence $x(n)$ and compute N-points of DFT. This requires $\frac{N}{2}\log_2 N$ number of complex multiplications and $N\log_2 N$ complex additions. In some applications, the DFT is to be computed only at selected values of 'K' (i.e., frequencies). When these selected values are less than $\log_2 N$, then direct computation of DFT becomes more efficient than FFT. This direct computation of DFT for selected values of frequencies can be realized through linear filtering of $x(n)$. Such linear filtering for computation of DFT can be implemented using goertzel algorithm.

From the equation of the DFT,

$$X(K) = \sum_{m=0}^{N-1} x(m) W_N^{Km} \longrightarrow (1)$$

If we multiply R.H.S of above equation by W_N^{-KN}, the value of $X(K)$ will be unaffected since $W_N^{-KN} = 1$,

i.e., $$X(K) = \sum_{m=0}^{N-1} x(m) W_N^{Km} W_N^{-KN}$$

$$= \sum_{m=0}^{N-1} x(m) W_N^{-K(N-m)} \quad ; 0 \le K \le N-1 \longrightarrow (2)$$

The Eq(2) is in the form of convolution sum. If we define,

$$y_K(n) = \sum_{m=0}^{N-1} x(m) W_N^{-K(N-m)} \longrightarrow (3)$$

then $y_K(n)$ is the convolution of the finite-duration input sequence $x(n)$ of length 'N' with a filter that has an impulse response $h_K(n) = W_N^{-Kn} u(n) \longrightarrow (4)$

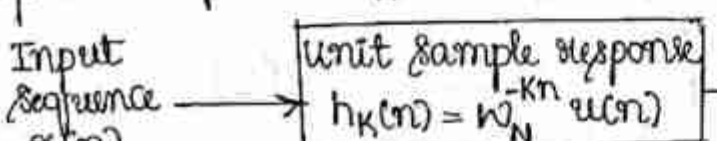


fig: The LTI System with unit sample response $h_K(n) = W_N^{Kn} u(n)$ and input $x(n)$

Scanned with CamScanner

Here 'K' is fixed (constant) for this system. W.K.T linear convolution of $x(n)$ and $h_K(n)$ gives the output sequence $y_K(n)$ of LTI system i.e.,

$$y_K(n) = \sum_{m=-\infty}^{\infty} x(m)\, h_K(n-m) \longrightarrow (5)$$

Substitute for $h_K(n)$ in above eqn from eq(4),

$$y_K(n) = \sum_{m=-\infty}^{\infty} x(m)\, W_N^{-K(n-m)}\, u(n-m)$$

$$= \sum_{m=0}^{N-1} x(m)\, W_N^{-K(n-m)} \longrightarrow (6)$$

, Since $u(n-m) = \begin{cases} 1 ; n \geq m \text{ (or) } m \leq n \\ 0 ; \text{Else} \end{cases}$

Now, Consider the output of the LTI system $y_K(n)$ at $n=N$. Then eq(6) becomes,

$$y_K(n)\Big|_{n=N} = \sum_{m=0}^{N-1} x(m)\, W_N^{-K(N-m)} \longrightarrow (7)$$

On comparing eq(7) with eq(2), we find that

$$\boxed{X(K) = y_K(n)\Big|_{n=N}}$$

This relation shows that the DFT $X(K)$ can be obtained as the output of LTI system at $n=N$. Note that such system can give $X(K)$ only at one value of 'K'. Such parallel bank of systems (also called as resonators (or) filters) can be used to get $X(K)$ at different values of 'K'.

The filter with impulse response $h_K(n)$ has the system function,

$$H_K(z) = \sum_{n=-\infty}^{\infty} h_K(n)\, z^{-n}$$

$$= \sum_{n=0}^{\infty} W_N^{-Kn}\, z^{-n} = \sum_{n=0}^{\infty} \left(W_N^{-K} z^{-1}\right)^n$$

$$H_K(z) = \frac{1}{1 - W_N^{-K} z^{-1}} \longrightarrow (8)$$

Scanned with CamScanner

The filter has a pole on the unit circle at the frequency $\omega_K = \frac{2\pi}{N} K$.

Thus, the entire DFT can be completed by passing the block of input data into a parallel bank of N single pole filters where each filter has a pole at the corresponding frequency of the DFT.

From Eq(8),

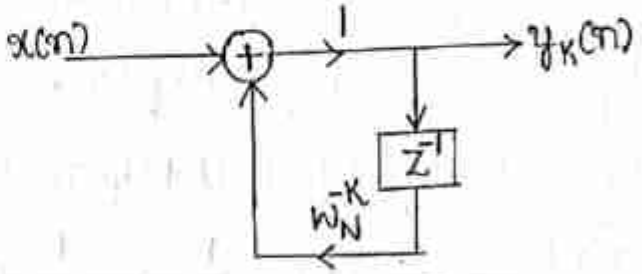
$$\frac{Y_K(z)}{X(z)} = \frac{1}{1 - W_N^{-K} z^{-1}}$$

$$Y_K(z) - W_N^{-K} z^{-1} Y_K(z) = X(z)$$

By inverse Z-transform,

$$y_K(n) - W_N^{-K} y_K(n-1) = x(n)$$

$$\langle y_K(n) = x(n) + W_N^{-K} y_K(n-1) \rangle$$



Direct form - I

The desired output is $y_K(n)$; $K = 0, 1, \ldots (N-1)$. To perform this computation, the phase factors W_N^{-K} are computed once and stored.

The complex multiplications and additions can be avoided by combining the pairs of filters possessing complex conjugate poles. This leads to a pole filter with system function.

$$H_K(z) = \frac{Y_K(z)}{X(z)} = \frac{(1 - W_N^{K} z^{-1})}{(1 - W_N^{-K} z^{-1})(1 - W_N^{K} z^{-1})}$$

$$= \frac{1 - W_N^{K} z^{-1}}{1 - W_N^{K} z^{-1} - W_N^{-K} z^{-1} + W_N^{-K} W_N^{K} z^{-1} z^{-1}}$$

$$= \frac{1 - W_N^{K} z^{-1}}{1 - W_N^{K} z^{-1} - W_N^{-K} z^{-1} + z^{-2}}$$

$$= \frac{1 - W_N^{K} z^{-1}}{1 - (W_N^{K} + W_N^{-K}) z^{-1} + z^{-2}}$$

Scanned with CamScanner

$$\frac{Y_K(z)}{X(z)} = \frac{1 - W_N^K z^{-1}}{1 - \left(e^{\frac{-j2\pi K}{N}} + e^{\frac{j2\pi K}{N}}\right) z^{-1} + z^{-2}}$$

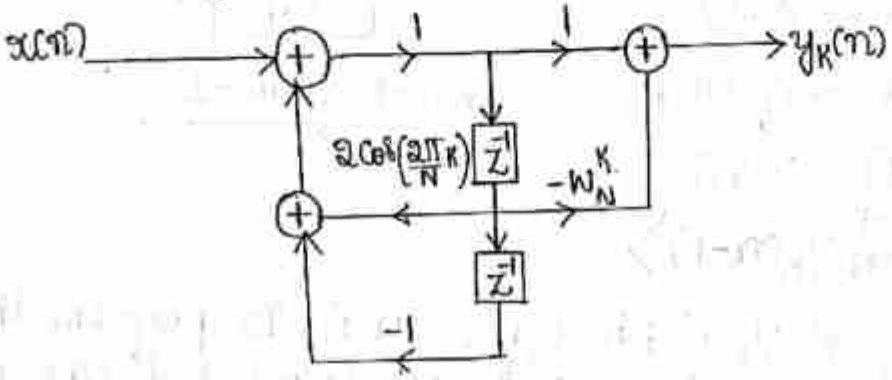
$$\frac{Y_K(z)}{X(z)} = \frac{1 - W_N^K z^{-1}}{1 - 2\cos\left(\frac{2\pi K}{N}\right) z^{-1} + z^{-2}}$$

$$\cos\theta = \frac{e^{j\theta} + e^{-j\theta}}{2}$$

$$Y_K(z) - 2\cos\left(\frac{2\pi K}{N}\right) z^{-1} Y_K(z) + Y_K(z) z^{-2} = X(z) - W_N^K z^{-1} X(z)$$

By inverse Z-transform,

$$y_K(n) - 2\cos\left(\frac{2\pi K}{N}\right) y_K(n-1) + y_K(n-2) = x(n) - W_N^K x(n-1)$$



In the direct Computation of DFT, 4N real multiplication and (4N-2) real additions are involved to Compute X(K) for a particular value of 'K'.
In Case of Goertzel algorithm, Computation is proportional to NM where X(K) is to be Evaluated for any 'M' Values of 'K'. In DIT and DIF algorithm, number of Computation is $N \log_2 N$. Thus, when $M < \log_2 N$, Goertzel algorithm Could be more Efficient than the FFT approach.

Advantages of Goertzel Algorithm :—

1) It is the best method to find DFT when $K \leq \log_2 N$.
Where 'K' is particular DFT point to be Evaluated and 'N' is length of the Sequence (or) number of DFT points to be Evaluated.

Scanned with CamScanner

2) Compare the direct Computation method, Computational Comp-lexity is less in goertzel algorithm.

$4N+2N^2$; Real multiplications.

$4N^2+4N$; Real additions

ii) Chirp-Z transform Algorithm :-

Chirp is a signal in which the frequency increases (or) decreases with time. The chirp-Z transform algorithm is used for the computation of DFT only of selected frequency. It is used to linear filtering approach for Computation. It is used to linear filtering approach for Computation. Consider, the Z-transform of $x(n)$ of set of points $\{z_k\}$. It is given by,

$$X(z_k) = \sum_{n=0}^{N-1} x(n) z_k^{-n} \; ; 0 \le k \le (N-1) \longrightarrow (1)$$

[Above eqn can be written as DFT eqn if $z_k = e^{\frac{j2\pi}{N}k} = W_N^{-k}$]

Let the Z-transform be Evaluated across the Circle of radius 'r' at points $w_k = \frac{2\pi}{N}k$. Then such points are expressed as,

$$z_k = re^{jw_k} = re^{j2\pi k/N} \; ; 0 \le k \le (N-1) \longrightarrow (2)$$

Substituting the above value of z_k in Eq(1),

$$X(z_k) = \sum_{n=0}^{N-1} x(n) \left[re^{j2\pi k/N}\right]^{-n}$$

$$= \sum_{n=0}^{N-1} \left[x(n) r^{-n}\right] e^{\frac{-j2\pi kn}{N}} \; ; 0 \le k \le (N-1) \longrightarrow (3)$$

The above eqn represents DFT of the sequence $x(n)r^{-n}$. Here, all the points fall on the Circle of radius 'r'. Let the points z_k fall on the arc which begins at the point z_0. This point can be represented as,

$$z_0 = r_0 e^{j\theta_0}$$

Scanned with CamScanner

If the arc spirals towards origin (or) away from it, they points Z_K can be represented as,

$$Z_0 = r_0 e^{j\theta_0} (R_0 e^{j\phi_0})^K \; ; \; 0 \le K \le (N-1) \longrightarrow (4)$$

Here if $R_0 < 1$, then the points fall on the Contour which spirals towards the origin and if $R_0 > 1$, then the points fall on the Contour that spirals away from the origin. If $R_0 = 1$, then the points fall on the circle of radius 'r_0' and if $R_0 = r_0 = 1$, then the points fall on the unit circle.

Substitute the value of Z_K in Eq(1),

$$X(Z_K) = \sum_{n=0}^{N-1} x(n) \left[r_0 e^{j\theta_0} (R_0 e^{j\phi_0})^K \right]^{-n}$$

$$X(Z_K) = \sum_{n=0}^{N-1} x(n) (r_0 e^{j\theta_0})^{-n} V^{-nK} \text{ , where } V = R_0 e^{j\phi_0} \longrightarrow (5)$$

The product 'nK' can be Considered as,

$$nK = \frac{1}{2} \left[K^2 + n^2 - (K-n)^2 \right] \longrightarrow (6)$$

$\therefore$ Eq(5) becomes,

$$X(Z_K) = \sum_{n=0}^{N-1} x(n) (r_0 e^{j\theta_0})^{-n} \cdot V^{\frac{-n^2}{2}} \cdot V^{\frac{-K^2}{2}} \cdot V^{\frac{(K-n)^2}{2}}$$

$$\text{Let } g(n) = x(n) (r_0 e^{j\theta_0})^{-n} \cdot V^{\frac{-n^2}{2}} \longrightarrow (7)$$

$$\therefore X(Z_K) = V^{\frac{-K^2}{2}} \sum_{n=0}^{N-1} g(n) \cdot V^{\frac{(K-n)^2}{2}} \longrightarrow (8)$$

Let there be a digital filter with unit sample response given as,

$$h(n) = V^{n^2/2} \longrightarrow (9)$$

$$h(K-n) = V^{\frac{(K-n)^2}{2}} \longrightarrow (10)$$

Scanned with CamScanner

∴ Eq(8) becomes,

$$X(Z_K) = V^{\frac{-K^2}{2}} \sum_{n=0}^{N-1} g(n)\, h(K-n) \longrightarrow (11)$$

Here, the summation represents the output of the digital filter (i.e., convolution) having input $x(n)$ and unit sample response of $h(n)$. Let this output be $y(K)$.

i.e., $$y(K) = \sum_{n=0}^{N-1} g(n)\, h(K-n) \; ; \; 0 \le K \le (N-1) \longrightarrow (12)$$

∴ Eq(11) becomes,

$$X(Z_K) = V^{-K^2/2} \cdot y(K)$$

From Eq(9), $V^{-K^2/2} = [h(K)]^{-1}$

$$\therefore X(Z_K) = [h(K)]^{-1} y(K) = \frac{y(K)}{h(K)} \; ; \; K = 0, 1, \ldots (N-1)$$

Here, the DFT at discrete points Z_K is the ratio of $y(K)$ and $h(K)$. The points Z_K fall on a circular (or) spiral contour and $y(K)$ is obtained as the output of the digital filter whose unit sample response is $h(K)$. This is nothing but linear filtering approach to the computation of DFT.

Substituting the value of 'V' of Eq(5) in Eq(9),

$$h(n) = [R_0 e^{j\phi_0}]^{n^2/2} = R_0^{n^2/2} e^{j\phi_0 n^2/2}$$

where, $h(n)$ has magnitude of $R_0^{n^2/2}$ and angles $\omega n = \frac{\phi_0 n^2}{2}$

$$\therefore \omega = \frac{\phi_0 n}{2}$$

Thus, 'ω' is frequency of complex exponential signal, which increases linearly with time (i.e., n). Such signals are called chirp signals. Hence, this method is called chirp-Z transform.

Scanned with CamScanner

Problems on Goertzel Algorithm :-

1) For the given sequence, $x(n) = \{1, 2, 1, 0\}$. find $X(1)$ using Goertzel algorithm.

Soln :-

W.K.T $X(K) = y_K(n)\big|_{n=N}$

Here $N = 4$ and $K = 1$,

$\therefore X(1) = y_1(n)\big|_{n=4}$

$X(1) = y_1(4)$

W.K.T $y_K(n) = x(n) + W_N^{-K}\, y_K(n-1)$

$y_1(n) = x(n) + W_4^{-1}\, y_1(n-1)$

$y_1(n) = x(n) + j\, y_1(n-1)$ $\qquad W_4^{-1} = e^{j\frac{2\pi}{4}} = \cos\frac{\pi}{2} + j\sin\frac{\pi}{2} = j$

@ $n=0$, $y_1(0) = x(0) + j\, y_1(-1) = 1 + j(0) = \underline{1}$

@ $n=1$, $y_1(1) = x(1) + j\, y_1(0) = 2 + j(1) = \underline{2+j}$

@ $n=2$, $y_1(2) = x(2) + j\, y_1(1) = 1 + j(2+j) = \underline{j2}$

@ $n=3$, $y_1(3) = x(3) + j\, y_1(2) = 0 + j(j2) = \underline{-2}$

@ $n=4$, $y_1(4) = x(4) + j\, y_1(3) = 0 + j(-2) = \underline{-j2}$

$\therefore X(1) = y_1(4) = -j2$

2) For the given sequence, $x(n) = \{1, 2, 1, 0\}$, determine $X(2)$ using Goertzel Algorithm. Assume the zero initial conditions.

Soln :-

W.K.T $X(K) = y_K(n)\big|_{n=N}$

Here $N = 4$ and $K = 2$

$X(2) = y_2(n)\big|_{n=4}$

$X(2) = y_2(4)$

Scanned with CamScanner

W.K.T $y_K(n) = x(n) + W_N^{-K} y_K(n-1)$

$y_2(n) = x(n) + W_4^{-2} y_2(n-1)$

$y_2(n) = x(n) - y_2(n-1)$

$W_4^{-2} = e^{\frac{j2\pi}{4}.2} = 1$

@ $n=0$, $y_2(0) = x(0) - y_2(-1) = 1 - 0 = \underline{1}$

@ $n=1$, $y_2(1) = x(1) - y_2(0) = 2 - 1 = \underline{1}$

@ $n=2$, $y_2(2) = x(2) - y_2(1) = 1 - 1 = \underline{0}$

@ $n=3$, $y_2(3) = x(3) - y_2(2) = 0 - 0 = \underline{0}$

@ $n=4$, $y_2(4) = x(4) - y_2(3) = 0 - 0 = \underline{0}$

$\langle \therefore X(2) = y_2(4) = 0 \rangle$

3) For the sequence $x(n) = \{2, 0, 2, 0\}$, determine $X(2)$ using Goertzel filter. Assume the zero initial conditions.

Soln:-

W.K.T $X(K) = y_K(n)\big|_{n=N}$

Here $N=4$ and $K=2$,

$X(2) = y_2(n)\big|_{n=4}$

$X(2) = y_2(4)$

W.K.T $y_K(n) = x(n) + W_N^{-K} y_K(n-1)$

$y_2(n) = x(n) + W_4^{-2} y_2(n-1)$

$y_2(n) = x(n) - y_2(n-1)$

$W_4^{-2} = e^{\frac{j2\pi}{4}.2} = \underline{-1}$

@ $n=0$, $y_2(0) = x(0) - y_2(-1) = 2 - 0 = \underline{2}$

@ $n=1$, $y_2(1) = x(1) - y_2(0) = 0 - 2 = \underline{-2}$

@ $n=2$, $y_2(2) = x(2) - y_2(1) = 2 - (-2) = \underline{4}$

@ $n=3$, $y_2(3) = x(3) - y_2(2) = 0 - 4 = \underline{-4}$

@ $n=4$, $y_2(4) = x(4) - y_2(3) = 0 - (-4) = \underline{4}$

$\langle \therefore y_2(4) = X(2) = 4 \rangle$

Scanned with CamScanner

4) Given $x(n) = \{1, 0, 1, 0\}$, find $X(2)$ using Goertzel Algorithm.

Sol^n:- W.K.T $X(K) = y_K(n)|_{n=N}$

Here $N=4$ and $K=2$,

$X(2) = y_2(n)|_{n=4} = y_2(4)$

W.K.T $y_K(n) = x(n) + W_N^{-K} y_K(n-1)$

$y_2(n) = x(n) + W_4^{-2} y_2(n-1)$

$y_2(n) = x(n) - y_2(n-1)$

@$n=0$, $y_2(0) = x(0) - y_2(-1) = 1 - 0 = \underline{1}$

@$n=1$, $y_2(1) = x(1) - y_2(0) = 0 - 1 = \underline{-1}$

@$n=2$, $y_2(2) = x(2) - y_2(1) = 1 - (-1) = \underline{2}$

@$n=3$, $y_2(3) = x(3) - y_2(2) = 0 - 2 = \underline{-2}$

@$n=4$, $y_2(4) = x(4) - y_2(3) = 0 - (-2) = \underline{2}$

$\langle \therefore y_2(4) = X(2) = 2 \rangle$

5) For the sequence $x(n) = \{5, 3-j2, -3, 3+j2\}$, determine $X(2)$ using Goertzel algorithm. Assume that the initial conditions are zero.

Sol^n:- W.K.T $X(K) = y_K(n)|_{n=N}$

Here $N=4$, $K=2$

$X(2) = y_2(n)|_{n=4}$

$X(2) = y_2(4)$

W.K.T $y_K(n) = x(n) + W_N^{-K} y_K(n-1)$

$y_2(n) = x(n) + W_4^{-2} y_2(n-1)$

$y_2(n) = x(n) - y_2(n-1)$

Scanned with CamScanner

@n=0, $y_2(0) = x(0) - y_2(-1) = 5 - 0 = \underline{5}$

@n=1, $y_2(1) = x(1) - y_2(0) = 3 - j2 - 5 = \underline{-2 - j2}$

@n=2, $y_2(2) = x(2) - y_2(1) = -3 - (-2 - j2) = \underline{-1 + j2}$

@n=3, $y_2(3) = x(3) - y_2(2) = 3 + j2 - (-1 + j2) = \underline{4}$

@n=4, $y_2(4) = x(4) - y_2(3) = 0 - 4 = \underline{-4}$

$\langle \therefore X(2) = y_2(4) = -4 \rangle$

6) In the Evaluation of 8-point DFT of the sequence $x(n) = \{2, 1, 2, 1\}$. Find X(3) using Goertzel algorithm.

Soln :- W.K.T $X(K) = y_K(n)\big|_{n=N}$

Here K = 3 and N = 8

$\therefore X(3) = y_3(n)\big|_{n=8}$

$X(3) = y_3(8)$

W.K.T $y_K(n) = x(n) + W_N^{-K}\, y_K(n-1)$

$y_3(n) = x(n) + W_8^{-3}\, y_3(n-1)$ $\qquad W_8^{-3} = e^{\frac{j2\pi}{8} \times 3} = -0.7071 + j0.7071$

$y_3(n) = x(n) + (-0.7071 + j0.7071)\, y_3(n-1)$

@n=0, $y_3(0) = x(0) + (-0.7071 + j0.7071)\, y_3(-1) = 2 + 0 = \underline{2}$

@n=1, $y_3(1) = x(1) + (-0.7071 + j0.7071)\, y_3(0) = 1 + (-0.7071 + j0.7071)(2)$

$= \underline{-0.4142 + j1.4142}$

@n=2, $y_3(2) = x(2) + (-0.7071 + j0.7071)\, y_3(1) = 2 + (-0.7071 + j0.7071)(-0.4142 + j1.4142)$

$= \underline{1.2929 - j1.2929}$

@n=3, $y_3(3) = x(3) + (-0.7071 + j0.7071)\, y_3(2)$

$= 1 + (-0.7071 + j0.7071)(1.2929 - j1.2929) = \underline{1 + j1.8284}$

@n=4, $y_3(4) = x(4) + (-0.7071 + j0.7071)\, y_3(3)$

$= 0 + (-0.7071 + j0.7071)(1 + j1.8284) = \underline{-2 - j0.586}$

Scanned with CamScanner

@n=5, $y_3(5) = x(5) + (-0.7071 + j0.7071)\, y_3(4) = 1.8286 - j1$

@n=6, $y_3(6) = x(6) + (-0.7071 + j0.7071)\, y_3(5) = -0.586 + j2$

@n=7, $y_3(7) = x(7) + (-0.7071 + j0.7071)\, y_3(6) = -1 - j1.8286$

@n=8, $y_3(8) = x(8) + (-0.7071 + j0.7071)\, y_3(7) = 2 + j0.586$

$\therefore y_3(8) = X(3) = 2 + j0.586$

7) For the given sequence, $x(n) = \{1, 2, 3, 4, 4, 3, 2, 1\}$. Find X(1) using Goertzel algorithm.

Sol^n:- $X(K) = y_K(n)\big|_{n=N}$

Here K=1, N=8

$X(1) = y_1(n)\big|_{n=8} = y_1(8)$

W.K.T $y_K(n) = x(n) + W_N^{-K}\, y_K(n-1)$

$y_1(n) = x(n) + W_8^{-1}\, y_1(n-1)$

$y_1(n) = x(n) + (0.7071 + j0.7071)\, y_1(n-1)$

$W_8^{-1} = e^{\frac{j2\pi}{8}} = \cos\frac{\pi}{4} + j\sin\frac{\pi}{4}$

$= 0.7071 + j0.7071$

@n=0, $y_1(0) = x(0) + (0.7071 + j0.7071)\, y_1(-1) = 1$

@n=1, $y_1(1) = x(1) + (0.7071 + j0.7071)\, y_1(0) = 2.7071 + j0.7071$

@n=2, $y_1(2) = x(2) + (0.7071 + j0.7071)\, y_1(1) = 4.4142 + j2.4142$

@n=3, $y_1(3) = x(3) + (0.7071 + j0.7071)\, y_1(2) = 5.4142 + j4.8284$

@n=4, $y_1(4) = x(4) + (0.7071 + j0.7071)\, y_1(3) = 4.4142 + j7.2425$

@n=5, $y_1(5) = x(5) + (0.7071 + j0.7071)\, y_1(4) = 1 + j8.2424$

@n=6, $y_1(6) = x(6) + (0.7071 + j0.7071)\, y_1(5) = -3.1211 + j6.5353$

@n=7, $y_1(7) = x(7) + (0.7071 + j0.7071)\, y_1(6) = -5.828 + j2.4142$

@n=8, $y_1(8) = x(8) + (0.7071 + j0.7071)\, y_1(7) = -5.828 - j2.4139$

$X(1) = y_1(8) = -5.828 - j2.4139$

Scanned with CamScanner

8) Compute the DFT of the sequence: $x(n) = \{1, 2, 1, 2\}$ using Goertzel algorithm.

Soln:-

$$X(K) = y_K(n) = \sum_{n=0}^{N-1} x(n) W_N^{-K(N-n)}$$

Here $N = 4$,

$$X(K) = \sum_{n=0}^{3} x(n) W_4^{-K(4-n)}$$

$$= x(0) W_4^{-4K} + x(1) W_4^{-3K} + x(2) W_4^{-2K} + x(1) W_4^{-K}$$

$$X(K) = W_4^{-4K} + 2W_4^{-3K} + W_4^{-2K} + 2W_4^{-K} \longrightarrow (1)$$

@ $K=0$, $X(0) = 1 + 2 + 1 + 2 = \underline{6}$

@ $K=1$, $X(1) = W_4^{-4} + 2W_4^{-3} + W_4^{-2} + 2W_4^{-1} = \underline{0}$

@ $K=2$, $X(2) = W_4^{-8} + 2W_4^{-6} + W_4^{-4} + 2W_4^{-2} = \underline{-2}$

@ $K=3$, $X(3) = W_4^{-12} + 2W_4^{-9} + W_4^{-6} + 2W_4^{-3} = \underline{0}$

$\therefore X(K) = \{6, 0, -2, 0\}$

Scanned with CamScanner

Problems on Efficient computations of the DFT :-

1) How many Complex multiplications and additions are required for computing 256-point DFT using direct computation of DFT and using FFT computations. which one is faster and at a faster rate?

Sol^n :- Given $N = 256$

Using direct Computation :-

a) Complex multiplications $= N^2 = (256)^2 = 65,536$

b) Complex additions $= N(N-1) = 256(256-1) = 65,280$

Using FFT algorithm :-

a) Complex multiplications $= \frac{N}{2} \log_2 N = \frac{256}{2} \log_2 256 = 1024$

b) Complex additions $= N \log_2 N = 256 \log_2 256 = 2048$

Speed improvement factor :-

$$\text{Speed improvement factor} = \frac{N^2}{\frac{N}{2} \log_2 N} = \frac{(256)^2}{\frac{256}{2} \log_2 256} = 64 \text{ times}$$

FFT is 64-times faster than direct computation in evaluation of DFT for $N = 256$.

2) what is the total number of Complex additions and multiplications required for $N = 512$ points, if DFT is Computed directly and if FFT is used and also find number of stages Computed and its memory requirements

Sol^n :- Given $N = 512$

Using direct Computation :-

a) Complex additions $= N(N-1) = 512(512-1) = 2,61,632$

b) Complex multiplications $= N^2 = (512)^2 = 2,62,144$

Scanned with CamScanner

Using FFT algorithm :-

a) Complex multiplications $= \frac{N}{2}\log_2 N = \frac{512}{2}\log_2 512 = 2304$

b) Complex additions $= N\log_2 N = 512\log_2 512 = 4608$

Number of stages required, $\gamma = \log_2 N = \log_2 512 = 9$ Stages

Memory requirements :-

Real registers + Complex registers $= 2N + \frac{N}{2} = 2\times512 + \frac{512}{2} = 1280$

∴ 1280 registers are used.

3) How many multiplications and additions are needed for 64-point sequence in calculations of DFT using FFT algorithm and using direct DFT computation. Specify number of real registers needed to perform these computation. Also find real multiplications and real additions are involved in DFT computations.

Sol^n :- Given $N = 64$

Using direct Computation :-

a) Complex multiplications $= N^2 = (64)^2 = 4096$

b) Complex additions $= N(N-1) = 64(64-1) = 4032$

Using FFT algorithm :-

a) Complex multiplications $= \frac{N}{2}\log_2 N = \frac{64}{2}\log_2 64 = 192$

b) Complex additions $= N\log_2 N = 64\log_2 64 = 384$

Memory requirements :-

Real registers + Complex registers $= 2N + \frac{N}{2} = 2\times64 + \frac{64}{2}$

$= 160$ registers

Scanned with CamScanner

Real multiplications and Real additions :-

a) Real multiplications $= 4N^2 = 4(64)^2 = 16384$

b) Real additions $= 4N^2 - 2N = 4(64)^2 - 2(64) = 16256$

4) Determine the following for a 128-point FFT Computation number of : a) Stages b) Butterflies in each stage c) Butterflies needed for Entire Computation d) total number of Complex multiplications e) total number of Complex additions f) Speed improvement factor Compared with direct Computation.

Sol^n :-

a) Number of stages, $r = \log_2 N = \log_2 128 = 7$ stages

b) Butterflies in each stage $= \frac{N}{2} = \frac{128}{2} = 64$ butterflies

c) Butterflies needed for Entire Computation $= \frac{N}{2} \log_2 N$

$= \frac{128}{2} \log_2 128 = 448$ Butterflies

d) Total number of Complex multiplications $= \frac{N}{2} \log_2 N$

$= \frac{128}{2} \log_2 128 = 448$

e) Total number of Complex additions $= N \log_2 N = 128 \log_2 128$

$= 896$

f) Speed improvement factor $= \frac{N^2}{\frac{N}{2} \log_2 N} = \frac{(128)^2}{\frac{128}{2} \log_2 128} = 36.57$ times

FFT is 36.57 times faster than direct Computation of DFT for $N = 128$.

Scanned with CamScanner

Module-4:

Anusha.M.N
Asst. professor
Dept. of ECE
BGSIT, B.G.Nagar

Realization of Digital filters:
Structures for IIR filters

For the design of digital filters, the system function H(z) (or) the impulse response h(n) must be specified. Then the digital filter structure can be implemented (or) synthesized in hardware/software form by its difference equation obtained directly from H(z) (or) h(n). Each difference equation can be implemented by using a digital computer (or) special purpose digital hardware.

In order to implement the specified difference equation of the system, the required basic operations are addition, delay and the multiplying by a constant.

W.K.T IIR system is given by the difference equation,

$$\sum_{K=0}^{N} a_K y(n-K) = \sum_{K=0}^{M} b_K x(n-K)$$

$$a_0 y(n) + \sum_{K=1}^{N} a_K y(n-K) = \sum_{K=0}^{M} b_K x(n-K)$$

Let $a_0 = 1$,

$$y(n) + \sum_{K=1}^{N} a_K y(n-K) = \sum_{K=0}^{M} b_K x(n-K)$$

∴ The output of a first order discrete-time LTI system can be expressed as a linear combination of the inputs and outputs,

$$y(n) = -\sum_{K=1}^{N} a_K y(n-K) + \sum_{K=0}^{M} b_K x(n-K) \longrightarrow (1)$$

Eq(1) is called as a linear constant co-efficient difference eqn.

Scanned with CamScanner

Where, a_K and b_K are constants with $a_0 = 1$ ($a_0 \neq 0$) and $M \leq N$.

The current output $y(n)$ is equal to the sum of past outputs from $y(n-1)$ to $y(n-N)$ which are scaled by the delay-dependent feedback co-efficient a_K plus the sum of future, present and past inputs, which are scaled by the delay-dependent feedfoward co-efficient b_K.

Taking Z-transform for eq(1),

$$Y(z) = -\sum_{K=1}^{N} a_K . z^{-K} Y(z) + \sum_{K=0}^{M} b_K . z^{-K} X(z)$$

$$Y(z)\left[1 + \sum_{K=1}^{N} a_K z^{-K}\right] = \left[\sum_{K=0}^{M} b_K z^{-K}\right] X(z)$$

$$\therefore \frac{Y(z)}{X(z)} = H(z) = \frac{\sum_{K=0}^{M} b_K . z^{-K}}{1 + \sum_{K=1}^{N} a_K z^{-K}} \longrightarrow (2)$$

$$\therefore \text{System transfer function, } H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \ldots + b_M z^{-M}}{1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_N z^{-N}}$$

The advantages of representing the digital system in block diagram forms are:

1) Just by inspection, the computation algorithm can be easily written.
2) The hardware requirements can be easily determined.
3) A variety of equivalent block diagram representations can be easily developed from the transfer function
4) The relationship between output and input can be determined.

Scanned with CamScanner

Types of Digital filter Structures :-

There are many types of realization (structures) of digital filters.

Digital filter Structures

IIR filter structures

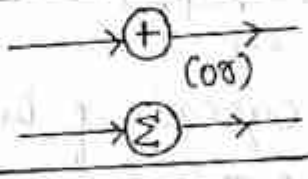
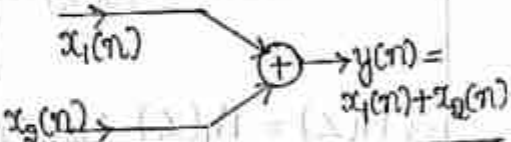
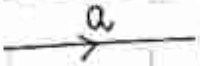
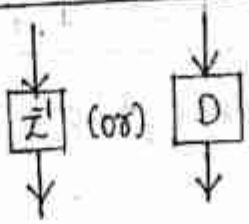
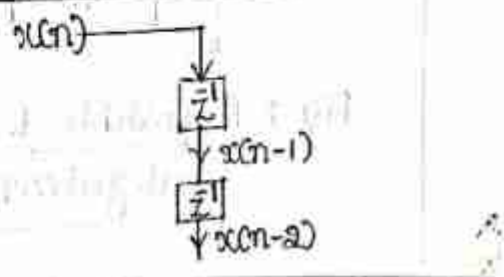
1. Direct form structures
2. Cascade form structures
3. parallel form structures
4. Signal flowgraphs & Transposed structure
5. Lattice structure
6. Lattice-ladder structure

FIR filter structures

1. Direct form structures
2. Cascade form structures
3. Lattice structure
4. Linear-phase FIR structure
5. Frequency-sampling structure

Basic network elements required for the construction of digital filters :-

An LTI digital filter can be conveniently represented as a block diagram. The block diagram of a digital filter consists of delay elements, multipliers and adders.

Sl. No.	Name of the block	Symbol	Example
1.	Adder		
2.	Constant Multiplier		
3.	Delay Elements		

Scanned with CamScanner

Note :-
If the number of delays in the structure is equal to order of the difference equation (or) order of the transfer function, then it is called "Canonical form realization".
If the number of delays in the structures are not same as order, then it is called "non-canonical realization".

Basic IIR filter Structures :-

Causal IIR systems are characterized by the linear Constant Co-efficient difference equation of Eq(1) (or) equivalently by the real rational transfer function of Eq(2). From these equations, it can be seen that the realization of IIR systems involves a recursive computational algorithm.

1) Direct-form structures :-

W.K.T $$H(z) = \frac{\sum_{K=0}^{M} b_K . z^{-K}}{1 + \sum_{K=1}^{N} a_K . z^{-K}}$$

a) Direct form-I :-

Let $H_1(z) = \frac{W(z)}{X(z)} = \sum_{K=0}^{M} b_K . z^{-K}$; Consists of zeros of $H(z)$.

and $H_2(z) = \frac{Y(z)}{W(z)} = \frac{1}{1 + \sum_{K=1}^{N} a_K . z^{-K}}$; Consists of poles of $H(z)$.

$\therefore H(z) = H_1(z) . H_2(z)$; Cascade of two functions.

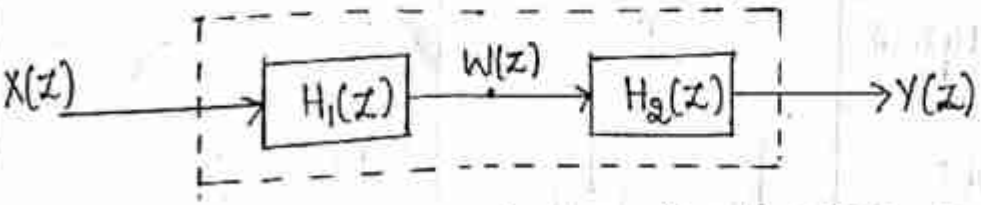


Fig : A possible IIR system direct form-I realization

Scanned with CamScanner

Consider a second-order ($N=2$) filter characterized by the system function:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{\sum_{K=0}^{3} b_K \cdot z^{-K}}{1 + \sum_{K=1}^{3} a_K z^{-K}} \longrightarrow (3)$$

Eq(3) can be written as,

$H(z) = H_1(z) \cdot H_2(z)$, where $H_1(z) = \frac{W(z)}{X(z)} = b_0 + b_1 z^{-1} + b_2 z^{-2} \longrightarrow (4)$

and $H_2(z) = \frac{Y(z)}{W(z)} = \frac{1}{1 + a_1 z^{-1} + a_2 z^{-2}} \longrightarrow (5)$

Taking I-Z T for Eq(4),

$W(n) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) \longrightarrow (6)$

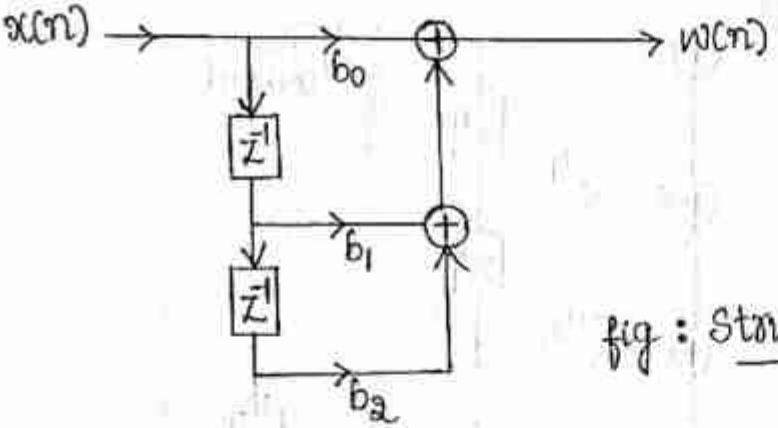


fig : Structure for $H_1(z)$

Taking I-ZT for Eq(5),

$W(n) = y(n) + a_1 y(n-1) + a_2 y(n-2)$

$\therefore y(n) = W(n) - a_1 y(n-1) - a_2 y(n-2) \longrightarrow (7)$

The realization of Eq(7) is,

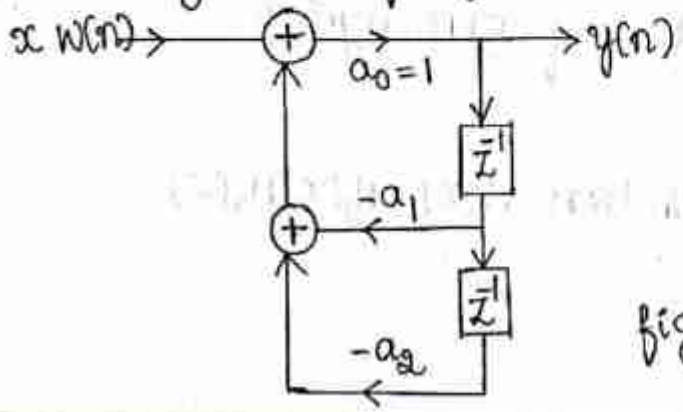


fig : Structure for $H_2(z)$

Scanned with CamScanner

Cascading $H_1(z)$ and $H_2(z)$, we get direct form-I structure:

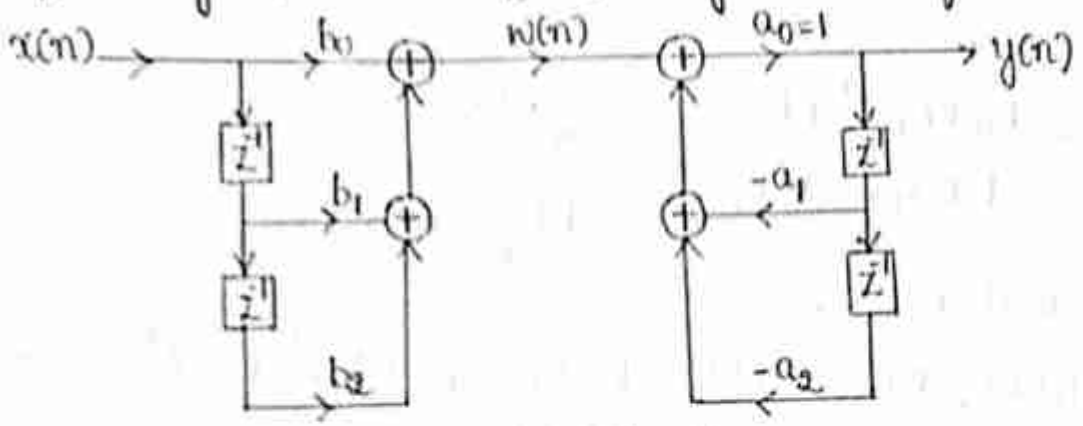


Fig : Direct form-I structure

Note :-
We observe that the direct form-I structure is "non-canonical" as it employs 4-delays for this second-order system.
Generally, we can write as shown below:

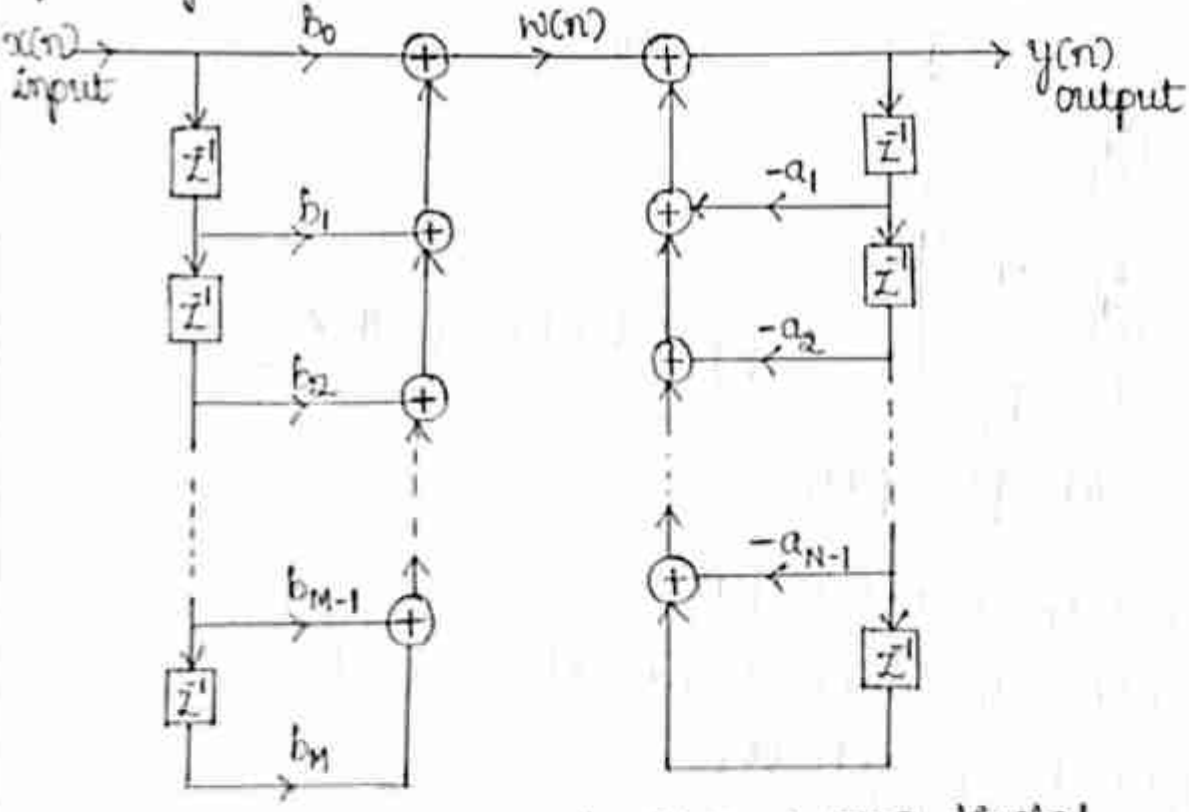


Fig : Direct form-I realization of IIR digital filter

Note :- The overall transfer function $H(z) = H_1(z)H_2(z)$
$H_1(z)$ are all zero filter
$H_2(z)$ are all pole filter

Scanned with CamScanner

→ (M+N+1) multiplications are required.
→ (M+N) additions are required.
→ (M+N) memory elements (or) delay elements are required.

Note :-

1) A system containing feedback (i.e., present output depends not only on inputs but also on the past outputs) is referred as "recursive system".

i.e., $y(n) = x(n) + y(n-1) + \frac{1}{2}y(n-2)$

2) A system which does not contain feedback (i.e., present output does not depend on past output) is referred as "non-recursive system".

i.e., $y(n) = \sum_{K=0}^{N} x(K)$

b) Direct form-II :-

Consider, $H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{K=0}^{M} b_K . z^{-K}}{1 + \sum_{K=1}^{N} a_K . z^{-K}}$

Now, $\frac{W(z)}{X(z)} . \frac{Y(z)}{W(z)} = \left[\frac{1}{1+\sum_{K=1}^{N} a_K z^{-K}}\right] \left[\sum_{K=0}^{M} b_K z^{-K}\right]$

Let $\frac{W(z)}{X(z)} = \frac{1}{1+\sum_{K=1}^{N} a_K . z^{-K}} \Rightarrow X(z) = W(z)\left[1 + \sum_{K=1}^{N} a_K z^{-K}\right]$

$X(z) = W(z) + \sum_{K=1}^{N} a_K . z^{-K} W(z)$

Taking I-ZT,

$x(n) = w(n) + \sum_{K=1}^{N} a_K w(n-K)$

$\therefore w(n) = x(n) - \sum_{K=1}^{N} a_K w(n-K) \longrightarrow (1)$

Scanned with CamScanner

Let $\frac{Y(z)}{W(z)} = \sum_{K=0}^{M} b_K . z^{-K} \Rightarrow Y(z) = \sum_{K=0}^{M} b_K . z^{-K} W(z)$

Taking I-ZT,

$$y(n) = \sum_{K=0}^{M} b_K w(n-K) \longrightarrow (2)$$

In direct form-II structure, we can observe that:

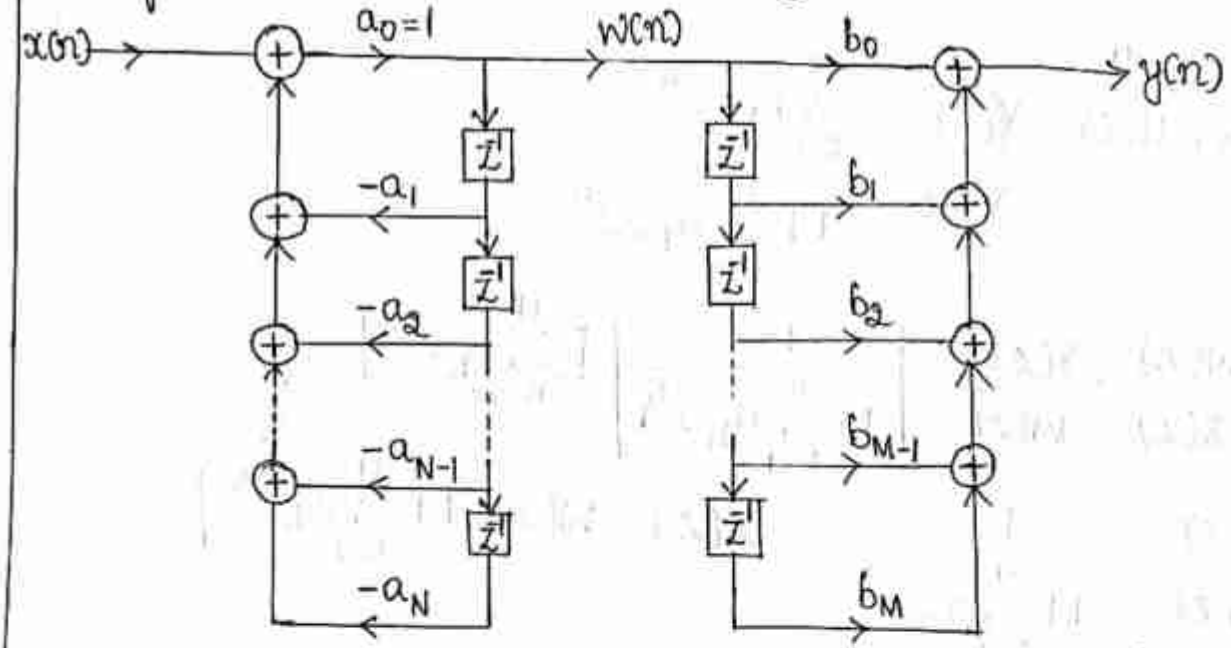
→ This structure requires (M+N+1) multiplications.

→ (M+N) additions are required.

→ max{M, N} memory locations.

→ Direct form-II is called as "Canonical form" because the order of the filter is equal to the number of delay Elements.

The Eq(1) and (2) can be written as:



We can merge these delays into one delay as shown below:

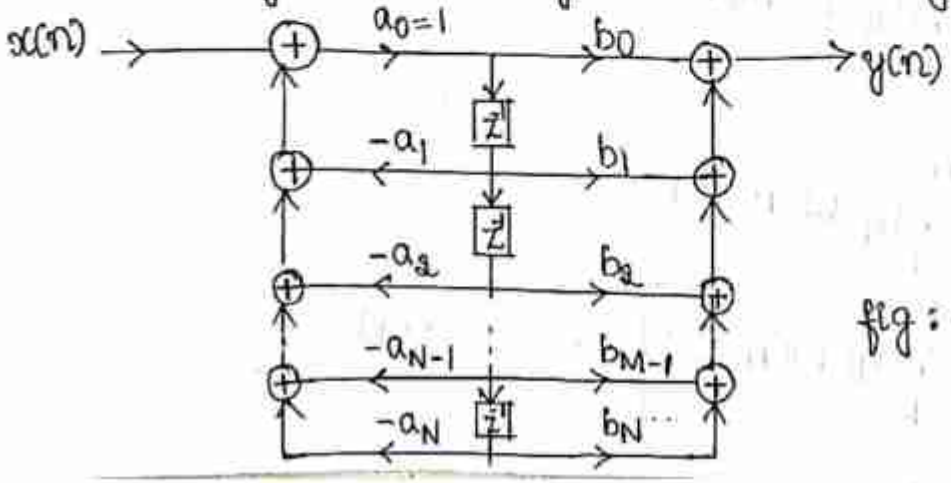


fig: Direct form-II Structure

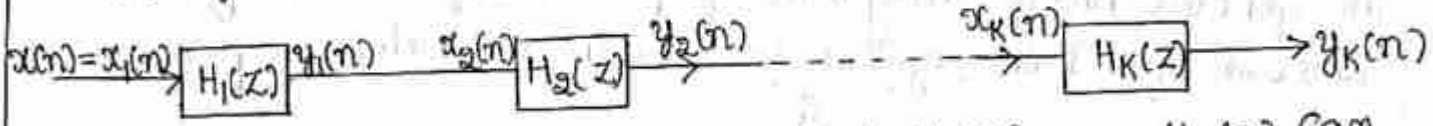
Scanned with CamScanner

2) Cascaded form structures :-

In cascade realization, the system function $H(z)$ is broken into product of transfer functions $H_1(z), H_2(z), \ldots\ldots H_K(z)$.

i.e., $H(z) = H_1(z) \cdot H_2(z) \ldots\ldots H_K(z)$

Here, $H_1(z), H_2(z), \ldots\ldots, H_K(z)$ are second-order polynomials. That is to obtain the cascade form realization, the numerator and denominator of given transfer function $H(z)$ is factored into product of second-order terms.



The each subtransfer function $H_1(z), H_2(z), \ldots\ldots H_K(z)$ can be realized using direct form-I and II structures. The total transfer function is obtained by connecting all second order subsystems in series as shown in the above figure.

$$H(z) = \frac{Y(z)}{X(z)} \Rightarrow H(z) = H_1(z) \cdot H_2(z) \ldots\ldots H_K(z)$$

$$\therefore H(z) = \frac{Y_1(z)}{X_1(z)} \cdot \frac{Y_2(z)}{X_2(z)} \ldots\ldots \frac{Y_K(z)}{X_K(z)}$$

Where, $Y_1(z), Y_2(z), \ldots\ldots Y_K(z)$ are different zero-polynomials; and $X_1(z), X_2(z), \ldots\ldots X_K(z)$ are different pole-polynomials.

$$\therefore H_K(z) = \frac{\sum_{K=0}^{2} b_K \cdot z^{-K}}{1 + \sum_{K=1}^{2} a_K \cdot z^{-K}} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \; ; \text{Here } M = N.$$

Note :-

1) $H(z)$ is a product of $H_1(z), H_2(z), \ldots\ldots H_K(z)$.

2) $H_1(z), H_2(z), \ldots\ldots, H_K(z)$ are system functions of the order 1 (or) 2.

Scanned with CamScanner

3) parallel form structures :-

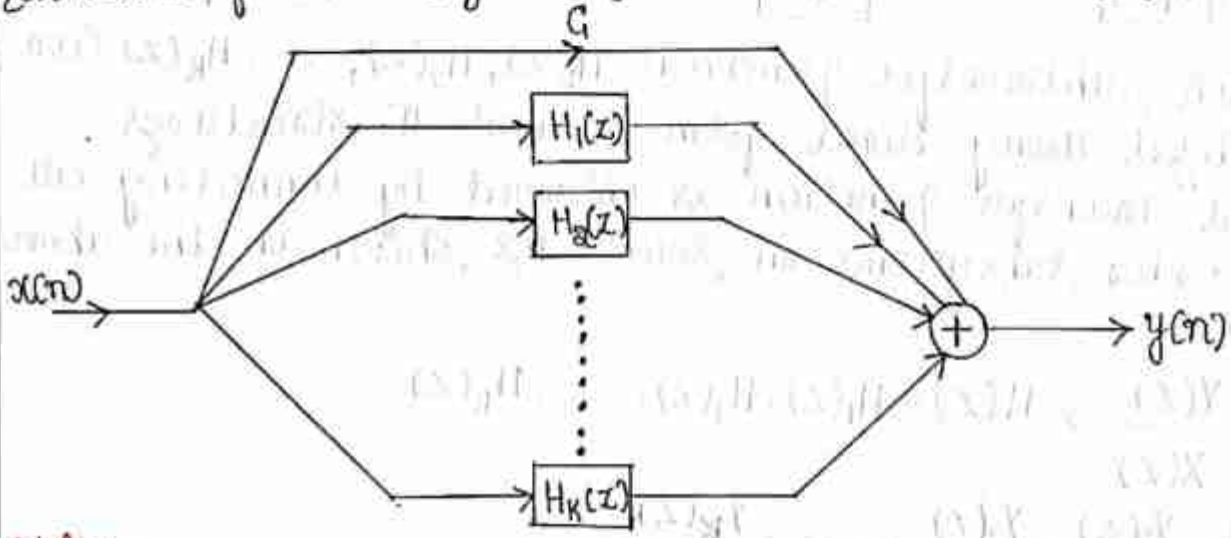
The system function $H(z)$ can be written as a sum of transfer functions $H_1(z), H_2(z), \ldots\ldots H_K(z)$ using partial fraction expansion as follows:

$$H(z) = H_1(z) + H_2(z) + H_3(z) + \ldots\ldots + H_K(z) + C$$

Where, C is a constant.

$H_1(z), H_2(z), \ldots\ldots H_K(z)$ are first (or) second-order subsystems.

The General block diagram of parallel form realization structure for IIR system is shown in below figure,



Note :-

The parallel form realization is generally used for high speed filtering applications. Since, this is a parallel connection, the process of filtering operation is performed parallely.

Note :-

1) The direct form-I and direct form-II structures are obtained directly from the corresponding transfer functions with any rearrangements. So, these structures are called direct-form structures.

2) The only advantage of direct-form realization is its implementation, which is Easy.

3) We prefer direct form-II structure compared to direct form-I structure because less memory Elements are required.

Scanned with CamScanner

4) Both direct form structures are sensitive to the effects of quantization errors in the co-efficients. So, in practical applications, these structures are not preferred.

By performing partial fraction expansion of $H(z)$,

$$H(z) = \left[\sum_{K=1}^{N} \frac{A_K}{1-p_K z^{-1}}\right] + C$$

Assume: $M < N$

$M \rightarrow$ no. of zeros

$N \rightarrow$ no. of poles

where, $p_K \rightarrow$ poles, $C \rightarrow$ Constant

$A_K \rightarrow$ Co-efficients of partial fraction expansion.

Note :- W.K.T $H(z) = C + H_1(z) + H_2(z) + \ldots\ldots + H_K(z)$

$$\frac{Y(z)}{X(z)} = C + H_1(z) + H_2(z) + \ldots\ldots + H_K(z)$$

$$\therefore Y(z) = C.X(z) + H_1(z).X(z) + H_2(z)X(z) + \ldots\ldots + H_K(z)X(z).$$

This equation can be realized in parallel form as shown below:

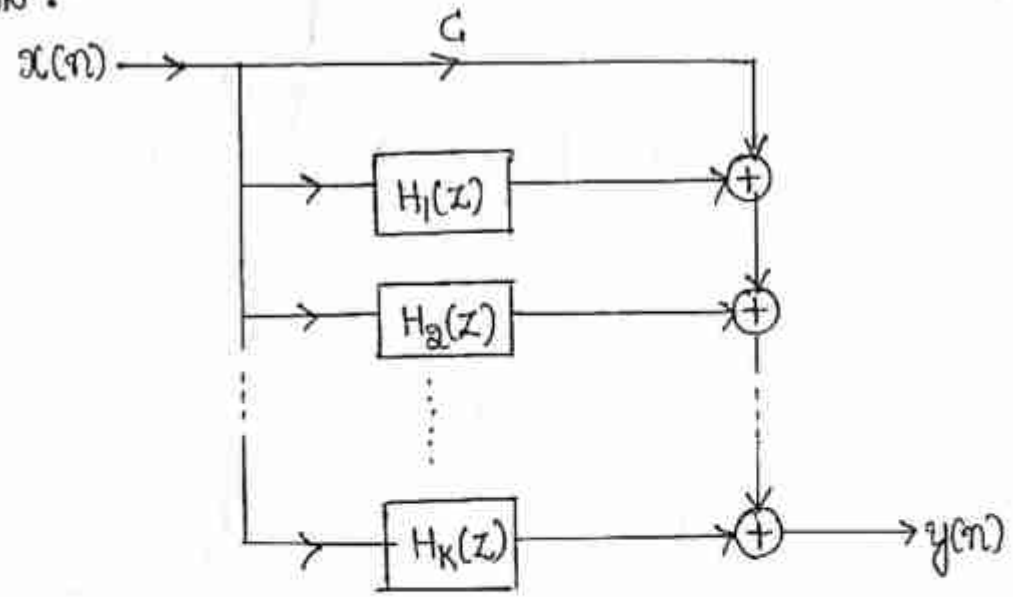


fig: parallel form realization

Scanned with CamScanner

Module-4:

Anusha.M.N
Asst. professor
Dept. of ECE
BGSIT

Structures for IIR Systems

Problems on realization of IIR filters :-

1) For the given difference equation, draw direct form-I, direct form-II, cascade and parallel structure:

$$2y(n) - \frac{3}{2}y(n-1) + \frac{1}{4}y(n-2) = \frac{1}{2}x(n-1)$$

Soln:-

Given $2y(n) - \frac{3}{2}y(n-1) + \frac{1}{4}y(n-2) = \frac{1}{2}x(n-1)$

Apply z-transform on both the sides,

$$2Y(z) - \frac{3}{2}z^{-1}Y(z) + \frac{1}{4}z^{-2}Y(z) = \frac{1}{2}z^{-1}X(z)$$

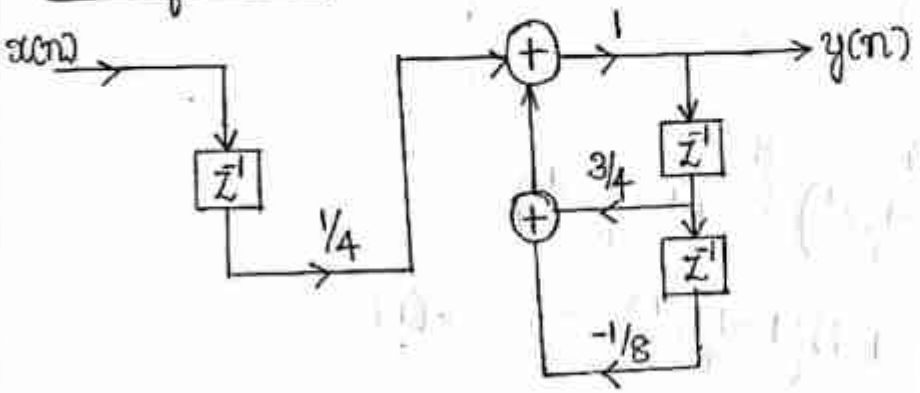
$$Y(z)\left[2 - \frac{3}{2}z^{-1} + \frac{1}{4}z^{-2}\right] = \frac{1}{2}z^{-1}X(z)$$

$$\frac{Y(z)}{X(z)} = \frac{\frac{1}{2}z^{-1}}{2 - \frac{3}{2}z^{-1} + \frac{1}{4}z^{-2}} = \frac{\frac{1}{2}z^{-1}}{2\left(1 - \frac{3}{4}z^{-1} + \frac{1}{8}z^{-2}\right)}$$

$$H(z) = \frac{\frac{1}{4}z^{-1}}{1 - \frac{3}{4}z^{-1} + \frac{1}{8}z^{-2}}$$

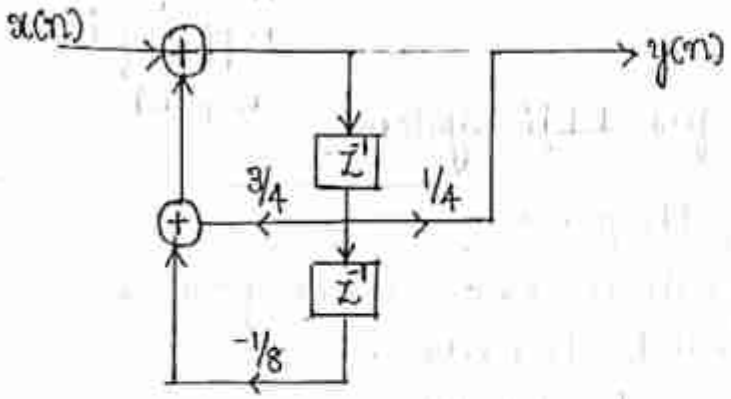
→ Change the sign of filter Co-efficients $a_1, a_2, \ldots a_N$

Direct form-I :-



Scanned with CamScanner

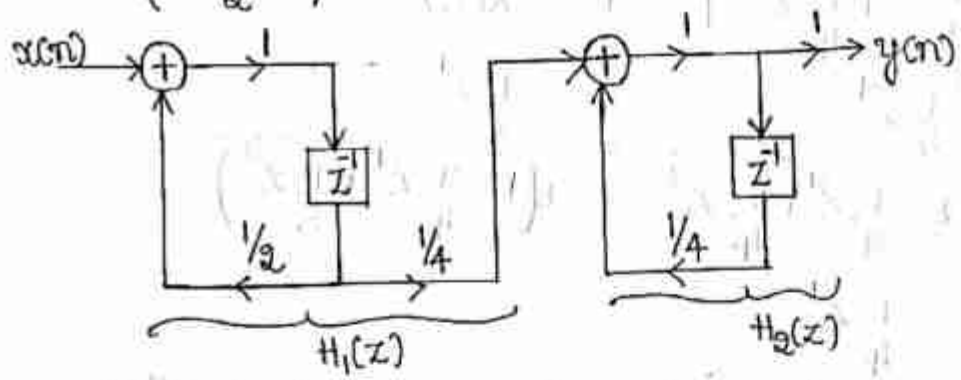
Direct form-II :-



Cascade form :-

$$H(z) = \frac{\frac{1}{4}z^{-1}}{1-\frac{3}{4}z^{-1}+\frac{1}{8}z^{-2}} = \frac{\frac{1}{4}z^{-1}}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)}$$

$$H_1(z) = \frac{\frac{1}{4}z^{-1}}{\left(1-\frac{1}{2}z^{-1}\right)} \quad ; \quad H_2(z) = \frac{1}{\left(1-\frac{1}{4}z^{-1}\right)}$$



Parallel form :-

$$H(z) = \frac{\frac{1}{4}z^{-1}}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)}$$; Here M<N, use partial fraction

$$\frac{\frac{1}{4}z^{-1}}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)} = \frac{A}{\left(1-\frac{1}{2}z^{-1}\right)} + \frac{B}{\left(1-\frac{1}{4}z^{-1}\right)}$$

$$\frac{1}{4}z^{-1} = A\left(1-\frac{1}{4}z^{-1}\right) + B\left(1-\frac{1}{2}z^{-1}\right) \longrightarrow (1)$$

Scanned with CamScanner

put $z^{-1}=2$ in eq(1),

$\frac{1}{4}(2) = A\left[1-\frac{1}{4}(2)\right]$

$\frac{1}{2} = A\left(\frac{1}{2}\right)$

$\langle A=1 \rangle$

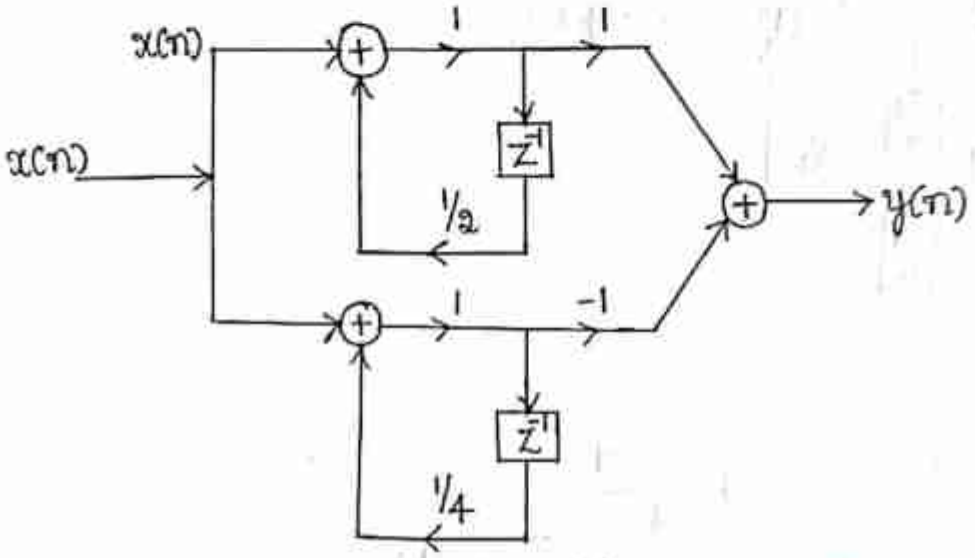
put $z^{-1}=4$ in eq(1),

$\frac{1}{4}(4) = B\left[1-\frac{1}{2}(4)\right]$

$1 = B(-1)$

$\langle B=-1 \rangle$

$\therefore H(z) = \frac{1}{\left(1-\frac{1}{2}z^{-1}\right)} + \frac{(-1)}{\left(1-\frac{1}{4}z^{-1}\right)}$



2) Given $x(n) = \delta(n) + \frac{1}{4}\delta(n-1) - \frac{1}{8}\delta(n-2)$ and $y(n) = \delta(n) - \frac{3}{4}\delta(n-1)$. Draw direct form-I and II, cascade and parallel form.

Sol^n:-

Given $x(n) = \delta(n) + \frac{1}{4}\delta(n-1) - \frac{1}{8}\delta(n-2)$

By Z-T, $X(z) = 1 + \frac{1}{4}z^{-1} - \frac{1}{8}z^{-2}$

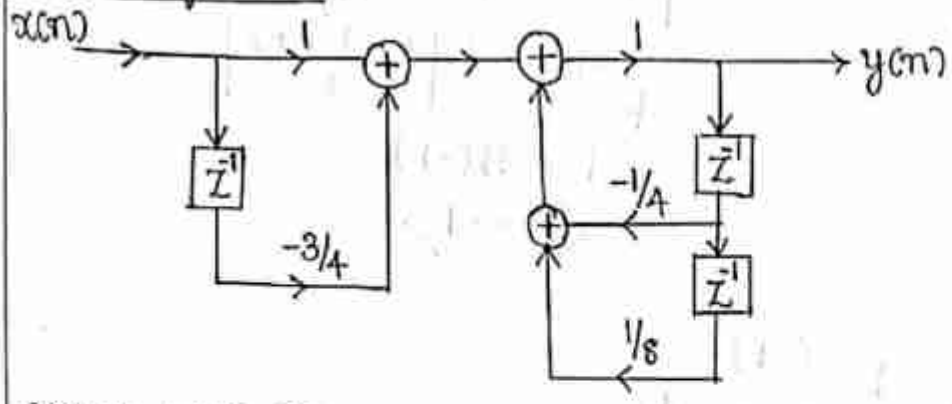
and $y(n) = \delta(n) - \frac{3}{4}\delta(n-1)$

By Z-T, $Y(z) = 1 - \frac{3}{4}z^{-1}$

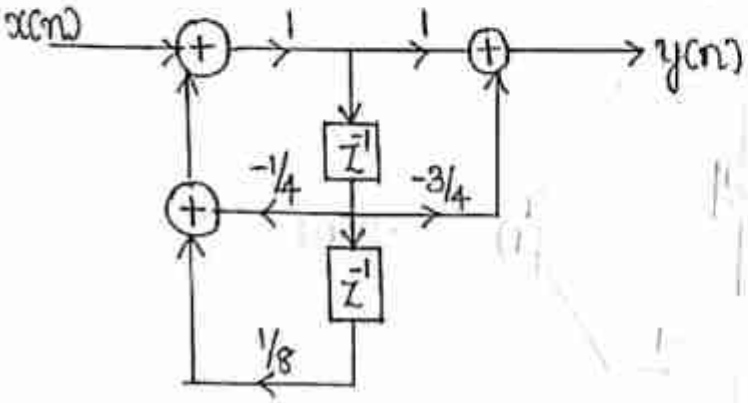
W.K.T $H(z) = \frac{Y(z)}{X(z)} = \frac{1-\frac{3}{4}z^{-1}}{1+\frac{1}{4}z^{-1}-\frac{1}{8}z^{-2}}$

Scanned with CamScanner

Direct form-I :-



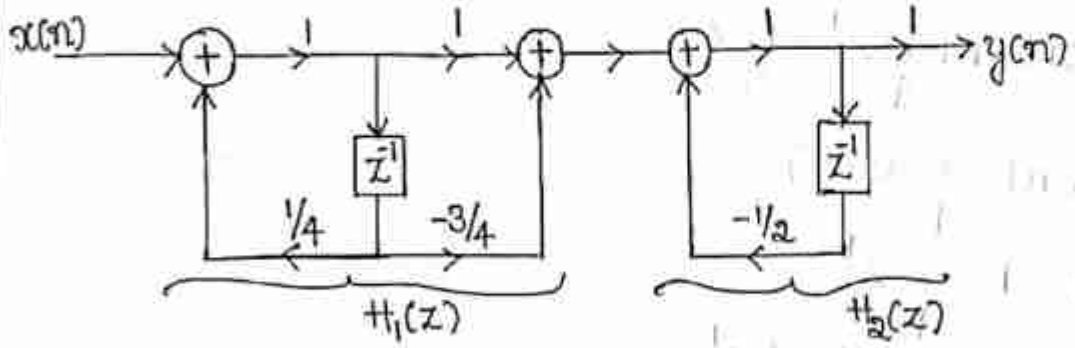
Direct form-II :-



Cascade form :-

$$H(z) = \frac{1-\frac{3}{4}z^{-1}}{1+\frac{1}{4}z^{-1}-\frac{1}{8}z^{-2}} = \frac{1-\frac{3}{4}z^{-1}}{\left(1-\frac{1}{4}z^{-1}\right)\left(1+\frac{1}{2}z^{-1}\right)}$$

$$H_1(z) = \frac{\left(1-\frac{3}{4}z^{-1}\right)}{\left(1-\frac{1}{4}z^{-1}\right)} \quad ; \quad H_2(z) = \frac{1}{\left(1+\frac{1}{2}z^{-1}\right)}$$



parallel form :-

$$H(z) = \frac{1-\frac{3}{4}z^{-1}}{1+\frac{1}{4}z^{-1}-\frac{1}{8}z^{-2}}$$; Here $M<N$, use partial fraction.

Scanned with CamScanner

$$\frac{1-\frac{3}{4}z^{-1}}{\left(1-\frac{1}{4}z^{-1}\right)\left(1+\frac{1}{2}z^{-1}\right)} = \frac{A}{\left(1-\frac{1}{4}z^{-1}\right)} + \frac{B}{\left(1+\frac{1}{2}z^{-1}\right)}$$

$$1-\frac{3}{4}z^{-1} = A\left(1+\frac{1}{2}z^{-1}\right) + B\left(1-\frac{1}{4}z^{-1}\right) \longrightarrow (1)$$

put $z^{-1}=4$ in eq(1),

$$1-\frac{3}{4}(4) = A\left[1+\frac{1}{2}(4)\right]$$

$$-2 = A(3)$$

$$\langle A = -\frac{2}{3} \rangle$$

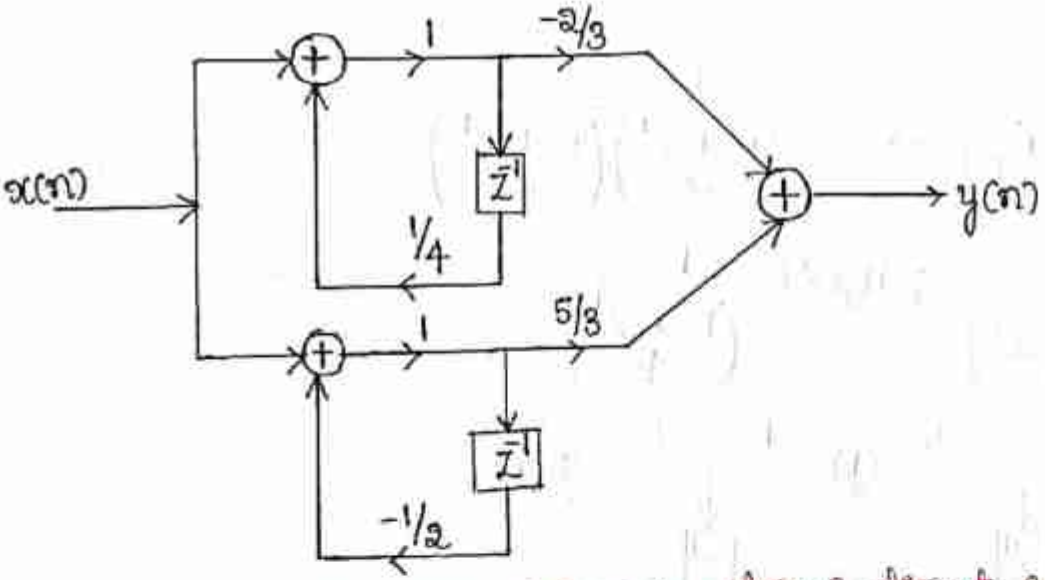
put $z^{-1}=-2$ in eq(1),

$$1-\frac{3}{4}(-2) = B\left[1-\frac{1}{4}(-2)\right]$$

$$\frac{5}{2} = B\left(\frac{3}{2}\right)$$

$$\langle B = \frac{5}{3} \rangle$$

$$\therefore H(z) = \frac{(-2/3)}{\left(1-\frac{1}{4}z^{-1}\right)} + \frac{(5/3)}{\left(1+\frac{1}{2}z^{-1}\right)}$$



3) For the given impulse response, draw direct form-I and II, cascade and parallel form : $h(n) = 4\left(\frac{1}{2}\right)^n u(n) - 2\left(\frac{1}{4}\right)^n u(n)$

(or)

$$h(n) = 4\left(\frac{1}{2}\right)^n - 2\left(\frac{1}{4}\right)^n ; \; n \geq 0$$

Sol^n:- Given $h(n) = 4\left(\frac{1}{2}\right)^n u(n) - 2\left(\frac{1}{4}\right)^n u(n)$

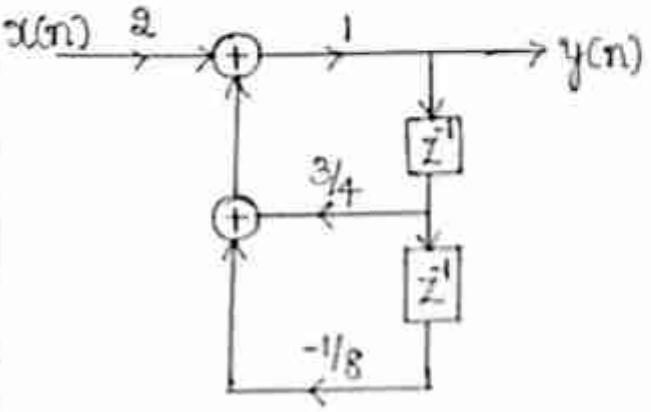
By z-transform, $H(z) = \frac{4}{\left(1-\frac{1}{2}z^{-1}\right)} - \frac{2}{\left(1-\frac{1}{4}z^{-1}\right)} \longrightarrow (1)$

Scanned with CamScanner

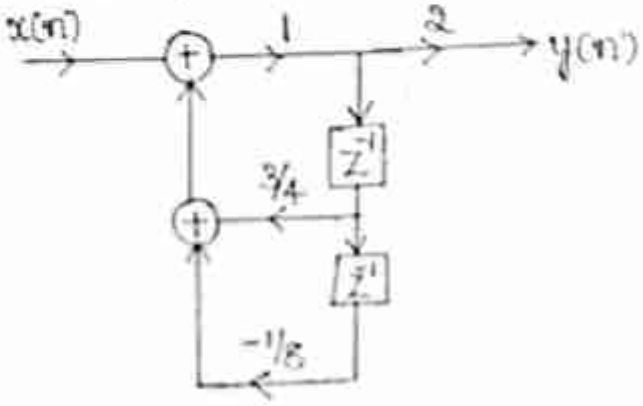
$$H(z) = \frac{4\left(1-\frac{1}{4}z^{-1}\right)-2\left(1-\frac{1}{2}z^{-1}\right)}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)} = \frac{4-z^{-1}-2+z^{-1}}{1-\frac{1}{4}z^{-1}-\frac{1}{2}z^{-1}+\frac{1}{8}z^{-2}}$$

$$H(z) = \frac{2}{1-\frac{3}{4}z^{-1}+\frac{1}{8}z^{-2}}$$

Direct form-I :-



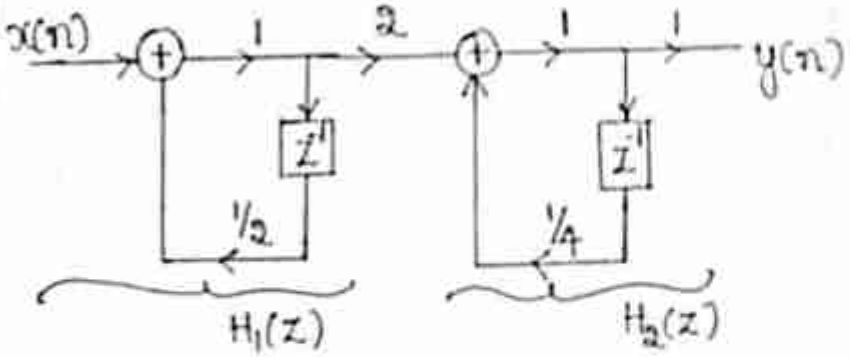
Direct form-II :-



Cascade form :-

$$H(z) = \frac{2}{1-\frac{3}{4}z^{-1}+\frac{1}{8}z^{-2}} = \frac{2}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)}$$

$$H_1(z) = \frac{2}{\left(1-\frac{1}{2}z^{-1}\right)} \quad ; \quad H_2(z) = \frac{1}{\left(1-\frac{1}{4}z^{-1}\right)}$$

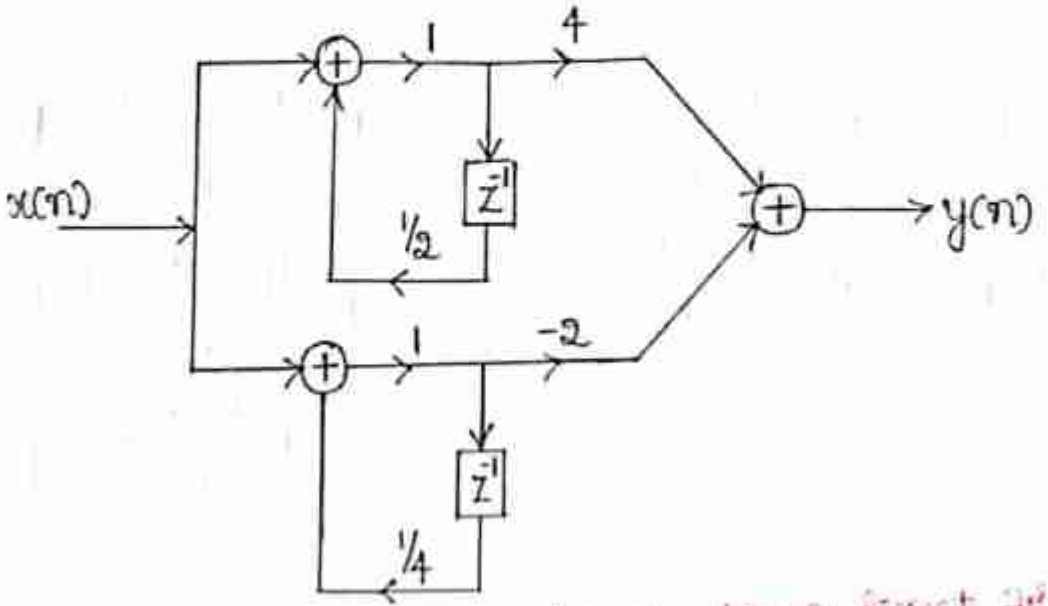


parallel form :-

In this problem, no need of using partial fraction directly. Consider eq(1) and write the parallel structure

Scanned with CamScanner

$$H(z) = \frac{4}{\left(1-\frac{1}{2}z^{-1}\right)} + \frac{(-2)}{\left(1-\frac{1}{4}z^{-1}\right)}$$



4) For the given set of functions, draw direct form-I and II, cascade and parallel form : $H(z) = 3 + \frac{4z}{z-0.5} - \frac{2}{z-0.25}$

Sol^n:- Given $H(z) = 3 + \frac{4z}{z-0.5} - \frac{2}{z-0.25}$

$$H(z) = \frac{3(z-0.5)(z-0.25)+4z(z-0.25)-2(z-0.5)}{(z-0.5)(z-0.25)}$$

$$= \frac{3(z^2-0.25z-0.5z+0.125)+4z^2-z-2z+1}{z^2-0.25z-0.5z+0.125}$$

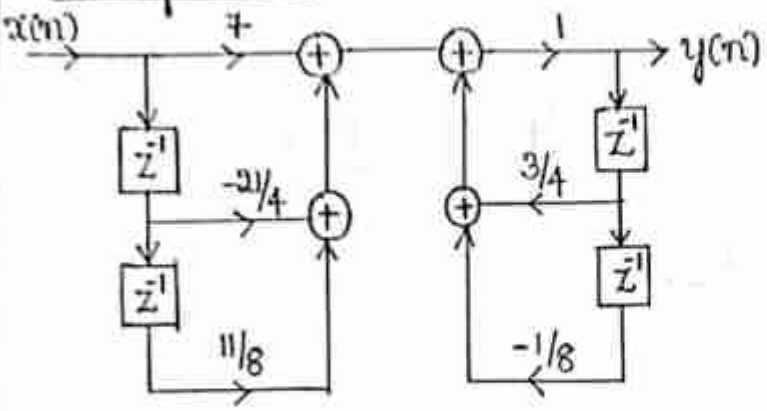
$$= \frac{3\left(z^2-\frac{3}{4}z+\frac{1}{8}\right)+4z^2-3z+1}{z^2-\frac{3}{4}z+\frac{1}{8}}$$

$$= \frac{7z^2-\frac{21}{4}z+\frac{11}{8}}{z^2-\frac{3}{4}z+\frac{1}{8}} = \frac{z^2\left[7-\frac{21}{4}z^{-1}+\frac{11}{8}z^{-2}\right]}{z^2\left[1-\frac{3}{4}z^{-1}+\frac{1}{8}z^{-2}\right]}$$

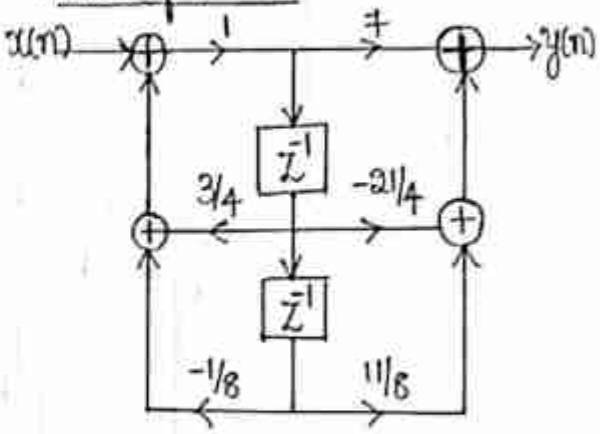
Scanned with CamScanner

$$H(z) = \frac{7 - \frac{21}{4}z^{-1} + \frac{11}{8}z^{-2}}{1 - \frac{3}{4}z^{-1} + \frac{1}{8}z^{-2}}$$

Direct form-I :-



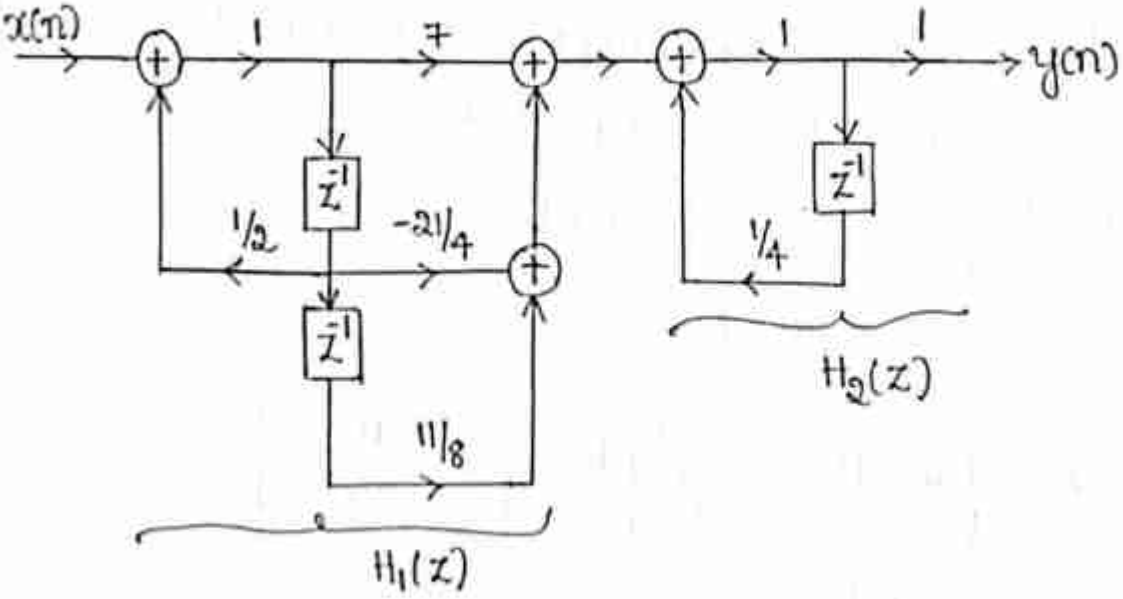
Direct form-II :-



Cascade form :-

$$H(z) = \frac{7 - \frac{21}{4}z^{-1} + \frac{11}{8}z^{-2}}{\left(1 - \frac{1}{2}z^{-1}\right)\left(1 - \frac{1}{4}z^{-1}\right)}$$

$$H_1(z) = \frac{7 - \frac{21}{4}z^{-1} + \frac{11}{8}z^{-2}}{\left(1 - \frac{1}{2}z^{-1}\right)} \quad \text{and} \quad H_2(z) = \frac{1}{\left(1 - \frac{1}{4}z^{-1}\right)}$$



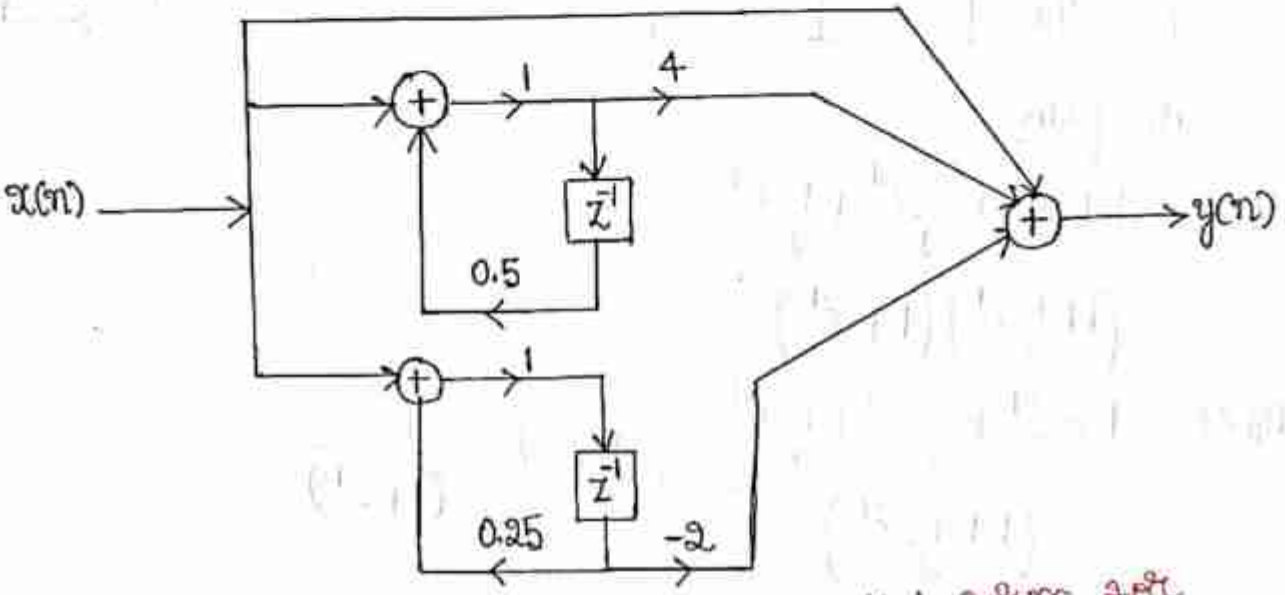
Scanned with CamScanner

Parallel form :-

Consider $H(z) = 3 + \frac{4z}{(z-0.5)} - \frac{2}{(z-0.25)}$

$$= 3 + \frac{4z}{z(1-0.5z^{-1})} - \frac{2}{z(1-0.25z^{-1})}$$

$$H(z) = 3 + \frac{4}{(1-0.5z^{-1})} + \frac{(-2z^{-1})}{(1-0.25z^{-1})}$$



5) Draw direct form, cascade and parallel form for

$$H(z) = \frac{z^3 + z^2 + \frac{3}{2}z + \frac{1}{2}}{z^3 + \frac{3}{2}z^2 + \frac{1}{2}z}$$

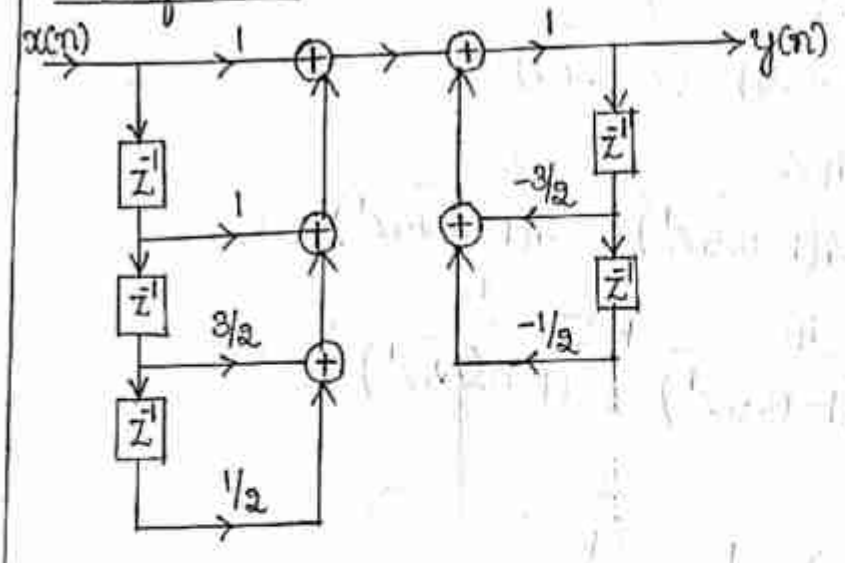
Sol^n :- Given $H(z) = \frac{z^3 + z^2 + \frac{3}{2}z + \frac{1}{2}}{z^3 + \frac{3}{2}z^2 + \frac{1}{2}z}$

$$= \frac{\cancel{z^3}\left[1 + z^{-1} + \frac{3}{2}z^{-2} + \frac{1}{2}z^{-3}\right]}{\cancel{z^3}\left[1 + \frac{3}{2}z^{-1} + \frac{1}{2}z^{-2}\right]}$$

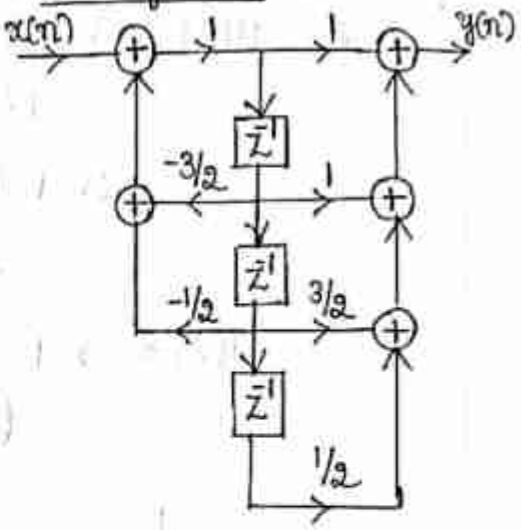
$$H(z) = \frac{1 + z^{-1} + \frac{3}{2}z^{-2} + \frac{1}{2}z^{-3}}{1 + \frac{3}{2}z^{-1} + \frac{1}{2}z^{-2}}$$

Scanned with CamScanner

Direct form-I :-



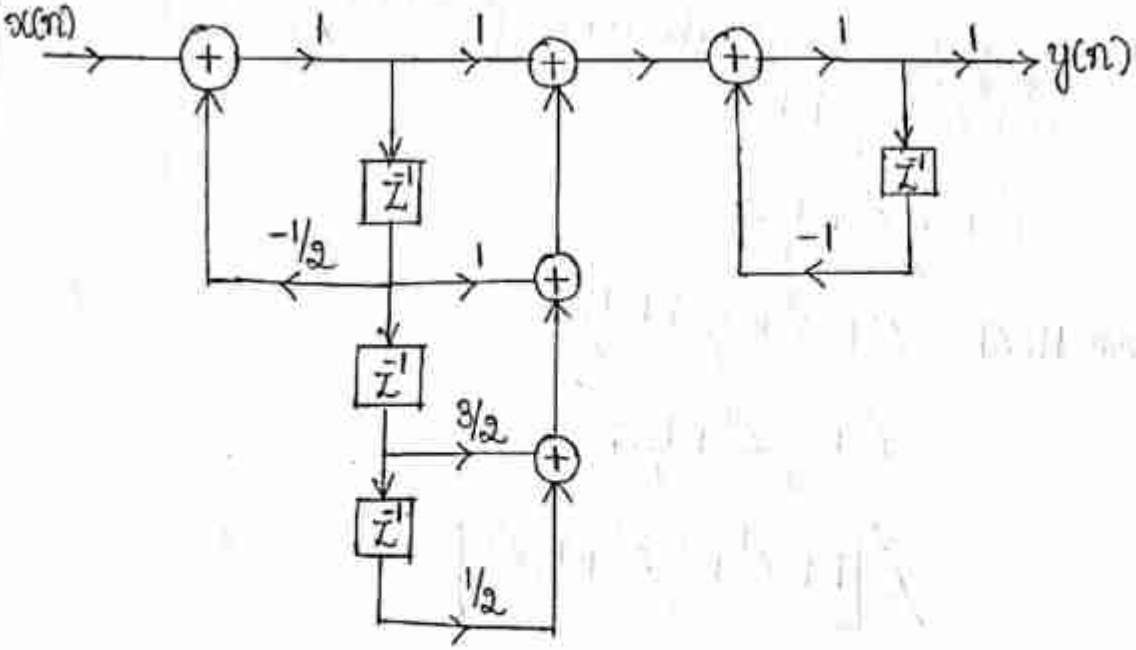
Direct form-II :-



Cascade form :-

$$H(z) = \frac{1 + z^{-1} + \frac{3}{2}z^{-2} + \frac{1}{2}z^{-3}}{\left(1+\frac{1}{2}z^{-1}\right)\left(1+z^{-1}\right)}$$

$$H_1(z) = \frac{1 + z^{-1} + \frac{3}{2}z^{-2} + \frac{1}{2}z^{-3}}{\left(1+\frac{1}{2}z^{-1}\right)} \quad ; \quad H_2(z) = \frac{1}{\left(1+z^{-1}\right)}$$



Scanned with CamScanner

Parallel form :-

$$H(z) = \frac{1+z^{-1}+\frac{3}{2}z^{-2}+\frac{1}{2}z^{-3}}{1+\frac{3}{2}z^{-1}+\frac{1}{2}z^{-2}}$$; Here M>N, use long division method

$$\frac{1}{2}z^{-2}+\frac{3}{2}z^{-1}+1 \;\Big|\; \frac{1}{2}z^{-3}+\frac{3}{2}z^{-2}+z^{-1}+1 \quad (\text{quotient } z^{-1})$$

$$\frac{1}{2}z^{-3}+\frac{3}{2}z^{-2}+z^{-1} \quad (-)$$

$$1$$

$$\therefore H(z) = z^{-1} + \frac{1}{\left(1+\frac{3}{2}z^{-1}+\frac{1}{2}z^{-2}\right)}$$

Consider, $$\frac{1}{1+\frac{3}{2}z^{-1}+\frac{1}{2}z^{-2}} = \frac{1}{\left(1+\frac{1}{2}z^{-1}\right)\left(1+z^{-1}\right)} = \frac{A}{\left(1+\frac{1}{2}z^{-1}\right)} + \frac{B}{\left(1+z^{-1}\right)}$$

$$1 = A\left(1+z^{-1}\right) + B\left(1+\frac{1}{2}z^{-1}\right) \longrightarrow (1)$$

put $z^{-1}=-2$ in Eq(1),

$1 = A[1-2]$

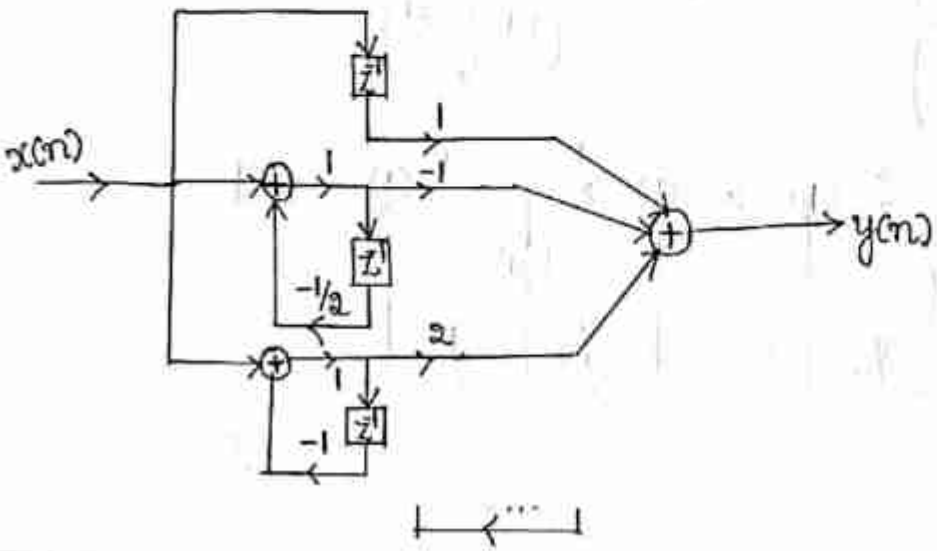
$\langle A=-1 \rangle$

put $z^{-1}=-1$ in Eq(1),

$1 = B\left[1+\frac{1}{2}(-1)\right]$

$\langle B=2 \rangle$

$$\therefore H(z) = z^{-1} + \frac{-1}{\left(1+\frac{1}{2}z^{-1}\right)} + \frac{2}{\left(1+z^{-1}\right)}$$



Scanned with CamScanner

6) For the given difference equation, Sketch direct form-I and II, Cascade and parallel form with single-pole 2nd subsystem:
$y(n) = 0.1y(n-1) + 0.2y(n-2) + 3x(n) + 3.6x(n-1) + 0.6x(n-2)$

Sol^n :-

Given $y(n) = 0.1y(n-1) + 0.2y(n-2) + 3x(n) + 3.6x(n-1) + 0.6x(n-2)$

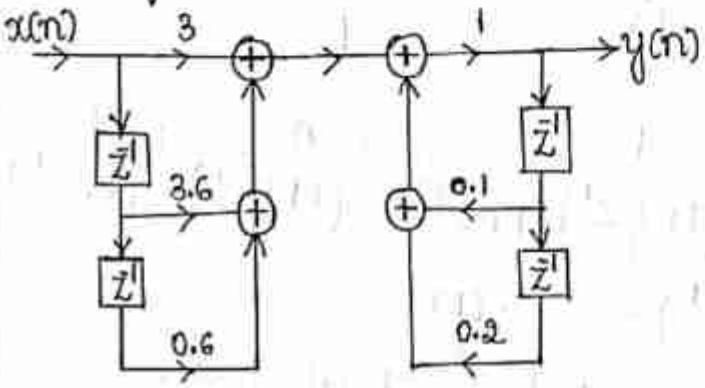
Apply z-transform on both the sides,

$$Y(z) = 0.1z^{-1}Y(z) + 0.2z^{-2}Y(z) + 3X(z) + 3.6z^{-1}X(z) + 0.6z^{-2}X(z)$$

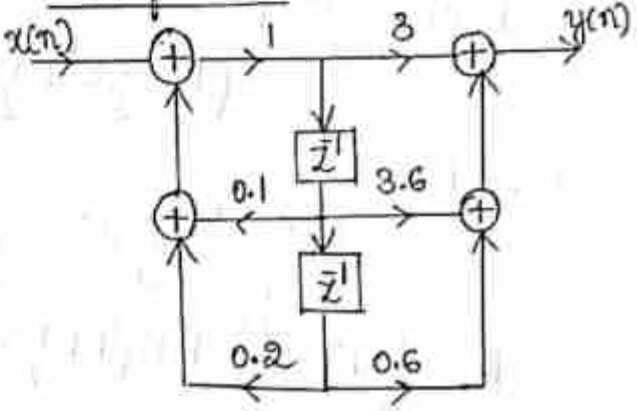
$$Y(z)[1 - 0.1z^{-1} - 0.2z^{-2}] = X(z)[3 + 3.6z^{-1} + 0.6z^{-2}]$$

$$\frac{Y(z)}{X(z)} = \frac{3 + 3.6z^{-1} + 0.6z^{-2}}{1 - 0.1z^{-1} - 0.2z^{-2}} = H(z)$$

Direct form-I :-



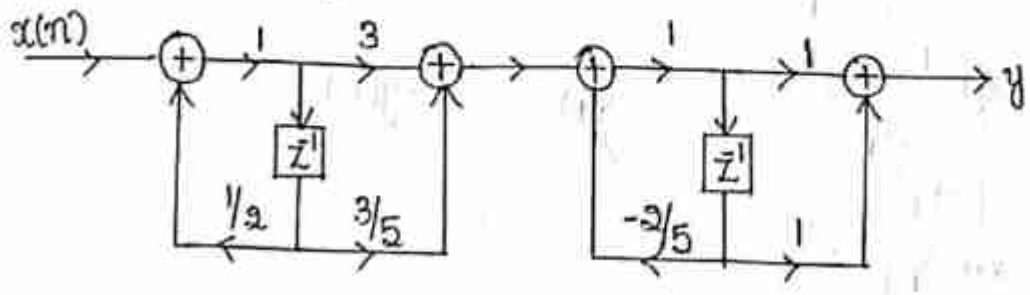
Direct form-II :-



Cascade form :-

$$H(z) = \frac{3 + \frac{18}{5}z^{-1} + \frac{3}{5}z^{-2}}{1 - \frac{1}{10}z^{-1} - \frac{1}{5}z^{-2}} = \frac{3\left(1 + \frac{1}{5}z^{-1}\right)\left(1 + z^{-1}\right)}{\left(1 - \frac{1}{2}z^{-1}\right)\left(1 + \frac{2}{5}z^{-1}\right)}$$

$$H_1(z) = \frac{3\left(1 + \frac{1}{5}z^{-1}\right)}{\left(1 - \frac{1}{2}z^{-1}\right)} \quad ; \quad H_2(z) = \frac{\left(1 + z^{-1}\right)}{\left(1 + \frac{2}{5}z^{-1}\right)}$$



Scanned with CamScanner

parallel form :—

$$H(z) = \frac{3 + \frac{18}{5}z^{-1} + \frac{3}{5}z^{-2}}{1 - \frac{1}{10}z^{-1} - \frac{1}{5}z^{-2}}$$; Here M=N, first use long division method.

$$\therefore H(z) = -3 + \frac{\left(6 + \frac{33}{10}z^{-1}\right)}{\left(1 - \frac{1}{10}z^{-1} - \frac{1}{5}z^{-2}\right)}$$

$-\frac{1}{5}z^{-2} - \frac{1}{10}z^{-1} + 1 \;\Big|\; \frac{3}{5}z^{-2} + \frac{18}{5}z^{-1} + 3$ (quotient -3)

$\frac{3}{5}z^{-2} + \frac{3}{10}z^{-1} - 3$ (−) (−) (+)

$\frac{33}{10}z^{-1} + 6$

Consider, $\frac{\left(6 + \frac{33}{10}z^{-1}\right)}{\left(1 - \frac{1}{10}z^{-1} - \frac{1}{5}z^{-2}\right)} = \frac{A}{\left(1 - \frac{1}{2}z^{-1}\right)} + \frac{B}{\left(1 + \frac{2}{5}z^{-1}\right)}$

$$6 + \frac{33}{10}z^{-1} = A\left(1 + \frac{2}{5}z^{-1}\right) + B\left(1 - \frac{1}{2}z^{-1}\right) \longrightarrow (1)$$

put $z^{-1} = 2$ in eq(1),

$$6 + \frac{33}{10}(2) = A\left[1 + \frac{2}{5}(2)\right]$$

$$\frac{65}{5} = A\left(\frac{9}{5}\right)$$

$$\langle A = 7 \rangle$$

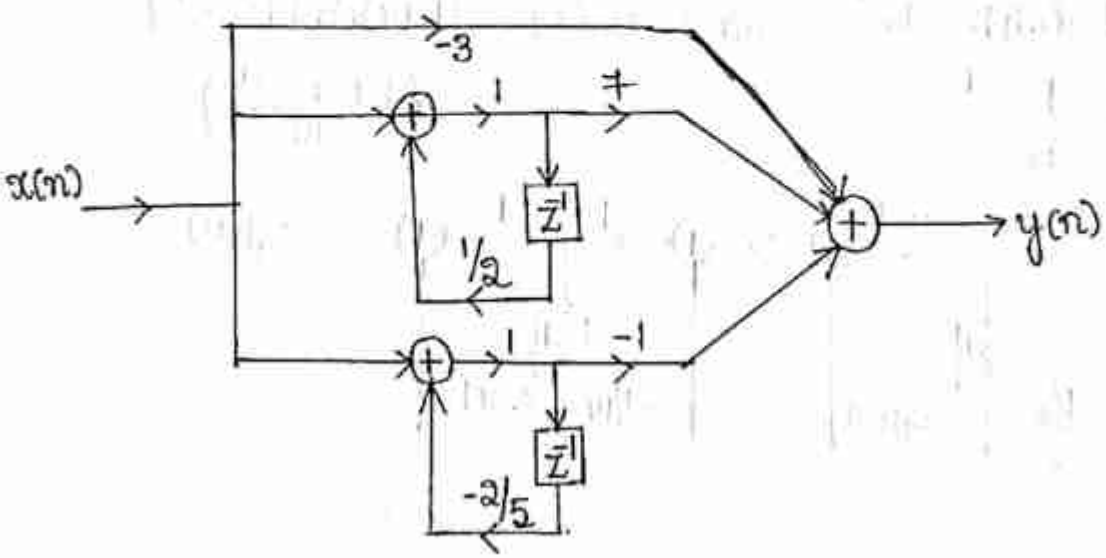
put $z^{-1} = -5/2$ in Eq(1),

$$6 + \frac{33}{10}\left(\frac{-5}{2}\right) = B\left[1 - \frac{1}{2}\left(\frac{-5}{2}\right)\right]$$

$$-\frac{9}{4} = B\left(\frac{9}{4}\right)$$

$$\langle B = -1 \rangle$$

$$\therefore H(z) = -3 + \frac{7}{\left(1 - \frac{1}{2}z^{-1}\right)} + \frac{(-1)}{\left(1 + \frac{2}{5}z^{-1}\right)}$$



Scanned with CamScanner

7) For the given difference equation, draw direct forms, cascade and parallel form:

$y(n) = -0.1y(n-1) + 0.72y(n-2) + 0.7x(n) - 0.25x(n-2)$

Soln:-

Given $y(n) = -0.1y(n-1) + 0.72y(n-2) + 0.7x(n) - 0.25x(n-2)$

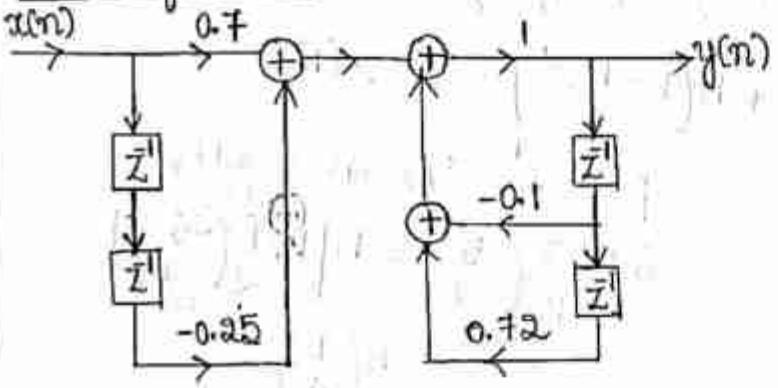
By Z-transform,

$$Y(z) = -0.1z^{-1}Y(z) + 0.72z^{-2}Y(z) + 0.7X(z) - 0.25z^{-2}X(z)$$

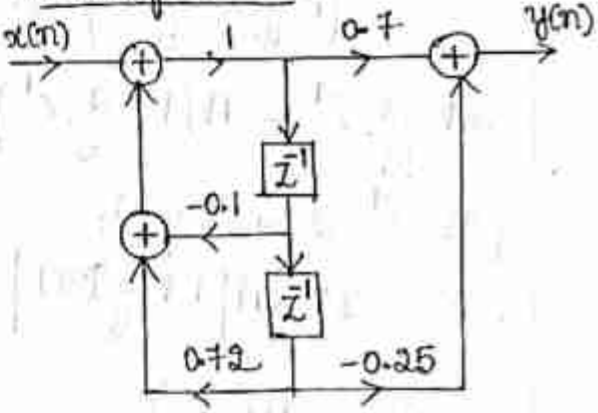
$$Y(z)[1 + 0.1z^{-1} - 0.72z^{-2}] = X(z)[0.7 - 0.25z^{-2}]$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{0.7 - 0.25z^{-2}}{1 + 0.1z^{-1} - 0.72z^{-2}}$$

Direct form-I :-



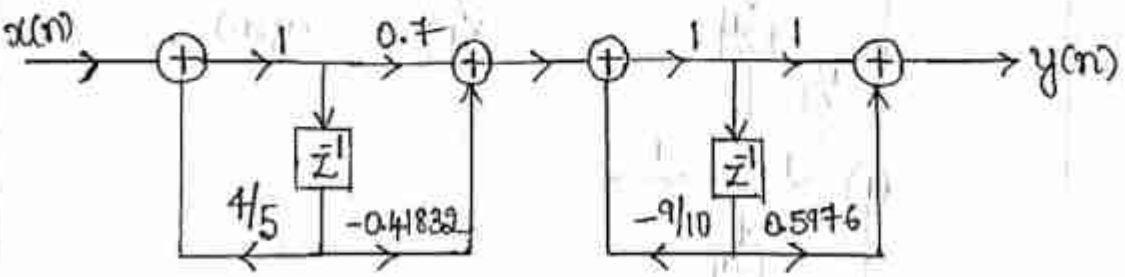
Direct form-II :-



Cascade form :-

$$H(z) = \frac{0.7 - 0.25z^{-2}}{1 + 0.1z^{-1} - 0.72z^{-2}} = \frac{0.7\left(1 - \frac{0.25}{0.7}z^{-2}\right)}{\left(1 - \frac{4}{5}z^{-1}\right)\left(1 + \frac{9}{10}z^{-1}\right)} = \frac{0.7(1 - 0.5976z^{-1})}{\left(1 - \frac{4}{5}z^{-1}\right)\left(1 + \frac{9}{10}z^{-1}\right)}$$

$$H_1(z) = \frac{0.7 - 0.41832z^{-1}}{1 - \frac{4}{5}z^{-1}} \quad \text{and} \quad H_2(z) = \frac{(1 + 0.5976z^{-1})}{\left(1 + \frac{9}{10}z^{-1}\right)}$$



Scanned with CamScanner

Parallel form :—

$$H(z) = \frac{\frac{7}{10} - \frac{1}{4}z^{-2}}{1 + \frac{1}{10}z^{-1} - \frac{18}{25}z^{-2}}$$; Here M = N, use long division method first.

$$H(z) = 0.3472 + \frac{(0.3528 - 0.03472z^{-1})}{\left(1 + \frac{1}{10}z^{-1} - \frac{18}{25}z^{-2}\right)}$$

$-\frac{18}{25}z^{-2} + \frac{1}{10}z^{-1} + 1$ | $-\frac{1}{4}z^{-2} + \frac{7}{10}$ | 0.3472

$-\frac{1}{4}z^{-2} + 0.03472z^{-1} + 0.3472$ (+) (−) (−)

$-0.03472z^{-1} + 0.3528$

Consider,

$$\frac{0.3528 - 0.03472z^{-1}}{\left(1 - \frac{4}{5}z^{-1}\right)\left(1 + \frac{9}{10}z^{-1}\right)} = \frac{A}{\left(1 - \frac{4}{5}z^{-1}\right)} + \frac{B}{\left(1 + \frac{9}{10}z^{-1}\right)}$$

$$0.3528 - 0.03472z^{-1} = A\left(1 + \frac{9}{10}z^{-1}\right) + B\left(1 - \frac{4}{5}z^{-1}\right) \longrightarrow (1)$$

put $z^{-1} = \frac{5}{4}$ in eq(1),

$$0.3528 - 0.03472\left(\frac{5}{4}\right) = A\left[1 + \frac{9}{10}\left(\frac{5}{4}\right)\right]$$

$$0.3094 = A(2.125)$$

$$\langle A = 0.1456 \rangle$$

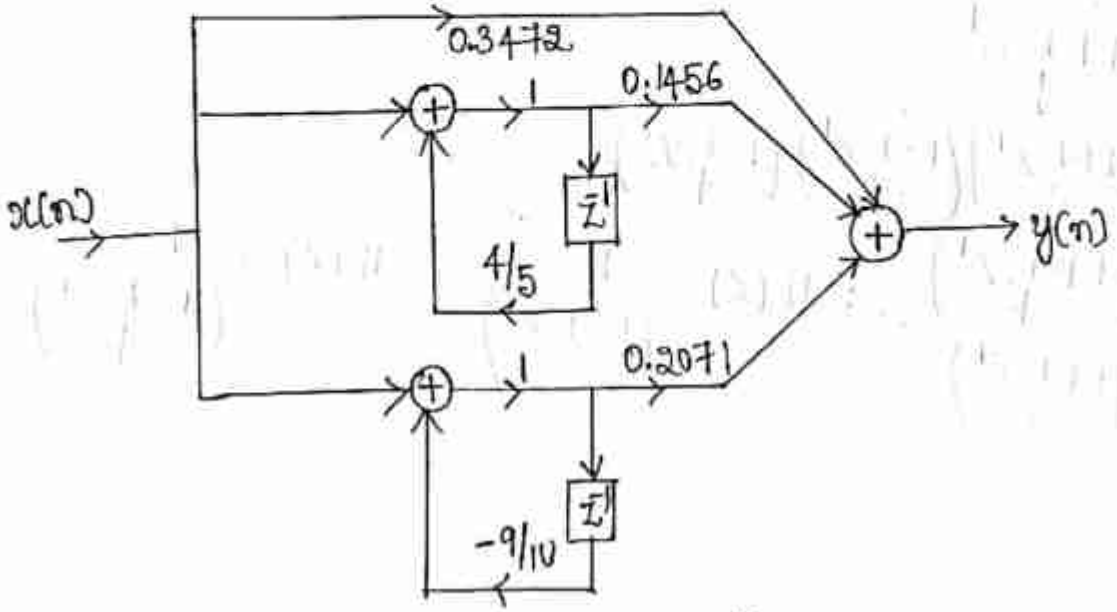
put $z^{-1} = \frac{-10}{9}$ in eq(1),

$$0.3528 - 0.03472\left(\frac{-10}{9}\right) = B\left[1 - \frac{4}{5}\left(\frac{-10}{9}\right)\right]$$

$$0.3914 = B(1.89)$$

$$\langle B = 0.2071 \rangle$$

$$\therefore H(z) = 0.3472 + \frac{0.1456}{\left(1 - \frac{4}{5}z^{-1}\right)} + \frac{0.2071}{\left(1 + \frac{9}{10}z^{-1}\right)}$$



Scanned with CamScanner

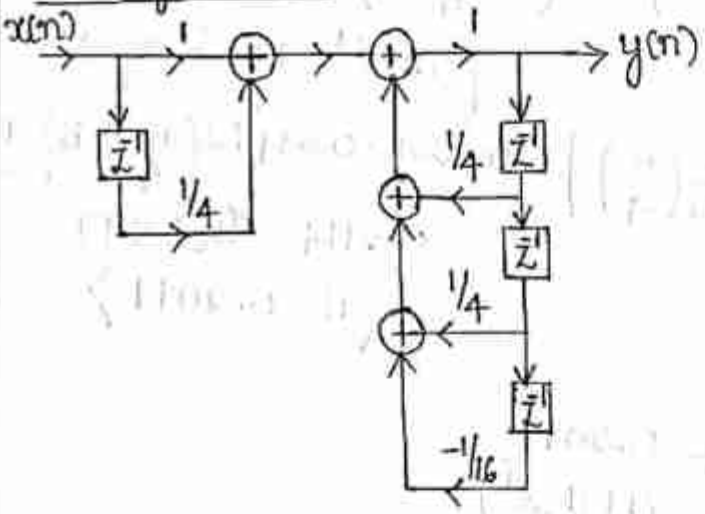
8) For the given system function, draw direct form-I and II, Cascade and parallel structure:

$$H(z) = \frac{\left(1+\frac{1}{4}z^{-1}\right)}{\left(1+\frac{1}{2}z^{-1}\right)\left(1-\frac{3}{4}z^{-1}+\frac{1}{8}z^{-2}\right)}$$

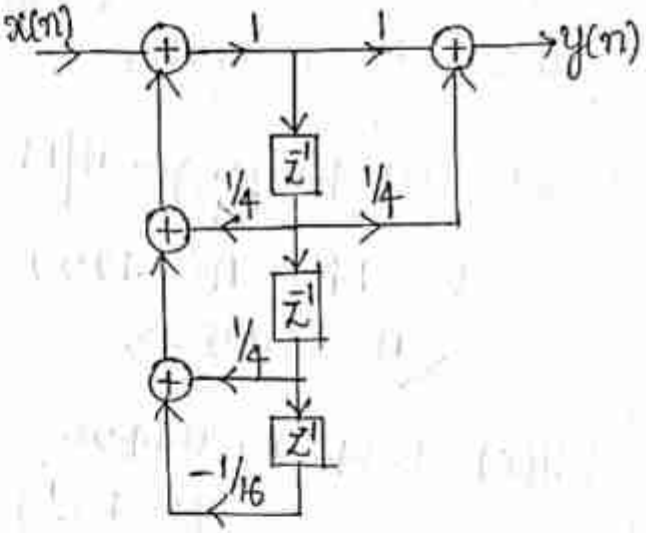
Sol^n:- $$H(z) = \frac{\left(1+\frac{1}{4}z^{-1}\right)}{1-\frac{3}{4}z^{-1}+\frac{1}{8}z^{-2}+\frac{1}{2}z^{-1}-\frac{3}{8}z^{-2}+\frac{1}{16}z^{-3}}$$

$$= \frac{\left(1+\frac{1}{4}z^{-1}\right)}{\left(1-\frac{1}{4}z^{-1}-\frac{1}{4}z^{-2}+\frac{1}{16}z^{-3}\right)}$$

Direct form-I :-



Direct form-II :-

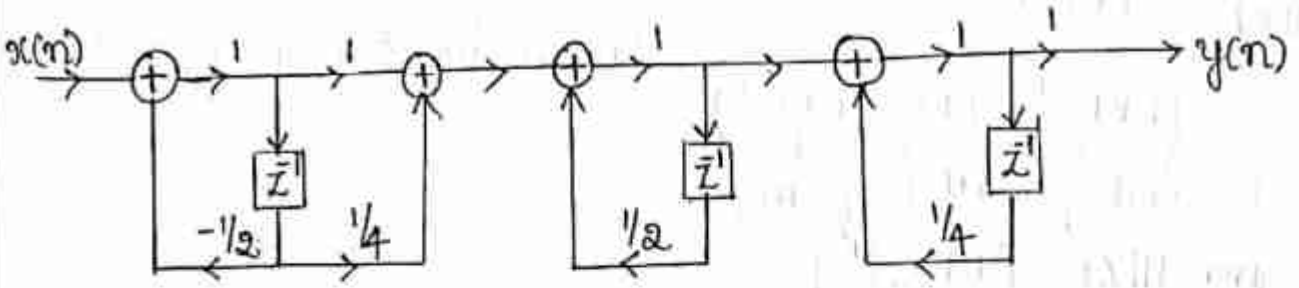


Cascade form :-

$$H(z) = \frac{1+\frac{1}{4}z^{-1}}{\left(1+\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)}$$

$$H_1(z) = \frac{\left(1+\frac{1}{4}z^{-1}\right)}{\left(1+\frac{1}{2}z^{-1}\right)} \; ; \; H_2(z) = \frac{1}{\left(1-\frac{1}{2}z^{-1}\right)} \; ; \; H_3(z) = \frac{1}{\left(1-\frac{1}{4}z^{-1}\right)}$$

Scanned with CamScanner



Parallel form :-

$$H(z) = \frac{1+\frac{1}{4}z^{-1}}{\left(1+\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)} = \frac{A}{\left(1+\frac{1}{2}z^{-1}\right)} + \frac{B}{\left(1-\frac{1}{2}z^{-1}\right)} + \frac{C}{\left(1-\frac{1}{4}z^{-1}\right)}$$

$$1+\frac{1}{4}z^{-1} = A\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right) + B\left(1+\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right) + C\left(1+\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}\right) \quad \hookrightarrow (1)$$

put $z^{-1} = -2$ in Eq(1),

$$1+\frac{1}{4}(-2) = A\left[1-\frac{1}{2}(-2)\right]\left[1-\frac{1}{4}(-2)\right]$$

$$\frac{1}{2} = A(2)\left(\frac{3}{2}\right)$$

$$\langle A = \frac{1}{6} \rangle$$

put $z^{-1} = 2$ in Eq(1),

$$1+\frac{1}{4}(2) = B\left[1+\frac{1}{2}(2)\right]\left[1-\frac{1}{4}(2)\right]$$

$$\frac{3}{2} = B(2)\left(\frac{1}{2}\right)$$

$$\langle B = \frac{3}{2} \rangle$$

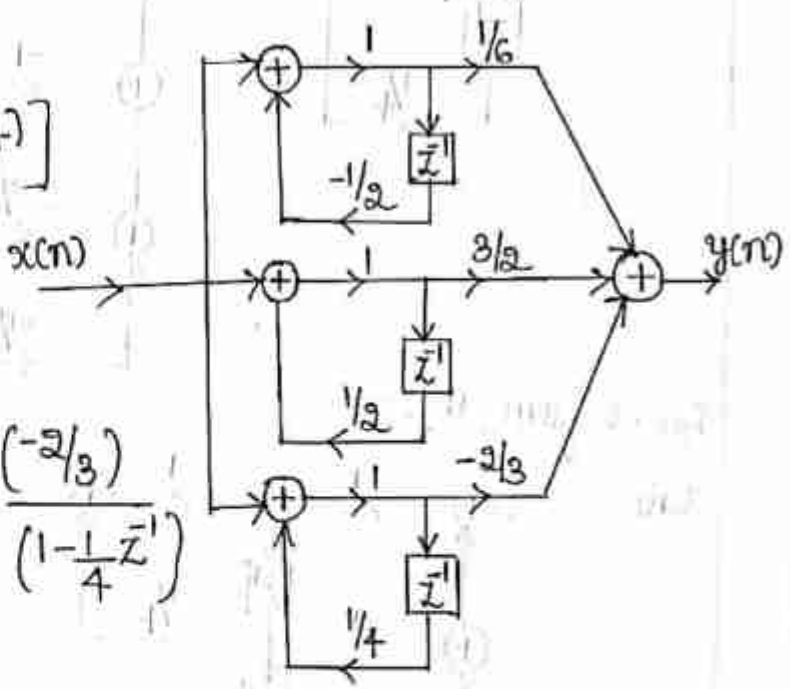
put $z^{-1} = 4$ in Eq(1),

$$1+\frac{1}{4}(4) = C\left[1+\frac{1}{2}(4)\right]\left[1-\frac{1}{2}(4)\right]$$

$$2 = C(3)(-1)$$

$$\langle C = -2/3 \rangle$$

$$\therefore H(z) = \frac{(1/6)}{\left(1+\frac{1}{2}z^{-1}\right)} + \frac{(3/2)}{\left(1-\frac{1}{2}z^{-1}\right)} + \frac{(-2/3)}{\left(1-\frac{1}{4}z^{-1}\right)}$$



Scanned with CamScanner

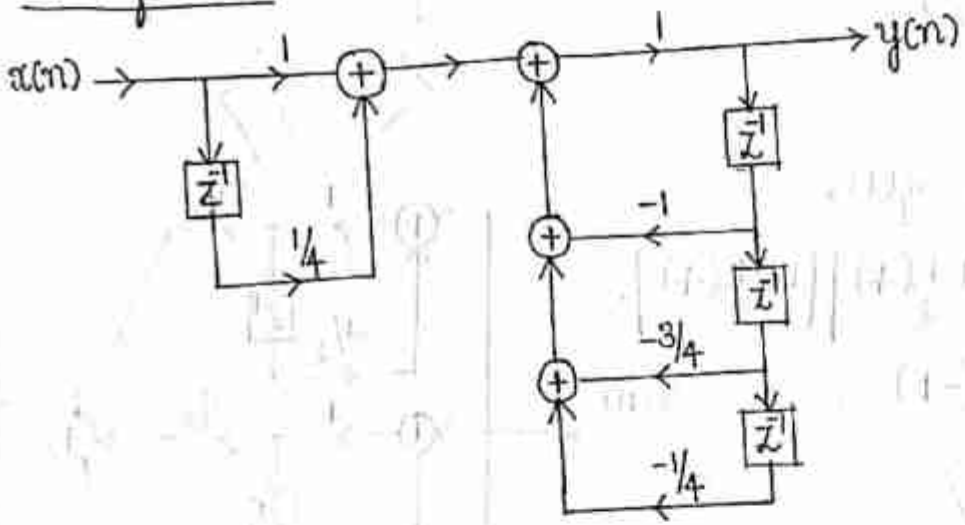
9) If $H(z) = \dfrac{1+\frac{1}{4}z^{-1}}{\left(1+\frac{1}{2}z^{-1}\right)\left(1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}\right)}$; Draw direct form-I and II, Cascade and parallel form.

Sol^n :- Given $H(z) = \dfrac{\left(1+\frac{1}{4}z^{-1}\right)}{\left(1+\frac{1}{2}z^{-1}\right)\left(1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}\right)}$

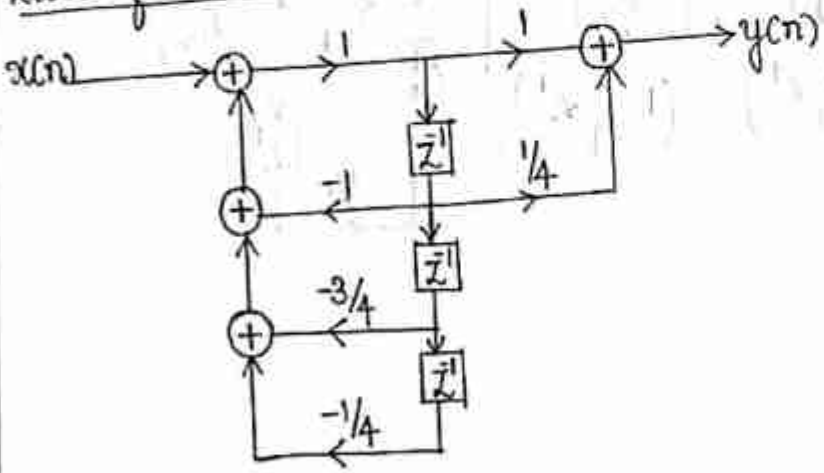
$$= \frac{\left(1+\frac{1}{4}z^{-1}\right)}{1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}+\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}+\frac{1}{4}z^{-3}}$$

$$H(z) = \frac{\left(1+\frac{1}{4}z^{-1}\right)}{\left(1+z^{-1}+\frac{3}{4}z^{-2}+\frac{1}{4}z^{-3}\right)}$$

Direct form-I :-



Direct form-II :-

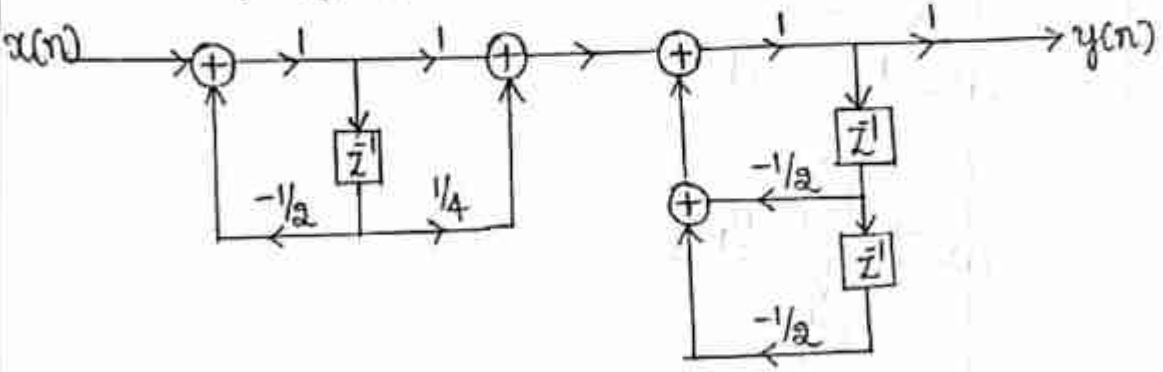


Scanned with CamScanner

Cascade form :-

$$H(z) = \frac{\left(1+\frac{1}{4}z^{-1}\right)}{\left(1+z^{-1}+\frac{3}{4}z^{-2}+\frac{1}{4}z^{-3}\right)} = \frac{\left(1+\frac{1}{4}z^{-1}\right)}{\left(1+\frac{1}{2}z^{-1}\right)\left(1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}\right)}$$

$$H_1(z) = \frac{\left(1+\frac{1}{4}z^{-1}\right)}{\left(1+\frac{1}{2}z^{-1}\right)} \quad \text{and} \quad H_2(z) = \frac{1}{\left(1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}\right)}$$



Parallel form :-

$$H(z) = \frac{\left(1+\frac{1}{4}z^{-1}\right)}{\left(1+\frac{1}{2}z^{-1}\right)\left(1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}\right)} = \frac{A}{\left(1+\frac{1}{2}z^{-1}\right)} + \frac{(Bz^{-1}+C)}{\left(1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}\right)}$$

$$1+\frac{1}{4}z^{-1} = A\left(1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}\right) + (Bz^{-1}+C)\left(1+\frac{1}{2}z^{-1}\right) \longrightarrow (1)$$

put $z^{-1} = -2$ in Eq(1),

$$1+\frac{1}{4}(-2) = A\left[1+\frac{1}{2}(-2)+\frac{1}{2}(-2)^2\right]$$

$$\frac{1}{2} = A(2)$$

$$\langle A = 1/4 \rangle$$

Comparing Co-efficients :

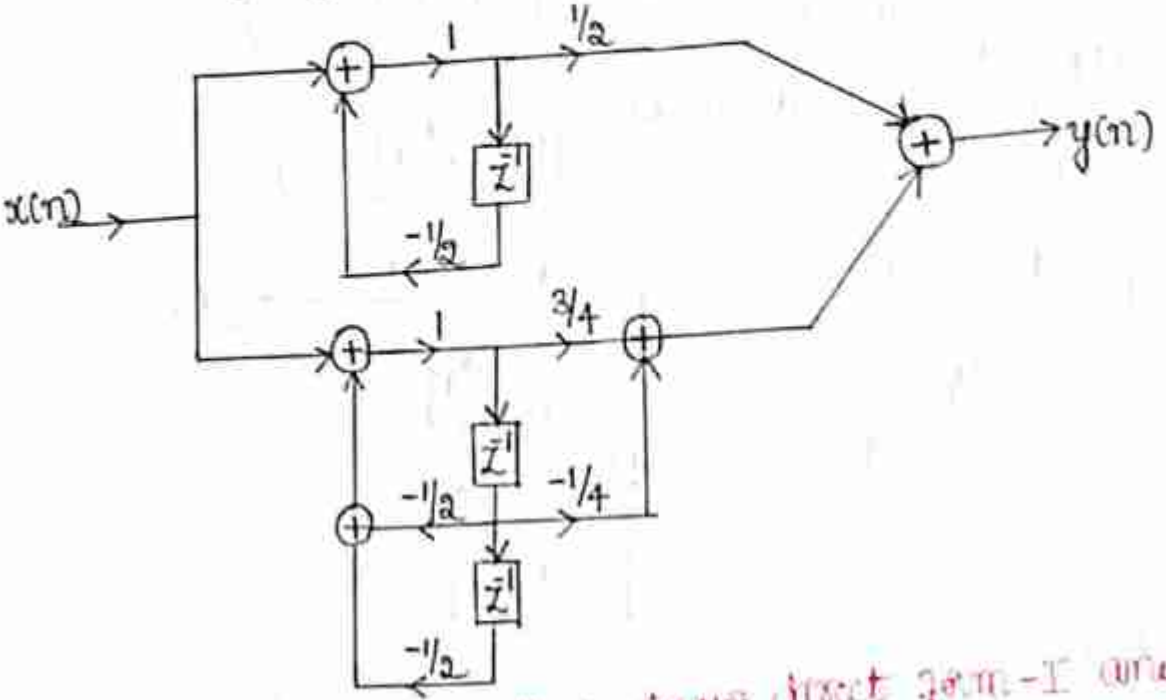
1) Constant : $1 = A + C \Rightarrow C = 1 - \frac{1}{4} = \frac{3}{4}$

2) z^{-1} : $\frac{1}{4} = \frac{1}{2}A + B + \frac{1}{2}C \Rightarrow B = \frac{1}{4} - \frac{1}{2}\left(\frac{1}{4}\right) - \frac{1}{2}\left(\frac{3}{4}\right)$

$$\left\langle B = -\frac{1}{4} \right\rangle$$

Scanned with CamScanner

$$\therefore H(z) = \frac{\left(\frac{1}{2}\right)}{\left(1+\frac{1}{2}z^{-1}\right)} + \frac{\left(\frac{3}{4}-\frac{1}{4}z^{-1}\right)}{\left(1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}\right)}$$



10) For the given system function, discuss direct form-I and II, Cascade form and parallel form:

$$H(z) = \frac{(1-z^{-1})^3}{\left(1-\frac{1}{4}z^{-1}\right)\left(1-z^{-1}+\frac{1}{2}z^{-2}\right)}$$

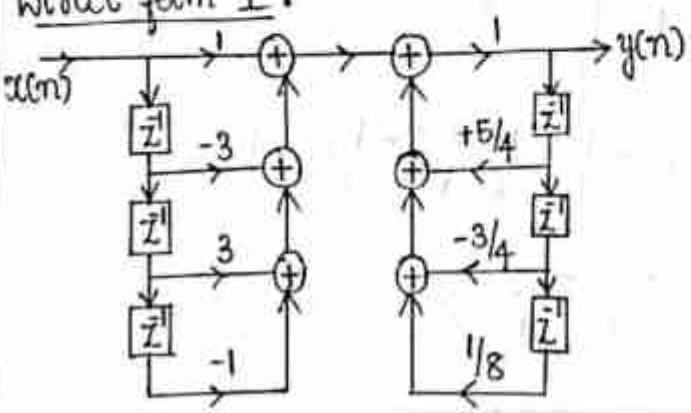
$$(a-b)^3 = a^3-b^3-3ab(a-b)$$

Soln:-

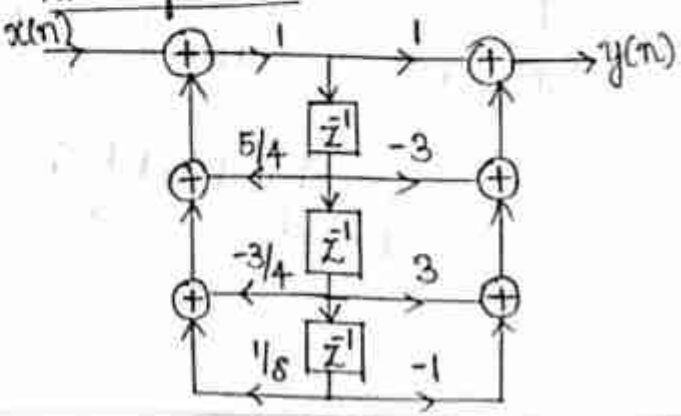
$$H(z) = \frac{1-z^{-3}-3z^{-1}(1-z^{-1})}{1-z^{-1}+\frac{1}{2}z^{-2}-\frac{1}{4}z^{-1}+\frac{1}{4}z^{-2}-\frac{1}{8}z^{-3}}$$

$$H(z) = \frac{1-3z^{-1}+3z^{-2}-z^{-3}}{1-\frac{5}{4}z^{-1}+\frac{3}{4}z^{-2}-\frac{1}{8}z^{-3}}$$

Direct form-I :-



Direct form-II :-

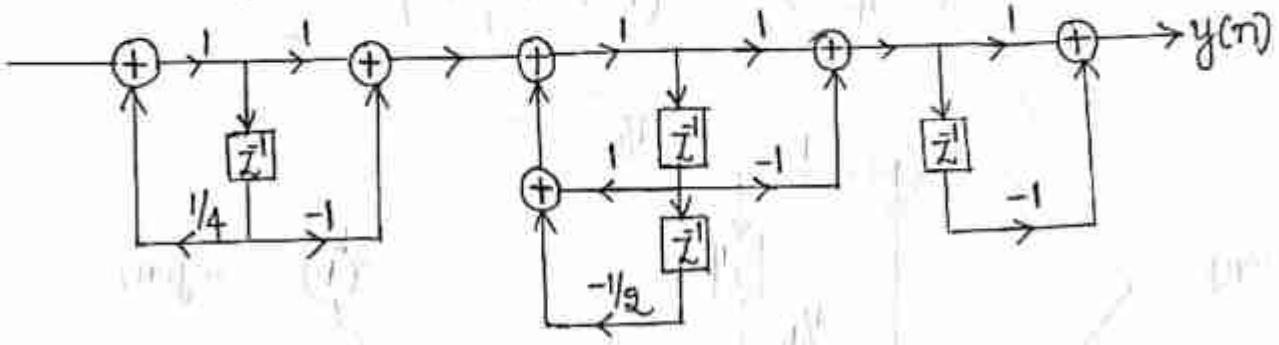


Scanned with CamScanner

Cascade form:-

$$H(z) = \frac{(1-z^{-1})^3}{(1-\frac{1}{4}z^{-1})(1-z^{-1}+\frac{1}{2}z^{-2})}$$

$$H_1(z) = \frac{(1-z^{-1})}{(1-\frac{1}{4}z^{-1})} \; ; \; H_2(z) = \frac{(1-z^{-1})}{(1-z^{-1}+\frac{1}{2}z^{-2})} \text{ and } H_3(z) = (1-z^{-1})$$



Parallel form:-

$$H(z) = \frac{1(-3z^{-1}+3z^{-2}-z^{-3}}{1-\frac{5}{4}z^{-1}+\frac{3}{4}z^{-2}-\frac{1}{8}z^{-3}} \; ; \text{ Here } M=N, \text{ use long division method.}$$

$$-\frac{1}{8}z^{-3}+\frac{3}{4}z^{-2}-\frac{5}{4}z^{-1}+1 \;\Big|\; -z^{-3}+3z^{-2}-3z^{-1}+1$$

$$-z^{-3}+6z^{-2}-10z^{-1}+8 \quad (8)$$

(+) (−) (+) (−)

$$-3z^{-2}+7z^{-1}-7$$

$$\therefore H(z) = 8 + \frac{(-7+7z^{-1}-3z^{-2})}{(1-\frac{5}{4}z^{-1}+\frac{3}{4}z^{-2}-\frac{1}{8}z^{-3})}$$

$$-7+7z^{-1}-3z^{-2} = A(1-z^{-1}+\frac{1}{2}z^{-2}) + (Bz^{-1}+C)(1-\frac{1}{4}z^{-1}) \longrightarrow (1)$$

put $z^{-1}=4$ in Eq(1),

$$-7+7(4)-3(4)^2 = A\left[1-4+\frac{1}{2}(4)\right]$$

$$-21 = A(5)$$

$$\boxed{A = -\frac{21}{5}}$$

Comparing Co-efficients:

1) Constant : $-7 = A+C \Rightarrow C = -7-\left(-\frac{21}{5}\right)$

$$\boxed{C = -\frac{8}{5}}$$

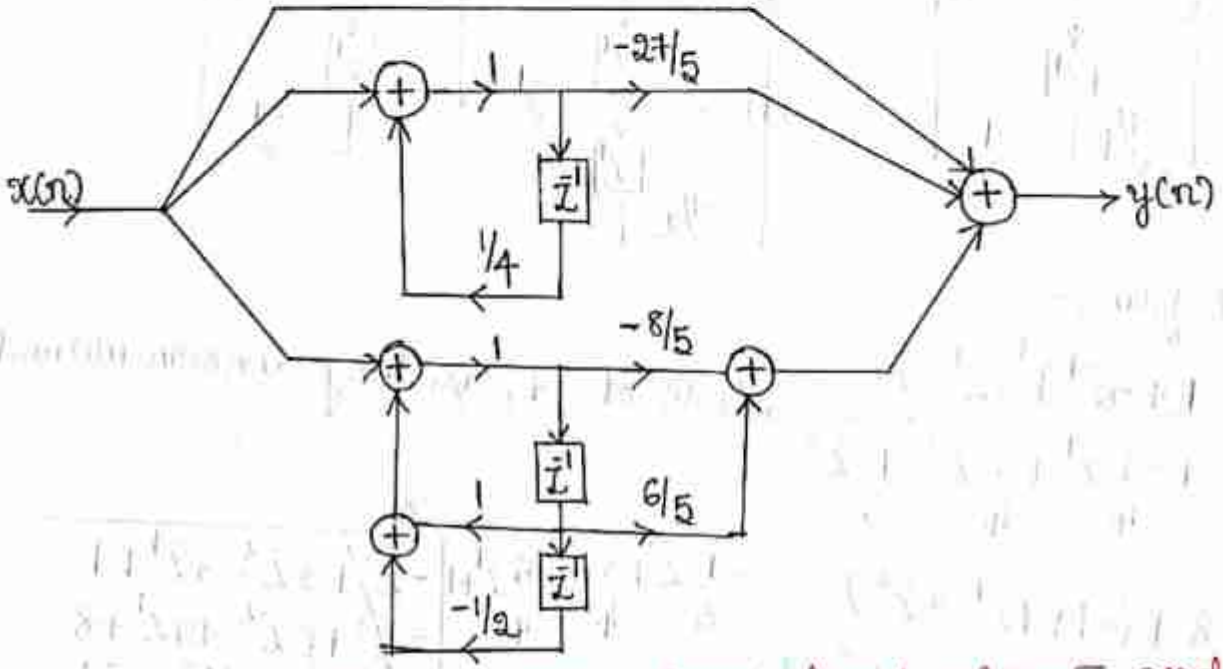
Scanned with CamScanner

2) $(z^{-1}$: $-7 = -A + B - \frac{1}{4}C \Rightarrow B = -7 + A + \frac{1}{4}C$

$= -7 - \frac{27}{5} + \frac{1}{4}\left(\frac{-8}{5}\right)$

$\left\langle B = \frac{6}{5} \right\rangle$

$\therefore H(z) = 8 + \frac{(-27/5)}{\left(1 - \frac{1}{4}z^{-1}\right)} + \frac{\left(\frac{-8}{5} + \frac{6}{5}z^{-1}\right)}{\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)}$



11) For the given system function, draw direct form-I and II, Cascade and parallel form : $H(z) = \frac{8z^3 - 4z^2 + 11z - 2}{\left(z - \frac{1}{4}\right)\left(z^2 - z + \frac{1}{2}\right)}$ (Series)

Sol^n:- Given $H(z) = \frac{8z^3 - 4z^2 + 11z - 2}{\left(z - \frac{1}{4}\right)\left(z^2 - z + \frac{1}{2}\right)}$

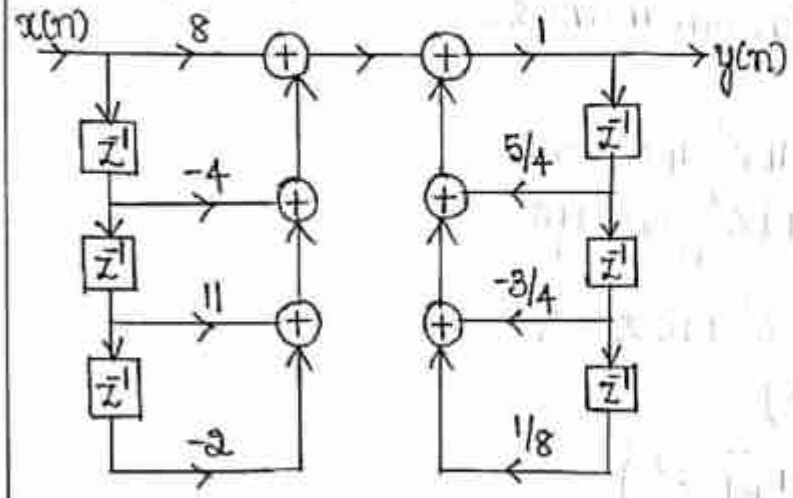
$= \frac{8z^3 - 4z^2 + 11z - 2}{z^3 - z^2 + \frac{1}{2}z - \frac{1}{4}z^2 + \frac{1}{4}z - \frac{1}{8}}$

$= \frac{8z^3 - 4z^2 + 11z - 2}{z^3 - \frac{5}{4}z + \frac{3}{4}z - \frac{1}{8}} = \frac{z^3\left[8 - 4z^{-1} + 11z^{-2} - 2z^{-3}\right]}{z^3\left[1 - \frac{5}{4}z^{-1} + \frac{3}{4}z^{-2} - \frac{1}{8}z^{-3}\right]}$

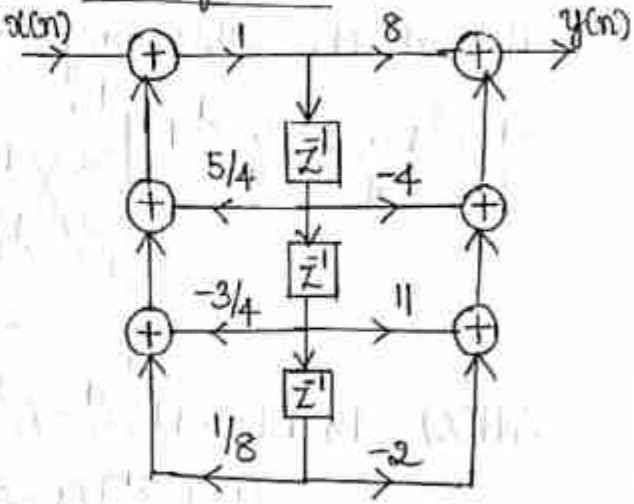
Scanned with CamScanner

$$\therefore H(z) = \frac{8 - 4z^{-1} + 11z^{-2} - 2z^{-3}}{1 - \frac{5}{4}z^{-1} + \frac{3}{4}z^{-2} - \frac{1}{8}z^{-3}}$$

Direct form-I :-



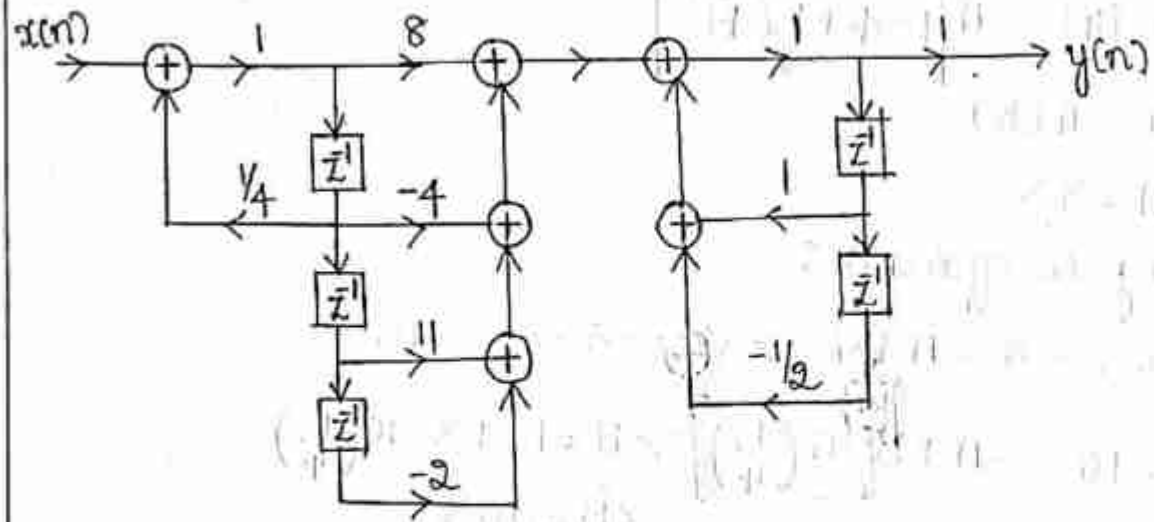
Direct form-II :-



Cascade form :-

$$H(z) = \frac{8z^3 - 4z^2 + 11z - 2}{\left(z - \frac{1}{4}\right)\left(z^2 - z + \frac{1}{2}\right)} = \frac{z^3\left(8 - 4z^{-1} + 11z^{-2} - 2z^{-3}\right)}{z\left(1 - \frac{1}{4}z^{-1}\right)z^2\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)}$$

$$H_1(z) = \frac{\left(8 - 4z^{-1} + 11z^{-2} - 2z^{-3}\right)}{\left(1 - \frac{1}{4}z^{-1}\right)} \quad ; \quad H_2(z) = \frac{1}{\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)}$$



Scanned with CamScanner

Parallel form :-

$$H(z) = \frac{8 - 4z^{-1} + 11z^{-2} - 2z^{-3}}{1 - \frac{5}{4}z^{-1} + \frac{3}{4}z^{-2} - \frac{1}{8}z^{-3}}$$

Here M = N, ∴ use long division method.

$$-\frac{1}{8}z^{-3} + \frac{3}{4}z^{-2} - \frac{5}{4}z^{-1} + 1 \;\Big|\; -2z^{-3} + 11z^{-2} - 4z^{-1} + 8 \quad (16$$

$$-2z^{-3} + 12z^{-2} - 20z^{-1} + 16 \quad (+)\ (-)\ (+)\ (-)$$

$$-z^{-2} + 16z^{-1} - 8$$

$$\therefore H(z) = 16 + \frac{(-8 + 16z^{-1} - z^{-2})}{\left(1 - \frac{1}{4}z^{-1}\right)\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)}$$

Consider, $\frac{-8 + 16z^{-1} - z^{-2}}{\left(1 - \frac{1}{4}z^{-1}\right)\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)} = \frac{A}{\left(1 - \frac{1}{4}z^{-1}\right)} + \frac{(Bz^{-1} + C)}{\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)}$

(1 − z⁻¹ + ½z⁻² : biq)

$$-8 + 16z^{-1} - z^{-2} = A\left[1 - z^{-1} + \frac{1}{2}z^{-2}\right] + (Bz^{-1} + C)\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right) \longrightarrow (1)$$

put $z^{-1} = 4$ in eq(1),

$$-8 + 16(4) - (4)^2 = A\left[1 - 4 + \frac{1}{2}(4)^2\right]$$

$$40 = A(5)$$

$$\langle A = 8 \rangle$$

Comparing Co-efficients :

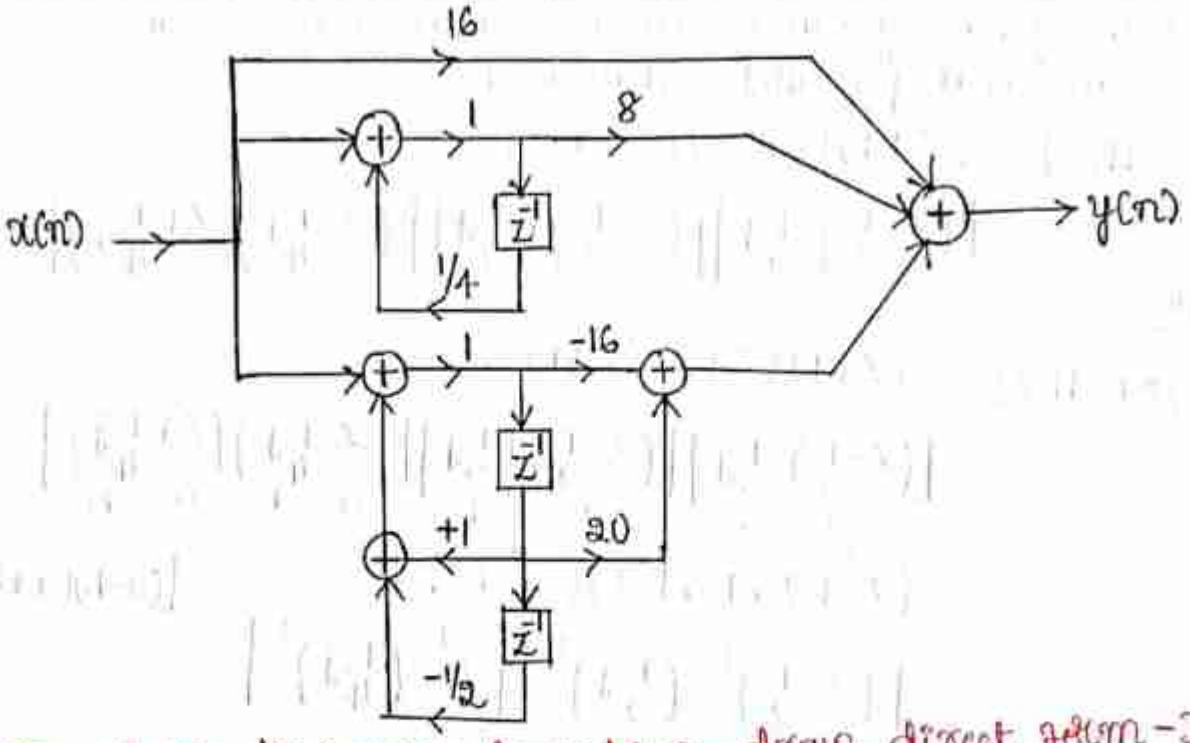
1) Constant : $-8 = A + C \Rightarrow C = -8 - 8 = \underline{\underline{-16}}$

2) z^{-1} : $16 = -A + B - C\left(\frac{1}{4}\right) \Rightarrow B = 16 + 8 - 16\left(\frac{1}{4}\right)$

$$\langle B = 20 \rangle$$

$$\therefore H(z) = 16 + \frac{8}{\left(1 - \frac{1}{4}z^{-1}\right)} + \frac{(-16 + 20z^{-1})}{\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)}$$

Scanned with CamScanner



12) For the given difference equation, draw direct form-I and canonical form (direct form-II):

$y(n) = 2r\cos w_0\, y(n-1) - r^2 y(n-2) + x(n) - r\cos w_0\, x(n-1)$

Sol^n:-

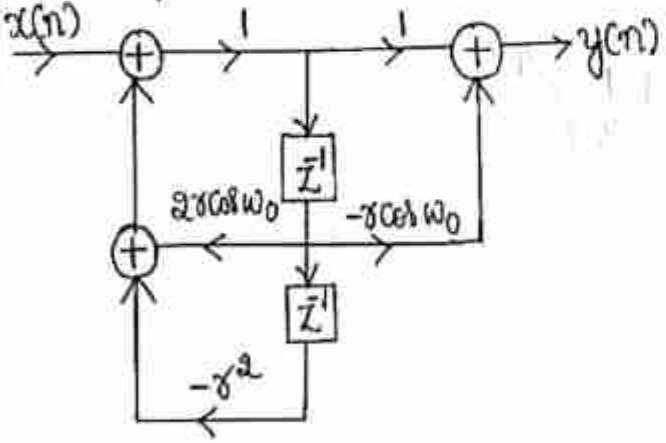
Given $y(n) = 2r\cos w_0\, y(n-1) - r^2 y(n-2) + x(n) - r\cos w_0\, x(n-1)$

Applying Z-transform on both the sides,

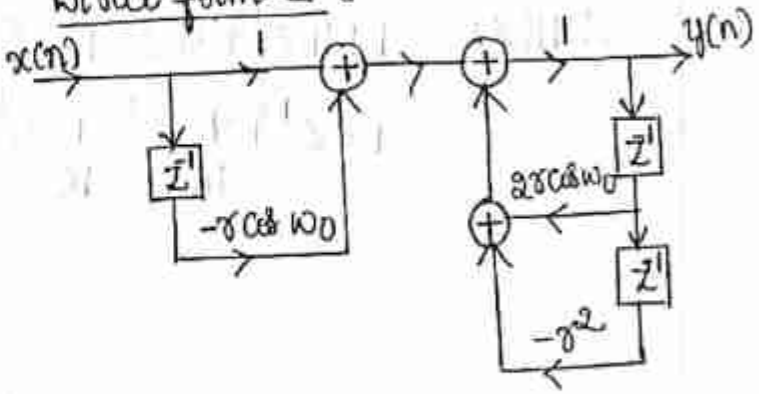
$Y(z) = 2r\cos w_0\, z^{-1} Y(z) - r^2 z^{-2} Y(z) + X(z) - r\cos w_0\, z^{-1} X(z)$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1 - r\cos w_0\, z^{-1}}{1 - 2r\cos w_0\, z^{-1} + r^2 z^{-2}}$$

Direct form-II :-



Direct form-I :-



Scanned with CamScanner

13) For the given system function, draw direct form-I & II, Cascade and parallel connection:

$$H(z) = \frac{(z+1)(z+2)(z+1)z}{\left[\left(z-\frac{1}{2}\right)-\frac{1}{2}j\right]\left[\left(z-\frac{1}{2}\right)+\frac{1}{2}j\right]\left[\left(z-\frac{1}{4}j\right)\left(z+\frac{1}{4}j\right)\right]}$$

Sol^n:-

Given $H(z) = \dfrac{(z+1)(z+2)(z+1)z}{\left[\left(z-\frac{1}{2}\right)-\frac{1}{2}j\right]\left[\left(z-\frac{1}{2}\right)+\frac{1}{2}j\right]\left[\left(z-\frac{1}{4}j\right)\left(z+\frac{1}{4}j\right)\right]}$

$$= \frac{(z^2+2z+z+2)(z^2+z)}{\left[\left(z-\frac{1}{2}\right)^2-\left(\frac{1}{2}j\right)^2\right]\left[z^2-\left(\frac{1}{4}j\right)^2\right]} \qquad [(a-b)(a+b)=a^2-b^2]$$

$$= \frac{z^4+z^3+3z^3+3z^2+2z^2+2z}{\left(z^2+\frac{1}{4}-z+\frac{1}{4}\right)\left(z^2+\frac{1}{16}\right)}$$

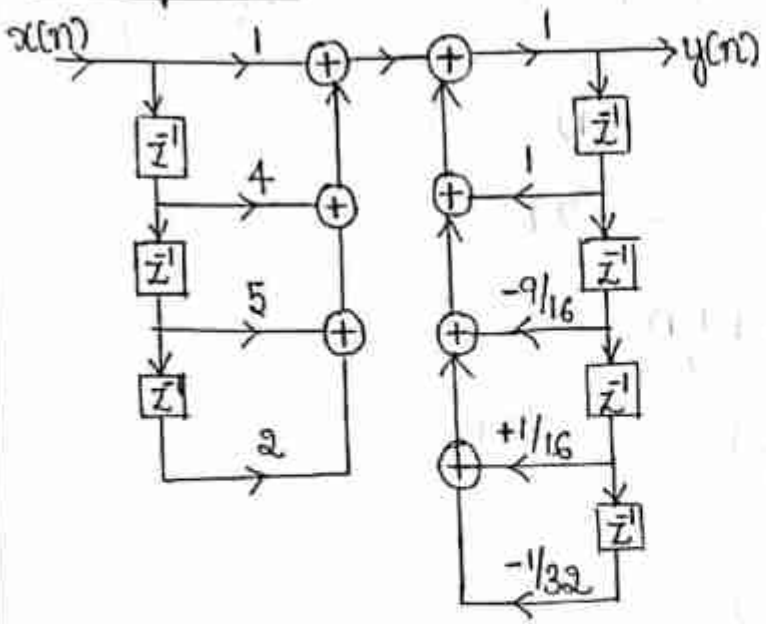
$$= \frac{z^4+4z^3+5z^2+2z}{\left(z^4+\frac{1}{16}z^2+\frac{1}{2}z^2+\frac{1}{32}-z^3-\frac{1}{16}z\right)}$$

$$= \frac{z^4+4z^3+5z^2+2z}{z^4-z^3+\frac{9}{16}z^2-\frac{1}{16}z+\frac{1}{32}} = \frac{z^4\left(1+4z^{-1}+5z^{-2}+2z^{-3}\right)}{z^4\left(1-z^{-1}+\frac{9}{16}z^{-2}-\frac{1}{16}z^{-3}+\frac{1}{32}z^{-4}\right)}$$

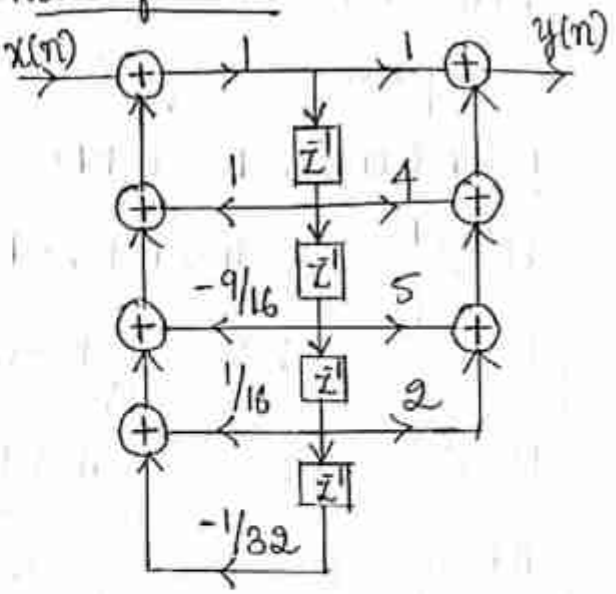
$$\therefore H(z) = \frac{1+4z^{-1}+5z^{-2}+2z^{-3}}{1-z^{-1}+\frac{9}{16}z^{-2}-\frac{1}{16}z^{-3}+\frac{1}{32}z^{-4}}$$

Scanned with CamScanner

Direct form-I :-



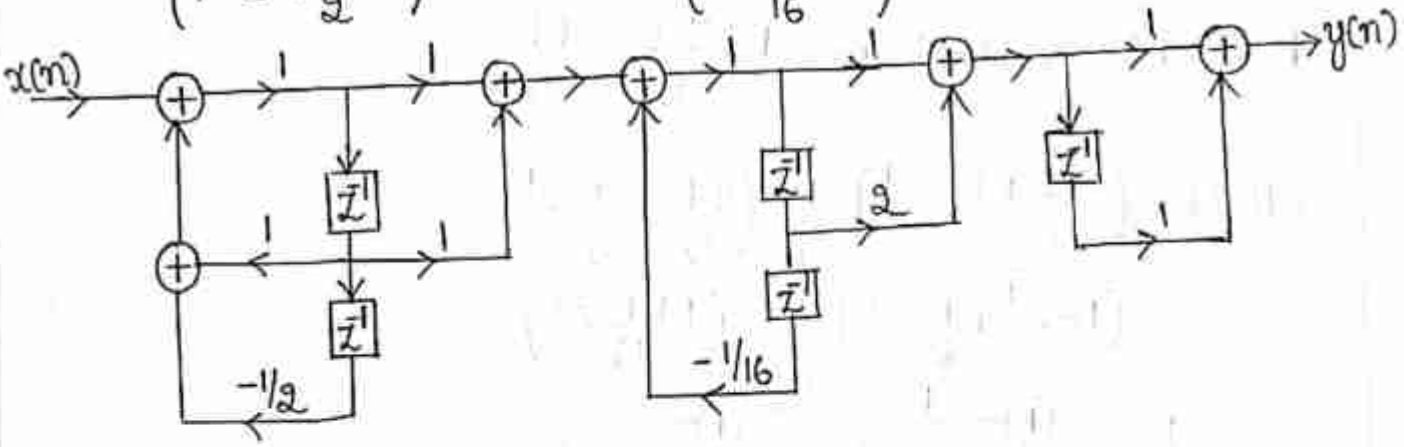
Direct form-II :-



Cascade form :-

$$H(z) = \frac{(z+1)(z+2)(z+1)z}{\left[z^2 - z + \frac{1}{2}\right]\left[z^2 + \frac{1}{16}\right]} = \frac{z^{\cancel{4}}(1+z^{-1})(1+2z^{-1})(1+z^{-1})}{z^{\cancel{4}}\left[1 - z^{-1} + \frac{1}{2}z^{-2}\right]\left[1 + \frac{1}{16}z^{-2}\right]}$$

$$H_1(z) = \frac{(1+z^{-1})}{\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)} \; ; \; H_2(z) = \frac{(1+2z^{-1})}{\left(1 + \frac{1}{16}z^{-2}\right)} \; ; \; H_3(z) = (1+z^{-1})$$



parallel form :-

$$H(z) = \frac{1 + 4z^{-1} + 5z^{-2} + 2z^{-3}}{\underbrace{\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)}_{\text{img}}\underbrace{\left(1 + \frac{1}{16}z^{-2}\right)}_{\text{img}}} = \frac{(Az^{-1} + B)}{\left(1 - z^{-1} + \frac{1}{2}z^{-2}\right)} + \frac{(Cz^{-1} + D)}{\left(1 + \frac{1}{16}z^{-2}\right)}$$

Scanned with CamScanner

$$1+4z^{-1}+5z^{-2}+2z^{-3} = (Az^{-1}+B)\left(1+\frac{1}{16}z^{-2}\right) + (Cz^{-1}+D)\left(1-z^{-1}+\frac{1}{2}z^{-2}\right) \longrightarrow (1)$$

Comparing Co-efficients,

1) Constant : $1 = B+D \longrightarrow (1)$

2) z^{-1} : $4 = A+C-D \longrightarrow (2)$

3) z^{-2} : $5 = \frac{1}{16}B - C + \frac{1}{2}D \longrightarrow (3)$

4) z^{-3} : $2 = \frac{1}{16}A + \frac{1}{2}C \longrightarrow (4)$

Adding Eq(1) and Eq(2),

$5 = A+B+C \longrightarrow (5)$

Divide Eq(2) by 2 and add result with Eq(3),

$$7 = \frac{1}{2}A + \frac{1}{16}B - \frac{1}{2}C \longrightarrow (6)$$

$$2 = \frac{1}{2}A + \frac{1}{2}C - \frac{1}{2}D$$

$$5 = \frac{1}{16}B - C + \frac{1}{2}D$$

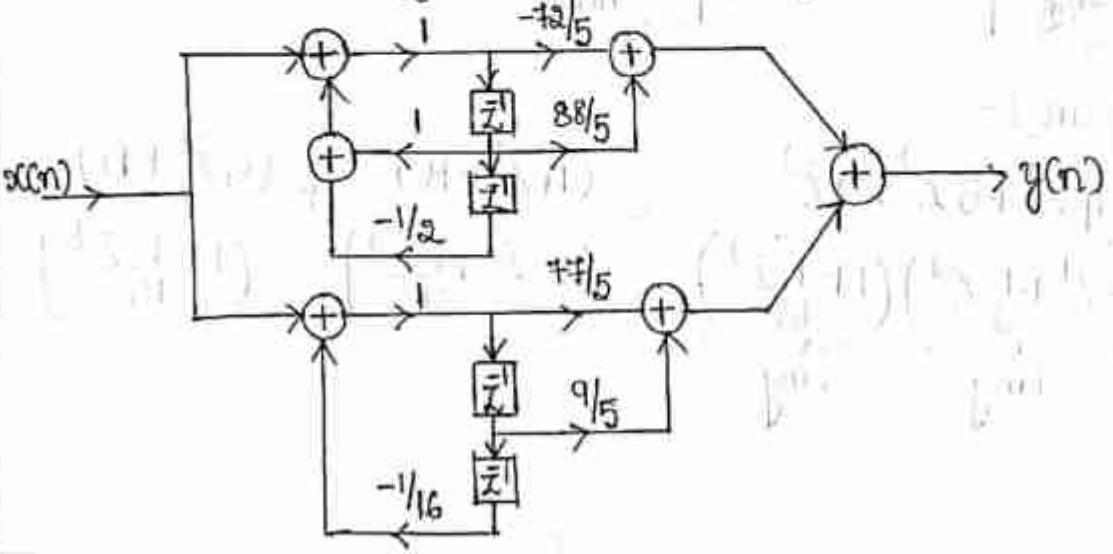
$$7 = \frac{1}{2}A + \frac{1}{16}B - \frac{1}{2}C$$

Solving Eq(4), Eq(5) and Eq(6), we get

$A = \frac{88}{5}$, $B = -\frac{72}{5}$, $C = \frac{9}{5}$

$1 = B+D \Rightarrow D = 1-B = 1+\frac{72}{5} = \frac{77}{5}$

$$\therefore H(z) = \frac{\left(\frac{-72}{5} + \frac{88}{5}z^{-1}\right)}{\left(1-z^{-1}+\frac{1}{2}z^{-2}\right)} + \frac{\left(\frac{77}{5} + \frac{9}{5}z^{-1}\right)}{\left(1+\frac{1}{16}z^{-2}\right)}$$



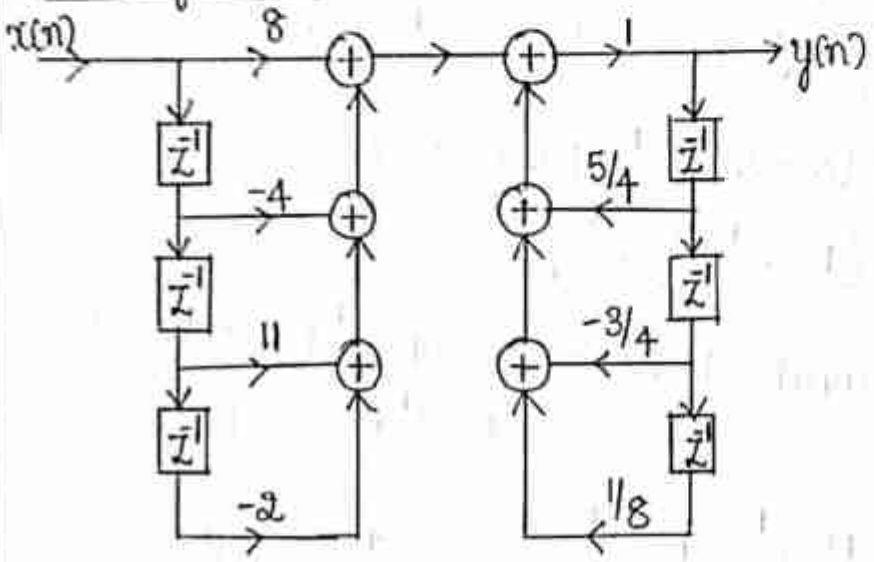
Scanned with CamScanner

14) For the given system function, draw direct form-I and II, Cascade and parallel connection use 2nd and 1st order equation to write cascade and parallel equation:

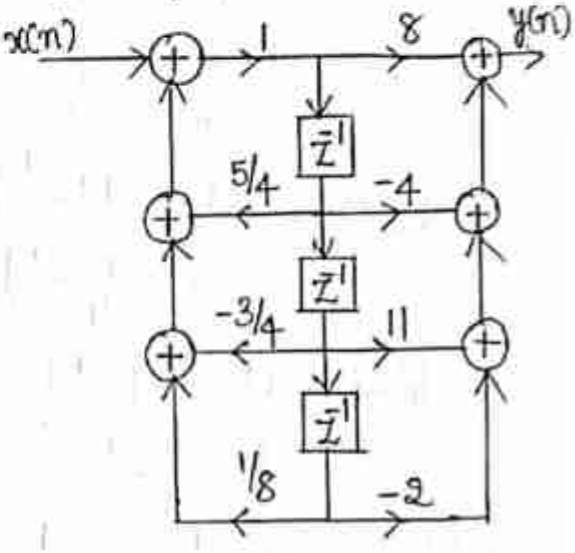
$$H(z) = \frac{8 - 4z^{-1} + 11z^{-2} - 2z^{-3}}{1 - \frac{5}{4}z^{-1} + \frac{3}{4}z^{-2} - \frac{1}{8}z^{-3}}$$

Sol^n:-

Direct form-I :-



Direct form-II :-



Cascade form :-

$$H(z) = \frac{8 - 4z^{-1} + 11z^{-2} - 2z^{-3}}{1 - \frac{5}{4}z^{-1} + \frac{3}{4}z^{-2} - \frac{1}{8}z^{-3}}$$

Consider, $8 - 4z^{-1} + 11z^{-2} - 2z^{-3}$

One of the root is : $0.19 \longrightarrow (1 - 0.19z^{-1})$

$z^{-1} = 0.19$

K	z^{-1}	z^{-2}	z^{-3}
8	-4	11	-2
0	1.52	-0.4712	2
8	-2.48	10.53	0

$\therefore (8 - 4z^{-1} + 11z^{-2} - 2z^{-3}) = (1 - 0.19z^{-1})(8 - 2.48z^{-1} + 10.53z^{-2})$

Scanned with CamScanner

Consider, $1-\frac{5}{4}z^{-1}+\frac{3}{4}z^{-2}-\frac{1}{8}z^{-3}$

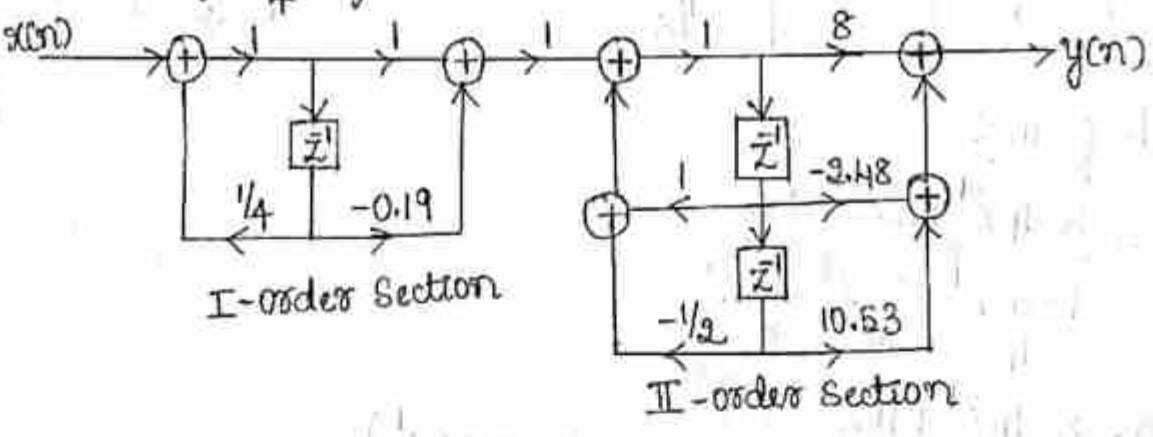
one of the root is : 0.25 $\longrightarrow (1-0.25z^{-1})$

	K	z^{-1}	z^{-2}	z^{-3}
$z^{-1}=0.25$	1	$-\frac{5}{4}$	$\frac{3}{4}$	$-\frac{1}{8}$
		$\frac{1}{4}$	$-\frac{1}{4}$	$\frac{1}{8}$
	1	-1	$\frac{1}{2}$	0

$\therefore \left(1-\frac{5}{4}z^{-1}+\frac{3}{4}z^{-2}-\frac{1}{8}z^{-3}\right) = (1-0.25z^{-1})\left(1-z^{-1}+\frac{1}{2}z^{-2}\right)$

$$H(z) = \frac{(1-0.19z^{-1})(8-2.48z^{-1}+10.53z^{-2})}{(1-\frac{1}{4}z^{-1})(1-z^{-1}+\frac{1}{2}z^{-2})}$$

$$H_1(z) = \frac{(1-0.19z^{-1})}{(1-\frac{1}{4}z^{-1})} \quad \text{and} \quad H_2(z) = \frac{(8-2.48z^{-1}+10.53z^{-2})}{(1-z^{-1}+\frac{1}{2}z^{-2})}$$



parallel form :-

$$H(z) = \frac{8-4z^{-1}+11z^{-2}-2z^{-3}}{1-\frac{5}{4}z^{-1}+\frac{3}{4}z^{-2}-\frac{1}{8}z^{-3}}$$; Here M=N, use long division method

$$H(z) = 16+\frac{(-8+16z^{-1}-z^{-2})}{(1-\frac{1}{4}z^{-1})(1-z^{-1}+\frac{1}{2}z^{-2})}$$

$-\frac{1}{8}z^{-3}+\frac{3}{4}z^{-2}-\frac{5}{4}z^{-1}+1 \;\big|\; -2z^{-3}+11z^{-2}-4z^{-1}+8$ (16)

$-2z^{-3}+12z^{-2}-20z^{-1}+16$

(+) (−) (+) (−)

$-z^{-2}+16z^{-1}-8$

Scanned with CamScanner

Consider, $\frac{-8+16\bar{z}^1-\bar{z}^2}{\left(1-\frac{1}{4}\bar{z}^1\right)\left(1-\bar{z}^1+\frac{1}{2}\bar{z}^2\right)} = \frac{A}{\left(1-\frac{1}{4}\bar{z}^1\right)} + \frac{(B\bar{z}^1+C)}{\left(1-\bar{z}^1+\frac{1}{2}\bar{z}^2\right)}$

$-8+16\bar{z}^1-\bar{z}^2 = A\left[1-\bar{z}^1+\frac{1}{2}\bar{z}^2\right] + (B\bar{z}^1+C)\left(1-\frac{1}{4}\bar{z}^1\right) \longrightarrow (1)$

put $\bar{z}^1=4$ in Eq(1),

$-8+16(4)-(4)^2 = A\left[1-4+\frac{1}{2}(4)^2\right]$

$40 = A(5)$

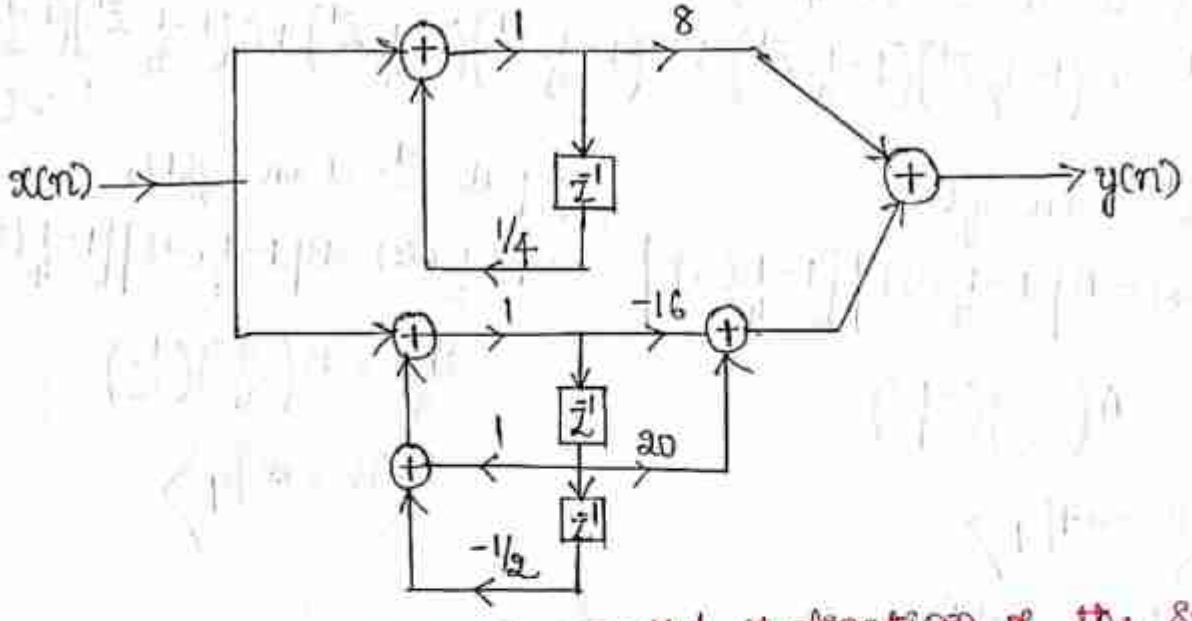
$\langle A=8 \rangle$

Comparing Co-Efficients,

1) Constant : $-8 = A+C \Rightarrow C=-8-8 \Rightarrow \boxed{C=-16}$

2) $\bar{z}^1$: $16 = -A+B-\frac{1}{4}C \Rightarrow B=16+8+\frac{1}{4}(-16) \Rightarrow \boxed{B=20}$

$\therefore H(z) = \frac{8}{\left(1-\frac{1}{4}\bar{z}^1\right)} + \frac{(-16+20\bar{z}^1)}{\left(1-\bar{z}^1+\frac{1}{2}\bar{z}^2\right)}$



15) obtain the Cascade and parallel realization of the System function : $H(z) = \frac{\left(1+\frac{1}{3}\bar{z}^1\right)}{\left(1-\frac{1}{5}\bar{z}^1\right)\left(1-\frac{3}{4}\bar{z}^1+\frac{1}{8}\bar{z}^2\right)}$

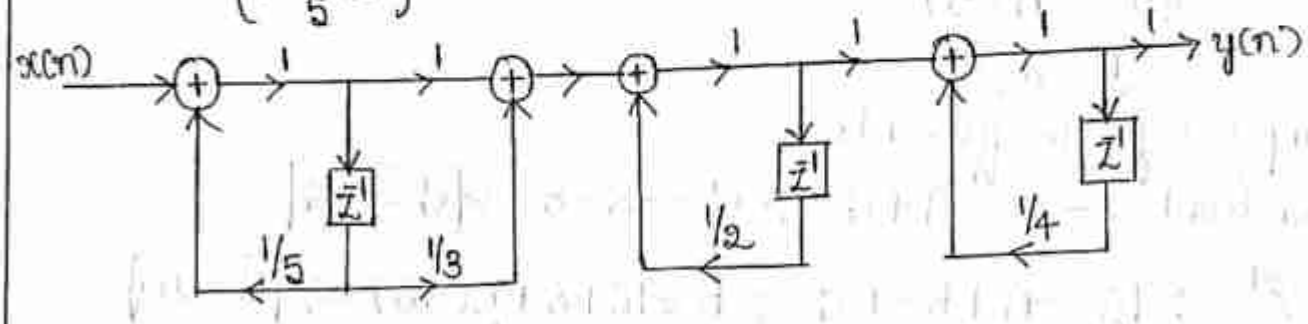
Scanned with CamScanner

Soln :-

Cascade form :-

$$H(z) = \frac{\left(1+\frac{1}{3}z^{-1}\right)}{\left(1-\frac{1}{5}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)}$$

$$H_1(z) = \frac{\left(1+\frac{1}{3}z^{-1}\right)}{\left(1-\frac{1}{5}z^{-1}\right)} \; ; \; H_2(z) = \frac{1}{\left(1-\frac{1}{2}z^{-1}\right)} \; ; \; H_3(z) = \frac{1}{\left(1-\frac{1}{4}z^{-1}\right)}$$



Parallel form :-

$$H(z) = \frac{\left(1+\frac{1}{3}z^{-1}\right)}{\left(1-\frac{1}{5}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)} = \frac{A}{\left(1-\frac{1}{5}z^{-1}\right)} + \frac{B}{\left(1-\frac{1}{2}z^{-1}\right)} + \frac{C}{\left(1-\frac{1}{4}z^{-1}\right)}$$

$$1+\frac{1}{3}z^{-1} = A\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right) + B\left(1-\frac{1}{5}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right) + C\left(1-\frac{1}{5}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}\right) \quad \hookrightarrow (1)$$

put $z^{-1}=5$ in Eq(1),

$$1+\frac{1}{3}(5) = A\left[1-\frac{1}{2}(5)\right]\left[1-\frac{1}{4}(5)\right]$$

$$\frac{8}{3} = A\left(\frac{-3}{2}\right)\left(\frac{-1}{4}\right)$$

$$\langle A = 64/9 \rangle$$

put $z^{-1}=4$ in Eq(1),

$$1+\frac{1}{3}(4) = C\left[1-\frac{1}{5}(4)\right]\left[1-\frac{1}{2}(4)\right]$$

$$\frac{7}{3} = C\left(\frac{1}{5}\right)(-1)$$

$$\langle C = -35/3 \rangle$$

put $z^{-1}=2$ in Eq(1),

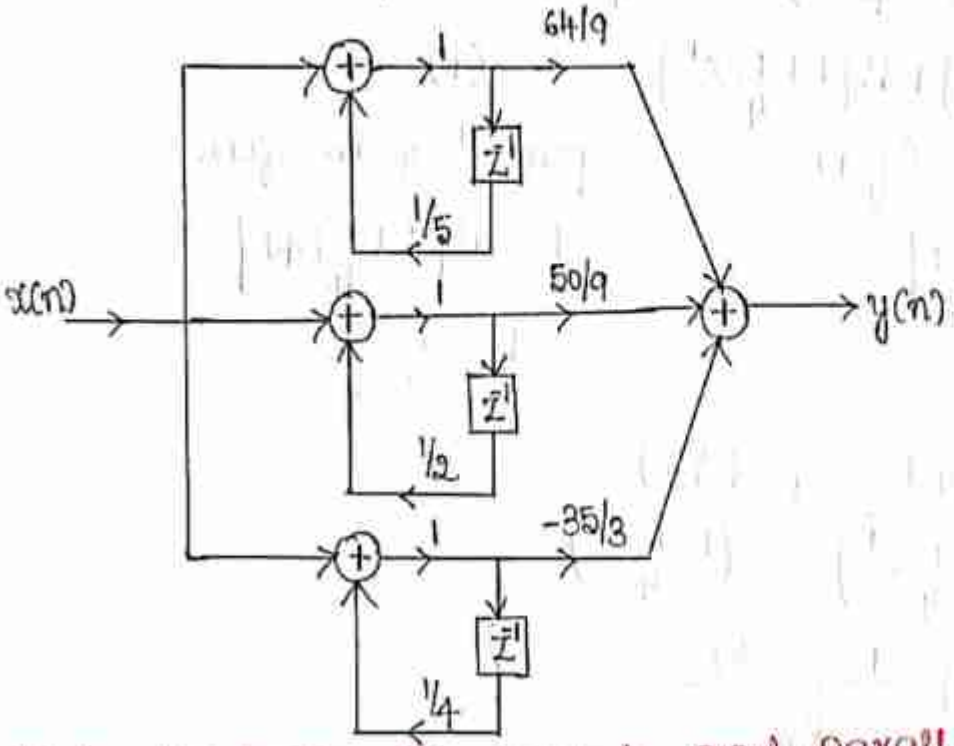
$$1+\frac{1}{3}(2) = B\left[1-\frac{1}{5}(2)\right]\left[1-\frac{1}{4}(2)\right]$$

$$\frac{5}{3} = B\left(\frac{3}{5}\right)\left(\frac{1}{2}\right)$$

$$\langle B = 50/9 \rangle$$

Scanned with CamScanner

$$\therefore H(z) = \frac{(64/9)}{\left(1-\frac{1}{5}z^{-1}\right)} + \frac{(50/9)}{\left(1-\frac{1}{2}z^{-1}\right)} + \frac{(-35/3)}{\left(1-\frac{1}{4}z^{-1}\right)}$$



16) Obtain direct form-II, cascade and parallel realization for the system described by $y(n) = \frac{1}{16}y(n-2) + x(n)$.

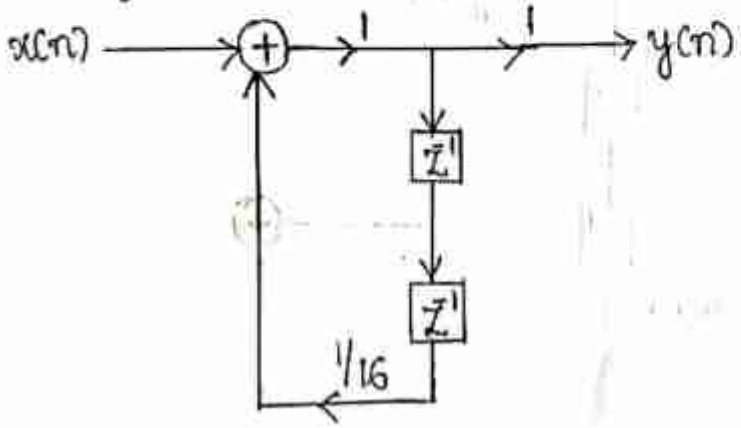
Sol^n :- Given $y(n) = \frac{1}{16}y(n-2) + x(n)$

Apply Z-transform on both the sides,

$$Y(z) - \frac{1}{16}z^{-2}Y(z) = X(z)$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1-\frac{1}{16}z^{-2}}$$

Direct form-II :-



Scanned with CamScanner

Parallel form :-

$$H(z) = \frac{1}{\left(1+\frac{1}{4}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)} = \frac{A}{\left(1+\frac{1}{4}z^{-1}\right)} + \frac{B}{\left(1-\frac{1}{4}z^{-1}\right)}$$

$$1 = A\left(1-\frac{1}{4}z^{-1}\right) + B\left(1+\frac{1}{4}z^{-1}\right) \longrightarrow (1)$$

put $z^{-1} = -4$ in Eq(1)

$$1 = A\left[1-\frac{1}{4}(-4)\right]$$

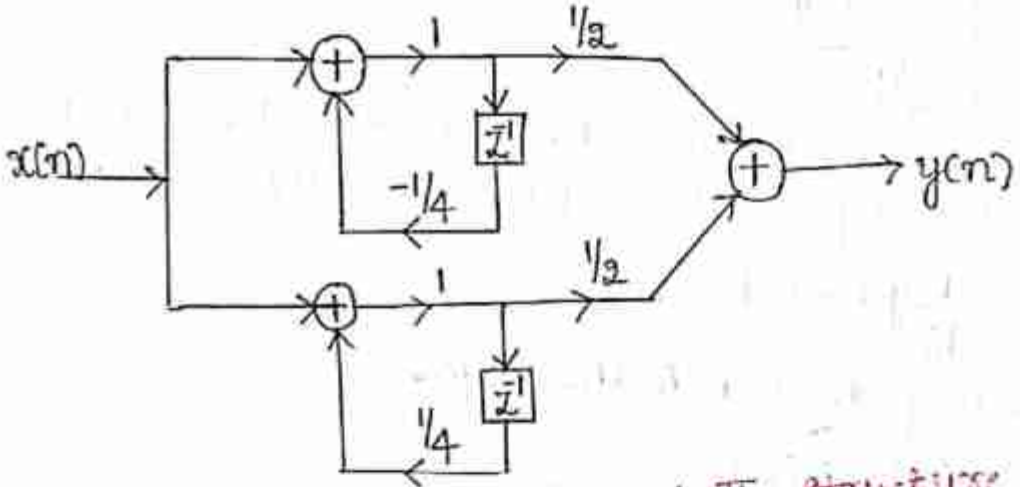
$$\langle A = \frac{1}{2} \rangle$$

put $z^{-1} = 4$ in Eq(1),

$$1 = B\left[1+\frac{1}{4}(4)\right]$$

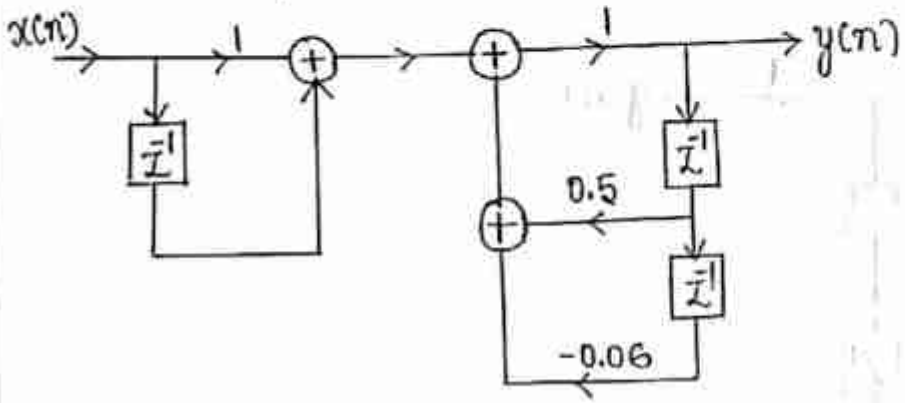
$$\langle B = \frac{1}{2} \rangle$$

$$\therefore H(z) = \frac{(1/2)}{\left(1+\frac{1}{4}z^{-1}\right)} + \frac{(1/2)}{\left(1-\frac{1}{4}z^{-1}\right)}$$



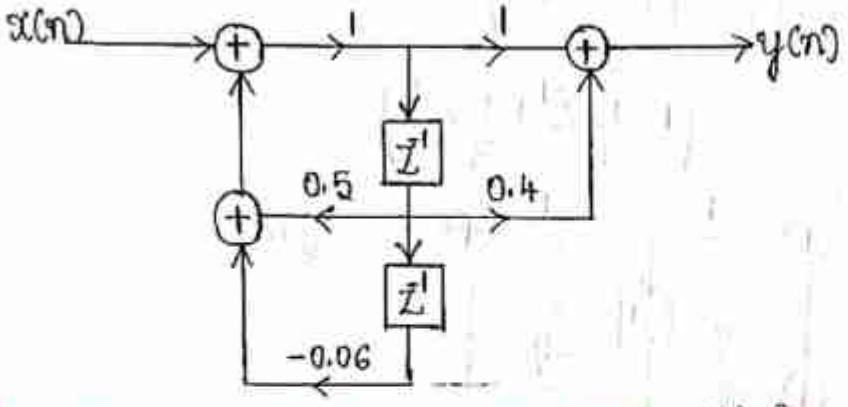
17) Obtain the direct form-I and II structure for the filters given by the system function, $H(z) = \frac{1+0.4z^{-1}}{1-0.5z^{-1}+0.06z^{-2}}$

Sol^n:- Direct form-I :-



Scanned with CamScanner

Direct form-II :-



18) Consider the System function, $H(z) = \frac{\left(1-\frac{1}{5}z^{-1}\right)}{\left(1-\frac{1}{2}z^{-1}+\frac{1}{3}z^{-2}\right)\left(1+\frac{1}{4}z^{-1}\right)}$

i) Realise the System in direct form-I.

ii) Realise the System in cascade form using first and second-order direct form-II sections.

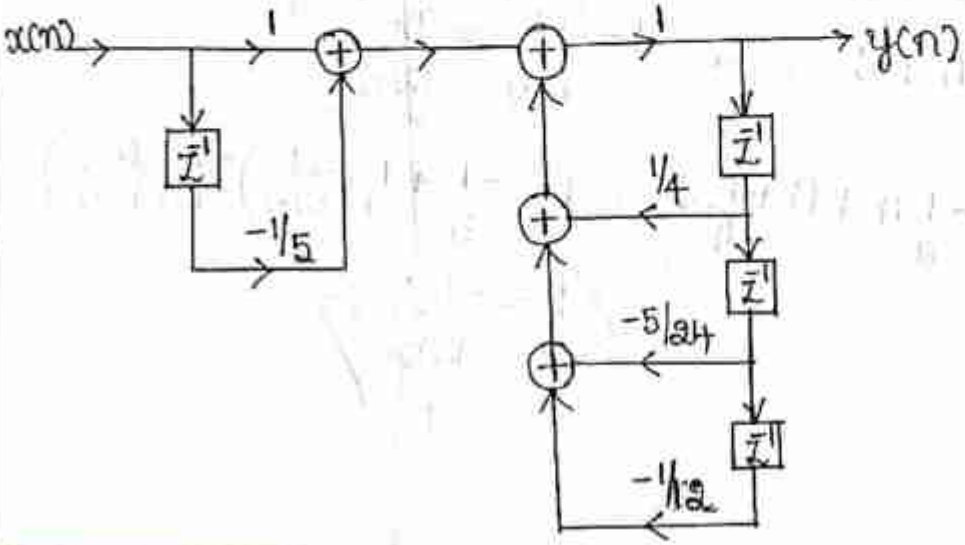
iii) Realise the System in parallel form using first and second-order direct form-II sections.

Soln :-

i) Direct form-I :-

$$H(z) = \frac{\left(1-\frac{1}{5}z^{-1}\right)}{1-\frac{1}{2}z^{-1}+\frac{1}{3}z^{-2}+\frac{1}{4}z^{-1}-\frac{1}{8}z^{-2}+\frac{1}{12}z^{-3}}$$

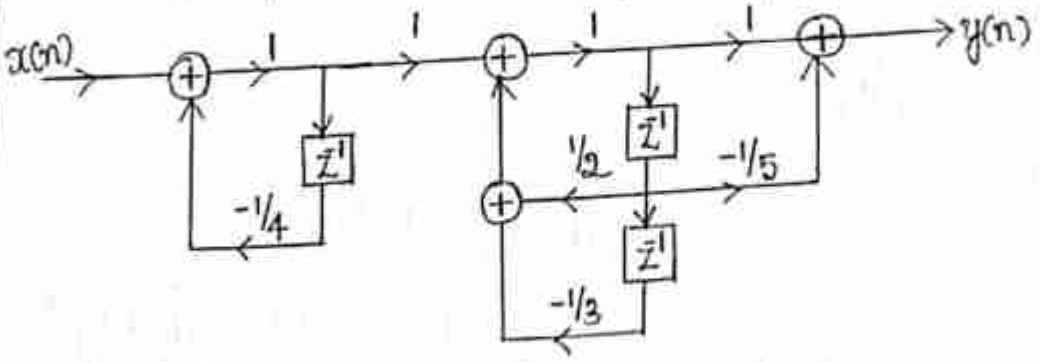
$$= \frac{\left(1-\frac{1}{5}z^{-1}\right)}{\left(1-\frac{1}{4}z^{-1}+\frac{5}{24}z^{-2}+\frac{1}{12}z^{-3}\right)}$$



Scanned with CamScanner

ii) Cascade form :-

$$H_1(z) = \frac{1}{\left(1+\frac{1}{4}z^{-1}\right)} \quad ; \quad H_2(z) = \frac{\left(1-\frac{1}{5}z^{-1}\right)}{\left(1-\frac{1}{2}z^{-1}+\frac{1}{3}z^{-2}\right)}$$



iii) parallel form :-

$$H(z) = \frac{\left(1-\frac{1}{5}z^{-1}\right)}{\left(1+\frac{1}{4}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}+\frac{1}{3}z^{-2}\right)} = \frac{A}{\left(1+\frac{1}{4}z^{-1}\right)} + \frac{(Bz^{-1}+C)}{\left(1-\frac{1}{2}z^{-1}+\frac{1}{3}z^{-2}\right)}$$

$$1-\frac{1}{5}z^{-1} = A\left(1-\frac{1}{2}z^{-1}+\frac{1}{3}z^{-2}\right)+(Bz^{-1}+C)\left(1+\frac{1}{4}z^{-1}\right) \longrightarrow (1)$$

put $z^{-1}=-4$ in Eq(1),

$$1-\frac{1}{5}(-4) = A\left[1-\frac{1}{2}(-4)+\frac{1}{3}(-4)^2\right]$$

$$\frac{9}{5} = A\left(\frac{25}{3}\right)$$

$$\boxed{A = \frac{27}{125}}$$

Comparing the Co-efficients :

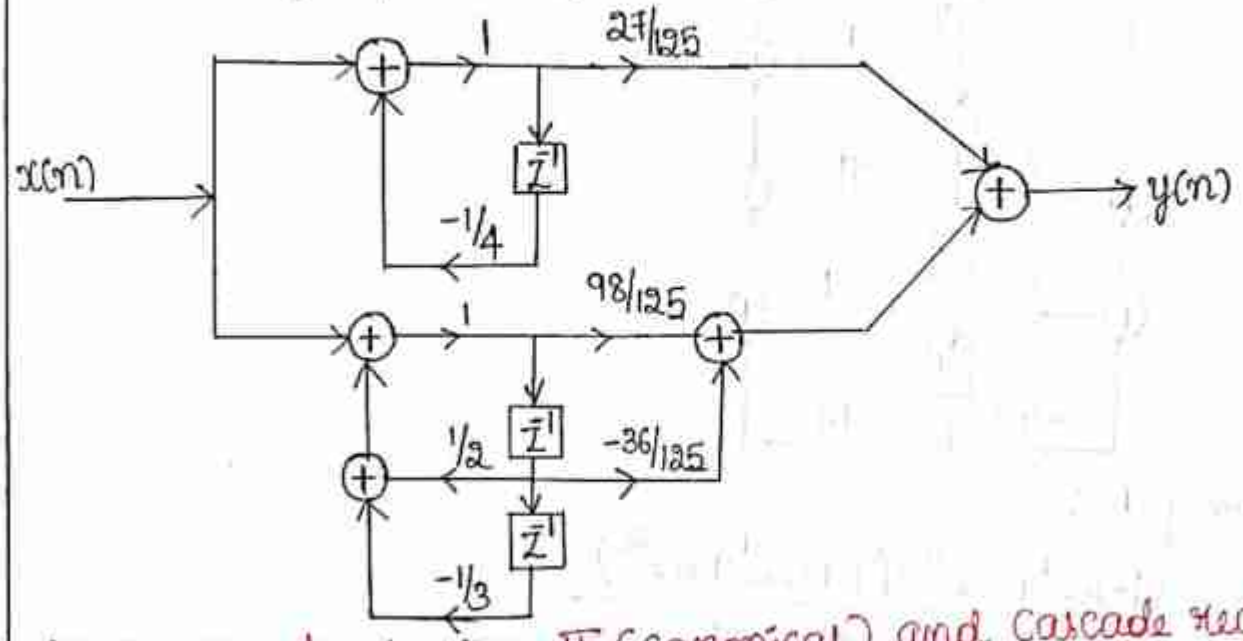
1) Constant : $1 = A + C \Rightarrow C = 1 - \frac{27}{125} = \frac{98}{125}$

2) z^{-1} : $-\frac{1}{5} = -\frac{1}{2}A + B + \frac{1}{4}C \Rightarrow B = -\frac{1}{5} + \frac{1}{2}\left(\frac{27}{125}\right) - \frac{1}{4}\left(\frac{98}{125}\right)$

$$\boxed{B = \frac{-36}{125}}$$

Scanned with CamScanner

$$\therefore H(z) = \frac{\left(27/125\right)}{\left(1+\frac{1}{4}z^{-1}\right)} + \frac{\left(\frac{98}{125} - \frac{36}{125}z^{-1}\right)}{\left(1-\frac{1}{2}z^{-1}+\frac{1}{3}z^{-2}\right)}$$



19) Obtain the direct form-II (canonical) and cascade realization of $H(z) = \frac{(z-1)(z^2+5z+6)(z-3)}{(z^2+6z+5)(z^2-6z+8)}$. The cascade system should consists of two biquadratic sections.

Sol^n:-

$$H(z) = \frac{(z^2-3z-z+3)(z^2+5z+6)}{(z^4-6z^3+8z^2+6z^3-36z^2+48z+5z^2-30z+40)}$$

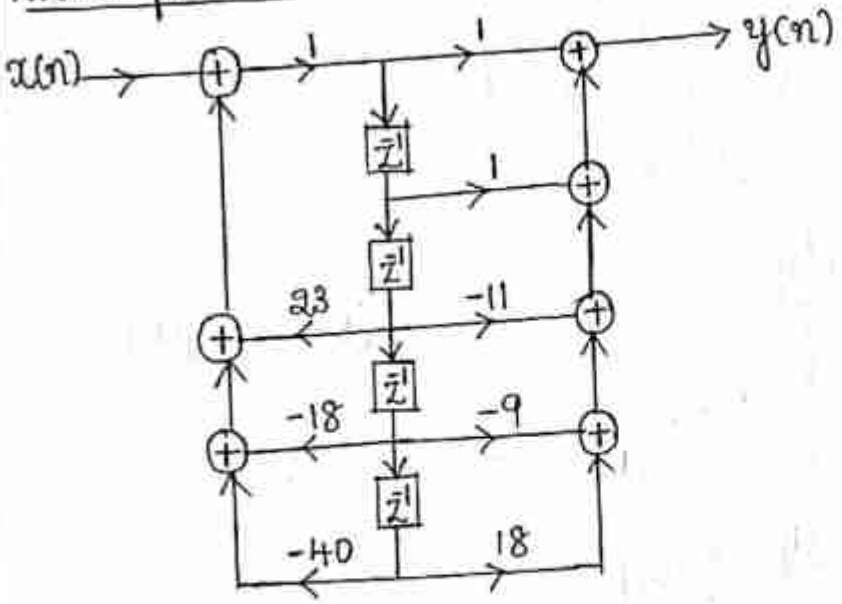
$$= \frac{(z^4+5z^3+6z^2-4z^3-20z^2-24z+3z^2+15z+18)}{(z^4-23z^2+18z+40)}$$

$$H(z) = \frac{z^4+z^3-11z^2-9z+18}{z^4-23z^2+18z+40}$$

$$H(z) = \frac{1+z^{-1}-11z^{-2}-9z^{-3}+18z^{-4}}{1-23z^{-2}+18z^{-3}+40z^{-4}}$$

Scanned with CamScanner

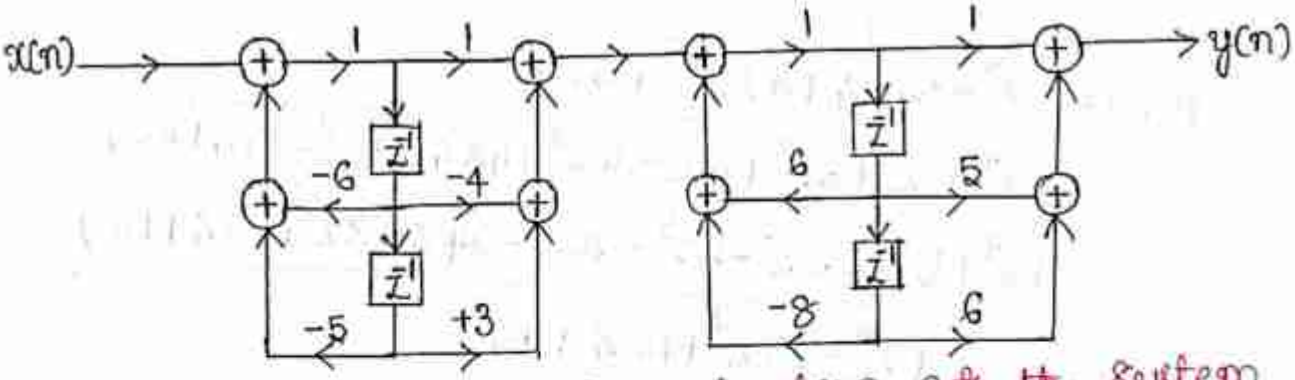
Direct form-II :-



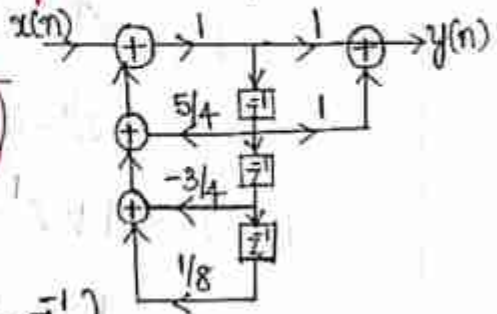
Cascade form :-

$$H(z) = \frac{(1-4z^{-1}+3z^{-2})(1+5z^{-1}+6z^{-2})}{(1+6z^{-1}+5z^{-2})(1-6z^{-1}+8z^{-2})}$$

$$H_1(z) = \frac{(1-4z^{-1}+3z^{-2})}{(1+6z^{-1}+5z^{-2})} \quad \text{and} \quad H_2(z) = \frac{(1+5z^{-1}+6z^{-2})}{(1-6z^{-1}+8z^{-2})}$$



2) Obtain a DF-II and Cascade realization for the System function : $H(z) = \dfrac{(1+z^{-1})}{\left(1-\frac{1}{4}z^{-1}\right)\left(1-z^{-1}+\frac{1}{2}z^{-2}\right)}$



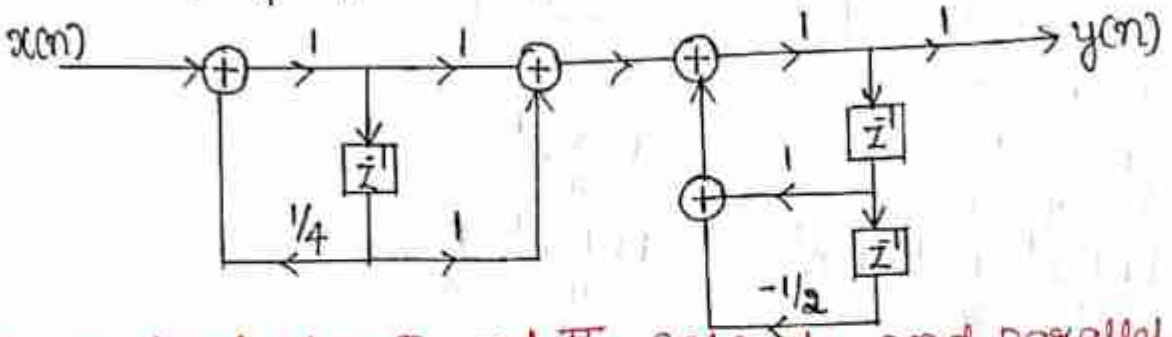
Sol^n :- Direct form-I

$$H(z) = \frac{(1+z^{-1})}{1-z^{-1}+\frac{1}{2}z^{-2}-\frac{1}{4}z^{-1}+\frac{1}{4}z^{-2}-\frac{1}{8}z^{-3}} = \frac{(1+z^{-1})}{\left(1-\frac{5}{4}z^{-1}+\frac{3}{4}z^{-2}-\frac{1}{8}z^{-3}\right)}$$

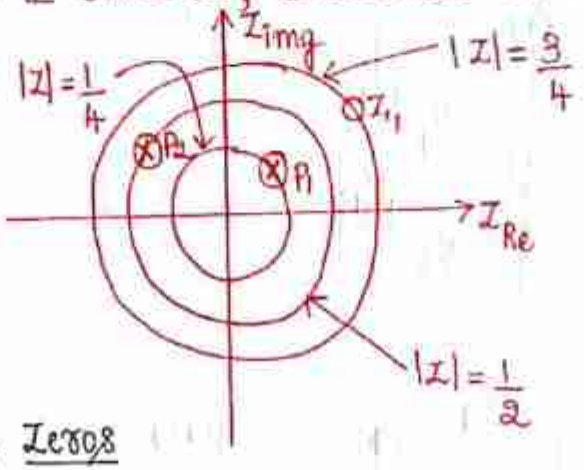
Scanned with CamScanner

Cascade form :-

$$H_1(z) = \frac{(1+z^{-1})}{(1-\frac{1}{4}z^{-1})} \quad \text{and} \quad H_2(z) = \frac{1}{(1-z^{-1}+\frac{1}{2}z^{-2})}$$



2) Draw direct form-I and II, Cascade and parallel structure for,



Sol^n :- $H(z) = \frac{Y(z)}{X(z)} = \frac{\text{Zeros}}{\text{Poles}}$

Zeros :- $z_1 = \frac{3}{4} \longrightarrow \left(1-\frac{3}{4}z^{-1}\right)$

Poles :- $P_1 = \frac{1}{4} \longrightarrow \left(1-\frac{1}{4}z^{-1}\right)$

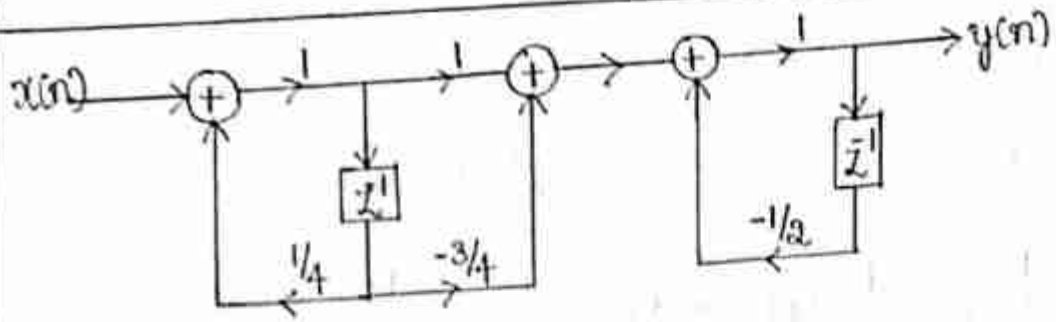
$P_2 = \frac{-1}{2} \longrightarrow \left(1+\frac{1}{2}z^{-1}\right)$

$$\therefore H(z) = \frac{\left(1-\frac{3}{4}z^{-1}\right)}{\left(1-\frac{1}{4}z^{-1}\right)\left(1+\frac{1}{2}z^{-1}\right)}$$

Cascade form :-

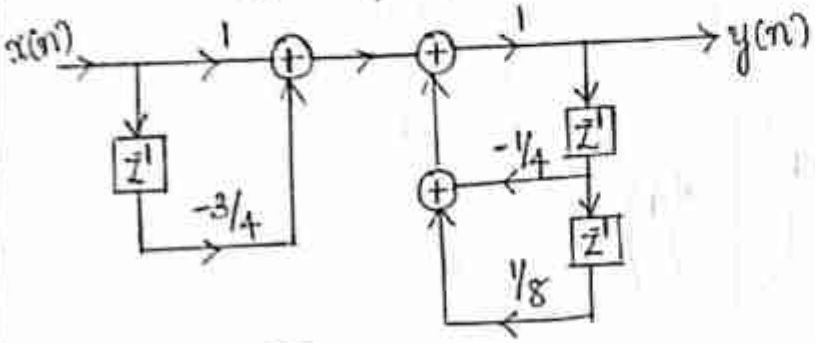
Let $H_1(z) = \frac{\left(1-\frac{3}{4}z^{-1}\right)}{\left(1-\frac{1}{4}z^{-1}\right)}$; $H_2(z) = \frac{1}{\left(1+\frac{1}{2}z^{-1}\right)}$

Scanned with CamScanner

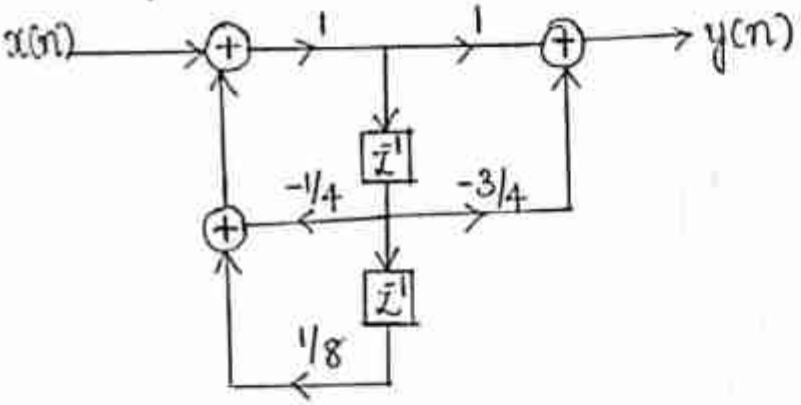


Direct form-I :-

$$H(z) = \frac{\left(1-\frac{3}{4}z^{-1}\right)}{1+\frac{1}{2}z^{-1}-\frac{1}{4}z^{-1}+\frac{1}{8}z^{-2}} = \frac{1-\frac{3}{4}z^{-1}}{1+\frac{1}{4}z^{-1}-\frac{1}{8}z^{-2}}$$



Direct form-II :-



parallel form :-

$$H(z) = \frac{\left(1-\frac{3}{4}z^{-1}\right)}{\left(1-\frac{1}{4}z^{-1}\right)\left(1+\frac{1}{2}z^{-1}\right)} = \frac{A}{\left(1-\frac{1}{4}z^{-1}\right)} + \frac{B}{\left(1+\frac{1}{2}z^{-1}\right)}$$

$$1-\frac{3}{4}z^{-1} = A\left(1+\frac{1}{2}z^{-1}\right) + B\left(1-\frac{1}{4}z^{-1}\right) \longrightarrow (1)$$

put $z^{-1}=4$ in Eq(1),

$$1-\frac{3}{4}(4) = A\left[1+\frac{1}{2}(4)\right]$$

$$-2 = A(3) \Rightarrow \boxed{A = -\frac{2}{3}}$$

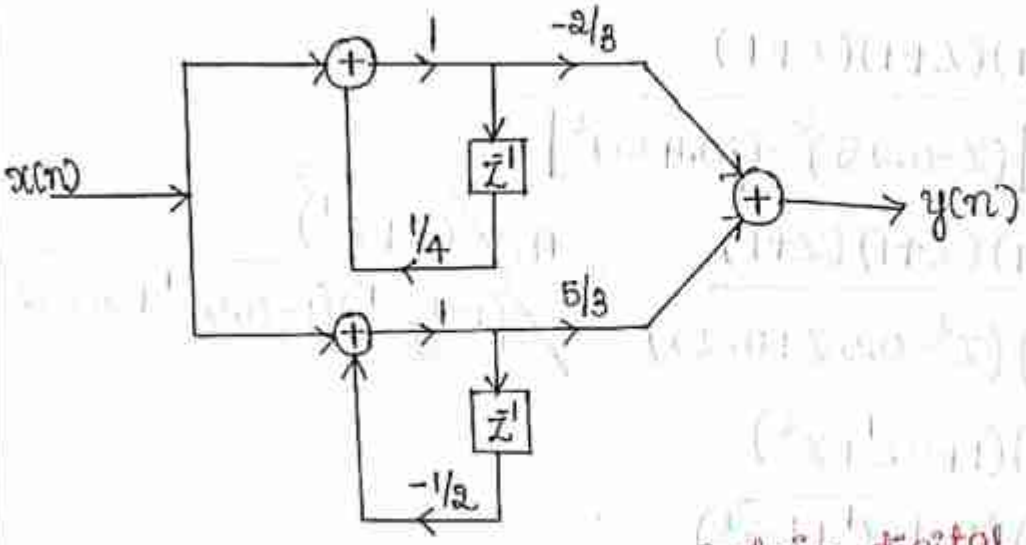
put $z^{-1}=-2$ in Eq(1),

$$1-\frac{3}{4}(-2) = B\left[1-\frac{1}{4}(-2)\right]$$

$$\frac{5}{2} = B\left(\frac{3}{2}\right) \Rightarrow \boxed{B = \frac{5}{3}}$$

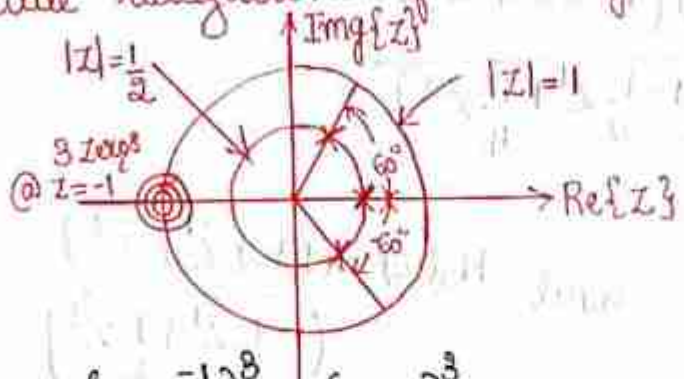
Scanned with CamScanner

$$\therefore H(z) = \frac{(-2/3)}{\left(1-\frac{1}{4}z^{-1}\right)} + \frac{(5/3)}{\left(1+\frac{1}{2}z^{-1}\right)}$$



22) A Z-plane pole-zero plot for a certain digital filter is shown in fig below. The filter has unity gain at DC. Determine the system function in the form:

$$H(z) = A\left[\frac{(1+a_1z^{-1})(1+b_1z^{-1}+b_2z^{-2})}{(1+c_1z^{-1})(1+d_1z^{-1}+d_2z^{-2})}\right]$$ giving the numerical values for parameters $A, a_1, b_1, b_2, c_1, d_1$ and d_2. Sketch the direct form-II and cascade realizations of the system.



Sol^n:- 3 zeros : $(1+z^{-1})^3 = (z+1)^3$

poles : $P_1 = \left(z-\frac{1}{2}\right) = \left(1-\frac{1}{2}z^{-1}\right)$

$$P_2 = \left[z-\frac{1}{2}e^{j60}\right] = \left[z-\frac{1}{2}(\cos 60 + j\sin 60)\right] = \left[(z-0.25)-j0.433\right]$$

$$P_3 = \left[z-\frac{1}{2}e^{-j60}\right] = \left[z-\frac{1}{2}(\cos 60 - j\sin 60)\right] = \left[(z-0.25)+j0.433\right]$$

Scanned with CamScanner

$$H(z) = \frac{A.(z+1)(z+1)(z+1)}{\left(z-\frac{1}{2}\right)[(z-0.25)-j0.433][(z-0.25)+j0.433]}$$

$$= \frac{A.(z+1)(z+1)(z+1)}{\left(z-\frac{1}{2}\right)[(z-0.25)^2-(j0.433)^2]}$$

$$= \frac{A.(z+1)(z+1)(z+1)}{\left(z-\frac{1}{2}\right)(z^2-0.5z+0.25)} = \frac{A.z^3(1+z^{-1})^3}{z^3\left(1-\frac{1}{2}z^{-1}\right)(1-0.5z^{-1}+0.25z^{-2})}$$

$$H(z) = \frac{A(1+z^{-1})(1+2z^{-1}+z^{-2})}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}\right)}$$

$H(z)\big|_{z=1} = 1$,

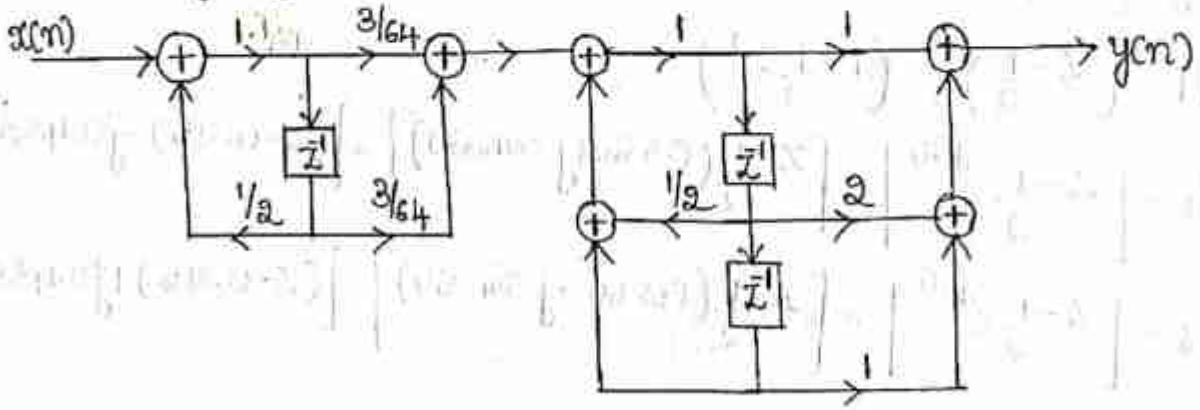
$$1 = \frac{A(1+1)(1+2+1)}{\left(1-\frac{1}{2}\right)\left(1-\frac{1}{2}+\frac{1}{4}\right)} \Rightarrow 1 = \frac{A\times 8}{\left(\frac{1}{2}\right)\left(\frac{3}{4}\right)} \Rightarrow 1 = A\left(\frac{64}{3}\right)$$

$$\langle A = \frac{3}{64} \rangle$$

$$\therefore H(z) = \frac{\left(\frac{3}{64}\right)(1+z^{-1})(1+2z^{-1}+z^{-2})}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}\right)}$$

Cascade form :-

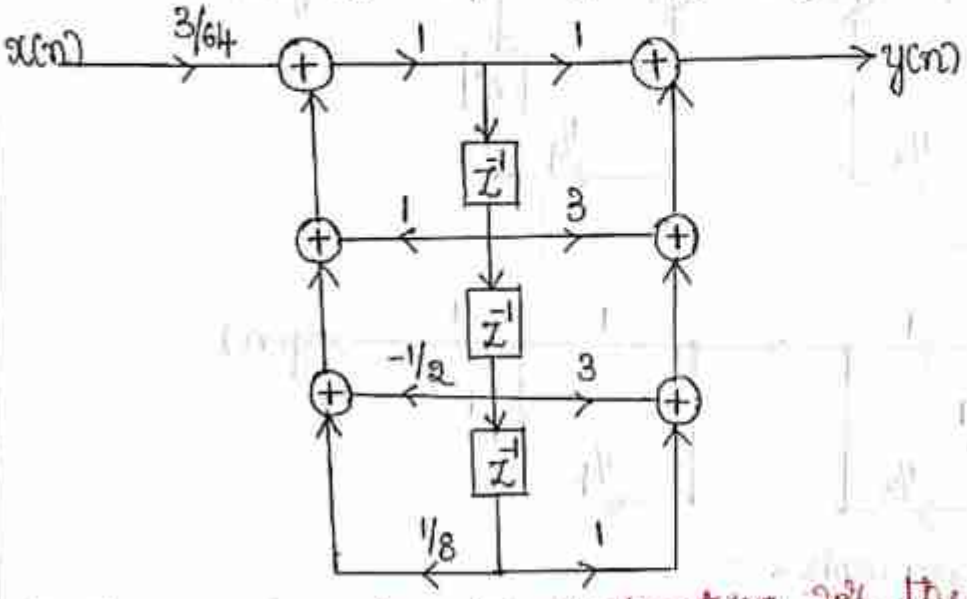
$$H_1(z) = \frac{\left(\frac{3}{64}\right)+\left(\frac{3}{64}\right)z^{-1}}{\left(1-\frac{1}{2}z^{-1}\right)} \quad \text{and} \quad H_2(z) = \frac{(1+2z^{-1}+z^{-2})}{\left(1-\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}\right)}$$



Scanned with CamScanner

Direct-form-II :-

$$H(z) = \left(\frac{3}{64}\right)\frac{(1+2z^{-1}+z^{-2}+z^{-1}+2z^{-2}+z^{-3})}{\left(1-\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}-\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}-\frac{1}{8}z^{-3}\right.} = \frac{\left(\frac{3}{64}\right)(1+3z^{-1}+3z^{-2}+z^{-3})}{\left(1-z^{-1}+\frac{1}{2}z^{-2}-\frac{1}{8}z^{-3}\right)}$$



23) obtain the Cascade-form realization for the given difference Equation : $y(n) = \frac{3}{4}y(n-1) - \frac{1}{8}y(n-2) + x(n) + \frac{1}{3}x(n-1)$. Also, draw the signal flowgraph and its transposed graph.

Sol^n :-

Given $y(n) = \frac{3}{4}y(n-1) - \frac{1}{8}y(n-2) + x(n) + \frac{1}{3}x(n-1)$

Applying Z-transform on both sides,

$$Y(z) - \frac{3}{4}z^{-1}Y(z) + \frac{1}{8}z^{-2}Y(z) = X(z) + \frac{1}{3}z^{-1}X(z)$$

$$Y(z)\left[1 - \frac{3}{4}z^{-1} + \frac{1}{8}z^{-2}\right] = X(z)\left[1 + \frac{1}{3}z^{-1}\right]$$

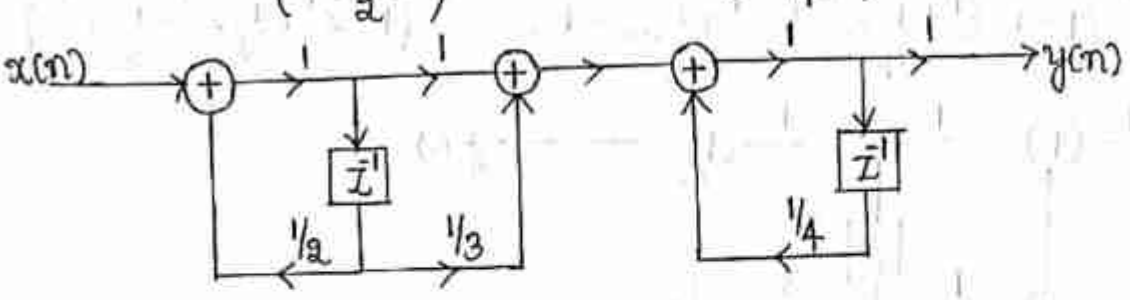
$$H(z) = \frac{Y(z)}{X(z)} = \frac{\left(1+\frac{1}{3}z^{-1}\right)}{\left(1-\frac{3}{4}z^{-1}+\frac{1}{8}z^{-2}\right)}$$

Cascade-form :-

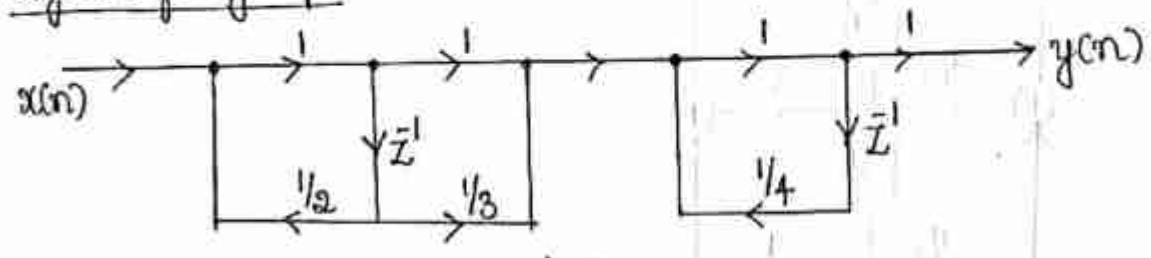
$$H(z) = \frac{\left(1+\frac{1}{3}z^{-1}\right)}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{4}z^{-1}\right)}$$

Scanned with CamScanner

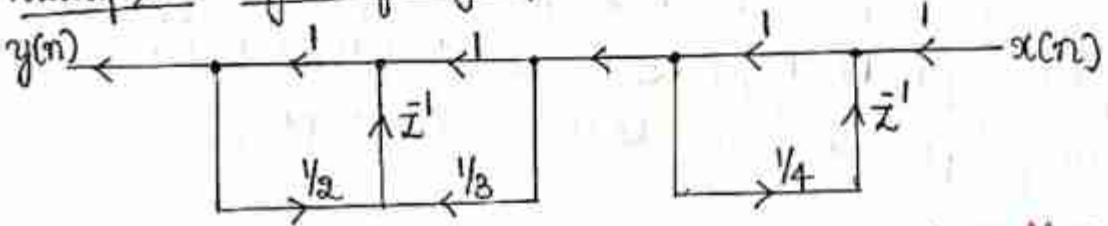
Let $H_1(z) = \dfrac{\left(1+\frac{1}{3}z^{-1}\right)}{\left(1-\frac{1}{2}z^{-1}\right)}$; $H_2(z) = \dfrac{1}{\left(1-\frac{1}{4}z^{-1}\right)}$



Signal flowgraph :—



Transposed Signal flowgraph :—



24) Consider, the causal linear shift invariant filter with system function : $H(z) = \dfrac{(1+0.875z^{-1})}{(1+0.2z^{-1}+0.9z^{-2})(1-0.7z^{-1})}$. Draw a signal flowgraph for this system using,

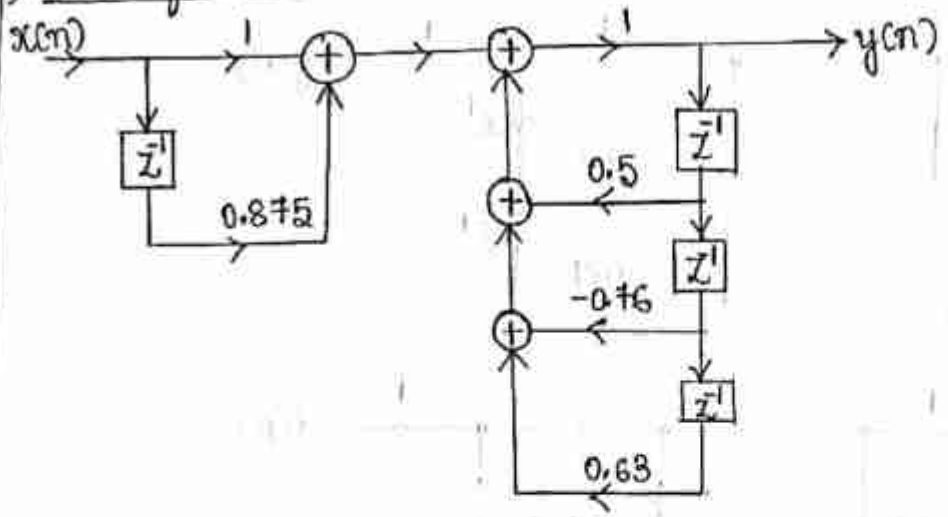
i) Direct form-I ii) Direct-form-II iii) A cascade of first and second-order systems realized in direct form-II iv) A cascade of first and second-order systems realized in transposed direct-form-II.

Sol^n :- $H(z) = \dfrac{(1+0.875z^{-1})}{(1+0.2z^{-1}+0.9z^{-2}-0.7z^{-1}-0.14z^{-2}-0.63z^{-3})}$

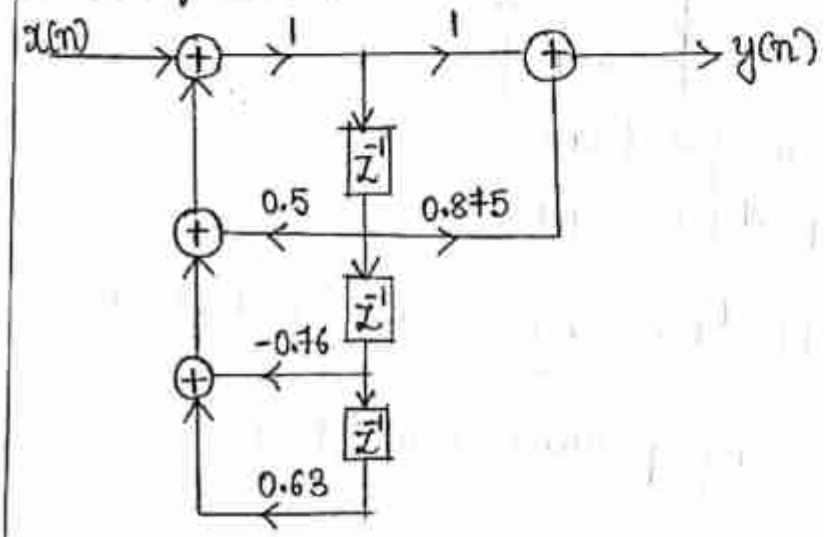
$$H(z) = \frac{(1+0.875z^{-1})}{(1-0.5z^{-1}+0.76z^{-2}-0.63z^{-3})}$$

Scanned with CamScanner

i) Direct form-I :-



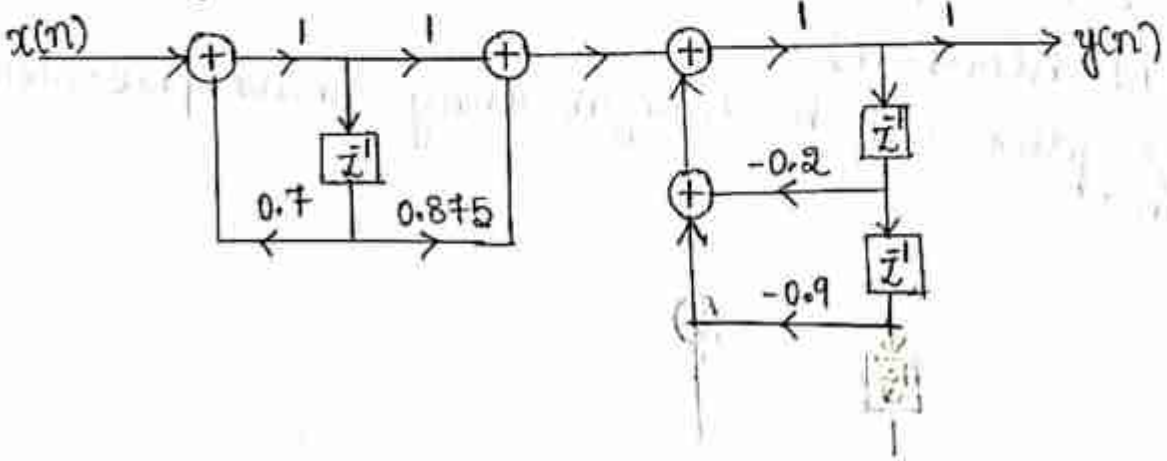
ii) Direct form-II :-



iii) Cascade form :-

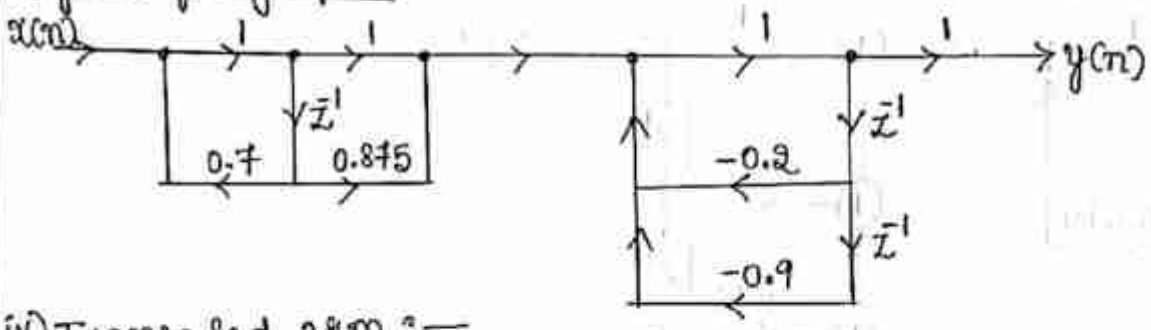
$$H(z) = \frac{(1+0.875z^{-1})}{(1-0.7z^{-1})(1+0.2z^{-1}+0.9z^{-2})}$$

$$H_1(z) = \frac{(1+0.875z^{-1})}{(1-0.7z^{-1})} \quad \text{and} \quad H_2(z) = \frac{1}{(1+0.2z^{-1}+0.9z^{-2})}$$

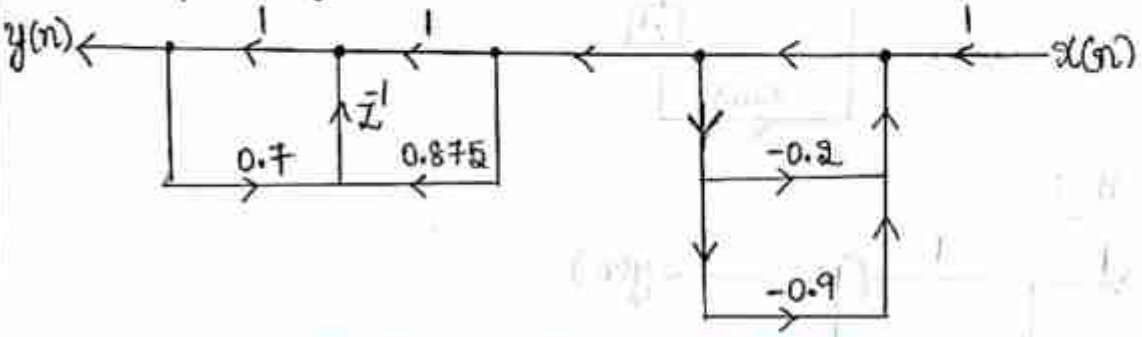


Scanned with CamScanner

iii) Signal flowgraph :-



iv) Transposed form :-



25) Realize the following system functions :

i) $H(z) = \frac{1}{2} + \frac{1}{3}z^{-1} + z^{-2} + \frac{1}{4}z^{-3} + z^{-4} + \frac{1}{3}z^{-5} + \frac{1}{2}z^{-6}$

ii) $H(z) = \left(\frac{1}{2} + z^{-1} + \frac{1}{2}z^{-2}\right).\left(1 + \frac{1}{3}z^{-1} + z^{-2}\right)$ by cascade realization.

iii) $H(z) = 3 + \frac{4z}{\left(z-\frac{1}{2}\right)} + \frac{2}{\left(z-\frac{1}{4}\right)}$ by parallel realization

iv) Realize first-order digital integrator.

Sol^n :-

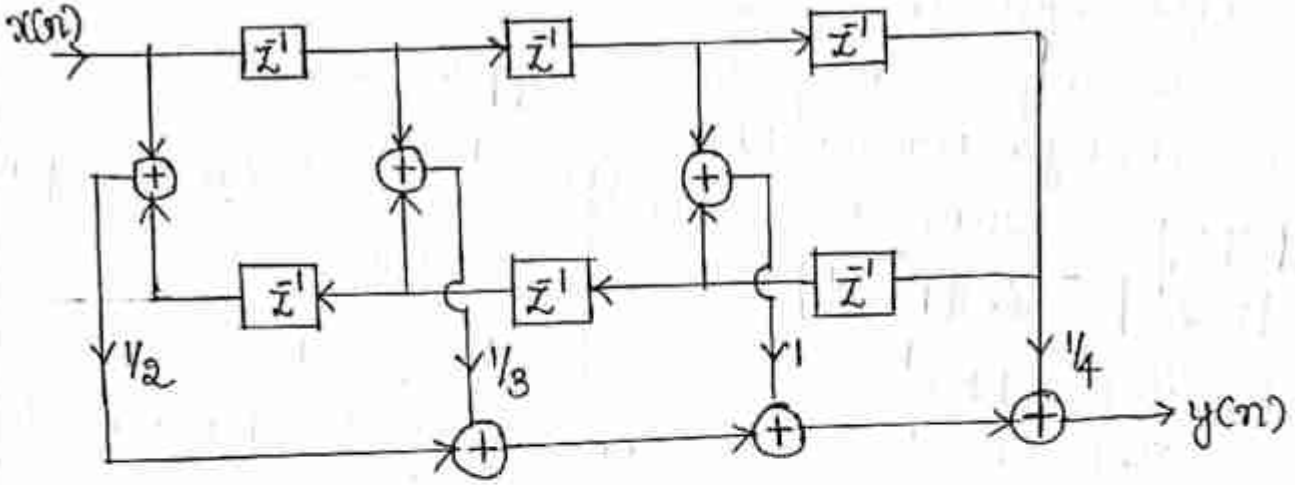
i) $H(z) = \frac{1}{2} + \frac{1}{3}z^{-1} + z^{-2} + \frac{1}{4}z^{-3} + z^{-4} + \frac{1}{3}z^{-5} + \frac{1}{2}z^{-6}$

$h(n) = \left\{\frac{1}{2}, \frac{1}{3}, 1, \frac{1}{4}, 1, \frac{1}{3}, \frac{1}{2}\right\}$; $M = 7$

Here $h(n) = h(M-1-n)$

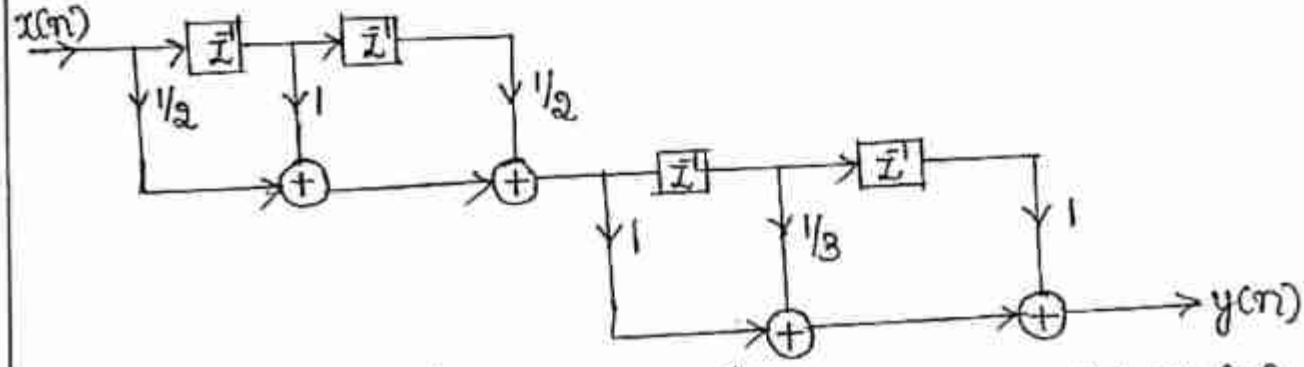
Hence, this filter can be realized using linear-phase structure as shown :

Scanned with CamScanner

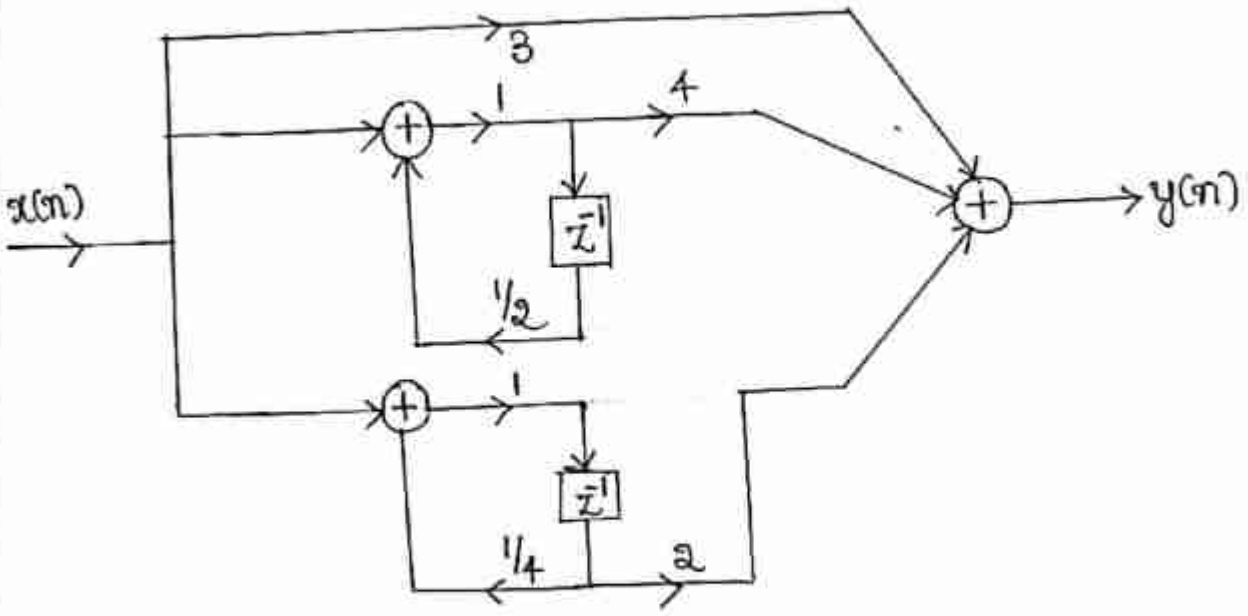


ii) Cascade form :—

$H_1(z) = \left(\frac{1}{2} + z^{-1} + \frac{1}{2}z^{-2}\right)$ and $H_2(z) = \left(1 + \frac{1}{3}z^{-1} + z^{-2}\right)$



iii) $H(z) = 3 + \frac{4z}{z\left(1-\frac{1}{2}z^{-1}\right)} + \frac{2z^{-1}}{\left(1-\frac{1}{4}z^{-1}\right)} = H_1(z) + H_2(z) + H_3(z)$



Scanned with CamScanner

iv) First-order digital integrator :-

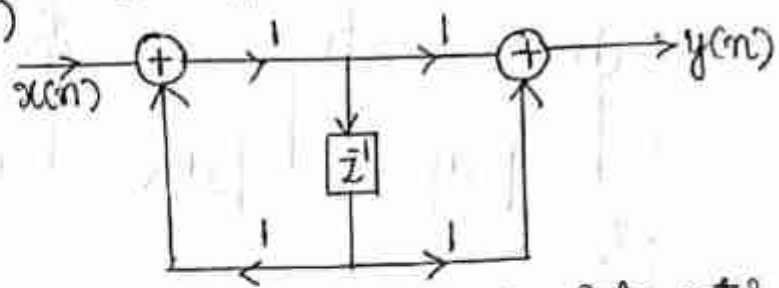
First-order digital integrator is approximated as,

$$y(n) = x(n) + y(n-1) + x(n-1)$$

Applying Z-transform,

$$Y(z)[1 - z^{-1}] = X(z)[1 + z^{-1}]$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1 + z^{-1}}{1 - z^{-1}}$$



DF-II for first-order integrator

Scanned with CamScanner

Module-5:

Anusha. M.N
Asst. professor
Dept. of ECE
BGSIT, B.G.Nagar

Realization of FIR Systems (Structures for FIR filters)

Basic FIR filter Structures:-

A finite impulse response(FIR) system doesnot have feedback. Hence, $y(n-K)$ terms (past output terms) are absent. Hence, the output of FIR system is given by the following difference equation:

$$y(n) = \sum_{K=-\infty}^{\infty} b_K x(n-K)$$

If there are 'M' Co-efficients, then

$$y(n) = \sum_{K=0}^{M-1} b_K x(n-K)$$

Taking Z-transform,

$$Y(z) = \sum_{K=0}^{M-1} b_K z^{-K} . X(z)$$

$$\frac{Y(z)}{X(z)} = H(z) = \sum_{K=0}^{M-1} b_K z^{-K} \longrightarrow (1)$$

This equation is called a system transfer function of FIR filter.

1) Direct-form realization of FIR System:-

The direct form realization of FIR filter can be obtained by using the equation of linear convolution,

$$y(n) = \sum_{K=-\infty}^{\infty} h(K) . x(n-K) \longrightarrow (1)$$

If we consider, 'M' samples, then Eq(1) becomes

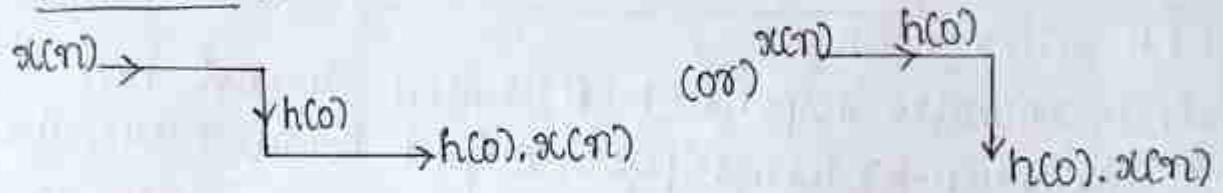
$$y(n) = \sum_{K=0}^{M-1} h(K) . x(n-K) \longrightarrow (2)$$

Scanned with CamScanner

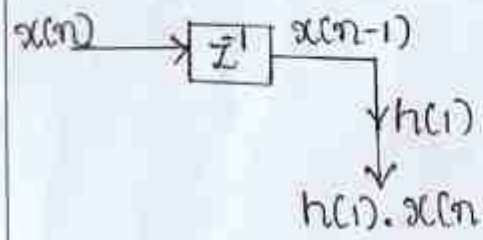
$\therefore y(n) = h(0).x(n) + h(1).x(n-1) + h(2).x(n-2) + \ldots + h(M-1)x(M-1)$ → (3)

Let us consider, the implementation of individual terms of the above Equation :

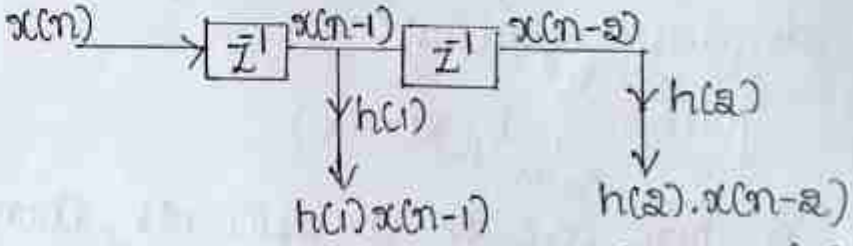
$h(0).x(n)$:-



$h(1).x(n-1)$:-



$h(2).x(n-2)$:-



Similarly, we can draw the block diagram for remaining terms of Eq(3).

The total structure for Eq(3) can be drawn by adding the block diagram for all terms.

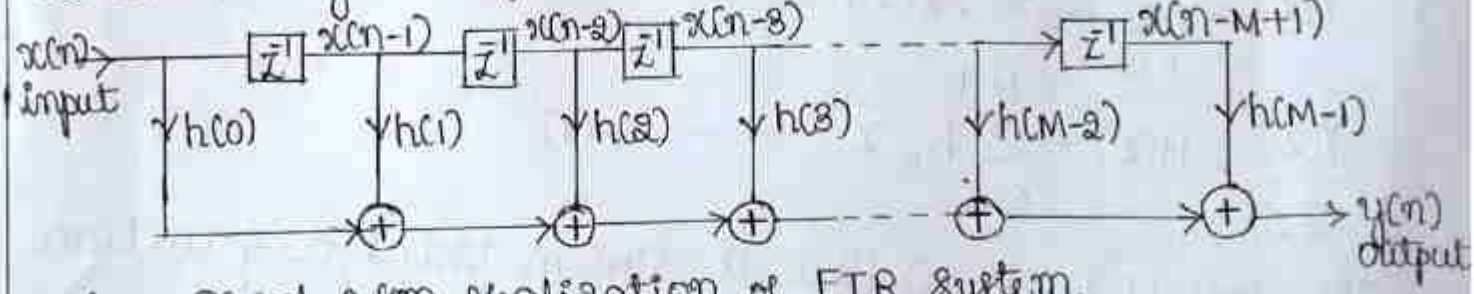


fig : Direct form realization of FIR System

→ There are (M-1) delay blocks, so this is canonical structure.

→ Input signal delays by (M-1) times. So, to store this delayed input signal (M-1) memory locations are required.

→ This structure has (M-1) memory locations

→ Direct form realization is also called as tapped (or) traversal delay line filter.

Scanned with CamScanner

Note:- An FIR filter is characterized by,

$$y(n) = \sum_{K=0}^{M-1} b_K x(n-K) \longrightarrow (1)$$

Also output of an FIR filter with impulse response $h(n)$ of length 'M' for an arbitrary input $x(n)$ is,

$$y(n) = h(n) * x(n) = \sum_{K=0}^{M-1} h(K).x(n-K) \longrightarrow (2)$$

Comparing Eq(1) and Eq(2),

$h(K) = b_K$

$h(n) = b_n$; $0 \le n \le M-1$

0 ; otherwise.

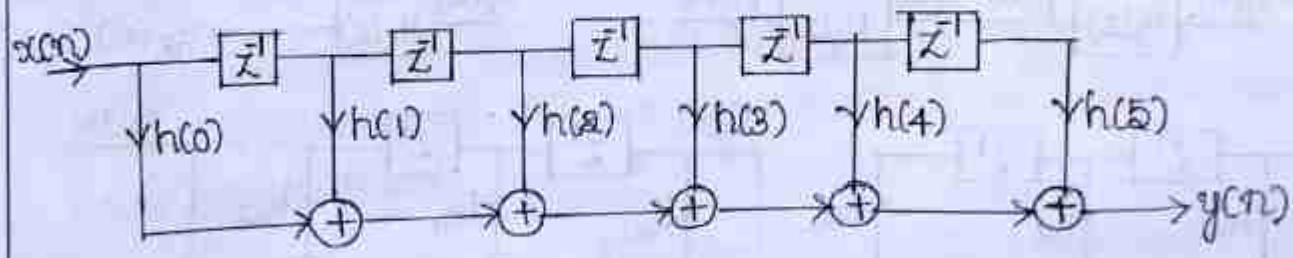
$\therefore y(n) = \sum_{K=0}^{M-1} h(K).x(n-K)$

Note:- FIR filters are often preferred in many applications, since they provide an exact linear phase over the whole frequency range and they are always BIBO stable independent of the filter co-efficients.

Ex:- Consider, the length of the impulse response of an FIR filter $M=6$.

$\therefore y(n) = \sum_{K=0}^{5} h(K).x(n-K)$

$y(n) = h(0)x(n) + h(1)x(n-1) + h(2)x(n-2) + h(3)x(n-3) + h(4)x(n-4) + h(5)x(n-5)$.



Scanned with CamScanner

2) Cascade form realization of FIR systems :-

The system function of FIR system is given by,

$$H(z) = \sum_{K=0}^{M-1} b_K z^{-K} \longrightarrow (1)$$

Generally, the higher order FIR filter is realized by using a series connection of different second-order transfer functions,

$$H(z) = H_1(z) \cdot H_2(z) \cdot H_3(z), \ldots, H_K(z)$$

where,

$H_K(z) = b_{K0} + b_{K_1} z^{-1} + b_{K_2} z^{-2}$; Second-order transfer function

W.K.T $H_K(z) = \dfrac{Y_K(z)}{X(z)} = b_{K0} + b_{K_1} z^{-1} + b_{K_2} z^{-2}$

$\therefore Y_K(z) = b_{K0} X(z) + b_{K_1} X(z) z^{-1} + b_{K_2} X(z) z^{-2}$

Applying inverse Z-transform,

$$y_K(n) = b_{K0} x(n) + b_{K_1} x(n-1) + b_{K_2} x(n-2) \longrightarrow (2)$$

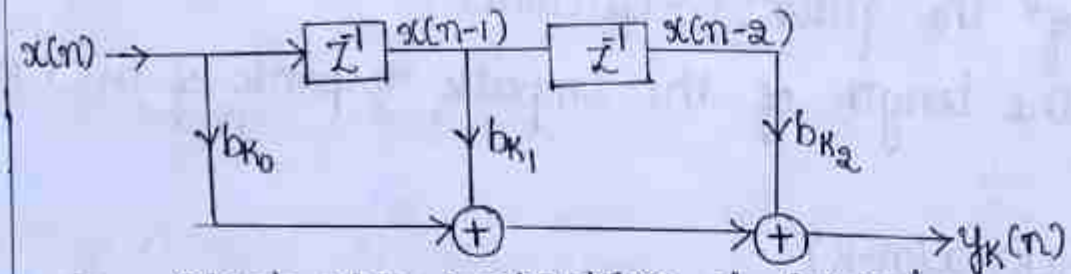


fig : Direct form realization of second-order system

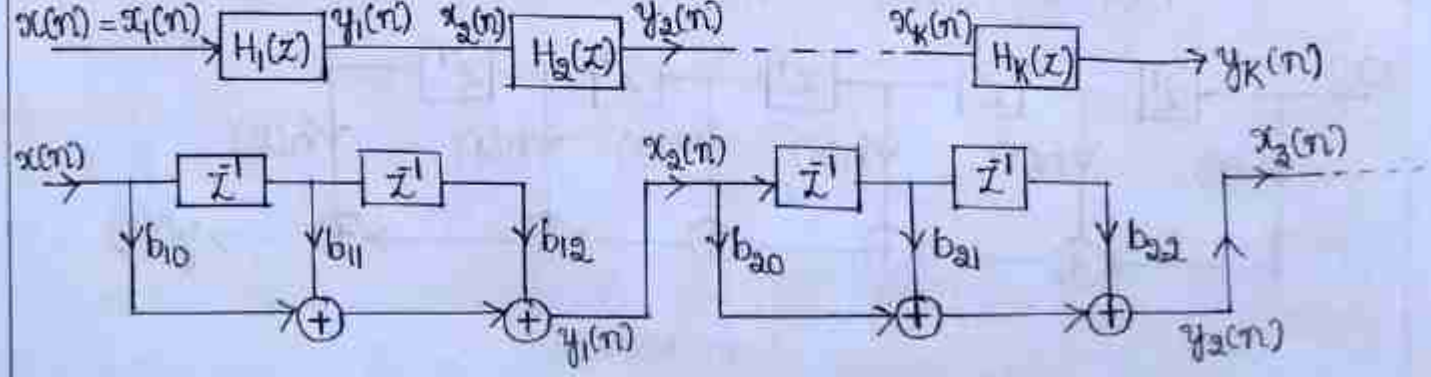


fig : Cascade realization of FIR system

Scanned with CamScanner

Note :- This Cascade realization is obtained by putting K=0 in I-stage and K=1 in II-stage.

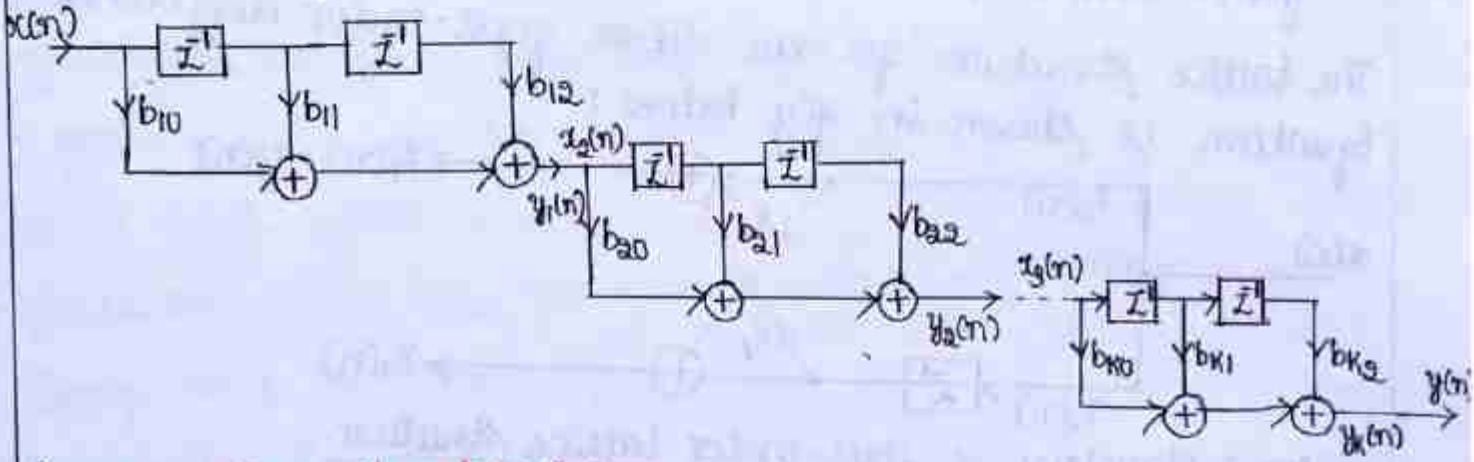
→ This cascade form is canonical with respect to delay.

→ It has (M-1) adders and 'M' multipliers for $(M-1)^{th}$ order FIR transfer functions.

→ It has (M-1) adders and 'M' multipliers for $(M-1)^{th}$ order FIR transfer functions.

→ $y(n) = \sum_{K=0}^{M-1} h(K).x(n-K)$ ⟹ Non-recursive difference equation of FIR filter.

The cascade form structure can also written as :



3) FIR Lattice structures :-

The lattice structure has many attractive features and some of them are :

1) In adaptive signal processing applications, the filter co-efficients are calculated using a minimization algorithm. Lattice structure makes the upgrading of filter orders quite simple and computationally cheap.

2) Lattice filters are computationally very efficient than other filter structures for the implementation of wavelet transforms using filter banks.

3) Lattice filters are less sensitive to finite wordlength effect than direct form filter structure.

Scanned with CamScanner

The transfer function of a FIR filter can be written as:

$$\frac{Y(z)}{X(z)} = A(z) = 1 + \sum_{i=1}^{m} a_m(i) z^{-i} \longrightarrow (1)$$

In Eq(1), 'm' is the order of the FIR filter and $a_m(0) = 1$.

When $m=1$, Eq(1) becomes:

$$\frac{Y(z)}{X(z)} = 1 + a_1(1) z^{-1}$$

$$Y(z) = X(z) + a_1(1) z^{-1} X(z)$$

Taking inverse Z-transform on both the sides,

$$y(n) = x(n) + a_1(1) x(n-1) \longrightarrow (2)$$

The lattice structure for the above first-order difference equation is shown in fig below:

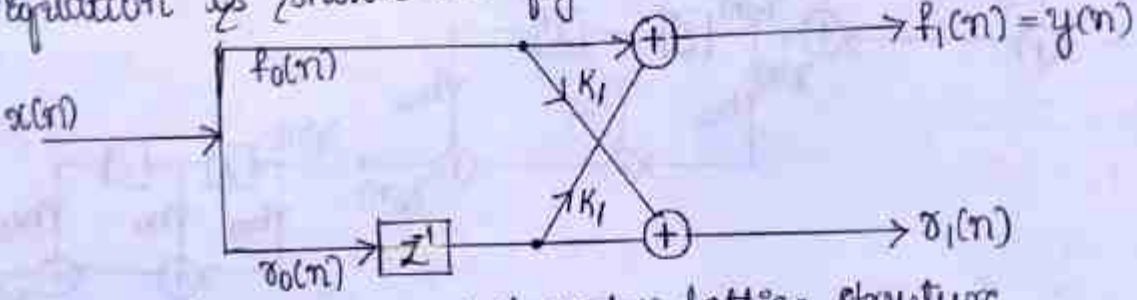


fig: Structure of first-order lattice structure

The first-order lattice structure shown in fig has two outputs $f_1(n)$ is known as the upper channel output and $g_1(n)$ is known as the lower channel output.

The subscripts denote the filter order. This is also called as a single-stage lattice.

The inputs to the first-order lattice are:

$$\boxed{f_0(n) = g_0(n) = x(n)}$$

The outputs are:

$$f_1(n) = f_0(n) + K_1 g_0(n-1) \longrightarrow (a)$$

$$g_1(n) = K_1 f_0(n) + g_0(n-1) \longrightarrow (b)$$

$\Big\} \longrightarrow (3)$

Scanned with CamScanner

From Eq(2) and Eq(3)(a), we find that:

$K_1 = a_1(1)$ then $f_1(n) = y(n)$

Next consider $m=2$ in Eq(1), then

$$\frac{Y(z)}{X(z)} = 1 + a_2(1)z^{-1} + a_2(2)z^{-2}$$

$$Y(z) = X(z) + a_2(1)z^{-1}X(z) + a_2(2)z^{-2}X(z)$$

Taking I-ZT on both the sides,

$$y(n) = x(n) + a_2(1)x(n-1) + a_2(2)x(n-2) \longrightarrow (4)$$

Let $y(n) = f_1(n) + K_2 r_1(n-1) \longrightarrow (5)$

Substituting (3(a)) and (3(b)) in Eq(5),

$$y(n) = f_0(n) + K_1 r_0(n-1) + K_2[K_1 f_0(n-1) + r_0(n-2)]$$

$$\Rightarrow y(n) = f_0(n) + K_1 r_0(n-1) + K_1 K_2 f_0(n-1) + K_2 r_0(n-2)$$

Since $f_0(n) = r_0(n) = x(n)$,

$$y(n) = x(n) + [K_1 + K_1 K_2]x(n-1) + K_2 x(n-2) \longrightarrow (6)$$

Comparing Eq(4) and Eq(6),

$$a_2(1) = K_1 + K_1 K_2$$

and $a_2(2) = K_2$

Solving the above two equations,

$$\boxed{K_1 = \frac{a_2(1)}{1 + a_2(2)} \quad \text{and} \quad K_2 = a_2(2)} \longrightarrow (7)$$

Eq(6) means that the lattice structure for a second-order filter is simply a cascade of two first-order filters as shown in fig below with K_1 and K_2 are defined in Eq(7),

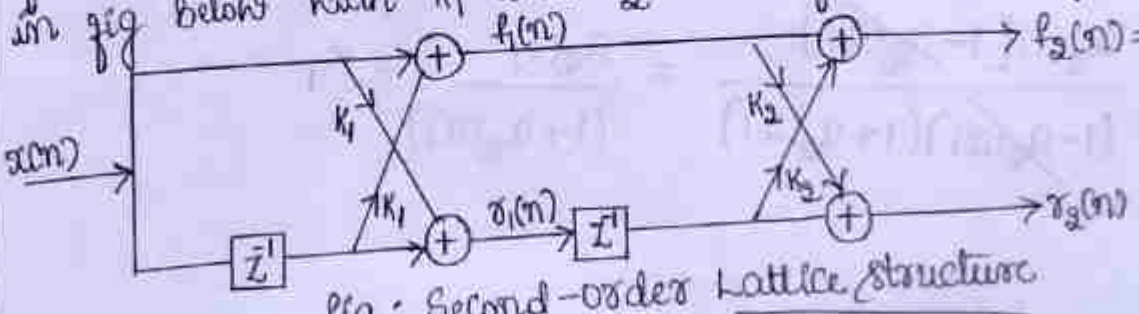


fig: Second-order Lattice Structure

Scanned with CamScanner

In a similar way, an M^{th} order FIR filter can be implemented by a lattice structure with M-stages as shown in fig below:

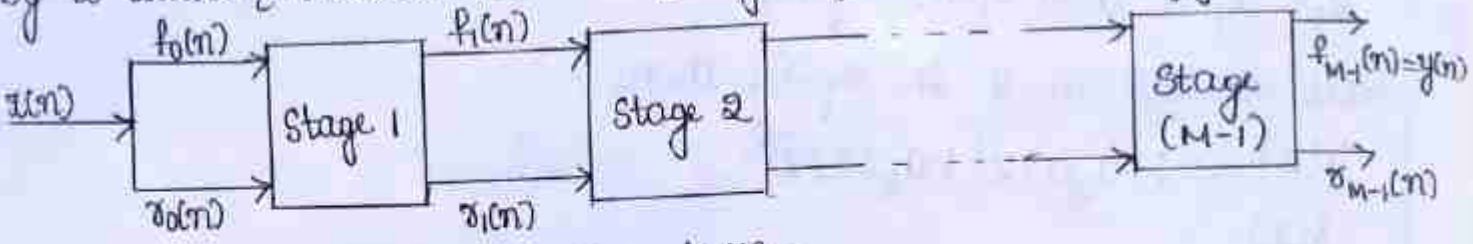


fig : M-stage lattice

The following formulae are useful for converting direct form-I to lattice :-

For $m = M, M-1, \ldots, 2, 1$ do

$$K_m = a_m(m) \longrightarrow (8)$$

$$a_{m-1}(i) = \frac{a_m(i) - a_m(m)\, a_m(m-i)}{1 - K_m^2} \; ; 1 \le i \le m-1 \longrightarrow (9)$$

From the above expression, it clears that it fails if $K_m = 1$. This is an indication that there is a zero on the unit circle. If $K_m = 1$, we can simply factor out this root from A(z) and the above recursive formula can be applied for the reduced order system.

Derive the lattice filter co-efficients K_1 and K_2 :-

For $m = 2$ and $m = 1$,

Eq(8) becomes, $\boxed{K_2 = a_2(2) \text{ and } K_1 = a_1(1)}$

For $m = 2$ and $i = 1$, Eq(9) gives

$$a_1(1) = \frac{a_2(1) - a_2(2)\, a_2(1)}{1 - K_2^2} = \frac{a_2(1)[1 - a_2(2)]}{1 - a_2^2(2)}$$

$$a_1(1) = \frac{a_2(1)[1 - a_2(2)]}{(1 - a_2(2))(1 + a_2(2))} = \frac{a_2(1)}{(1 + a_2(2))} = K_1$$

Scanned with CamScanner

5) Frequency Sampling Structure :-

In frequency sampling structure, the set of equally spaced frequency samples in the desired frequency response:

$$W_k = \frac{2\pi}{M}(K+\alpha) \; ; \; K = 0, 1, \ldots \left(\frac{M-1}{2}\right) \; ; \; M = \text{odd}$$

$$; \; K = 0, 1, \ldots \left(\frac{M}{2} - 1\right) \; ; \; M = \text{Even}$$

Where $\alpha = 0$ (or) $1/2$ are the parameters that characterize the filter.

W.K.T frequency response is given by,

$$H(w) = H(e^{jw}) = \sum_{n=0}^{M-1} h(n)\, e^{-jwn}$$

∴ The values of $H(w)$ at frequencies $W_k = \frac{2\pi}{M}(K+\alpha)$ are,

$$H(K+\alpha) = H\left(\frac{2\pi}{M}(K+\alpha)\right)$$

$$= \sum_{n=0}^{M-1} h(n)\, e^{-j\frac{2\pi}{M}(K+\alpha)n} \; ; \; K = 0, 1, \ldots (M-1) \longrightarrow (1)$$

These set of values $\{H(K+\alpha)\}$ are called the frequency samples of the frequency response $H(w)$.

Expressing $h(n)$ in terms of $H(K+\alpha)$,

$$h(n) = \frac{1}{M} \sum_{K=0}^{M-1} H(K+\alpha)\, e^{j\frac{2\pi}{M}(K+\alpha)n} \; ; \; n = 0, 1, \ldots (M-1) \longrightarrow (2)$$

when $\alpha = 0$, Eq(2) is IDFT of $H(K)$.

∴ $$H(z) = \sum_{n=0}^{M-1} h(n)\, z^{-n} \longrightarrow (3)$$

Substitute Eq(2) in Eq(3),

$$H(z) = \sum_{n=0}^{M-1} \left[\frac{1}{M} \sum_{K=0}^{M-1} H(K+\alpha)\, e^{j\frac{2\pi}{M}(K+\alpha)n}\right] z^{-n}$$

Scanned with CamScanner

Interchanging the order of summations,

$$H(z) = \sum_{K=0}^{M-1} H(K+\alpha)\left[\frac{1}{M}\sum_{n=0}^{M-1}\left(e^{j\frac{2\pi}{M}(K+\alpha)}z^{-1}\right)^n\right]$$

$$= \sum_{K=0}^{M-1} H(K+\alpha)\cdot\frac{1}{M}\cdot\frac{1-\left(e^{j\frac{2\pi}{M}(K+\alpha)}z^{-1}\right)^M}{1-e^{j\frac{2\pi}{M}(K+\alpha)}z^{-1}}$$

$$= \underbrace{\frac{1-e^{j2\pi\alpha}z^{-M}}{M}}_{H_1(z)} \underbrace{\sum_{K=0}^{M-1}\frac{H(K+\alpha)}{1-e^{j\frac{2\pi}{M}(K+\alpha)}z^{-1}}}_{H_2(z)} \longrightarrow (4)$$

Thus, the system function $H(z)$ is characterized by the set of frequency samples $\{H(K+\alpha)\}$ instead of $\{h(n)\}$. The cascade realization for Eq(4) is shown in fig below:

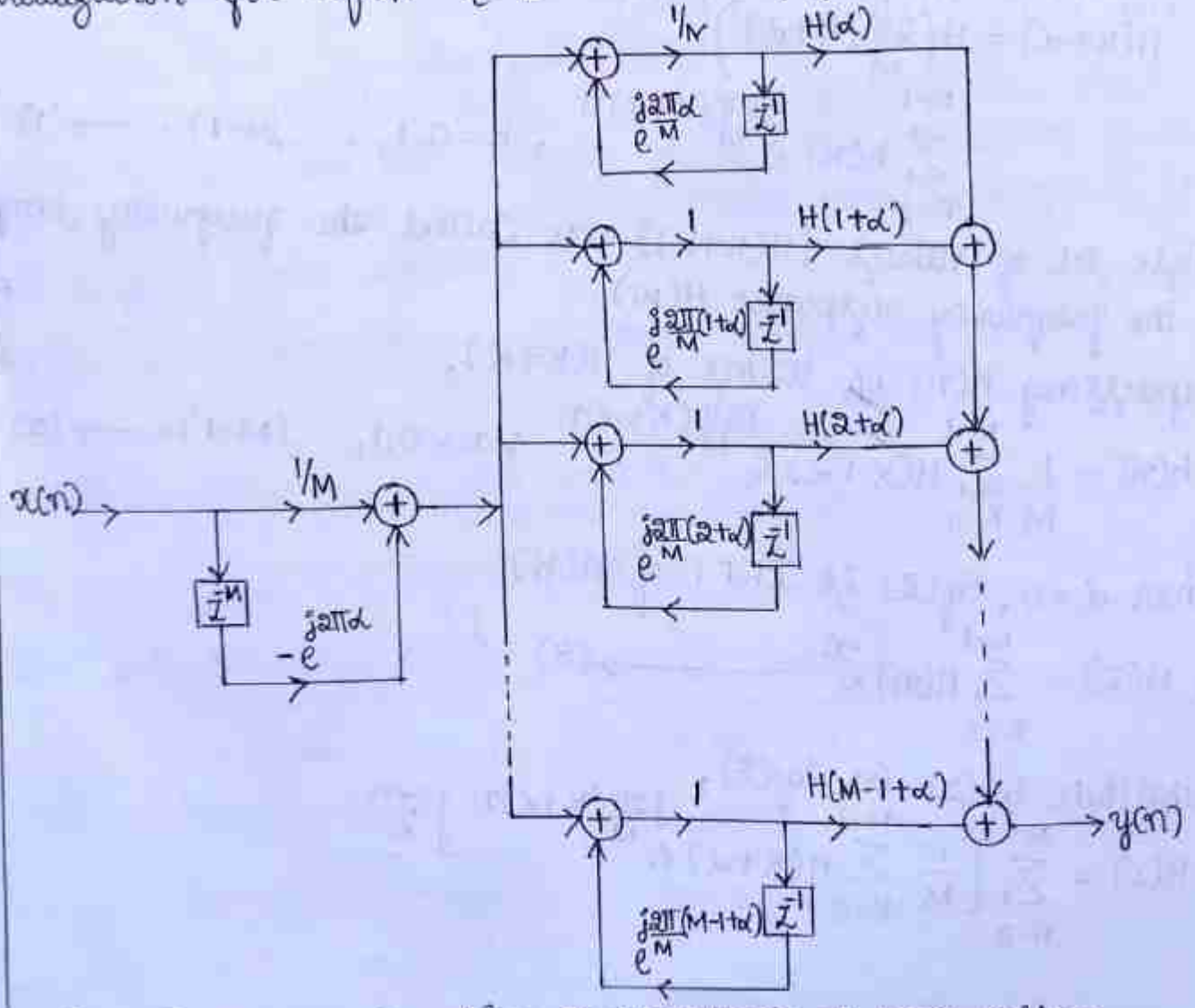


fig: Frequency-Sampling realization of FIR filter.

Scanned with CamScanner

The frequency sampling structure can be simplified further Exploiting the symmetry in $H(K+\alpha)$,

i.e., $H(K) = H^*(M-K)$; for $\alpha = 0$

and $H\left(K+\frac{1}{2}\right) = H\left(M-K-\frac{1}{2}\right)$; for $\alpha = \frac{1}{2}$.

Thus for $\alpha = 0$ the system function $H_2(z)$ in Eq(4) reduces to,

$$H_2(z) = \frac{H(0)}{1-z^{-1}} + \sum_{K=1}^{\frac{M-1}{2}} \frac{A(K) + B(K)z^{-1}}{1 - 2\cos\left(\frac{2\pi K}{M}\right)z^{-1} + z^{-2}} \quad ; M = \text{odd}$$

and

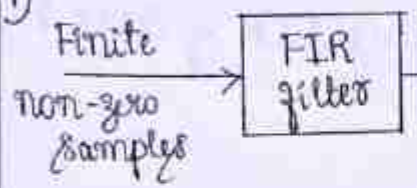
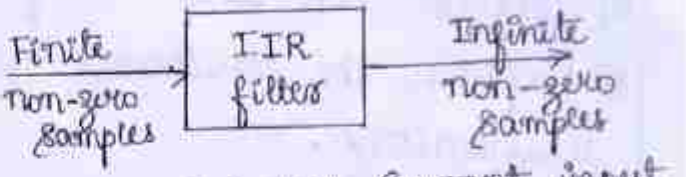
$$H_2(z) = \frac{H(0)}{1-z^{-1}} + \frac{H\left(\frac{M}{2}\right)}{1+z^{-1}} + \sum_{K=1}^{\frac{M}{2}-1} \frac{A(K) + B(K)z^{-1}}{1 - 2\cos\left(\frac{2\pi K}{M}\right)z^{-1} + z^{-2}} \quad ; M = \text{even} \qquad \rightarrow (5)$$

where $A(K) = H(K) + H(M-K)$

and $B(K) = H(K)e^{-j\frac{2\pi K}{M}} + H(M-K)e^{j\frac{2\pi K}{M}}$ $\longrightarrow (6)$

Scanned with CamScanner

Comparison of FIR and IIR filters :-

FIR filter	IIR filter
1) 	1) 
2) FIR filter uses only current and past input digital samples to obtain a current output sample value. It doesnot utilize past output samples.	2) IIR filter uses current input sample value, past inputs and outputs to obtain current output sample value.
3) Simple FIR equation is ; $y(n) = h(0)x(n) + h(1)x(n-1) + h(2)x(n-2) + h(3)x(n-3) + h(4)x(n-4)$	3) Simple IIR equation is, $y(n) = b(0)x(n) + b(1)x(n-1) + b(2)x(n-2) + b(3)x(n-3) + a(1)y(n-1) + a(2)y(n-2) + a(3)y(n-3)$
4) Transfer function of FIR filter will have only zeros.	4) Transfer function of IIR filter will have both zeros and poles.
5) FIR filters are preferred due to its linear phase response and also they are non-recursive. Feedback is not involved in FIR. Hence, they are stable.	5) IIR filters are not stable as they are recursive in nature and feedback is also involved in the process of calculating output sample values.
6) FIR filter consume low power.	6) IIR filter need more power due to more co-efficients in the design.
7) FIR have no analog equivalent	7) IIR filters have analog equivalent.
8) FIR filters are less efficient.	8) IIR filters are more efficient.
9) FIR filters are used as antialising, lowpass and baseband filters.	9) IIR filters are used as notch (bandstop), bandpass filters.

$\rightarrow y(n)$

Scanned with CamScanner

Questions :-

1) Realize FIR linear-phase filter for 'M' to be Even or 'M' to be odd.
2) Compare IIR and FIR filters.
3) Explain the structures used for realizing FIR filters by illustrations.

Scanned with CamScanner

Problems on FIR filter :-

1) For the given difference equation, draw direct form, linear-phase and Cascade structure :

$$y(n) = x(n) + \frac{5}{2}x(n-1) + 2x(n-2) + 2x(n-3)$$

Sol^n :-

Given: $y(n) = x(n) + \frac{5}{2}x(n-1) + 2x(n-2) + 2x(n-3)$

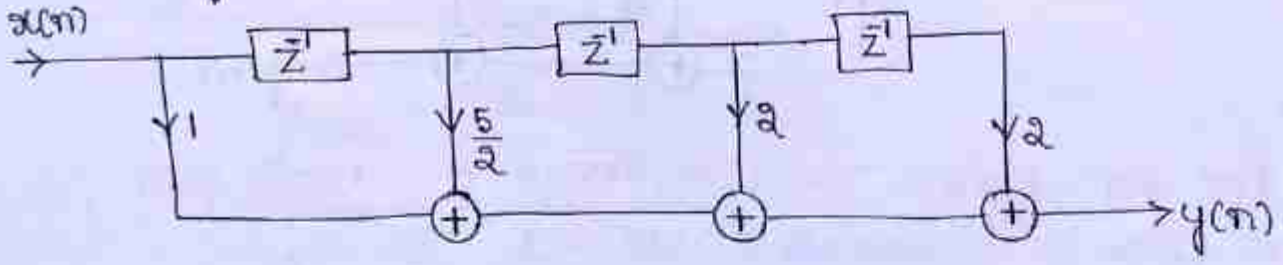
Apply Z-transform on both sides,

$$Y(z) = X(z) + \frac{5}{2}X(z)z^{-1} + 2X(z)z^{-2} + 2X(z)z^{-3}$$

$$Y(z) = X(z)\left[1 + \frac{5}{2}z^{-1} + 2z^{-2} + 2z^{-3}\right]$$

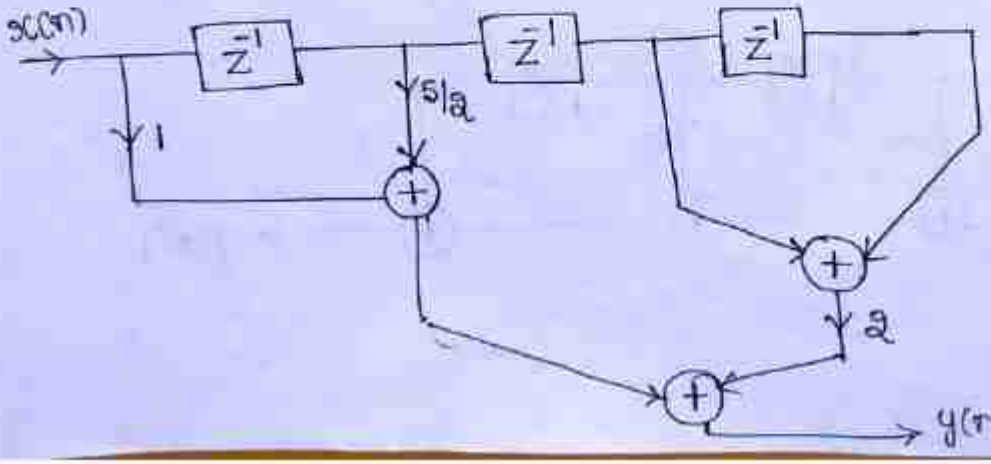
$$H(z) = \frac{Y(z)}{X(z)} = 1 + \frac{5}{2}z^{-1} + 2z^{-2} + 2z^{-3}$$

Direct form :-



Linear-phase Structure :-

$$H(z) = 1 + \frac{5}{2}z^{-1} + 2z^{-2} + 2z^{-3} = 1 + \frac{5}{2}z^{-1} + 2(z^{-2} + z^{-3})$$



Scanned with CamScanner

Cascade Structure :-

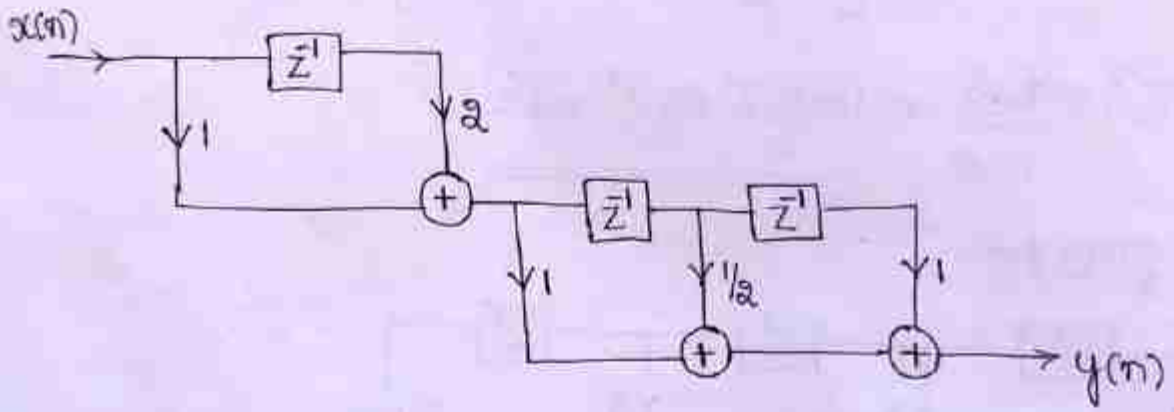
$$H(z) = 1 + \frac{5}{2} z^{-1} + 2z^{-2} + 2z^{-3}$$

One of the root is $-2 \longrightarrow (1 + 2z^{-1})$

$z^{-1} = -2$

	K	z^{-1}	z^{-2}	z^{-3}
	1	$\frac{5}{2}$	2	2
	0	-2	-1	-2
	1	$\frac{1}{2}$	1	0

$$\therefore H(z) = \left(1 + \frac{5}{2} z^{-1} + 2z^{-2} + 2z^{-3}\right) = \underbrace{(1 + 2z^{-1})}_{H_1(z)} \underbrace{\left(1 + \frac{1}{2} z^{-1} + z^{-2}\right)}_{H_2(z)}$$

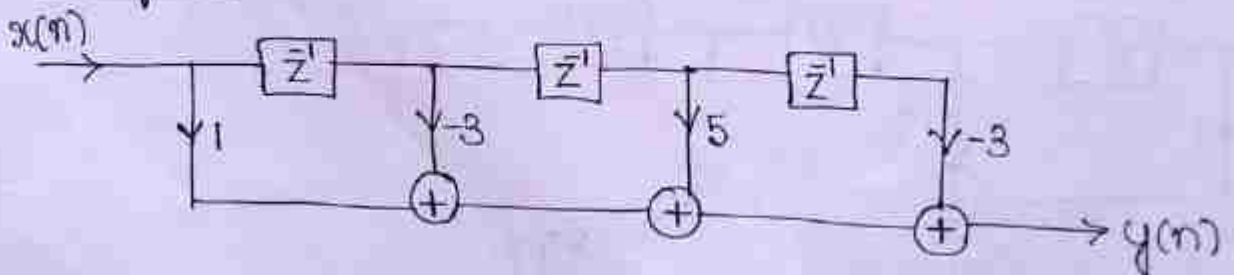


2) For the given System function, draw the direct form, linear phase and cascade structure :

$$H(z) = 1 - 3z^{-1} + 5z^{-2} - 3z^{-3}$$

Soln : Given : $H(z) = 1 - 3z^{-1} + 5z^{-2} - 3z^{-3}$

Direct form :-

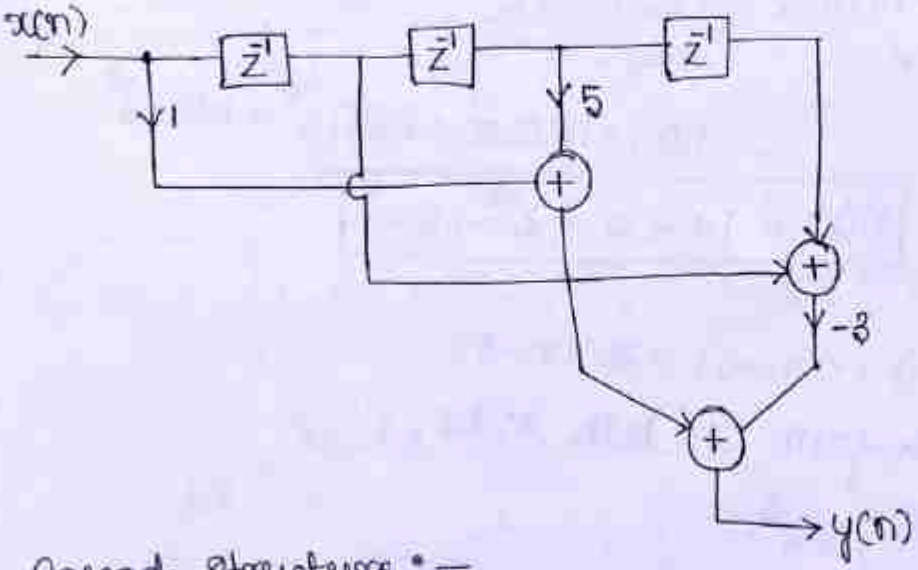


Scanned with CamScanner

Linear-phase Structure :-

$$H(z) = 1 - 3z^{-1} + 5z^{-2} - 3z^{-3}$$

$$= 1 + 5z^{-2} - 3(z^{-1} + z^{-3})$$



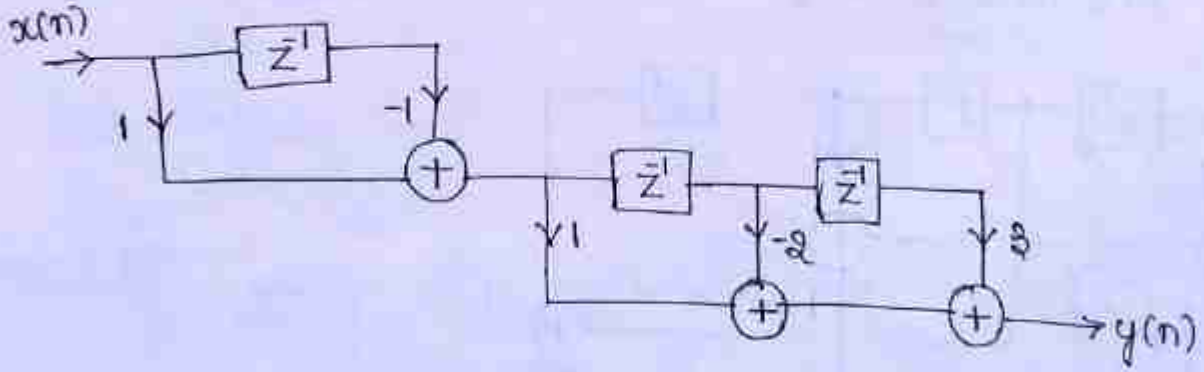
Cascade Structure :-

$$H(z) = 1 - 3z^{-1} + 5z^{-2} - 3z^{-3}$$

One of the root is 1 $\longrightarrow (1 - z^{-1})$

	K	z^{-1}	z^{-2}	z^{-3}
1	1	-3	5	-3
	0	1	-2	3
	1	-2	3	0

$$H(z) = (1 - z^{-1})(1 - 2z^{-1} + 3z^{-2})$$



Scanned with CamScanner

3) For the given impulse response, draw direct form, linear phase and cascade form: $h(n) = \{1, 2, 1, 2\}$

Sol^n: Given $h(n) = \{\underset{\uparrow}{1}, 2, 1, 2\}$

W.K.T. $H(z) = \sum_{n=-\infty}^{\infty} h(n) z^{-n} = \sum_{n=0}^{3} h(n) z^{-n}$

$= h(0) + h(1) z^{-1} + h(2) z^{-2} + h(3) z^{-3}$

$$\boxed{H(z) = 1 + 2z^{-1} + z^{-2} + 2z^{-3}}.$$

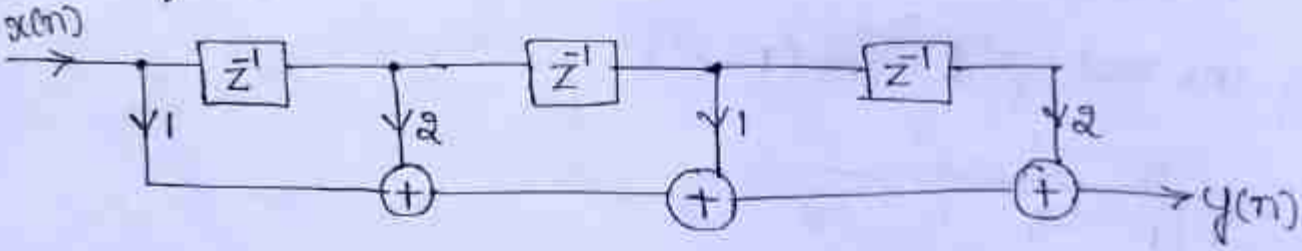
(or)

$h(n) = \delta(n) + 2\delta(n-1) + \delta(n-2) + 2\delta(n-3)$

Apply Z-transform on both sides,

$H(z) = 1 + 2z^{-1} + z^{-2} + 2z^{-3}$.

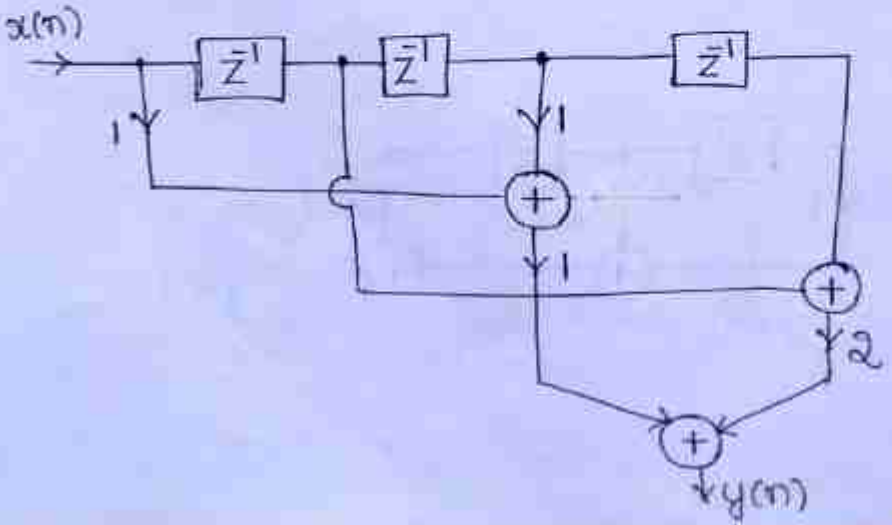
Direct form :-



Linear-phase structure :-

$H(z) = 1 + 2z^{-1} + z^{-2} + 2z^{-3}$

$H(z) = (1 + z^{-2}) + 2(z^{-1} + z^{-3})$



Scanned with CamScanner

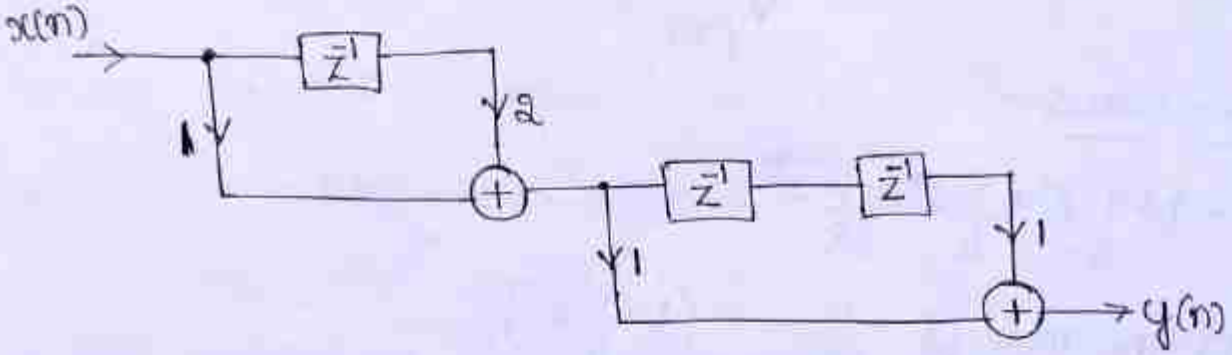
Cascade Structure :-

$H(z) = 1 + 2z^{-1} + z^{-2} + 2z^{-3}$

One of the root is $-2 \longrightarrow (1+2z^{-1})$

	K	z^{-1}	z^{-2}	z^{-3}
-2	1	2	1	2
	0	-2	0	-2
	1	0	1	0

$\therefore H(z) = (1+2z^{-1}+z^{-2}+2z^{-3}) = (1+2z^{-1})(1+z^{-2})$



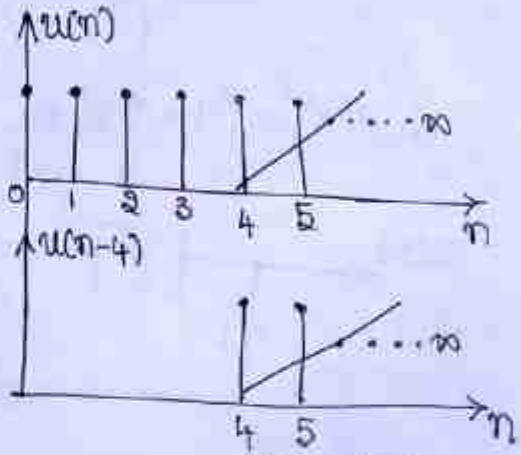
4) For the given impulse response, draw direct form, linear phase structure and cascade form:

$h(n) = \left(\frac{1}{2}\right)^n [u(n) - u(n-4)]$

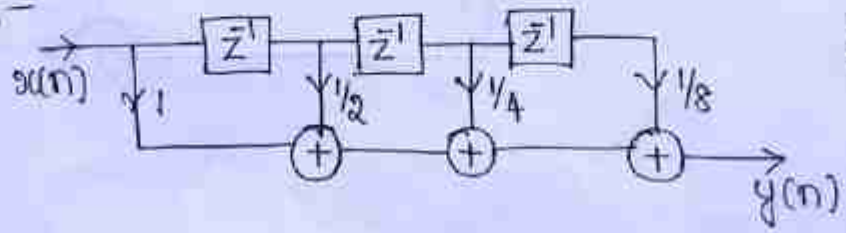
Soln:

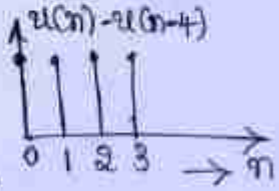
$$H(z) = \sum_{n=0}^{3} h(n) z^{-n} = \sum_{n=0}^{3} \left(\frac{1}{2}\right)^n z^{-n}$$

$$H(z) = 1 + \frac{1}{2}z^{-1} + \frac{1}{4}z^{-2} + \frac{1}{8}z^{-3}$$



Direct form :-

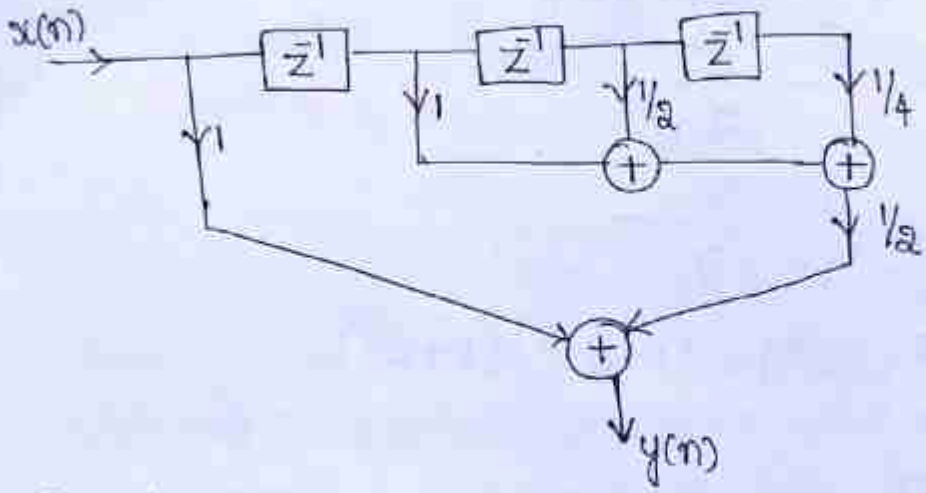




Scanned with CamScanner

Linear phase :-

$$H(z) = 1+\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}+\frac{1}{8}z^{-3} = 1+\frac{1}{2}\left(z^{-1}+\frac{1}{2}z^{-2}+\frac{1}{4}z^{-3}\right)$$



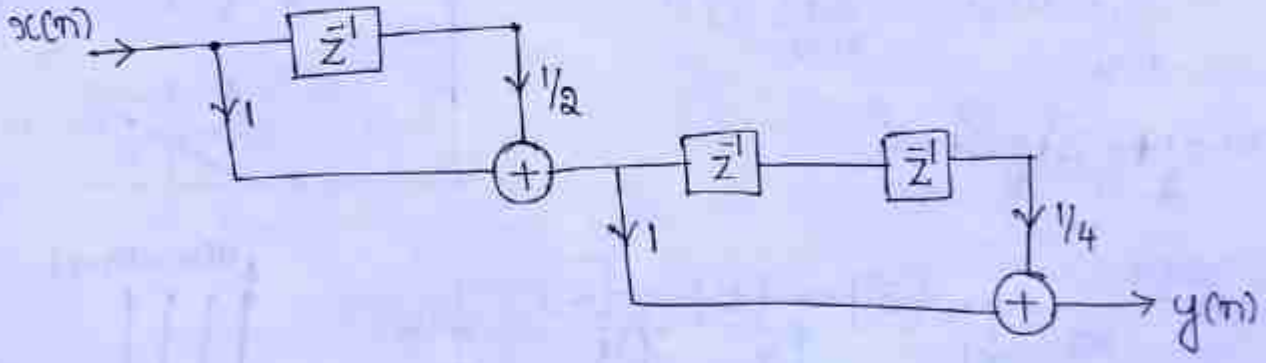
Cascade form :-

$$H(z) = 1+\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}+\frac{1}{8}z^{-3}$$

One of the root is $-\frac{1}{2} \longrightarrow \left(1+\frac{1}{2}z^{-1}\right)$

	K	z^{-1}	z^{-2}	z^{-3}
$-\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$
	0	$-\frac{1}{2}$	0	$-\frac{1}{8}$
	1	0	$\frac{1}{4}$	0

$$\therefore H(z) = \left(1+\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2}+\frac{1}{8}z^{-3}\right) = \left(1+\frac{1}{2}z^{-1}\right)\left(1+\frac{1}{4}z^{-2}\right)$$



Scanned with CamScanner

5) Given $H(z) = (1+0.6z^{-1})^5$ (i) Realize in direct form.
(ii) Realize as a cascade of 1st order section only.
(iii) Realize as a cascade of 1st and 2nd order.

Sol^n: $H(z) = (1+0.6z^{-1})^5$

$= (1+0.6z^{-1})(1+0.6z^{-1})^2(1+0.6z^{-1})^2$

$= (1+0.6z^{-1})(1+0.36z^{-2}+1.2z^{-1})(1+0.36z^{-2}+1.2z^{-1})$

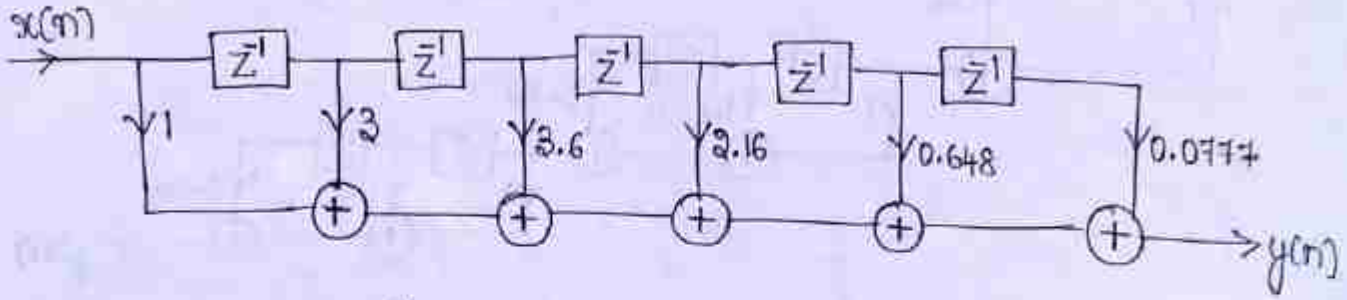
$= (1+0.6z^{-1})(1+0.36z^{-2}+1.2z^{-1}+0.36z^{-2}+0.1296z^{-4}+$
$+0.432z^{-3}+1.2z^{-1}+0.432z^{-3}+1.44z^{-2})$

$= (1+0.6z^{-1})(1+2.16z^{-2}+2.4z^{-1}+0.864z^{-3}+0.1296z^{-4})$

$= 1+2.4z^{-1}+2.16z^{-2}+0.864z^{-3}+0.1296z^{-4}+0.6z^{-1}+1.296z^{-3}$
$+1.44z^{-2}+0.5184z^{-4}+0.0777z^{-5}$

$H(z) = 1+3z^{-1}+3.6z^{-2}+2.16z^{-3}+0.648z^{-4}+0.0777z^{-5}$

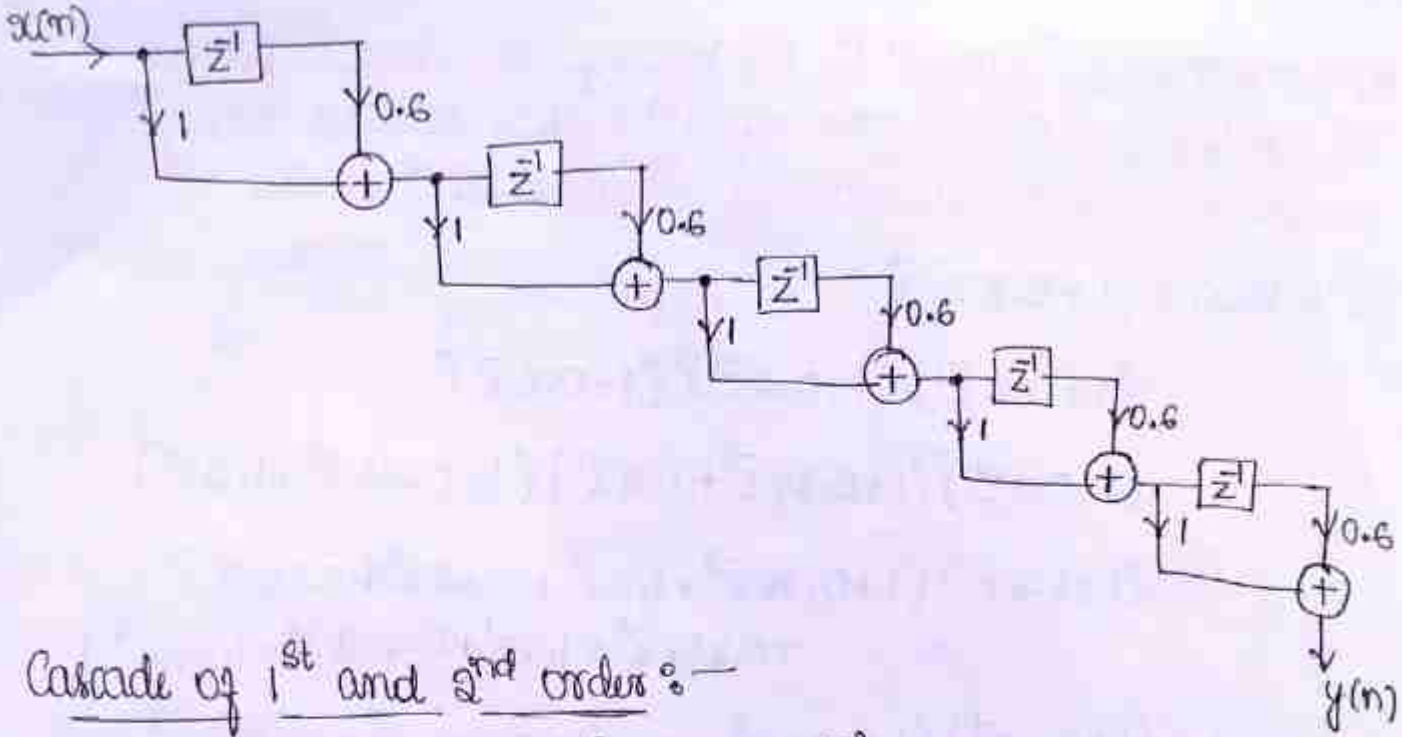
Direct form :-



Cascade form in 1st order :-

$H(z) = (1+0.6z^{-1})(1+0.6z^{-1})(1+0.6z^{-1})(1+0.6z^{-1})(1+0.6z^{-1})$

Scanned with CamScanner



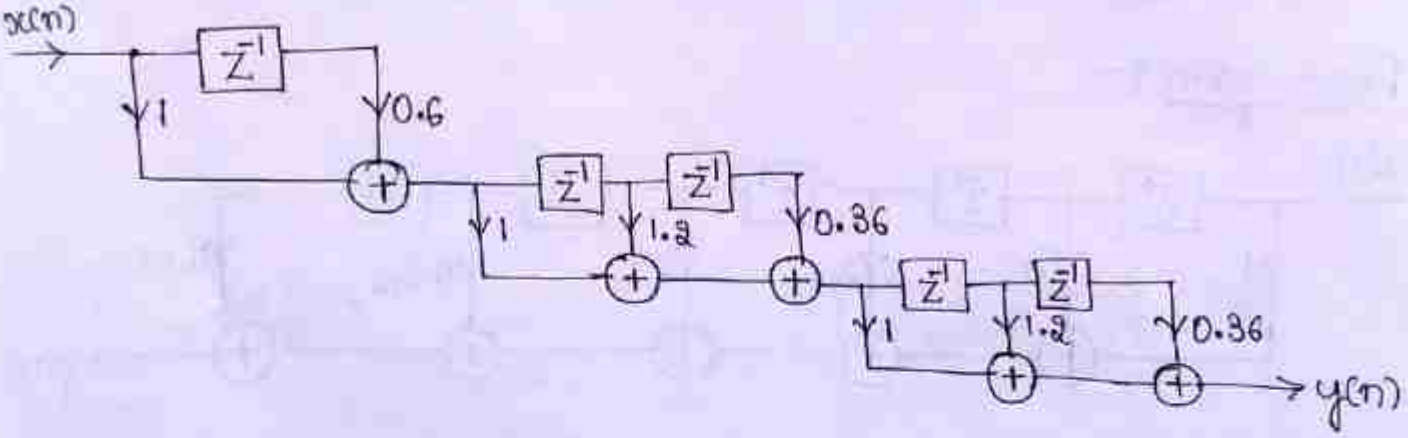
Cascade of 1st and 2nd order :-

$$H(z) = (1+0.6z^{-1})(1+0.6z^{-1})^2(1+0.6z^{-1})^2$$

$$H_1(z) = (1+0.6z^{-1})$$

$$H_2(z) = (1+0.6z^{-1})^2 = (1+0.36z^{-2}+1.2z^{-1}) = 1+1.2z^{-1}+0.36z^{-2}.$$

$$H_3(z) = (1+0.6z^{-1})^2 = 1+1.2z^{-1}+0.36z^{-2}.$$



6) Obtain direct form-I and lattice structure for the system described by the difference equation,

$$y(n) = x(n) + \frac{2}{5}x(n-1) + \frac{3}{4}x(n-2) + \frac{1}{3}x(n-3)$$

Soln :- Given : $y(n) = x(n) + \frac{2}{5}x(n-1) + \frac{3}{4}x(n-2) + \frac{1}{3}x(n-3)$

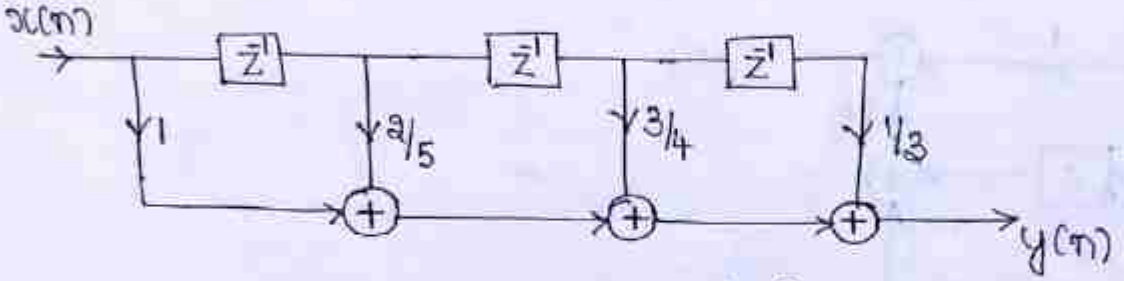
Apply Z-transform on both sides,

$$Y(z) = X(z) + \frac{2}{5}z^{-1}X(z) + \frac{3}{4}z^{-2}X(z) + \frac{1}{3}z^{-3}X(z)$$

Scanned with CamScanner

$$\therefore H(z) = \frac{Y(z)}{X(z)} = 1 + \frac{2}{5}z^{-1} + \frac{3}{4}z^{-2} + \frac{1}{3}z^{-3}$$

Direct form :



For lattice structure, refer page No. 8

* Realize a linear phase FIR filter with the following impulse response:

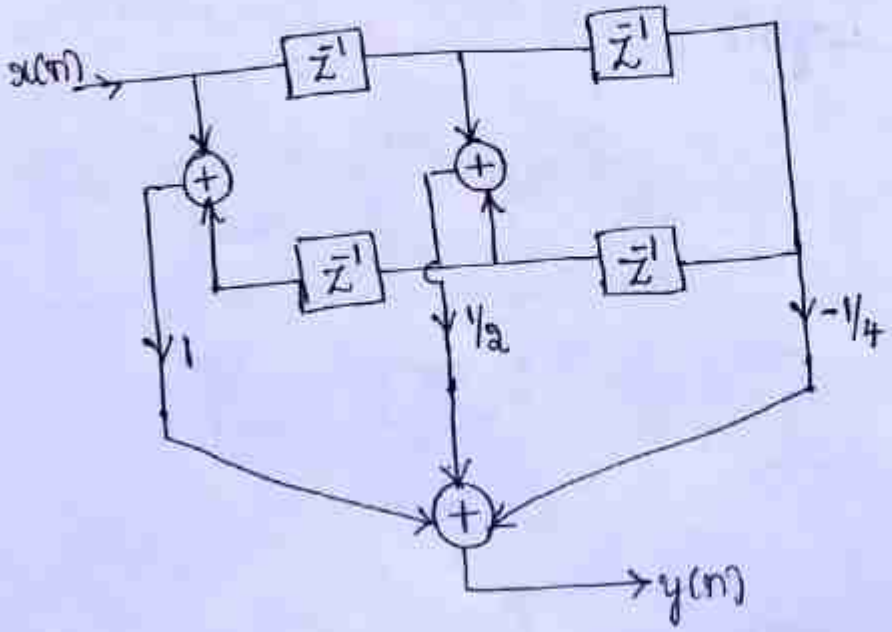
$$h(n) = \delta(n) + \frac{1}{2}\delta(n-1) - \frac{1}{4}\delta(n-2) + \frac{1}{2}\delta(n-3) + \delta(n-4)$$

Sol^n:-

Given $h(n) = \delta(n) + \frac{1}{2}\delta(n-1) - \frac{1}{4}\delta(n-2) + \frac{1}{2}\delta(n-3) + \delta(n-4)$

Apply Z-transform on both sides,

$$H(z) = 1 + \frac{1}{2}z^{-1} - \frac{1}{4}z^{-2} + \frac{1}{2}z^{-3} + z^{-4}$$

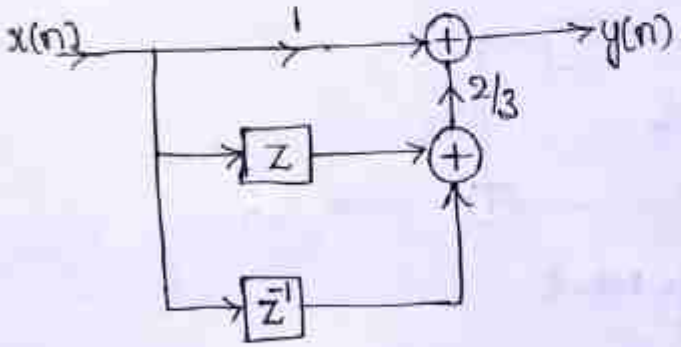


Scanned with CamScanner

* Realize the following system function by linear phase FIR structure $H(z) = \frac{2}{3}z + 1 + \frac{2}{3}z^{-1}$.

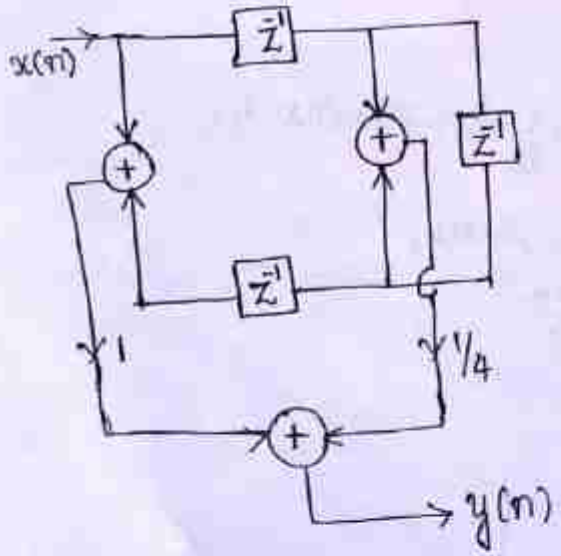
Soln:-

$$H(z) = 1 + \frac{2}{3}(z + z^{-1})$$



* Obtain linear-phase realization of $H(z) = 1 + \frac{z^{-1}}{4} + \frac{z^{-2}}{4} + z^{-3}$

Soln:-



Scanned with CamScanner

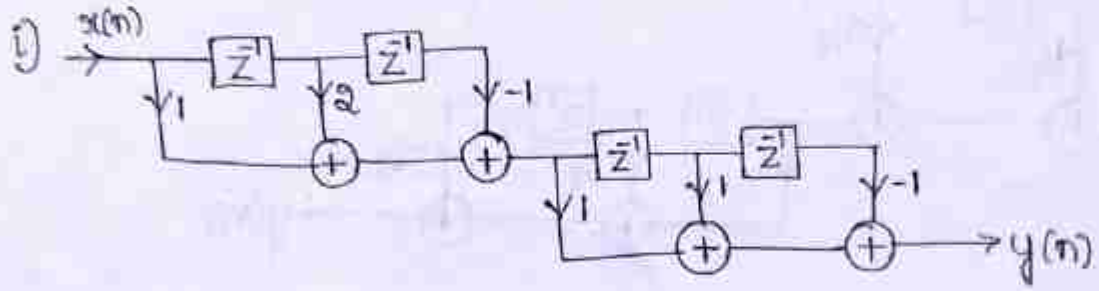
7) i) Obtain the cascade realization of the system function

$H(z) = (1+2z^{-1}-z^{-2})(1+z^{-1}-z^{-2})$

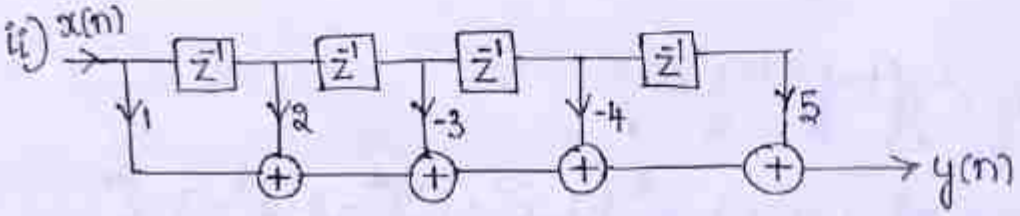
ii) Determine the direct form realization of the system function $H(z) = 1+2z^{-1}-3z^{-2}-4z^{-3}+5z^{-4}$.

Soln:

i)



ii)

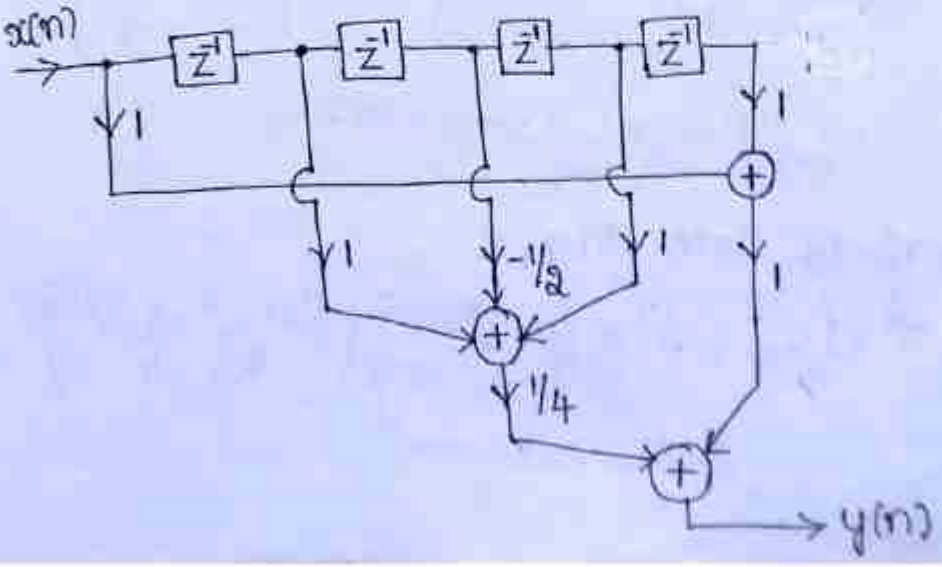


8) Realize the linear-phase FIR filter having the following impulse response $h(n) = \delta(n) + \frac{1}{4}\delta(n-1) - \frac{1}{8}\delta(n-2) + \frac{1}{4}\delta(n-3) + \delta(n-4)$.

Soln:- Apply Z-transform,

$$H(z) = 1 + \frac{1}{4}z^{-1} - \frac{1}{8}z^{-2} + \frac{1}{4}z^{-3} + z^{-4}$$

$$H(z) = (1+z^{-4}) + \frac{1}{4}\left(z^{-1} - \frac{1}{2}z^{-2} + z^{-3}\right)$$

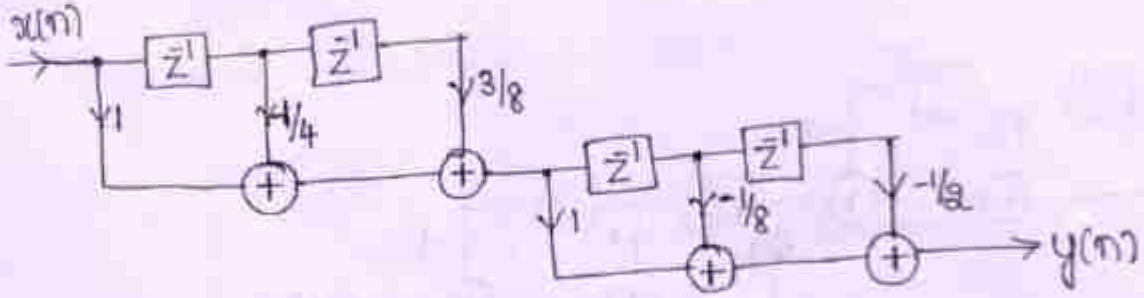


Scanned with CamScanner

9) Obtain the direct form and cascade form realizations for the transfer function: $H(z) = \left(1 - \frac{1}{4}z^{-1} + \frac{3}{8}z^{-2}\right)\left(1 - \frac{1}{8}z^{-1} - \frac{1}{2}z^{-2}\right)$

Soln:-

Cascade form :-

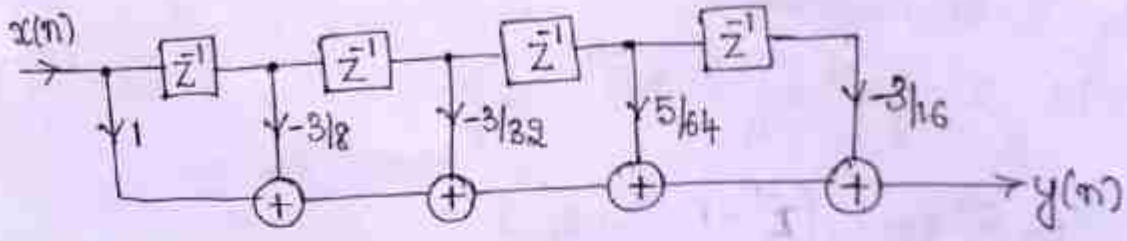


Direct form :-

$$H(z) = \left(1 - \frac{1}{4}z^{-1} + \frac{3}{8}z^{-2}\right)\left(1 - \frac{1}{8}z^{-1} - \frac{1}{2}z^{-2}\right)$$

$$= 1 - \frac{1}{8}z^{-1} - \frac{1}{2}z^{-2} - \frac{1}{4}z^{-1} + \frac{1}{32}z^{-2} + \frac{1}{8}z^{-3} + \frac{3}{8}z^{-2} - \frac{3}{64}z^{-3} - \frac{3}{16}z^{-4}$$

$$H(z) = 1 - \frac{3}{8}z^{-1} - \frac{3}{32}z^{-2} + \frac{5}{64}z^{-3} - \frac{3}{16}z^{-4}$$



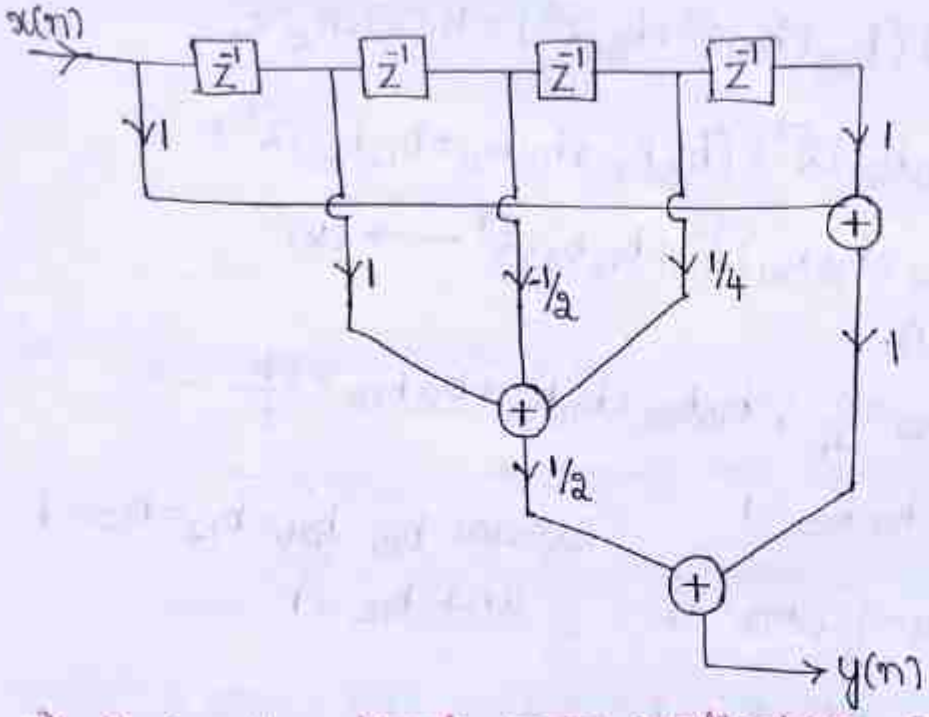
10) Realize the linear-phase FIR filter having the following impulse response:

$$h(n) = \delta(n) + \frac{1}{2}\delta(n-1) - \frac{1}{4}\delta(n-2) + \frac{1}{8}\delta(n-3) + \delta(n-4).$$

Soln: Apply Z-transform on both sides,

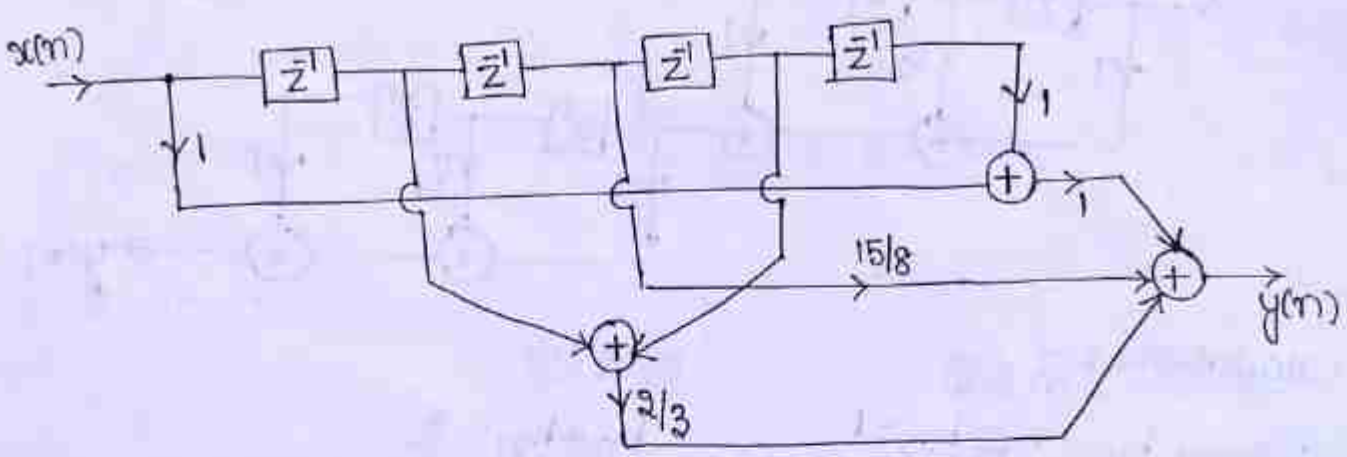
$$H(z) = 1 + \frac{1}{2}z^{-1} - \frac{1}{4}z^{-2} + \frac{1}{8}z^{-3} + z^{-4} = (1 + z^{-4}) + \frac{1}{2}\left(z^{-1} - \frac{1}{2}z^{-2} + \frac{1}{4}z^{-3}\right)$$

Scanned with CamScanner



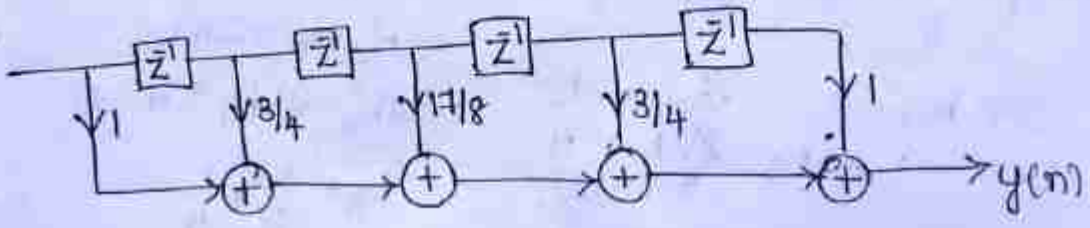
11) Obtain the direct form realization of linear phase FIR system given by, $H(z) = 1 + \frac{2}{3}z^{-1} + \frac{15}{8}z^{-2} + \frac{2}{3}z^{-3} + z^{-4}$

Soln: $H(z) = (1 + z^{-4}) + \frac{2}{3}(z^{-1} + z^{-3}) + \frac{15}{8}z^{-2}$



12) Realize the following system function in cascade form.
$H(z) = 1 + \frac{3}{4}z^{-1} + \frac{17}{8}z^{-2} + \frac{3}{4}z^{-3} + z^{-4}$ in (i) Direct form (ii) cascaded form. → (1)

Soln:-
1)



Scanned with CamScanner

Cascade form :-

$$H(z) = (b_{10} + b_{11}z^{-1} + b_{12}z^{-2})(b_{20} + b_{21}z^{-1} + b_{22}z^{-2}) = H_1(z) \cdot H_2(z)$$

$$= b_{10}b_{20} + (b_{10}b_{21} + b_{11}b_{20})z^{-1} + (b_{10}b_{22} + b_{11}b_{21} + b_{12}b_{20})z^{-2} + (b_{11}b_{22} + b_{12}b_{21})z^{-3} + b_{12}b_{22}z^{-4} \longrightarrow (2)$$

Compare eq(1) and eq(1),

$b_{10}b_{20} = 1$; $b_{10}b_{21} + b_{11}b_{20} = \frac{3}{4}$; $b_{10}b_{22} + b_{11}b_{21} + b_{12}b_{20} = \frac{17}{8}$

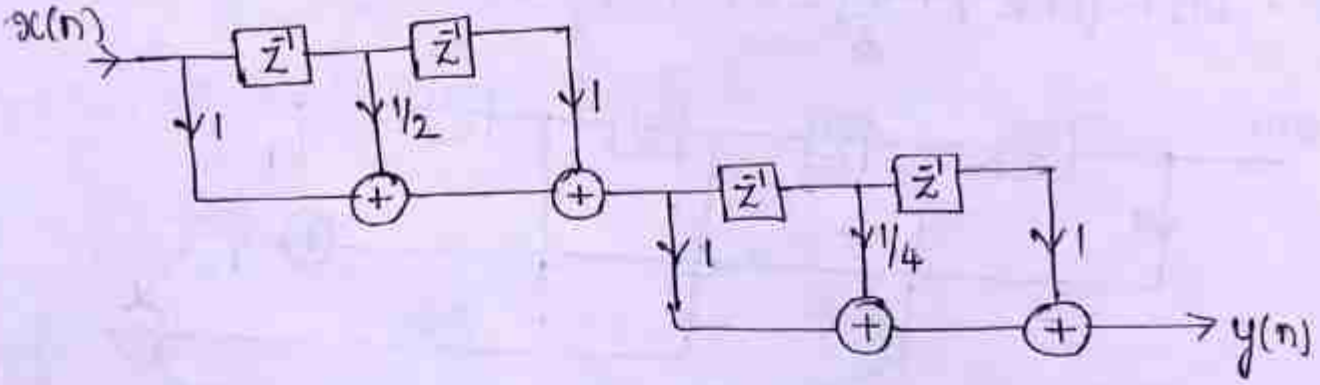
$b_{11}b_{22} + b_{12}b_{21} = \frac{3}{4}$; $b_{12}b_{22} = 1$

Assume $b_{10} = 1$ and $b_{12} = 1$

on solving, $b_{10} = 1$, $b_{11} = \frac{1}{2}$, $b_{12} = 1$,

and $b_{20} = 1$, $b_{21} = \frac{1}{4}$, $b_{22} = 1$

$\therefore H(z) = \left(1 + \frac{1}{2}z^{-1} + z^{-2}\right)\left(1 + \frac{1}{4}z^{-1} + z^{-2}\right)$



Calculations :-

① & ⑤

Assume $b_{10} = 1 \Rightarrow b_{20} = 1$

Assume $b_{12} = 1 \Rightarrow b_{22} = 1$

$(3) \Rightarrow b_{11}b_{21} + 2 = \frac{17}{8}$

$b_{11}b_{21} = \frac{17}{8} - 2 = \frac{17 - 16}{8} = \frac{1}{8}$

$\Rightarrow b_{21} = \frac{1}{8b_{11}} = \frac{2}{8 \times 1} = \frac{1}{4}$

② & ④

$b_{11} + b_{21} = \frac{3}{4}$

$b_{11} + \frac{1}{8b_{11}} = \frac{3}{4}$

$\frac{8b_{11}^2 + 1}{8b_{11}} = \frac{3}{4}$

$32b_{11}^2 + 4 = 24b_{11}$

$32b_{11}^2 - 24b_{11} + 4 = 0$

$b_{11} = \frac{1}{2}, \frac{1}{4}$

Scanned with CamScanner

FIR lattice Structure:

1) A FIR filter is given by, $y(n) = x(n) + \frac{2}{5}x(n-1) + \frac{3}{4}x(n-2) + \frac{1}{3}x(n-3)$. Draw the lattice structure.

Soln: Given : $y(n) = x(n) + \frac{2}{5}x(n-1) + \frac{3}{4}x(n-2) + \frac{1}{3}x(n-3)$

Apply z-transform on both sides,

$$Y(z) = X(z) + \frac{2}{5}z^{-1}X(z) + \frac{3}{4}z^{-2}X(z) + \frac{1}{3}z^{-3}X(z)$$

$$H(z) = \frac{Y(z)}{X(z)} = 1 + \frac{2}{5}z^{-1} + \frac{3}{4}z^{-2} + \frac{1}{3}z^{-3} = 1 + a_3(1)z^{-1} + a_3(2)z^{-2} + a_3(3)z^{-3}$$

The Conversion formulae used is,

For $m = M, M-1, \ldots 2, 1$ do

$$K_m = a_m(m)$$

$$a_{m-1}(i) = \frac{a_m(i) - a_m(m)\,a_m(m-i)}{1 - K_m^2} \quad ; 1 \le i \le m-1$$

Since, there are 3 zeros in $H(z)$, take $m = 3$

Step 1 :-

$m = 3$;

$$K_3 = a_3(3) = \frac{1}{3}$$

$$a_2(i) = \frac{a_3(i) - a_3(3)\,a_3(3-i)}{1 - K_3^2} \quad ; 1 \le i \le 2$$

$i = 1$, $a_2(1) = \frac{a_3(1) - a_3(3)\,a_3(2)}{1 - K_3^2} = \frac{\left(\frac{2}{5}\right) - \left(\frac{1}{3}\right)\left(\frac{3}{4}\right)}{1 - \left(\frac{1}{3}\right)^2} = \frac{0.15}{0.89} = 0.171$

Scanned with CamScanner

$$i=2,\ a_2(2)=\frac{a_3(2)-a_3(3)a_3(1)}{1-K_3^2}=\frac{\left(\frac{3}{4}\right)-\left(\frac{1}{3}\right)\left(\frac{2}{5}\right)}{1-\left(\frac{1}{3}\right)^2}=\frac{0.617}{0.89}=0.693$$

Step 2 :-

$m=2$;

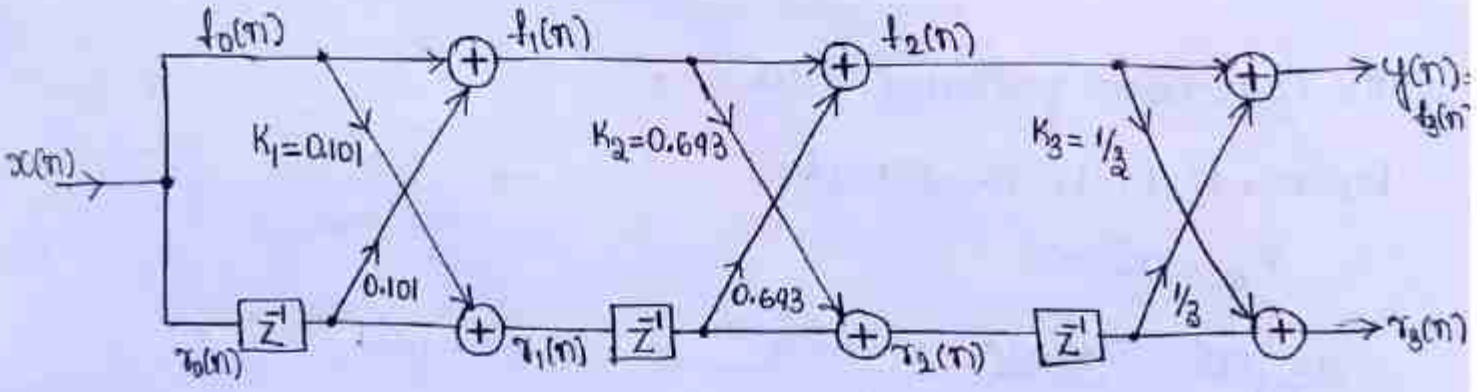
$$K_2=a_2(2)=0.693$$

$$a_1(i)=\frac{a_2(i)-a_2(2)a_2(2-i)}{1-K_2^2}\ ;\ 1\le i\le 1$$

$$i=1,\ a_1(1)=\frac{a_2(1)-a_2(2)a_2(1)}{1-K_2^2}=\frac{0.171-0.693(0.171)}{1-(0.693)^2}=\frac{0.0525}{0.52}$$

$$a_1(1)=0.101=K_1$$

The lattice structure is as follows :



2) Consider a FIR filter with system function :

$H(z)=1+2.82z^{-1}+3.4048z^{-2}+1.74z^{-3}$. Draw the lattice structure.

Soln :-

Given : $H(z)=1+2.82z^{-1}+3.4048z^{-2}+1.74z^{-3}$ ——→(1)

$$H(z)=1+a_3(1)z^{-1}+a_3(2)z^{-2}+a_3(3)z^{-3} \text{ ——→(2)}$$

Comparing eq(1) and eq(2),

$$\boxed{a_3(1)=2.82,\ a_3(2)=3.4048,\ a_3(3)=1.74}$$

Scanned with CamScanner

The Conversion formulae used is,

For $m = M, M-1, \ldots 2, 1$ do

$$K_m = a_m(m)$$

$$a_{m-1}(i) = \frac{a_m(i) - a_m(m)\,a_m(m-i)}{1 - K_m^2} \quad ; \; 1 \le i \le m-1$$

Since, there are 3 zeros in $H(z)$, take $m = 3$

Step 1 :–

$m = 3$;

$$K_3 = a_3(3) = \underline{1.74}$$

$$a_2(i) = \frac{a_3(i) - a_3(3)\,a_3(3-i)}{1 - K_3^2} \quad ; \; 1 \le i \le 2$$

$i = 1$, $a_2(1) = \dfrac{a_3(1) - a_3(3)\,a_3(2)}{1 - K_3^2} = \dfrac{2.82 - 1.74(3.4048)}{1 - (1.74)^2} = \dfrac{-3.1044}{-2.0276} = \underline{1.531}$

$i = 2$, $a_2(2) = \dfrac{a_3(2) - a_3(3)\,a_3(1)}{1 - K_3^2} = \dfrac{3.4048 - 1.74(2.82)}{1 - (1.74)^2} = \dfrac{-1.502}{-2.0276} = \underline{0.7407}$

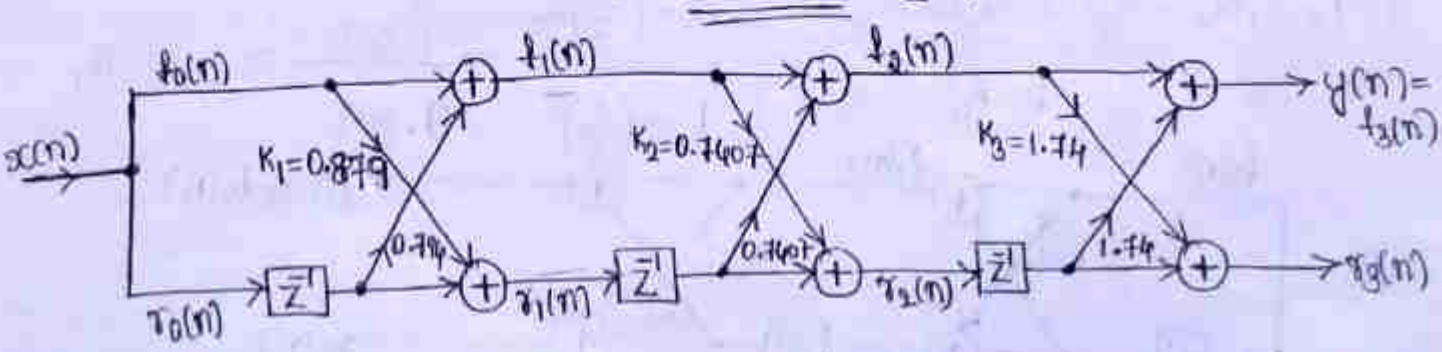
Step 2 :–

$m = 2$;

$$K_2 = a_2(2) = \underline{0.7407} \quad ; \; a_1(i) = \frac{a_2(i) - a_2(2)\,a_2(2-i)}{1 - K_2^2} \quad ; \; 1 \le i \le 1$$

$i = 1$, $a_2(1) = \dfrac{a_2(1) - a_2(2)\,a_2(1)}{1 - K_2^2} = \dfrac{1.531 - (0.7407)(1.531)}{1 - (0.7407)^2} = \dfrac{0.39699}{0.4514}$

$$K_1 = \underline{0.879} = a_2(1)$$



Scanned with CamScanner

3) Determine the Co-efficients K_m of lattice filter whose transfer function corresponding to FIR filter described by the transfer function $H(z) = 1 + 2z^{-1} + \frac{1}{3}z^{-2}$. Also draw the corresponding II-order lattice structure.

Soln :- Given : $H(z) = 1 + 2z^{-1} + \frac{1}{3}z^{-2}$ ⟶ (1)

$$H(z) = 1 + a_2(1)z^{-1} + a_2(2)z^{-2} \longrightarrow (2)$$

Comparing eq(1) and eq(2),

$$\boxed{a_2(1) = 2 \;;\; a_2(2) = \frac{1}{3}}$$

The Conversion formulae used is,

For $m = M, M-1, \ldots 2, 1$ do

$K_m = a_m(m)$

$$a_{m-1}(i) = \frac{a_m(i) - a_m(m)\,a_m(m-i)}{1 - K_m^2} \;;\; 1 \le i \le m-1$$

Since, there are 2 zeros in $H(z)$, take $m = 2$

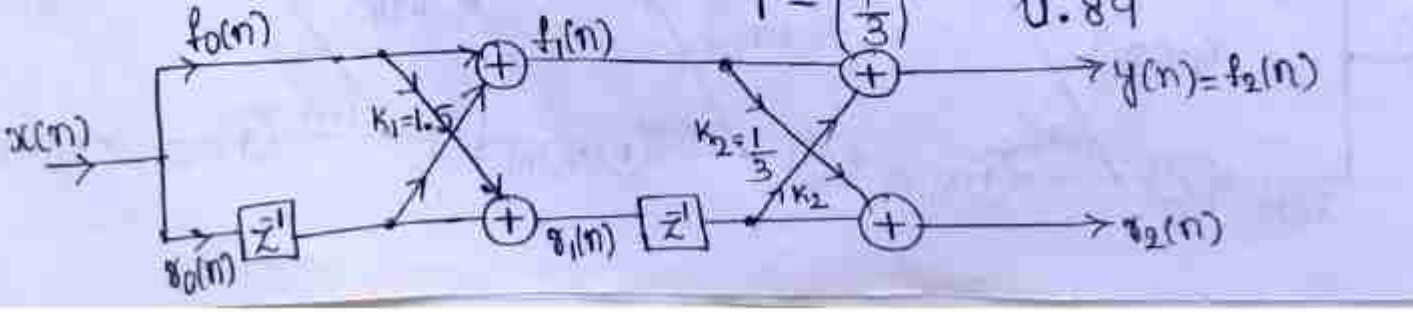
Step 1 :-

$m = 2$;

$$K_2 = a_2(2) = \frac{1}{3}$$

$$a_1(i) = \frac{a_2(i) - a_2(2)\,a_2(2-i)}{1 - K_2^2}$$

$i = 1$, $a_1(1) = \dfrac{a_2(1) - a_2(2)\,a_2(1)}{1 - K_2^2} = \dfrac{2 - \left(\frac{1}{3}\right)^2}{1 - \left(\frac{1}{3}\right)^2} = \dfrac{1.33}{0.89} = 1.5 = K_1$



Scanned with CamScanner

4) Determine the parameters K_m of the lattice filter corresponding to the FIR filter described by the system function:

$H(z) = 1 + 1.38z^{-1} + 1.311z^{-2} + 1.337z^{-3} + 0.9z^{-4}$

Sol^n:

Given: $H(z) = 1 + 1.38z^{-1} + 1.311z^{-2} + 1.337z^{-3} + 0.9z^{-4}$

Here, $a_4(1) = 1.38$; $a_4(2) = 1.311$; $a_4(3) = 1.337$; $a_4(4) = 0.9$

The conversion formulae used is,

For $m = M, M-1, \ldots 2, 1$ do

$K_m = a_m(m)$

$$a_{m-1}(i) = \frac{a_m(i) - a_m(m)a_m(m-i)}{1 - K_m^2} \quad ; 1 \le i \le m-1$$

Since, there are 4 zeros in $H(z)$, take $m = 4$

Step 1: $m = 4$;

$K_4 = a_4(4) = \underline{0.9}$

$$a_3(i) = \frac{a_4(i) - a_4(4)a_4(4-i)}{1 - K_4^2} \quad ; 1 \le i \le 3$$

$i = 1$, $a_3(1) = \dfrac{a_4(1) - a_4(4)a_4(3)}{1 - K_4^2} = \dfrac{1.38 - 0.9(1.337)}{1 - (0.9)^2} = \dfrac{0.1767}{0.19} = \underline{0.93}$

$i = 2$, $a_3(2) = \dfrac{a_4(2) - a_4(4)a_4(2)}{1 - K_4^2} = \dfrac{1.311 - 0.9(1.311)}{1 - (0.9)^2} = \dfrac{0.1311}{0.19} = \underline{0.69}$

$i = 3$, $a_3(3) = \dfrac{a_4(3) - a_4(4)a_4(1)}{1 - K_4^2} = \dfrac{1.337 - 0.9(1.38)}{1 - (0.9)^2} = \dfrac{0.095}{0.19} = \underline{0.5}$

Scanned with CamScanner

Step 2: $m = 3$;

$$K_3 = a_3(3) = \underline{0.5}$$

$$a_2(i) = \frac{a_3(i) - a_3(3)\,a_3(3-i)}{1 - K_3^2} \quad ; 1 \le i \le 2$$

$i=1$, $a_2(1) = \dfrac{a_3(1) - a_3(3)\,a_3(2)}{1 - K_3^2} = \dfrac{0.93 - 0.5(0.69)}{1 - (0.5)^2} = \dfrac{0.585}{0.75} = \underline{0.78}$

$i=2$, $a_2(2) = \dfrac{a_3(2) - a_3(3)\,a_3(1)}{1 - K_3^2} = \dfrac{0.69 - 0.5(0.93)}{1 - (0.5)^2} = \dfrac{0.225}{0.75} = \underline{0.3}$

Step 3: $m = 2$;

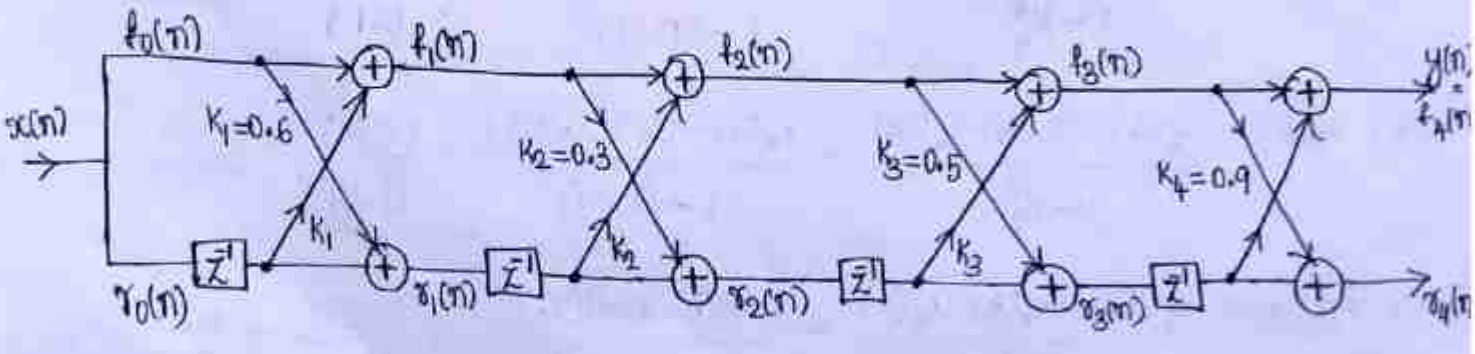
$$K_2 = a_2(2) = \underline{0.3}$$

$$a_1(i) = \frac{a_2(i) - a_2(2)\,a_2(2-i)}{1 - K_2^2} \quad ; 1 \le i \le 1$$

$i=1$, $a_1(1) = \dfrac{a_2(1) - a_2(2)\,a_2(1)}{1 - K_2^2} = \dfrac{0.78 - 0.3(0.78)}{1 - (0.3)^2} = \dfrac{0.546}{0.91} = \underline{0.6}$

Step 4: $m = 1$;

$$K_1 = a_1(1) = \underline{0.6}$$



Scanned with CamScanner

5) For the given difference equation, draw the lattice structure,
$y(n) = x(n) + 3.1x(n-1) + 5.5x(n-2) + 4.2x(n-3) + 2.3x(n-4)$.

Soln: Given: $y(n) = x(n) + 3.1x(n-1) + 5.5x(n-2) + 4.2x(n-3) + 2.3x(n-4)$

Apply z-transform,

$$Y(z) = X(z) + 3.1z^{-1}X(z) + 5.5z^{-2}X(z) + 4.2z^{-3}X(z) + 2.3z^{-4}X(z)$$

$$H(z) = \frac{Y(z)}{X(z)} = 1 + 3.1z^{-1} + 5.5z^{-2} + 4.2z^{-3} + 2.3z^{-4} \longrightarrow (1)$$

For FIR lattice structure,

$$H(z) = 1 + \sum_{i=1}^{M} a_m(i) z^{-i}$$

Here $M = 4$, $H(z) = 1 + \sum_{i=1}^{4} a_4(i) z^{-i}$

$$H(z) = 1 + a_4(1)z^{-1} + a_4(2)z^{-2} + a_4(3)z^{-3} + a_4(4)z^{-4} \longrightarrow (2)$$

Comparing the co-efficients in Eq(1) and Eq(2),

$a_4(1) = 3.1$; $a_4(2) = 5.5$; $a_4(3) = 4.2$; $a_4(4) = 2.3 = K_4$

To find the lattice co-efficients K_1, K_2 and K_3 :-

$K_1 = a_1(1)$, $K_2 = a_2(2)$, $K_3 = a_3(3)$

$$a_{m-1}(i) = \frac{a_m(i) - a_m(m)a_m(m-i)}{1 - K_m^2} \quad ; K_m = a_m(m)$$

Step 1 :-

$$m = 4, \quad a_3(i) = \frac{a_4(i) - a_4(4)a_4(4-i)}{1 - K_4^2} \quad ; K_4 = a_4(4) = 2.3$$

$1 \le i \le 3$

$$i = 1, \quad a_3(1) = \frac{a_4(1) - a_4(4)a_4(3)}{1 - K_4^2} = \frac{3.1 - 2.3(4.2)}{1 - (2.3)^2} = 1.5291$$

Scanned with CamScanner

$i=2$,

$$a_3(2) = \frac{a_4(2) - a_4(4)\,a_4(2)}{1-K_4^2} = \frac{5.5 - 2.3(5.5)}{1-(2.3)^2} = \underline{1.67}$$

$i=3$,

$$a_3(3) = \frac{a_4(3) - a_4(4)\,a_4(1)}{1-K_4^2} = \frac{4.2 - 2.3(3.1)}{1-(2.3)^2} = \underline{0.683}$$

Step 2 :-

$m=3$,

$$K_3 = a_3(3) = \underline{0.683}$$

$$a_2(i) = \frac{a_3(i) - a_3(3)\,a_3(3-i)}{1-K_3^2} \quad ; 1 \le i \le 2$$

$i=1$, $a_2(1) = \dfrac{a_3(1) - a_3(3)\,a_3(2)}{1-K_3^2} = \dfrac{1.5291 - 0.683(1.67)}{1-(0.683)^2} = \underline{0.7282}$

$i=2$, $a_2(2) = \dfrac{a_3(2) - a_3(3)\,a_3(1)}{1-K_3^2} = \dfrac{1.67 - 0.683(1.5291)}{1-(0.683)^2} = \underline{1.173}$

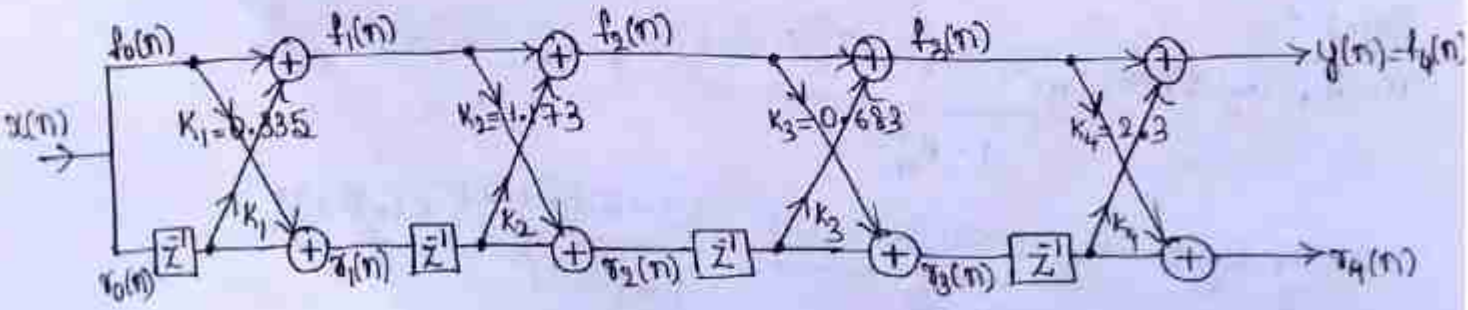
Step 3 :-

$m=2$,

$$K_2 = a_2(2) = \underline{1.173}$$

$$a_1(i) = \frac{a_2(i) - a_2(2)\,a_2(2-i)}{1-K_2^2} \quad ; 1 \le i \le 1$$

$i=1$, $a_1(1) = \dfrac{a_2(1) - a_2(2)\,a_2(1)}{1-K_2^2} = \dfrac{0.7282 - 1.173(0.7282)}{1-(1.173)^2} = \underline{0.335} = K_1$



Scanned with CamScanner

6) $H(z) = 1+0.18z^{-1}+0.236z^{-2}+0.3z^{-3}$. Draw lattice structure.

Soln:

Given : $H(z) = 1+0.18z^{-1}+0.236z^{-2}+0.3z^{-3}$ ——→ (1)

$$H(z) = 1 + a_3(1)z^{-1} + a_3(2)z^{-2} + a_3(3)z^{-3} \longrightarrow (2)$$

Comparing eq(1) and eq(2),

$a_3(1) = 0.18$; $a_3(2) = 0.236$; $a_3(3) = 0.3$

To find lattice Co-efficients :

$$K_m = a_m(m)$$

$$a_{m-1}(i) = \frac{a_m(i) - a_m(m)\,a_m(m-i)}{1-K_m^2} \quad ; \; 1 \le i \le m-1$$

Since, there are 3 zeros in $H(z)$, Take $m=3$

Step 1 :- $m=3$;

$$K_3 = a_3(3) = 0.3$$

$$a_2(i) = \frac{a_3(i) - a_3(3)\,a_3(3-i)}{1-K_3^2}$$

$$i=1, \quad a_2(1) = \frac{a_3(1) - a_3(3)\,a_3(2)}{1-K_3^2} = \frac{0.18 - 0.3(0.236)}{1-(0.3)^2} = 0.12$$

$$i=2, \quad a_2(2) = \frac{a_3(2) - a_3(3)\,a_3(1)}{1-K_3^2} = \frac{0.236 - 0.3(0.18)}{1-(0.3)^2} = 0.2$$

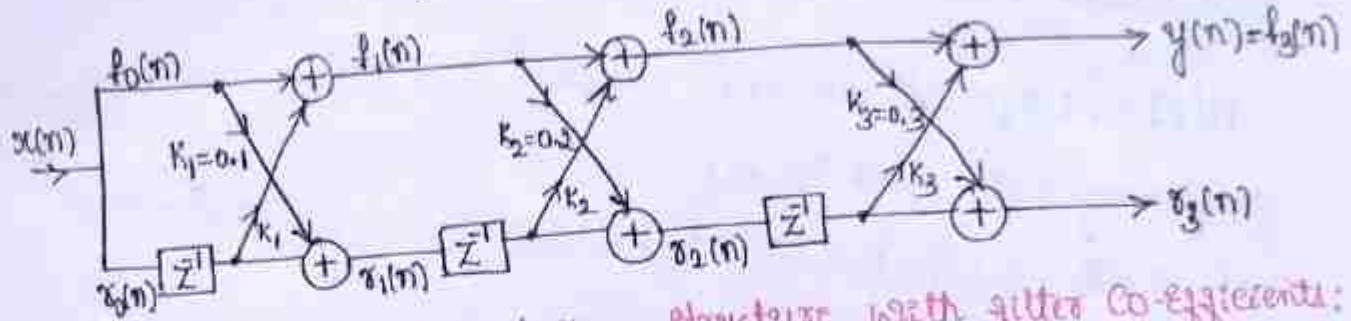
Step 2 :- $m=2$;

$$K_2 = a_2(2) = 0.2$$

$$a_1(i) = \frac{a_2(i) - a_2(2)\,a_2(2-i)}{1-K_2^2} \quad ; \; 1 \le i \le 1$$

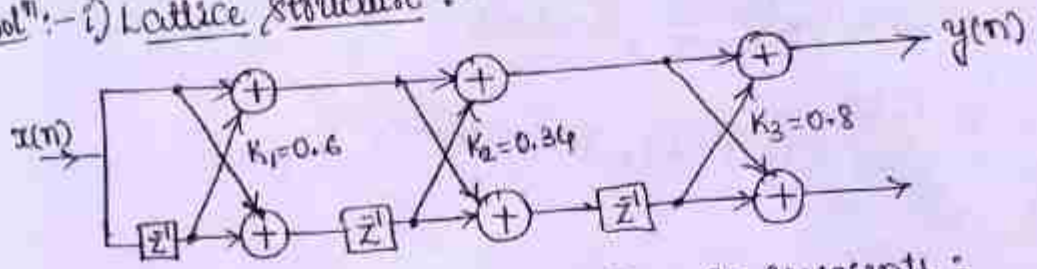
Scanned with CamScanner

$i=1,\ a_1(1) = \dfrac{a_2(1) - a_2(2)\,a_2(1)}{1-K_2^2} = \dfrac{0.12 - 0.2(0.12)}{1-(0.2)^2} = 0.1 = K_1$



7) Consider, a 3-stage FIR lattice structure with filter Co-efficients: $K_1=0.1$, $K_2=0.34$, $K_3=0.8$. Draw i) lattice structure ii) Direct-form structure.

Sol^n:- i) Lattice Structure :-



ii) To find Co-efficients of filter Co-efficients :

$K_1 = 0.1 = a_1(1)$; $K_2 = 0.34 = a_2(2)$; $K_3 = 0.8 = a_3(3)$

$$H(z) = 1 + \sum_{i=1}^{M} a_m(i)\, z^{-i} = 1 + \sum_{i=1}^{3} a_3(i)\, z^{-i}$$

$$H(z) = 1 + a_3(1)z^{-1} + a_3(2)z^{-2} + a_3(3)z^{-3} \longrightarrow (1)$$

To find $a_3(1)$ and $a_3(2)$:-

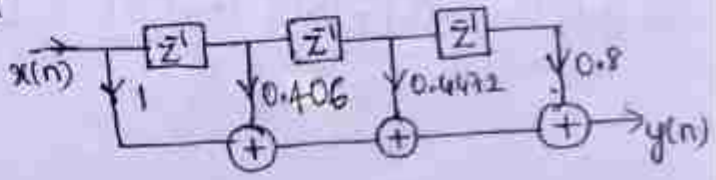
$$a_m(i) = a_{m-1}(i) + a_m(m)\,a_{m-1}(m-i) \;;\; 1 \le i \le m-1$$

$m=2,\ i=1$; $a_2(1) = a_1(1) + a_2(2)\,a_1(1) = 0.1 + 0.34(0.1) = 0.134$

$m=3,\ i=1$; $a_3(1) = a_2(1) + a_3(3)\,a_2(2) = 0.134 + 0.8^{\times 0.34} = 0.406$

$i=2$; $a_3(2) = a_2(2) + a_3(3)\,a_2(1) = 0.34 + 0.8(0.134) = 0.4472$

$\therefore H(z) = 1 + 0.406z^{-1} + 0.4472z^{-2} + 0.8z^{-3}$



Scanned with CamScanner

Problems on Lattice to direct form-I structure :-

1) Consider an FIR lattice filter with Co-efficients $K_1=0.65$, $K_2=-0.34$, $K_3=0.8$. Find its impulse response. Draw the equivalent direct-form structure.

Sol^n :- To find Co-efficients of filter Co-efficients :-

$K_1 = a_1(1) = 0.65$; $K_2 = a_2(2) = -0.34$, $K_3 = a_3(3) = 0.8$

$$H(z) = 1 + \sum_{i=1}^{M} a_m(i) z^{-i} = 1 + \sum_{i=1}^{3} a_3(i) z^{-i}$$

$$H(z) = 1 + a_3(1) z^{-1} + a_3(2) z^{-2} + a_3(3) z^{-3} \longrightarrow (1)$$

The Conversion formulae for direct form from lattice structure is,

$m = 2, 3, \ldots, M-1, M$; $m = 2, 3$

$$a_m(i) = a_{m-1}(i) + a_m(m)\, a_{m-1}(m-i) \; ; \; 1 \le i \le m-1$$

Step 1 :- When $m=2$,

$$a_2(i) = a_1(i) + a_2(2)\, a_1(2-i) \; ; \; 1 \le i \le 1$$

@ $i=1$, $a_2(1) = a_1(1) + a_2(2)\, a_1(1) = 0.65 - 0.34(0.65) = 0.429$

Step 2 :- When $m=3$,

$$a_3(i) = a_2(i) + a_3(3)\, a_2(3-i) \; ; \; 1 \le i \le 2$$

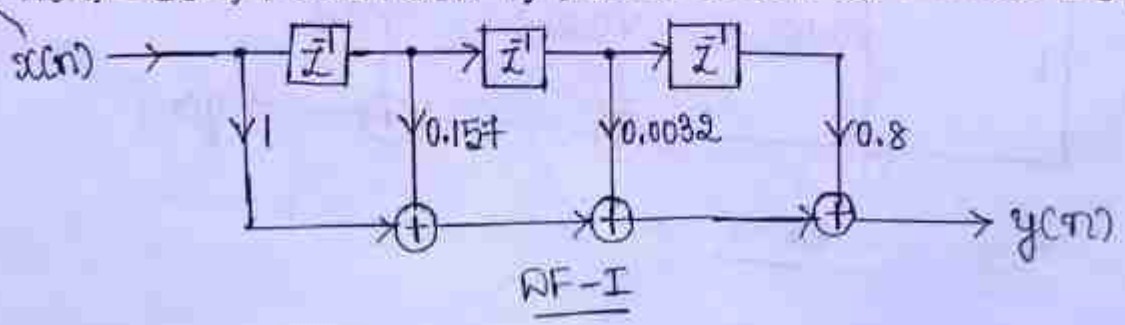
@ $i=1$, $a_3(1) = a_2(1) + a_3(3) a_2(2) = 0.429 + 0.8(-0.34) = 0.157$

@ $i=2$, $a_3(2) = a_2(2) + a_3(3) a_2(1) = -0.34 + 0.8(0.429) = 0.0032$

$\therefore H(z) = 1 + 0.157 z^{-1} + 0.0032 z^{-2} + 0.8 z^{-3}$

Applying I-ZT on both sides,

$$h(n) = \delta(n) + 0.157\,\delta(n-1) + 0.0032\,\delta(n-2) + 0.8\,\delta(n-3)$$



DF-I

Scanned with CamScanner

2) Let the Co-efficients of a three-stage FIR lattice structure be $K_1 = 0.1$, $K_2 = 0.2$, $K_3 = 0.3$. Find the Co-efficient of direct form-I FIR filter and draw its block diagram.

Sol^n:- $K_1 = a_1(1) = 0.1$; $K_2 = a_2(2) = 0.2$; $K_3 = a_3(3) = 0.3$

$$H(z) = 1 + \sum_{i=1}^{m} a_m(i) z^{-i} \quad ; \text{Here } m = 3$$

$$= 1 + \sum_{i=1}^{3} a_3(i) z^{-i}$$

$$H(z) = 1 + a_3(1) z^{-1} + a_3(2) z^{-2} + a_3(3) z^{-3} \longrightarrow (1)$$

The Conversion formulae for direct form from lattice structure is,

$m = 2, 3, \ldots M-1, M$; $m = 2, 3$

$$a_m(i) = a_{m-1}(i) + a_m(m)\, a_{m-1}(m-i) \quad ; 1 \le i \le (m-1)$$

Step 1 :- when $m = 2$,

$$a_2(i) = a_1(i) + a_2(2)\, a_1(2-i) \quad ; 1 \le i \le 1$$

@ $i = 1$, $a_2(1) = a_1(1) + a_2(2)\, a_1(1) = 0.1 + 0.2(0.1) = \underline{0.12}$

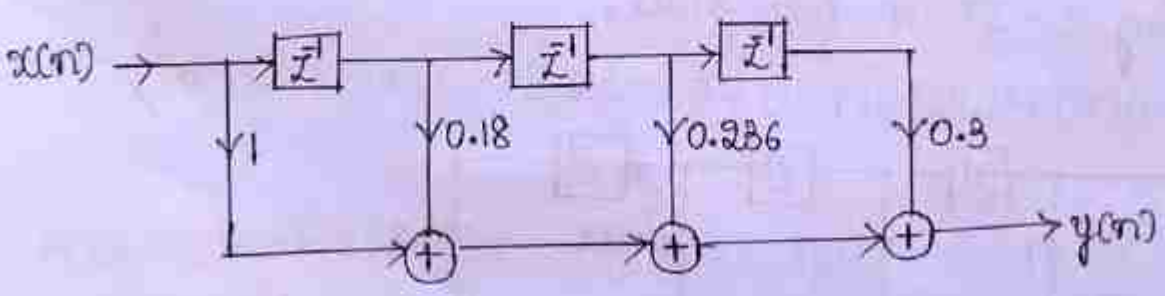
Step 2 :- when $m = 3$,

$$a_3(i) = a_2(i) + a_3(3)\, a_2(3-i) \quad ; 1 \le i \le 2$$

@ $i = 1$, $a_3(1) = a_2(1) + a_3(3)\, a_2(2) = 0.12 + 0.3(0.2) = \underline{0.18}$

@ $i = 2$, $a_3(2) = a_2(2) + a_3(3)\, a_2(1) = 0.2 + 0.3(0.12) = \underline{0.236}$

$\therefore H(z) = 1 + 0.18 z^{-1} + 0.236 z^{-2} + 0.3 z^{-3}$



DF-I

Scanned with CamScanner

3) Consider an FIR lattice filter with Co-efficients $K_1 = 0.2$, $K_2 = 0.3$, $K_3 = 0.4$, $K_4 = 0.5$. Find its impulse response. Draw the equivalent direct form structure.

Sol^n :- $K_1 = a_1(1) = 0.2$; $K_2 = a_2(2) = 0.3$; $K_3 = a_3(3) = 0.4$; $K_4 = a_4(4) = 0.5$

$$H(z) = 1 + \sum_{i=1}^{M} a_m(i)\, z^{-i} = 1 + \sum_{i=1}^{4} a_4(i)\, z^{-i}$$

$$H(z) = 1 + a_4(1) z^{-1} + a_4(2) z^{-2} + a_4(3) z^{-3} + a_4(4) z^{-4} \longrightarrow (1)$$

The Conversion formulae for direct form from lattice structure is,

$m = 2, 3, \ldots, M-1, M$; $m = 2, 3, 4$

$$a_m(i) = a_{m-1}(i) + a_m(m)\, a_{m-1}(m-i) \; ; \; 1 \le i \le m-1$$

Step 1 :- When $m = 2$,

$$a_2(i) = a_1(i) + a_2(2)\, a_1(2-i) \; ; \; 1 \le i \le 1$$

@ $i = 1$, $a_2(1) = a_1(1) + a_2(2) a_1(1) = 0.2 + 0.3(0.2) = \underline{0.26}$

Step 2 :- When $m = 3$,

$$a_3(i) = a_2(i) + a_3(3)\, a_2(3-i) \; ; \; 1 \le i \le 2$$

@ $i = 1$, $a_3(1) = a_2(1) + a_3(3) a_2(2) = 0.26 + 0.4(0.3) = \underline{0.38}$

@ $i = 2$, $a_3(2) = a_2(2) + a_3(3) a_2(1) = 0.3 + 0.4(0.26) = \underline{0.404}$

Step 3 :- When $m = 4$,

$$a_4(i) = a_3(i) + a_4(4)\, a_3(4-i) \; ; \; 1 \le i \le 3$$

@ $i = 1$, $a_4(1) = a_3(1) + a_4(4) a_3(3) = 0.38 + 0.5(0.4) = \underline{0.58}$

@ $i = 2$, $a_4(2) = a_3(2) + a_4(4) a_3(2) = 0.404 + 0.5(0.404) = \underline{0.606}$

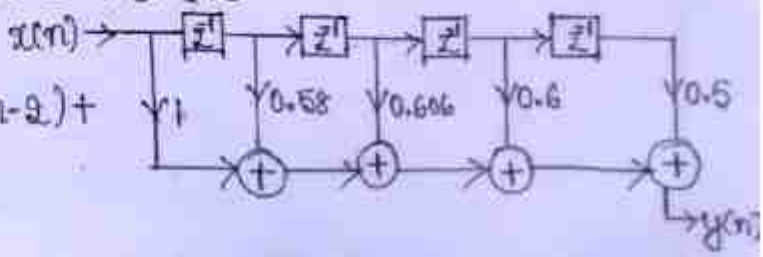
@ $i = 3$, $a_4(3) = a_3(3) + a_4(4) a_3(1) = 0.4 + 0.5(0.38) = \underline{0.59}$

∴ Eq(1) becomes,

$$H(z) = 1 + 0.58 z^{-1} + 0.606 z^{-2} + 0.6 z^{-3} + 0.5 z^{-4}$$

By I-ZT,

$$h(n) = \delta(n) + 0.58\delta(n-1) + 0.606\,\delta(n-2) + 0.6\delta(n-3) + 0.5\,\delta(n-4)$$



Scanned with CamScanner

Problems on frequency-Sampling Structure :-

1) Sketch the frequency-Sampling Structure ($\alpha=0$) for linear phase (Symmetric) FIR filter which has frequency samples.

$$H\left(\frac{2\pi}{32}K\right) = 1 \; ; \; K=0,1,2$$
$$= \frac{1}{2} \; ; \; K=3$$
$$= 0 \; ; \; K=4,5,\ldots 15$$

Sol^n :- Given $M=32$ (Even) and $\alpha=0$

The System function,

$$H(z) = \frac{1-e^{j2\pi\alpha}z^{-M}}{M}\left[\frac{H(0)}{1-z^{-1}} + \frac{H\left(\frac{M}{2}\right)}{1+z^{-1}} + \sum_{K=1}^{\frac{M}{2}-1}\frac{A(K)+B(K)z^{-1}}{1-2\cos\left(\frac{2\pi K}{M}\right)z^{-1}+z^{-2}}\right]$$

$$= \frac{1-z^{-M}}{M}\left[\frac{H(0)}{1-z^{-1}} + \frac{H\left(\frac{M}{2}\right)}{1+z^{-1}} + \sum_{K=1}^{\frac{M}{2}-1}\frac{A(K)+B(K)z^{-1}}{1-2\cos\left(\frac{2\pi K}{M}\right)z^{-1}+z^{-2}}\right]$$

where $A(K) = H(K) + H(M-K)$

and $B(K) = H(K)e^{-j\frac{2\pi K}{M}} + H(M-K)e^{j\frac{2\pi K}{M}}$

Due to Symmetry,

$H(0) = 1$

$H(1) = H(31) = 1 \; ; \; A(1) = 2 \; ; \; B(1) = 2\cos\left(\frac{\pi}{16}\right)$

$H(2) = H(30) = 1 \; ; \; A(2) = 2 \; ; \; B(2) = 2\cos\left(\frac{\pi}{8}\right)$

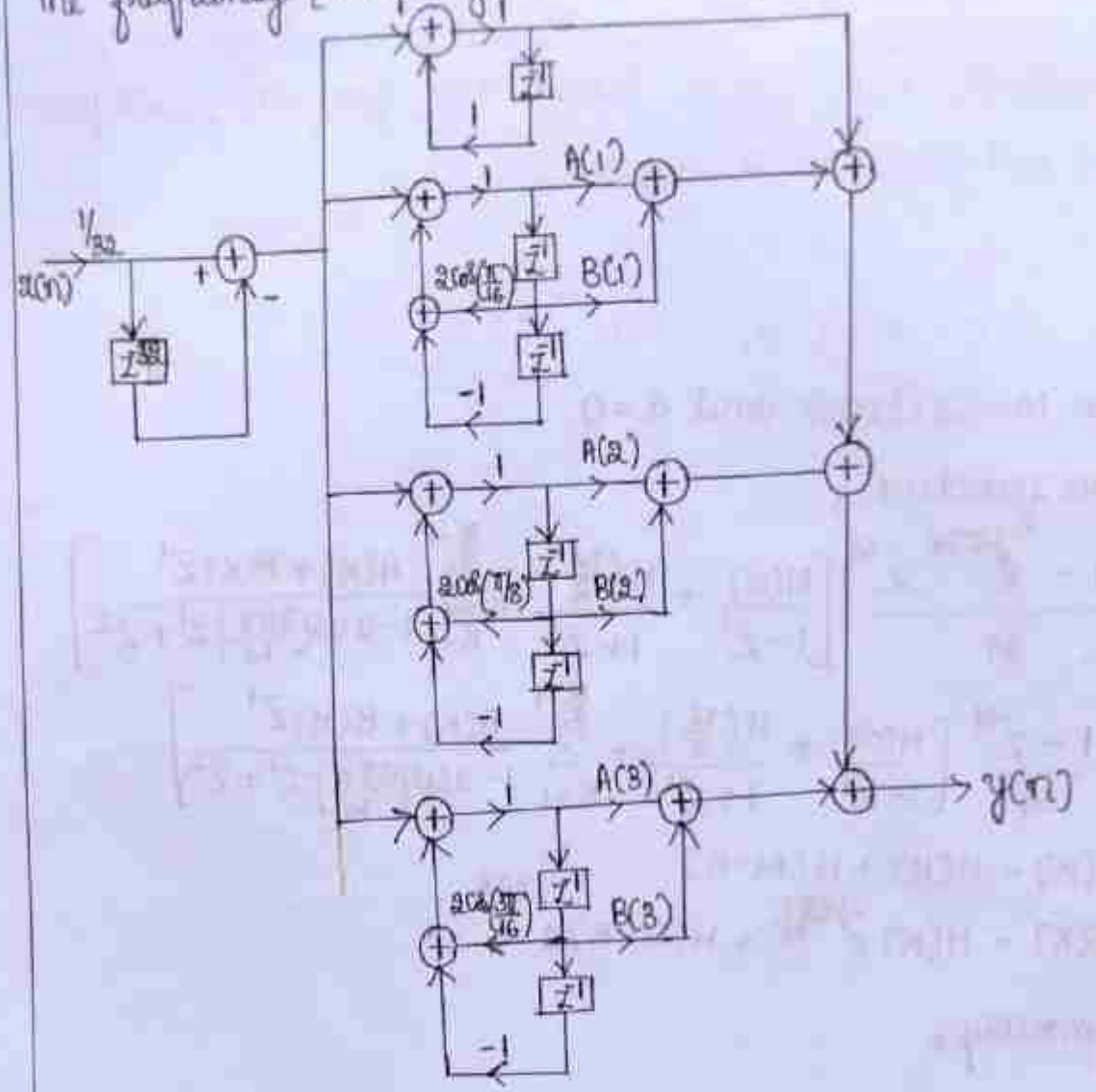
$H(3) = H(29) = \frac{1}{2} \; ; \; A(3) = 1 \; ; \; B(3) = \cos\left(\frac{3\pi}{16}\right)$

and all other $H(K)$'s are equal to zero.

$$\therefore H(z) = \frac{1-z^{-32}}{32}\left[\frac{1}{1-z^{-1}} + \frac{[A(1)+B(1)z^{-1}]}{1-2\cos\left(\frac{\pi}{16}\right)+z^{-2}} + \frac{[A(2)+B(2)z^{-1}]}{1-2\cos\left(\frac{\pi}{8}\right)+z^{-2}} + \frac{[A(3)+B(3)z^{-1}]}{1-2\cos\left(\frac{3\pi}{16}\right)+z^{-2}}\right]$$

Scanned with CamScanner

The frequency sampling structure is shown below :



Scanned with CamScanner

Conversion from lattice to direct form-I :—

for $m = 1, 2, \ldots M$ do

$a_m(0) = 1$

$a_m(m) = K_m$

$$a_m(i) = a_{m-1}(i) + a_m(m)\, a_{m-1}(m-i) \; ; \; 1 \le i \le m-1$$

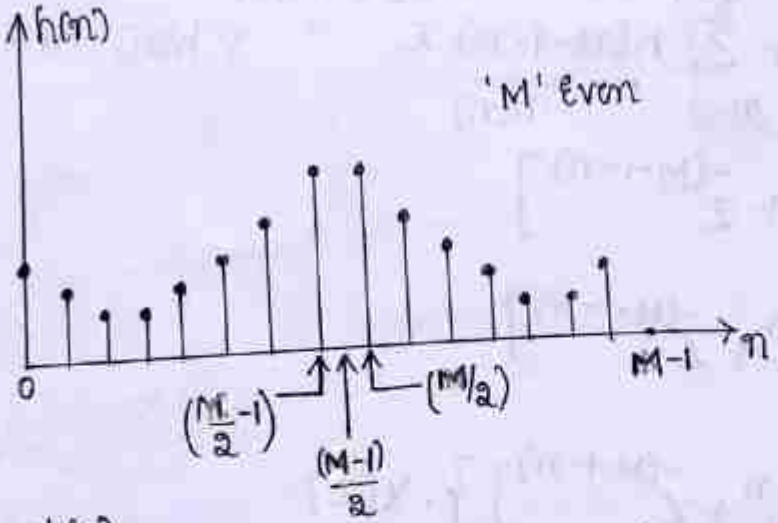
4) Realization of Linear-phase FIR filters :—

If $h(n)$ represents the impulse response of a discrete-time system, a necessary and sufficient condition for linear phase is that $h(n)$ have finite duration 'N' and that it be symmetric about its midpoint.

For causal FIR filter whose impulse response begins at zero and ends at N-1, $h(n)$ must satisfy :

$$h(n) = h(M-1-n) \text{ for } 0 \le n \le (M-1) \longrightarrow (1)$$

For this condition, the shapes of $h(n)$ that gives linear phase are as shown in fig for 'M' even and odd.



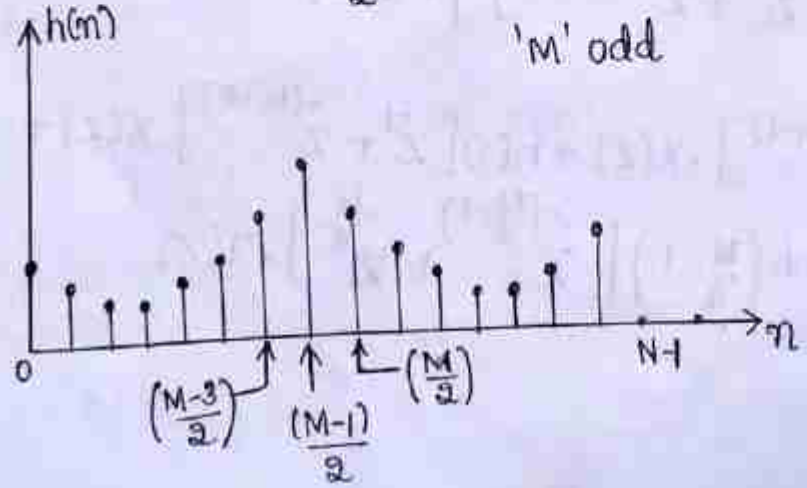


fig : General shapes of impulse response h(n) that gives linear phase for 'M' Even and odd

Scanned with CamScanner

For a causal filter whose h(n) begins at zero and ends at (M-1) this symmetry was expressed for all $n = 0, 1, 2, \ldots (M-1)$ as follows:

$$h(n) = h(M-1-n) \longrightarrow (2) \text{ for } 0 \le n \le (M-1)$$

where, 'M' indicates number of samples.

Case(I): For 'M' Even :-

The transfer function of FIR filter is given by,

$$H(z) = \sum_{n=0}^{M-1} h(n) z^{-n}$$

$$= \sum_{n=0}^{\frac{M}{2}-1} h(n) z^{-n} + \sum_{n=\frac{M}{2}}^{M-1} h(n) z^{-n}$$

For linear phase, $h(n) = h(M-1-n)$

put $m = M-1-n$

when $n = \frac{M}{2}$, $m = M-1-\frac{M}{2} = \frac{M}{2}-1$

and $n = M-1$, $m = N-1-N+1 = 0$

$$\therefore H(z) = \sum_{n=0}^{\frac{M}{2}-1} h(n) \cdot z^{-n} + \sum_{n=0}^{\frac{M}{2}-1} \underbrace{h(M-1-n)}_{h(n)} z^{-(M-1-n)} \quad \because h(n) = h(M-1-n)$$

$$H(z) = \sum_{n=0}^{\frac{M}{2}-1} h(n) \left[z^{-n} + z^{-(M-1-n)} \right]$$

$$\frac{Y(z)}{X(z)} = \sum_{n=0}^{\frac{M}{2}-1} h(n) \left[z^{-n} + z^{-(M-1-n)} \right]$$

$$Y(z) = \left\{ \sum_{n=0}^{\frac{M}{2}-1} h(n) \left[z^{-n} + z^{-(M-1-n)} \right] \right\} \cdot X(z)$$

$$= h(0)\left[1 + z^{-(M-1)}\right] \cdot X(z) + h(1)\left[z^{-1} + z^{-(M-2)}\right] X(z) + \ldots$$

$$\ldots\ldots + h\left(\frac{M}{2}-1\right)\left[z^{-\left(\frac{M}{2}-1\right)} + z^{-\frac{M}{2}} \right] \cdot X(z)$$

Scanned with CamScanner

$$Y(z) = h(0)\left[X(z) + X(z).z^{-(M-1)}\right] + h(1)\left[z^{-1}X(z) + z^{-(M-2)}X(z)\right] + \cdots$$
$$\cdots\cdots + h\left(\frac{M}{2}-1\right)\left[z^{-\left(\frac{M}{2}-1\right)}.X(z) + z^{-\frac{M}{2}}.X(z)\right]$$

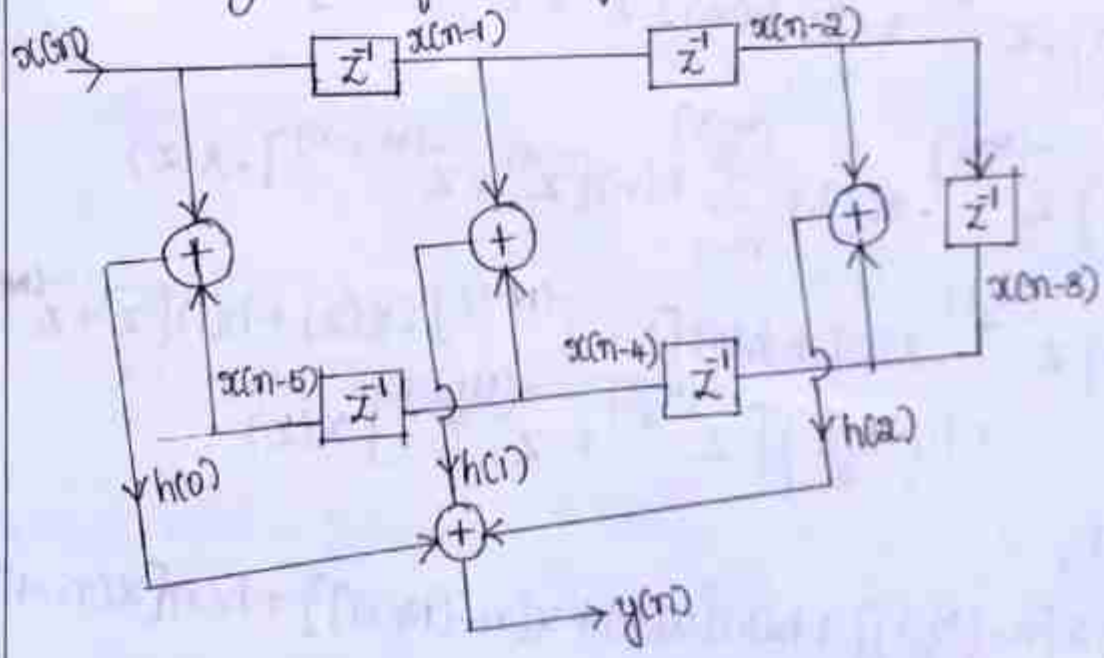
Taking Inverse Z-transform on both sides,

$$y(n) = h(0)\left[x(n) + x(n-M+1)\right] + h(1)\left[x(n-1) + x(n-M+2)\right] + \cdots$$
$$\cdots\cdots + h\left(\frac{M}{2}-1\right)\left[x\left(n-\frac{M}{2}+1\right) + x\left(n-\frac{M}{2}\right)\right] \longrightarrow (8)$$

Let $M=6$,

$$y(n) = h(0)\left[x(n) + x(n-5)\right] + h(1)\left[x(n-1) + x(n-4)\right] + h(2)\left[x(n-2) + x(n-3)\right]$$

The realization of this equation is shown in fig below:



ii) For 'M' odd :-

$$H(z) = \sum_{n=0}^{M-1} h(n)z^{-n}$$
$$= h\left(\frac{M-1}{2}\right)z^{-\left(\frac{M-1}{2}\right)} + \sum_{n=0}^{\left(\frac{M-3}{2}\right)} h(n)z^{-n} + \sum_{n=\frac{M+1}{2}}^{(M-1)} h(n)z^{-n}$$

For a linear phase, $h(n) = h(M-1-n)$

$$= h\left(\frac{M-1}{2}\right)z^{-\left(\frac{M-1}{2}\right)} + \sum_{n=0}^{\left(\frac{M-3}{2}\right)} h(n)z^{-n} + \sum_{n=\frac{M+1}{2}}^{(M-1)} h(M-1-n)z^{-(M-1-n)}$$

Scanned with CamScanner

Let $m = M-1-n$,

When $n = \frac{M+1}{2}$, $m = M-1-\left(\frac{M+1}{2}\right) = \frac{M}{2} - \frac{3}{2} = \left(\frac{M-3}{2}\right)$

and $n = M-1$, $m = M-1-(M-1) = 0$

$$\therefore H(z) = h\left(\frac{M-1}{2}\right) z^{-\left(\frac{M-1}{2}\right)} + \sum_{n=0}^{\left(\frac{M-3}{2}\right)} h(n) z^{-n} + \sum_{n=0}^{\left(\frac{M-3}{2}\right)} \underbrace{h(M-1-n)}_{h(n)} z^{-(M-1-n)}$$

But from Symmetry property, $h(n) = h(M-1-n)$

$$H(z) = h\left(\frac{M-1}{2}\right) z^{-\left(\frac{M-1}{2}\right)} + \sum_{n=0}^{\left(\frac{M-3}{2}\right)} h(n)\left[z^{-n} + z^{-(M-1-n)}\right]$$

Here $\left(\frac{M-1}{2}\right)^{th}$ term written Separately since 'M' is odd.

$$\frac{Y(z)}{X(z)} = h\left(\frac{M-1}{2}\right) \cdot z^{-\left(\frac{M-1}{2}\right)} + \sum_{n=0}^{\left(\frac{M-3}{2}\right)} h(n)\left[z^{-n} + z^{-(M-1-n)}\right]$$

$$Y(z) = h\left(\frac{M-1}{2}\right) z^{-\left(\frac{M-1}{2}\right)} \cdot X(z) + \sum_{n=0}^{\left(\frac{M-3}{2}\right)} h(n)\left[z^{-n} + z^{-(M-1-n)}\right] \cdot X(z)$$

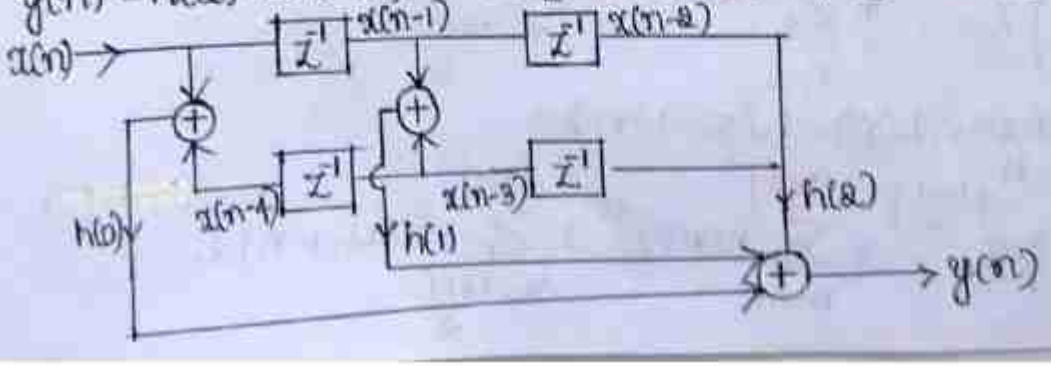
$$Y(z) = h\left(\frac{M-1}{2}\right) z^{-\left(\frac{M-1}{2}\right)} \cdot X(z) + h(0)\left[1 + z^{-(M-1)}\right] \cdot X(z) + h(1)\left[z^{-1} + z^{-(M-2)}\right] X(z)$$
$$+ \ldots\ldots + h\left(\frac{M-3}{2}\right)\left[z^{-\left(\frac{M-3}{2}\right)} + z^{-\left(\frac{M+1}{2}\right)}\right] X(z)$$

Taking I-ZT,

$$y(n) = h\left(\frac{M-1}{2}\right) x\left[n-\left(\frac{M-1}{2}\right)\right] + h(0)\{x(n) + x[n-(M-1)]\} + h(1)[x(n-1) + x[n-(M-2)]\} + \ldots\ldots + h\left(\frac{M-3}{2}\right)\left\{x\left[n-\left(\frac{M-3}{2}\right)\right] + x\left[n-\left(\frac{M+1}{2}\right)\right]\right\}$$

Let $M = 5$,

$$y(n) = h(2)\,x(n-2) + h(0)[x(n) + x(n-4)] + h(1)[x(n-1) + x(n-3)]$$

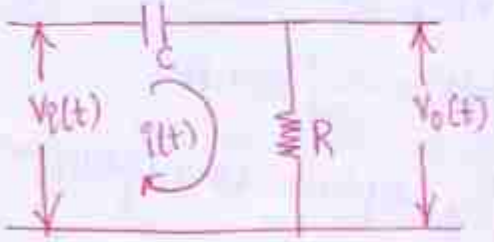


Scanned with CamScanner

$$\therefore H(z) = \frac{1}{2.5413\left(\frac{1-z^{-1}}{1+z^{-1}}\right)+1}$$

$$\langle H(z) = \frac{1+z^{-1}}{3.5413-1.5413z^{-1}} \rangle$$

18) Design a digital highpass filter for the following specification: $R=1\Omega$, $C=1F$ and $f_s=8f_c$.



Sol^n:

$$V_i(t) = R\,i(t) + \frac{1}{C}\int i(t)dt \qquad V_o(t) = R\,i(t)$$

By LT,

$$V_I(S) = RI(S) + \frac{1}{CS}I(S)$$

By LT,

$$V_o(S) = R\,I(S) \longrightarrow (2)$$

$$V_I(S) = I(S)\left[\frac{RCS+1}{CS}\right] \longrightarrow (1)$$

W.K.T. $$H(S) = \frac{V_o(S)}{V_I(S)} = \frac{R\,I(S)}{\left(\frac{RCS+1}{CS}\right)I(S)} = \frac{1}{\left(\frac{S+1}{S}\right)} = \frac{S}{S+1}$$

Given: $$f_s = 8f_c = 8\cdot\frac{1}{2\pi RC} = \frac{8}{2\pi} = 1.27$$

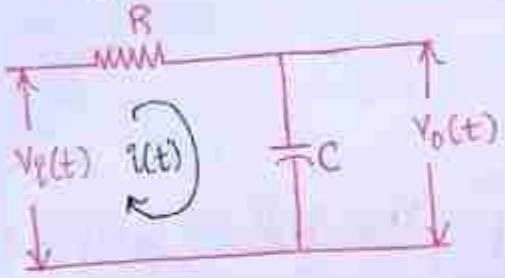
$$T = \frac{1}{f_s} = \frac{1}{1.27} = 0.787 \text{ Sec}$$

$$H(z) = H(S)\Big|_{S\to\frac{2}{T}\left(\frac{1-z^{-1}}{1+z^{-1}}\right)} \to 2.54\left(\frac{1-z^{-1}}{1+z^{-1}}\right)$$

Scanned with CamScanner

17) Design a digital LPF for the following Specifications:

$R = 1\Omega$, $C = 1F$ & $f_s = 8f_c$



Sol^n:-

Apply KVL to the i/p side,

$$V_i(t) = Ri(t) + \frac{1}{C}\int i(t)\,dt$$

By Laplace transform,

$$V_I(s) = RI(s) + \frac{1}{Cs} I(s)$$

$$= I(s)\left\{R + \frac{1}{Cs}\right\}$$

$$V_I(s) = I(s)\left\{\frac{RCs+1}{Cs}\right\} \longrightarrow (1)$$

Apply KVL to the o/p side,

$$V_o(t) = \frac{1}{C}\int i(t)\,dt$$

By Laplace transform,

$$V_o(s) = \frac{1}{Cs} I(s) \longrightarrow (2)$$

W.K.T. $$H(s) = \frac{V_o(s)}{V_I(s)} = \frac{\frac{1}{Cs} I(s)}{\left[\frac{RCs+1}{Cs}\right] I(s)} = \frac{1}{RCs+1}$$

$$\therefore H(s) = \frac{1}{s+1} \quad [\because R = 1\Omega, C = 1F]$$

Given: $$f_s = 8f_c = 8 \cdot \frac{1}{2\pi RC} = \frac{8}{2\pi} = 1.27\,Hz$$

$$T = \frac{1}{f_s} = \frac{1}{1.27} = 0.787\,sec$$

$$H(z) = H_a(s)\Big|_{s \to \frac{2}{T}\left(\frac{1-z^{-1}}{1+z^{-1}}\right)} \to \frac{2}{0.787}\left(\frac{1-z^{-1}}{1+z^{-1}}\right) \to 2.5413\left(\frac{1-z^{-1}}{1+z^{-1}}\right)$$

Scanned with CamScanner

MJ

Explain i) DIT-FFT, ii) DIF-FFT.

Develope DIT-FFT for N=9 or N=6.

Develope bit reversal property used in FFT algorithm for N=16.

Discuss memory requirement & inplace computation related to DIT & DTF-FFT.

what are the difference & similarities b/w DIF-FFT & DIT-FFT algorithm.

what is the need of FFT.

Unit-5 :- Eto

Compare butterworth & Chebyshev filters.

Distinguish b/w IIR & FIR filters.

Derive an expression for order of a lowpass butterworth filter. (& cutoff frequency)

Derive an expression for poles of a low pass butterworth filter.

Unit-7 :- (10M) Each

what is bilinear transformation. Derive the expression for the bilinear transformation to transform a analog filter to a digital filter by Trapezoidal rule & explain the maping from s-plane to z-plane.

Derive the S to z-plane transformation based on finite backward difference method. Also show that the entire left half of S-plan poles are mapped inside the smaller circle of radius 1/2 centered at z = 1/2 inside the unit ole in the z-plane.

Scanned with CamScanner

DSP Theory Questions

unit-1:-

Prove that the sampling of DTFT of a sequence x(n) results in N-point DFT.

What is zero padding. What are its uses

Define DFT. Establish a relationship b/w
a) Fourier series co-efficient of a continuous time signal & DFT.
b) Z-Transform & DFT.
c) DFT & DTFT.

State & Prove the following properties.
a) Symmetric property
b) Periodicity property of a twidal factor.

unit-3:-

1) What is sectioned convolution. Explain.
   i) Overlap add method
   ii) Overlap Save method
2) Difference b/w linear & circular convolution.

unit-4:-

1) Write a short note on chirp-z transform.
2) Explain inplace computation of FFT.
3) What is Goertzel algorithm? Obtain direct form-II realisation of two pole resonator for computing DFT.
4) What is Chirp signal?

Scanned with CamScanner

Unit -7

Explain how an analog filter is mapped on to a digital filter using impulse invariance method. what are the limitations of the method.

Unit - 8 :-

Realize FIR linear phase filter for 'N' to be even or 'N' to be odd.

Scanned with CamScanner

An FIR filter has the impulse response of $h(n) = \{1, 2, 3\}$. Determine the response of the filter to the input sequence $x(n) = \{1, 2\}$. Use DFT and IDFT and Verify the result using direct computation of linear convolution.

Sol^n:-

Computation of $y(n)$ using DFT and IDFT :-

Here $M = 3$, $L = 2$

$N = M + L - 1 = 3 + 2 - 1 = 4$

$\therefore x(n) = \{1, 2, \boxed{0, 0}\}$ and $h(n) = \{1, 2, 3, \boxed{0}\}$

($M-1$) zeros ($L-1$) zeros

W.K.T $X_4 = W_4 x_4$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ 1-j2 \\ -1 \\ 1+j2 \end{bmatrix}$$

$$\langle X(K) = \{3, 1-j2, -1, 1+j2\} \rangle$$

W.K.T $H_4 = W_4 h_4$

$$\begin{bmatrix} H(0) \\ H(1) \\ H(2) \\ H(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} h(0) \\ h(1) \\ h(2) \\ h(3) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 0 \end{bmatrix} = \begin{bmatrix} 6 \\ -2-j2 \\ 2 \\ -2+j2 \end{bmatrix}$$

Scanned with CamScanner

$\langle H(K) = \{6, -2-j2, 2, -2+j2\} \rangle$

W.K.T $Y(K) = X(K)\, H(K)$

$\langle Y(K) = \{18, -6+j2, -2, -6-j2\} \rangle$

$$y_4 = \frac{1}{4} W_4^* Y_4$$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} W_4^{-0} & W_4^{-0} & W_4^{-0} & W_4^{-0} \\ W_4^{-0} & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ W_4^{-0} & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ W_4^{-0} & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix} \begin{bmatrix} Y(0) \\ Y(1) \\ Y(2) \\ Y(3) \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 18 \\ -6+j2 \\ -2 \\ -6-j2 \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 18-6+j2-2-6-j2 \\ 18-j6-2+2+j6-2 \\ 18+6-j2-2+6+j2 \\ 18+j6+2+2-j6+2 \end{bmatrix}$$

$$= \frac{1}{4} \begin{bmatrix} 4 \\ 16 \\ 28 \\ 24 \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 7 \\ 6 \end{bmatrix}$$

$\langle y(n) = \{1, 4, 7, 6\} \rangle$

Direct Computation of $y(n)$ using linear Convolution :-

$h(n) \rightarrow$ 1 2 3

$x(n) \rightarrow$ 1 2

2 4 6

1 2 3

$y(n) \rightarrow$ 1 4 7 6

(or) $x(n)$ ↓ \ $h(n)$ →	1	2	3
1	1	2	3
2	2	4	6

$\therefore y(n) = \{1, 4, 7, 6\}$

Scanned with CamScanner

Determine the response of the FIR filter whose unit sample response is given as, $h(n) = \{1, 2\}$ When input applied is $x(n) = \{2, 1\}$. Use Circular Convolution and Verify your result using linear Convolution.

Sol^n:- Here $M = 2$ and $L = 2$

$$N = L + M - 1$$
$$= 2 + 2 - 1$$

$\langle N = 3 \rangle$ output has 3 samples, we should perform DFT (or) Circular Convolution for atleast '3' points (or) more. Hence, take $L = 4$ points.

$h(n) = \{1, 2, 0, 0\}$ and $x(n) = \{2, 1, 0, 0\}$

$$y(n) = x(n) \circledast_4 h(n) = \begin{bmatrix} 2 & 0 & 0 & 1 \\ 1 & 2 & 0 & 0 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 2 \\ 0 \end{bmatrix}$$

$\langle \therefore y(n) = \{2, 5, 2, 0\} \rangle$

Verification using Linear Convolution :-

$h(n) \rightarrow 1 \quad 2$

$x(n) \rightarrow 2 \quad 1$

$1 \quad 2$

$2 \quad 4 \quad +$

$y(n) \rightarrow 2 \quad 5 \quad 2$

$\langle y(n) = \{2, 5, 2\} \rangle$

3) The impulse response of a system is given by $h(n) = \{3, 2\}$. Find the output of the system for an input $x(n) = \{0, 1, 0, 1\}$ using circular Convolution.

Sol^n:- Here $x(n) = \{0, 1, 0, 1\}$ i.e., $L = 4$

$h(n) = \{3, 2\}$ i.e., $M = 2$

Scanned with CamScanner

The length of $y(n)$ will be $N = L+M-1$

$\therefore x(n) = \{0, 1, 0, 1, \underset{\uparrow}{0}\}$ ($M-1$ zeros) and $h(n) = \{3, 2, \underset{\uparrow}{0, 0, 0}\}$ ($L-1$ zeros)

$N = 4+2-1 = 5$

$y(n) = x(n) \circledast_5 h(n)$

$$= \begin{bmatrix} 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 3 \\ 2 \\ 3 \\ 2 \end{bmatrix}$$

$\therefore y(n) = \{0, 3, 2, 3, 2\}$

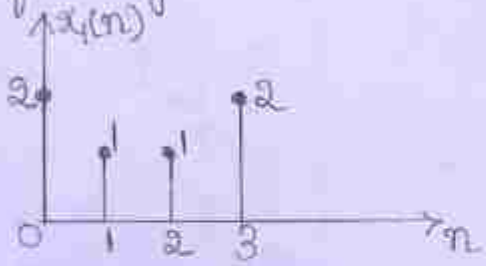
Given the sequences $x_1(n)$ and $x_2(n)$:

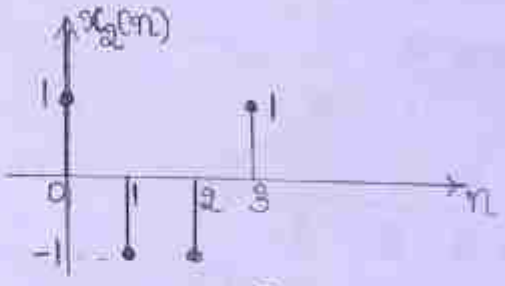
a) Compute the circular convolution, $x_1(n) \textcircled{N} x_2(n)$ for $N = 4, 5, 6, 7, 8$.

b) Compute the linear convolution, $x_1(n) * x_2(n)$.

c) What value of 'N' is necessary so that linear and circular convolution are the same on the N-point interval.

d) Determine the non-zero lengths of $x_1(n)$ and $x_2(n)$ and how the results in part(iii) could be obtained without performing calculations?





Sol^n:- $x_1(n) = \{\underset{\uparrow}{2}, 1, 1, 2\}$ and $x_2(n) = \{\underset{\uparrow}{1}, -1, -1, 1\}$

a) Computation of Circular Convolution:—

For $N = 4$, $y(n) = x_1(n) \circledast_4 x_2(n)$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} 2 & 2 & 1 & 1 \\ 1 & 2 & 2 & 1 \\ 1 & 1 & 2 & 2 \\ 2 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2-2-1+1 \\ 1-2-2+1 \\ 1-1-2+2 \\ 2-1-1+2 \end{bmatrix} = \begin{bmatrix} 0 \\ -2 \\ 0 \\ 2 \end{bmatrix}$$

Scanned with CamScanner

$\therefore y(n) = x_1(n) \circledast_4 x_2(n) = \{0, -2, 0, 2\}$

For $N=5$, $y(n) = x_1(n) \circledast_5 x_2(n)$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \end{bmatrix} = \begin{bmatrix} 2 & 0 & 2 & 1 & 1 \\ 1 & 2 & 0 & 2 & 1 \\ 1 & 1 & 2 & 0 & 2 \\ 2 & 1 & 1 & 2 & 0 \\ 0 & 2 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2+0-2+1+0 \\ 1-2+0+2+0 \\ 1-1-2+0+0 \\ 2-1-1+2+0 \\ 0-2-1+1+0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ -2 \\ 2 \\ -2 \end{bmatrix}$$

$\therefore y(n) = x_1(n) \circledast_5 x_2(n) = \{1, 1, -2, 2, -2\}$

For $N=6$, $y(n) = x_1(n) \circledast_6 x_2(n)$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \\ y(5) \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 2 & 1 & 1 \\ 1 & 2 & 0 & 0 & 2 & 1 \\ 1 & 1 & 2 & 0 & 0 & 2 \\ 2 & 1 & 1 & 2 & 0 & 0 \\ 0 & 2 & 1 & 1 & 2 & 0 \\ 0 & 0 & 2 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 4 \\ -1 \\ -2 \\ 2 \\ -2 \\ -1 \end{bmatrix}$$

$\therefore y(n) = x_1(n) \circledast_6 x_2(n) = \{4, -1, -2, 2, -2, -1\}$

For $N=7$, $y(n) = x_1(n) \circledast_7 x_2(n)$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \\ y(5) \\ y(6) \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 & 2 & 1 & 1 \\ 1 & 2 & 0 & 0 & 0 & 2 & 1 \\ 1 & 1 & 2 & 0 & 0 & 0 & 2 \\ 2 & 1 & 1 & 2 & 0 & 0 & 0 \\ 0 & 2 & 1 & 1 & 2 & 0 & 0 \\ 0 & 0 & 2 & 1 & 1 & 2 & 0 \\ 0 & 0 & 0 & 2 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \\ -2 \\ 2 \\ -2 \\ -1 \\ 2 \end{bmatrix}$$

$\therefore y(n) = x_1(n) \circledast_7 x_2(n) = \{2, -1, -2, 2, -2, -1, 2\}$

Scanned with CamScanner

For $N=8$, $y(n) = x_1(n) \circledast_8 x_2(n)$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \\ y(5) \\ y(6) \\ y(7) \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 2 & 1 & 1 \\ 1 & 2 & 0 & 0 & 0 & 0 & 2 & 1 \\ 1 & 1 & 2 & 0 & 0 & 0 & 0 & 2 \\ 2 & 1 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \\ -2 \\ 2 \\ -2 \\ -1 \\ 2 \\ 0 \end{bmatrix}$$

$\therefore y(n) = x_1(n) \circledast_8 x_2(n) = \{2, -1, -2, 2, -2, -1, 2, 0\}$

ii) Linear Convolution of $x_1(n)$ and $x_2(n)$:—

$x_1(n) = \{2, 1, 1, 2\}$; $L=4$

$x_2(n) = \{1, -1, -1, 1\}$; $M=4$

The length of $y(n)$ will be $N = L+M-1$

$N = 4+4-1 = 7$

$x_1(n) \rightarrow$ 2 1 1 2

$x_2(n) \rightarrow$ 1 -1 -1 1

		2	1	1	2	
	-2	-1	-1	-2		
-2	-1	-1	-2			
2	1	1	2			
2	-1	-2	2	-2	-1	2 $\Rightarrow y(n)$

$\therefore y(n) = \{2, -1, -2, 2, -2, -1, 2\}$

iii) For $N=7$, the linear convolution and circular convolution are same.

iv) Let the sequence $x_1(n)$ has length 'L' and $x_2(n)$ has the length 'M'.
Then, linear convolution of $x_1(n)$ and $x_2(n)$ will contain $L+M-1$ samples.

Scanned with CamScanner

Hence, linear and Circular Convolutions will be same if $N = L + M - 1$.

Obtain the linear convolved output, $y(n) = x(n) * h(n)$ using Circular Convolution. Given that, $x(n) = \{1, 1, 0, -1, -1\}$ and $h(n) = \{1, 2, 3, 2, 1\}$

Soln:- $x(n) = \{1, 1, 0, -1, -1\}$ i.e., $L = 5$

$h(n) = \{1, 2, 3, 2, 1\}$ i.e., $M = 5$

The length of $y(n) = x(n) * h(n)$ will be $N = L + M - 1$

$N = 5 + 5 - 1$

$\langle N = 9 \rangle$

$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \\ y(5) \\ y(6) \\ y(7) \\ y(8) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & -1 \\ -1 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 2 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 5 \\ 4 \\ 0 \\ -4 \\ -5 \\ -3 \\ -1 \end{bmatrix}$$

$\therefore y(n) = x(n) \circledast_9 h(n) = \{1, 3, 5, 4, 0, -4, -5, -3, -1\}$

Scanned with CamScanner

INFORMATION THEORY AND CODING

5TH SEM E&C



JAYANTHDWIJESH H P **M.tech (DECS)**

Assistant Professor – Dept of E&C

B.G.S INSTITUTE OF TECHNOLOGY (B.G.S.I.T)

B.G Nagara, Nagamangala Tq, Mandya District- 571448

FORMULAS FOR REFERENCE

MODULE -1(INFORMATION THEORY)

- **Amount of information or Self information.**

  $I_K = \log\left(\frac{1}{P_K}\right)$ or $I_K = \log_2\left(\frac{1}{P_K}\right)$ or $I\,(m_K) = \log\left(\frac{1}{P_K}\right)$

- **Entropy of source or Average information content of the source.**

  $H = \sum_{i=1}^{M} P_i \log_2\left(\frac{1}{P_i}\right)$ bits/symbol or $H = \sum_{K=1}^{M} P_K \log_2\left(\frac{1}{P_K}\right)$ bits/symbol or

  $H(S) = \sum_{i=1}^{q} P_i \log_2\left(\frac{1}{P_i}\right)$ bits/symbol or $H(S) = \sum_{i=1}^{q} P_i \log\left(\frac{1}{P_i}\right)$ bits/symbol or

  $H(S) = \sum_{K=1}^{N} P_K \log_2\left(\frac{1}{P_K}\right)$ bits/symbol

- **Information rate or average information rate.**

  $R_S = r_s H(S)$ bits/sec or $R = r_s H$ bits/sec or $R = r\,H$ bits/sec

- **Bits.**

  $I_K = \log_2\left(\frac{1}{P_K}\right)$ bits

- **Hartley's or Decits.**

  $I_K = \log_{10}\left(\frac{1}{P_K}\right)$ Hartley's or Decits

- **Nats or Neper.**

  $I_K = \log_e\left(\frac{1}{P_K}\right)$ Nats or Neper.

- **Extremal or Upper bound or Maximum entropy**

  $H(S)_{max} = \log_2 q$ bits/message-symbol or $H(S)_{max} = \log_2 N$ bits/message-symbol.

- **Source efficiency**

  $\eta_S = \frac{H(S)}{H(S)_{max}}$ or $\eta_S = \frac{H(S)}{H(S)_{max}} \times 100\%$

- **Source redundancy**

  $R_{\eta_S} = 1 - \eta_S = \left(1 - \frac{H(S)}{H(S)_{max}}\right) \times 100\%$

- **The average information content of the symbols emitted from the i th state.**

  $H_i = \sum_{j=1}^{n} P_{ij} \log_2\left(\frac{1}{P_{ij}}\right)$ bits/symbol or

  $H_i = \sum_{j=1}^{n} P_{ij} \log\left(\frac{1}{P_{ij}}\right)$ bits/symbol

- **The average information content of the symbols emitted from the k th state.**

$H_k = \sum_{l=1}^{M} P_{lK} \log_2 \left(\frac{1}{P_{lK}}\right)$ bits/symbol

- **The average information content per symbol in a message of length N.**

  $G_N = \frac{1}{N} \sum_i P(m_i) \log \frac{1}{P(m_i)}$ or $G_N = -\frac{1}{N} \sum_i P(m_i) \log P(m_i) = \frac{1}{N} H(\bar{s})$

- **The entropy of the second order symbols.**

  $G_N = \frac{1}{N} H(\bar{s}^N)$ where N=2.

- **The entropy of the third order symbols.**

  $G_N = \frac{1}{N} H(\bar{s}^N)$ where N=3.

- **Log properties**

  1. $\log_a b = \frac{1}{\log_b a}$

  2. $\frac{\log_x b}{\log_x a} = \log_a b$

  3. $\log_e 10 = \ln(10)$

CBCS – Scheme

MODULE-1

INFORMATION THEORY

Generic Information System:-

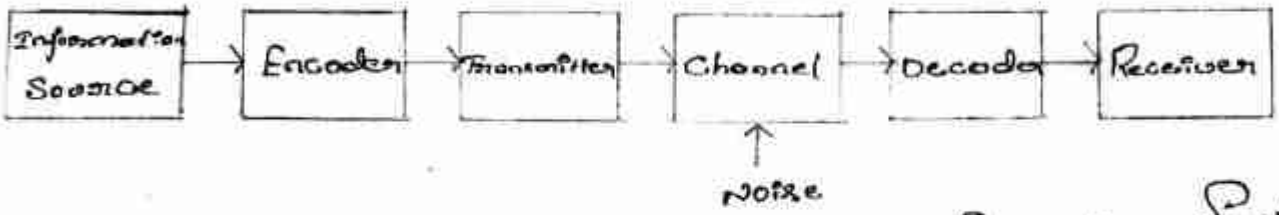


Fig:- Block diagram of an Information System

* The meaning of the word "Information" in Information theory is "message" or "intelligence". The message may be an Electrical message such as voltage, Current (or) power (or) Speech message (or) picture message. Such as - facsimile (or) Television (or) music message.

A Source which produces these messages is called Information Source.

* In General an Information Source Can be Classified as an analog Information Source (or) a discrete Information Source:

* Analog Information Source like a microphone actuated by speech (or) a video Camera produces one (or) more Continuous amplitude electrical Signals with respect to time.

* The Discrete Information Source such as a key board of a digital handheld terminal or a Computer produces a sequence of discrete Symbol (or) letter.

SOURCE ENCODER.

→ The Source Encoder Transforms the Sources o/p into (or) the Symbol Sequence into a binary Sequence of '0's and '1's.

→ Binary Coding is preferred because of its high efficiency of Transmission and also the ease with which they can be Transmitted over the Channel.

→ The Important parameters of a Source Coder namely block size, Length of code words, average data Rate and the Grander efficiency.

TRANSMITTER (or) MODULATOR.

→ The Transmitter couples the input message signal to the channel

→ Signal processing operations performed by the Transmitter include amplification, filtering and modulation.

→ The most important of these operations is modulation - a process designed to match the properties of transmitted signal to the channel through the use of a carrier wave.

CHANNEL.

→ A communication channel provides the electrical connection b/w the source and the destination.

→ The channel can be a pair of wires (or) a telephone cable (or) free space over which the information signal is transmitted. (or) radiated.

→ Due to the limited availability of the channel bandwidth, the information signal gets distorted. as it travels through the channel.

→ The signal is corrupted by the unwanted and unpredictable electrical disturbances called the noise.

DECODER:

→ The Source decoder converts the binary o/p of the Channel decoder into a Symbol sequence.

→ A Decoder performs the Reverse function of an Encoder.

→ The function of the decoder is to Convert the Corrupted Signals into a Symbol Sequence.

RECEIVER:

→ The function of the Receiver is to identify the Symbol Sequence and match it with the correct Sequence.

***

DEFINATION of INFORMATION [OR] MEASURE of INFORMATION :- JAN-09 [2M], DEC-2012 [2M].

[OR] SELF INFORMATION :- 2016-June [2M], JAN-2015 [2M], DEC-2013 [2M], DEC-2012 [1M], DEC-2012 [old scheme-2M]

[OR] AMOUNT of INFORMATION :- DEC-2010 [2M]

Let us Consider the Communication System which Transmits messages $m_1, m_2, m_3 \ldots\ldots$ with probability of occurence $p_1, p_2, p_3 \ldots\ldots$ The Amount of Information Transmitted through the message m_k with probability p_k is given as.

Amount of Information. $I_k = \log \frac{1}{p_k}$

[OR] $I_k = \log_2 \left[\frac{1}{p_k}\right]$,

TO CALCULATE THE UNITS OF INFORMATION :- we have

* If the base of the Logarithm is "2" Then the units are called "BITS" which is the short-form of "BINARY UNITS".

Ex:- $I = \log_2(1/p)$ ..... Bits //

* If the base is "10" the units are "HARTLEYS" (or) "DECITS".

Ex:- $I = \log_{10}(1/p)$ ..... decits (or) Hartleyunits

* If the base is "e" the units are "NATS".

Ex:- $I = \log_e(1/p)$ ..... nepers units/nats.

* If the base is general "n" The units are called "n - ary units".

Ex:- $I = \log_n(1/p)$ ..... //

NOTE:- $\log_2 x = \dfrac{\log_{10} x}{\log_{10} 2}$ //

***

DEFINATION OF INFORMATION [OR] MEASURE OF INFORMATION. :- Jan-09 [2M] Dec-2012 [2M]

[OR] SELF INFORMATION. :- 2016-June [2M], JAN-2015 [2M] DEC-2013 [2M], DEC-2012 [1M], DEC-2012 [old scheme-2M],

[OR] AMOUNT OF INFORMATION. :- DEC-2010 [2M]

Let us Conider the Communication System which Transmits messages $m_1, m_2, m_3 \ldots$ with probabilities of occurence $p_1, p_2, p_3 \ldots$ the Amount of Information Transmitted through the message m_k with probability p_k is given as.

$$\text{Amount of Information. } I_k = \log \frac{1}{p_k}$$

$$[\text{Or}] \quad I_k = \log_2 \left[\frac{1}{p_k}\right]$$

//.

PROBLEMS :-

1) A Source puts out one of five possible messages. during Each message Interval. The probabilities of these messages are $p_1 = 1/2$, $p_2 = 1/4$, $p_3 = 1/8$, $p_4 = 1/16$, $p_5 = 1/16$ Find the Information Content of Each of these messages.

Sol :- $I_1 = \log_2\left(\frac{1}{1/2}\right) = \log_2(2) = \frac{\log(2)}{\log(2)} = 1 \text{ bit}$ //

$$I_2 = \log_2\left(\frac{1}{1/4}\right) = \log_2(4) = \frac{\log(4)}{\log(2)} = 2 \text{ bits} \text{ //}$$

$I_3 = \log_2\left(\frac{1}{1/8}\right) = \log_2(8) = \frac{\log(8)}{\log(2)} = 3$ bits //

$I_4 = \log_2\left(\frac{1}{1/16}\right) = \log_2(16) = \frac{\log(16)}{\log(2)} = 4$ bits //

$I_5 = \log_2\left(\frac{1}{1/16}\right) = \log_2(16) = \frac{\log(16)}{\log(2)} = 4$ bits //

2) The binary Symbol "0" and "1" are Transmitted with probabilities 1/4 and 3/4 Respectively - find the Corresponding Self-Informations.

Sol:-

Self Information in a "0" $= I_0 = \log_2\left(\frac{1}{P_0}\right)$

$= \log_2\left(\frac{1}{1/4}\right) = \log_2(4)$

$= \frac{\log(4)}{\log(2)} = 2$ bits //

Self Information in a "1" $= I_1 = \log_2\left(\frac{1}{P_1}\right)$

$= \log_2\left(\frac{1}{3/4}\right) = \log_2\left(\frac{4}{3}\right)$

$= \frac{\log(4/3)}{\log(2)} = 0.415$ bits

3) Calculate the amount of Information if $P_K = 1/4$

Sol :-

Amount of Information is given as.

$$I_K = \log_2(1/P_K) = \log_2(1/1/4).$$

$$= \frac{\log 4}{\log 2} = 2 \text{ bits} //$$

* * *

4) In binary PCM if "0" occur with probability $1/4$ and "1" occur with probability $3/4$ Then Calculate Amount of Information Conveyed by each bit.

Sol :- Here bit '0' has $P_1 = 1/4$.

Here bit '1' has $P_2 = 3/4$

Then Amount of Information is given by.

$$I_K = \log_2(1/P_K)$$

* $P_1 = 1/4$, $I_1 = \log_2(1/1/4) = \log_2(4) = \frac{\log(4)}{\log(2)}$

$= 2 \text{ bits} //$

* $P_2 = 3/4$, $I_2 = \log_2(1/3/4) = \frac{\log(4/3)}{\log(2)} = 0.415 \text{ bits} //$

5) Calculate the Amount of Information of binary digits. (Binits) occurs with Equal likelihood in binary PCM.

Sol:- We know that in binary PCM, there are only two binary levels i.e 1 & 0. Since they occurs with Equal likehood, their probabilities of occurrence will be

$$P_1(\text{"0 Level"}) = P_2(\text{"1 level"}) = 1/2.$$

$$I_1 = \log_2\left(\frac{1}{P_1}\right) = \log_2\left(\frac{1}{1/2}\right) = \log_2(2)$$

$$= \frac{\log(2)}{\log(2)} = 1 \text{ bit} //$$

$$\text{Similarly } I_2 = \frac{\log(2)}{\log(2)} = 1 \text{ bit} //$$

***

6) Consider a discrete memoryless source "C" that output two bits at a time. This source comprises two binary sources 'A' and 'B', whose outputs are equally likely to occur and. Each source contributing one bit. Suppose that the two source within the source "C" are Independent what is the Information Content of each o/p from the Source "C"? [JAN-FEB 2008 (6 M)]

Sol:-

Source A and B emit "1" bit each. Hence probabilities of occurance of 1-bit from these sources will be

$p(A) = 1/2$ and $p(B) = 1/2$. Since 1 bit can be "1" or "0"

the o/p of source "C" is comprised of one bit each from 'A' and "B" hence the sample space of "C" will be $S(C) = \{0,0;\ 0,1;\ 1,0;\ 1,1\}$.

Since sources A and B are independent and equally likely the probability of each o/p of source "C" will be

$$p(C) = p(A) \times p(B) = 1/2 \times 1/2 = 1/4 \text{ //}$$

The Information content of each o/p from source "C" will be

$$I(C) = \log_2 \frac{1}{p(C)}$$

$$= \log_2 (1/1/4) = \log_2^{(4)} = \frac{\log(4)}{\log(2)}$$

$$= 2 \text{ bits} //$$

* * *

7) Consider a binary source which tosses a fair coin and outputs a '1' if a head appears and a '0' if a tail appears. What is the information content of each output from the source? Now suppose that the source is memoryless, what then is the self information of an m-bit block from the source.
Jan/Feb-2008 (HW)

Sol:- There are two o/p's '1' and '0' from the source. Both are equally likely. Hence probability of each o/p is.

$$P_k = 1/2 \quad \text{Here } k = 0,1$$

And Information Content of each o/p will be.

$$I = \log_2 \frac{1}{P_k} = \log_2\left(\frac{1}{1/2}\right) = \log_2(2).$$

$$= \frac{\log(2)}{\log(2)} = 1 \text{ bit}$$

If "m" bits are to be transmitted, or one single message block, the probability of such message will be

$$P(x) = 1/2 \times 1/2 \times 1/2 \times \text{---} \; 1/2 \text{ (m-times)} = 1/2^m.$$

Assuming that each m-bit message block is equally likely. the Information Content of such message will be.

$$I(x) = \log_2\left(\frac{1}{P(x)}\right) = \log_2\left(\frac{1}{1/2^m}\right) = \log_2(2^m)$$

$$= m \text{ bits}$$

AVERAGE INFORMATION CONTENT [ENTROPY] OF SYMBOLS IN LONG INDEPENDENT SEQUENCES.

OR

DERIVE AN EXPRESSION FOR AVERAGE INFORMATION CONTENT OF LONG INDEPENDENT MESSAGES.

July-2009 [2M], DEC-2010 [4M], July-2011 [4M], July-2016 [3M], JAN-2019 [4M] July-2014 [3M], July-2011, July-2009 [3M], Jan-2010 [3M], Jan-10 [4M] Jan-2009 [3M], DEC-2012 [3M], DEC-2010 [4M],

⇒ A source that emits one of 'M' possible symbols $S_1, S_2, \ldots S_M$ in a statistically independent sequence.

→ Let $P_1, P_2, \ldots P_M$ be the probabilities of occurrence of "M" symbols, respectively.

→ Now in a long message containing N symbols, the symbols "S_1" will occur on the average P_1N times, the symbols "S_2" will occur on the average P_2N times, and in general the symbol S_i will occur P_iN times.

→ The information content of the i^{th} symbol is $\log_2\left(\frac{1}{P_i}\right)$ bits.

→ ∴ P_iN occurrences of S_i contributes an information content of $P_iN \log_2\left(\frac{1}{P_i}\right)$ bits.

→ The Total Information Content of the message is. Then the Sum of the Contribution due to each of the 'M' Symbols of the Source alphabet is. given by. $I_{total} = \sum_{i=1}^{M} NP_i \log_2\left(\frac{1}{P_i}\right)$ bits.

→ The Average Information per Symbol by dividing the total Information Content of the message by the number of Symbols in the message as.

$$H = \frac{I_{total}}{N} \quad \sum_{i=1}^{M} P_i \log_2\left(\frac{1}{P_i}\right) \text{ bits/Symbol} \quad \text{---(1)}$$

∴ The definition of "H" given in eq (1) is based on "Time averaging".

→ The average Information Content per Symbol is also called the "Source Entropy" //

OR

→ Let us Consider a zero-memory source producing Independent Sequence of Symbols. while the Receiver of this Sequence may interpret the Entire message as a Single unit, Communication Systems often have to deal with Individual Symbols.

→ Let us Consider the Source alphabet $S = \{s_1, s_2, \ldots s_q\}$ with probabilities $p = \{p_1, p_2, \ldots p_q\}$ respectively.

→ Let us Consider a long Independent Sequence of length "L" Symbols. This long Sequence the Contains :-

⇒ $p_1 L$ number of messages of type s_1,
$p_2 L$ number of messages of type s_2,
⋮
and $p_q L$ number of messages of types s_q.

→ The Self Information is given by

$$s_k = I_k = \log \frac{1}{p_k}$$

Hey the Self Information of $s_1 = \log 1/p_1$ bits.

∴ $p_1 L$ number of messages of type s_1 Contain $p_1 L \log 1/p_1$ bits of Information.

∴ $p_2 L$ number of message of type s_2 Contain $p_2 L \log 1/p_2$ bits of Information.

⋮

$P_q L$ number of messages of Type S_q Contain.

$P_q L \log \frac{1}{P_q}$ bits of Information.

→ the Total Self Information Content of all these message Symbols is given by

$$I_{total} = P_1 L \log \frac{1}{P_1} + P_2 L \log \frac{1}{P_2} + \cdots + P_q L \log \frac{1}{P_q} \text{ bits}$$

$$= L \sum_{i=1}^{q} P_i \log \frac{1}{P_i} \text{ bits}$$

→ Average Self Information $= \frac{I_{total}}{L}$.

$$= \sum_{i=1}^{q} P_i \log \frac{1}{P_i} \text{ bits/message Symbol.}$$

→ Average Self Information is also called "ENTROPY" of Source 'S' denoted by $H(S)$

$$\therefore H(S) = \sum_{i=1}^{q} P_i \log \frac{1}{P_i} \text{ bits/message Symbol}$$

NOTE: Log properties:

1) $\log_b b = \frac{1}{\log_b a}$ 2) $\frac{\log_x b}{\log_x a} = \log_a b$ 3) $\log_e 10 = \ln$

OR

→ A Zero memory Source (or) a Discrete memoryless Source is the one in which the emission of the Consecutive symbol is not dependent on the Emission of the previous symbols.

→ Consider a Source Emitting symbol $S = \{S_1, S_2 \cdots S_N\}$ with Respective probabilities $P = \{P_1, P_2, P_3 \cdots P_N\}$.

→ Now Consider a long message of length "L" emitted by the source. Then it Contains.

$P_1 L$ number of symbols of 'S_1'

$P_2 L$ number of symbols of 'S_2'

$P_N L$ number of symbols of 'S_N'

→ The self Info of 'S_1' is given by

$$I_{S1} = \log_2\left(\frac{1}{P_1}\right) \text{ bits.}$$

→ In other words. Each symbol of 'S_1' Conveys Information of $\log_2(1/P_1)$ bits and. Such $P_1 L$ number of 'S_1' symbols are present on an average in a Length of "L" symbols.

∴ The Total Information Conveyed by symbol of type 'S_1' is

$$P_1 L \log_2 \frac{1}{P_1} \text{ bits.}$$

Illy. Total Information Conveyed by Symbol of S_2 is

$$P_2 L \log_2 \frac{1}{P_2} \text{ bits.}$$

⋮

Total Information Conveyed by Symbol of S_N is.

$$P_N L \log_2 \frac{1}{P_N} \text{ bits.}$$

∴ the Total Information Conveyed by the source is Simply the Sum of all these Information Content that is.

$$I_{TOTAL} = P_1 L \log_2 \frac{1}{P_1} + P_2 L \log_2 \frac{1}{P_2} + \cdots + P_N L \log_2 \frac{1}{P_N}$$

∴ the Average Information Conveyed by the Source by emitting "L" Symbol is denoted by it's Entropy $H(S)$. which is given by the Expression

$$H(S) = \frac{I_{TOTAL}}{L}.$$

$$H(s) = P_1 L \log\frac{1}{P_1} + P_2 L \log\frac{1}{P_2} + \cdots + P_N L \log_2\frac{1}{P_N}.$$

$$= \frac{L}{L}\left[P_1 \log\frac{1}{P_1} + P_2 \log\frac{1}{P_2} + \cdots + P_N \log\frac{1}{P_N}\right]$$

$$= P_1 \log\frac{1}{P_1} + P_2 \log\frac{1}{P_2} + \cdots + P_N \log_2\frac{1}{P_N}.$$

$$H(s) = \sum_{k=1}^{N} P_k \log_2\frac{1}{P_k} \text{ bits/sym.}$$

INFORMATION RATE :- Dec-2008(2M), Dec-2012(2M), Dec-2012(2M) Jan-2014(2M), Jan-2015(2M), July-2016(2M)

The Information rate is represented by "R" and it is given as.

Information Rate $R = r_s H$ bits/sec

(or) $R_s = r_s H(s)$ bits/sec //

Here 'R' is Information rate.

'H' is Entropy (or) average Information.

and r_s is rate at which messages are generated.

"Information rate 'R' is represented in average number of bits of Information per second."

It is Calculated as follows.

$$R = \left(r_s \text{ in } \frac{\text{messages}}{\text{second}}\right) \times \left(H \text{ in } \frac{\text{Information bits}}{\text{message}}\right)$$

= Information bits/second //

SYMBOL RATE :-

The number of Symbols emitted from the Source per second is known as Symbol rate denoted as "r_s"

Unit : Symbol/sec.

PROBLEMS :-

1) A discrete source emits one of five symbols once every millisecond. The symbols once every millisecond. The symbol probabilities are $1/2$, $1/4$, $1/8$, $1/16$ and $1/16$ respectively. Find the source entropy and information rate.

NOTE

$P \rightarrow \{1/2, 1/4, 1/8, 1/16, 1/16\}$ $i=1 \dots\dots i=5$

Sol:-

Entropy :-

$$H = \sum_{i=1}^{5} P_i \log_2\left(\frac{1}{P_i}\right) \text{ bits/symbol}$$

$$= 1/2 \log_2(2) + 1/4 \log_2(4) + 1/8 \log_2(8) + 1/16 \log_2(16) + 1/16 \log_2(16)$$

$$= 1/2 \frac{\log 2}{\log 2} + 1/4 \frac{\log 4}{\log 2} + 1/8 \frac{\log 8}{\log 2} + 1/16 \frac{\log 16}{\log 2} + 1/16 \frac{\log 16}{\log 2}$$

$$= 1/2 (1) + 1/4 (2) + 1/8 (3) + 1/16 (4) + 1/16 (4)$$

$$= 0.5(1) + 0.25(2) + 0.125(3) + 0.0625(4) + 0.0625(4)$$

$$= 1.875 \text{ bits/symbol} //$$

Information Rate. $R = r_s H$ bits/sec

$$= (1000)(1.875)$$

$$= 1875 \text{ bits/sec} //$$

2) Consider a source $S = \{S_1, S_2, S_3\}$ with $P = \{1/2, 1/4, 1/4\}$.
Find (a) Self Information of each messages
(b) Entropy of source 'S'.

Sol: Let $P_1 = 1/2$, $P_2 = 1/4$, $P_3 = 1/4$

(a)

Self Information of $S_1 = I_1 = \log_2 \frac{1}{P_1}$

$= \log_2 \frac{1}{1/2} = \frac{\log_{10}(2)}{\log_{10}(2)}$

$= 1$ bits //

Self Information of $S_2 = I_2 = \log_2 \frac{1}{P_2} = \log_2\left(\frac{1}{1/4}\right) = \log_2(4)$

$= \frac{\log_{10}(4)}{\log_{10}(2)}$

$= 2$ bits //

Self Information of $S_3 = I_3 = \log_2 \frac{1}{P_3} = \log_2\left(\frac{1}{1/4}\right)$

$= \frac{\log_{10}(4)}{\log_{10}(2)} = 2$ bits //

(b) Entropy is given by. $H(S) = \sum_{i=1}^{q} P_i \log_2 \frac{1}{P_i}$

$\hookrightarrow I_i$

$= \sum_{i=1}^{q} P_i I_i$

Consider $q = 3$

$H(S) = \sum_{i=1}^{3} P_i I_i = P_1 I_1 + P_2 I_2 + P_3 I_3$

note -

Self Information $I_K = \log_{10} \frac{1}{P_K}$

Entropy $H(S) = \sum_{i=1}^{q} P_i \log \frac{1}{P_i}$

$$H(s) = \left(\frac{1}{2}\right)(1) + \left(\frac{1}{4}\right)(2) + \left(\frac{1}{4}\right)(2).$$

$$H(s) = 1.5 \text{ bits/message. Symbol}$$

***

3) The output of an Information Source consists of 150 symbols, 32 of which occur with a probability of $\frac{1}{64}$ and the remaining 118 occur with a probability of $\frac{1}{236}$. The Source emits 2000 symbols/sec. Assuming that the symbols are chosen independently find the average Information rate of this source.

→ July-2011 6[5M], Jan-2014 [4M]

Sol:- The Entropy is given by

$$H(s) = \sum_{i=1}^{2} P_i \log_2 \frac{1}{P_i}$$

$$= \sum_{i=1}^{32} P_i \log_2 \frac{1}{P_i} + \sum_{i=33}^{150} P_i \log \frac{1}{P_i}$$

$$= \left[\frac{1}{64} \log_2\left(\frac{1}{\frac{1}{64}}\right)\right] \times 32 + \left[\frac{1}{236} \log_2\left(\frac{1}{\frac{1}{236}}\right)\right] \times 118$$

$$= \left[\frac{1}{64} \log_2(64)\right] \times 32 + \left[\frac{1}{236} \log_2[236]\right] \times 118$$

$$= \left[\frac{1}{64} \times \frac{\log(64)}{\log 2}\right] \times 32 + \left[\frac{1}{236} \times \frac{\log 236}{\log 2}\right] \times 118$$

$$= \frac{1}{64} \times 6 \times 32 + \frac{1}{236} \times 7.884 \times 118$$

$H(s) = 3 + 3.9413 = 6.9413$ bits/ message Symbol.

∴ Given $r_s = 2000$ message Symbols/sec

∴ Average Information rate $= R = r_s H$ bits/sec.

(or) $R_s = r_s H(s)$ bits/sec

$= 2000 \times 6.9413$

$= 13,882.6$ bits/sec.

* * *

4) The output of an Information Source consists of 128 Symbols of 16 of which occurs with a probability of 1/32 and. Remaining occurs with a probability of 1/224. The Source emits 1000 Symbols/sec assuming that Symbols are chosen independently. Find the average Information rate of the Source.

July-2016[5M], Jan-2014[4M]

Sol:- The Entropy is given by

$$H(s) = \sum_{i=1}^{q} P_i \log_2\left(\frac{1}{P_i}\right) = \sum_{i=1}^{128} P_i \log_2\left(\frac{1}{P_i}\right)$$

(or) $$H(s) = \sum_{i=1}^{16} P_i \log_2 \frac{1}{P_i} + \sum_{i=17}^{128} P_i \log_2 \frac{1}{P_i}$$

$$= \left[\frac{1}{32} \cdot \log_2(32)\right] \times 16 + \left[\frac{1}{224} \cdot \log_2(224)\right] \times 112.$$

$$= \frac{1}{32}(5)(16) + \left(\frac{1}{224}\right)(7.8073)(112)$$

$$= 6.40365 \text{ bits/symbol} //$$

$\therefore$ Given $r_s = 1000$ symbol/sec.

$$R_s = r_s H(z) \text{ bits/sec.}$$

$$= 1000 \text{ symbol/sec.} \times 6.40365 \text{ bits/symbol}$$

$$= 6403.65 \text{ bits/sec} //$$

* A code is composed of dots and dashes. Assuming that a dash is 3 times as long as a dot and has one-third the probability of occurrence. Calculate.

i) The information in a dot and dash.

ii) The entropy of dot-dash code.

iii) The average rate of information if a dot lasts for 10-msec and the time is allowed b/w symbols.

Sol :-

Since only dots and dashes are present the must have.

$P_{dot} + P_{dash} = 1 \quad -(1)$

Given: $P_{dash} = \frac{1}{3} P_{dot} \quad -(2)$

NOTE
sum of all probs equal to one
$P_1 + P_2 \cdots P_n = 1$
i.e. $P_{dot} + P_{dash} = 1$

eq (2) Sub in to eq (1)

$P_{dot} + \frac{1}{3} P_{dot} = 1$

$4/3 P_{dot} = 1$

$P_{dot} = 3/4 // \quad -(3)$

eq (3) Sub into eq (2)

$P_{dash} = (1/3)(3/4) = 1/4 //$

i) Information in a dot $= I_{dot} = \log_2 \frac{1}{P_{dot}} = \log_2 \frac{1}{3/4} = \log 4/3$

$= 0.4150 \text{ bits}$

Information in a dash $= I_{dash} = \log_2 \frac{1}{P_{dash}} = \log_2 \frac{1}{1/4} = \log_2(4)$

$= 2 \text{ bits} //$

(i) The entropy of dot dash code is.

$$H(S) = \sum_{i=1}^{2} P_i \log_2 (1/P_i)$$

$$= P_{dot} \log \frac{1}{P_{dot}} + P_{dash} \log \frac{1}{P_{dash}}$$

$$= 3/4 \log (4/3) + 1/4 \log (4)$$

$$= 0.8113 \text{ bits/message-symbol}$$ //

(ii) Since $P_{dot} = \frac{3}{4}$ and $P_{dash} = \frac{1}{4}$ for every 4 symbols sent on an average. There will be 3 dots and 1 dash. with a dot lasting for 10m-sec. a dash for 30m-sec and with 10m-sec gap in b/w 2 successive symbols, a total of 100m-sec is required to transmit 4 symbols.

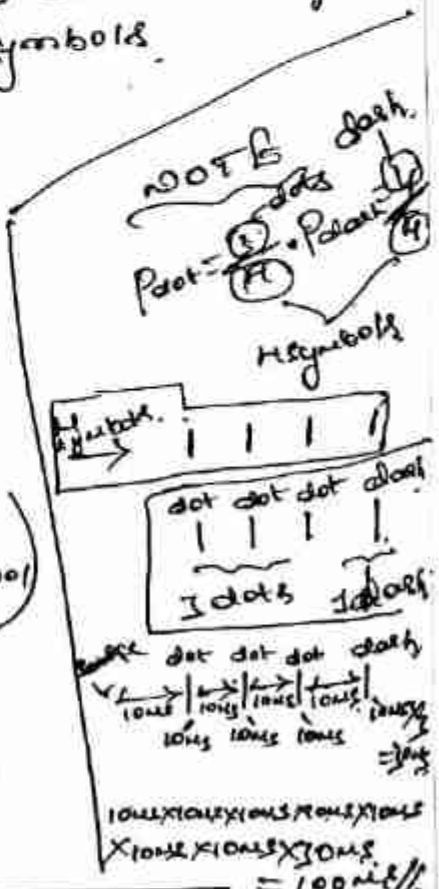
$\therefore$ Symbol rate $= r_s = 4 \text{ symbols}/100\text{m sec}$

$$= \frac{4}{100 \times 10^{-3}} = 40 \text{ symbols/sec}$$

$\therefore$ Information Rate $= R_s = r_s H(S)$

$$= (40 \text{ symbols/sec}) \times (0.8113 \text{ bits/symbol})$$

$$= 32.452 \text{ bits/sec}$$ //



* A Source emits one of four Symbols S_0, S_1, S_2, and S_3 with probabilities $1/3$, $1/6$, $1/4$ and $1/4$ respectively. The successive Symbols emitted by the source are statistically independent. Calculate the entropy of the source.

Sol:- The entropy of the source is given by -

$$H(S) = \sum_{i=0}^{3} P_i \log_2 \frac{1}{P_i}$$

$$= P_0 \log_2 \frac{1}{P_0} + P_1 \log_2 \frac{1}{P_1} + P_2 \log_2 \frac{1}{P_2} + P_3 \log_2 \frac{1}{P_3}$$

$$= \frac{1}{3}\log(3) + \frac{1}{6}\log(6) + \frac{1}{4}\log(4) + \frac{1}{4}\log(4)$$

$$= 0.5283 + 0.4308 + 0.5 + 0.5$$

$$H(S) = 1.9591 \text{ bits/message Symbol}$$

* * *

* A Card is drawn from a deck.

1) You are told it is a Spade. How much Information did you Receive?

2) How much Information did you Receive if you are told that the Card drawn is an ace?

3) If you are told that the Card drawn is an ACE of SPADES, how much Information did you Receive?

4) Is the Information obtained in (3) the sum of Information obtained in (1) and (2)?

Sol:- 1) Since there are "13" Spade cards in a deck of 52 cards.

$$P_{spade} = \frac{13}{52} = 0.25 //$$

$\therefore$ Self Information $I_{spade} = \log \frac{1}{P_{spade}} = \log_2 4 = 2 \text{ bits} //$

2) Since the are "4" ACES in a deck of 52 cards.

$$P_{ACE} = \frac{4}{52} = \frac{1}{13} //$$

$\therefore$ Self Information $I_{ace} = \log \frac{1}{P_{ace}} = \log 13 = 3.7 \text{ bits}$ //

3) Since there is only "one" ace-of-spades in a deck of 52 cards.

$$P_{ace\text{-}of\text{-}spades} = \frac{1}{52}$$

$\therefore$ Self Information $I_{ace\text{-}of\text{-}spades} = \log_2 \frac{1}{P_{ace\text{-}of\text{-}spade}}$

$= \log_2 52$

$= 5.7 \text{ bits} //$

4) "YES". The Information obtained in (3) is the Sum of Information obtained in (1) and (2). //

* A fair coin is tossed repeatedly. Let A = {event of getting 3 heads out of 5 tosses} B = {event of getting 8 heads out of 8 tosses} which event conveys more Information. Support your answer by numerical computation of respective amount of Information. June / July-2013 [5M]

Sol:- Let us first find out the probabilities of events A and B by using the Concept of binomial distribution. It is given by

$$f(x) = P(X=x) = {}_nC_x\, p^x\, q^{n-x}, \quad x = 0, 1, 2, \ldots n. \quad \text{---(1)}$$

Let us define the Random variable.

$X \rightarrow$ number of heads.

= binomial Random variable [since tossing a Coin Results in only 2 outcomes → head (or) tail]

n = number of Throws. [5 and 8]

p = probability of getting head = 0.5

$q = 1 - p = 0.5$ //

From eq (1)

$$P(X=x) = {}_nC_x\, p^x\, q^{n-x}$$

$$\therefore P(A) = P(x=3) = {}_5C_3\, (0.5)^3\, (0.5)^{5-3}$$

$$= 0.3125 \text{ //}$$

$$P(B) = P(x=5) = {}_8C_3\, (0.5)^5\, (0.5)^{8-5} = 0.21875 \text{ //}$$

Jayashankar Vijendra B.P, BE (E&C), M.Tech (DECS).
Assistant Professor - Dept of E&C
BGSIT, B.G Nagara, Nagamangala Tq, Mandya.

Self Info of event A : $I_A = \log_2 \frac{1}{P(A)} = \log_2 \frac{1}{0.3125}$

$= 1.678$ bits //

self Info of Event B : $I_B = \log_2 \frac{1}{P(B)} = \log_2 \frac{1}{0.21875}$

$= 2.1936$ bits //

∴ Event - B Conveys more amount of Info. //

* The corrected voltage of Centaro Computer is either b/w -5 and -12 volts. The voltage can take on only three values -5, -6, -7, -9, -12 volts with respective probabilities $\frac{1}{6}, \frac{1}{3}, \frac{1}{12}, \frac{1}{12}, \frac{1}{16}, \frac{1}{16}$. This voltage is recorded with a pen recorder. Determine the average self-Info associated with the Record in terms of bits/level.

Sol:

$$H(S) = \sum_{i=1}^{6} P_i \log \frac{1}{P_i} \text{ bits/level}$$

$$= \frac{1}{6}\log 6 + \frac{1}{3}\log 3 + \frac{1}{12}\log 12 + \frac{1}{12}\log 12 + \frac{1}{16}\log 16 + \frac{1}{16}\log 16$$

$= 2.418$ bits/level //

* FIND RELATIONSHIP B/W HARTLEYS, NATS AND BITS
June-2012 [6M].

Sol:- from definition we have

Hartleys $\Rightarrow I = \log_{10}\frac{1}{p}$ Hartleys. ---(1)

nats $\Rightarrow I = \log_e \frac{1}{p}$ nats. ---(2)

Bits $\Rightarrow I = \log_2 \frac{1}{p}$ bits ----(3)

1) Hartley $\longleftrightarrow$ Bits

From eq (1)

$$1 \text{ Hartley} = \frac{I}{\log_{10}\frac{1}{p}}$$

NOTE

$I = \log_{10}\frac{1}{p}$ Hartleys

Hartleys $= \frac{I}{\log_{10}\frac{1}{p}}$

using eq (3)

$$1 \text{ Hartley} = \frac{\log_2 \frac{1}{p}}{\log_{10}\frac{1}{p}} \text{ bits.}$$

$$= \frac{-\log_2(p)}{-\log_{10}(p)} \text{ bits.}$$

NOTE

$y = \log_2(1/a)$

$= -\log_2(a)$

$$= \frac{\log_2(p)}{\log_{10}(p)}$$

$= \frac{\log_2(P)}{\log_{10}(P)}$ bits.

$\because \log_a(b) = \frac{1}{\log_b(a)}$

$= \frac{\frac{1}{\log_p(2)}}{\frac{1}{\log_p(10)}} = \frac{\log_p(10)}{\log_p(2)} = \log_2(10)$ bits

$= \frac{\log 10}{\log 2}$ bits.

$= 3.32$ bits

2) Hartley $\longleftrightarrow$ nats.

From eq (1) $I = \log_{10}\left(\frac{1}{P}\right)$ Hartley

$1 \text{ Hartley} = \frac{I}{\log_{10}\left(\frac{1}{P}\right)}$

Using eq (2) $\Rightarrow$

$1 \text{ Hartley} = \frac{\log_e\left(\frac{1}{P}\right)}{\log_{10}\left(\frac{1}{P}\right)}$ nats.

$= \frac{-\log_e(P)}{-\log_{10}(P)} = \frac{\log_e(P)}{\log_{10}(P)}$ nats

DOUBT

$$\therefore \quad \frac{1}{\log_b(a)} = \log_a(b).$$

1 Hartley $= \dfrac{1}{\dfrac{\log_p(e)}{\log_p(10)}} = \dfrac{\log_p(10)}{\log_p(e)}$ nats

$= \log_e(10)$ nats.

$= \ln(10)$ nats.

$= 2.3025$ nats

3) nats $\longleftrightarrow$ Bits.

From eq (3) $I = \log_e(1/p)$ nats.

1 nat $= \dfrac{I}{\log_e(1/p)}$

using eq. (3)

1 nat $= \dfrac{\log_2(1/p)}{\log_e(1/p)}$ bits $= \dfrac{-\log_2(p)}{-\log_e(p)}$ bits

$= \dfrac{1}{\dfrac{\log_p(2)}{\log_p(e)}} = \dfrac{\log_p(e)}{\log_p(2)} = \log_2(e) = 1.4426$ bits

V.V.V.V.V.
MRP(V.T.U).

* A Discrete Source emits one of the following five Symbols in every 1s. The Symbol probabilities are $\{\frac{1}{4}, \frac{1}{8}, \frac{1}{8}, \frac{3}{16}, \frac{5}{16}\}$. Find the average Information Content of the Source in NATS/SYM and HARTLEY/SYM.

Sol:-

The Source entropy is given by.

NATS/SYM :-

$$H(S) = \sum_{K=1}^{N} P_K \log_e \frac{1}{P_K} \text{ nats/Symbol}$$

(or)

$$H(S) = \sum_{i=1}^{q} P_i \log_e \frac{1}{P_i} \text{ nats/Symbol.}$$

let $[S_1, S_2, S_3, S_4, S_5] = \{\frac{1}{4}, \frac{1}{8}, \frac{1}{8}, \frac{3}{16}, \frac{5}{16}\}$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$P_1 \quad P_2 \quad P_3 \quad P_4 \quad P_5$

$$H(S) = \sum_{i=1}^{5} P_i \log_e \frac{1}{P_i} \text{ nats/Symbol.}$$

$$= \frac{1}{4} \log_e \left(\frac{1}{1/4}\right) + \frac{1}{8} \log_e \left(\frac{1}{1/8}\right) + \frac{3}{16} \log_e \left(\frac{1}{3/16}\right) + \frac{5}{16} \log_e \left(\frac{1}{5/16}\right)$$

$$= \frac{1}{4} \log_e (4) + \frac{1}{8} \log_e (8) + \frac{1}{8} \log_e (8) + \frac{3}{16} \log_e \left(\frac{16}{3}\right) + \frac{5}{16} \log_e \left(\frac{16}{5}\right)$$

$$= \frac{1}{4} \ln(4) + \frac{1}{8} \ln(8) + \frac{1}{8} \ln(8) + \frac{3}{16} \ln\left(\frac{16}{3}\right) + \frac{5}{16} \ln\left(\frac{16}{5}\right)$$

$$= 1.543 \text{ nats/Symbol} //$$

IIIly.

HARTLEY/SYM :-

$$H(s) = \sum_{i=1}^{5} P_i \log_{10} \frac{1}{P_i} \quad \text{Hartley/Symbol.}$$

$$= \frac{1}{4}\log_{10}\left(\frac{1}{1/4}\right) + \frac{1}{8}\log_{10}\left(\frac{1}{1/8}\right) + \frac{1}{8}\log_{10}\left(\frac{1}{1/8}\right) + \frac{3}{16}\log_{10}\left(\frac{1}{3/16}\right) + \frac{5}{16}\log_{10}\left(\frac{1}{5/16}\right)$$

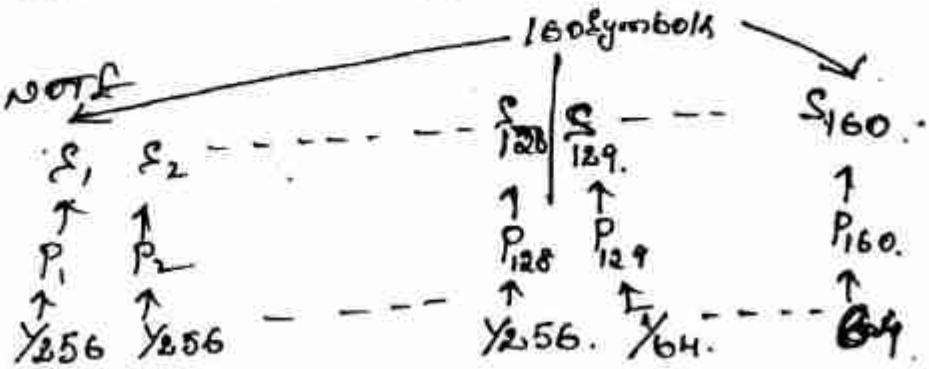
$$= \frac{1}{4}\log_{10}(4) + \frac{1}{8}\log_{10}(8) + \frac{1}{8}\log_{10}(8) + \frac{3}{16}\log_{10}\left(\frac{16}{3}\right) + \frac{5}{16}\log_{10}\left(\frac{16}{5}\right)$$

$$= 0.6704 \text{ Hartley/Symbol} \;//$$

IM
V.V.V.V

* The output of an Info Source Contains 160 symbols. 128 of which occur with a probability of 1/256 and. Remaining with a probability of 1/64 each. find the Average Information Rate of the Source if the Source emits 10000 symbols/sec.

Sol :-



The Source entropy is given by.

$$H(S) = \sum_{i=1}^{160} P_i \log_2 \left(\frac{1}{P_i}\right)$$

$$= (128)\left(\frac{1}{256}\right)\log_2 \frac{1}{(1/256)} + (32)\left(\frac{1}{64}\right)\log_2 \frac{1}{(1/64)}$$

$$= 7 \text{ bits/symbol} \quad //$$

(or)

$$H(S) = \sum_{i=1}^{128} P_i \log\left(\frac{1}{P_i}\right) + \sum_{i=129}^{160} P_i \log\left(\frac{1}{P_i}\right)$$

$$= (128)\left(\frac{1}{256}\right)\log_2\left(\frac{1}{1/256}\right) + (32)\left(\frac{1}{64}\right)\log_2\left(\frac{1}{1/64}\right)$$

$$= 7 \text{ bits/symbol} \quad //$$

The average Information rate is given by

$$R_s = r_s H(S) \text{ bits/sec}$$

(or) $R = r_s H(S)$ bits/sec

$$R = 10000 \text{ symbol/sec} \times 7 \text{ bits/symbol}$$

$$= 70000 \text{ bits/sec} \quad //$$

* A Discrete memoryless source emits 5 symbols every 2ms. The symbol probabilities are {0.5, 0.25, 0.125, 0.0625, 0.0625}. Find the average Information rate of the source.

Sol :- $H(S) = \sum_{i=1}^{q} P_i \log_2\left(\frac{1}{P_i}\right)$ bits/symbol

Let $[S_1, S_2, S_3, S_4, S_5] = [0.5, 0.25, 0.125, 0.0625, 0.0625]$

$H(S) = \sum_{i=1}^{5} P_i \log_2\left(\frac{1}{P_i}\right)$ bits/symbol

$= (0.5)\log_2\left(\frac{1}{0.5}\right) + (0.25)\log_2\left(\frac{1}{0.25}\right) + (0.125)\log_2\left(\frac{1}{0.125}\right)$
$+ (0.0625)\log_2\left(\frac{1}{0.0625}\right) + (0.0625)\log_2\left(\frac{1}{0.0625}\right)$

$= (0.5)(1) + (0.25)(2) + 0.125(3) + 0.0625(4) + 0.0625(4)$

$= 1.875$ bits/symbol //

average Information rate is given by.

$R = r_s H$ bits/sec

[or] $R_s = r_s H(S)$ bits/sec

$\therefore r_s = \frac{1}{2m_s} = \frac{1}{2\times10^{-3}} = 500$ symbols/sec.

$R_s = 500$ symbols/sec $\times 1.875$ bits/symbol

$= 937.5$ bits/sec //

* Consider a source emitting 2 symbol S_0 and S_1 with the corresponding probabilities $3/4$ and $1/4$ respectively. find the self information of the symbol in
(a) Bits (b) Decits (c) nats.

Sol:- (a) $I_K = \log_2 \left(\frac{1}{P_K}\right)$ bits.

Let $S_0 = 3/4$ $S_1 = 1/4$.

$I_{S_0} = \log_2 \frac{1}{P(S_0)} = \log_2 \left(\frac{1}{3/4}\right) = \log_2 \left(\frac{4}{3}\right) = 0.415$ bits //

$I_{S_1} = \log_2 \frac{1}{P(S_1)} = \log_2 \frac{1}{(1/4)} = \log_2 (4) = 2$ bits //

(b) $I_K = \log_{10} \left(\frac{1}{P_K}\right)$ decits.

$I_{S_0} = \log_{10} \frac{1}{P(S_0)} = \log_{10} \left(\frac{1}{3/4}\right) = 0.124$ decits //

$I_{S_1} = \log_{10} \frac{1}{P(S_1)} = \log_{10} \left(\frac{1}{1/4}\right) = 0.602$ decits //

(c) $I_K = \log_e \left(\frac{1}{P_K}\right)$ nats.

$I_{S_0} = \log_e \frac{1}{P(S_0)} = \log_e \left(\frac{1}{3/4}\right) = 0.287$ nats //

$I_{S_1} = \log_e \frac{1}{P(S_1)} = \log_e \frac{1}{(1/4)} = \ln(4)$

$= 1.386$ nats //

* * *

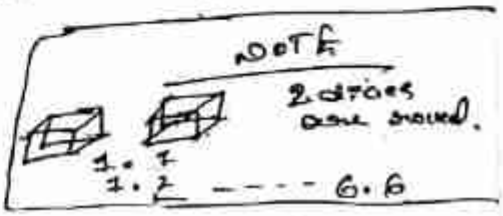
A pair of dice is rolled. Simultaneously. The outcome of the first dice is considered to be "x" and that of the second as "y". Three events are defined as.

P = {(x,y) such that (x+y) is exactly divisible by 3}

Q = {(x,y) such that (x+y) is an even number}

R = {(x,y) such that (x+y) = 7}

Which event conveys more Information? Justify your answer. Justify mathematical Calculation.



Sol:-

Sample is given by

$$S = \begin{Bmatrix} (1,1) & (1,2) & (1,3) & (1,4) & (1,5) & (1,6) \\ (2,1) & (2,2) & (2,3) & (2,4) & (2,5) & (2,6) \\ (3,1) & (3,2) & (3,3) & (3,4) & (3,5) & (3,6) \\ (4,1) & (4,2) & (4,3) & (4,4) & (4,5) & (4,6) \\ (5,1) & (5,2) & (5,3) & (5,4) & (5,5) & (5,6) \\ (6,1) & (6,2) & (6,3) & (6,4) & (6,5) & (6,6) \end{Bmatrix}$$

$$P = \begin{Bmatrix} (1,2) & (1,5) & (2,1) & (2,4) & (3,3) & (3,6) \\ (4,2) & (4,5) & (5,1) & (5,4) & (6,3) & (6,6) \end{Bmatrix}$$

NOTE

1+2 = 3 exactly divisible by 3

$^2/_3 = 1/1$

Like 1+5 = 6

$^6/_3 = 2/1$

6+6 = 12

$^{12}/_3 = 4$ //

$Q = \{(1,1)\ (1,3)\ (1,5)\ (2,2)\ (2,4)\ (2,6)\ (3,1)\ (3,3)\ (3,5)$
$(4,2)\ (4,4)\ (4,6)\ (5,1)\ (5,3)\ (5,5)\ (6,2)\ (6,4)\ (6,6)\}$

NOTE
$1+1=2$
$1+3=4$
$\vdots$
$6+6=12$
→ even number

$R = \{(1,6)\ (2,5)\ (3,4)\ (4,3)\ (5,2)\ (6,1)\}$

NOTE
$1+6=7$
$2+5=7$
$\vdots$
$6+1=7$
→ exactly equal to 7.

Then $P(P) = \frac{12}{36}$ //

$P(Q) = \frac{18}{36}$ // $P(R) = \frac{6}{36}$ //

Considering the self Information of event P, Q & R.

$I_k = \log_2 \frac{1}{P_k}$ bits

$I_P = \log_2 \frac{1}{P(P)} = \log_2 \frac{1}{(\frac{12}{36})} = 1.584$ bits //

$I_Q = \log_2 \frac{1}{P(Q)} = \log_2 \frac{1}{18/36} = 1$ bits //

$I_R = \log_2 \frac{1}{P(R)} = \log_2 \frac{1}{6/36} = 2.584$ bits //

Ans. $I_R > I_P$ & I_Q event "R" Conveys more Information.

* A discrete source emits one of the following five symbols in every 1 s. The symbol probabilities are $\{\frac{1}{4}, \frac{1}{8}, \frac{1}{8}, \frac{3}{16}, \frac{5}{16}\}$. Find the average Information Content of the source in nats/symbol and Hartley/symbol.

Sol :- $H(s) = \sum_{i=1}^{q} p_i \log_e \frac{1}{p_i}$ nats/symbol.

NATS/SYMBOL :- $H(s) = \frac{1}{4}\log_e(4) + \frac{1}{8}\log_e(8) + \frac{1}{8}\log_e(8) + \frac{3}{16}\log_e\left(\frac{16}{3}\right)$.

$= \frac{1}{4}\ln(4) + \frac{1}{8}\ln(8) + \frac{1}{8}\ln(8) + \frac{3}{16}\ln\left(\frac{16}{3}\right)$

$= 1.1803$ nats/symbol //

HARTLEY/SYMBOL :-

$H(s) = \sum_{i=1}^{q} p_i \log_{10}\frac{1}{p_i}$ Hartley/symbol

$= \frac{1}{4}\log_{10}(4) + \frac{1}{8}\log_{10}(8) + \frac{1}{8}\log_{10}(8) + \frac{3}{16}\log_{10}\left(\frac{16}{3}\right)$

$= 0.5125$ Hartley/symbol //

Jayanthdwijesh h.p, BE (E&C), M.tech (DECS).
Assistant Professor - Dept of E&C
BGSIT, b.g Nagara, Nagamangala Tq, Mandya.

Ex. 4.4

A binary source is emitting an independent sequence of 0's and 1's with probabilities 'p' and 1−p respectively. plot the Entropy of source versus 'p'.

OR

A source transmits two independent messages with probabilities of 'p' and (1−p) respectively. Prove the Entropy is maximum when both the messages are equally likely. plot the variation of Entropy (H) as a function of probability 'p' of the messages.

Aug-2001 [6M], Feb-2002 [10M], Feb-2006 [6M], Aug 2008 [6M], June-2010 [ ], June-2012 [ ].

Sol:- Entropy is given by $H(S) = \sum_{i=1}^{q} p_i \log_2\left[\frac{1}{p_i}\right]$

for the messages above equation can be

$$H(S) = \sum_{i=1}^{2} p_i \log_2\left(\frac{1}{p_i}\right)$$

$$H(S) = p_1 \log_2 \frac{1}{p_1} + p_2 \log_2 \frac{1}{p_2}$$

Let $p_1 = p$ & $p_2 = 1-p$

$$H(S) = p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{(1-p)}$$

//

$$H(s) = p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{(1-p)}$$

Let $p = 0.1$ $\quad H(s) = 0.1 \log_2 \frac{1}{(0.1)} + (1-0.1) \log_2 \frac{1}{(1-0.1)}$.

$= 0.469$ bits/symbol //.

Let $p = 0.2$ $\quad H(s) = 0.2 \log_2 \frac{1}{(0.2)} + (1-0.2) \log_2 \frac{1}{(1-0.2)}$

$= 0.722$ bits/symbol //

Let $p = 0.3$ $\quad H(s) = 0.3 \log_2 \frac{1}{(0.3)} + (1-0.3) \log_2 \frac{1}{(1-0.3)}$

$= 0.881$ bits/symbol //

Let $p = 0.4$ $\quad H(s) = 0.4 \log_2 \frac{1}{(0.4)} + (1-0.4) \log_2 \frac{1}{(1-0.4)}$

$= 0.971$ bits/Symbol //

Let $p = 0.5$ $\quad H(s) = 0.5 \log_2 \frac{1}{0.5} + (1-0.5) \log_2 \frac{1}{(1-0.5)}$

$= 1$ bits/Symbol //

Let $p = 0.6$ $\quad H(s) = 0.6 \log_2 \frac{1}{0.6} + (1-0.6) \log_2 \frac{1}{(1-0.6)}$

$= 0.971$ bits/Symbol //.

Let $p = 0.7$ $\quad H(s) = 0.7 \log_2 \frac{1}{0.7} + (1-0.7) \log_2 \frac{1}{(1-0.7)}$

$= 0.881$ bits/Symbol //.

Let $p=0.8$

$$H(s) = 0.8 \log_2 \frac{1}{0.8} + (1-0.8) \log_2 \frac{1}{(1-0.8)}.$$

$$= 0.722 \text{ bits/symbol} //$$

Let $p=0.9$ $\quad H(s) = 0.9 \log_2 \frac{1}{0.9} + (1-0.9) \log_2 \frac{1}{(1-0.9)}$

$$= 0.469 \text{ bits/symbol} //$$

Let $p=0$ and $p=1$ $\quad H(s) = 0 //$.

The above calculated values can be tabulated as shown in below Table.

p	H(s) bits/m-sym
0	0
0.1	0.469
0.2	0.722.
0.3	0.881
0.4	0.971
0.5	1
0.6	0.971
0.7	0.881
0.8	0.722.
0.9	0.469.
1	0

Table :- Entropy values for various probabilities.

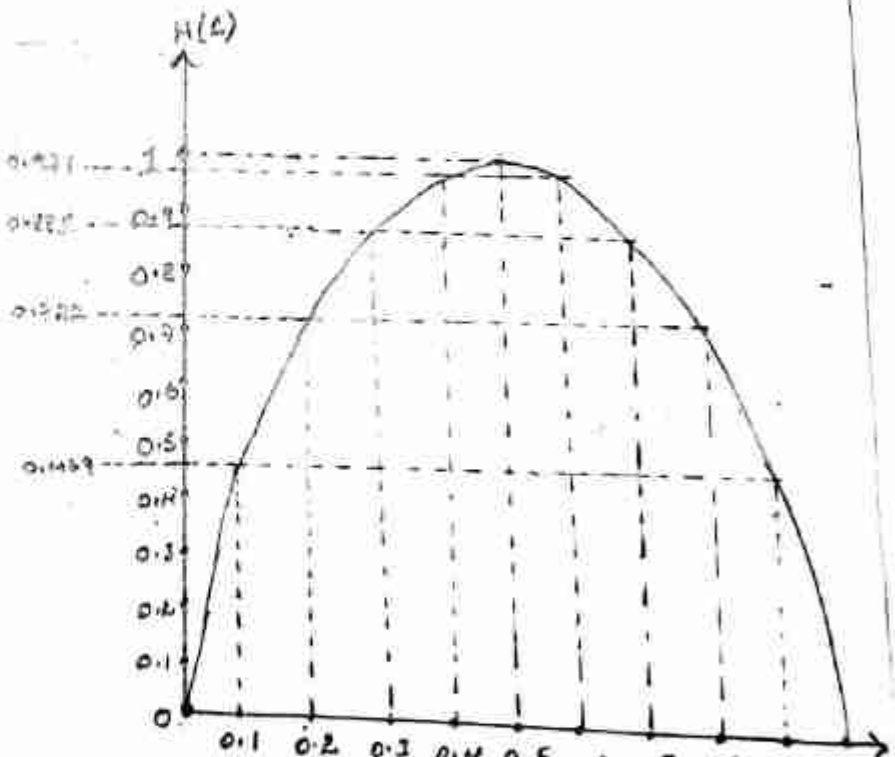


Fig :- plot of H(s) versus p

PROPERTIES OF ENTROPY:-

The following are the properties of Entropy.

property 1 :- Entropy is a Continuous function of probability

property 2 :- Entropy is a Symmetric function of its arguments. In other words the Entropy of a system is the same irrespective of the order in which the symbol are arranged in the list.

V.T.U. ***IM

property 3 :- EXTREMAL PROPERTY OR UPPER BOUND ON ENTROPY.

DEC-2013/JAN-2014. [4M], DEC-2009/JAN-2012 [10M] DEC-08/09 [8M].

The extremal property defines the upper bound on the value of Entropy. It gives the higher value of H(S)

proof :- Consider an Information Source 'S' emitting 'q' Symbols. $S = \{s_1, s_2, \ldots, s_q\}$ With probabilities $p = \{p_1, p_2, \ldots p_q\}$ Such that

p.T.O

The Entropy of 'S' is given by

$$H(s) = \sum_{i=1}^{q} P_i \log \frac{1}{P_i}$$

and $$\sum_{i=1}^{q} P_i = 1$$

Let us now prove that the Entropy H(s) has an "upperbound" by Considering a quantity

$$\therefore \log q - H(s).$$

$$\therefore \log q - H(s) = \left[\sum_{i=1}^{q} P_i\right] \log q - \sum_{i=1}^{q} P_i \log \frac{1}{P_i}$$

Since $\sum_{i=1}^{q} P_i = 1$ //.

$$= \sum_{i=1}^{q} P_i \left[\log q - \log \frac{1}{P_i}\right]$$

$$= \sum_{i=1}^{q} P_i \left[\log q - (-\log(P_i))\right]$$

$$= \sum_{i=1}^{q} P_i \left[\log q + \log P_i\right]$$

$$= \sum_{i=1}^{q} P_i \log(qP_i).$$

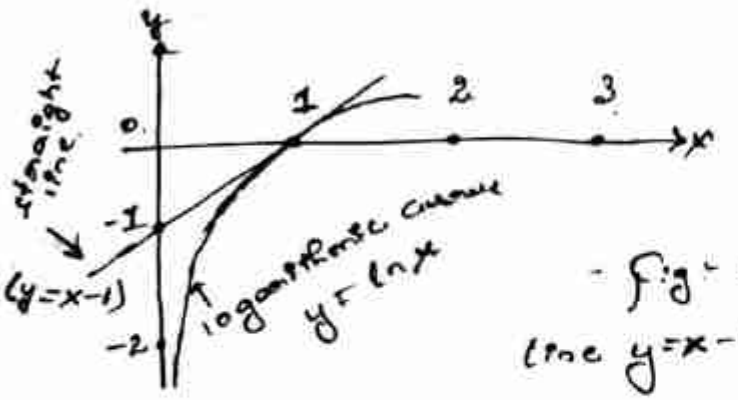
NOTE: $\log A + \log B = \log AB$

Assume $$\log q - H(s). = \sum_{i=1}^{q} P_i \frac{\log_e(qP_i)}{\log_e(2)}$$

$$= \frac{1}{\log_e(2)} \sum_{i=1}^{q} P_i \log_e(qP_i)$$

NOTE: $\log_b(a) = \frac{1}{\log_a(b)}$, $\log_b(a) = \frac{1}{\log_a(b)}$

$$\log q - H(s). = \log_2(e) \sum_{i=1}^{q} P_i \ln(qP_i) \quad \text{---(1)}$$



- Fig - above. Consider a graph of straight line $y=x-1$ and Logarithmic Curve $y=\ln x$

From the above graph it is Evident that the straight line $y=x-1$ always lies above the Logarithmic Curve $y=\ln x$ except at $x=1$.

$\therefore \ln x \leq x-1$

Xy by -1 we get

$-\ln x \geq 1-x$ (or) $\ln \frac{1}{x} \geq 1-x$

If $x = \frac{1}{qP_i}$

Then $\ln \frac{1}{(qP_i)} \geq 1-\left[\frac{1}{qP_i}\right]$

$\therefore \ln (qP_i) \geq 1-\left(\frac{1}{qP_i}\right)$ — (2)

Xy eq (2) both Sides by "P_i" and Then Taking Summation for all $i = 1, 2, \ldots q$ we get

$$\sum_{i=1}^{q} P_i \ln (qP_i) \geq \sum_{i=1}^{q} P_i \left[1 - \frac{1}{qP_i}\right]$$

Xy both Sides by $\log_2 e$ we get

$$\log_2 e \sum_{i=1}^{q} P_i \ln (qP_i) \geq \log_2 e \left[\sum_{i=1}^{q} P_i - \sum_{i=1}^{q} \frac{1}{q}\right] \quad \text{— (3)}$$

From eq (2) L.H.S of equation (3) is $\log q - H(x)$ and R.H.S of equation (3) is always "0"

$\therefore \log q - H(x) \geq 0$

or $H(x) \leq \log_2 q$

The equality sign holds good when $p_i - \frac{1}{q} = 0$ for all values of 'i'

i.e $p_i = \frac{1}{q}$ for all $i = 1, 2, \ldots q$ (4)

When condition of equation (4) is satisfied, then the entropy becomes maximum given by.

$H(x)_{max} = \log_2 q$ bits/message-symbol

i.e the entropy attains a "maximum value" when all the source symbol becomes "equiprobable".

property 4:- PROPERTY OF ADDITIVITY / SPLIT SYMBOL ENTROPY :-

Suppose that we split the symbol s_q into 'n' sub-symbols such that $s_q = s_{q1}, s_{q2}, \ldots s_{qn}$ occurring with probabilities $p_{q1}, p_{q2}, \ldots p_{qn}$ such that.

$$p_q = p_{q1} + p_{q2} + \ldots + p_{qn} = \sum_{j=1}^{n} p_{qj}$$

Then the splitted symbol entropy is

$$H'(x) = H(p_1, p_2 \ldots p_{q-1}, p_{q1}, p_{q2} \ldots p_{qn})$$

$$= \sum_{i=1}^{q-1} p_i \log \frac{1}{p_i} + \sum_{j=1}^{n} p_{qj} \log \frac{1}{p_{qj}}$$

$$H'(s) = \sum_{i=1}^{q} P_i \log \frac{1}{P_i} - P_q \log \frac{1}{P_q} + \sum_{j=1}^{n} P_{qj} \log \frac{1}{P_{qj}}$$

$$= \sum_{i=1}^{q} P_i \log \frac{1}{P_i} - \sum_{j=1}^{n} P_{qj} \log \frac{1}{P_q} + \sum_{j=1}^{n} P_{qj} \log \frac{1}{P_{qj}}$$

$$= \sum_{i=1}^{q} P_i \log \frac{1}{P_i} + \sum_{j=1}^{n} P_{qj} \left[\log \frac{1}{P_{qj}} - \log \frac{1}{P_q} \right]$$

$$= \sum_{i=1}^{q} P_i \log \frac{1}{P_i} + P_q \sum_{j=1}^{n} \frac{P_{qj}}{P_q} \left[\log \frac{P_q}{P_{qj}} \right]$$

NOTE: $\log(A) - \log(B) = \log \frac{A}{B}$, $\log \frac{1}{P_{qj}} = \log P_q - \log P_{qj}$

NOTE: $xy \div by\ P_q$ → 2

$\therefore H'(s) = H(s) +$ a positive quantity Since $P_{qj} \leq P_q$ for all j

$\therefore H'(s) \geq H(s)$

i.e partitioning of Symbols into Sub-Symbols Cannot decrease the entropy //

5. :- SOURCE EFFICIENCY :-

The "SOURCE EFFICIENCY" denoted by η_s is given by

$$\eta_s = \frac{H(s)}{H(s)_{max.}} \quad \text{OR} \quad \eta_s = \frac{H(s)}{H(s)_{max}} \times 100\%$$

and The "SOURCE REDUNDANCY" denoted by R_{η_s} is given by

$$R_{\eta_s} = 1 - \eta_s \ //$$

NOTE :- η_s and R_{η_s} Can also be Expressed as percentage

* Consider a System Emitting one of the Three Symbols A, B and C with Respective probabilities 0.7, 0.15, and 0.15 Calculate its efficiency and Redundancy.

Sol: The efficiency is given by

$$\eta_s = \frac{H(s)}{H(s)_{max}} \times 100\%$$

Entropy:-

$$\therefore H(s) = \sum_{i=1}^{q} P_i \log_2\left(\frac{1}{P_i}\right) \text{ bits/symbol.}$$

$$= \sum_{i=1}^{3} P_i \log_2\left(\frac{1}{P_i}\right) \text{ bits/symbol}$$

$$= 0.7 \log_2\left(\frac{1}{0.7}\right) + 0.15 \log_2\left(\frac{1}{0.15}\right)(2)$$

$$= 1.1812 \text{ bits/sym.} //$$

The upper bound of the Entropy is given by

$$H(s)_{max} = \log_2(q) \text{ bits/symbol}$$

$$= \log_2(3) = 1.5849 \text{ bits/symbol} //$$

$\therefore$ efficiency is given by

$$\eta_s = \frac{1.1812}{1.5849} \times 100\% = 74.52\% //$$

$\therefore$ The Redundancy is given by

$$R_{\eta_s} = 1 - \eta_s = 1 - 74.52\% = 25.48\% //$$

*A Discrete memory less Source Contains Source alphabet $S = \{S_1, S_2, S_3, S_4\}$ with $p = \{1/2, 1/4, 1/8, 1/8\}$. Calculate the Entropy of the Source, Calculate the Entropy of 2nd order extension of the Source.

Sol:-

Entropy is given by $H(S) = \sum_{i=1}^{q} p_i \log_2\left(\frac{1}{p_i}\right)$

$$= \frac{1}{2}\log_2\left(\frac{1}{1/2}\right) + \frac{1}{4}\log_2\left(\frac{1}{1/4}\right) + \frac{1}{8}\log_2\left(\frac{1}{1/8}\right) + \frac{1}{8}\log_2\left(\frac{1}{1/8}\right)$$

$$= \frac{1}{2}\log_2(2) + \frac{1}{4}\log_2(4) + \frac{1}{8}\log_2(8) + \frac{1}{8}\log_2(8).$$

$$= 1.75 \text{ bits/message} - \text{Symbol.} \quad //$$

Maximum Entropy is given by

$$H(S)_{max} = \log_2 q \text{ bits/message-Symbol}$$

where $q = 4$ //

NOTE

S_1, S_2, S_3, S_4

$\underbrace{\qquad\qquad}_{4}$

$$H(S)_{max} = \log_2(4)$$

$$H(S)_{max} = 2 \text{ bits/message Symbol} \quad //$$

* * *

In a facsimile Transmission of pictures, there are about 2.25×10^6 pixels / frame. For a good reproduction 12 brightness levels are necessary. Assume all these levels are equally likely to occur. Find the rate of Information if one picture is to be transmitted every "3" minutes. What is the Source efficiency of this facsimile Transmitter? June 2010 [5M].

Sol:-

Total number of pixel in one frame $= 2.25 \times 10^6$

Number of brightness levels $= 12$.

Total number of different frames possible $= (12)^{2.25 \times 10^6}$.

The next maximum Information Content per frame is.

$$I = H(s)_{max} = \log_2 q$$

$$= \log_2 (12)^{2.25 \times 10^6}$$

$$= (2.25 \times 10^6) \log_2 (12).$$

$$= 8.066 \times 10^6 \text{ bits/picture}$$

//

Given that 1 picture is Transmitted in 3 minutes.

∴ The rate of Transmission is given by

$$r_s = \frac{1}{3 \text{ minutes}} = \frac{1}{3 \times 60} \text{ pictures/sec.}$$

∴ the average rate of Information is given by

$$R = r_s H \text{ bits/sec}$$

(or)

$$R = r_s H(x)_{max} \text{ bits/sec}$$

$$= r_s I = \frac{1}{3 \times 60} \times 8.066 \times 10^6$$

$$= 44818 \text{ bits/sec}$$

The Information Transmitted is maximum. The source efficiency

$$\eta_s = \frac{H(x) \text{ (or) } H}{H(x)_{max}} = \frac{H(x)_{max}}{H(x)_{max}}$$

$$= 1$$

$$= 100\%$$

*In a facsimile Transmission of a picture there are 4×10^6 pixels/frame. For a good Reconstruction of the Image at least 8 brightness levels are necessary. Assuming all these levels are equally likely to occur find the average Information Rate If one picture is transmitted every 4 s

Sol: Given

Total number of pixels = 4×10^6

Total number of brightness levels = 8

Total number of different frames $N = 8^{(4 \times 10^6)}$

All the brightness levels are equally likely to occur, all these frames are equiprobable. So

$$I = H(s) = H(s)_{max} = \log_2 q \text{ (or) } \log_2 N$$

$$= \log_2 (8)^{4 \times 10^6}$$

$$H(s)_{max} = 4 \times 10^6 \log_2 (8) = 12 \times 10^6 \text{ bits/frame}$$

Given $r_s = \frac{1}{4}$ frames/sec.

$$R_s = H(s) \times r_s = 12 \times 10^6 \times \frac{1}{4}$$

$$= 3 \times 10^6 \text{ bits/sec}$$

* A black and white TV picture consists of 525 lines of picture information. Assume that each line consists of 525 pixels with each pixel having 256 equiprobable brightness level. pictures are repeated at the rate of 30 frames/per second. Calculate the Rate of Information conveyed by a TV set to the viewer. Jan-2014 [6M]

Sol: * TOTAL number of pixel in one frame $= 525 \times 525$
$= 2,75,625$ pixels.

* Each pixel can have 256 different brightness level & from fully black to fully white. Then the total number of different frames possible $= (256)^{2,75,625}$ frames.

* Let us assume that all these frames occur with equal probability. The net maximum Inf^n Content per frame is

$$I = H(x)_{max} = \log_2 q = \log_2 (256)^{2,75,625} = 2,75,625 \log_2 (256)$$
$$= 22.05 \times 10^5 \text{ bits/frame.}$$

* given $r_s = 30$ frames/sec

∴ the average Information Rate given by

$$R_s = r_s H(x)_{max} \Rightarrow R_s = r_s I$$
$$= (30 \text{ frames/sec})(22.05 \times 10^5 \text{ bits/frame})$$
$$= 66150000 \text{ bits/sec}$$

* A black and white TV picture consists of 525 lines of picture information. Assume that each line consists of 525 picture elements (pixels) and that each consists have 255 brightness levels picture is repeated at the rate 30 frames/sec. Calculate the average rate of information conveyed by a TV set to a viewer.

Jan-2015 [6M]

Sol :- * Total number of pixels in one frame = 525×525

$= 276676$ pixels

* Each pixel can have 255 different brightness levels from fully black to fully white. Then the total number of different frames possible (q) = $(255)^{276676}$ frames //

* Let us assume that all these frames occur with equal probability. The net maximum Inf^n content per frame is.

$$I = H(s)_{max} = \log_2 q = \log_2 (255)^{276676}$$

$$= 276676 \log_2 (255)$$

$$= 22.118405 \times 10^5 \text{ bits/frame}$$

* given $r_s = 30$ frames/sec.

$$R_s = r_s H(s)_{max} = r_s I$$

$$= (30 \text{ frames/sec}) \times (22.118405 \times 10^5) \text{ bits/frame}$$

$$= 66355350 \text{ bits/sec} //$$

* Suppose that S_1 and S_2 are two - memory source with probabilities $p_1, p_2, \ldots p_n$ for source S_1 and $q_1, q_2, \ldots q_n$ for source S_2. Show that the Entropy of source S_1

$$H(S_1) \leq \sum_{k=1}^{n} p_k \log \frac{1}{q_k}$$

Sol:- Given S_1 and S_2 are zero-memory Info source.

$$\therefore H(S_1) = \sum_{k=1}^{n} p_k \log \frac{1}{p_k} \quad \text{---(1)}$$

and $\sum_{k=1}^{n} p_k = 1$ // ——— (X)

$$\therefore H(S_2) = \sum_{k=1}^{n} q_k \log \frac{1}{q_k} \quad \text{---(2)}$$

and $\sum_{k=1}^{n} q_k = 1$ // ——— (Y)

Consider: $H(S_1) - \sum_{k=1}^{n} p_k \log \frac{1}{q_k} = \sum_{k=1}^{n} p_k \log \frac{1}{p_k} - \sum_{k=1}^{n} p_k \log \frac{1}{q_k}$

$$H(S_1) - \sum_{k=1}^{n} p_k \log \frac{1}{q_k} = \sum_{k=1}^{n} p_k \left[\log \frac{1}{p_k} - \log \frac{1}{q_k} \right]$$

$$= \sum_{k=1}^{n} p_k \left[\log \left(\tfrac{1}{p_k}\right) - \left(-\log(q_k)\right) \right]$$

$$= \sum_{k=1}^{n} p_k \left[\log \left(\tfrac{1}{p_k}\right) + \log q(k) \right]$$

$$= \sum_{k=1}^{n} p_k \log \left(\tfrac{1}{p_k}\right)(q_k)$$

$$= \sum_{k=1}^{n} p_k \log \left(\frac{q_k}{p_k}\right).$$

NOTE: $\log(a) + \log(b) = \log(ab)$ //

$$= \sum_{k=1}^{n} P_k \cdot \frac{\log_e\left(\frac{q_k}{P_k}\right)}{\log_e(2)}$$

ROOT1 $\log c = \ln$

$$= \frac{1}{\log_e(2)} \sum_{k=1}^{n} P_k \ln\left(\frac{q_k}{P_k}\right)$$

ROOT2 $\frac{-1}{\log_{(b)}(a)} = \log_b(a)$

$$= \log_2(e) \sum_{k=1}^{n} P_k \ln\left(\frac{q_k}{P_k}\right) \longleftarrow$$

// ⟵ (3)

Consider equation.

$$\ln \frac{1}{x} \geq 1 - x$$

$$-\ln(x) \geq 1 - x$$

Multiply -1 on both side

$$\ln(x) \leq -1 + x \quad \text{---} \quad (a)$$

where $x = \frac{q_k}{P_k}$ we have. ⟵ (b)

eq (a) sub into eq (b)

$$\ln\left(\frac{q_k}{P_k}\right) \leq -1 + \left(\frac{q_k}{P_k}\right)$$

multiply by "P_k". Taking Summation for all $k = 1, 2, \ldots n$ and the multiply by $\log_2 e$ on both side we get

$$\log_2 e \sum_{k=1}^{n} P_k \ln\left(\frac{q_k}{P_k}\right) \leq \log_2 e \sum_{k=1}^{n} P_k \left[\frac{q_k}{P_k} - 1\right]$$

⟵ (4)

eq (3) Subst into eq (4)

$$H(s_1) - \sum_{k=1}^{n} p_k \log \frac{1}{q_k} \leq \log_2 e \sum_{k=1}^{n} p_k \left[\frac{q_k}{p_k} - 1\right]$$

$$\leq \log_2 e \sum_{k=1}^{n} \frac{p_k q_k}{p_k} - p_k$$

$$\leq \log_2 e \left[\sum_{k=1}^{n} q_k - \sum_{k=1}^{n} p_k\right] - (5)$$

Using eq (x) & (y) subst into eq (5)

$$H(s_1) - \sum_{k=1}^{n} p_k \log \frac{1}{q_k} \leq \log_2 e \left[1 - 1\right]$$

$$\leq 0.$$

$$\therefore \quad H(s_1) \leq \sum_{k=1}^{n} p_k \log \frac{1}{q_k}. \quad \text{Proved.}$$

* A Discrete message source "S" emits two Independent Symbol X and Y with probabilities 0.55 and 0.45 Respectively Calculate the efficiency of the Source and its Redundancy.

Sol :- $P_x = P(X) = 0.55$ $P_y = P(Y) = 0.45$

Entropy $H(s) = \sum_{i=1}^{2} P_i \log \frac{1}{P_i}$

$= \sum_{i=X}^{Y} P_i \log \frac{1}{P_i}$

$= P_x \log \frac{1}{P_x} + P_y \log \frac{1}{P_y}$

$= 0.55 \log \frac{1}{0.55} + 0.45 \log \frac{1}{0.45}$

$H(s) = 0.9928$ bits/message symbol //

The maximum Entropy is given by.

$H(s)_{max} = \log_2 2 = \log_2(2) = 1$ bits/message symbol //

The Source efficiency is given by

$\eta_s = \frac{H(s)}{H(s)_{max}} = \frac{0.9928}{1} \times 100 = 99.28\%$ //

The Source Redundancy is given by.

$R_{ny} = 1 - \eta_s = (1 - 0.9928) \times 100\% = (0.0072) \times 100\%$

$= 0.72\%$ //

* A certain data source has 8 symbols that are produced in blocks of four at a rate of 500 blocks/sec. The first symbol in each block is always the same (presumably for synchronization). The remaining three are filled by any of the 8 symbols with equal probability. What is the entropy rate of this source?

Ex.1:-

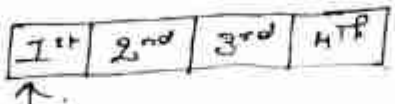


Fig:- Illustrating above example.

Given, $q = 8$ symbols.

$r_s = 500$ block/sec.

* The symbol are produced in blocks of four. The total entropy H_T is.

$$H_T = H_1 + H_2 + H_3 + H_4 \quad \text{---(1)}$$

where H_4 is the entropy of the 4th bit.

* Since 1st bit is always same. No information is carried by the sure event and. Hence

$$H_1 = 0. \quad \text{---(2)}$$

* Given that the 2nd position is occupied by any of the 8 symbols with equal probability.

$$\therefore H_2 = H(x)_{max} = \log_2(q)$$

$$H_2 = \log_2(8) = 3 \text{ bits/Message-symbol} \quad \text{---(3)}$$

Since 3rd and 4th position are also occupied by any of the 8 equiprobable symbol. We have

$$H_3 = H_4 = 3 \text{ bits/message symbol} \quad \text{---} \quad (4)$$

eq (2), (3) & (4) Substituting into eq (1)

$$H_T = 0 + 3 + 3 + 3$$

$$= 9 \text{ bits/block}$$

$\therefore$ Entropy rate is given by

$$R = r_s H(s)$$

(or) $$R_s = r_s H(s)$$

(or) $$R_s = r_s H_i$$

$$= (500 \text{ blocks/sec})(9 \text{ bits/block})$$

$$= 4500 \text{ bits/sec}$$

* A certain digital frame consists of 15 fields. The first and the last field of each frame are same. The remaining 13 fields can be filled by any of the 16-symbols with equal probability. Find the average information conveyed by the frame. Also find the average rate of information if 100 frames are transmitted every second.

Sol: Consider the frame with 15 fields as shown below.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

The total average information conveyed by the source is simply the sum of the individual entropies of each field in the frame i.e.

$$H(s)_{Total} = H_1 + H_2 + \dots + H_{15}$$

As the 1st and last fields (15) are same for all the frame. It conveys no information.

i.e $H_1 = H_{15} = 0$ //.

P.T.O

Remaining fields can be filled with one of the 16 Symbol with Equal probability:

∴ The corresponding entropy is the maximum entropy $H(s)_{max}$ So.

$H_2 = H_3 = \cdots = H_{14} = H(s)_{max} = \log_2 N$

$= \log_2(16)$

$= 4$ bits/-field.

$=$

∴ $H(s)_{total} = H_1 + H_2 + H_3 + \cdots + H_{15}$

$= 0 + 4 + 4 + 4 + 4 + 4 + 4 + 4 + 4 + 4 + 4 + 4 + 4 + 0$

$= 4 \times 13 = 52$ bits/frame //

∴ Information Rate is given by

$R_s = r_s H(s)$ bits/sec

$R_s = 52 \times 100$

$R_s = 52000$ bits/sec //

Given,

$r_s = 100$ frames/sec.

* Find the Info. Content of a message that Consists of a digital word "9" digits long in which each digit may take on one of five possible levels. The probability of sending any of the five levels is assumed to be equal, and the level in any digit does not depend on the values taken by previous digits

Sol :-

Given :- Length of digital word = 9 digits.

Number of levels = 5 equiprobable.



↑ Entropy H_1 ↑ Entropy H_2. - - - - ↑ Entropy H_9.

The Total Entropy of 9 digit word is given by.

$$H_T = H_1 + H_2 + - - - - + H_9$$

Where H_i is the Entropy of the i^{th} digit, $i = 1, 2, - - - 9$.

Since each digit can taken on one of five equiprobable levels, the Entropy of each digit is given by.

$$H = H(s)_{max} = \log_2 q = \log_2(5) = 2.322 \text{ bits/level}$$

$$H_1 = H_2 = - - - - - - - - H_9 = 2.322 \text{ bits/level}$$

∴ Total Info. Content $H_T = (9)(2.322)$.

$$= 20.898 \text{ bits/level}$$

* A pair of dice are tossed simultaneously. The outcome of the first dice is recorded as X_1 and that of second dice as X_2. Two events are defined as follows.

$A = \{(X_1, X_2) \text{ such that } X_1 + X_2 \leq 7\}$

$B = \{(X_1, X_2) \text{ such that } X_1 > X_2\}$

which event conveys more Info? Support your Answer by numerical computation.

Sol:- When a pair of dice are tossed simultaneously. Then the sample space 'S' consists of 36. combinations of (X_1, X_2) given by

$$S = \begin{bmatrix} (1,1) & (1,2) & (1,3) & (1,4) & (1,5) & (1,6) \\ (2,1) & (2,2) & (2,3) & (2,4) & (2,5) & (2,6) \\ (3,1) & (3,2) & (3,3) & (3,4) & (3,5) & (3,6) \\ (4,1) & (4,2) & (4,3) & (4,4) & (4,5) & (4,6) \\ (5,1) & (5,2) & (5,3) & (5,4) & (5,5) & (5,6) \\ (6,1) & (6,2) & (6,3) & (6,4) & (6,5) & (6,6) \end{bmatrix}$$

The event 'A' contains the pairs as given by

$A = \{(1,1)\ (1,2)\ (1,3)\ (1,4)\ (1,5)\ (1,6)\ (2,1)\ (2,2)\ (2,3)\ (2,4)\ (2,5)\ (3,1)\ (3,2)\ (3,3)\ (3,4)\ (4,1)\ (4,2)\ (4,3)\ (5,1)\ (5,2)\ (6,1)\}$.

Note:- adding 2 numbers equal to below, leaves on exact leaves.

Example
1+2=3
1+4=5
6+1=7 //

Note
$x_1 > x_2$
$2 > 1$ - - -
- - - $6 > 5$

The Event 'B' Contains the pairs

$B = \{(2,1)\ (3,1)\ (3,2)\ (4,1)\ (4,2)\ (4,3)\ (5,4)\ (5,2)$
$(5,1)\ (5,4)\ (6,1)\ (6,2)\ (6,3)\ (6,4)\ (6,5)\}$

Since all the pairs are equiprobable,

$P(A) = \frac{21}{36} = \frac{7}{12}$ // & $P(B) = \frac{15}{36} = \frac{5}{12}$ //

∴ Self Information of event $A = I_A = \log \frac{1}{P(A)}$

$= \log \frac{12}{7}$

$= 0.7776$ bits //

∴ Self-Infn. of Event $B = I_B = \log \frac{1}{P(B)}$

$= \log \frac{12}{5} = 1.263$ bits

∴ $I_B > I_A$.

The Event 'B' Conveys more Information that Event 'A' //

* Shortly before a horse-race, a book-maker believes that several horses entered in the race have the following probability of winning:

Horse →	A	B	C	D	E
P(winning) →	0.04	0.42	0.31	0.12	0.11

He then receives a message that owing to a injury one of the horses is not participating in the race. Explain how would you assess from an Information Theory point of view the Information value of this message.

(1) If the horse in question is known.

(2) If it is not known.

NOTE: Info value of the msg when the horse in question is known is nothing but the self Info

Sol:-

1) * If the horse "A" is not running then the self Info of the message.

$$I_A = \log_2 \frac{1}{p(A)} = \log \frac{1}{0.04} = 4.644 \text{ bits}$$

* If the horse "B" is not running then.

$$I_B = \log_2 \frac{1}{p(B)} = \log_2 \frac{1}{0.42} = 1.252 \text{ bits}$$

* If the horse "C" is not running then.

$$I_C = \log_2 \frac{1}{p(C)} = \log_2 \frac{1}{0.31} = 1.69 \text{ bits}$$

* If the horse "D" is not winning then.

$$I_D = \log \frac{1}{p(D)} = \log \frac{1}{0.12} = 3.059 \text{ bits} //$$

* If the horse "E" is not winning then.

$$I_E = \log \frac{1}{p[E]} = \log \frac{1}{0.11} = 3.184 \text{ bits} //$$

2) The Info. value of the message when the horse in question is not known is nothing but the average Info is given by

$$H(s) = \sum_{i=1}^{q} p(i) \log_2 \frac{1}{p(i)} = \sum_{i=1}^{E} p(i) \log_2 \frac{1}{p(i)}$$

$$= p(A) \log_2 \frac{1}{p(A)} + p(a) \log_2 \frac{1}{p(a)} + p(c) \log_2 \frac{1}{p(a)} + p(D) \log_2 \frac{1}{p(D)} + p(E) \log_2 \frac{1}{p(E)}.$$

$$= 0.04 \log_2 \frac{1}{0.04} + 0.42 \log_2 \frac{1}{0.42} + 0.21 \log_2 \frac{1}{0.21} + 0.12 \log_2 \frac{1}{0.12} + 0.11 \log_2 \frac{1}{0.11}$$

$$H(s) = (0.04)(4.644) + (0.42)(1.252) + (0.21)(1.69) + (0.12)(3.059) + (0.11)(3.184)$$

$$= 1.953 \text{ bits / horse} //$$

EXTENSION OF ZERO MEMORY SOURCE.

Consider a zero memory source emitting two symbols S_1 and S_2 with probabilities P_1 and P_2 respectively. Obviously $P_1 + P_2 = 1$. Also the entropy of the source 'S' is given by.

$$H(S) = P_1 \log_2 \frac{1}{P_1} + P_2 \log_2 \frac{1}{P_2}.$$

Now consider the second-order extension of the source S. This source is denoted as S^2. Now the source S^2 will have four combinations viz. S_1S_1, S_1S_2, S_2S_1 and S_2S_2. Corresponding probabilities are given by

$$S_1S_1 = P(s_1)P(s_1) = P_1 \times P_1 = P_1^2.$$

$$S_1S_2 = P(s_1)P(s_2) = P_1 \times P_2 = P_1 \cdot P_2$$

$$S_2S_1 = P(s_2)P(s_1) = P_2 \times P_1 = P_2 \cdot P_1$$

$$S_2S_2 = P(s_2)P(s_2) = P_2 \times P_2 = P_2^2$$

∴ The entropy of the second-order extension of the source is given by

$$H(S^2) = \sum_{k=1}^{4} P_k \log_2 \frac{1}{P_k} = P_1^2 \log_2 \frac{1}{P_1^2} + P_1P_2 \log_2 \frac{1}{P_1P_2} + P_2P_1 \log_2 \frac{1}{P_2P_1} + P_2^2 \log_2 \frac{1}{P_2^2}.$$

$$= 2P_1^2 \log \frac{1}{P_1} + 2P_1P_2 \log \frac{1}{P_1P_2} + 2P_2^2 \log \frac{1}{P_2}$$

↳ P.T.O.

$$= 2P_1^2 \log\frac{1}{P_1} + 2P_1P_2 \log\frac{1}{P_1P_2} + 2P_2^2 \log\frac{1}{P_2}$$

$$= 2P_1^2 \log\frac{1}{P_1} + 2P_1P_2 \log\frac{1}{P_1} + 2P_1P_2 \log\frac{1}{P_2} + 2P_2^2 \log\frac{1}{P_2}$$

NOTE

$\log(A.B)$

$\log(1/A) + \log(1/B)$

$$= 2P_1(P_1+P_2)\log\frac{1}{P_1} + 2P_2(P_1+P_2)\log\frac{1}{P_2}$$

$$= 2\left[P_1 \log\frac{1}{P_1} + P_2 \log\frac{1}{P_2}\right](P_1+P_2)$$

$= 2H(s) \times 1$ where $(P_1+P_2)=1$

NOTE

When $A+B=1$

$P_1+P_2=1$.

$H(s^2) = 2.H(s)$ //

Similarly we can show that

$$H(s^3) = 3 \cdot H(s)$$

$$H(s^4) = 4 \cdot H(s)$$

and so on. In general for an n^{th}-order extension of 'S'

$$H(s^n) = n \cdot H(s)$$ //

* A zero memory source has a source alphabet $S = \{S_1, S_2\}$ with $P = \{1/2, 1/4, 1/4\}$. Find the entropy of this source. Also determine the entropy of its 2nd extension and verify that $H(S^2) = 2H(S)$

Sol:-

$$H(S) = \sum_{i=1}^{3} P_i \log \frac{1}{P_i}$$

$$= \frac{1}{2}\log(2) + \frac{1}{4}\log(4) + \frac{1}{4}\log(4)$$

$$= 1.5 \text{ bits/message symbol} \quad //$$

The 2nd extension of the basic source with 3 symbols will have $3^2 = 9$ symbols which can be listed along with their probability of occurrence as given below.

Notes
2nd extension of the source $= S^2$
$S_1 S_2 S_3$ (3 symbols)
$s = 3$
3^{rd} ext $= (3)^2 = 9$ //

The symbol $S_1 S_1$ occurs with probability $= (1/2)(1/2) = 1/4$ //

$S_1 S_2$ occurs with prob $= (1/2)(1/4) = 1/8$ //

$S_1 S_3$ occurs with probability $= (1/2)(1/4) = 1/8$ //

$S_2 S_1$ occurs with prob $= (1/4)(1/2) = 1/8$ //

$S_2 S_2$ occurs with prob $= (1/4)(1/4) = 1/16$ //

$S_2 S_3$ occurs with prob $= (1/4)(1/4) = 1/16$ //

$S_3 S_1$ occurs with prob $= (1/4)(1/2) = 1/8$ //

$S_3 S_2$ occurs with prob $= (1/4)(1/4) = 1/16$ //

$S_3 S_3$ occurs with prob $= (1/4)(1/4) = 1/16$ //

The entropy of the 2nd. Extended Source is given by

$$H(s^2) = \sum_{i=1}^{9} P_i \log \frac{1}{P_i}$$

$$= \frac{1}{4}\log_2(4) + 4\left(\frac{1}{8}\right)\log_2(8) + 4\left(\frac{1}{16}\right)\log_2(16)$$

$$= 3 \text{ bits/message symbol}$$

$$H(s^2) = 2H(s)$$

$\therefore 2H(s) = 2 \times 1.5$ bits/message symbol

$2H(s) = 3$ bits/message symbol

Finally, $H(s^2) = 2H(s)$ proved //

NOTE

2nd extn Source is given by

$$H(s^2) = \sum_{i=1}^{9} P_i \log\left(\frac{1}{P_i}\right)$$

(or)

$$H(s^2) = \sum_{K=1}^{N} P_K \log\left(\frac{1}{P_K}\right)$$

* A zero memory Source Contains.

$X = \{x_1, x_2, x_3, x_4\}$ with $P(X) = \{1/2, 1/4, 1/8, 1/8\}$.

1) Determine entropy of Source.

2) Determine 2nd extension of the Source.

S.T $H(X^2) = 2H(X)$

Jan/Feb-2002 [10M], July/Aug-2004, 2008 [8M]

Sol:- 1) Entropy $H(s)$, (or) $H(X)$ is given by

$$H(s) = H(X) = \sum_{K=1}^{M} P_K \log_2 \frac{1}{P_K} \text{ bits/Symbol.}$$

$$= P_1 \log\frac{1}{P_1} + P_2 \log_2\frac{1}{P_2} + P_3 \log_2\frac{1}{P_3} + P_4 \log_2\frac{1}{P_4}$$

$$= \frac{1}{2}\log_2(2) + \frac{1}{4}\log_2(4) + \frac{1}{8}\log_2(8) + \frac{1}{8}\log_2(8)$$

$$H(X) = 7/4 \text{ bits/symbol} //$$

∴ The 2nd. Extension of the basic Source with 4. Symbols, will have. $4^2 = 16$ Symbols //

$x_1 x_1$ occure with probability $= \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$ //

$x_1 x_2 = \frac{1}{2} \times \frac{1}{4} = \frac{1}{8}$ //

$x_1 x_3 = \frac{1}{2} \times \frac{1}{8} = \frac{1}{16}$ //

$x_1 x_4 = \frac{1}{2} \times \frac{1}{8} = \frac{1}{16}$ //

$x_2 x_1 = \frac{1}{4} \times \frac{1}{2} = \frac{1}{8}$ //

$x_2 x_2 = \frac{1}{4} \times \frac{1}{4} = \frac{1}{16}$ //

$x_2 x_3 = \frac{1}{4} \times \frac{1}{8} = \frac{1}{32}$ //

$x_2 x_4 = \frac{1}{4} \times \frac{1}{8} = \frac{1}{32}$ //

$x_3 x_1 = \frac{1}{8} \times \frac{1}{2} = \frac{1}{16}$ //

$x_3 x_2 = \frac{1}{8} \times \frac{1}{4} = \frac{1}{32}$ //

$x_3 x_3 = \frac{1}{8} \times \frac{1}{8} = \frac{1}{64}$ //

$x_3 x_4 = \frac{1}{8} \times \frac{1}{8} = \frac{1}{64}$ //

$x_4 x_1 = \frac{1}{8} \times \frac{1}{2} = \frac{1}{16}$ //

$x_4 x_2 = \frac{1}{8} \times \frac{1}{4} = \frac{1}{32}$ //

$x_4 x_3 = \frac{1}{8} \times \frac{1}{8} = \frac{1}{64}$ //

$x_4 x_4 = \frac{1}{8} \times \frac{1}{8} = \frac{1}{64}$ //

NOTE

2nd extn = Σ^2 (80)

$= 4^2$

2nd ext. = 16 //

$X = (x_1 x_2 x_3 x_4)$

4 symbols

$N = 4$.

$$H(X^2) = \sum_{K=1}^{N} P_K \log_2 \frac{1}{P_K} = \sum_{K=1}^{16} P_K \log_2 \left(\frac{1}{P_K}\right)$$

$= \frac{1}{4}\log_2(4) + (2)\left(\frac{1}{8}\right)\log_2(8) + (5)\left(\frac{1}{16}\right)\log_2(16) + (4)\left(\frac{1}{32}\right)\log_2(32) + (4)\left(\frac{1}{64}\right)\log_2(64)$

$= \frac{1}{4}(2) + (2)(\frac{1}{8})(3) + (5)(\frac{1}{16})(4) + (4)(\frac{1}{32})(5) + (4)(\frac{1}{64})(6)$

$H(X^2) = 3.5$ bits/symbol. (or) $\frac{7}{2}$ bits/symbol //

$\therefore \quad H(X^2) = 2H(X)$ //

$2H(X) = 2 \times \frac{7}{4} = 7/2$,

$H(X^2) = 2H(X)$ proved. //

IM.

* An analog signal is band limited to 500Hz and is sampled at "Nyquist Rate" The samples are quantized into 4 levels. The quantization levels are assumed to be independent and occur with probabilities $P_1 = P_2 = \frac{1}{8}$, $P_2 = P_3 = \frac{3}{8}$. Find the Information rate of the source.

Sol:- $H(S) = \sum_{i=1}^{4} P_i \log \frac{1}{P_i} = (\frac{1}{8}\log 8)(2) + (\frac{3}{8})\log(\frac{8}{3})(2)$

$= 1.8113$ bits/symbol //

The Symbol rate r_s is given by

$r_s = 2B = 2 \times 500 = 1000$ symbol/sec //

P.T.O

∴ Information rate. $R_s = r_s H(S)$

$= (1000)(1.81)$

$= 1810$ bits/sec //

* Consider a discrete memoryless source with alphabet $S = \{s_0, s_1, s_2\}$ with source statistics $\{0.7, 0.15, 0.15\}$.
(a) Calculate the entropy of source.
(b) Calculate the entropy of the 2nd order extension of the source

Sol:(a) $H(S) = \sum_{i=0}^{2} p_i \log_2 \frac{1}{p_i}$

$H(S) = 0.7 \log_2 \frac{1}{0.7} + 0.15 \log_2 \frac{1}{0.15} + 0.15 \log_2 \frac{1}{0.5}$

$= 0.3602 + 0.41054 + 0.41054$

$= 1.18128$ bits/symbol //

(b). The 2nd extension of the basic source with 3 symbols will have $3^2 = 9$ symbols //

$S_1S_1 = (0.7)(0.7) = 0.49$ //
$S_1S_2 = (0.7)(0.15) = 0.105$ //
$S_1S_3 = (0.7)(0.15) = 0.105$ //
$S_2S_1 = (0.15)(0.7) = 0.105$ //
$S_2S_2 = (0.15)(0.15) = 0.0225$ //

$S_2 S_3 = (0.15)(0.15) = 0.0225$ //

$S_3 S_1 = (0.15)(0.7) = 0.105$ //

$S_1 S_2 = (0.15)(0.15) = 0.0225$ //

$S_3 S_3 = (0.15)(0.15) = 0.0225$ //

$$H(S^2) = \sum_{i=1}^{9} P_i \log_2 \frac{1}{P_i}$$

$$= (0.49)\log_2 \frac{1}{(0.49)} + (4)(0.105)\log_2 \frac{1}{(0.105)} + (4)(0.0225)\log_2 \frac{1}{(0.0225)}$$

$$= 2.36256 \text{ bits/symbol} //$$

$\therefore$ $H(S^2) = 2H(S)$

$2H(S) = 2 \times 1.8128$ bits/symbol

$2H(S) = 2.36256$ bits/symbol

Hence equally $H(S^2) = 2H(S)$ proved //

***

* A Source Emits one of the four possible messages M_1 M_2 M_3 & M_4 with probabilities of $\frac{7}{16}$, $\frac{5}{16}$, $\frac{1}{8}$ and $\frac{1}{8}$ respectively. Find the Entropy of the Source. List all the elements for the 2nd. Extension of this Source. Hence show $H(S^2) = 2H(S)$

Sol:- $S = \{M_1, M_2, M_3, M_4\}$

$P = \{7/16, \frac{5}{16}, 1/8, 1/8\}$.

Entropy $H(S) = \sum_{i=1}^{4} p(M_i) \log_2 \frac{1}{p(M_i)}$ (or) $\sum_{i=1}^{4} P_i \log_2 \frac{1}{P_i}$

$= \frac{7}{16} \log_2 \frac{16}{7} + \frac{5}{16} \log_2 \frac{16}{5} + \frac{1}{8} \log_2 (8) (2)$.

$= 1.7962$ bits/ Symbol //

The 2nd. Extension S^2 of the basic Source will have $4^2 = 16$ Symbols.

$M_1M_1 = (7/16)(7/16) = 49/256$ //

$M_1M_2 = (7/16)(5/16) = \frac{35}{256}$ //

$M_1M_3 = (7/16)(1/8) = \frac{7}{128}$ //

$M_1M_4 = (7/16)(1/8) = \frac{7}{128}$ //

$M_2M_1 = (5/16)(7/16) = \frac{35}{256}$ //

$M_2M_2 = (5/16)(5/16) = \frac{25}{256}$ //

$M_2M_3 = (5/16)(1/8) = \frac{5}{128}$ //

$M_2M_4 = (5/16)(1/8) = \frac{5}{128}$ //.

$M_3M_1 = (1/8)(7/16) = 7/128$ // $M_4M_1 = (1/8)(7/16) = 7/128$ //

$M_3M_2 = (1/8)(5/16) = 5/128$ // $M_4M_2 = (1/8)(5/16) = 5/128$ //

$M_3M_3 = (1/8)(1/8) = 1/64$ // $M_4M_3 = (1/8)(1/8) = 1/64$ //

$M_3M_4 = (1/8)(1/8) = 1/64$ // $M_4M_4 = (1/8)(1/8) = 1/64$ //

The Entropy of the 2nd extended Source is given by

$$H(S^2) = \sum_{i=1}^{16} P_i \log_2 \frac{1}{P_i}$$

$$= \frac{49}{256}\log_2\frac{256}{49} + \left(\frac{35}{256}\right)(2)\log_2\left(\frac{256}{35}\right) +$$

$$\frac{25}{256}\log_2\left(\frac{256}{25}\right) + (4)\left(\frac{7}{128}\right)\log_2\left(\frac{128}{7}\right) + (4)\left(\frac{5}{128}\right)\log_2\left(\frac{128}{5}\right) +$$

$$(4)(1/64)\log_2(64)$$

$= 3.5924$ bits/message Symbol //

$H(S)^2 = 2H(S)$

$2H(S) = 2 \times 1.7962$ bits/message Symbol

$= 3.5924$ bits/message Symbol //

$H(S^2) = 2H(S)$ proved //

* Find. The entropy of a source in nats/symbol of a source that emits one out of four symbols A, B, C and D in a statistically Independent sequence with proba. $1/2$, $1/4$, $1/8$ and $\frac{1}{8}$.

Sol :- $H(s) = \sum_{i=A}^{D} P_i \log_2 \frac{1}{P_i} = \sum_{i=1}^{4} P_i \log_2 \frac{1}{P_i}$

$= 1/2 \log_2(2) + \frac{1}{4}\log_2(4) + \frac{1}{8}\log_2(8)\,(2)$.

$H(s) = 1/2 + \frac{1}{4}(2) + \frac{1}{8}(3)\,(2) = 1.75$ bits/symbol //

$\therefore$ 1 nat = 1.443 bits.

1 bit = $\frac{1}{1.443}$ nat = 0.693 nats //.

Entropy of Source $H(s) = (1.75)(0.693$ nats/symbol)

$= 1.213$ nats/symbol //

OR

$$H(s) = \sum_{i=1}^{4} P_i \log_e \frac{1}{P_i} \text{ nats/symbol}$$

$= 1/2 \log_e(2) + (1/4)\log_e(4) + (2)(1/8)\log_e(8)$.

$= 1.213$ nats/symbol //

MARKOFF STATISTICAL MODEL FOR INFORMATION SOURCES

July/Aug - 2005, Jan/Feb - 2008.

In Real life Sources, there is interdependence present such that.

"The occurrence of X_i in the zeroth position of the message depends on the previous 'q' symbols $\{S_1, S_2 \dots S_q\}$ of the source are generally specified by set of Condition probabilities.

$$P\{X_i | S_1, S_2 \dots S_q\}$$

where $X_i \rightarrow$ Symbol in the 0^{th} position. Each 'S' has an 'M' symbol alphabet $\{X_1, X_2, \dots X_m\}$. $\{S_1, S_2, \dots S_q\}$ are the previous 'q' symbols emitted. Since $P(X_i)$ depends on the earlier 'q' symbols. Such a source is known as q^{th} order of "markoff source" (or) "markov source".

ENTROPY AND INFORMATION RATE } Jan/Feb 2007

* The entropy of the source is defined as the weighted average of the entropy of the symbols emitted from each state, where the entropy of state "i" denoted by H_i is defined as the average Info Content of the symbol emitted from ith state.

$$\therefore H_i = \sum_{j=1}^{n} P_{ij} \log \frac{1}{P_{ij}} \text{ bits/msg-symbol} //$$

* The entropy of the source is then the average of entropy of each state i.e.

$$H = \sum_{i=1}^{n} P_i H_i = \sum_{i=1}^{n} P_i \left[\sum_{j=1}^{n} P_{ij} \log \frac{1}{P_{ij}} \right] \text{ bits/msg symbol}$$

where. n is the total number of states.

P_i is the probability of the ith state.

* The average Information rate R_s for the source is given by.

$$R_s = r_s H \text{ bits/sec}$$

where. r_s is the number of state transitions per second. (or) the Symbol rate of the source.

Average Information per symbol :-

If $p(M_i)$ is the probability of a sequence M_i of "N" Symbols from the Source and if

$$G_N = \frac{1}{N} \sum_i p(M_i) \log_2 \frac{1}{p(M_i)}$$

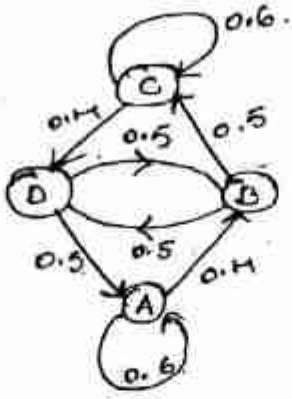
$$= \frac{1}{N} H(\bar{s}^N)$$

where $H(\bar{s})$ is the Entropy of adjoint Source.

where the Sum is over all Sequence M_i containing "N" Symbols. Then G_N is monotonically decreasing function of "N" and

$$\lim_{N \to \infty} G_N = H \text{ bits/sec} \quad //.$$

* Consider the state diagram of the markov source of the below fig.



1) Compute the state probabilities.
2) Find Entropy of each state
3) Find the Entropy of the source.

Fig:- Markov Source.

Sol:- From the state diagram, the state Equations can be immediately written as.

1)
$$p(A) = 0.6p(A) + 0.5p(D) \quad -(1)$$
$$p(B) = 0.4p(A) + 0.5p(D) \quad -(2)$$
$$p(C) = 0.6p(C) + 0.5p(B) \quad -(3)$$
$$p(D) = 0.4p(C) + 0.5p(B) \quad -(4)$$

From Eq (1).

$$p(A) = 0.6p(A) + 0.5p(D)$$
$$p(A) - 0.6p(A) = 0.5p(D)$$
$$p(A)(1-0.6) = 0.5p(D)$$
$$p(A)(0.4) = 0.5p(D)$$
$$p(A) = 1.25p(D) \quad -(a)$$

//

Consider eq (2)

$$p(B) = 0.4\,p(A) + 0.5\,p(B)$$

Substituting eq (2) into eq (3)

$$p(B) = 0.4\,(1.25)\,p(B) + 0.5\,p(B)$$

$$p(B) = 0.5\,p(B) + 0.5\,p(B)$$

$$p(B) = 1\,p(B) \quad \text{//} \longrightarrow (4b)$$

From eq (3) $p(C) = 0.6\,p(C) + 0.5\,p(B)$

$$p(C) - 0.6\,p(C) = 0.5\,p(B)$$

$$p(C)\,0.4 = 0.5\,p(B)$$

$$p(C) = \frac{0.5}{0.4}\,p(B)$$

$$p(C) = 1.25\,p(B) \quad \text{//}$$

Substituting eq (4b) into eq (1) $\longrightarrow$ $p(C) = 1.25\,(1)\,p(B) = 1.25\,p(B)$ (4c)

But $p(A) + p(B) + p(C) + p(D) = 1$ $\longrightarrow$ (5)

eq (a) & (b) & (c) Subst into eq (5)

$$1.25\,p(B) + 1\,p(B) + 1.25\,p(B) + p(B) = 1$$

$$4.5\,p(B) = 1$$

$$p(B) = \frac{1}{4.5}$$

$$= 2/9 \quad \text{//}$$

Consider a Eq (a)

$$p(A) = 1.25\,p(D)$$

$$p(A) = 1.25\,(2/9) = \frac{5}{18}$$ //

Consider a eq (b)

$$\therefore\ p(B) = p(D)$$

$$p(B) = \frac{2}{9}$$ //

Considering a eq (c)

$$p(C) = 1.25\,p(D)$$

$$p(C) = 1.25\,(2/9) = \frac{5}{18}$$ //

∴ the probabilities are $p(A) = p(C) = \frac{5}{18}$ // $p(B) = p(D) = \frac{2}{9}$ //

2) The Entropy of each State is.

$$H = \sum_{i=1}^{n} P_{ij} \log_2 \frac{1}{P_{ij}}$$

For 1st State.

$$H_A = P_{AA} \log_2 \frac{1}{P_{AA}} + P_{AB} \log_2 \frac{1}{P_{AB}} + P_{AC} \log_2 \frac{1}{P_{AC}} + P_{AD} \log_2 \frac{1}{P_{AD}}$$

$$= 0.6 \log_2 \frac{1}{0.6} + 0.4 \log_2 \frac{1}{0.4} + 0 + 0$$

$$= 0.6 \log_2 (1.666) + 0.4 \log_2 (2.5)$$

$$= 0.6 \frac{\log(1.666)}{\log(2)} + 0.4 \frac{\log(2.5)}{\log(2)}$$

$$= 0.4418 + 0.32877$$

$$= 0.971 \text{ bits/state.}$$ (or) Average - Symbol //

By State 2.

$$H_B = P_{BA} \log \frac{1}{P_{BA}} + P_{BB} \log \frac{1}{P_{BB}} + P_{BC} \log \frac{1}{P_{BC}} + P_{BD} \log \frac{1}{P_{BD}}$$

$$= 0 + 0 + 0.5 \log_2 \frac{1}{0.5} + 0.5 \log_2 \frac{1}{0.5}$$

$$= 1 \text{ bit/state} //$$

By State 3

$$H_C = P_{CA} \log \frac{1}{P_{CA}} + P_{CB} \log \frac{1}{P_{CB}} + P_{CC} \log \frac{1}{P_{CC}} + P_{CD} \log \frac{1}{P_{CD}}$$

$$= 0 + 0 + 0.6 \log_2 \frac{1}{0.6} + 0.4 \log_2 \frac{1}{0.4}$$

$$= 0.971 \text{ bits/state} //$$

By State 4.

$$H_D = P_{DA} \log \frac{1}{P_{DA}} + P_{DB} \log \frac{1}{P_{DB}} + P_{DC} \log \frac{1}{P_{DC}} + P_{DD} \log \frac{1}{P_{DD}}$$

$$= 0.5 \log_2 \frac{1}{0.5} + 0.5 \log_2 \frac{1}{0.5} + 0 + 0$$

$$= 1 \text{ bit/state} //$$

3) The Entropy of the Source $S = \{A, B, C, D\}$ is Calculated. using eq as.

$$H(s) = -H = \sum_{i=1}^{n} P_i H_i$$

$$= P(A) H_A + P(B) H_B + P(C) H_C + P(D) H_D$$

$$= \left(\frac{5}{18}\right)(0.971) + \left(\frac{2}{9}\right)(1) + \left(\frac{5}{18}\right)(0.971) + \left(\frac{2}{9}\right)(1)$$

$H = 0.9839$ bits/binary digit. //

* Consider the State diagram of markov source of the below fig. DEC-2010 [8M]

1) State probabilities
2) Find Entropy of each state.
3) Find the Entropy of the source.

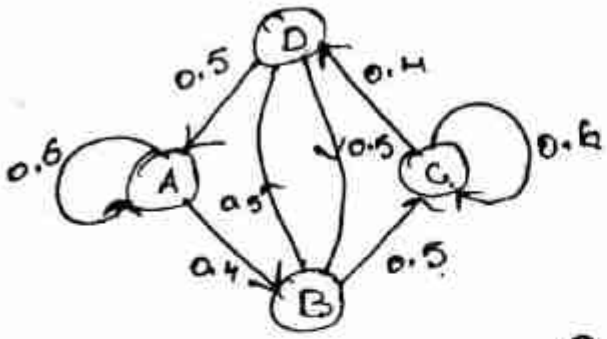


Fig :- markov source.

Sol :- From the State diagram the State Equations can be immediately written. as.

$$p(A) = 0.6\,p(A) + 0.5\,p(D) \quad \text{— (1)}$$

$$p(B) = 0.4\,p(A) + 0.5\,p(D) \quad \text{— (2)}$$

$$p(C) = 0.6\,p(C) + 0.5\,p(B) \quad \text{— (3)}$$

$$p(D) = 0.4\,p(C) + 0.5\,p(B) \quad \text{— (4)}$$

↑ ↑ ↑ ↑

NOTE → See Similar to the previous problem.

* * *

Consider the State diagram of the markov source of the below fig.

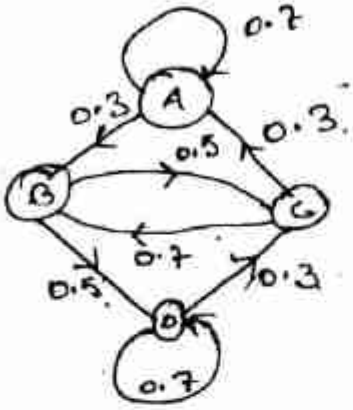


Fig: markov source.

find.

1) Compute the state probabilities

2) find Entropy of Each state.

3) find the entropy of the source.

Sol :-

1) $p(A) = 0.7p(A) + 0.3p(C)$ — (1)

$p(B) = 0.3p(A) + 0.7p(C)$ — (2)

$p(C) = 0.5p(B) + 0.3p(D)$ — (3)

$p(D) = 0.5p(B) + 0.7p(D)$ — (4)

from eq (1)

$p(A) = 0.7p(A) + 0.3p(C)$

$p(A) - 0.7p(A) = 0.3p(C)$

$p(A)(0.6) = 0.3p(C)$

$p(A) = \frac{0.3}{0.6} p(C) = 0.5p(C)$ — (a)

//

Consider eq (2)

$$p(B) = 0.3\,p(A) + 0.7\,p(C)$$

Sub eq (a) into eq (2)

$$p(B) = 0.3\,(0.5)\,p(C) + 0.7\,p(C)$$

$$p(B) = 0.15\,p(C) + 0.7\,p(C)$$

$$p(B) = 0.85\,p(C) \quad //. \quad \text{——— (b)}$$

From eq (4)

$$p(D) = 0.5\,p(B) + 0.7\,p(D)$$

$$p(D) - 0.7\,p(D) = 0.5\,p(B)$$

$$p(D)\,(0.3) = 0.5\,p(B)$$

$$p(D) = \frac{0.5}{0.3}\,p(B) = 1.666\,p(B)$$

using eq (b)

$$p(D) = (1.666)\,(0.85)\,p(C)$$

$$p(D) = 1.4161\,p(C) \quad \text{——— (c)}$$

But $p(A) + p(B) + p(C) + p(D) = 1$ ——— (5)

eq (a) & (b) & (c) Subs into eq (5)

$$0.5\,p(C) + 0.85\,p(C) + p(C) + 1.4161\,p(C) = 1$$

$$3.7661\, p(c) = 1$$

$$p(c) = \frac{1}{3.7661} = 0.26552 //$$

Consider a eq (a)

$$\therefore \quad p(A) = 0.5\, p(c)$$
$$= 0.5\,(0.26552)$$
$$= 0.13276 //$$

Consider a eq (b)

$$p(B) = 0.85\, p(c)$$
$$= 0.85\,(0.26552)$$
$$= 0.225692 //$$

Consider a eq (c)

$$p(D) = 1.4161\, p(c)$$
$$= 1.4161\,(0.26552)$$
$$= 0.37600 //$$

$\therefore$ The probabilities are $p(A) = 0.13276$, $p(B) = 0.225692$, $p(C) = 0.26552$, $p(D) = 0.37600$ //

2) The entropy of each state is

$$H = \sum_{i=1}^{n} P_i \log_2 \frac{1}{P_i}$$

For 1st state.

$$H_A = P_{AA} \log_2 \frac{1}{P_{AA}} + P_{AB} \log_2 \frac{1}{P_{AB}} + P_{AC} \log_2 \frac{1}{P_{AC}} + P_{AD} \log_2 \frac{1}{P_{AD}}$$

$$= 0.7 \log_2 \frac{1}{0.7} + 0.3 \log_2 \frac{1}{0.3} + 0 + 0$$

$$= 0.7 \log_2 (1.4285) + 0.3 \log_2 (3.333)$$

$$= 0.7 \frac{\log (1.4285)}{\log (2)} + 0.3 \frac{\log (3.333)}{\log (2)}$$

$$= 0.7 (0.5145) + 0.3 (1.73682)$$

$$= 0.88119 \text{ bits/sec}$$

State 2 :-

$$H_B = P_{BA} \log \frac{1}{P_{BA}} + P_{BB} \log \frac{1}{P_{BB}} + P_{BC} \log \frac{1}{P_{BC}} + P_{BB} \log \frac{1}{P_{BB}}$$

$$= 0 + 0 + 0.5 \log \frac{1}{0.5} + 0.5 \log \frac{1}{0.5}$$

$$= 1 \text{ bits/sec}$$

State 3 :

$$H_C = P_{CA} \log \frac{1}{P_{CA}} + P_{CB} \log \frac{1}{P_{CB}} + P_{CC} \log \frac{1}{P_{CC}} + P_{CD} \log \frac{1}{P_{CD}}$$

$$= 0.3 \log \frac{1}{0.3} + 0.7 \log \frac{1}{0.7} + 0 + 0$$

$$= 0.3 \frac{\log(3.33)}{\log(2)} + 0.7 \frac{\log(1.4285)}{\log(2)}$$

$$= 0.5206 + 0.3601$$

$$= 0.8807 \text{ bits/sec}$$

State 4.

$$H_D = P_{DA} \log \frac{1}{P_{DA}} + P_{DB} \log \frac{1}{P_{DB}} + P_{DC} \log \frac{1}{P_{DC}} + P_{DD} \log \frac{1}{P_{DD}}$$

$$= 0 + 0 + 0.3 \log \frac{1}{0.3} + 0.7 \log \frac{1}{0.7}$$

$$= 0 + 0 + 0.3 \frac{\log(3.33)}{\log(2)} + 0.7 \frac{\log(1.4285)}{\log(2)}$$

$$= 0.8807 \text{ bits/sec}$$

3) The entropy of the source $S = \{A, B, C, D\}$ is calculated. using eq 2.9.

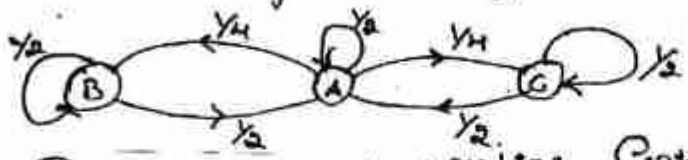
$$H(S) = H = \sum_{i=1}^{n} p_i H_i$$

$$= p(A) H_A + p(B) H_B + p(C) H_C + p(D) H_D$$

$$= (0.13276)(0.88119) + (0.22569)(1) + (0.26552)(0.8807) + (0.37600)(0.8807)$$

$$= 0.9076 \text{ bits/binary digits}$$

* Give the model of a markoff source fig



Find the state probabilities, Entropy of each state. and the Entropy of the source. DEC-2012 [10M]

Sol:- The state probabilities can be calculated from the state diagram as below.

for state A → $p(A) = \frac{1}{2}p(A) + \frac{1}{2}p(B) + \frac{1}{2}p(C)$ ——— (1)

B → $p(B) = \frac{1}{4}p(A) + \frac{1}{2}p(B)$ ——— (2)

C → $p(C) = \frac{1}{2}p(C) + \frac{1}{4}p(A)$ ——— (3)

From eq (1)

$p(A) = \frac{1}{2}p(A) + \frac{1}{2}p(B) + \frac{1}{2}p(C)$

$= \frac{1}{2}[p(A) + p(B) + p(C)]$

$= \frac{1}{2}[1] = \frac{1}{2}$ // ——— (a)

NOTE: sum of all probabilities equal to one. $p(A)+p(B)+p(C)=1$

from eq (2) $p(B) = \frac{1}{4}p(A) + \frac{1}{2}p(B)$

∴ eq (a) sub into eq (2)

$p(B) - \frac{1}{2}p(B) = \frac{1}{4}(\frac{1}{2})$

$\frac{1}{2}p(B) = \frac{1}{4}(\frac{1}{2})$

$p(B) = \frac{1}{4}$ //

from eq (3) $p(c) = \frac{1}{2}p(c) + \frac{1}{4}p(A)$.

$p(c) - \frac{1}{2}p(c) = \frac{1}{4}p(A)$

Using eq (4)

$p(c)(0.5) = \frac{1}{4}(\frac{1}{2})$

$p(c) = \frac{1}{4}$ //

$\therefore$ the probabilities are $p(A) = \frac{1}{2}$, $p(B) = p(c) = \frac{1}{4}$ //

The Entropy of each State is

$H_A = P_{AA}\log\frac{1}{P_{AA}} + P_{AB}\log\frac{1}{P_{AB}} + P_{AC}\log\frac{1}{P_{AC}}$.

$= \frac{1}{2}\log 2 + \frac{1}{4}\log 4 + \frac{1}{4}\log 4$

$= 1.5$ bits/message-Symbol (or) state //

$H_B = P_{BA}\log\frac{1}{P_{BA}} + P_{BB}\log\frac{1}{P_{BB}} + P_{BC}\log\frac{1}{P_{BC}}$.

$= \frac{1}{2}\log 2 + \frac{1}{2}\log 2 + 0$.

$= 1$ bit/message-Symbol //

$H_C = P_{CA}\log\frac{1}{P_{CA}} + P_{CB}\log\frac{1}{P_{CB}} + P_{CC}\log\frac{1}{P_{CC}}$

$= \frac{1}{2}\log 2 + 0 + \frac{1}{2}\log 2$.

$= 1$ bit/message-Symbol //

The entropy of the source is calculated as.

$$H = \sum_{i=1}^{n} P_i H_i$$

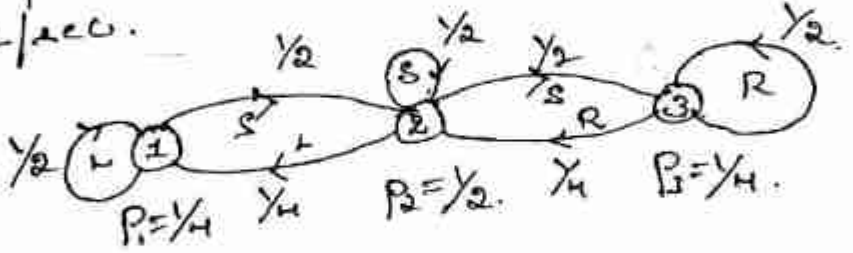
$$= P(A) H_A + P(B) H_B + P(C) H_C$$

$$= (1/2)(1.5) + (1/4)(1) + (1/4)(1)$$

$$= 1.25 \text{ bits/message-Symbol}$$

*** June/July-2013 [10M]

* The state diagram of stationary markoff source is shown in fig below. find the average information rate for the source if symbol rate of source is 1/sec.



Sol:- Given:- $P_1 = 1/4$, $P_2 = 1/2$, $P_3 = 1/4$.

The entropy of each state is.

$$\therefore H_1 = P_{11} \log \frac{1}{P_{11}} + P_{12} \log \frac{1}{P_{12}} + P_{13} \log \frac{1}{P_{13}}$$

$$= 1/2 \log(2) + 1/2 \log(2) + 0$$

$$= 1 \text{ bit/message Symbol}$$

$$H_2 = P_{21}\log\frac{1}{P_{21}} + P_{22}\log\frac{1}{P_{22}} + P_{23}\log\frac{1}{P_{23}}$$

$$= \frac{1}{4}\log(4) + \frac{1}{2}\log(2) + \frac{1}{2}\log(2)$$

$$= \frac{1}{4}\frac{\log(4)}{\log(2)} + \frac{1}{2}\frac{\log(2)}{\log(2)} + \frac{1}{2}\frac{\log(2)}{\log(2)}$$

$$= \frac{1}{4}(2) + \frac{1}{2}(1) + \frac{1}{2}(1)$$

$$= 1.5 \text{ bits/message-symbol} //$$

$$H_3 = P_{31}\log\frac{1}{P_{31}} + P_{32}\log\frac{1}{P_{32}} + P_{33}\log\frac{1}{P_{33}}$$

$$= 0 + \frac{1}{4}\log(4) + \frac{1}{2}\log(2)$$

$$= \frac{1}{4}(2) + \frac{1}{2}(1)$$

$$= 1 \text{ bit/message-Symbol} //$$

The entropy of the source is calculated as.

$$H = \sum_{i=1}^{n} P_i H_i = P_1 H_1 + P_2 H_2 + P_3 H_3$$

$$= \frac{1}{4}(1) + \frac{1}{2}(1.5) + \frac{1}{4}(1) = 1.25 \text{ bits/message symbol}$$

The average rate of Information is given by

$$R_s = r.H$$

$$= 1/\text{sec} \times 1.25 \text{ bits/message Symbol}$$

$$= 1.25 \text{ bits/sec} //$$

* The state diagram of a markoff source is shown in fig.

1) Find the Entropy "H" of the source.

2) Find G_1, G_2 and G_3 and verify that $G_1 > G_2 > G_3 > H$.

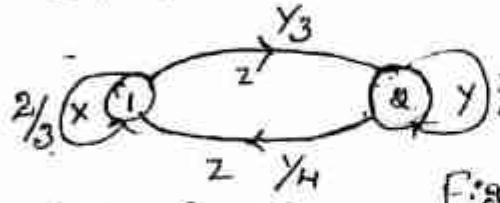


Fig:- markoff source.

DEC-2013/JAN-2014. [10M]

Sol:- 1)

$$p(1) = \frac{2}{3}p(1) + \frac{1}{4}p(2) \quad \text{— (1)}$$

$$p(2) = \frac{3}{4}p(2) + \frac{1}{3}p(1) \quad \text{— (2)}$$

From eq (1) $\quad p(1) - \frac{2}{3}p(1) = \frac{1}{4}p(2)$

$$\frac{1}{3}p(1) = \frac{1}{4}p(2)$$

$$p(1) = \frac{\frac{1}{4}}{\frac{1}{3}} = p(2) = \frac{3}{4}p(2) \quad \text{— (3)}$$

Consider $\quad p(1) + p(2) = 1 \quad \text{— (4)}$

eq (3) Substitute eq (4) $\quad \frac{3}{4}p(2) + p(2) = 1$

$$p(2)\frac{7}{4} = 1$$

$$p(2) = 4/7 \quad \text{— (5)}$$

Consider eq (5) Substitute into eq (3)

$$p(1) = \frac{3}{4} \times 4/7 = 3/7$$

$\therefore p(1) = 3/7 \quad p(2) = 4/7$

The entropy of each state is given by

1st State:-

$$H_i = \sum_{j=1}^{n} P_{ij} \log \frac{1}{P_{ij}}$$

$$H_1 = \sum_{j=1}^{2} P_{1j} \log \frac{1}{P_{1j}}$$

$$= P_{11} \log \frac{1}{P_{11}} + P_{12} \log \frac{1}{P_{12}}$$

$$= 2/3 \log 3/2 + 1/3 \log 3$$

$$= 0.9183 \text{ bits/ Symbol} //$$

2nd State:-

$$H_2 = \sum_{j=1}^{2} P_{2j} \log \frac{1}{P_{2j}}$$

$$= P_{21} \log \frac{1}{P_{21}} + P_{22} \log \frac{1}{P_{22}}$$

$$= 1/4 \log 4 + 3/4 \log 4/3$$

$$= 0.8113 \text{ bits/ msg Symbol} //$$

The Entropy of the Source is given by

$$H = \sum_{i=1}^{n} P_i H_i = \sum_{i=1}^{n} P(i) H_i$$

$$= P(1) H_1 + P(2) H_2$$

$$= (3/4)(0.9183) + (1/4)(0.8113)$$

$$= 0.8572 \text{ bits/msg Symbol} //$$

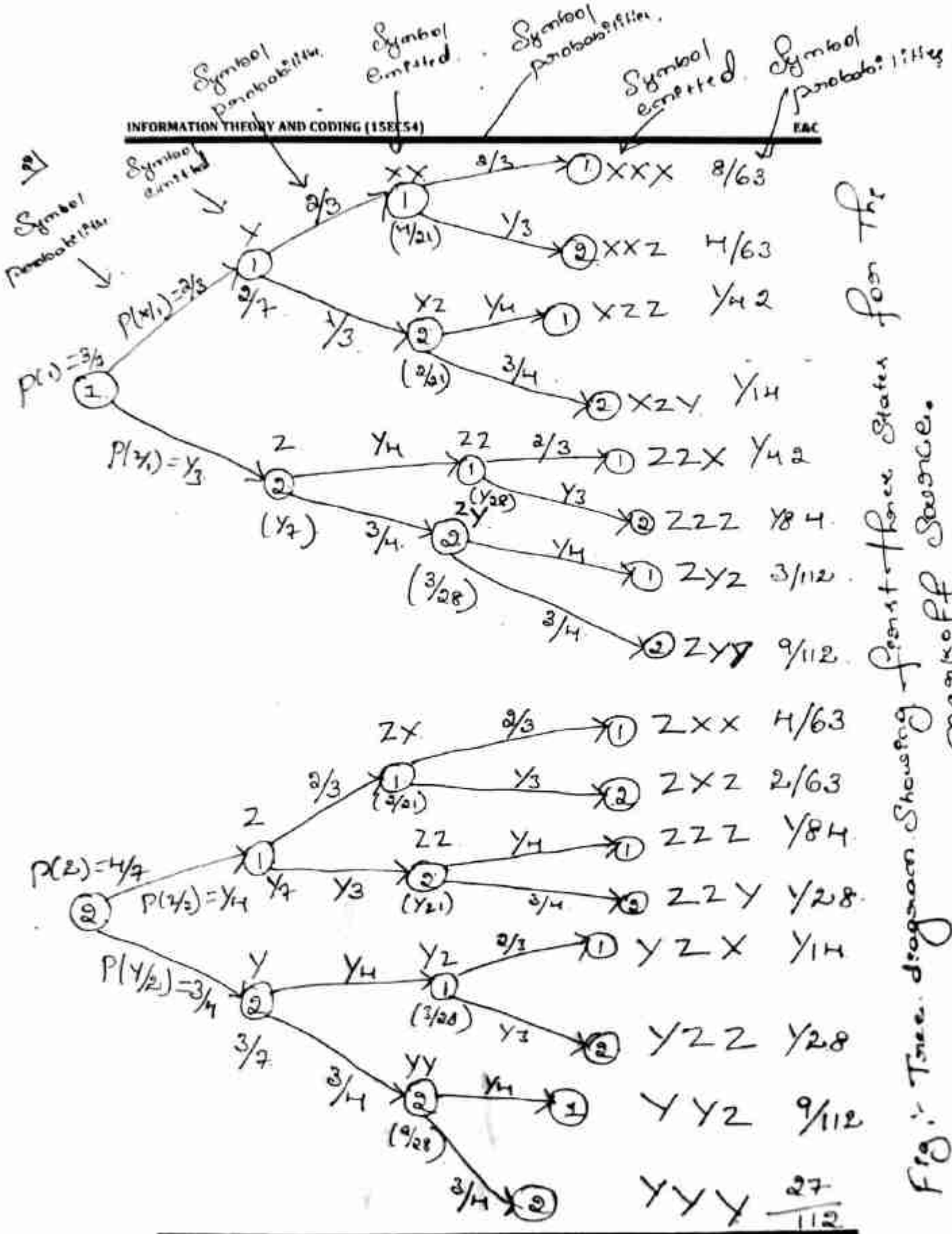


Fig: Tree diagram showing symbols, their probabilities and messages for the second extension of the source.

* The Symbols produced by the source will have an alphabet given by $S = \{X, Y, Z\}$ and the corresponding probabilities at the end of first Symbol Interval is calculated as

$$P(X) = \left(\frac{3}{7}\right) \times \left(\frac{2}{3}\right) = \frac{1}{7} //$$

$$P(Y) = \left(\frac{4}{7}\right) \times \left(\frac{3}{4}\right) = \frac{3}{7} //$$

$$P(Z) = \frac{7}{3} \times \frac{1}{3} + \frac{4}{7} \times \frac{1}{4} = \frac{2}{7} //$$

$$P(S) = \left\{\frac{2}{7}, \frac{3}{7}, \frac{1}{7}\right\}$$

$$G_N = \frac{1}{N} \sum_{i=1}^{3} P(m_i) \log \frac{1}{P(m_i)}$$

for $N = 1$ we have. $G_N = H(S) = \sum_{i=1}^{3} P(m_i) \log \frac{1}{P(m_i)}$

$$G_1 = H(S) = \frac{2}{7} \log 7/2 + 3/7 \log 7/3 + 2/7 \log 7/2$$

$$= 1.557 \text{ bits/ msg Symbol} //$$

* The Symbol produced by the source at the end of Second Symbol Interval will have an alphabet given by (note that ZZ is repeated)

$$S^2 = \{XX, XZ, ZZ, ZY, ZX, YZ, YY\}$$

The corresponding probabilities are calculated as

$$P(XX) = (3/7)(2/3)(2/3) = 4/21 //$$

$$P(XZ) = (3/7)(2/3)(1/3) = 2/21 //$$

$P(zz) = (3/7)(3/3)(3/4) + (4/7)(3/4)(3/3) = \frac{1}{28} + \frac{1}{21} = \frac{1}{12}$ //

$P(zy) = (3/7)(3/3)(3/4) = 3/28$ //

$P(zx) = (4/7)(3/4)(2/3) = 2/21$ //

$P(yz) = (4/7)(3/4)(3/4) = 3/28$ //

$P(yy) = (4/7)(3/4)(3/4) = 9/28$ //

$\therefore P(\bar{S}^2) = \{4/21, 2/21, 1/21, 3/28, 2/21, 3/28, 9/28\}$

$\therefore G_2 = H(\bar{S}^2) = \sum_{i=1}^{7} P(m_i) \log \frac{1}{P(m_i)}$

$H(\bar{S}^2) = \frac{4}{21}\log\frac{21}{4} + \left(\frac{2}{21}\log\frac{21}{2}\right)(2) + \frac{1}{12}\log 12 + \left(\frac{3}{28}\log\frac{28}{3}\right)(2) + \frac{9}{28}\log\frac{28}{9}$

$H(\bar{S}^2) = 2.6174$ bits/message symbol //

$N = 2$ // $G_2 = \frac{H(\bar{S}^N)}{N}$ $G_2 = \frac{H(\bar{S}^2)}{2} = \frac{2.6174}{2}$

$G_2 = 1.3087$ bits/ message symbol //

The Symbols produced by the source at the end of 3rd Symbol Interval will have an alphabet given by.
(NOTE that ZZZ is repeated)

$S^3 = \{XXX, XX2, X22, X2Y, 22X, 222, 2Y2, 2YY, 2XX, 2X2, 22Y, Y2Y, YY2, YYY\}$.

$P(XXX) = (3/7)(2/3)(2/3)(0/3) = \frac{8}{63}$ //

$P(XX2) = (3/7)(2/3)(2/3)(1/3) = 4/63$ //

$P(X22) = (3/7)(2/3)(1/3)(1/4) = \frac{1}{42}$ //

$P(X2Y) = (3/7)(2/3)(1/3)(3/4) = \frac{1}{14}$ //

$P(22X) = (3/7)(1/3)(1/4)(2/3) = \frac{1}{42}$ //

$P(222) = (3/7)(1/3)(1/4)(1/3) + (4/7)(1/4)(1/3)(1/4)$

$P(2Y2) = (3/7)(1/3)(3/4)(1/4) = \frac{3}{112}$ //

$P(2YY) = (3/7)(1/3)(3/4)(3/4) = \frac{9}{112}$ //

$P(2YX) = (4/7)(1/4)(2/3)(2/3) = 4/63$ //

$P(2X2) = (4/7)(1/4)(2/3)(1/3) = 2/63$ //

$P(22Y) = (4/7)(1/4)(1/3)(3/4) = 1/28$ //

$P(Y2X) = (4/7)(3/4)(1/4)(2/3) = 1/14$ //

$P(Y22) = (4/7)(3/4)(1/4)(1/3) = \frac{1}{28}$ //

$P(Y\,Y\,2) = (4/7)(3/4)(3/4)(1/4) = \frac{9}{112}$ //

$P(YYY) = (4/7)(3/4)(3/4)(3/4) = \frac{27}{112}$ //

$P(\bar{s}^1) = \{\frac{8}{63}, \frac{4}{63}, \frac{1}{42}, \frac{1}{14}, \frac{1}{42}, \frac{1}{42}, \frac{3}{112}, \frac{9}{112}, \frac{4}{63}, \frac{2}{63},$
$\frac{1}{28}, \frac{1}{14}, \frac{1}{28}, \frac{9}{112}, \frac{27}{112}\}$

$$H(\bar{s}^1) = \sum_{i=1}^{15} P(m_i) \log \frac{1}{P(m_i)}$$

$$= \frac{8}{63}\log\frac{63}{8} + \left(\frac{4}{63}\log\frac{63}{4}\right)(2) + \left(\frac{1}{42}\log 42\right)(3) + \frac{1}{14}\log 14 +$$
$$\frac{3}{112}\log\frac{112}{3} + \left(\frac{9}{112}\log\frac{112}{9}\right)(2) + \frac{2}{63}\log\frac{63}{2} + \left(\frac{1}{28}\log 28\right)(2)$$
$$+ \frac{1}{14}\log 14 + \frac{27}{112}\log\frac{112}{27}$$

$= 3.533$ bits / msg. Symbol //

$$G_1 = \frac{H(s^{-1})}{N}$$

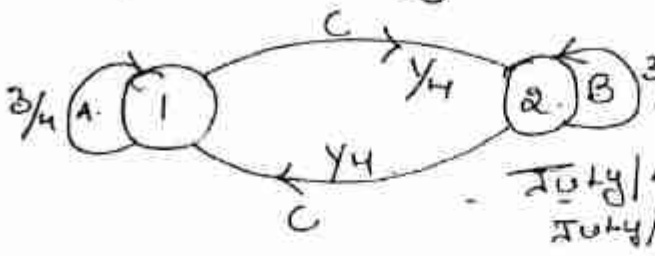
$N = 1$

$$G_1 = \frac{3.533}{3} = 1.178 \text{ bits / msg Symbol}$$ //

It can easily observed. That.

$G_1 > G_2 > G_3 > H$ //

* The markoff Source model Shown in below fig.. Compute H(S) and verify that $G_1 \geq G_2 \geq G_3 \geq H(S)$



July/Aug -2002, 2007 (10M), July/Aug -2008 (12M). Dec2011 (12M).

Sol: $p(1) = \frac{3}{4}p(1) + \frac{1}{4}p(2)$ — (1)

$p(2) = \frac{3}{4}p(2) + \frac{1}{4}p(1)$ — (2)

from eq (1)

$p(1) - \frac{3}{4}p(1) = \frac{1}{4}p(2)$

$p(1)\frac{1}{4} = \frac{1}{4}p(2)$

$p(1) = p(2)$ // — (3)

we have. $p(1) + p(2) = 1$.

$p(2) + p(2) = 1$

$2p(2) = 1$

$p(2) = \frac{1}{2}$ // — (4)

eq (4) sub into eq (3)

$p(1) = \frac{1}{2}$ //

$\therefore p(1) = p(2) = \frac{1}{2}$ //

Entropy of each State is given by.

$$H_i = \sum_{j=1}^{n} P_{ij} \log \frac{1}{P_{ij}}$$

for State 1 $H_1 = \sum_{j=1}^{2} P_{1j} \log \frac{1}{P_{1j}}$

$= P_{11} \log \frac{1}{P_{11}} + P_{12} \log \frac{1}{P_{12}}$.

$= 3/4 \log(4/3) + (1/4) \log(4)$

$= 0.811278$ bits/ message symbol

State 2. :- $H_2 = \sum_{j=1}^{2} P_{2j} \log P_{2j}$

$= P_{21} \log \frac{1}{P_{21}} + P_{22} \log \frac{1}{P_{22}}$.

$= \frac{1}{4} \log(4) + 3/4 \log(4/3)$

$= 0.811278$ bits / message symbol.

the Entropy of the source is given by,

$$H = \sum_{i=1}^{n} P_i H_i = \sum_{i=1}^{2} P(i) H_i$$

$= P(1) H_1 + P(2) H_2$

$= (1/2)(0.81127) + (1/2)(0.81127)$

$= 0.81127$ bits/ message symbol

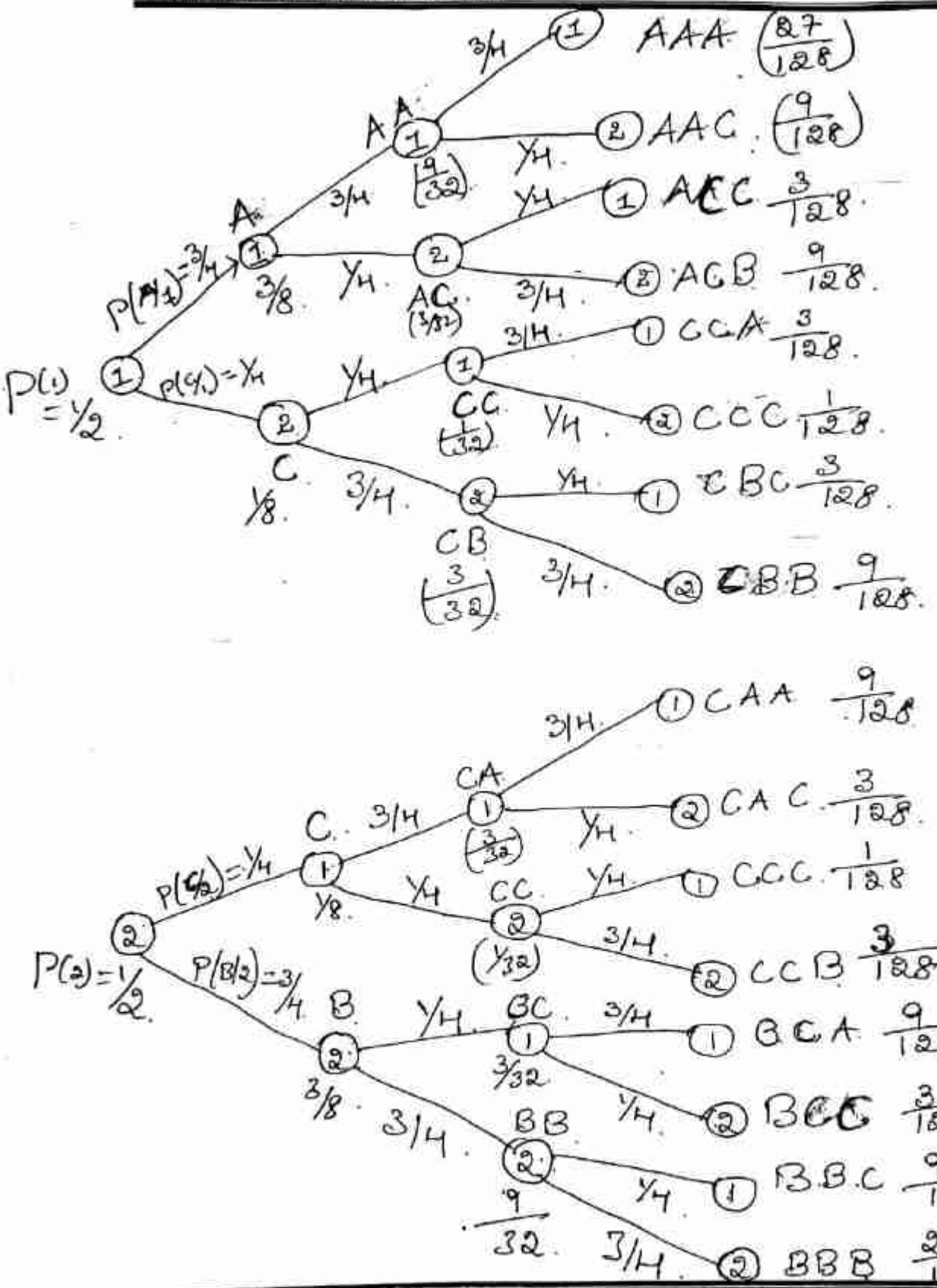


Fig :- Tree diagram showing first four symbols of the second extension of the source. Three first order states.

* The Symbol produced by the source will have an alphabet given by $S = \{A\ B\ C\}$

$$P(A) = (1/2)(3/4) = 3/8 //$$
$$P(B) = (1/2)(3/4) = 3/8 //$$
$$P(C) = (1/2)(1/4) + (1/2)(1/4) = (1/4) //$$
$$P(S) = \{3/8,\ 3/8,\ 1/4\} //$$

$$\therefore G_N = \frac{1}{N} \sum_{i=1}^{3} P(m_i) \log \frac{1}{P(m_i)}$$

$$G_N = \frac{1}{N} H(S)$$

$$H(S) = 3/8 \log(8/3) + 3/8 \log(8/3) + (1/4)\log(4)$$
$$= 1.561287 \text{ bits/message symbol} //$$

$$G_1 = \frac{1}{1}(1.561) = 1.56127 \text{ bit/sec} //$$

* The Symbol produced by the source at the end of Second symbol Interval. Will have an alphabet given by

$$S^2 = \{AA,\ AC,\ CC,\ CB,\ CA,\ BC,\ BB\}.$$

$$P(AA) = 1/2 \times 3/4 \times 3/4 = 9/32 //$$
$$P(AC) = 1/2 \times 3/4 \times 1/4 = 3/32 //$$
$$P(CC) = 1/2 \times 1/4 \times 1/4 + 1/2 \times 1/4 \times 1/4 = 1/16 //$$
$$P(CB) = 1/2 \times 1/4 \times 3/4 = 3/32 //$$
$$P(CA) = 1/2 \times 1/4 \times 3/4 = 3/32 //$$

$P(BC) = 1/2 \times 3/4 \times 1/4 = 3/32$

$P(BB) = 1/2 \times 3/4 \times 3/4 = 9/32$

$\therefore P(\bar{S}^2) = \{9/32, 3/32, 1/16, 3/32, 3/32, 3/32, 9/32\}$

$$H(\bar{S}^2) = \sum_{i=1}^{7} P(m_i) \log \frac{1}{P(m_i)}$$

$$= 9/32 \log\left(\frac{32}{9}\right)(2) + \frac{3}{32}\log\left(\frac{32}{3}\right)(4) + \frac{1}{16}\log(16)$$

$= 2.5600$ bits / message Symbol

When $N=2$, $G_N = \frac{H(\bar{S}^N)}{N} \Rightarrow G_2 = \frac{H(\bar{S}^2)}{2} = \frac{2.5600}{2} = 1.28$ bits/Symbol

The Symbols produced by the Source at the end of 3rd Symbol Interval will have an alphabet given by

$\bar{S}^3 = \{AAA, AAC, ACC, ACB, CCA, CCC, CBC, CBB, CAA, CAC, CCB, BCA, BCC, BBC, BBB\}$

$P(AAA) = (1/2)(3/4)(3/4)(3/4) = 27/128$

$P(AAC) = (1/2)(3/4)(3/4)(1/4) = 9/128$

$P(ACC) = (1/2)(3/4)(1/4)(1/4) = 3/128$

$P(ACB) = (1/2)(3/4)(1/4)(3/4) = 9/128$

$P(CCA) = (1/2)(1/4)(1/4)(3/4) = 3/128$

$P(CCC) = (1/2)(1/4)(1/4) + (1/2)(1/4)(1/4) = \frac{2}{128}$.

$P(CBC) = (1/2)(1/4)(3/4)(1/4) = \frac{3}{128}$//

$P(CBB) = (1/2)(1/4)(3/4)(3/4) = \frac{9}{128}$//

$P(CAA) = (1/2)(1/4)(3/4)(3/4) = 9/128$//

$P(CAC) = (1/2)(1/4)(3/4)(1/4) = 3/128$//

$P(CCB) = (1/2)(1/4)(1/4)(3/4) = 3/128$//

$P(BCA) = (1/2)(3/4)(1/4)(3/4) = \frac{9}{128}$//

$P(BCC) = (1/2)(3/4)(1/4)(1/4) = \frac{3}{128}$//

$P(BBC) = (1/2)(3/4)(3/4)(1/4) = 9/128$//

$P(BBB) = (1/2)(3/4)(3/4)(3/4) = \frac{27}{128}$//

$P(S^3) = \{\frac{27}{128}, \frac{9}{128}, 3/128, 9/128, 3/128, \frac{9}{128}, 3/128,$

$\frac{9}{128}, \frac{9}{128}, \frac{3}{128}, \frac{3}{128}, \frac{9}{128}, \frac{3}{128}, \frac{9}{128}, \frac{27}{128}\}$

$$H(S^1) = \sum_{i=1}^{15} p(m_i) \log_2 \frac{1}{p(m_i)}$$

$$= \frac{27}{128} \log_2 \frac{128}{27}(2) + \frac{9}{128} \log_2 \frac{128}{9}(6) + \frac{3}{128} \log_2 \left(\frac{128}{3}\right)(6) + \frac{2}{128} \log_2 \left(\frac{128}{2}\right)$$

$$= \frac{27}{128}(2.2451)(2) + (9/128)(3.8300)(6) + \frac{3}{128}(5.4150)(6) + \frac{2}{128}(6)$$

$$= 3.4181 \text{ bits/message symbol}$$

$$G_N = \frac{H(S^N)}{N}$$

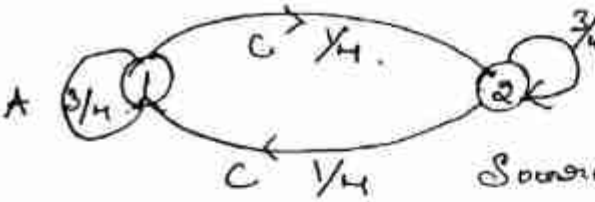
$N = 1$

$$G_1 = \frac{H(S^1)}{1} = \frac{3.4181}{3} = 1.13939 \text{ bits/msg symbol}$$

I + Can easily observed that

$$G_1 > G_2 > G_3 > H.$$

* Given is the model of a markoff source in below fig find. 1) state probability 2) Entropy of 1st order source and second order H(S) and H(S^2) 3) Gr1 & Gr2 . . . .

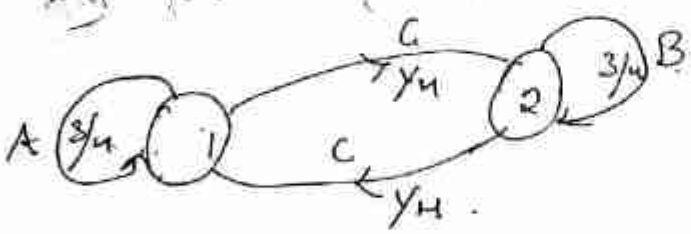


June – July 2011 [10M]

NOTE:

Lee → Similar to the previous problem

* For the source model shown in below figure. find the source entropy and the average information content per symbol in message containing one hundred symbols.

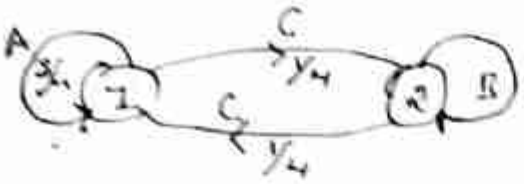


DEC – 2011 (12M).

NOTE: Lee – Similar to the previous problem.

* For the given first order markov source in below fig shown find. 1) State probabilities 2) Entropy of each state 3) Entropy of the source

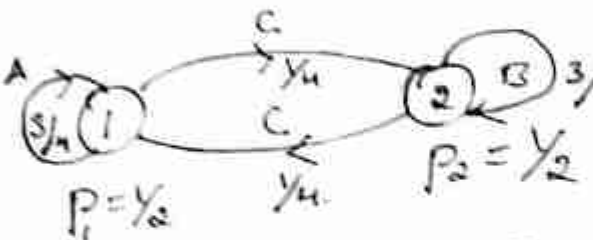
4) Find G_1, G_2.



DEC 2014 / Jan 2015 [10M]

Note 1
Sol :- Similar to the previous problem.

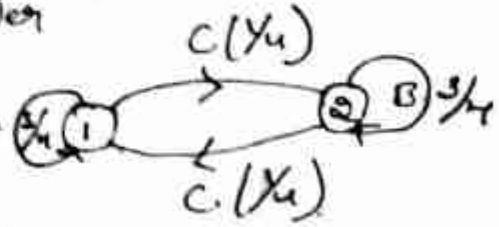
* Find H, G_1 and G_2 for the following model.



DEC 2016 / Jan 2017 [12M]

Note 2
Sol :- Similar to the previous problem.

* The state diagram of a markov source is given fig
1) Find the state probabilities
2) Find the Entropy of the Hi
3) Find the Entropy of the first order source H(S)

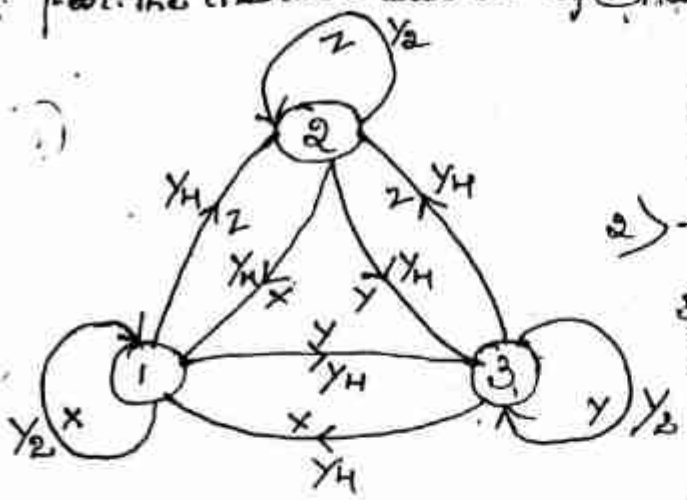


4) Find G_1 G_2 and verify that.
$G_1 \geq G_2 \geq H(S)$.

DEC-09 - Jan-10 (10M)

Note Sol - Similar to the previous problem.

* For the markov source of Shown in below figure.



Find 1) The Entropy of Each State.
2) The Entropy of The Source.
3) Draw the tree diagram representing the States at the end of 2nd Symbol Interval and find the corresponding probabilities.
4) G_1, G_2, and Then Show that $G_1 > G_2 > H$.

May/June -2010 [11M]

Assume $p(1) = p(2) = p(3) = 1/3$.

Sol:-

Assume $p(1) = p(2) = p(3) = 1/3$.

1) The Entropy of each state is given by equation as.

$$H_i = \sum_{j=1}^{2} p_{ij} \log \frac{1}{p_{ij}}$$

State 1 $$H_1 = p_{11} \log \frac{1}{p_{11}} + p_{12} \log \frac{1}{p_{12}} + p_{13} \log \frac{1}{p_{13}}$$

$$= \frac{1}{2} \log 2 + \frac{1}{4} \log 4 + \frac{1}{4} \log 4.$$

$$= 1.5 \text{ bits/message-symbol} \; //$$

$$= \frac{1}{2}\log(2) + \frac{1}{2}\log(2)$$

$$= \frac{1}{2}\frac{\log(2)}{\log(2)} + \frac{1}{2}\frac{\log(2)}{\log(2)}$$

$G_1 = \frac{1}{2}(1) + \frac{1}{2}(1) = 1$ bits/message-symbol

Compare H & G_1 values, we conclude that $G_1 > H$ //

$\therefore N = 2$ (i.e at the start of the 2nd symbol interval) we have

$$G_2 = \frac{1}{2} H(S^2)$$

$$= \frac{1}{2}\left[\frac{3}{8}\log\frac{8}{3} + \frac{1}{8}\log 8 + \frac{1}{8}\log 8 + \frac{3}{8}\log\frac{8}{3}\right]$$

$$= \frac{1}{2}\left[\frac{3}{8}\frac{\log\frac{8}{3}}{\log 2} + \frac{1}{8}\frac{\log 8}{\log 2} + \frac{1}{8}\frac{\log 8}{\log 2} + \frac{3}{8}\frac{\log\frac{8}{3}}{\log 2}\right]$$

$$= \frac{1}{2}\left[\frac{3}{8}(1.4150) + \frac{1}{8}(3) + \frac{1}{8}(3) + \frac{3}{8}(1.4150)\right]$$

$$= \frac{1}{2}[1.81125]$$

$G_2 = 0.905625$ bits/ message symbol.

Aergaen. Comparing values of H, G_1 and G_2, we Conclude that

$G_2 > G_1 > H$ //

State 2: $H_2 = P_{21}\log\frac{1}{P_{21}} + P_{22}\log\frac{1}{P_{22}} + P_{23}\log\frac{1}{P_{23}}$

$= \frac{1}{4}\log(4) + \frac{1}{2}\log(2) + \frac{1}{4}\log(4)$

$= 1.5$ bits/msg Symbol //

State 3:- $H_3 = P_{31}\log\frac{1}{P_{31}} + P_{32}\log\frac{1}{P_{32}} + P_{33}\log\frac{1}{P_{33}}$

$= \frac{1}{4}\log(4) + \frac{1}{4}\log(4) + \frac{1}{2}\log(2)$

$= 1.5$ bits/msg - Symbol //

2) The Entropy of the Source is given by.

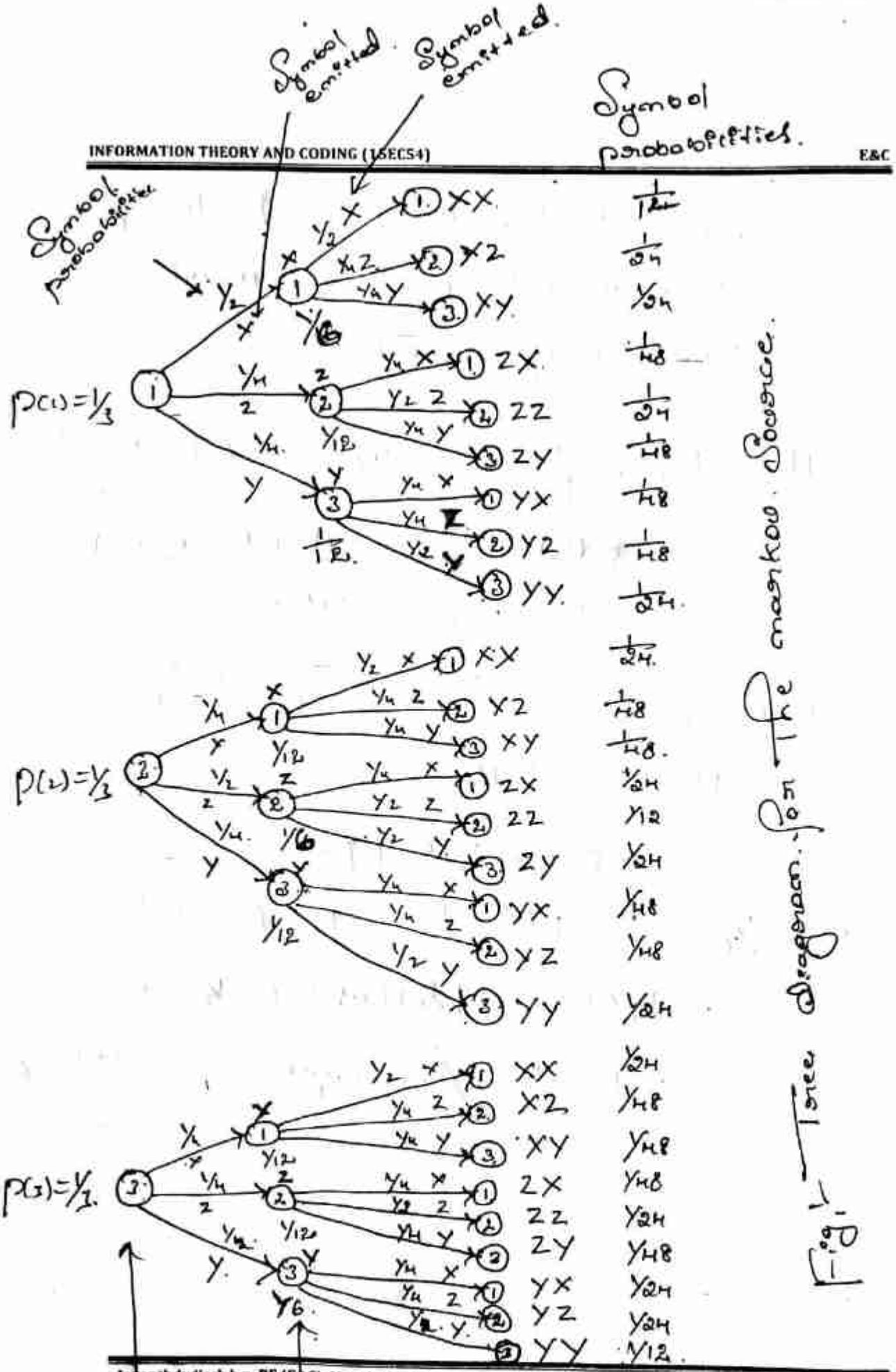
$$H_i = \sum_{i=1}^{3} P_i H_i$$

$= P_1H_1 + P_2H_2 + P_3H_3$

$P_1 = P(1) = 1/3 \quad P_2 = P(2) = 1/3 \quad P_3 = P(3) = 1/3$

$= \frac{1}{3}(1.5) + (\frac{1}{3})(1.5) + (\frac{1}{3})(1.5)$

$= 1.5$ bits/message - Symbol //


Symbol emitted
Symbol emitted
Symbol probabilities
Symbol probabilities
Initial state
State at the end of first Symbol Interval
State at the end of Second Symbol Interval
Fig: Tree Diagram for the markov Source

From the Tree diagrams. It is clear that Three Symbols X, Y, Z are generated. Their probabilities are different depending upon Initial State.

$$P(X) = 1/6 + 1/12 + 1/12 = 1/3 //$$

$$P(Y) = \frac{1}{12} + \frac{1}{12} + \frac{1}{6} = 1/3 //$$

$$P(Z) = 1/12 + \frac{1}{6} + \frac{1}{12} = 1/3 //$$

$\therefore P(\bar{x}) = \{1/3, 1/3, 1/3\}$

From the Tree diagrams, note that There are distinct 9 Sequences of Length 2, Their probabilities are, different depending upon initial State.

$$P(XX) = (1/3)(1/2)(1/2) + (1/3)(1/4)(1/2) + (1/3)(1/4)(1/2) = 1/6 //$$

$$P(XZ) = (1/3)(1/2)(1/4) + (1/3)(1/4)(1/4) + (1/3)(1/4)(1/4) = 1/12 //$$

$$P(XY) = (1/3)(1/2)(1/4) + (1/3)(1/4)(1/4) + (1/3)(1/4)(1/4) = 1/12 //$$

$$P(ZX) = (1/3)(1/4)(1/4) + (1/3)(1/2)(1/4) + (1/3)(1/4)(1/4) = 1/12 //$$

$$P(ZZ) = (1/3)(1/4)(1/2) + (1/3)(1/2)(1/2) + (1/3)(1/4)(1/2) = 1/6 //$$

$$P(ZY) = (1/3)(1/4)(1/4) + (1/3)(1/2)(1/4) + (1/3)(1/4)(1/4) = 1/12 //$$

$$P(YX) = (1/3)(1/4)(1/4) + (1/3)(1/4)(1/4) + (1/3)(1/2)(1/4) = 1/12 //$$

$$P(YZ) = (1/3)(1/4)(1/4) + (1/3)(1/4)(1/4) + (1/3)(1/2)(1/4) = 1/12 //$$

$$P(YY) = (1/3)(1/4)(1/2) + (1/3)(1/4)(1/2) + (1/3)(1/2)(1/2) = 1/6 //$$

4) $G_N = \frac{1}{N} \sum_{i=1}^{n} p(m_i) \log\left(\frac{1}{p(m_i)}\right)$

for $N=1$

$\therefore G_1 = \frac{1}{1} \sum_{i=1}^{n} p(m_i) \log \frac{1}{p(m_i)} = \sum_{i=1}^{2} p(m_i) \log \frac{1}{p(m_i)}$.

$G_1 = H(\bar{s}) = \frac{1}{3}\log 3 + \frac{1}{3}\log 3 + \frac{1}{3}\log 3$

$G_1 = H(\bar{s}) = 1.585$ bits/message - symbol //

We can conclude that $G_1 \geq H$ //

for $N=2$.

$G_2 = \frac{1}{2} \cdot \sum_{i=1}^{2} p(m_i) \log \frac{1}{p(m_i)}$

$G_2 = \frac{1}{2} [H\bar{s}^2]$

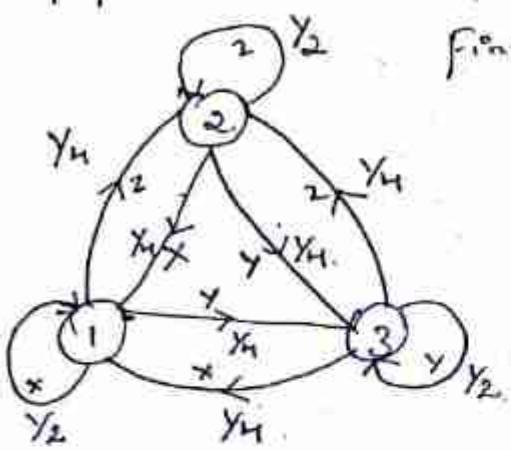
$= \frac{1}{2} \left[(6)\left(\frac{1}{12}\right)\log(12) + (3)\left(\frac{1}{6}\right)\log(6) \right]$

$= \frac{1}{2} [3.08496]$

$= 1.5425$ bits/message //

Thus $G_1 \geq G_2 \geq H$ //

* For the markov source of shown fn. below figure.



Find. 1) Then entropy of state H_i ($i=1,2,3$)
2) Find the entropy of the source H.
3) Find $G_1 G_2 G_3$ and. Shown. That $G_1 \geq G_2 \geq G_3 \geq H$. Assume $p(1)=p(2)=p(3)=1/3$

Jan/feb-2006 (12 M). Jan/feb-2007 (8M)

Sol :- Assume $p(1) = p(2) = p(3) = 1/3$ //.

1) The entropy of each state is given by Equation. as. $H_i = \sum_{j=1}^{n} P_{ij} \log \frac{1}{P_{ij}}$.

State 1

$$H_1 = P_{11} \log \frac{1}{P_{11}} + P_{12} \log \frac{1}{P_{12}} + P_{13} \log \frac{1}{P_{13}}$$
$$= 1/2 \log(2) + 1/4 \log(4) + 1/4 \log(4)$$
$$= 1.5 \text{ bits/ symbol} //$$

State 2.

$$H_2 = P_{21} \log \frac{1}{P_{21}} + P_{22} \log \frac{1}{P_{22}} + P_{23} \log \frac{1}{P_{23}}$$
$$= 1/4 \log(4) + 1/2 \log(2) + 1/4 \log(4)$$
$$= 1.5 \text{ bits/ msg symbol} //.$$

State I : $H_1 = P_{11}\log\frac{1}{P_{11}} + P_{12}\log\frac{1}{P_{12}} + P_{13}\log\frac{1}{P_{13}}$

$= \frac{1}{4}\log(4) + \frac{1}{4}\log(4) + \frac{1}{2}\log(2)$

$= 1.5$ bits/msg-Symbol //

2) The Entropy of the Source is given by

$$H_i = \sum_{i=1}^{n} P_i H_i$$

$= P_1H_1 + P_2H_2 + P_3H_3$

$P_1 = P(1) = \frac{1}{3}$ $P_2 = P(2) = \frac{1}{3}$ $P_3 = P(3) = \frac{1}{3}$

$= (\frac{1}{3})(1.5) + (\frac{1}{3})(1.5) + (\frac{1}{3})(1.5)$

$= 1.5$ bits/ msg-Symbol //

P.T.O.

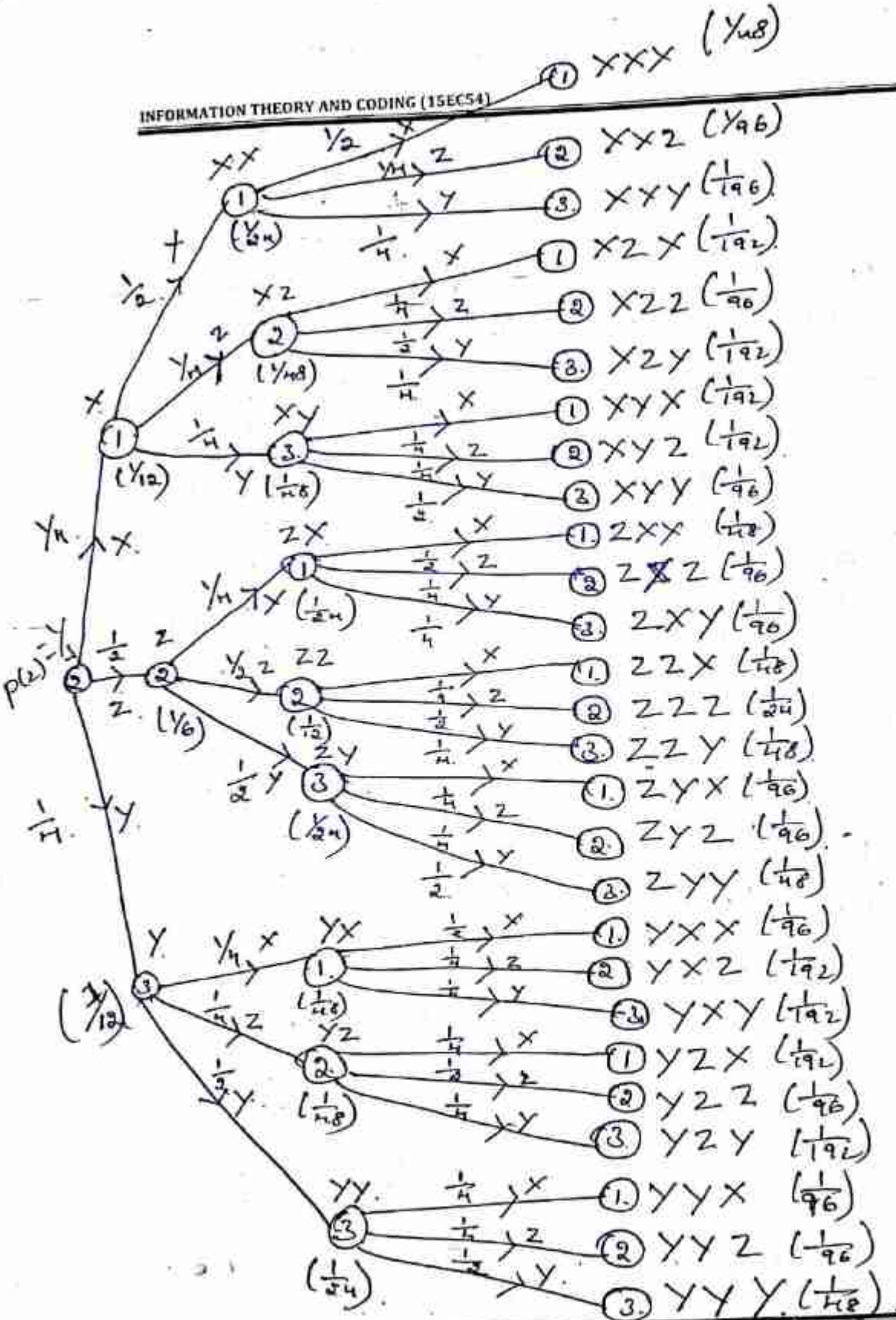


Fig :- Sequence and their probabilities generated Starting from state 2.

Jayanthdwijesh h.p, BE (E&C), M.tech (DECS).
Assistant Professor -Dept of E&C
BGSIT, b.g Nagara, Nagamangala Tq, Mandya.

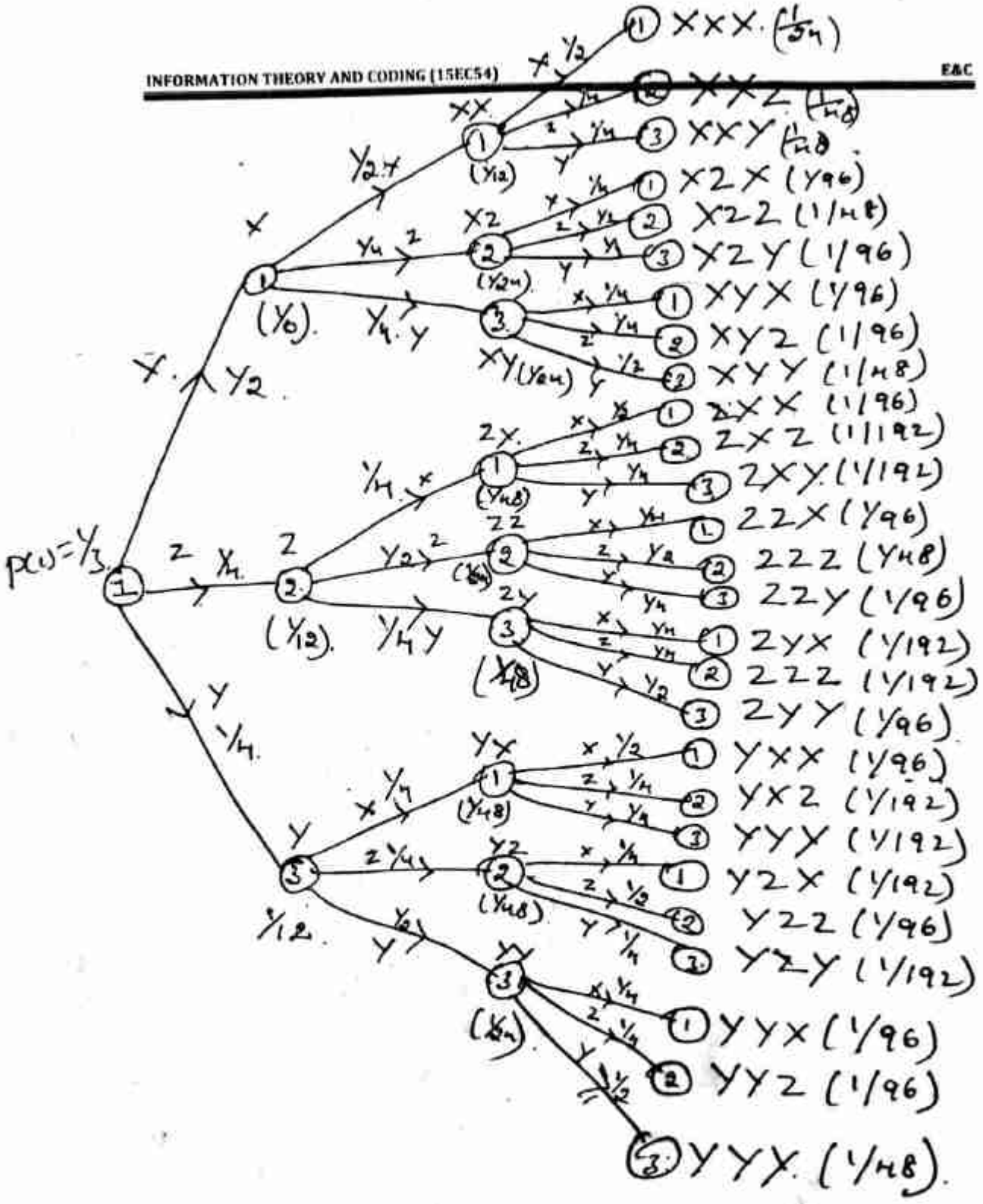


Fig :- Sequence & Their probabilities generated. Starting from State 1

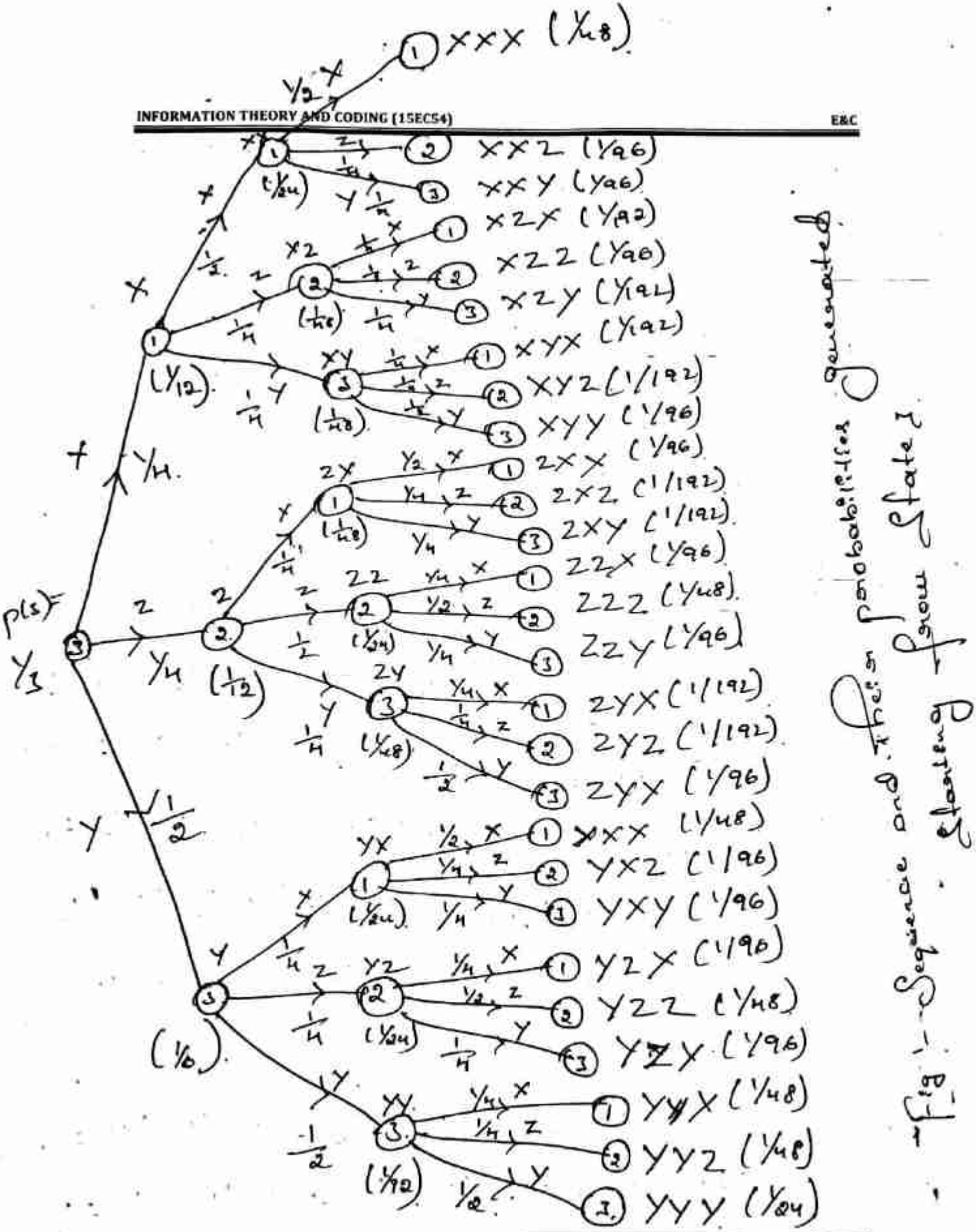


Fig 1 - Sequence and Probabilities for generated source output states

From the Tree diagram it is clear that three Symbols X, Y, Z are generated. Their probabilities are different depending upon initial state.

$P(X) = (1/3)(1/2) + (1/3)(1/4) + (1/3)(1/4) = 1/3$ //

$P(Y) = (1/3)(1/4) + (1/3)(1/4) + (1/3)(1/2) = 1/3$ //

$P(Z) = (1/3)(1/4) + (1/3)(1/2) + (1/3)(1/4) = 1/3$ //

$P(I) = \{1/3, 1/3, 1/3\}$ //

From the Tree diagram we note that there are distinct a sequence of length 2. Their probabilities are different depending upon initial state.

$P(XX) = (1/3)(1/2)(1/2) + (1/3)(1/4)(1/2) + (1/3)(1/4)(1/2) = 1/6$ //

$P(XZ) = (1/3)(1/2)(1/4) + (1/3)(1/4)(1/4) + (1/3)(1/4)(1/4) = 1/12$ //

$P(XY) = (1/3)(1/2)(1/4) + (1/3)(1/4)(1/4) + (1/3)(1/4)(1/4) = 1/12$ //

$P(ZX) = (1/3)(1/4)(1/4) + (1/3)(1/2)(1/4) + (1/3)(1/4)(1/4) = 1/12$ //

$P(ZZ) = (1/3)(1/4)(1/2) + (1/3)(1/2)(1/2) + (1/3)(1/4)(1/2) = 1/6$ //

$P(ZY) = (1/3)(1/4)(1/4) + (1/3)(1/2)(1/4) + (1/3)(1/4)(1/4) = 1/12$ //

$P(YX) = (1/3)(1/4)(1/4) + (1/3)(1/4)(1/4) + (1/3)(1/2)(1/4) = 1/12$ //

$P(YZ) = (1/3)(1/4)(1/4) + (1/3)(1/4)(1/4) + (1/3)(1/2)(1/4) = 1/12$ //

$P(YY) = (1/3)(1/4)(1/2) + (1/3)(1/4)(1/2) + (1/3)(1/2)(1/2) = 1/6$ //

$P(\bar{S}^2) = \{1/6 \cdot 1/12 \cdot 1/12 \cdot 1/12 \cdot 1/6 \cdot 1/12, 1/12 \cdot 1/12 \cdot 1/6\}$

From the Tree diagrams notes that there are distinct a sequence of length 3 their probabilities are different depending upon initial state.

$P(XXX) = (1/24) + (1/48) + 1/48 = \frac{1}{12}$ //

$P(XX2) = \frac{1}{48} + \frac{1}{96} + \frac{1}{96} = \frac{1}{24}$ //

$P(XXY) = \frac{1}{48} \frac{1}{96} + \frac{1}{96} = \frac{1}{24}$ //

$P(X2X) = \frac{1}{96} + \frac{1}{192} + \frac{1}{192} = \frac{1}{48}$ //

$P(X22) = \frac{1}{48} + \frac{1}{96} + \frac{1}{96} = \frac{1}{24}$ //

$P(X2Y) = \frac{1}{96} + \frac{1}{192} + \frac{1}{192} = \frac{1}{48}$ //

$P(XYX) = \frac{1}{96} + \frac{1}{192} + \frac{1}{192} = \frac{1}{48}$ //

$P(XY2) = \frac{1}{96} + \frac{1}{192} + \frac{1}{192} = \frac{1}{48}$ //

$P(XYY) = \frac{1}{48} + \frac{1}{96} + \frac{1}{96} = \frac{1}{24}$ //

$P(2XX) = \frac{1}{96} + \frac{1}{48} + \frac{1}{96} = \frac{1}{24}$ //

$P(2X2) = \frac{1}{192} + \frac{1}{96} + \frac{1}{192} = \frac{1}{48}$ //

$P(2XY) = \frac{1}{192} + \frac{1}{96} + \frac{1}{192} = \frac{1}{48}$ //

$P(22X) = \frac{1}{96} + \frac{1}{48} + \frac{1}{96} = \frac{1}{24}$ //

$P(222) = \frac{1}{48} + \frac{1}{24} + \frac{1}{48} = 1/12$ //

$P(ZZY) = \frac{1}{96} + \frac{1}{48} + \frac{1}{96} = \frac{1}{24}$//

$P(ZYX) = \frac{1}{192} + \frac{1}{96} + \frac{1}{192} = \frac{1}{48}$//

$P(ZYZ) = \frac{1}{192} + \frac{1}{96} + \frac{1}{192} = \frac{1}{48}$//

$P(ZYY) = \frac{1}{96} + \frac{1}{48} + \frac{1}{96} = \frac{1}{24}$//

$P(YXX) = \frac{1}{96} + \frac{1}{96} + \frac{1}{48} = \frac{1}{24}$//

$P(YXZ) = \frac{1}{192} + \frac{1}{192} + \frac{1}{96} = \frac{1}{48}$//

$P(YXY) = \frac{1}{192} + \frac{1}{192} + \frac{1}{96} = \frac{1}{48}$//

$P(YZX) = \frac{1}{192} + \frac{1}{192} + \frac{1}{96} = \frac{1}{48}$//

$P(YZZ) = \frac{1}{96} + \frac{1}{96} + \frac{1}{48} = \frac{1}{24}$//

$P(YZY) = \frac{1}{192} + \frac{1}{196} + \frac{1}{96} = \frac{1}{48}$//

$P(YYX) = \frac{1}{96} + \frac{1}{96} + \frac{1}{48} = \frac{1}{24}$//

$P(YYZ) = \frac{1}{96} + \frac{1}{96} + \frac{1}{48} = \frac{1}{24}$//

$P(YYY) = \frac{1}{48} + \frac{1}{48} + \frac{1}{24} = \frac{1}{12}$//

$$G_N = \frac{1}{N} \sum_{i=1}^{n} P(M_i) \log\left(\frac{1}{P(M_i)}\right)$$

For $N = 1$

$$G_1 = \frac{1}{1} \sum_{i=1} P(M_i) \log \frac{1}{P(M_i)}$$

$G_1 = H(S) = \frac{1}{3}\log 3 + \frac{1}{3}\log 3 + \frac{1}{3}\log 3 = 1.585$ bits/ symbol

For $N=2$.

$$G_2 = \frac{1}{N} \sum_{i=1}^{n} P(m_i) \log\left(\frac{1}{P(m_i)}\right)$$

$$G_2 = \frac{1}{2}\left[H(\bar{s}^2)\right]$$

$$= \tfrac{1}{2}\left[(6)(1/12)\log(12) + (3)(1/6)\log(6)\right]$$

$$= \tfrac{1}{2}\left[1.0849\right].$$

$$= 1.5425 \text{ bits/message}$$

For $N=3$

$$G_3 = \frac{1}{N} \sum_{i=1}^{n} P(m_i) \log_2\left(\frac{1}{P(m_i)}\right)$$

$$G_3 = \frac{1}{3}\left[H(\bar{s}^3)\right]$$

$$= \tfrac{1}{3}\left[(3)(1/12)\log_2(12) + (12)(1/24)\log_2(24) + (12)(1/48)\log_2(48)\right]$$

$$G_3 = 1.5883 \text{ bits/message}$$

$\therefore$ $H = 1.58 \text{ bits/message symbol}$

$G_1 = 1.5849 \text{ bits/msg. symbol}$

$G_2 = 1.5425 \text{ bits/symbols}$

$G_3 = 1.5883 \text{ bits/symbols}$

Thus $G_1 > G_2 > G_3 > H$.

* For the first order markov source with a source alphabet $S = \{A, B, C\}$ shown in below figure
1) Compute the probabilities of state.
2) find $H(S)$ and $H(S^2)$

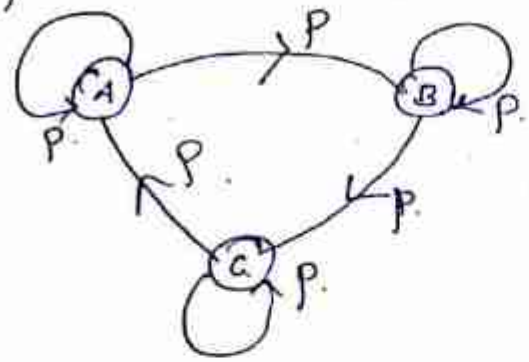


fig: markov source.

1) $p(A) = p\,p(A) + p\,p(C)$ — (1)

$p(B) = p\,p(B) + p\,p(A)$ — (2)

$p(C) = p\,p(C) + p\,p(B)$ — (3)

Adding all the (1), (2), (3)

$p(A) + p(B) + p(C) = 2p\,p(A) + p\,p(C) + p\,p(B) + p\,p(A) + p\,p(C) + p\,p(B)$

$= 2p^2 p(A) + 2p\,p(B) + 2p\,p(C)$

$= 2p\,[p(A) + p(B) + p(C)]$

$[1] = 2p \times [1]$

$p = 1/2$ //

note

$p(A) + p(B) + p(C) = 1$

Sum of all probs equal to one.

using eq (1)

$$P(A) = \frac{1}{2} P(A) + \frac{1}{2} P(C)$$

$$P(A) = P(C) \quad \text{---------- (4)}$$

Again using $P = 1/2$ in eq (2)

$$P(B) = \frac{1}{2} P(B) + \frac{1}{2} P(A)$$

$$P(B) - \frac{1}{2} P(B) = \frac{1}{2} P(A)$$

$$\frac{1}{2} P(B) = \frac{1}{2} P(A)$$

$$P(B) = P(A) \quad \text{---------- (5)}$$

Combining eq (4) & (5) we get

$$P(A) = P(B) = P(C)$$

But $P(A) + P(B) + P(C) = 1$

$$P(A) = P(B) = P(C) = \frac{1}{3}$$

NOTE

$P(A) + P(A) + P(A) = 1$

$\therefore P(A) = 1/3$

$P(B) + P(B) + P(B) = 1$

$P(B) = 1/3$

$P(C) + P(C) + P(C) = 1$

$3P(C) = 1$

$P(C) = 1/3$

2) $$H_i = \sum_{i=1}^{a} P_{ij} \log \frac{1}{P_{ij}}$$

$$H_A = P_{AA} \log \frac{1}{P_{AA}} + P_{AB} \log \frac{1}{P_{AB}} + P_{AC} \log \frac{1}{P_{AC}}$$

$$= P \log \frac{1}{P} + P \log \frac{1}{P} + 0$$

$$= 2P \log \frac{1}{P} = 2 \times \frac{1}{2} \log(2)$$

$$= 1 \text{ bit/msg-symbol}$$

$$H_B = P_{BA}\log\frac{1}{P_{BA}} + P_{BB}\log\frac{1}{P_{BB}} + P_{BC}\log\frac{1}{P_{BC}}$$

$$= 0 + p\log\frac{1}{p} + p\log\frac{1}{p} = 2p\log\frac{1}{p}$$

$$= 2(1/2)\log(2)$$

$$= 1 \text{ bit/msg-symbol}$$

$$H_C = P_{CA}\log\frac{1}{P_{CA}} + P_{CB}\log\frac{1}{P_{CB}} + P_{CC}\log\frac{1}{P_{CC}}$$

$$= p\log\frac{1}{p} + 0 + p\log\frac{1}{p} = 2p\log\frac{1}{p}$$

$$= 2(1/2)\log(2)$$

$$= 1 \text{ bit/msg-symbol}$$

- $H(S) = H = \sum_{i=1}^{n} P_i H_i$

$$= p(A)H_A + p(B)H_B + p(C)H_C$$

$$= (1/3)(1) + (1/3)(1) + (1/3)(1)$$

$$= 1 \text{ bit/msg-symbol}$$

$$H(S^2) = 2H(S)$$

$$= 2\,(1 \text{ bit/msg-symbol})$$

$$= 2 \text{ bits/msg-symbol}$$

* For the markov Source shown in fig find.

a) State probabilities.
b) State entropies.
c) Source entropy.
d) G_1 G_2.
S.T $G_1 > G_2 > H$

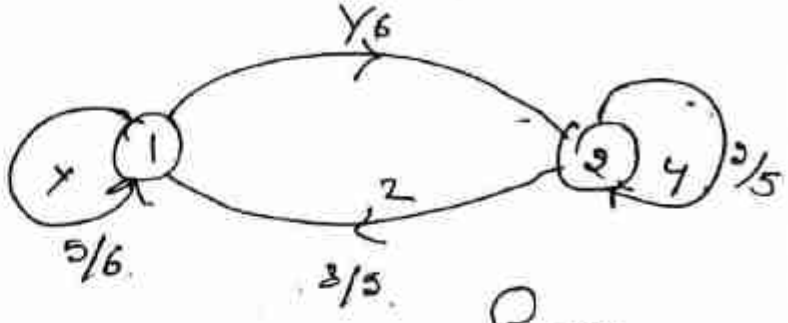


Fig: markov Source

Sol: a. State 1

$$p(1) = \frac{5}{6} p(1) + \frac{3}{5} p(2) \quad \text{— (1)}$$

$$p(2) = \frac{1}{6} p(1) + \frac{2}{5} p(2) \quad \text{— (2)}$$

$$\therefore p(1) + p(2) = 1 \quad \text{— (3)}$$

Consider eq (1)

$$p(1) - \frac{5}{6} p(1) = \frac{3}{5} p(2)$$

$$\frac{1}{6} p(1) = \frac{3}{5} p(2)$$

$$p(1) = \frac{18}{5} p(2) \quad \text{— (4)}$$

eq (4) Substitute into eq (3)

$$\frac{18}{5} p(2) + p(2) = 1$$

$$p(2) = \frac{5}{23} \quad \text{— (5)}$$

eq (5) Sub Put eq (4)

$$P(1) = \frac{18}{5} \times \frac{5}{23} = \frac{18}{23} //$$

b) $H_i = \sum_{j=1}^{2} P_{ij} \log \frac{1}{P_{ij}} = \sum_{j=1}^{2} P_{ij} \log \frac{1}{P_{ij}}$

State 1 :- $H_1 = P_{11} \log \frac{1}{P_{11}} + P_{12} \log \frac{1}{P_{12}}$

$= \frac{5}{6} \log \frac{6}{5} + \frac{1}{6} \log (6)$

$= 0.65$ bits/symbol //

State 2 :- $H_2 = P_{21} \log \frac{1}{P_{21}} + P_{22} \log \frac{1}{P_{22}}$

$= \frac{3}{5} \log \frac{5}{3} + \frac{2}{5} \log \frac{5}{2}$

$= 0.9709$ bits/symbol //

c) Source entropy is given by

$$H = \sum_{i=1}^{2} P(i) H_i$$

$= P(1) H_1 + P(2) H_2$

$= \frac{18}{23} (0.65) + \frac{5}{23} (0.9709)$

$= 0.7197$ bits/sym //

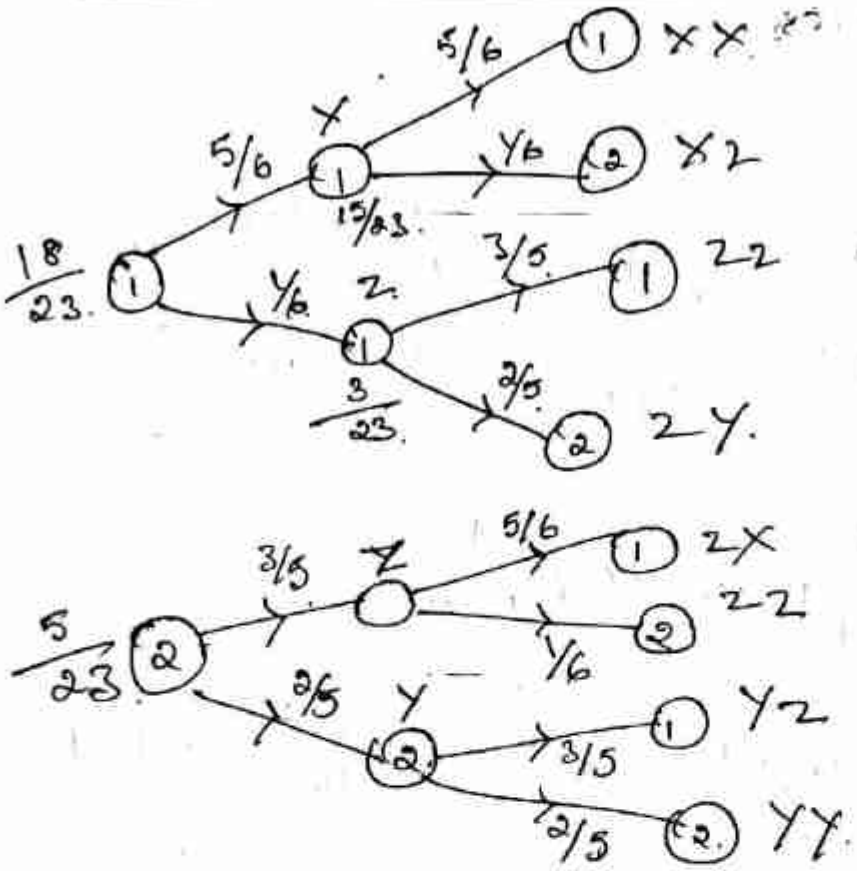


Fig Tree diagram.

From the Tree diagram.

$\therefore P(X) = \frac{15}{23}$ $P(Y) = \frac{2}{23}$ $P(Z) = \frac{3}{23} + \frac{3}{23}$

$= 6/23$ //.

Where $N = 1$

$$G_N = \frac{1}{N} \sum_{i=1}^{3} P(x_i) \log \frac{1}{P(x_i)}$$

$$G_1 = \left[\left(\frac{15}{23}\right)\log\frac{23}{15} + \left[\log\left(\frac{23}{2}\right)\right]\left(\frac{2}{23}\right) + \left(\frac{6}{23}\right)\log\frac{23}{6}\right]$$

$G_1 = 1.2142$ bits/sym //

Symbol Length two :-

$P(xx) = \frac{25}{46}$ · $P(x2) = \frac{5}{46}$ $P(22) = \frac{1}{10}$ $P(24) = \frac{6}{115}$

$P(2x) = \frac{5}{46}$ $P(y2) = \frac{6}{115}$ $P(yy) = \frac{4}{115}$

$N = 2$

$$G_N = \frac{1}{N} \sum_{i=1}^{7} P(m_i) \log \frac{1}{P(m_i)}$$

$$= \frac{1}{2} \left[\frac{25}{46} \log \frac{46}{25} + \frac{5}{46} \left[\log \frac{5}{46} \right] (2) + \frac{1}{10} \log(10) + \left(\frac{6}{115} \right) (2) \log \left(\frac{115}{6} \right) + \frac{4}{115} \log \left(\frac{115}{4} \right) \right]$$

$= 1.0597$ bits/sym //

$\therefore G_1 \geq G_2 \geq H$ //

MISCELLANEOUS EXAMPLES

* An analog Signal band limited to 6KHz is Sampled at thrice the nyquist rate and then quantized into 11 Levels. $Q_1, Q_2, \ldots Q_{11}$. of these three levels occur with probability of $\frac{1}{6}$ each, four levels with probability of $\frac{1}{12}$ each and the remaining four levels with probability $\frac{1}{24}$ each. Find the Rate of Info associated with the Analog Sig?

Sol :- Sampling rate $r_s = 3 \times$ Nyquist rate

$= 3 \times 2B$

$= 3 \times 2 \times 6K = 36$ K Samples/sec //

The entropy of the analog signal is given by

$$H(S) = \sum_{i=1}^{11} P_i \log \frac{1}{P_i} \text{ bits/message Samples}$$

$$= 3 \times \frac{1}{6} \log(6) + 4 \times \frac{1}{12} \log(12) + \left(\frac{1}{24}\right)(4) \log(24)$$

$= 3.252$ bits/Sample //

Rate of Information $(R_1) = R_1 = r_s H(S)$

$R_1 = (36.K \text{ Samples/sec})(3.252 \text{ bits/Sample})$

$R_1 = 117.072$ KBPS. //

* A pair Coin is Tossed repeatedly

Let X = {event of getting 5 heads out of 8 trials}

Y = {event of getting 7 heads out of 11 trials}

which event conveys more Information? Support your answer by numerical Computation of respective amounts of Information?

Sol:- Z = number of heads occurring

= Binomial R.V with parameters.

n = number of Trials.

p = probability of getting head. $= 0.5$

$q = 1 - p = 1 - 0.5 = 0.5$ //

Binomial Random Variable is probability distribution function is given by.

$$P(X=x) = {}^{n}C_x\, p^x\, q^{n-x} \quad x = 0,1,2 \cdots n$$

Note: $q = 1-p$

$$P(X=x) = {}^{n}C_x\, p^x (1-p)^{n-x}$$

$$P(x) = P(x=5) = {}^{8}C_5\, (0.5)^5 (1-0.5)^{8-5} = 0.21875 //$$

$$P(Y) = P(Y=7) = {}^{n}C_y\, p^y (1-p)^{n-y}$$

$$= {}^{11}C_7\, (0.5)^7 (1-0.5)^{11-7}$$

$$= 0.16113 //$$

Self Infⁿ of $x = I(x) = \log \frac{1}{P(x)}$

$= \log \frac{1}{0.21875} = 2.193$ bits

Self Infⁿ of $y = I(y) = \log \frac{1}{P(y)} = \log \frac{1}{0.16113}$

$I(y) = 2.634$ bits

∴ y = carries more Information

* A pair of dice are tossed. Simultaneously. The outcome of 1st dice is recorded as X_1 and that of second dice as X_2. Three Events are defined as below.

$A = \{(X_1, X_2)$ Such that $(X_1 + X_2)$ is divisible exactly by 4$\}$

$B = \{(X_1, X_2)$ Such that $6 \le (X_1 + X_2) \le 8\}$

$C = \{(X_1, X_2)$ Such that $X_1 X_2$ is divisible Exactly by 3$\}$

which Events Convey more Infⁿ? Support your answer by numerical Computation.

Sol :- The Sample Space "S" has thirty Six Combination, is given by

$$S = \begin{bmatrix} (1,1) & (1,2) & (1,3) & (1,4) & (1,5) & (1,6) \\ (2,1) & (2,2) & (2,3) & (2,4) & (2,5) & (2,6) \\ (3,1) & (3,2) & (3,3) & (3,4) & (3,5) & (3,6) \\ (4,1) & (4,2) & (4,3) & (4,4) & (4,5) & (4,6) \\ (5,1) & (5,2) & (5,3) & (5,4) & (5,5) & (5,6) \\ (6,1) & (6,2) & (6,3) & (6,4) & (6,5) & (6,6) \end{bmatrix}$$

* The Event A Contains the pairs as given by.

$A = \{(1,3)\ (2,2)\ (2,6)\ (3,1)\ (3,5)\ (4,4)\ (5,3)\ (6,2)\ (6,6)\}$

$P(A) = \frac{9}{36} = 0.25$ //

NOTE: 1 exactly opposite pair by H.
ex: $(1,3) = (x_1 + x_2) = 1+3 = 4$ //
$4/4 = 1$ //

* The event B Contains :-

$B = \{(1,5)\ (1,6)\ (2,4)\ (2,5)\ (2,6)\ (3,3)\ (3,4)\ (3,5)\ (4,2)\ (4,3)\ (4,4)\ (5,1)\ (5,2)\ (5,3)\ (6,1)\ (6,2)\}$

$P(B) = \frac{16}{36} = 0.444$ //

NOTE: Event B (b) exactly equal 8

* The event "C" Contains.

$C = \{(1,3)\ (1,6)\ (2,3)\ (2,6)\ (3,1)\ (3,2)\ (3,3)\ (3,4)\ (3,5)\ (3,6)\ (4,3)\ (4,6)\ (5,3)\ (5,6)\ (6,1)\ (6,2)\ (6,3)\ (6,4)\ (6,5)\ (6,6)\}$

$P(C) = \frac{20}{36} = 0.556$ //

∴ Self Info of A $= I_A = \log \frac{1}{0.25} = 2$ bits //.

Self Info of B $= I_B = \log \frac{1}{0.444} = 1.17$ bits //.

Self Info of C $= I_C = \log \frac{1}{0.555} = 0.848$ bits //

∴ The Event "A" Carries the maximum Info^n //.

* A voice signal in a "PCM" system is sampled at 2.5 times the Nyquist rate and is quantized into 16 levels with the following probabilities

$P_1 = P_2 = P_3 = P_4 = 0.08$

$P_5 = P_6 = P_7 = P_8 = 0.065$

$P_9 = P_{10} = P_{11} = P_{12} = 0.055$

$P_{13} = P_{14} = P_{15} = P_{16} = 0.05$

Calculate the entropy and Information rate of the PCM signal if the bandwidth of signal is 3.5KHz

Sol:- Entropy $= H(x) = \sum_{i=1}^{16} P_i \log \frac{1}{P_i}$

$H(x) = 4 \times (0.08) \log \frac{1}{0.08} + 4 \times 0.065 \log \frac{1}{0.065} + 4 \times 0.055 \log \frac{1}{0.055}$

$+ 4 \times 0.05 \log \frac{1}{0.05}$

$H(x) = 3.9763$ bits/level (or) Sample //

Sampling rate $=$ (2.5) (Nyquist rate)

$=$ (2.5) (2 B)

$=$ (2.5) (2) (3.5K)

$=$ 17.5 K. Samples/sec //

Information rate $R_1 = r_s H(x)$

$= (17.5\text{K})\ (3.976\text{x})$

$=$ 69.585 KBPS //

* A Discrete Source emits the messages M_1, M_2, M_3 and M_4 with probabilities $\frac{7}{15}, \frac{4}{15}, \frac{3}{15}$ and $\frac{1}{15}$ respectively. Calculate the Information Content of each message and the average Information Content per message. Show that the Entropy of the Source Increases when the Symbol M_4 is partitioned into Three Sub-Symbol M_{41}, M_{42} and M_{43} with respective probabilities $\frac{1}{240}, \frac{1}{60}$ and $\frac{1}{120}$.

Sol:- Self Infn in $m_1 = I_{m_1} = \log_2\left(\frac{1}{7/15}\right) = 1.0995$ bits //

Self Infn in $m_2 = I_{m_2} = \log_2\left(\frac{15}{4}\right) = 1.9069$ bits //

Self Infn in $M_3 = I_{M_3} = \log_2\left(\frac{15}{1}\right) = 2.322$ bits //

Self Infn in $M_4 = I_{M_4} = \log_2\left(\frac{15}{1}\right) = 3.9069$ bits //

Average Information $= H(s) = \sum_{i=1}^{4} P_{m_i} \log_2 \frac{1}{P_{m_i}}$

$$H(s) = \frac{7}{15}\log_2\left(\frac{15}{7}\right) + \frac{4}{15}\log_2\left(\frac{15}{4}\right) + \frac{3}{15}\log_2\left(\frac{15}{1}\right) + \frac{1}{15}\log_2(15)$$

$H(s) = 1.7465$ bits/message Symbol //.

P.T.O.

when m_4 is partitioned into three sub-symbols then the source 'S' will have six symbols given by $m_1, m_2, m_3, m_{41}, m_{42}$ and m_{43} with respective probabilities of $\frac{7}{15}, \frac{4}{15}, \frac{3}{15}, \frac{1}{24}, \frac{1}{60}$ & $\frac{1}{120}$. The new entropy of source 'S' is given by

NOTE: partitioned into

Consider $m_1\ m_2\ m_3\ m_{41}\ m_{42}\ m_{43}$

$\frac{7}{15}\ \frac{4}{15}\ \frac{3}{15}\ \frac{1}{24}\ \frac{1}{60}\ \frac{1}{120}$.

$$H(S') = \sum_{i=1}^{6} P_i \log \frac{1}{P_i}$$

$$= \frac{7}{15} \log\left(\frac{15}{7}\right) + \frac{4}{15} \log \frac{15}{4} + \frac{3}{15} \log \frac{15}{3} + \frac{1}{24} \log 24 + \frac{1}{60} \log 60 + \frac{1}{120} \log 120$$

$$= 1.8331 \text{ bits/message symbol} \quad //$$

$$H(S') > H(S)$$

$$1.8331 > 1.7465 \quad //$$

* Consider a source with alphabets m_1 and m_2 with Respective probabilities of $\frac{5}{6}$ and $\frac{1}{6}$. Determine the entropy of source 'S' and the entropy of it's third extension. Hence show that $H(S^3) = 3H(S)$.

Sol:- Entropy of source 'S' is given by

given $(m_1, m_2) = \{5/6, 1/6\}$ //

m_i

$$H(S) = \sum_{i=1}^{2} P_{mi} \log \frac{1}{P_{mi}}$$

$$= \frac{5}{6} \log 6/5 + 1/6 \log 6.$$

$$= 0.65 \text{ bits/message symbol} //$$

3rd. Extension :-

Symbols (σ_i)	$m_1m_1m_1$	$m_1m_1m_2$	$m_1m_2m_1$	$m_1m_2m_2$	$m_2m_1m_1$	$m_2m_1m_2$	$m_2m_2m_1$	$m_2m_2m_2$
$P(\sigma_i)$	$\frac{125}{216}$	$\frac{25}{216}$	$\frac{25}{216}$	$\frac{25}{216}$	$\frac{5}{216}$	$\frac{5}{216}$	$\frac{5}{216}$	$\frac{1}{216}$

Entropy of 3rd. extended source is given by

$$H(S^3) = \sum_{i=1}^{8} P(\sigma_i) \log \frac{1}{P(\sigma_i)}$$

$$= \frac{125}{216} \log \frac{216}{125} + 3 \times \frac{25}{216} \log \frac{216}{25} + 3 \times \frac{5}{216} \log \frac{216}{5} + \frac{1}{216} \log 216.$$

$H(S^3) = 1.95$ bits / message symbol //

(OR)

$= 3 \times 0.65$ bits / Symbol //

$H(S^3) = 3H(S)$ // proved. //

A zero memory source contains $X = \{x_1, x_2, x_3, x_4\}$ with $P(X) = \{1/2, 1/4, 1/8, 1/8\}$.

1) Determine Entropy of the source.

2) Determine Second Extention of the source.

that $H(X^2) = 2H(X)$. // Jan/Feb-2002(10M), July/Aug-2004(8M), July/Aug-2008(8M)

Sol:- $P(x_1) = 1/2$, $P(x_2) = 1/4$, $P(x_3) = 1/8$, $P(x_4) = 1/8$.

The Entropy is given by Equation.

$$H(X) = \sum_{K=1}^{4} P_K \log_2 \frac{1}{P_K}$$

$$= \sum_{K=1}^{4} P_K \log_2 \frac{1}{P_K}$$

$$= 1/2 \log_2 1/2 + 1/4 \log_2 \frac{1}{1/4} + 1/8 \log_2 \frac{1}{1/8} + 1/8 \log_2 \frac{1}{1/8}$$

$$= 1/2 \log_2(2) + 1/4 \log_2(4) + 1/8 \log_2(8) + 1/8 \log_2(8)$$

$$H(X) = 7/4 \text{ bits/symbol} //$$

$$2H(X) = 2 \times 7/4 = 7/2 //$$

NOTE

Entropy formula

1) $H = \sum_{i=1}^{M} P_i \log_2 (1/P_i)$

2) $H(x) = \sum_{i=1}^{M} P_K \log_2 (1/P_K)$ (or)

3) $H(S) = \sum_{i=1}^{q} P_i \log_2 (1/P_i)$

The Second Extension X^2 of the basic source will have $4^2 = 16$ Symbol which can be coded along with their respective probabilities of occurance as shown in below Table.

SYMBOL	PROBABILITY
$x_1 x_1$	$1/2 \times 1/2 = 1/4$ //
$x_1 x_2$	$1/2 \times 1/4 = 1/8$ //
$x_1 x_3$	$1/2 \times 1/8 = 1/16$ //
$x_1 x_4$	$1/2 \times 1/8 = 1/16$ //
$x_2 x_1$	$1/4 \times 1/2 = 1/8$ //
$x_2 x_2$	$1/4 \times 1/4 = 1/16$ //
$x_2 x_3$	$1/4 \times 1/8 = 1/32$ //
$x_2 x_4$	$1/4 \times 1/8 = 1/32$ //
$x_3 x_1$	$1/8 \times 1/2 = 1/16$ //
$x_3 x_2$	$1/8 \times 1/4 = 1/32$ //
$x_3 x_3$	$1/8 \times 1/8 = 1/64$ //
$x_3 x_4$	$1/8 \times 1/8 = 1/64$ //
$x_4 x_1$	$1/8 \times 1/2 = 1/16$ //
$x_4 x_2$	$1/8 \times 1/4 = 1/32$ //
$x_4 x_3$	$1/8 \times 1/8 = 1/64$ //
$x_4 x_4$	$1/8 \times 1/8 = 1/64$ //

Table :- Source Symbol & probabilities for 2nd. Extended Source of above example.

The Entropy of the 2nd Extended Sources is given by

$$H(X^2) = \sum_{k=1}^{16} P_k \log_2 \frac{1}{P_k}$$

$$= \tfrac{1}{4}\log_2\left(\tfrac{1}{1/4}\right) + \left[\tfrac{1}{8}\log_2\left(\tfrac{1}{1/8}\right)\right](2) + \left[\tfrac{1}{16}\log_2\left(\tfrac{1}{1/16}\right)\right](6)$$

$$+\left[\tfrac{1}{32}\log_2\left(\tfrac{1}{1/32}\right)\right](4) + \left[\tfrac{1}{64}\log_2\left(\tfrac{1}{1/64}\right)\right](4)$$

$$= \tfrac{1}{4}\log_2(4) + \left[\tfrac{1}{8}\log_2(8)\right](2) + \left[\tfrac{1}{16}\log_2(16)\right](5)$$

$$+\left[\tfrac{1}{32}\log_2(32)\right](4) + \left[\tfrac{1}{64}\log_2(64)\right](4)$$

$$= \tfrac{1}{4}\cdot\frac{\log 4}{\log 2} + \left[\tfrac{1}{8}\left(\frac{\log 8}{\log 2}\right)\right](2) + \left[\tfrac{1}{16}\left(\frac{\log 16}{\log 2}\right)\right](5) +$$

$$\left[\tfrac{1}{32}\left(\frac{\log 32}{\log 2}\right)\right](4) + \left[\tfrac{1}{64}\cdot\frac{\log 64}{\log 2}\right](4)$$

$$= \frac{1}{4}(2) + \frac{1}{8}(3)(2) + \frac{1}{16}(4)(5) + \frac{1}{32}(5)(4)$$

$$+\tfrac{1}{64}(6)(4)$$

$$H(X^2) = 0.5 + 0.75 + 1.25 + 0.625 + 0.375 = 3.5 = 7/2 \text{ //}$$

The above result Confirm the following.

$$H(X)^2 = 7/2 \text{ //} \qquad = 2H(X) = 7/4 \times 2 = 7/2 \text{ //}$$

Hence $H(X^2) = 2H(X)$ // proved.

* Design a system to report the heading of collection of 400 cars. The heading levels are Heading Straight (S), turning Left (L) and turning right (R). This Information is to be Transmitted. Every second. Construct a model on the basis of following test dat

(a) on an average, during Reporting intervals, 800 Cars. were heading Straight, 100 were turning Left and. Remaining were turning right.

(b). out of 800 Cars that reported heading Straight 100 reported going Straight during next reporting period, 50 turning Left and. Remaining turning right during the next period.

(c) out of 100 cars that reported as turning during a Signaling period. 50 Continued. the turn and. Remaining headed. Straight. during next Signaling interval

(d). The dynamics of the Cars did not allow them to Change their heading from Left to right

(e) right to Left during Subsequent reporting period.

Find The Entropies of each States & Source entropy and Rate of Infn.

Sol: There are Three States in the model
Heading Straight (S)
Turning Left (L).
Turing Right (R)

* The markoff model can be constructed on the basis of data given in the problem. It is given that 200 cars were heading Straight, 100 were turning Left and Remaining 100 were turning Right in any turning interval on an average. And there are 400 cars in Total. Therefore the probabilities of the States can be determined as follows.

$P(S) = \frac{200}{400} = 1/2$

$P(L) = \frac{100}{400} = 1/4$

$P(R) = \frac{100}{400} = 1/4$

Also from the 2nd & 3rd statement, transition probabilities can be determined. The model constructed on the basis of given data is as shown in below.

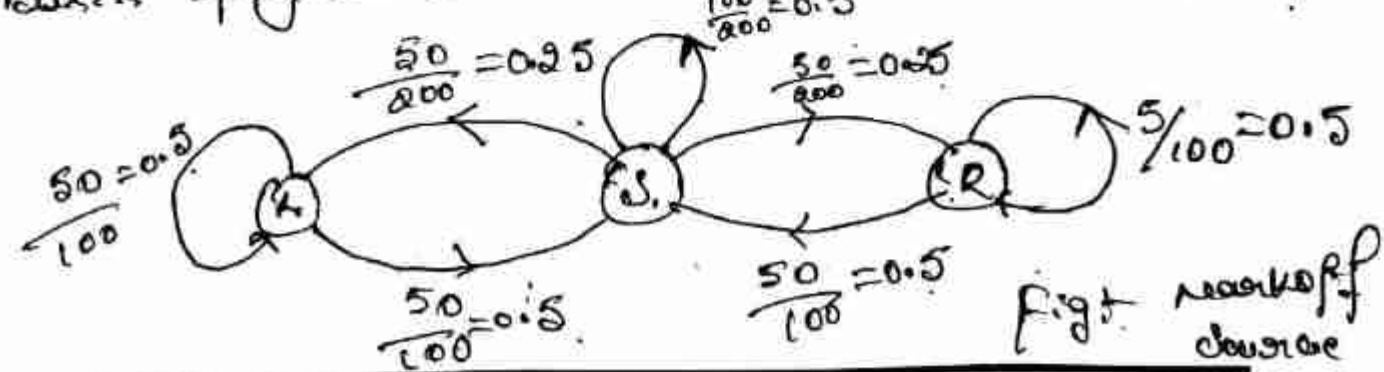


Fig: markoff source

State entropies is given by

$$H_i = \sum_{i=1}^{n} P_{ij} \log \frac{1}{P_{ij}} \text{ bits/message-symbol}$$

State S1

$$H_S = 0.5 \log_2 \frac{1}{0.5} + 0.25 \log_2 \frac{1}{0.25} + 0.25 \log_2 \frac{1}{0.25}$$

$$= 1.5 \text{ bits/sym} //$$

$$H_L = 0.5 \log_2 \frac{1}{0.5} + 0.5 \log_2 \frac{1}{0.5}$$

$$= 1 \text{ bits/sym} //$$

$$H_R = 0.5 \log_2 \frac{1}{0.5} + 0.5 \log_2 \frac{1}{0.5} = 1 \text{ bits/sym} //$$

Source entropy is given by.

$$H = \sum_{i=1}^{n} P_i H_i \text{ bits/sym.}$$

$$= P(S).H_S + P(L) H_L + P(R) H_R$$

$$= \frac{1}{2}(1.5) + 1/4 (1) + (1/4)(1)$$

$$= 1.25 \text{ bits/symbol} //$$

* The Average Infn Rate of Source is given by.

$$R_s = r_s H \text{ bits/s.}$$

Given. $r_s = 1$ sym/s $\therefore$ $R_s = 1 \times 1.25 = 1.25$ bits/sec

* You are asked to design an Information System which gives the Information every year for about 200 Student passing out with B.E Electronics & Communication degree from a Certain University. The Students can get into one of three fields as given below.

1) Go abroad for higher Studies → A
2) Join MBA (or) Civil Services → B
3) Join Industries In India → C.

Based on the data given below. Construct the model for the source and find Source entropy.

(a) On the average 100 Students are going abroad.

(b) out of 100 going abroad this year. 50 were reported going abroad next year. while 25 Each went to MBA and Civil Services. (or) Joined Industries In India.

c) out of 100 Remaining In India this year, 60 Continued to do so. while 50 went abroad next year.

d) Those going MBA and Civil Services (or) Industry. Could not swap the two fields next year//

Sol :-

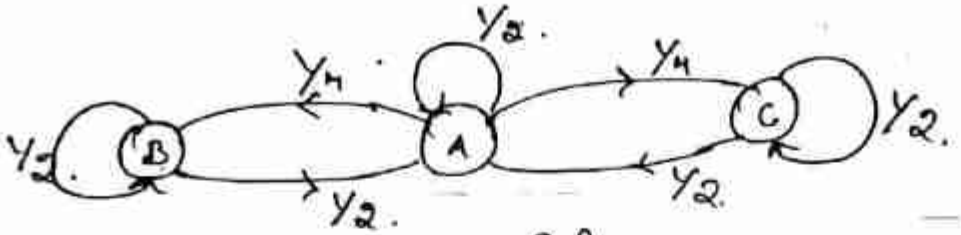


fig: mark off source.

For State A → $P(A) = \frac{1}{2} P(A) + \frac{1}{2} P(B) + \frac{1}{2} P(C)$ ——(1)

State B → $P(B) = \frac{1}{2} P(B) + \frac{1}{4} P(A)$ ——(2)

State C → $P(C) = \frac{1}{2} P(C) + \frac{1}{4} P(A)$ ——(3)

Consider eq no (1)

$$P(A) = \frac{1}{2} [P(A) + P(B) + P(C)]$$

$$P(A) = \frac{1}{2} (1) = \frac{1}{2} //$$

from eq (2)

$$P(B) = \frac{1}{2} P(B) + \frac{1}{4} (\frac{1}{2})$$

$$P(B) - \frac{1}{2} P(B) = \frac{1}{8}$$

$$\frac{1}{2} P(B) = \frac{1}{8}$$

$$P(B) = \frac{1}{4} //$$

from eq (3)

$$P(C) = \frac{1}{2} P(C) + \frac{1}{4} (\frac{1}{2}) = \frac{1}{4} //$$

$P(A) = 1/2$ // $P(B) = 1/4$ // $P(C) = 1/4$ //

The Entropy of Each State is

$$H_A = P_{AA} \log \frac{1}{P_{AA}} + P_{AB} \log \frac{1}{P_{AB}} + P_{AC} \log \frac{1}{P_{AC}}$$

$$= 1/2 \log 2 + \frac{1}{4} \log 4 + 1/4 \log 4$$

$$= 1.5 \text{ bits / msg - symbol} //$$

$$H_B = P_{AB} \log \frac{1}{P_{BA}} + P_{BB} \log \frac{1}{P_{BB}} + P_{BC} \log \frac{1}{P_{BC}}$$

$$= 1/2 \log 2 + \frac{1}{2} \log 2 + 0. = 1 \text{ bits/msg symbol}$$

$$H_C = P_{CA} \log \frac{1}{P_{CA}} + P_{CB} \log \frac{1}{P_{CB}} + P_{CC} \log \frac{1}{P_{CC}}$$

$$= 1/2 \log 2 + 0 + 1/2 \log 2$$

$$= 1 \text{ bits / msg symbol} //$$

The entropy of the Source is Calculated as.

$$H = \sum_{i=1}^{3} P_i H_i = P(A) H_A + P(B) H_B + P(C) H_C$$

$$= (1/2)(1.5) + (1/4)(1) + (1/4)(1)$$

$$H. = 1.25 \text{ bits/msg - symbol} //$$

INFORMATION THEORY AND CODING

5TH SEM E&C



JAYANTHDWIJESH H P **M.tech (DECS)**

Assistant Professor – Dept of E&C

B.G.S INSTITUTE OF TECHNOLOGY (B.G.S.I.T)

B.G Nagara, Nagamangala Tq, Mandya District- 571448

FORMULAS FOR REFERENCE

MODULE - 2 (SOURCE CODING)

- **Entropy of source or Average information content of the source.**

  $H(S) = \sum_{i=1}^{q} P_i \log\left(\frac{1}{P_i}\right)$ bits/symbol or $H(S) = \sum_{K=1}^{N} P_K \log_2 \left(\frac{1}{P_K}\right)$ bits/symbol

- **Average length**

  $L = \sum_{i=1}^{q} P_i \ l_i$ bits/symbol or $L = \sum_{i=1}^{N} P_i \ l_i$ bits/symbol

- **Source or code efficiency**

  $\eta_S = \frac{H(S)}{L} \times 100\%$ or $\eta_C = \frac{H(S)}{L} \times 100\%$

- **Source or code redundancy**

  $R_{\eta S} = 1 - \eta_S = \left(1 - \frac{H(S)}{L}\right) \times 100\%$ or $R_{\eta C} = 1 - \eta_C = \left(1 - \frac{H(S)}{L}\right) \times 100\%$

- **Compute the number of stages required for the encoding operation, which is given by**

  $n = \frac{N-r}{r-1}$ or $\alpha = \frac{q-r}{r-1}$

- **The probability of "0"s and "1"s and "2" s in the code are found using the formulas**

  P (0) = $\frac{1}{L} \sum_{i=1}^{q}$[Number of "0" s in the code for X_i] [p_i] or

  P (0) = $\frac{1}{L} \sum_{i=1}^{N}$[Number of "0" s in the code for X_i] [p_i] .

  P (1) = $\frac{1}{L} \sum_{i=1}^{q}$[Number of "1" s in the code for X_i] [p_i] or

  P (1) = $\frac{1}{L} \sum_{i=1}^{N}$[Number of "1" s in the code for X_i] [p_i] .

  P (2) = $\frac{1}{L} \sum_{i=1}^{q}$[Number of "2" s in the code for X_i] [p_i] or

  P (2) = $\frac{1}{L} \sum_{i=1}^{N}$[Number of "2" s in the code for X_i] [p_i] .

- **The variance of the word length is calculated from**

  $\text{Var}(l_i) = E[(l_i - L)^2 = \sum_{i=1}^{q} P_i (l_i - L)^2$

- **The Smallest integer value of l_i if found using**

  $2^{l_i} \geq \frac{1}{P_i}$ or $l_i \geq \log_2 \frac{1}{P_i}$

- **The average length L_2 of the 2nd extension is given by**

  $L_2 = \sum_{i=1}^{q} P_i \ l_i$ bits/symbol or $L_2 = \sum_{i=1}^{N} P_i \ l_i$ bits/symbol

- The average length L_3 of the 3rd extension is given by

  $L_3 = \sum_{i=1}^{q} P_i \; l_i$ bits/symbol or $L_3 = \sum_{i=1}^{N} P_i \; l_i$ bits/symbol

- The entropy of the 2nd extended source is calculated as

  $H(S^2) = 2\,H(S)$

- The entropy of the 3rd extended source is calculated as

  $H(S^3) = 3H(S)$

- Source or code efficiency of the 2nd extended source is

  $\eta^{(2)}_S = \frac{H(S^2)}{L_2} \times 100\%$ or $\eta^{(2)}_C = \frac{H(S^2)}{L_2} \times 100\%$

- Source or code redundancy of the 2nd extended source is

  $R^{(2)}_{\eta_S} = 1 - \eta^{(2)}_S = \left(1 - \frac{H(S^2)}{L_2}\right) \times 100\%$ or $R^{(2)}_{\eta_C} = 1 - \eta^{(2)}_C = \left(1 - \frac{H(S^2)}{L_2}\right) \times 100\%$

- Source or code efficiency of the 3rd extended source is

  $\eta^{(3)}_S = \frac{H(S^3)}{L_3} \times 100\%$ or $\eta^{(3)}_C = \frac{H(S^3)}{3} \times 100\%$

- Source or code redundancy of the 3rd extended source is

  $R^{(3)}_{\eta_S} = 1 - \eta^{(3)}_S = \left(1 - \frac{H(S^3)}{L_2}\right) \times 100\%$ or $R^{(3)}_{\eta_C} = 1 - \eta^{(3)}_C = \left(1 - \frac{H(S^3)}{L_3}\right) \times 100\%$

- The average length L^3 of the Huffman ternary code is given by

  $L^{(3)} = \sum_{i=1}^{q} P_i \; l_i$ trinits /Msg- symbol or $L^{(3)} = \sum_{i=1}^{N} P_i \; l_i$ trinits / Msg- symbol

- The average length L^4 of the Huffman quaternary code is given by

  $L^{(4)} = \sum_{i=1}^{q} P_i \; l_i$ quaternary digits /Msg- symbol or

  $L^{(4)} = \sum_{i=1}^{N} P_i \; l_i$ quaternary digits / Msg- symbol

- The entropy in ternary units/ message symbol is found by using equation

  $H_3(S) = \frac{H(S)}{\log_2 3}$ ternary units/ message symbol or

  $H_3(S) = \sum_{K=1}^{N} P_K \log_3 \left(\frac{1}{P_K}\right)$ ternary units/ message symbol or

  $H_3(S) = \sum_{i=1}^{q} P_i \log_3 \left(\frac{1}{P_i}\right)$ ternary units/ message symbol

- The entropy in quaternary units/ message symbol is found by using equation

  $H_4(S) = \frac{H(S)}{\log_2 4}$ quaternary units/ message symbol or

$H_4(S) = \sum_{K=1}^{N} P_K \log_4 \left(\frac{1}{P_K}\right)$ quaternary units/ message symbol or

$H_4(S) = \sum_{i=1}^{q} P_i \log_4 \left(\frac{1}{P_i}\right)$ quaternary units/ message symbol

- Source or code efficiency of the ternary is given by

$\eta_{s_{(3)}} = \frac{H_3(S)}{L^{(3)}} \times 100\%$ or $\eta_{c_{(3)}} = \frac{H_3(S)}{L^{(3)}} \times 100\%$ or

$\eta_S = \frac{H_3(S)}{L} \times 100\%$ or $\eta_C = \frac{H_3(S)}{L} \times 100\%$

- Source or code efficiency of the quaternary is given by

$\eta_{s_{(4)}} = \frac{H_4(S)}{L^{(4)}} \times 100\%$ or $\eta_{c_{(4)}} = \frac{H_4(S)}{L^{(4)}} \times 100\%$ or

$\eta_S = \frac{H_4(S)}{L} \times 100\%$ or $\eta_C = \frac{H_4(S)}{L} \times 100\%$

- Source or code redundancy of the ternary is given by

$R_{\eta_{s_{(3)}}} = 1 - \eta_{s_{(3)}} = \left(1 - \frac{H_3(S)}{L^{(3)}}\right) \times 100\%$ or

$R_{\eta_{c_{(3)}}} = 1 - \eta_{c_{(3)}} = \left(1 - \frac{H_3(S)}{L^{(3)}}\right) \times 100\%$ or

$R_{\eta_S} = 1 - \eta_S = \left(1 - \frac{H_3(S)}{L}\right) \times 100\%$ or $R_{\eta_C} = 1 - \eta_C = \left(1 - \frac{H_3(S)}{L}\right) \times 100\%$

- Source or code redundancy of the quaternary is given by

$R_{\eta_{s_{(4)}}} = 1 - \eta_{s_{(4)}} = \left(1 - \frac{H_4(S)}{L^{(4)}}\right) \times 100\%$ or

$R_{\eta_{c_{(4)}}} = 1 - \eta_{c_{(4)}} = \left(1 - \frac{H_4(S)}{L^{(4)}}\right) \times 100\%$ or

$R_{\eta_S} = 1 - \eta_S = \left(1 - \frac{H_4(S)}{L}\right) \times 100\%$ or $R_{\eta_C} = 1 - \eta_C = \left(1 - \frac{H_4(S)}{L}\right) \times 100\%$

Module - 2

SOURCE CODING

Source Encoding is a process by which the output of an Information Source is converted into a binary sequence as its output. The functional block that performs this task in the Communication System is called "SOURCE ENCODER".

The input to the Source Encoder is a Symbol Sequence emitted by Information Source. The Source Encoder assigns variable length binary code words to block of Symbols and produce a binary Sequence as it's output.

PROPERTIES OF CODES OR TYPES OF CODES

May-June-2010(5M), July-2013(6M), Dec-Jan-2010(6M)

1) Block Codes.
2) Non-Singular Codes.
3) Uniquely Decodable Codes.
4) Instantaneous Codes.
5) optimal Codes.

P.T.O.

1) Block codes :- A block code is a code which maps each of the symbol of the source alphabet "S" into some "finite sequence" of code symbols from the code alphabet "X" and each of these finite sequence is called a "code word"

Example :- Consider a source "S" emitting only four symbols which are to be encoded using binary coding

Then. $S = \{ s_1, s_2, s_3, s_4 \}$

& $X = \{ 0, 1 \}$.

A block code can be constructed by considering a pair of code symbols for each source symbol.

Source Symbol	Code-A
s_1	00
s_2	01
s_3	10
s_4	11

(OR)

Block codes :- In this type each symbol will be mapped onto a block of code symbol defined from the code alphabet. The block codes can be either fixed or variable length.

Example :-

00	1
01	01
10	110
11	111

2) Non-Singular Code:- A block code is said to be "non-singular" if and only if all the code words are "distinct" and easily "distinguishable" from one another.

Example:- Consider $S = \{s_1, s_2, s_3, s_4\}$ with $X = \{0, 1\}$. Let an arbitrary code-words for each of these source symbols as shown in below Table.

Source Symbol	Code-B
s_1	0
s_2	00
s_3	01
s_4	11

The code-word given appears to be non-singular. but actually it is not so. This can be confirmed by considering the 2nd extension of these code-words which is shown below.

Source Symbol	CODE-B.	Source Symbol	CODE-B
$s_1 s_1$	00	$s_3 s_1$	010
$s_1 s_2$	000	$s_3 s_2$	0100
$s_1 s_3$	001	$s_3 s_3$	0101
$s_1 s_4$	011	$s_3 s_4$	0111
$s_2 s_1$	000	$s_4 s_1$	100
$s_2 s_2$	0000	$s_4 s_2$	1100
$s_2 s_3$	0001	$s_4 s_3$	1101
$s_2 s_4$	0011	$s_4 s_4$	1111

∴ The code words for $s_1 s_2$ and $s_2 s_1$ are the "same". Thus we can conclude that the 2nd extension of "code-B" are not distinct and hence they becomes "singular".

(or)

Non-Singular Codes:-

A block code is said to be non-singular if all its codewords are distinct.

for Example:-

00	11
01	10
10	11
11	001
non-singular.	Singular. //

3) Uniquely decodable codes:-

A non-singular code is said to be "uniquely decodable" (or) "uniquely deciphenable". If every code-word present in a long received sequence can be uniquely identified.

To understand this property let us consider 2nd extension of code "A" which is uniquely decodable code.

Source symbol	CODE-A	Source symbol	CODE-A
S_1S_1	0000	S_3S_1	1000
S_1S_2	0001	S_3S_2	1001
S_1S_3	0010	S_3S_3	1010
S_1S_4	0011	S_3S_4	1011
S_2S_1	0100	S_4S_1	1100
S_2S_2	0101	S_4S_2	1101
S_2S_3	0110	S_4S_3	1110
S_2S_4	0111	S_4S_4	1111

From the above Table, we can conclude that all codes are non-singular hence they can be uniquely decoded.

OR

Uniquely decodable codes :-

A block code is said to be uniquely decodable if it's nth extension is also non-singular for all finite values of 'n'.

for example.

00	0
01	10
10	110
11	111

4) Instantaneous codes :-

A uniquely decodable code is said to be an instantaneous code if the end of any codeword can be determined without the interpretation of the succeeding symbol. They are also called prefix codes.

5) Optimal Codes :-

Instantaneous codes are called optimal codes if the lengths of the codewords assigned to the symbols are of minimum length.

SOURCE CODING THEOREM:-

Consider an Information Source as shown in below fig. Let S_i be the Symbol Emitted from the Source. The o/p of the source given to the Source encoder whose o/p is the coded Information bit.

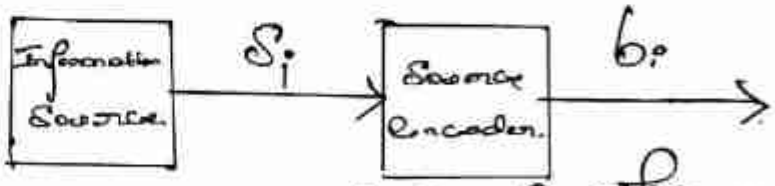


Fig:- Information System.

Let l_i be the length of the codeword for the i^{th} Symbol S_i emitted by the source. The average codeword length "L" is given by.

$$L = \sum_{i=1}^{N} P_i l_i \quad \text{---(1)}$$

The efficiency of the Source Encoder is given by the Ratio of minimum average length to that of average Code word length:

$$\eta_c = \frac{L_{min}}{L} \times 100\% \quad \text{---(2)}$$

As Entropy gives the average Information Conveyed by the Source no Symbol Emitted from the Source can be Represented with Lesser bits than that of its Entropy Thus we have

$$L \geq H(S)$$

$$L_{min} = H(S). \quad \text{---(3)}$$

Thus we require at least H(s) number of bits to represent any of the symbols emitted by the source in order to have lossless communication.

∴ Substituting L_{min} in eq (2) we get.

$$\eta_c = \frac{H(s)}{L} \times 100\%$$

The Redundancy of the source is given by.

$$R_{ns} = 1 - \eta_c \quad //$$

CODE EFFICIENCY AND REDUNDANCY

* The average length 'L' of the code is given by.

$$L = \sum_{i=1}^{q} p_i l_i \quad (or) \quad L = \sum_{i=1}^{n} p_i l_i \text{ bits/msg-symbol} \quad -(1)$$

where $p_i \rightarrow p_1, p_2, \ldots p_q$ the respective probabilities of the source symbols $s_1, s_2, \ldots s_q$.

$l_i \rightarrow l_1, l_2, \ldots l_q$ the respective coword lengths in bits of the code words of the symbols $s_1, s_2 \ldots s_q$.

* The entropy is given by.

$$H(s) = \sum_{i=1}^{q} p_i \log \frac{1}{p_i} \text{ bits/msg-symbol} \quad -(2)$$

The probabilities of source symbols P_i and the codeword lengths l_i of the code-words, the average length "L" is calculated using eq. (1) and the entropy using eq (2).

$$L \geq H(S) \text{ for binary codes} \quad -(3)$$

and $L \geq H_r(S)$ for r-ary codes $-(4)$

where $H_r(S)$ = entropy in r-ary units/msg symbol is given by

$$H_r(S) = \frac{H(S)}{\log_2 r}.$$

where $H(S)$ = is measured in bits/msg-symbol.

r = number of different symbols used in code-alphabet.

The "CODE EFFICIENCY" denoted by η_c is defined as

$$\eta_c = \frac{H(S)}{L} \rightarrow \text{for binary codes.}$$

and $\eta_c = \frac{H_r(S)}{L} \rightarrow$ for r-ary codes,

Since the denominator "L" is always greater than (or) equal to numerator, we find that

$$\eta_c \leq 100\%$$

The "CODE REDUNDANCY" denoted by R_{nc}. is defined as

$$R_{nc} = 1 - \eta_c //$$

NOTE:-

η_c and R_{nc} are usually expressed as percentages.

SHANNON'S FIRST THEOREM [NOISELESS CODING THEOREM] :-

* Shannon Suggested That the Length "l_i" Can be known Using The formula. $l_i = \log_{2r} \frac{1}{P_i}$ ——— (1)

* Equation (1) above Signifies That Larger than value of Symbol probability Smaller value of "l_i" and a few Coading digits will be Sufficient.

* If l_i happens to be a fraction as Calculated from eq (1) then it is Rounded-off to next integer as given by.

$$\log_{2r} \frac{1}{P_i} \leq l_i \leq 1 + \log_{2r} \frac{1}{P_i} \quad \text{——— (2)}$$

* Using the property of Logarithms eq (2) Can be written as

$$\frac{\log_2 \frac{1}{P_i}}{\log_2 2r} \leq l_i \leq 1 + \frac{\log_2 \frac{1}{P_i}}{\log_2 2r}$$

* multiplying throughout by P_i and Taking Summation for all "i" varying from 1 to q we get.

$$\frac{1}{\log_2 2r} \sum_{i=1}^{q} P_i \log_2 \frac{1}{P_i} \leq \sum_{i=1}^{q} l_i P_i \leq \sum_{i=1}^{q} P_i + \frac{1}{\log_2 2r} \sum_{i=1}^{q} P_i \log_2 \frac{1}{P_i}$$

where. $\sum_{i=1}^{q} P_i \log_2 \frac{1}{P_i} = H(s)$ [entropy]

$\sum_{i=1}^{q} P_i l_i = L$ [average length]

$\sum_{i=1}^{q} P_i = 1$ [sum of all prob Equal to one]

$$\frac{1}{\log_2 r} H(s) \leq L \leq 1 + H(s) \frac{1}{\log_2 r} \quad \text{---} (3)$$

Entropy of an r-ary units/message Symbol is given by

$$H_r(s) = \frac{H(s)}{\log_2 r} \quad \text{---} (4)$$

Eq (4) Sub into eq (3)

$$H_r(s) \leq L \leq 1 + H_r(s)$$

$\therefore L \geq H_r(s)$.

Shannon Suggested that equation (3) still holds good for nth Extension Source "S" to get better efficiency.

$$\frac{H(s^n)}{\log_2 r} \leq L_n \leq 1 + \frac{H(s^n)}{\log_2 r} \quad \text{---} (5)$$

where L_n = average Length of code-words of nth Extended Source Symbols.

The entropy of the nth Extended Source is given by

$$H(s^n) = nH(s) \quad \text{---} (6)$$

Substituting eq (6) in to eq (5)

$$\frac{nH(s)}{\log_2 r} \leq L_n \leq 1 + \frac{nH(s)}{\log_2 r} \quad \text{---} (7)$$

Dividing Throughout by 'n'..

$$\frac{H(s)}{\log_2 r} \leq \frac{L_n}{n} \leq \frac{1}{n} + \frac{H(s)}{\log_2 r}$$

$$\frac{H(S)}{\log_2 r} \leq \frac{L_n}{n} \leq \frac{1}{n} + \frac{H(S)}{\log_2 r}$$

NOTE: $H_r(S) = \frac{H(S)}{\log_2 r}$

$$H_r(S) \leq \frac{L_n}{n} \leq \frac{1}{n} + H_r(S) \quad \text{--- (8)}$$

Taking limits as $n \to \infty$. it follows from the first inequality of Equation (8)

$$\lim_{n\to\infty} \frac{L_n}{n} = H_r(S) \; //$$

where 'L' is the average length of code-words for the base source 'S' then for all values of Extension 'n'

$$\therefore \frac{L_n}{n} \geq L \; //$$

SHANNON'S FIRST ENCODING THEOREM

OR SHANNON'S ENCODING ALGORITHM

July-2009(4M), Dec-2010(4M), July-2013(6M), July-2009(4M), July-2011(4M) Jan-2010(4M)

Shannon has proposed one of the first algorithms on source encoding. The encoding procedure is as follows.

STEP1 :- Given the source alphabet 'S' and the corresponding probabilities 'P' for a given information source.

STEP2 :- Arrange the probabilities in the non-increasing order.

STEP3 :- Compute the length "l_i" for the codeword corresponding to each symbol "S_i" given by.

$$l_i \geq \log_2 \frac{1}{P_i} \; //$$

where P_i is the probability of symbol S_i

Step 4 :- Define the following parameters from the probability set

$q_1 = 0$

$q_2 = P_1 = q_1 + P_1$

$q_3 = P_1 + P_2 = q_2 + P_2$

$q_4 = P_1 + P_2 + P_3 = q_3 + P_3$

= = = = =

= = = = =

= = = = =

= = = = =

$q_{N+1} = 1$

where 'N' is the Total number of Symbol in the Source alphabet.

Step 5 :- Expand q_i in binary till l_i number of places after decimal point.

Step 6 :- The numbers after decimal places in the binary Representation of 'q_i' are the Codewords for the Corresponding Symbol S_i

07

P.T.O.

Step no-1 :-

List the Source Symbols in the order of non-Increasing probabilities:

given $S = \{s_1, s_2 \cdots s_q\}$ with.

$P = \{p_1, p_2, \cdots p_q\}$; $p_1 \geq p_2 \geq \cdots \geq p_q$

Step 2 :- Compute the Sequences.

$\alpha_1 = 0$

$\alpha_2 = p_1 = p_1 + \alpha_1$

$\alpha_3 = p_2 + p_1 = p_2 + \alpha_2$

$\alpha_4 = p_3 + p_2 + p_1 = p_3 + \alpha_3$

$\alpha_{q+1} = p_q + p_{q-1} + \cdots + p_2 + p_1 = p_q + \alpha_q$

Step 3 :-

Determine the Smallest Integer value of l_i using the inequality.

$2^{l_i} \geq \frac{1}{p_i}$ for all $i = 1, 2, \cdots q$.

Step 4 :-

Expand the decimal number α_i in binary form upto l_i places neglecting Expansion beyond "l_i" places.

Step 5 :-

Remove the binary point to get the desired code

* Consider a source with source alphabet $S = (A, B, C, D)$ with the corresponding probabilities $P = (0.1, 0.2, 0.3, 0.4)$ find the codewords for symbols using Shannon's algorithm. find the source efficiency and Redundancy.

Sol :-

STEP 1 : Arrange the probabilities in the non-increasing order and corresponding symbol to get.

$P = (0.4, 0.3, 0.2, 0.1)$ $S = (D, C, B, A)$.

STEP 2 :- find the minimum value of l_i such that

$$l_i \geq \log_2 \frac{1}{P_i}$$

$l_1 \geq \log_2 \frac{1}{P_1} = \log_2 \frac{1}{0.4} = 1.3219.$

$\therefore l_1 = 1.3219.$ ← NOTE 1.3219 is near to 2.0

$= 2$ //

$l_2 \geq \log_2 \frac{1}{P_2} = \log_2 \frac{1}{0.3} = 1.736. = 2$ //

$l_3 \geq \log_2 \frac{1}{P_3} = \log_2 \frac{1}{0.2} = 2.321 = 3$ //.

$l_4 \geq \log_2 \frac{1}{P_4} = \log_2 \frac{1}{0.1} = 3.321 = 4$ //.

STEP 3 :- calculate the parameter q_i

$q_1 = 0$ ← NOTE q_1 is always zero.

$q_2 = P_1 = 0.4$ //

$q_3 = P_1 + P_2 = 0.4 + 0.3 = 0.7$ //.

$q_4 = P_1 + P_2 + P_3 = 0.4 + 0.3 + 0.2 = 0.9$ //.

$q_5 = P_1 + P_2 + P_3 + P_4 = 0.4 + 0.3 + 0.2 + 0.1 = 1$ //.

STEP 4 :- Represent q_1, q_2, q_3 and q_4 in binary upto l_i places after decimal points.

$q_1 = (0.0)_{10} = (0.00)_2$ //.

$q_2 = (0.4)_{10}$.

$0.4 \times 2 = 0.8$ with Carry 0

$0.8 \times 2 = 0.6$ with Carry 1

$0.6 \times 2 = 0.2$ with Carry 1

$\therefore q_2 = (0.4)_{10} = (0.011)_2$ //

$q_3 = (0.7)_{10}$.

$0.7 \times 2 = 0.4$ with Carry 1

$0.4 \times 2 = 0.8$ with Carry 0.

$0.8 \times 2 = 0.6$ with Carry 1.

$\therefore q_3 = (0.7)_{10} = (0.101)_2$

$q_4 = (0.9)_{10}$.

$0.9 \times 2 = 0.8$ with Carry 1

$0.8 \times 2 = 0.6$ with Carry 1.

$0.6 \times 2 = 0.2$ with Carry 1

$0.2 \times 2 = 0.4$ with Carry 0.

$\therefore q_4 = (0.9)_{10} = (0.1110)_2$ //

$\therefore q_1 = 0$ and $l_1 = 2$. code for D $\rightarrow$ 00

$\therefore q_2 = (0.01)_2$ and $l_2 = 2$. code for C $\rightarrow$ 01

$\therefore q_3 = (0.101)_2$ and $l_3 = 3$ code for B $\rightarrow$ 101

$\therefore q_4 = (0.1110)_2$ and $l_4 = 4$ code for A $\rightarrow$ 1110.

SYMBOL	PROBABILITY (P_i)	CODEWORD	LENGTH (L_i)
D	0.4	00	2
C	0.3	01	2
B	0.2	101	3
A	0.1	1110	4

Table : Codewords using Shannon's algorithm.

Step 5 : $\eta_c = \frac{H(s)}{\bar{L}} \times 100$.

$H(s) = \sum_{k=1}^{n} p_k \log_2 \frac{1}{p_k} = 1.8464$ bits/symbol

$\bar{L} = \sum_{i=1}^{n} p_i l_i = 0.4 \times 2 + 0.3 \times 2 + (0.2 \times 3) + 0.1 \times 4 = 2.4$ bits/symbol

(prob, length)

$\therefore \eta_c = \frac{1.8464}{2.4} \times 100 = 76.93\%$ $\therefore R_{\eta_c} = 1 - \eta_c = 1 - 0.7693$

$= 0.2306 \times 100 = 23.07\%$

OR

Sol : $p = (0.4, 0.3, 0.2, 0.1)$. $S = (D, C, B, A)$.

given $2^{l_i} \geq \frac{1}{p_i}$ for all $i = 1, 2, \ldots, q$.

$i=1$ $2^{l_1} \geq \frac{1}{p_1} \geq \frac{1}{0.4} \geq 2.5$

$\therefore l_1 \geq \frac{2.5}{\log 2} = 1.25$

Smallest value $l_1 = 2$.

$i=2$. $2^{l_2} \geq \frac{1}{p_2}$. $\geq \frac{1}{0.3}$ $2^{l_2} \geq 3.333$ $l_2 \geq 2$

$i=3$. $2^{l_3} \geq \frac{1}{p_3}$ $\geq \frac{1}{0.2}$ $2^{l_3} \geq 5$ $l_3 = 3$

$i=4$. $2^{l_4} \geq \frac{1}{p_4}$. $\geq \frac{1}{0.1}$ $2^{l_4} \geq 10$. $l_4 \geq 4$

* The following sequence of α's are computed.

$\alpha_1 = 0$. $\alpha_2 = p_1 = 0.4$ $\alpha_3 = p_1 + p_2 = 0.4 + 0.3 = 0.7$

$\alpha_4 = p_1 + p_2 + p_3 = 0.4 + 0.3 + 0.2 = 0.9$ $\alpha_5 = p_1 + p_2 + p_3 + p_4 = 1$

* The decimal numbers α_i are expanded in binary from upto l_i places as given below.

$\alpha_1 = 0$. and $l_1 = 2$. $\therefore$ Code for $S_1 \rightarrow 00$.

$\alpha_2 = (0.4)_{10}$.

$0.4 \times 2 = 0.8$ with Carry 0.

$0.8 \times 2 = 0.6$ with Carry 1.

$0.6 \times 2 = 0.2$ with Carry 1

$\therefore \alpha_2 = (0.4)_{10} = (0.011)_2$,

$\alpha_2 = (0.011)_2$ and $L_2 = 2$. Code for $S_2 \rightarrow 01$

$\alpha_3 = (0.7)_{10}$

$0.7 \times 2 = 0.4$ with Carry 1

$0.4 \times 2 = 0.8$ with Carry 0

$0.8 \times 2 = 0.6$ with Carry 1

$\therefore \alpha_3 = (0.7)_{10} = (0.101)_2$

$\alpha_3 = (0.101)_2$ & $L_3 = 3$

$\therefore$ code for $S_3 \rightarrow 101$

$\alpha_4 = (0.9)_{10}$.

$0.9 \times 2 = 0.8$ with Carry 1

$0.8 \times 2 = 0.6$ with Carry 1

$0.6 \times 2 = 0.2$ with Carry 1

$0.2 \times 2 = 0.4$ with Carry 0.

$\alpha_4 = (0.9)_{10} = (0.1110)_2$

$\alpha_4 = (0.1110)$ & $L_4 = 4$. $\therefore$ Code for $S_4 \rightarrow 111$

SYMBOL	PROBABILITY P_i	CODEWORD	LENGTH l_i
D	0.4	00	2
C	0.3	01	2
B	0.2	101	3
A	0.1	1110	4

Table : Code Table

Average length "L" is given by.

$$L = \sum_{i=1}^{5} P_i l_i = (0.4)(2) + (0.3)(2) + (0.2)(3) + (0.1)(4) + \ldots$$

$$= 2.4 \text{ bits / Symbol}$$

$$H(S) = \sum_{i=1}^{5} P_i \log \frac{1}{P_i} = 0.4 \log\left(\frac{1}{0.4}\right) + 0.3 \log\left(\frac{1}{0.3}\right) + 0.2 \log\left(\frac{1}{0.2}\right) + 0.1 \log\left(\frac{1}{0.1}\right)$$

$$= 1.84642 \text{ bits / Symbol}$$

code efficiency $\eta_c = \frac{H(s)}{L} = \frac{1.846M}{2.M} = 0.7693 \times 100$

$= 76.93\%$ //

$R_{\eta_c} = 1 - \eta_c = 1 - 0.7693 = 0.2306 \times 100$

$= 23.06$ //

---

* Construct binary code for the following source using Shannon's binary encoding procedure.

$S = \{S_1, S_2, S_3, S_4, S_5\}$

$P = \{0.4, 0.25, 0.15, 0.12, 0.08\}$

July/Aug-2003, 2018-June [10M].

Sol:- The basic source.

S_1	S_2	S_3	S_4	S_5
0.4	0.25	0.15	0.12	0.08

* calculate parameters q_i

$q_1 = 0$ //

$q_2 = p_1 = 0.4$ //

$q_3 = p_1 + p_2 = 0.4 + 0.25 = 0.65$ //

$q_4 = p_1 + p_2 + p_3 = 0.4 + 0.25 + 0.15 = 0.8$ //

$q_5 = p_1 + p_2 + p_3 + p_4 = 0.4 + 0.25 + 0.15 + 0.12 = 0.92$ //

$q_6 = p_1 + p_2 + p_3 + p_4 + p_5 = 0.4 + 0.25 + 0.15 + 0.12 + 0.08 = 1$ //

* The minimum values of l_i such that

$$l_i \geq \log_2 \frac{1}{p_i}$$

$l_1 \geq \log_2 \frac{1}{p_1} = \log_2 \frac{1}{0.4} = 1.3219$ // $\therefore l_1 = 2$ //

$l_2 \geq \log_2 \frac{1}{p_2} = \log_2 \frac{1}{0.25} = 2$ // $\therefore l_2 = 2$ //

$l_3 = \log_2\left(\frac{1}{0.15}\right) = 2.936$ // $\therefore l_3 = 3$ //

$l_4 = \log_2 \frac{1}{0.12} = 3.058$ $\therefore l_4 = 4$ //

$l_5 = \log_2 \frac{1}{0.08} = 3.64$ $\therefore l_5 = 4$ //

---

* $q_1 = 0$ //

$q_2 = (0.4)_{10}$

$0.4 \times 2 = 0.8$ with Carry 0.

$0.8 \times 2 = 0.6$ with Carry 1.

$0.6 \times 2 = 0.2$ with Carry 1.

$q_2 = (0.4)_{10} = (0.011)_2$ //

$q_3 = (0.65)_{10}$

$0.65 \times 2 = 0.3$ with Carry 1

$0.3 \times 2 = 0.6$ with Carry 0

$0.6 \times 2 = 0.2$ with Carry 1

$q_3 = (0.65)_{10} = (0.101)_2$ //

$q_4 = (0.8)_{10}$.

$0.8 \times 2 = 0.6$ with Carry 1.

$0.6 \times 2 = 0.2$ with Carry 1.

$0.2 \times 2 = 0.4$ with Carry 0.

$0.4 \times 2 = 0.8$ with Carry 0.

$q_4 = (0.8)_{10} = (0.1100)_2$

$q_5 = (0.92)_{10}$.

$0.92 \times 2 = 0.84$ with Carry 1.

$0.84 \times 2 = 0.68$ with Carry 1

$0.68 \times 2 = 0.36$ with Carry 1

$0.36 \times 2 = 0.72$ with Carry 0

$q_5 = (0.92)_{10} = (0.1110)_2$ //

$q_1 = 0$. and $l_1 = 2$. $\therefore$ code for $S_1 \rightarrow 00$.

$q_2 = (0.011)_2$ and $l_2 = 2$. $\therefore$ code for $S_2 \rightarrow 01$

$q_3 = (0.101)_2$ and $l_3 = 3$ $\therefore$ code for $S_3 \rightarrow 101$.

$q_4 = (0.1100)_2$ and $l_4 = 4$. $\therefore$ code for $S_4 \rightarrow 1100$.

$q_5 = (0.1110)_2$ and $l_5 = 4$. $\therefore$ code for $S_5 \rightarrow 1110$ //.

SOURCE SYMBOLS.	PROBABILITY.	CODEWORD.	LENGTH
S_1	0.4	00	2
S_2	0.25	01	2
S_3	0.15	101	3
S_4	0.12	1100	4
S_5.	0.08.	1110	4.

Table: codeword, Coding & Shannon's algorithm

*Apply Shannon's Encoding (binary) algorithm to the following messages.

S_1 S_2 S_3

0.5 0.3 0.2.

1). Find code efficiency and Redundancy?

2). If the same technique is applied to the 2nd order extension of the source, how much will the code efficiency be improved?

DEC-2012/JAN-2014 [10M]

Sol: $P = (0.5, 0.3, 0.2)$ $S = \{S_1, S_2, S_3\}$.

* The minimum values of l_i such that.

$$l_i \geq \log_2 \frac{1}{P_i}.$$

$$l_1 \geq \log_2 \frac{1}{P_1} = \log_2 \frac{1}{0.5} = 1 \therefore l_1 = 1$$

$$l_2 \geq \log_2 \frac{1}{P_2} = \log_2 \frac{1}{0.3} = 1.736 \therefore l_2 = 2$$

$$l_3 \geq \log_2 \frac{1}{P_3} = \log_2 \frac{1}{0.2} = 2.321 \therefore l_3 = 3$$

* Calculate the parameter q_i.

$q_1 = 0$ // $\quad q_3 = p_1 + p_2 = 0.8$ //

$q_2 = p_1 = 0.5$ // $\quad q_4 = p_1 + p_2 + p_3 = 1$ //

* Represent q_1, q_2 and q_3 in binary up to 3 place after decimal points.

$q_1 = 0$ //

$q_2 = (0.5)_{10}$.

$0.5 \times 2 = 0.00$ with Carry 1

$q_2 = (0.5)_{10} = (0.1)_2$ //.

$q_3 \equiv (0.8)_{10}$.

$0.8 \times 2 = 0.6$ with Carry 1

$0.6 \times 2 = 0.2$ with Carry 1

$0.2 \times 2 = 0.4$ with Carry 0

$q_3 = (0.8)_{10} = (0.110)_2$

* $q_1 = 0$ and $l_1 = 1$ ∴ code for $s_1 \rightarrow 0$.

$q_2 = (0.1)_2$ and $l_2 = 2$. ∴ code for $s_2 \rightarrow 10$.

$q_3 = (0.110)_2$ and $l_3 = 3$. ∴ code for $s_3 = 110$.

SYMBOL.	PROBABILITY	CODEWORD	LENGTH
s_1	0.5	0	1
s_2	0.3	10	2
s_3	0.2.	110.	3

Table : Codewords using Shannons algorithm.

efficiency, $\eta = \eta_c = \frac{H(s)}{L} \times 100\%$.

$$H(s) = \sum_{i=1}^{2} p_i \log \frac{1}{p_i} = \sum_{k=1}^{N} p_k \log_2 \frac{1}{p_k}$$

$$= 0.5 \log_2 \frac{1}{0.5} + 0.3 \log_2 \frac{1}{0.3} + 0.2 \log_2 \frac{1}{0.2}$$

$$= 0.5 \times 1 + 0.3 (1.7369) + 0.2 (2.3219)$$

$$= 1.4854 \text{ bits/symbol}$$ //

$L = \sum_{i=1}^{n} P_i l_i = (0.5\times1) + (0.3\times2) + (0.2\times3) = 1.7$ bits/symbol.

$\therefore \eta_c = \frac{1.4854}{1.7} \times 100\% = 87.37\%$ //

Redundancy $R_{\eta_c} = 1 - \eta_c \times 100 = (1 - 0.87376)100\% = 12.763\%$ //

2nd order extension:-

Symbol	probability
$S_1S_1 = 0.5\times0.5$	0.25
$S_1S_2 = 0.5\times0.3$	0.15
$S_1S_3 = 0.5\times0.2$	0.10
$S_2S_1 = 0.3\times0.5$	0.15
$S_2S_2 = 0.3\times0.3$	0.09
$S_2S_3 = 0.3\times0.2$	0.06
$S_3S_1 = 0.2\times0.5$	0.10
$S_3S_2 = 0.2\times0.3$	0.06
$S_3S_3 = 0.2\times0.2$	0.04

Step 1:- Arrange the probabilities in the non-increasing order and the corresponding symbols.

$P = (0.25, 0.15, 0.15, 0.10, 0.10, 0.09, 0.06, 0.06, 0.04)$

$S = (S_1S_1, S_1S_2, S_2S_1, S_1S_3, S_3S_1, S_2S_2, S_2S_3, S_3S_2, S_3S_3)$

Step 2:- Find the minimum value of l_i such that

$l_i \geq \log_2 \frac{1}{P_i}$

$l_1 \geq \log_2 \frac{1}{P_1} = \log_2 \frac{1}{0.25} \Rightarrow l_1 = 2$ //

$l_2 \geq \log_2 \frac{1}{P_2} = \log_2 \frac{1}{0.15} \Rightarrow l_2 = 3$ //

$l_3 \geq \log_2 \frac{1}{P_3} = \log_2 \frac{1}{0.15} \Rightarrow l_3 = 3$ //

$l_4 \geq \log_2 \frac{1}{P_4} = \log_2 \frac{1}{0.10} \Rightarrow l_4 = 4$ //

$l_5 \geq \log_2 \frac{1}{P_5} = \log_2 \frac{1}{0.10} \Rightarrow l_5 = 4$ //

$l_6 \geq \log_2 \frac{1}{P_6} = \log_2 \frac{1}{0.09} \Rightarrow l_6 = 4$ //

$l_7 \geq \log_2 \frac{1}{P_7} = \log_2 \frac{1}{0.06} \Rightarrow l_7 = 5$

$l_8 \geq \log_2 \frac{1}{P_8} = \log_2 \frac{1}{0.06} \Rightarrow l_8 = 5$ //

$l_9 \geq \log_2 \frac{1}{P_9} = \log_2 \frac{1}{0.04} \Rightarrow l_9 = 5$

Step 3 :- Finding the parameter q_i

$q_1 = 0$.

$q_2 = P_1 + P_2 = 0.25$ //

$q_3 = P_1 + P_2 = 0.40$

$q_4 = P_1 + P_2 + P_3 = 0.55$ //

$q_5 = P_1 + P_2 + P_3 + P_4 = 0.65$

$q_6 = P_1 + P_2 + P_3 + P_4 + P_5 = 0.95$

$q_7 = P_1 + P_2 + P_3 + P_4 + P_5 + P_6 = 0.84$ //

$q_8 = P_1 + P_2 + P_3 + P_4 + P_5 + P_6 + P_7 = 0.90$ //

$q_9 = P_1 + P_2 + P_3 + P_4 + P_5 + P_6 + P_7 + P_8 = 0.96$

$q_{10} = P_1 + P_2 + P_3 + P_4 + P_5 + P_6 + P_7 + P_8 + P_9 = 1$ //

Step 4 :-

$\alpha_1 = q_1 = 0$. $\therefore$ $q_1 = (0)_2$ //

$q_2 = (0.25)_{10}$.

$0.25 \times 2 = 0.50$ with Carry 0.

$0.50 \times 2 = 0.00$ with Carry 1

$\therefore q_2 = (0.25)_{10} = (0.01)_2$ //

$q_3 = (0.4)_{10}$.

$0.4 \times 2 = 0.8$ with carry 0

$0.8 \times 2 = 0.6$ with Carry 1

$0.6 \times 2 = 0.2$ with Carry 1

$\therefore q_3 = (0.4)_{10} = (0.011)_2$

$q_4 = (0.55)_{10}$.

$0.55 \times 2 = 0.10$ with Carry 1
$0.10 \times 2 = 0.20$ with Carry 0
$0.20 \times 2 = 0.40$ with Carry 0
$0.40 \times 2 = 0.80$ with Carry 0

$\therefore q_4 = (0.55)_{10} = (0.1000)_2$ //.

$q_5 = (0.65)_{10}$

$0.65 \times 2 = 0.3$ with Carry 1
$0.3 \times 2 = 0.6$ with Carry 0.
$0.6 \times 2 = 0.2$ with Carry 1
$0.2 \times 2 = 0.4$ with Carry 0

$\therefore q_5 = (0.65)_{10} = (0.1010)_2$ //.

$q_6 = (0.75)_{10}$.

$0.75 \times 2 = 0.50$ with Carry 1
$0.50 \times 2 = 0.00$ with Carry 1

$\therefore q_6 = (0.75)_{10} = (0.11)_2$

$q_7 = (0.84)_{10}$.

$0.84 \times 2 = 0.68$ with Carry 1
$0.68 \times 2 = 0.36$ with Carry 1
$0.36 \times 2 = 0.72$ with Carry 0.
$0.72 \times 2 = 0.44$ with Carry 1
$0.44 \times 2 = 0.88$ with Carry 0.

$\therefore q_7 = (0.84)_{10} = (0.11010)_2$.

$q_8 = (0.9)_{10}$

$0.9 \times 2 = 0.80$ with Carry 1
$0.80 \times 2 = 0.60$ with Carry 1
$0.60 \times 2 = 0.20$ with Carry 1
$0.20 \times 2 = 0.40$ with Carry 0
$0.40 \times 2 = 0.80$ with Carry 0.

$\therefore q_8 (0.9)_{10} = (0.11100)_2$

$q_9 = (0.96)_{10}$

$0.96 \times 2 = 0.92$ with Carry 1
$0.92 \times 2 = 0.84$ with Carry 1
$0.84 \times 2 = 0.68$ with Carry 1
$0.68 \times 2 = 0.36$ with Carry 1
$0.36 \times 2 = 0.72$ with Carry 0

$\therefore q_9 = (0.96)_{10} = (0.11110)_2$

=

Step no 5 :

$q_1 = 0$. and $L_1 = 2$ bits ∴ code from $S_1S_1 \rightarrow 00$.

$q_2 = (0.01)_2$ and $L_2 = 3$ ∴ code from $S_1S_2 \rightarrow 010$.

$q_3 = (0.011)_2$ and $L_3 = 3$ ∴ code from $S_2S_1 \rightarrow 011$

$q_4 = (0.1000)_2$ and $L_4 = 4$ ∴ code from $S_1S_3 \rightarrow 1000$.

$q_5 = (0.1010)_2$ and $L_5 = 4$ ∴ code from $S_3S_1 \rightarrow 1010$.

$q_6 = (0.11)_2$ and $L_6 = 4$ ∴ code from $S_2S_2 \rightarrow 1100$.

$q_7 = (0.11010)_2$ and $L_7 = 5$ ∴ code from $S_2S_3 \rightarrow 11010$.

$q_8 = (0.11100)_2$ and $L_8 = 5$ ∴ code from $S_3S_2 \rightarrow 11100$.

$q_9 = (0.11110)_2$ and $L_9 = 5$ ∴ code from $S_3S_3 \rightarrow 11110$

Source symbol	probability (P_i)	Code	Length (L_i)
S_1S_1	0.25	00	2
S_1S_2	0.15	010	3
S_2S_1	0.15	011	3
S_2S_3	0.10	1000	4
S_3S_1	0.10	1010	4
S_2S_2	0.09	1100	4
S_1S_3	0.06	11010	5
S_3S_2	0.06	11100	5
S_3S_3	0.04	11110	5

∴ the average length is given by

$$L(\text{or}) = L_2 = \sum_{i=1}^{9} P_i L_i$$

$$= (0.25)(2) + (0.15)(3) + (0.15)(3) + (0.1)(4) + (0.1)(4) + (0.09)(4) + (0.06)(5) + (0.06)(5) + (0.04)(5)$$

∴ $L_2 = 3.36$ bits / msg symbol //

$H(S^2) = \sum_{k=1}^{N} P_k \log_2 \frac{1}{P_k}$ (or) $\sum_{i=1}^{q} p_i \log \frac{1}{p_i}$

(or)

$H(S^2) = 2H(S) = 2 \times 1.4855$ bits (or) bits/symbol

$= 2.971$ bits/symbol //

∴ code efficiency η_S (or) $\eta_C = \eta_S^2$ (or) $\eta_C^2 = \left(\frac{H(S^2)}{L_2}\right) \times 100\%$

$\eta_C^2 = \frac{2.971}{3.36} \times 100\%$

$= 88.42\%$ //

∴ Code Redundancy R_S^2 (or) $R_C^2 = (1 - \eta_C^2) \times 100$

$= (1 - 0.8842) \times 100$

$= 0.11577 \times 100$

$= 11.577\%$ //

(or)

$= 11.58$ //

∴ code efficiency Improved by $= \eta_C^2 - \eta_C$ (or) $\eta_S^2 - \eta_S$

$= (88.42 - 87.38)\%$

$= 1.04\%$ (Improved) //

∴ We have $L_2 = 3.36$ bits/message symbol

Consider $\frac{L_2}{2} = \frac{3.36}{2} = 1.68$ bits/symbol //

and $L = 1.70$ bits/symbol //

∴ $\frac{L_2}{2} < L$ ... Satisfied //

* The source emits the message consisting of two symbols each. Three messages and their probabilities are given in below Table. Design the Source Encoder using Shannon's Encoding algorithm and also find Encoder efficiency.

message M_i	probability P_i
AA	9/32
AC	3/32
CC	1/16
CB	3/32
CA	3/32
BC	3/32
BB	9/32

EC/TE. July/Aug-2002 [8M], July-2013 [8M]

Sol:-

step 1:-

AA	AB	AC	CB	CA	BC	CC
9/32	9/32	3/32	3/32	3/32	3/32	1/16

step 2:

$q_1 = \alpha_1 = 0$.

$q_2 = P_1 = 9/32 = 0.28125$

$q_3 = P_1 + P_2 = 9/32 + 9/32 = \frac{18}{32} = 0.5625$

$q_4 = P_1 + P_2 + P_3 = \frac{18}{32} + \frac{3}{32} = \frac{21}{32} = 0.65625$

$q_5 = P_1 + P_2 + P_3 + P_4 = \frac{21}{32} + \frac{3}{32} = \frac{24}{32} = 0.75$

$q_6 = \frac{24}{32} + \frac{3}{32} = \frac{27}{32} = 0.84375$ //

$q_7 = \frac{27}{32} + \frac{3}{32} = \frac{30}{32} = 0.9375$ //

$q_8 = \frac{30}{32} + \frac{1}{10} = 1$ //

Step No 3 :-

$l_i \geq \log_2 \frac{1}{P_i}$

$l_1 \geq \log_2 \frac{1}{P_1} \geq \log_2 \frac{32}{9} = 1.830$

$= 2$ //

$l_2 \geq \log_2 \frac{1}{P_2} \geq \log_2 \frac{32}{9} = 1.830 = 2$ //

$l_3 \geq \log_2 \frac{1}{P_3} \geq \log_2 \frac{32}{3} = 3.4150 = 4$ //

$l_4 \geq \log_2 \frac{1}{P_4} = \log_2 \frac{32}{3} = 3.4150 = 4$ //

$l_5 \geq \log_2 \frac{1}{P_5} = \log_2 \frac{32}{3} = 3.4150 = 4$ //

$l_6 \geq \log_2 \frac{1}{P_6} = \log_2 \frac{32}{3} = 3.4150 = 4$ //

$l_7 \geq \log_2 \frac{1}{P_7} = \log_2 16 = 4$ //

Step No 4 :-

$q_1 = 0.$

$q_2 = (0.28125)_{10}$

$0.28125 \times 2 = 0.5625$ with Carry 0

$0.5625 \times 2 = 0.125$ with Carry 1

$0.125 \times 2 = 0.25$ with Carry 0

$q_2 = (0.28125)_{10} = (0.010)_2$ //

$q_3 = (0.5625)_{10}$

$0.5625 \times 2 = 0.125$ with Carry 1,
$0.125 \times 2 = 0.25$ with Carry 0.
$0.25 \times 2 = 0.5$ with Carry 0.
$0.5 \times 2 = 0$ with Carry 1.

$q_3 = (0.5625)_{10} = (0.1001)_2$ //

$q_4 = (0.65625)_{10}$.

$0.65625 \times 2 = 0.3125$ with Carry 1
$0.3125 \times 2 = 0.625$ with Carry 0.
$0.625 \times 2 = 0.25$ with Carry 1
$0.25 \times 2 = 0.5$ with Carry 0.

$q_4 = (0.65625)_{10} = (0.1010)_2$ //

$q_5 = (0.75)_{10}$.

$0.75 \times 2 = 0.5$ with Carry 1.
$0.5 \times 2 = 0$ with Carry 1

$q_5 = (0.75)_{10} = (0.11)_2 = (0.1100)_2$ //

$q_6 = (0.84375)_{10}$.

$0.84375 \times 2 = 0.6875$ with Carry 1
$0.6875 \times 2 = 0.375$ with Carry 1
$0.375 \times 2 = 0.75$ with Carry 0.
$0.75 \times 2 = 0.5$ with Carry 1.

$q_6 = (0.84375)_{10} = (0.1101)_2$ //

$q_7 = (0.9375)_{10}$

$0.9375 \times 2 = 0.875$ with Carry 1
$0.875 \times 2 = 0.75$ with Carry 1
$0.75 \times 2 = 0.5$ with Carry 1
$0.5 \times 2 = 0$ with Carry 1

$\therefore q_7 = (0.9375)_{10}$
$= (0.1111)_2$ //

Step 5 :-

$q_1 = 0$. and $L_1 = 2$. $\therefore$ Code for AA → 00

$q_2 = (0.010)_2$ and $L_2 = 2$ $\therefore$ Code for BB → 01

$q_3 = (0.1001)_2$ and $L_3 = 4$ $\therefore$ Code for AC → 1001

$q_4 = (0.1010)_2$ and $L_4 = 4$. $\therefore$ Code for CB → 1010

$q_5 = (0.1100)_2$ and $L_5 = 4$. $\therefore$ Code for CA → 1100

$q_6 = (0.1101)_2$ and $L_6 = 4$. $\therefore$ Code for BC → 1101

$q_7 = (0.1111)_2$ and $L_7 = 4$. $\therefore$ Code for CC → 1111

Source Symbol	Probability P_i	Code	Length
AA	9/32	00	2
BB	9/32	01	2
AC	3/32	1001	4
CB	3/32	1010	4
CA	3/32	1100	4
BC	3/32	1101	4
CC	1/16	1111	4

The Average Length "L" is given by

$$L = \sum_{i=1}^{7} P_i l_i = \frac{9}{32} \times 2 + \frac{9}{32} \times 2 + \frac{3}{32} \times 4 + \frac{3}{32} \times 4 + \frac{3}{32} \times 4 + \frac{3}{32} \times 4 + \frac{1}{16} \times 4$$

$$= 2.875 \text{ bits/msg Symbol}$$

The Entropy is given by.

$$H(S) = \sum_{i=1}^{q} p_i \log \frac{1}{p_i} = \sum_{k=1}^{\infty} p_k \log \frac{1}{p_k}$$

$$= \left(\frac{9}{32} \log \frac{32}{9}\right)(2) + \left(\frac{3}{32} \log \frac{32}{3}\right) \times 4 + \frac{1}{16} \log(16)$$

$$= 2.56 \text{ bits/msg symbol} //$$

Encoder efficiency $= \eta_s = \eta_c = \frac{H(S)}{L} \times 100\%$

$$= \left(\frac{2.56}{2.875}\right) \times 100\%$$

$$\eta_c = (0.890) \times 100$$

$$= 89.0\% //$$

* A Source emits an independent Sequence of Symbols from an alphabet Consisting of five Symbols A, B, C, D and E with probabilities of $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{8}$, $\frac{3}{16}$ and $\frac{5}{16}$ Respectively. find the Shannon Code for each Symbol and efficiency of the coding Scheme.

July-2016 [8M], Jan-2014 [6M], June-2010 [6M], Dec-2011 [8M].

2014 * E A D B C.

$\frac{5}{16}$ $\frac{1}{4}$ $\frac{3}{16}$ $\frac{1}{8}$ $\frac{1}{8}$

↓ ↓ ↓ ↓ ↓

P_1 P_2 P_3 P_4 P_5

* $\alpha_1 = Q_1 = 0$

$\alpha_2 = q_2 = \frac{5}{16} = 0.3125$

$\alpha_3 = q_3 = \frac{5}{16} + \frac{1}{4} = \frac{9}{16} = 0.5625$

$\alpha_4 = \frac{5}{16} + \frac{1}{4} + \frac{3}{16} = \frac{12}{16} = 0.75$

$\alpha_5 = \frac{5}{16} + \frac{1}{4} + \frac{3}{16} + \frac{1}{8} = \frac{14}{16} = 0.875$

$\alpha_6 = \frac{5}{16} + \frac{1}{4} + \frac{3}{16} + \frac{1}{8} + \frac{1}{8} = 1$

* Find the minimum value of l_i such that.

$2^{l_i} \geq \frac{1}{P_i}$ for $i = 1, 2, 3, 4, 5$

or

$l_i \geq \log_2 \frac{1}{P_i}$

$l_1 \geq \log_2 \frac{1}{5/16} = 1.6 = 2 \therefore l_1 = 2$

$l_4 \geq \log_2 (\frac{1}{3/8}) = 3$

$l_2 \geq \log_2 (4) = 2$

$l_3 \geq \log_2 [16/3] = 2.415 = 3$

$l_5 \geq \log_2 (8) = 3$

* $\alpha_1 = q_1 = (0.0)_{10} = (0.00)_2$ //

$\alpha_2 = q_2 = (0.3125)_{10}$

$0.3125 \times 2 = 0.625$ with Carry 0.
$0.625 \times 2 = 0.25$ with Carry 1
$0.25 \times 2 = 0.5$ with Carry 0.

$\therefore \alpha_2 = (0.3125)_{10} = (0.010)_2$.

$\alpha_3 = q_3 = (0.5625)_{10}$

$0.5625 \times 2 = 0.125$ with Carry 1
$0.125 \times 2 = 0.25$ with Carry 0.
$0.25 \times 2 = 0.5$ with Carry 0.

$\therefore \alpha_3 = (0.5625)_{10} = (0.100)_2$.

$\alpha_4 = q_4 = (0.75)_{10}$

$0.75 \times 2 = 0.5$ with Carry 1
$0.5 \times 2 = 0$ with Carry 1

$\therefore \alpha_4 = (0.75)_{10} = (0.11)_2$.

$\alpha_5 = (0.875)_{10}$

$0.875 \times 2 = 0.75$ with Carry 1
$0.75 \times 2 = 0.5$ with Carry 1
$0.5 \times 2 = 0$ with Carry 1

$\therefore \alpha_5 = (0.875)_{10} = (0.111)_2$.

* $\alpha_1 = q_1 = 0$ and $l_1 = 2$ $\therefore$ code for E $\rightarrow$ 00
$\alpha_2 = q_2 = (0.10)_2$ and $l_2 = 2$ $\therefore$ code for A $\rightarrow$ 01
$\alpha_3 = q_3 = (0.100)_2$ and $l_3 = 3$ $\therefore$ code for D $\rightarrow$ 100
$\alpha_4 = q_4 = (0.11)_2$ and $l_4 = 3$ $\therefore$ code for B $\rightarrow$ 110
$\alpha_5 = q_5 = (0.111)_2$ and $l_5 = 3$ $\therefore$ code for C $\rightarrow$ 111

SYMBOL	PROBABILITY	CODEWORD	LENGTH
E	5/16	00	2
A	1/4	01	2
D	3/16	100	3
B	1/8	110	3
C	1/8	111	3

* η_c (or) $\eta_s = \frac{H(s)}{L} \times 100$.

$$H(s) = \sum_{k=1}^{5} p_k \log \frac{1}{p_k} = \frac{5}{16}\log\frac{16}{5} + \frac{1}{4}\log(4) + \frac{3}{16}\log\frac{16}{3} + \left(\frac{1}{8}\log(2)\right)(2)$$

$$= 2.2272 \text{ bits/msg symbol}$$

$$L = \sum_{i=1}^{5} p_i l_i = \sum_{i=1}^{5} \frac{5}{16}\times 2 + \frac{1}{4}\times 2 + \frac{3}{16}\times 3 + \frac{1}{8}\times 3 + \frac{1}{8}\times 3$$

$$= 2.4375 \text{ bits/ msg symbol}$$

$$\therefore \eta_c = \frac{H(s)}{L} \times 100\% = \frac{2.2272}{2.4375} \times 100\%$$

$$= 91.37\%$$

* Apply Shannons Encoding (binary) algorithm to the following set of messages and obtain code efficiency and Redundancy.

Dec-2010 [10M], June-July 2011 [10M],

m_1	m_2	m_3	m_4	m_5
1/8	1/16	3/16	1/4	3/8

Sol:-

m_1	m_2	m_3	m_4	m_5
1/8	1/16	3/16	1/4	3/8
(0.125)	(0.0625)	(0.1875)	(0.25)	(0.375)

* Now Increasing probabilities.

m_5	m_4	m_3	m_1	m_2
3/8	1/4	3/16	1/8	1/16
↓	↓	↓	↓	↓
P_1	P_2	P_3	P_4	P_5

(OR)

m_5	m_4	m_3	m_1	m_2
6/16	4/16	3/16	2/16	1/16
↓	↓	↓	↓	↓
P_1	P_2	P_3	P_4	P_5

* The following Sequences of 'α" are Computed.

$q_1 = \alpha_1 = 0.$

$q_2 = \alpha_2 = \frac{3}{8} = 0.375$ //

$q_3 = 3/8 + 1/4 = 0.675$ //

$q_4 = 3/8 + 1/4 + 3/16 = 0.8125$ //

$q_5 = 3/8 + 1/4 + 3/16 + 1/8 = 0.9375$ //

$q_6 = 3/8 + 1/4 + \frac{3}{16} + 1/8 + \frac{1}{16} = 1$ //

(or)

$q_1 = 0$

$q_2 = \frac{6}{16} = 0.375$ //

$q_3 = \frac{6}{16} + \frac{4}{16} = \frac{10}{16} = 0.625$ //

$q_4 = \frac{10}{16} + \frac{3}{16} = \frac{13}{16} = 0.8125$ //

$q_5 = \frac{13}{16} + \frac{2}{16} = \frac{15}{16} = 0.9375$

$q_6 = \frac{15}{16} + \frac{1}{16} = 1$ //

* $l_i \geq \log_2 \frac{1}{P_i}$

$l_1 \geq \log_2 \frac{1}{P_1}$

$\geq \log_2 \frac{8}{3}$

$l_1 = 1.4150$

$l_1 = 2$ //

$l_2 \geq \log_2 \frac{1}{P_2}$

$l_2 \geq \log_2 \frac{1}{1}$

$l_2 = 2$ //

$l_3 \geq \log_2 \frac{1}{P_3}$

$l_3 \geq \log_2 \frac{16}{3}$

$l_3 = 2.4150$

$= 3$ //

$l_4 \geq \log_2 \frac{1}{P_4}$

$\geq \log_2 (8)$

$= 3$ //

$l_5 \geq \log_2 \frac{1}{P_5}$

$\geq \log_2 (16)$

$= 4$ //

* α_1 (or) $q_1 = 0$ //

$\alpha_2 = q_2 = (0.375)_{10}$

$0.375 \times 2 = 0.75$ with Carry 0

$0.75 \times 2 = 0.50$ with Carry 1

$0.50 \times 2 = 0.00$ with Carry 1

$q_2 = (0.375)_{10} = (0.011)_2$ //

$q_3 = \alpha_3 = (0.625)_{10}$

$0.625 \times 2 = 0.25$ with Carry 1
$0.25 \times 2 = 0.50$ with Carry 0
$0.50 \times 2 = 0.00$ with Carry 1

$q_3 = (0.625)_{10} = (0.101)_2$ //

$q_4 = (0.8125)_{10}$

$0.8125 \times 2 = 0.625$ with Carry 1
$0.625 \times 2 = 0.25$ with Carry 1
$0.25 \times 2 = 0.50$ with Carry 0.
$0.50 \times 2 = 0.00$ with Carry 1

$q_4 = (0.8125)_{10} = (0.1101)_2$

$q_5 = (0.9375)_{10}$

$0.9375 \times 2 = 0.875$ with Carry 1
$0.875 \times 2 = 0.75$ with Carry 1.
$0.75 \times 2 = 0.50$ with Carry 1
$0.50 \times 2 = 0.00$ with Carry 1

$q_5 = (0.9375)_{10} = (0.1111)_2$ //

* $q_1 = 0$ and $l_1 = 2$. ∴ code for $m_5 \rightarrow 00$.

$q_2 = (0.011)_2$ and $l_2 = 2$ ∴ code for $m_4 \rightarrow 01$

$q_3 = (0.101)_2$ and $l_3 = 3$ ∴ code for $m_3 \rightarrow 101$

$q_4 = (0.1101)_2$ and $l_4 = 3$ ∴ code for $m_4 \rightarrow 110$

$q_5 = (0.1111)_2$ and $l_5 = 4$ ∴ code for $m_5 \rightarrow 1111$

Source Symbol	p_i	Code.	l_i in bits
m_5	3/8	00	2
m_4	1/4	01	2
m_3	3/16	101	3
m_2	1/8	110	3
m_1	1/16.	1111	4

$$L = \sum_{i=1}^{5} p_i l_i$$

$$= p_1 l_1 + p_2 l_2 + p_3 l_3 + p_4 l_4 + p_5 l_5.$$

$$= (3/8)(2) + (1/4)(2) + (3/16)(3) + (1/8)(3) + (1/16)(4)$$

$$= 2.4375 \text{ bits/message symbol} //$$

$$H(s) = \sum_{k=1}^{5} p_k \log \frac{1}{p_k} \quad \text{(or)} \quad \sum_{i=1}^{5} p_i \log \frac{1}{p_i}$$

$$= \frac{3}{8} \log \frac{8}{3} + \frac{1}{4} \log(4) + \frac{3}{16} \log \frac{16}{3} + \frac{1}{8} \log(8) + \frac{1}{16} \log 16.$$

$$= 2.1085 \text{ bits/msg-symbol} //$$

Code efficiency is given by.

$$\eta_c = \left(\frac{H(s)}{L}\right) \times 100 = \left(\frac{2.1085}{2.4375}\right) 100 = (0.865) \times 100\%$$

$$= 86.5\% //$$

Code Redundancy is given by.

$$R_{\eta_c} = 1 - \eta_c = 1 - 0.865 = 0.135 //$$

$$= 0.135 \times 100\%$$

$$= 13.5\% //$$

* Design an Encoder using Shannon's Encoding algorithm for a Source having Six symbol and probability Statistics $P = \{1/2, 1/4, 1/8, 1/16, 1/32, 1/32\}$. Dec-2012 [10M]

Sol:- Non-Increasing probabilities.

$\frac{1}{2}$ $\frac{1}{4}$ $\frac{1}{8}$ $\frac{1}{16}$ $\frac{1}{32}$ $\frac{1}{32}$.

P_1 P_2 P_3 P_4 P_5 P_6.

$\therefore q_1$ (or) $\alpha_1 = 0$.

$\alpha_2 = p_1 + \alpha_1 = 0.5$

$\alpha_3 = p_2 + \alpha_2 = 0.75$

$\alpha_4 = p_3 + \alpha_3 = 0.875$

$\alpha_5 = p_4 + \alpha_4 = 0.9375$

$\alpha_6 = p_5 + \alpha_5 = 0.96875$

$\alpha_7 = p_6 + \alpha_6 = 1$ //

Codeword length of l_i such that:-

$2^{l_i} \geq \frac{1}{p_i}$ (or) $l_i \geq \log_2 \frac{1}{p_i}$

$l_1 \geq \log_2 \frac{1}{p_1} = \log_2 \frac{1}{1/2} = 1$ //

$l_2 \geq \log_2 \frac{1}{p_2} = \log_2(4) = 2$ //

$l_3 \geq \log_2 \frac{1}{p_3} = \log_2(8) = 3$ //

$l_4 \geq \log_2 \frac{1}{p_4} = \log_2(16) = 4$ //

$l_5 \geq \log_2 \frac{1}{p_5} = \log_2(32) = 5$ Smallest integer $l_5 = 4$

$l_6 \geq \log_2 \frac{1}{p_6} = \log_2(32) = 5$ // Smallest Integer $l_6 = 4$

$\alpha_1 = 0.$

$\alpha_1 = (0.5)_{10}.$

$0.5 \times 2 = 1.00$ with Carry 1.

$(0.5)_{10} = (0.10)_2.$

$\alpha_2 = (0.75)_{10}$

$0.75 \times 2 = 0.5$ with Carry 1.

$0.5 \times 2 = 0.0$ with Carry 1.

$(0.75)_{10} = (0.11)_2$

$\alpha_4 = (0.825)$

$0.825 \times 2 = 0.65$ with Carry 1

$0.65 \times 2 = 0.3$ with Carry 1

$0.3 \times 2 = 0.6$ with Carry 0.

$0.6 \times 2 = 0.2$ with Carry 1.

$(0.825)_{10} = (0.1101)_2$ //

$\alpha_5 = 0.9375$

$0.9375 \times 2 = 0.875$ with Carry 1

$0.875 \times 2 = 0.75$ with Carry 1

$0.75 \times 2 = 0.5$ with Carry 1

$0.5 \times 2 = 0.0$ with Carry 1

$(0.9375)_{10} = (0.1111)_2$

$\alpha_6 = 0.96875$

$0.96875 \times 2 = 0.9375$ with Carry 1

$0.9375 \times 2 = 0.875$ with Carry 1

$0.875 \times 2 = 0.75$ with Carry 1

$0.75 \times 2 = 0.5$ with Carry 1

$(0.96875)_{10} = (0.1111)_2$ //.

Symbol	P	Code	Length
		0	1
P_1	$1/2$	10	2
P_2	$1/4$		
P_3	$1/8$	110	3
P_4	$1/16$	1101	4
P_5	$1/32$	1111	4
P_6	$1/32$	1111	4

//.

SHANNON - FANO ENCODING ALGORITHM:-

This is an Improvement over Shannon's 1st algorithm on Source Coding in the sense that it offers better Coding efficiency Compared to Shannon's algorithm.

The algorithm is as follows:

Step 1: Arrange the probabilities in the non-Increasing order.

Step 2: Group the probabilities into Exactly two sets such that the sum of probabilities in both the groups is almost Equal. Assign bit '0' to all elements of the 1st group and bit '1' to all elements of group 2.

Step 3: Repeat Step 2 by dividing each group into subgroups till no division is possible.

OR

Shannon - Fano Encoding procedure for getting a Compact code with minimum Redundancy is given below.

1. The symbol are arranged according to non-Increasing probabilities.
2. The symbols are divided into two groups so that the sum of probabilities in each group is approximately Equal.
3. All the symbols in the 1st group are designated by a 1st and 2nd group by a "0".
4. The 1st group is again subdivided into two Sub-groups such that each sub-group probabilities are approximately Equal.

5. All the Symbols of the 1st Sub-group are designated by 1 and the 2nd. Sub-group by a "0"

6. The 2nd. group is subdivided into two more Sub-groups and. Step no-5 is Repeated.

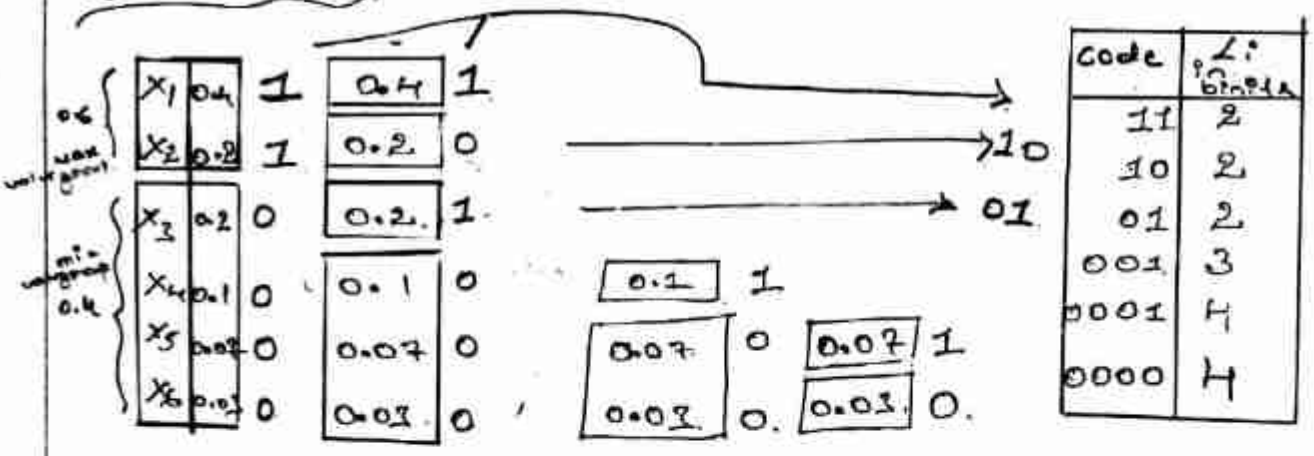
7. This process is Continued till further Sub-division is Impossible.

* Given the messages X_1, X_2, X_3, X_4, X_5 and X_6 with Respective probabilities 0.4, 0.2, 0.2, 0.1, 0.07 and 0.03. Construct a binary code by applying Shannon-Fano Encoding procedure. determine code efficiency and Redundancy of the code.

NOTE:
1st group - 1
2nd group - 0.

Sol :-

1st Way :-



Average length. $L = \sum_{i=1}^{2} P_i l_i = \sum_{i=1}^{6} P_i l_i$

$= (0.4)(2) + (0.2)(2) + (0.1)(3) + (0.07)(4) + (0.03)(4)$

$= 2.3$ bits / message Symbol

$$H(S) = \sum_{i=1}^{7} p_i \log \frac{1}{p_i} = 0.4 \log \frac{1}{0.4} + 2 \times 0.2 \log \frac{1}{0.2} + 0.1 \log \frac{1}{0.1} + 0.07 \log \frac{1}{0.07} + 0.03 \log \frac{1}{0.03}$$

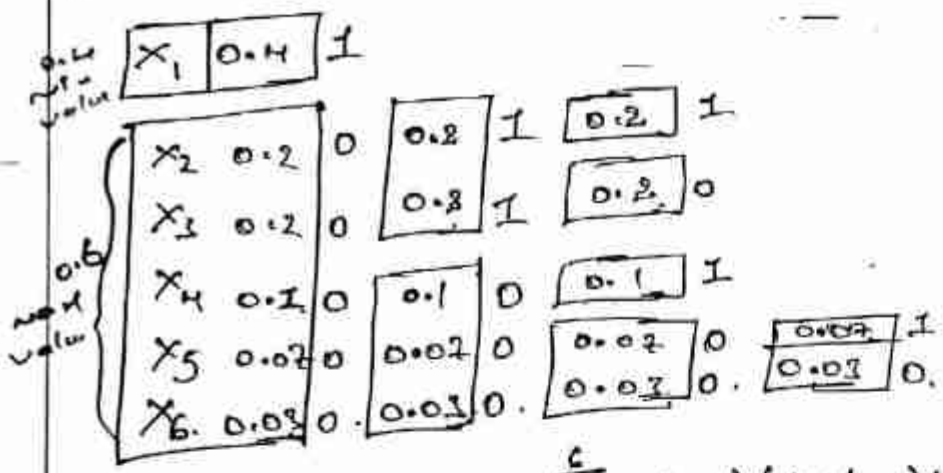
$$= 2.209 \text{ bits/symbol}$$

$\therefore$ Code efficiency $\eta_c = \frac{H(S)}{L} = \frac{2.209 \text{ bits/symbol}}{2.3 \text{ bits/symbol}} = 96.04\%$

$\therefore$ Code Redundancy $R\eta_c = (1 - \eta_c) \times 100\% = 3.96\%$

2nd way :-

Code	Length
1	1
011	3
010	3
001	3
0001	4
0000	4



Average Length $L = \sum_{i=1}^{6} = (0.4)(1) + (0.2)(3) + (0.2)(3) + (0.1)(3) + (0.07)(4) + (0.03)(4) = 2.3$ bits/msg-symbol

Entropy $H(S) = \sum_{i=1}^{6} p_i \log \frac{1}{p_i}$

$= 0.4 \log \frac{1}{0.4} + \left(0.2 \log \frac{1}{0.2}\right) \times 2 + 0.1 \log \frac{1}{0.1} + 0.07 \log \frac{1}{0.07} + 0.03 \log \frac{1}{0.03}$

$= 2.209$ bits/symbol

$\therefore$ Code efficiency $\eta_c = \frac{H(S)}{L} = \frac{2.209}{2.3} = 96.04\%$

$\therefore$ Code Redundancy $R\eta_c = (1 - \eta_c) \times 100 = 3.96\%$

The code-tree corresponding to both the ways can be drawn as shown in fig. (a) & (b)

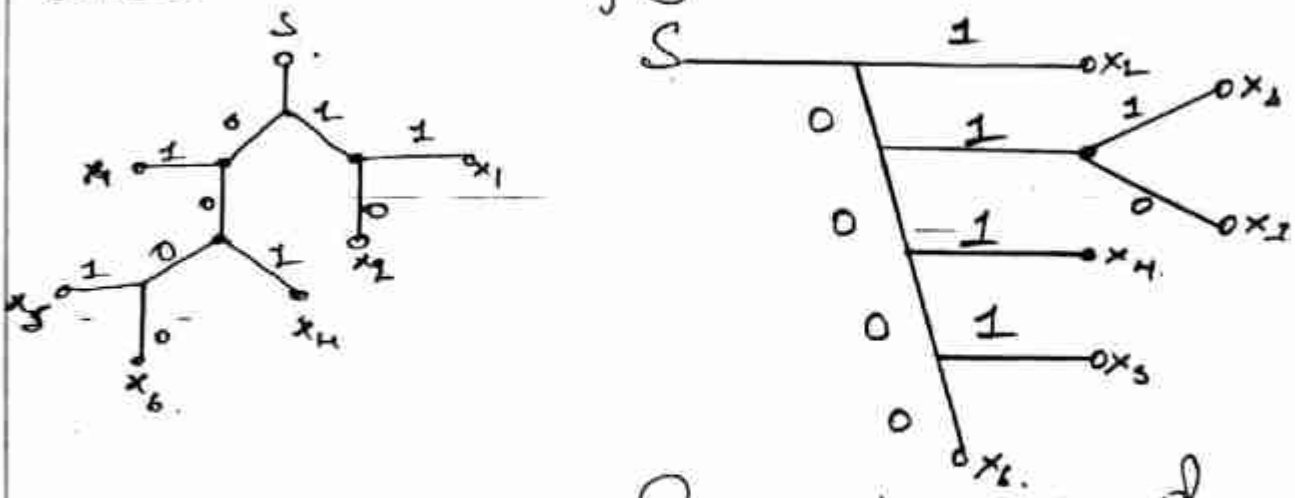


Fig:- Code-Tree for (a) 1st way (b) 2nd way

* Consider the following source:

S = (A, B, C, D, E, F)

P = (0.10, 0.15, 0.25, 0.35, 0.08, 0.07)

Find the code words for the source using Shannon-Fano algorithm. Also find the source efficiency and Redundancy.

Sol:- S = (D, C, B, A, E, F)

P = (0.35, 0.25, 0.15, 0.10, 0.08, 0.07)

D	0.35	0	0.35	0		
C	0.25	0	0.25	1		
B	0.15	1	0.15	0		
A	0.10	1	0.10	1	0.2	0
E	0.08	1	0.08	1	0.08	1
F	0.07	1	0.07	1	0.07	1

Shivaprakasha. NOTE (ITC) book. ← 1st group - 0, 2nd group - 1

(or)

Jayanth Dwijesh (ITC) book. ← 1st group - 1, 2nd group - 0

both method finally we get a same entropy & same length

Answer

Code	Length
00	2
01	2
10	2
110	3
1110	4
1111	4

don't Confuse

use any one method (Left your Choice)

Source efficiency. $\eta_s = \frac{H(S)}{L} \times 100\%$

$$H(S) = \sum_{i=1}^{n} p_i \log \frac{1}{p_i} \quad \text{or} \quad \sum_{k=1}^{n} p_k \log \frac{1}{p_k}$$

$$= 0.15 \log \frac{1}{0.15} + 0.25 \log \frac{1}{0.25} + 0.15 \log \frac{1}{0.15} + 0.10 \log \frac{1}{0.10} + 0.08 \log \frac{1}{0.08} + 0.07 \log \frac{1}{0.07}$$

$$= 2.13 \text{ bits / msg symbol}$$

$$L = \sum_{i=1}^{n} p_i l_i = (0.35 \times 2) + (0.25 \times 2) + (0.15 \times 2) + (0.10 \times 3) + (0.08 \times 4) + (0.07 \times 4)$$

$$= 2.4 \text{ bits/symbol}$$

$$\eta_s = \frac{2.13}{2.4} \times 100\% = 97.08\%$$

$$R_{\eta_s} = 1 - \eta_s = 2.92\%$$

* Consider the following Source.

$S = (A, B, C, D, E, F)$

$P = (0.4, 0.2, 0.2, 0.1, 0.08, 0.02)$

Sol:-

0.40	0	0.40	0				
0.20	0	0.20	1				
0.20	1	0.20	0				
0.10	1	0.10	1	0.10	0		
0.08	1	0.08	1	0.08	1	0.08	
0.02	1	0.02	1	0.02	1	0.02	

NOTE

Shannon-Fano algorithm → 1st group → 0, 2nd group → 1

Huffman code → 1st → 1, 2nd → 0

(Left most choice use any one method.)

finally we get a Source code another answer and same length answer.

Codeword	Length l_i
00	2
01	2
10	2
110	3
1110	4
1111	4

Source efficiency is given by. $\eta_s = \frac{H(s)}{L} \times 100$

$$H(s) = \sum_{k=1}^{N} p_k \log \frac{1}{p_k} \quad \text{(or)} \quad \sum_{q=1}^{q} p_q \log \frac{1}{p_q}$$

$$= 0.40 \log \frac{1}{0.40} + 0.20 \log \frac{1}{0.20} + 0.20 \log \frac{1}{0.10} + 0.10 \log \frac{1}{0.10} + 0.08 \log \frac{1}{0.08} + 0.02 \log \frac{1}{0.02}.$$

$$= 2.124 \text{ bits/symbol}$$

$$L = \sum_{i=1}^{N} p_i l_i = 0.4 \times 2 + 0.2 \times 2 + 0.2 \times 2 + 0.1 \times 3 + 0.08 \times 4 + 0.02 \times 4 = 2.3 \text{ bits/symbol}$$

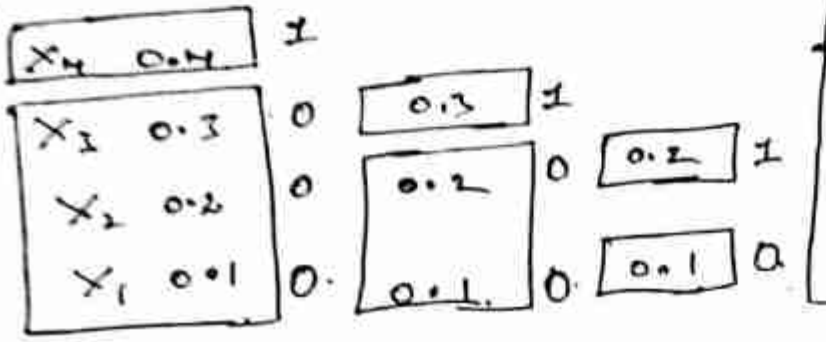
$$\eta_s = \left(\frac{2.124 \text{ bits/symbol}}{2.3 \text{ bits/symbol}} \right) \times 100 = 95.39\%$$

$$R_{\eta_s} = (1 - \eta_s) \times 100 = 4.61\%$$

* You are given 4 messages X_1, X_2, X_3 and X_4 with respective probabilities 0.1, 0.2, 0.3, 0.4.
1) Device a code with prefix property (Shannon's Fano code) for these messages and draw the Code Tree
2) Calculate the efficiency and Redundancy of the code
3) Calculate the probabilities of 0's and 1's in the code

Sol:-

1)



Code	Length
1	1
01	2
001	3
000	3

2) Avg Length $L = \sum_{i=1}^{4} p_i l_i = 0.4 \times 1 + 0.3 \times 2 + 0.2 \times 3 + 0.1 \times 3$

$= 1.9$ bits/msg-symbol //

* $H(S) = \sum_{i=1}^{4} p_i \log \frac{1}{p_i}$

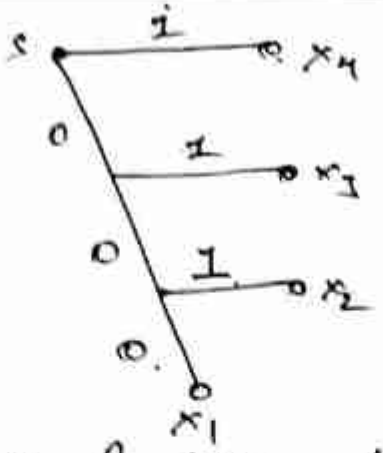
$= 0.4 \log \frac{1}{0.4} + 0.3 \log \frac{1}{0.3} + 0.2 \log \frac{1}{0.2} + 0.1 \log \frac{1}{0.1}$

$= 1.846$ bits/msg-Symbol //

* Code efficiency $\eta_c = \frac{H(S)}{L} \times 100\% = \frac{1.846}{1.9} = 97.15\%$ //
* Code Redundancy $R_{\eta c} = (1 - \eta_c)\ 100\%$

$= 2.85\%$ //

* Code - Tree:



* The probability of '0's and '1's in the code are found using the formulas.

$$p(0) = \frac{1}{L} \sum_{i=1}^{M} (\text{number of '0's in the code for } x_i)[P_i]$$

$$p(0) = \frac{1}{1.9} [(3)(0.1) + (2)(0.2) + (1)(0.3) + (0)(0.4)]$$

$$= 0.5263 //$$

$$p(1) = \frac{1}{L} \sum_{i=1}^{4} (\text{number of '1's in the code for } x_i)[P_i]$$

$$= \frac{1}{1.9} [(0)(0.1) + (1)(0.2) + (1)(0.3) + (1)(0.4)]$$

$$= 0.4737 //$$

* Consider a source $S = \{S_1, S_2\}$ with probabilities $3/4$ and $1/4$ Respectively. obtain Shannon-Fano Code for source S, 4th 2nd and 3rd Extension. Calculate efficiencies for Each Code.

Sol:-

S_1	$3/4$	1
S_2	$1/4$	0

Code	Length l_i
1	1
0	1

NOTE
Warshal Text book method
group - 1
group - 0.

Average Length $L = \sum_{i=1}^{2} p_i l_i = 3/4(1) + 1/4(1)$
$= 1$ binits (or) bits/msg-Symbol

Entropy $H(S) = \sum_{i=1}^{2} p_i \log \frac{1}{p_i} = 3/4 \log 4/3 + 1/4 \log 4$.
$= 0.8113$ bits/msg-Symbol

Code efficiency $\eta_c^{(1)} = \frac{H(S)}{L} = \frac{0.8113}{1} = 81.13\%$

NOTE
$\eta_c^{(1)} \rightarrow$ Indicates 1st order extension

2nd-order extension:

∴ the 2nd extension will have $[S]^2 = [2]^2 = 4$ Symbols given by S_1S_1, S_1S_2, S_2S_1, S_2S_2. with probabilities $S_1S_1 = (3/4)(3/4) = 9/16$, $S_1S_2 = (3/4)(1/4) = 3/16$, $S_2S_1 = (1/4)(3/4) = 3/16$, $S_2S_2 = (1/4)(1/4) = 1/16$ Respectively

S_1S_1	$9/16$	1
S_1S_2	$3/16$	0
S_2S_1	$3/16$	0
S_2S_2	$1/16$	0

$3/16$	1
$3/16$	0
$1/16$	0

$3/16$	1
$1/16$	0

Code	Length
1	1
0 1	2
0 0 1	3
0 0 0	3

The average length L_2 of the 2nd extension is given by

$$L_2 = \sum_{i=1}^{4} p_i l_i = (9/16)(1) + (3/16)(2) + (3/16)(3) + (1/16)(3)$$

$$= 1.6875 \text{ bits / symbol} //$$

Entropy of the 2nd extended source is given by

$$H(S^2) = 2H(S) = 2[0.811] = 1.6226 \text{ bits/msg-symbol}$$

Coding efficiency of the 2nd extended source is given by

$$\eta_c^{(2)} = \frac{H(S^2)}{L_2} = \frac{1.6226}{1.6875} = 96.15\% //$$

3rd Extension :- The 3rd extension will have $(2)^3 = 8$ //

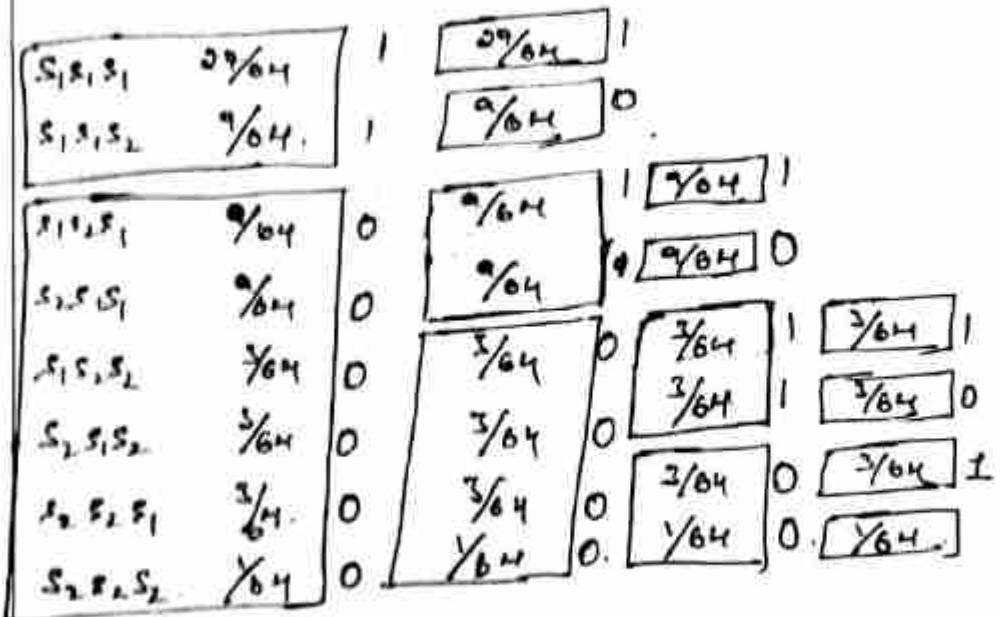
Symbols which are listed below in the non-increasing order.

Symbols :- $S_1S_1S_1$ $S_1S_1S_2$, $S_1S_2S_1$, $S_1S_2S_2$, $S_2S_1S_1$, $S_2S_1S_2$, $S_2S_2S_1$, $S_2S_2S_2$

prob. :- $27/64$, $9/64$, $9/64$, $3/64$, $9/64$, $3/64$, $3/64$, $1/64$.

non-increasing order :- $S_1S_1S_1$, $S_1S_1S_2$, $S_1S_2S_1$, $S_2S_1S_1$, $S_1S_2S_2$, $S_2S_1S_2$, $S_2S_2S_1$, $S_2S_2S_2$

$27/64$ $9/64$ $9/64$ $9/64$ $3/64$ $3/64$ $3/64$ $1/64$.



Code	Length
11	2
10	2
011	3
010	3
0011	4
0010	4
0001	4
0000	4

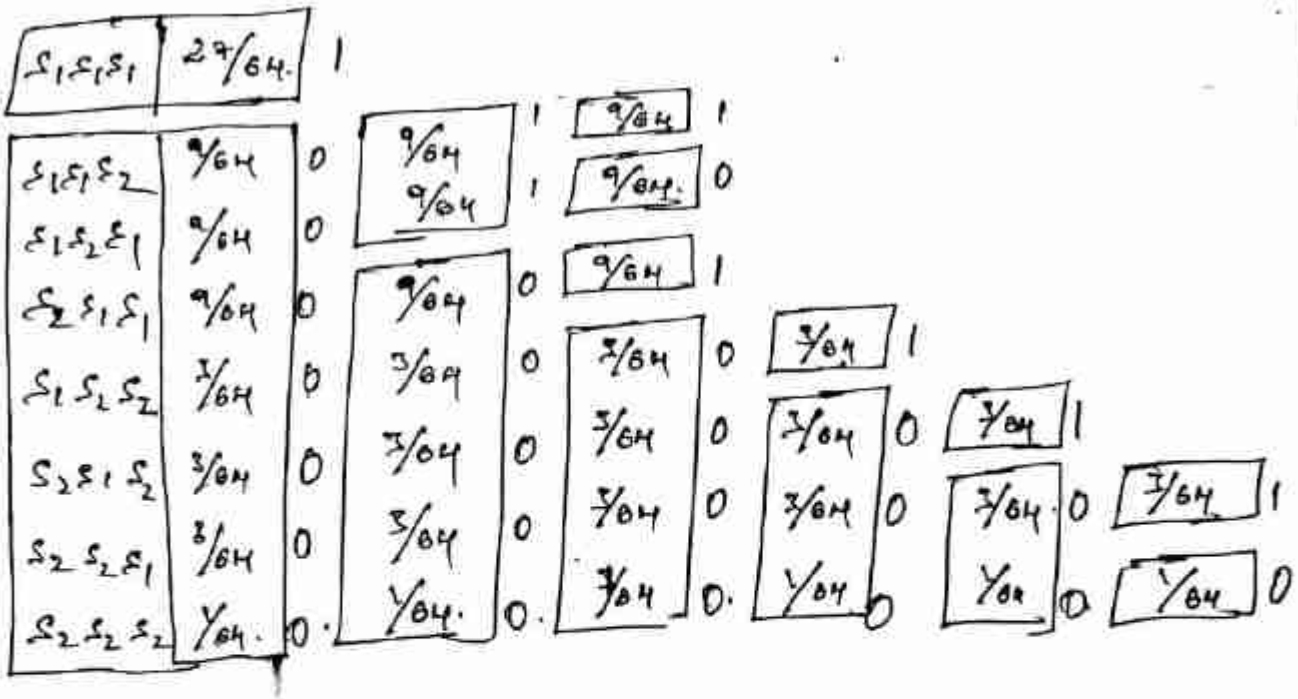
Length $L_3 = \sum_{i=1}^{8} p_i l_i = \left(\frac{27}{64}\right)(2) + \left(\frac{9}{64}\right)(2) + \left(\frac{9}{64}\right)(3) + \left(\frac{9}{64}\right)(3) + \left(\frac{3}{64}\right)(4) + \left(\frac{3}{64}\right)(4) + \left(\frac{3}{64}\right)(4) + \left(\frac{1}{64}\right)(4)$

$= 2.5937$ bits/msg-symbol //

$H(S^3) = 3H(S) = 3[0.8113] = 2.4339$ bits/msg-symbol //

code efficiency $\eta_c^{(3)} = \frac{H(S^3)}{L_3} = \frac{2.4339}{2.59375} = 93.84\%$ //

NOTE:
$\eta_c^{(3)} = 93.84\%$ which is less than efficiency of the 2nd extended Source indeed // and Shannon's 1st theorem of $L = \log_2 \frac{1}{p_i}$ is violated!
// while following step no 2 of the Shannon's Fano-procedure for 3rd extension //



Code.	Length. (l_i)
1	1
01 1	2
0 1 0	3
0 0 1	3
0 0 0 1	4
0 0 0 0 1	5
0 0 0 0 0 1	6
0 0 0 0 0 0	6

$$L_1 = \sum_{i=1}^{8} p_i l_i$$

$$= \left(\frac{27}{64}\right)(1) + \left(\frac{9}{64}\right)(2) + \left(\frac{9}{64}\right)(3) + \left(\frac{3}{64}\right)(4) + \left(\frac{3}{64}\right)(5) + \left(\frac{3}{64}\right)(6) + \left(\frac{1}{64}\right)(6).$$

$L_1 = 2.484375$ bits/msg-symbol //

Entropy $H(S^3) = 3H(S) = 2.4219$ bits/msg-symbol //

∴ Code efficiency of the 3rd Extended Source is

$$\eta_c^{(3)} = \frac{H(S^3)}{L_3} = \frac{2.4219}{2.484375} = 97.97\% //$$

which is greater than the 2nd-extension efficiency of 96.15% //

* Consider a discrete memoryless source whose alphabet consists of K equiprobable symbols.

(a) Explain why the use of a fixed length code for the representation of such a source is about as efficient as any code can be

(b) What conditions have to be satisfied by "K" and the code length for the coding efficiency to be 100%?

Sol:-

(a) Given the source alphabet "S" with "K" equiprobable symbols, The probability of any symbol S_K is given by equation i.e. $P_i = \frac{1}{q}$ for all $i = 1, 2, \ldots q$.

$\therefore P(s_K) = P_K = \frac{1}{K}$ for all $K = 1, 2, 3 \ldots$

The average code-word length is then given by.

$\therefore$ Equation i.e. $L = \sum_{i=1}^{q} P_i l_i$ bits/message symbol.

$$L_{av} = \sum_{i=1}^{K} p_i l_i = \frac{1}{K} \sum_{i=1}^{K} l_i$$

$$= \frac{1}{K} [l_1 + l_2 + \ldots l_K]$$

where the code-word length remains same for all code-words. [fixed-length Code], Then $l_1 = l_2 = \ldots = l_K = I$ (say)

$\therefore L_{av} = L$

Thus we can conclude that with "K" symbols in the code, Any other choice for l_i yields a value for L_{av} no less than "l"

(b). The Entropy of the Source "S" is given by.

$$H(S) = \sum_{i=1}^{K} p_K \log \frac{1}{p_K} = \log_2 K \quad //$$

Since the 'K' number of Symbols are Equiprobable.

$\therefore$ Coding efficiency $= \eta_c = \frac{H(S)}{L_{av}} = \frac{H(S)}{L}$.

Given $\eta_c = 1$.

$\therefore L = H(S) = \log K$

$\therefore$ The Condition to be Satisfied is That.

$K = 2^L$. //

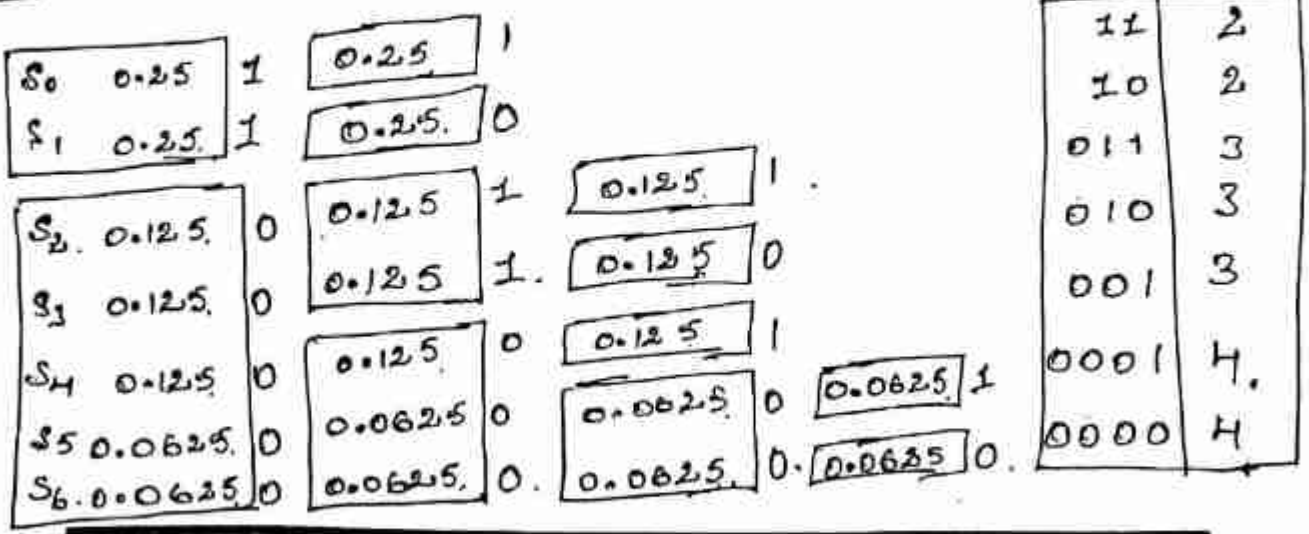
6.7

* A Discrete memoryless Source has an alphabet of Seven Symbols with probabilities for its output as described below:

Symbol	S_0	S_1	S_2	S_3	S_4	S_5	S_6
probability	0.25	0.25	0.125	0.125	0.125	0.0625	0.0625

Compute Shannon–Fano code for this Source. Find Coding efficiency. Explain why the Computed Source code has an efficiency of 100%. //

Sol:-


S_0 0.25
S_1 0.25
S_2 0.125
S_3 0.125
S_4 0.125
S_5 0.0625
S_6 0.0625
1
0
0.25
0.25
0.125
0.125
0.125
0.0625
0.0625
1
0
1
0
1
0
1
0
0
0

The Average Length 'L' is given by.

$$L = \sum_{i=0}^{6} P_i l_i$$

$$= (0.25)(2) + (0.25)(2) + (0.125)(3) + (0.125)(3) + (0.125)(3) + (0.0625)(4) + (0.0625)(4)$$

$$= 2.625 \text{ bits/message-Symbol} //$$

The Entropy H(z) is given by.

$$H(z) = \sum_{i=0}^{6} P_i \log \frac{1}{P_i}$$

$$= \left[(0.25)\log\frac{1}{(0.25)}\right] \times 2 + \left[(0.125)\log\frac{1}{(0.125)}\right] \times 3 +$$

$$\left[(0.0625)\log\frac{1}{(0.0625)}\right] \times 2. = 2.625 \text{ bits/message-Symbol}$$

$$\therefore \text{Coding efficiency} = \eta_c = \frac{H(z)}{L} = \frac{2.625}{2.625} = 100\%$$

Thus, we observe that the Coding efficiency is 100%

The Reason for the coding efficiency to be 100% is as given below:

The Coding efficiency is given by.

$$\eta_c = \frac{H(z)}{L} = \frac{\sum_{i=0}^{6} P_i \log \frac{1}{P_i}}{\sum_{i=0}^{6} P_i l_i}$$

If $l_i = \log \frac{1}{P_i}$ for $i = 0, 1, \ldots 6$. then η_c as seen from the above expression becomes 100% from the Code table, we observe that

When $i=0$. $p_0=0.25$. $\log\frac{1}{p_0}=\log\frac{1}{0.25}=2=l_0$.

$i=1$ $p_1=0.25$. $\log\frac{1}{p_1}=\log\frac{1}{0.25}=2=l_1$

$i=2$. $p_2=0.125$. $\log\frac{1}{p_2}=\log\frac{1}{0.125}=3=l_2$

$i=3$ $p_3=0.125$. $\log\frac{1}{p_3}=\log\frac{1}{0.125}=3=l_3$.

$i=4$ $p_4=0.125$. $\log\frac{1}{p_4}=\log\frac{1}{0.125}=3=l_4$.

$i=5$ $p_5=0.0625$. $\log\frac{1}{p_5}=\log\frac{1}{0.0625}=4=l_5$

$i=6$. $p_6=0.0625$ $\log\frac{1}{p_6}=\log\frac{1}{0.0625}=4=l_6$.

Thus we observe that $p_i=\log\frac{1}{p_i}$ for all $i=0,2,\ldots 6$, and hence the coding efficiency is 100%

* A source produces 100 symbols S_1 and S_2 with probabilities $7/8$ and $1/8$ respectively. Devise a coding scheme using Shannon's - fano encoding procedure to get a coding efficiency of at least 75%

Sol:

S_1 $7/8$	
S_2 $1/8$	

Code	length of l_i in bits
1	1
0	1

Average Length $L=\sum_{i=1}^{2} p_i l_i=\left(\frac{7}{8}\right)(1)+\left(\frac{1}{8}\right)(1)$

$=1$ bit/msg-symbol //

Entropy $H(s)=\sum_{i=1}^{2} p_i \log\frac{1}{p_i}=\frac{7}{8}\log\frac{8}{7}+\frac{1}{8}\log(8)$

$=0.54356$ //

∴ Coding efficiency $\eta_c = \frac{H(S)}{L} = \frac{0.54356}{1} = 54.356\%$

∴ This efficiency is less than the given efficiency of 45%

Now let us consider the 2nd extension of the given code.

2nd extension,

Symbol	probability
S_1S_1	49/64
S_1S_2	7/64
S_2S_1	7/64
S_2S_2	1/64

Code	Length in bits
1	1
01	2
001	3
000	3

Average Length $L_2 = \sum_{i=1}^{4} P_i l_i$

$= \left(\frac{49}{64}\right)(1) + \left(\frac{7}{64}\right)(2) + \left(\frac{7}{64}\right)(3) + \left(\frac{1}{64}\right)(3)$

$= 1.359375$ bits/msg-symbol

Entropy $H(S^2) = 2H(S) = 2 \times 0.54356$

$= 1.08712$ bits/msg-symbol

$\eta_c^{(2)} = \left(\frac{H(S^2)}{L_2}\right)_{max} = \frac{1.08712}{1.359375} \times 100\% = 80\%$

which is greater than the Required value of 75%.

Thus the 2nd Extension Code is the Required Coding Scheme.

A Source produces two Symbols "A" and "B" with probabilities 0.05 and 0.95 Respectively. Construct a suitable binary Code such that the efficiency of Coding is at least 65%

ECTE – July/Aug – 2003

Sol :-

B 0.95	1
A 0.05	0

Code	Length of l_i in bits
1	1
0	1

Average Length $L = \sum_{i=1}^{2} p_i l_i$

$= (0.95)(1) + (0.05)(1)$

$= 1$ bits/msg-Symbol //

Entropy $H(x) = \sum_{i=1}^{2} p_i \log \frac{1}{p_i}$

$= 0.95 \log \frac{1}{0.95} + 0.05 \log \frac{1}{0.05}$

$= 0.286$ bits/msg symbol //

$\therefore$ Coding efficiency $\eta_c = \frac{H(x)}{L} = \frac{(0.286)}{1} = 28.6\%$ //

This efficiency is less than the given efficiency of 65% NOW let us Consider the 2nd extension of the given Source

Symbol probability

BB	0.9025	1				
BA	0.0475	0	0.0475	1		
AB	0.0475	0	0.0475	0	0.0475	1
AA	0.0025	0	0.0025	0	0.0025	0

Code	Length l_i in bits
1	1
01	2
001	3
000	3

Average length $L_2 = \sum_{i=1}^{n} p_i l_i$

$= (0.9025)(1) + (0.04875)(2) + (0.04875)(3) + (0.0025)(3)$

$= 1.1475$ bits/msg-symbol //

$H(S^2) = 2H(S) = (2)(0.2864)$

$= 0.5728$ bits/msg-symbol //

$\eta_c^{(2)} = \frac{H(S^2)}{L_2} = \left(\frac{0.5728}{1.1475}\right) \times 100 = 49.92\%$ //

This efficiency is less than the given efficiency of 65%. Now let us consider the 3rd extension of the given code.

Symbol	probability								
BBB	0.857375	1							
BBA	0.045125	0	0.045125	1	0.045125	1			
BAB	0.045125	0	0.045125	1	0.045125	0			
ABB	0.045125	0	0.045125	0	0.045125	1			
BAA	0.002375	0	0.002375	0	0.002375	0	0.002375	1	0.002375 1
ABA	0.002375	0	0.002375	0	0.002375	0	0.002375	1	0.002375 0
AAB	0.002375	0	0.002375	0	0.002375	0	0.002375	0	0.002375 1
AAA	0.000125	0	0.000125	0	0.000125	0	0.000125	0	0.000125 0

Code.	Length i.e. no. bits.
1	1
011	3
010	3
001	3
00011	5
00010	5
00001	5
00000.	5

Average Length $L_1 = \sum_{i=1}^{8} P_i l_i$

$= (0.857375)(1) + (0.044125)(3) + (0.044125)(3) + (0.002175)(3) + (0.000125)(5) + (0.000125)(5) + (0.000125)(5) + (0.000125)(5)$

$= 1.29975$ bits/msg-symbol //

$H(S^3) = 3H(S)$

$= 3(0.2864)$

$= 0.8592$ //

$\eta_c^{(1)} = \frac{H(S^1)}{L_1} = \frac{0.8592}{1.29975} = 66.11\%$ //

Thus efficiency is greater than the Required value of 65% Thus the 3rd Extension Code of the Required Coding Scheme.

SHANNON - FANO TERNARY CODE:-

The Seven steps observed for getting a binary code are slightly modified for ternary code as given below:

1. The symbols are arranged according to non-Increasing probabilities.
2. The symbols are divided into three groups so that the sum of probabilities in each group is approximately Equal.
3. All the symbols in the 1st group are designated by a "2", the 2nd group by a "1" and the 3rd group by a "0".
4. The 1st group is again subdivided into three more sub-groups such that each sub-group probabilities are approximately Equal.
5. All the symbols of the 1st sub-group are designated by a "2" the 2nd sub-group by a "1" and the 3rd sub-group by a "0".
6. The 2nd and 3rd group are subdivided into three more sub-groups each and step NO 5 is Repeated.
7. This process is Continued till further sub-division is Impossible.

* June-July 2016 [8M]

* Construct a Shannon-fano Ternary code for the following Ensemble and find Code efficiency and Redundancy. Also draw the corresponding Code-Tree.

$S = \{S_1, S_2, S_3, S_4, S_5, S_6, S_7\}$.

$P = \{0.3, 0.2, 0.12, 0.12, 0.06, 0.06, 0.04\}$ with $X\{0,1,2\}$

Sol:

S_1	0.3	2				
S_2	0.2	1				
S_3	0.12	0	0.12	2		
S_4	0.12	0	0.12	1		
S_5	0.06	0	0.06	0	0.06	2
S_6	0.06	0	0.06	0	0.06	1
S_7	0.04	0	0.04	0	0.04	0

Code.	L.P.O. bPath.
2	1
1	1
02	2
01	2
002	3
001	3
000	3

$$L = \sum_{i=1}^{7} P_i l_i$$

$= (0.3)(1) + (0.2)(1) + (0.12)(2) + (0.12)(2) + (0.06)(2) +$
$(0.06)(2) + (0.04)(1)$.

$= 1.56$ tnPath/Msg-Symbol //.

$$H(s) = \sum_{i=1}^{7} P_i \log \frac{1}{P_i}$$

$= 0.3 \log \frac{1}{0.3} + 0.2 \log \frac{1}{0.2} + 0.12 \log \frac{1}{0.12} + 0.12 \log \frac{1}{0.12}$
$+ 0.06 \log \frac{1}{0.06} + 0.06 \log \frac{1}{0.06} + 0.04 \log \frac{1}{0.04}$

$= 2.4491$ BPth/Msg Symbol //.

The Entropy of r-ary units / message symbol is given by

$$H_r(s) = \frac{H(s)}{\log_2 r}$$

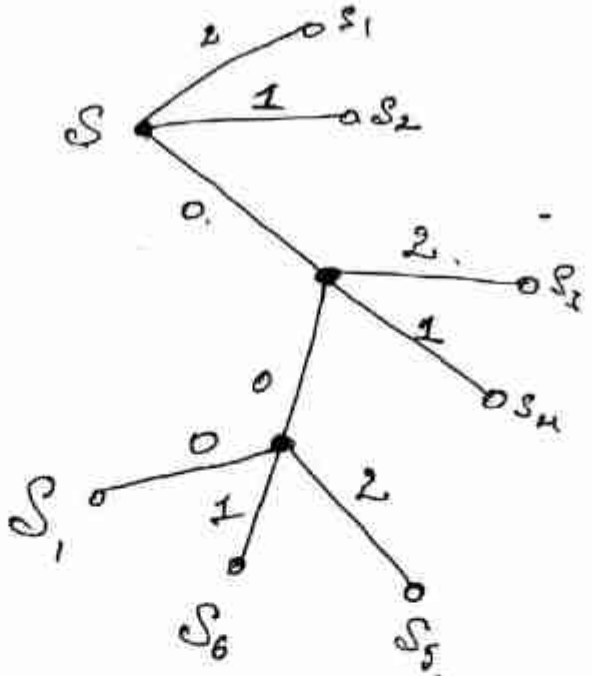
$r=3 \quad H_3(s) = \frac{H(s)}{\log_2 3} = \frac{2.4M19}{\log_2 3} = 1.5452$ Ternary units / message-symbol

∴ the Ternary Coding efficiency is given by

$$\eta_c = \frac{H_3(s)}{L} = \frac{1.5452}{1.56} = 0.9905$$

$$\eta_c = (0.9905) \times 100\% = 99.05\%$$

Code - Tree :



HUFFMAN CODE (COMPACT CODES) OR COMPACT CODE: HUFFMAN MINIMUM - REDUNDANCY CODE

Huffman Codes achieve the minimum Codeword length amongst all other Coding algorithm. Hence they are also called Compact Code. Huffman Codes are Computationally Simpler and are widely used in the modern communication System.

The algorithm for Encoding the Information into an r-ary Codeword is detailed as follows.

Step 1:- Compute the number of Stages required for the Encoding operation which is given by.

$$n = \frac{N-r}{r-1}$$

where 'N' is the Total number of Symbols in the Source alphabet.

Step 2:- The value of 'n' has to be an Integer. If not append minimum number of dummy Symbols with zero probability So as to get 'n' as an Integer value. However, 'n' will always take an Integer value for binary Encoding as the denominator in the Expression of 'n' is always. ($r-1 = 2-1 = 1$ for binary).

Step 3:- Arrange the probabilities in the descending order.

Step 4:- Combine the Last "r" probabilities in the set by Summing up as a Single probability and place the Sum at the appropriate position in the set by Reordering it. Now the set contains "r-1" elements less than the previous Stage

Step-5:- Continue Step "4" till we Reach the position where we have only "r" elements, Assign the symbols "0", "1", ---, "r-1" to those elements.

Step-6:- Traverse back by one stage. find the "r" probabilities in the stage above. Sum is placed in the Succeeding Stage. Assign the codeword assigned to the sum the prefix for all probabilities. To differentiate, again assign "0", "1", ----, "r-1" to those probabilities in front of their prefix.

Step-7:- Repeat Step-6 till first Stage is reached.

OR

Huffman Coding procedure for obtaining a compact code with least Redundancy is given below.

1. The Source Symbols are listed in the non-increasing order of probabilities.
2. Consider the Equation.

$$q = r + (r-1)\alpha$$

Where q = number of Source Symbols.

r = number of different Symbols used in the Code alphabet.

The quantity "α" is Calculated and it should be an integer.

If it is not, then "dummy Symbol" with "zero" probabilities are added to "q" to make the quantity "α" an integer.

for binary codes, 'r' will always be an integer and hence this step is not needed. while Constructing Huffman binary codes.

3. The last "r" symbols of Source 'S' are now Combined into a "Single Composite Symbol" by adding their probabilities to get a Reduced Source 'S$_a$'. The symbols of this Reduced Source 'S$_a$' are now arranged in the order of non-increasing probabilities.

4. The last "r" symbol of Source 'S$_a$' are now Combined to form another Composite Symbol by adding their probabilities to get a further Reduced Source 'S$_b$'. The symbols of 'S$_b$' are now arranged in the order of non-increasing probabilities.

5. This process of Combining last "r" symbols every time for a new Reduced Source, is Continued till we arrive at a last Source having "r" symbols.

6. The last Source with "r" message symbols are now Encoded with 'r' different code symbols (say) 0, 1, 2, ----- (r-1).

7. In binary Coding, the last Source with 2 symbols are Encoded with two words '0' and '1'. As we pass on to the Source "Going backwards" with "3" symbols, either "0" may be recomposed as "00" and '01' or '1' may be Recomposed as "10" and "11" depending on which two-out-of-three have been Combined to get the last Reduced Source symbols.

In ternary coding '0' is Recomposed as '00', '01' and '02', '1' as '10', '11' and '12' and finally '2' as '20', '21' and '22'..... etc.//

8. As we part on from Source-to-Source Coarking. backwards, The Recomposition of one Code-word (each time is done in order to form new code-words. 9. This procedure is Continued till we assign Code-words to all the Source Symbols of alphabets of Source S. If any "dummy Symbols" are used, They are discarded.

Ex* Given. The messages S_1, S_2, S_3 and S_4 with Respective probabilities of 0.4, 0.3, 0.2 and 0.1 Construct a binary code by applying Huffman Encoding procedures. Determine the efficiency and Redundancy of the code so formed.

Soln

Source Symbols	P_i	Code	Source S_a		Source S_b	
			P_i	Code	P_i	Code
S_1	0.4	1	0.4	1	→ 0.6	0
S_2	0.3	00	0.3	00	→ 0.4	1
S_3	0.2	010	→ 0.3	01		
S_4	0.1	011				

Average Length. 'L' is Calculated using equation as.

$$L = \sum_{i=1}^{4} P_i l_i = (0.4)(1) + (0.3)(2) + (0.2)(3) + (0.1)(3).$$

$$= 1.9 \text{ bits/msg-Symbol}$$

Entropy, H

$$H(s) = \sum_{i=1}^{4} P_i \log \frac{1}{P_i} = 0.4 \log \frac{1}{0.4} + 0.3 \log \frac{1}{0.3} + 0.2 \log \frac{1}{0.2} + 0.1 \log \frac{1}{0.1} = 1.846 \text{ bits/msg Sy}$$

$\therefore$ Code efficiency $\eta_c = \left(\frac{H(s)}{L}\right) \times 100\% = \frac{1.846}{1.9} = 97.15\%$

$\therefore$ code Redundancy $R_{nc} = (1 - \eta_c) \times 100\%$

$= 2.85\%$

* Given the messages X_1, X_2, X_3, X_4, X_5 and X_6 with respective of probabilities of 0.4, 0.2, 0.2, 0.1, 0.09 and 0.01 Construct a binary code by applying Huffman Encoding procedures. Determine the efficiency and Redundancy of the Code so formed.

Sol :-

Source Symbol	P_i	Code	Source S_a		Source S_b		Source S_c		Source S_d	
			P_i	code	P_i	Code	P_i	Code	P_i	Code
X_1	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
X_2	0.2	01	0.2	01	0.2	01	0.4	00	0.4	1
X_3	0.2	000	0.2	000	0.2	000	0.2	01		
X_4	0.1	0010	0.1	0010	0.2	001				
X_5	0.09	00110	0.1	0011						
X_6	0.01	00111								

$$L = \sum_{i=1}^{6} P_i l_i = (0.4)(1) + (0.2)(2) + (0.2)(3) + (0.1)(4) + (0.09)(5) + (0.01)(5)$$

$= 2.3$ bits/msg Symbol

$$H(s) = \sum_{i=1}^{6} P_i \log \frac{1}{P_i}$$

$$= 0.4 \log \frac{1}{0.4} + 2 \times 0.2 \log \frac{1}{0.2} + 0.1 \log \frac{1}{0.1} + 0.07 \log \frac{1}{0.07} + 0.01 \log \frac{1}{0.01} = 2.209 \text{ bits/msg Symbol}$$

$\therefore$ Code efficiency $\eta_c = \frac{H(s)}{L} = \frac{2.209}{2.3} = 96.04\%$ //

Code Redundancy $R_{\eta_c} = (1 - \eta_c) \times 100\%$

$= 3.96\%$ //

* Consider a zero-memory source with.

$S = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$

$P = \{0.4, 0.2, 01, 0.1, 01, 0.05, 0.05\}$

1) Construct a binary Huffman Code by placing the Composite Symbol as low as you can.

2) Repeat (1) by moving the Composite Symbol "as high as possible"

In each of the cases 1) and 2) above, Compute the variances of the Codeword Length and Comment on the Result. DEC-2010 [8M].

1)

Source Symbol	P_i	Code.	Source S_a		Source S_b		Source S_c		Source S_d		Source S_e	
			P_i	code	P_i	code	P_i	code	P_i	code	P_i	code
S_1	0.4	1	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
S_2	0.2	01	0.2	01	0.2	01	0.2	01	0.4	00	0.4	1
S_3	0.1	0010	0.1	0010	0.2	000	0.2	000	0.2	01		
S_4	0.1	0011	0.1	0011	0.1	0010	0.2	001				
S_5	0.1	0000	0.1	0000	0.1	0011						
S_6	0.05	00010	0.1	0001								
S_7	0.05	00011										

Average length $L = \sum_{i=1}^{7} P_i l_i$

$= (0.4)(1) + (0.2)(2) + (0.1)(4) + (0.1)(4) + (0.1)(4) + (0.05)(5) + (0.05)(5)$

$= 2.5$ binits/msg-symbol //.

The Variance is defined as

$$Var(x) = E[(x-\mu)^2]$$

where μ = average value.

The variance of word-length is calculated from.

$$Var(l_i) = E[(l_i - L)^2]$$

$$= \sum_{i=1}^{7} P_i (l_i - L)^2$$

$= (0.4)(1-2.5)^2 + (0.2)(2-2.5)^2 + (0.1)(4-2.5)^2 + (0.1)(4-2.5)^2 + (0.1)(4-2.5)^2 + (0.05)(5-2.5)^2 + (0.05)(5-2.5)^2$

$= 2.25$ //

2) Below Table gives the Complete Code-Table with the Composite Symbol placed. "as high as possible".

Source Symbol	P_i	Code	Source S_a		Source S_b		Source S_c		Source S_d		Source S_e	
			P_i	code	P_i	code	P_i	code	P_i	code	P_i	code
S_1	0.4	00	0.4	00	0.4	00	0.4	00	0.4	1	0.6	0
S_2	0.2	11	0.2	01	0.2	10	0.2	01	0.4	00	0.4	1
S_3	0.1	011	0.1	010	0.2	11	0.2	10	0.2	01		
S_4	0.1	100	0.1	011	0.1	010	0.2	11				
S_5	0.1	101	0.1	100	0.1	011						
S_6	0.05	0100	0.1	101								
S_7	0.05	0101										

$$L = \sum_{i=1}^{7} P_i l_i = (0.4)(2) + (0.2)(2) + (0.1)(3) + (0.1)(3) + (0.1)(3) + (0.05)(4) + (0.05)(4)$$

$= 2.5$ bits/msg - Symbol //

$$\text{Var}(l_i) = E[(l_i - L)^2] = \sum_{i=1}^{7} P_i (l_i - L)^2$$

$$= (0.4)(2-2.5)^2 + (0.2)(2-2.5)^2 + (0.1)(3-2.5)^2 + (0.1)(3-2.5)^2 + (0.1)(3-2.5)^2 + (0.05)(4-2.5)^2 + (0.05)(4-2.5)^2$$

$= 0.45$ //

* Consider a source with 8 alphabets A to H with respective probabilities of 0.22, 0.20, 0.18, 0.15, 0.10, 0.08, 0.05, 0.02.

a) Construct a binary Compact (Huffman) code and determine the code efficiency

b) Construct a ternary Compact Code and determine the efficiency of the code.

c) Construct a quaternary Compact Code & determine the code efficiency. Compare & comment on the result. Draw code-trees for all three cases.

June-2012 [12M], Dec-2012 [10M]

a) binary Code:

Source Symbol	P_i	code	Source S_a	Source S_b	Source S_c	Source S_d	Source S_e	Source S_f
A	0.22	10	0.22	0.22	0.25	0.33	0.42	0.58
B	0.20	11	0.20	0.22	0.22	0.25	0.33	0.42
C	0.18	000	0.18	0.20	0.22	0.25	0.25	
D	0.15	001	0.15	0.18	0.20	0.22		
E	0.10	011	0.10	0.15	0.18			
F	0.08	0100	0.08	0.10				
G	0.05	01010	0.05					
H	0.02	01011						

(Sort in descending order of probability)

$L(x) = \sum_{i=1}^{8} P_i l_i = (0.22)(2) + (0.20)(2) + (0.18)(3) + (0.15)(3) + (0.10)(3) + (0.08)(4) + (0.05)(5) + (0.02)(5)$

$= 2.8$ bits/msg-symbol //.

(a) $H_r(x) = \sum_{i=1}^{8} p_i \log \frac{1}{p_i}$

$= 0.12 \log \frac{1}{0.12} + 0.20 \log \frac{1}{0.20} + 0.18 \log \frac{1}{0.18} + 0.15 \log \frac{1}{0.15}$
$+ 0.10 \log \frac{1}{0.10} + 0.08 \log \frac{1}{0.08} + 0.05 \log \frac{1}{0.05} + 0.02 \log \frac{1}{0.02}$

$= 2.7535$ bits/msg symbol //

$\eta_{r(x)} = \frac{H_r(x)}{L} \times 100 = \frac{2.7535}{2.8} \times 100 = 98.34\%$ //

$R_{\eta_{r(x)}} = (1 - \eta_{r(x)}) \times 100 = 1.66\%$ //

b) Ternary Codes :-

Let us consider equation is given by
$q = n + (r-1)\alpha$.
where $r = 3$ (Ternary)
$q = 3 + (3-1)\alpha = 3 + (2)(\alpha)$
$q - 3 = 2\alpha$.
$\therefore \alpha = \frac{q-3}{2}$ //

NOTE
q = total number of symbols (or) probs. = 8

$\alpha = \frac{8-3}{2} = \frac{5}{2}$ //

Let us Consider Equation is given by
$n = \frac{N-r}{r-1}$

(or) $n = \frac{N-3}{3-1}$

$n = \frac{N-3}{2}$ //

$n = \frac{8-3}{2} = \frac{5}{2}$ //

which is not an integer, The next minimum value for 'N' (or) N to have an integer value for.
'n' is $N = 9$ (or) α is $q = 9$
$\therefore$ add one dummy symbol with probabilities for which 'N' (or) 'q' become.

$\alpha = \frac{9-3}{2} = \frac{6}{2} = 3$ //

(or) $n = \frac{9-3}{2} = \frac{6}{2} = 3$ //

Source Symbols	P_i	Code	Source S_a		Source S_b		Source S_c	
			P_i	Code	P_i	Code	P_i	Code
A	0.22	2	0.22	2	0.25	1	0.53	0
B	0.20	00	0.20	00	0.20	2	0.25	1
C	0.18	01	0.18	01	0.20	00	0.22	2
D	0.15	02	0.15	02	0.18	01		
E	0.10	10	0.10	10	0.15	02		
F	0.08	11	0.08	11				
G	0.05	120	0.07	12				
H	0.02	121						
I	0	122						

Dummy Symbol

Discard

Length. $L^{(3)} = \sum_{i=1}^{8} P_i l_i$ (note: ternary)

$= (0.22)(1) + (0.20)(2) + (0.18)(2) + (0.15)(2) + (0.10)(2) + (0.08)(2) + (0.05)(3) + (0.02)(3)$

$= 1.85$ ternaits / msg - Symbol //

$H_3(S) = \frac{H_2(S)}{\log_2 3} = \frac{H(S)}{\log_3(3)} = \frac{2.7585}{\log_3(3)} = 1.7393$ ternary / msg-Symbol //

Code efficiency $\eta_{c(3)} = \left(\frac{H_3(S)}{L^{(3)}}\right)_{max} = \left(\frac{1.7373 \text{ ternary/msg symbol}}{1.85 \text{ ternary/msg symbol}}\right)$

$= 93.91\%$

$R_{\eta_{c(3)}} = (1 - \eta_{c(3)}) \times 100 = 6.09\%$ //

c) Hoffman Quarternary Code.:

$q = r + (r-1)\alpha$.

$r = 4$.

$q = 4 + 3\alpha$

$\alpha = \frac{q-4}{3}$

$= \frac{8-4}{3} = 4/3$.

(or)

$n = \frac{N-r}{r-1}$

$= \frac{N-4}{4-1} = \frac{N-4}{3}$

$= \frac{8-4}{3} = 4/3$

Which is not an Integer, The next consecutive value for 'N' to have an integer value for 'q' is 10.

For $N = 10$, $q = 10$.

$\therefore$ add two dummy Symbols with '0' probabilities for which 'n' becomes 2.

$\alpha = \frac{10-4}{3} = 6/3 = 2$

$n = \frac{N-r}{r-1} = \frac{10-4}{4-1} = 2$

Hence,

$n = 2$ (integer)

Source Symbol	P_i	Code.	Source S_a.		Source S_b.	
			P_i	Code	P_i	Code
A	0.22	1	0.22	1	0.40	0
B	0.20	2.	0.20	2	0.22	1
C	0.18	3	0.18	3	0.20	2
D	0.15	00	0.15	00	0.18	3.
E	0.10	01	0.10	01		
F	0.08	02.	0.08	02		
G	0.05	030	0.07.	03		
H	0.02	031				
I	0	032				
J	0	033	Discard.			

Dummy Symbols.

The average length $L^{(4)}$ is given by

$$L^{(4)} = \sum_{i=1}^{8} P_i l_i$$

$= (0.22)(1) + (0.20)(1) + (0.18)(1) + (0.15)(2) + (0.10)(2) + (0.08)(2) + (0.05)(3) + (0.02)(3)$

$L^{(4)} = 1.47$ quaternary digits / msg - symbol

$$H_4(s) = \frac{H(s)}{\log_2(4)} = \frac{2.7535}{2} = 1.37675 \text{ quaternary / msg symbol}$$

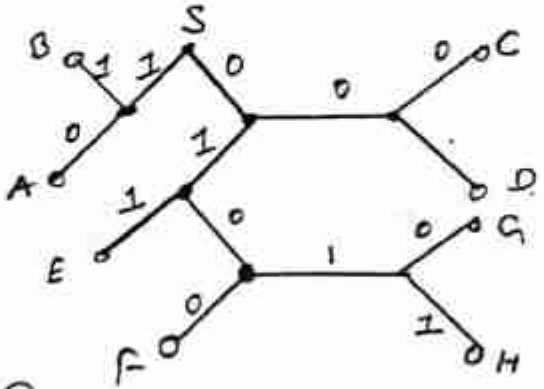
$\therefore$ Code efficiency $\eta_{C(4)} = \frac{H_4(s)}{L^{(4)}} \times 100 = \left(\frac{1.37675}{1.47}\right) \times 100$

$\eta_{C(4)} = 93.66\%$

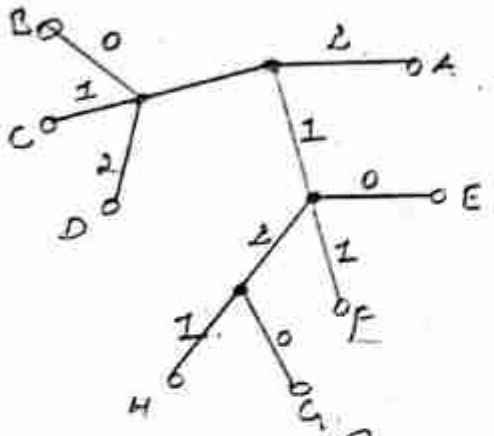
$R_{n_{C(4)}} = (1 - \eta_{C(4)}) \times 100 = 6.34\%$

Sl. No.	Coding Type	r	Coding efficiency	Coding Redundancy
1	Binary coding	2	98.14%	1.66%
2	Ternary Coding	3	93.91%	6.09%
3.	Quaternary Coding	4	93.66%	6.34%

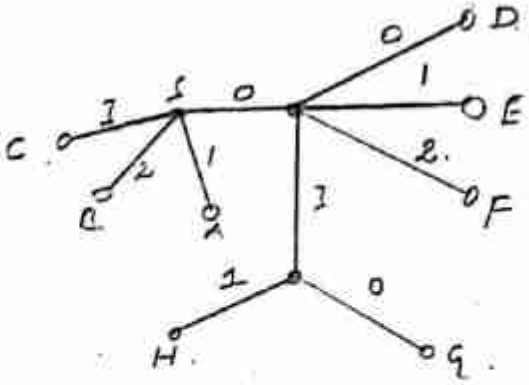
Tree diagram :-



a) Code - Tree for binary Coding.



b) Code-Tree for ternary Coding.



c) code-Tree for Quaternary Coding.

*a) Apply Huffman Encoding procedure for the following set of messages and determine the efficiency of the binary code so formed.

X_1	X_2	X_3
0.7	0.15	0.15

b) If the same Technique is applied to the 2nd order extension for the above messages, how much will the efficiency be Improved?

July/Aug-2008. [10M], July/Aug-2005 [12M].

Sol:-

Source Symbols.	P_i	Code.	Source S_0	
			P_i	Code.
X_1	0.7	0	0.7	0
X_2	0.15	10	0.3	1
X_3	0.15	11		

The average length "L" is given by

$$L = \sum_{i=1}^{3} P_i l_i = (0.7)(1) + (0.15)(2) + (0.15)(2)$$

$$= 1.3 \text{ bits/msg-Symbol} //$$

$$H(S) = \sum_{i=1}^{3} P_i \log \frac{1}{P_i} = 0.7 \log \frac{1}{0.7} + 2 \times 0.15 \log \frac{1}{0.15}$$

$$= 1.1813 \text{ bits/msg Symbol} //$$

$\therefore$ code efficiency $\eta_c = \left[\frac{H(S)}{L}\right] \times 100\% = \left[\frac{1.1813}{1.3}\right] \times 100\%$

$$= 90.87\% //$$

2nd extension :- $S^2 = (3)^2 = 9$ Symbols //

$x_1x_1 = (0.7)(0.7) = 0.49$ //

$x_1x_2 = (0.7)(0.15) = 0.105$

$x_1x_3 = (0.7)(0.15) = 0.105$

$x_2x_1 = (0.15)(0.7) = 0.105$

$x_2x_2 = (0.15)(0.15) = 0.0225$

$x_2x_3 = (0.15)(0.15) = 0.0225$

$x_3x_1 = (0.15)(0.7) = 0.105$

$x_3x_2 = (0.15)(0.15) = 0.0225$

$x_3x_3 = (0.15)(0.15) = 0.0225$

$\therefore$ Non-Increasing probabilities :-

x_1x_1	x_1x_2	x_1x_3	x_2x_1	x_3x_1	x_2x_2	x_2x_3	x_3x_2	x_3x_3
0.49	0.105	0.105	0.105	0.105	0.0225	0.0225	0.0225	0.0225

Source Symbols	P_i	Code	Source S_a		Source S_b		Source S_c		Source S_d		Source S_e		Source S_f		Source S_g	
			P_i	Code	P_i	Code	P_i	Code	P_i	Code	P_i	Code	P_i	Code	P_i	Code
X_1X_1	0.449	1	0.449	1	0.449	1	0.449	1	0.449	1	0.449	1	0.449	1	0.51	0
X_1X_2	0.105	001	0.105	001	0.105	001	0.105	001	0.195	000	0.21	01	0.30	00	0.49	1
X_1X_3	0.105	010	0.105	010	0.105	010	0.105	010	0.105	001	0.195	000	0.21	01		
X_2X_1	0.105	011	0.105	011	0.105	011	0.105	011	0.105	010	0.105	001				
X_3X_1	0.105	0000	0.105	0000	0.105	0000	0.105	0000	0.105	011						
X_2X_2	0.0225	000110	0.045	00010	0.045	00010	0.090	0001								
X_2X_3	0.0225	000111	0.0225	000110	0.045	00011										
X_3X_2	0.0225	000100	0.0225	000111												
X_3X_3	0.0225	00010														

Table : Code Table for 2nd extension of Source.

the average length L_2 is given by

$$L_2 = \sum_{i=1}^{9} P_i l_i = (0.449)(1) + (0.105)(3) + (0.105)(3) + (0.105)(3) + (0.105)(4)$$
$$+ (0.0225)(6) + (0.0225)(6) + (0.0225)(6) + (0.0225)(6)$$

(L_2 → 2nd extension.)

$\therefore L_2 = 2.395$ bits/msg-Symbol

The Entropy of the 2nd. Extended Source is given by

$$H(S^2) = 2H(S) = 2(1.1813)$$
$$= 2.3626 \text{ bits/msg-Symbol}$$

∴ code efficiency $= \eta_c^{(2)} = \frac{H(S^2)}{L_2} = \left(\frac{2.3626}{2.395}\right) \times 100$

$= 98.65\%$ //

∴ The efficiency Improved by $98.65 - 90.87\%$

$\eta_c^{(2)} - \eta_c'$

$= 7.78\%$ (Improved)

* Design a trinary (Ternary) source code for the Source shown. using Huffman's Coding procedure

$S = \{S_1, S_2, S_3, S_4, S_5, S_6\}$, $P = \{1/3, 1/4, 1/8, 1/8, 1/12, 1/12\}$,

$X = \{0, 1, 2\}$.

2002 - Feb (5M). Jan - 2011 [12M]

2011 :- $q = r + (r-1)\alpha$ (OR) $n = \frac{N-r}{r-1}$

$\alpha = \frac{q-r}{r-1} = \frac{6-3}{3-1} = 3/2$ // $n = \frac{6-3}{3-1} = 3/2$ //

which is not an integer. The next consecutive value of "q" to have an integer value, for "α" is $q = 7$.

∴ add one dummy Symbols with '0' probability for which 'q' becomes.

$\alpha = \frac{7-3}{3-1} = \frac{4}{2} = 2$ // (OR) $n = \frac{7-3}{3-1} = 2$ //

Source Symbols	P_i	Ternary Code	Source S_a		Source S_b	
			P_i	Code	P_i	Code
S_1	$1/2$	1	$1/2$	1	$5/12$	0
S_2	$1/4$	2	$1/4$	2	$1/3$	1
S_3	$1/8$	01	$1/6$	00	$1/4$	2
S_4	$1/8$	02	$1/8$	01		
S_5	$1/12$	000	$1/8$	02		
S_6	$1/12$	001				
S_7	0	002				

* Design a quarternary and binary source code for the source shown using Huffman Coding procedure.

$S = \{ S_1\ S_2\ S_3\ S_4\ S_5\ S_6\ S_7 \}$.

$P = \{ 9/12, 3/12, 1/12, 2/12, 9/32, 1/12, 1/12 \}$

$Y = \{ 0, 1, 2, 3 \}$ & $X = \{ 0, 1 \}$ and find coding efficiency

July/Aug-2002, Jan-2010[10M]

Sol:- binary source:-

Source Symbol	P_i	Binary Code	Source S_a		Source S_b		Source S_c		Source S_d		Source S_e	
			P_i	Code	P_i	Code	P_i	Code	P_i	Code	P_i	Code
S_1	$9/32$	00	$9/32$	00	$9/32$	00	$9/32$	00	$14/32$	1	$18/32$	0
S_5	$9/32$	01	$9/32$	01	$9/32$	01	$9/32$	01	$9/32$	00	$14/32$	1
S_2	$3/32$	101	$6/32$	100	$6/32$	11	$8/32$	10	$9/32$	01		
S_3	$3/32$	110	$3/32$	101	$3/32$	100	$6/32$	11				
S_6	$3/32$	111	$3/32$	110	$3/32$	101						
S_7	$3/32$	1000	$3/32$	111								
S_4	$2/32$	1001										

$$L = \sum_{i=1}^{n} P_i l_i = \frac{9}{32}\times 2 + \frac{9}{32}\times 2 + \frac{3}{32}\times 3 + \frac{3}{32}\times 3 + \frac{3}{32}\times 3 + \frac{3}{32}\times 4 + \frac{1}{32}\times 4 + \frac{2}{32}\times 4 = 2.59375 \text{ brits/msg symbol}$$

$$H(s) = \sum_{i=1}^{7} P_i \log \frac{1}{P_i}$$

$$= \left(\frac{9}{32}\log\frac{32}{9}\right)(2) + \left(\frac{3}{32}\log\frac{32}{3}\right)(3) + \frac{2}{32}\log\frac{32}{1}$$

$$= 2.5 \text{ bits/msg symbol}$$

$$\therefore \text{coding efficiency } \eta_c = \left(\frac{H(s)}{L}\right)\times 100\% = \frac{2.56}{2.59375} = 98.7\%$$

Quartenary code, $r = 4$.

$$q = n + (n-1)\alpha = \frac{q-n}{n-1} = \frac{7-4}{4-1} = \frac{3}{3} = 1$$

Where $n = 4$, $q = 7$.

Source Symbols	P_i	Quartenary Code	Source S_q	
			P_i	Code
s_1	9/32	1	11/32	0
s_5	9/32	2	9/32	1
s_2	3/32	3	9/32	2
s_3	3/32	00	3/32	3
s_6	3/32	01		
s_7	3/32	02		
s_4	3/32	03		

The average length $L^{(M)}$ is given by.

$$L^{(M)} = \sum_{i=1}^{7} p_i l_i = \frac{9}{32}\times 1 + \frac{9}{32}\times 1 + \frac{1}{32}\times 1 + \frac{1}{32}\times 2 + \frac{1}{32}\times 3 + \frac{1}{32}\times 2 + \frac{3}{32}\times 2$$

$$= \frac{43}{32} = 1.34375 \text{ quarternary digits/msg-symbol}$$

entropy :-

$$H_4(S) = \frac{H(S)}{\log_2(4)} = \frac{2.56}{\log_2(4)} = 1.28 \text{ quarternary units/msg symbol}$$

$$\therefore \text{Coding efficiency } \eta_c^{(A)} = \frac{H_4(S)}{L^{(M)}} = \frac{1.28}{1.34375} \times 100\%$$

$$= 95.26\%$$

* The '5' symbols S_1, S_2, S_3, S_4 and S_5 of a source have probabilities 0.4, 0.2, 0.2, 0.1 & 0.1 Respectively. find the Code words using Huffman's algorithm.

Jan-Feb (2005) [5M] July-2011 [10M].

Source Symbols	P_i	Binary code.	Source S_a		Source S_b		Source S_c	
			P_i	Code	P_i	Code	P_i	Code
S_1	0.4	1	0.4	1	0.4	1	0.6	0
S_2	0.2	01	0.2	01	0.4	00	0.4	1
S_3	0.2	000	0.2	000	0.2	01		
S_4	0.1	0010	0.2	001				
S_5	0.1	0011						

* An Information Source produces a sequences of Independent Symbols having the following probabilities.

A	B	C	D	E	F	G
1/3	1/27	1/3	1/9	1/9	1/27	1/27

Construct binary and ternary code. using Huffman Encoding procedures and find its efficiency.

Sol:- Step 1:- Firstly. find the values of "n"

$$n = \frac{N-r}{r-1} = \frac{7-2}{2-1} = 5 \text{ //.} \rightarrow \text{(stages)}$$

As "n" is an integer. There is no need for adding any dummy Symbols.

Source Symbols	P_i	Code	Stage 1 Source S_a P_i	Stage 1 Source S_a Code	Stage 2 Source S_b P_i	Stage 2 Source S_b Code	Stage 3 Source S_c P_i	Stage 3 Source S_c Code	Stage 4 Source S_d P_i	Stage 4 Source S_d Code	Stage 5 Source S_e P_i	Stage 5 Source S_e Code
A	1/3	1	1/3	1	1/3	1	1/3	1	1/3	1	2/3	0
C	1/3	00	1/3	00	1/3	00	1/3	00	1/3	00	1/3	1
D	1/9	011	1/9	011	1/9	011	1/9	010	1/3	01		
E	1/9	0100	1/9	0100	1/9	0100	1/9	011				
B	1/27	01011	2/27	01010	2/9	0101						
F	1/27	010100	1/27	01011								
G	1/27	010101										

Table:- Binary Coding

Source efficiency is given by.

$$\eta_s = \frac{H(s)}{L} \times 100\%$$

$$H(s) = \sum_{K=1}^{N} P_K \log \frac{1}{P_K} = 2 \times \tfrac{1}{3} \log(3) + 2 \times \tfrac{1}{9} \log(9) + \tfrac{2}{27} \log 27$$

$$= 2.2892 \text{ bits/symbol} //$$

$$L = \sum_{i=1}^{N} P_i l_i = 2 \times \tfrac{1}{3} + \tfrac{1}{3} \times 2 + \tfrac{1}{9} \times 1 + \tfrac{1}{9} \times 4 + \tfrac{1}{27} \times 5 + \tfrac{1}{27} \times 6 + \tfrac{1}{27} \times 6. = 2.4074 \text{ bits/symbol} //$$

$$\eta_s = \left(\frac{2.2892}{2.4074}\right) \times 100\% = 95\% //$$

$$R_{\eta_s} = 1 - \eta_s = 4.90\% //$$

Ternary coding :- $n = \frac{N-r}{r-1} = \frac{7-3}{3-1} = 2 //$

which is an integer. Hence no need to add any dummy symbols.

Symbol	P_i	Code	Stage 1 (S1) Source S_a		Stage 2 (S2) Source S_b	
			P_i	Code	P_i	Code
A	1/3	0	1/3	0	1/3	0
	1/3	1	1/3	1	1/3	1
	1/9	20	1/9	20	1/3	2
	1/9	21	1/9	21		
	1/27	220	1/9	22		
F	1/27	221				
G.	1/27	222				

$$\eta_s = \frac{H_3(s)}{N} \times 100\%$$

$$H_3(s) = \sum_{k=1}^{N} P_k \log_3 \frac{1}{P_k}$$

(OR)

$$\frac{H(s)}{\log_2 3} = \frac{2.2892}{\left(\frac{\log(1)}{\log(2)}\right)}$$

$$= 1.4441 \text{ ternary/symbol}$$

$$L = \sum_{i=1}^{N} P_i l_i = Y_1(1) + (Y_1)(1) + (Y_2)(2) + (Y_4)(2) + (Y_{12})(3) + (Y_{27})(3) + (Y_{29})(1)$$

$$= 1.4444 \text{ ternary/symbol}$$

$$\eta_s = \frac{1.4444}{1.4444} \times 100\% = 100\%$$

$$R_{\eta_s} = (1 - \eta_s) \times 100\% = 0\%$$

* A Source has an alphabet Consisting of Seven Symbols A, B, C, D, E, F & G with probabilities of $\frac{1}{4}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{16}, \frac{1}{16}$ Respectively. Construct Huffman Quaternary Code. Find Coding efficiency July-2016 [5M]

Sol: $r = 4$

$$\alpha = \frac{q-r}{r-1} = \frac{7-4}{4-1} = \frac{3}{3} = 1$$

Source Symbols	P_i	Code	Source S_a	
			P_i	Code
A	0.25	1	0.375	0
B	0.25	2	0.25	1
C	0.125	3	0.25	2
D	0.125	00	0.125	3
E	0.125	01		
F	0.0625	02		
G	0.0625	03		

$L^{(1)}$ (or) $L = \sum_{i=1}^{7} P_i l_i = (0.25)(1) + (2)(0.25) + (0.125)(1) + (0.125)(2) + (0.125)(2) + (0.0625)(2) + (0.0625)(2)$

$= 1.375$ bits/msg Symbol

$$H(x) = \sum_{i=1}^{7} P_i \log \frac{1}{P_i} = \left(0.25 \log \frac{1}{0.25}\right)(2) + \left(0.125 \log \frac{1}{0.125}\right)(3) + (0.0625)(2)\left(\log \frac{1}{0.0625}\right)$$

$H(z) = 2.625$ bits/msg symbol //

$$H_4(z) = \frac{H(z)}{\log_2(4)} = \sum_{i=1}^{q} P_i \log_4 \frac{1}{P_i}$$

$$= \frac{2.625}{2} = 1.3125 \text{ bits/msg symbol} //$$

$$\eta_c = \left(\frac{H_4(z)}{L}\right)_{100\%} = \left(\frac{1.3125}{1.375}\right) 100\%$$

$$= 95.45 //$$

MISCELLANEOUS EXAMPLES

* A source produces Eight messages S_1 ---- S_8 with respective probabilities of 0.5, 0.125, 0.125, 0.0625, 0.0625, 0.03125 and 0.03125. obtain Shannon - Fano binary code and find efficiency and Redundancy. Also draw the code-tree.

S_1	0.5	1										
S_2	0.125	0	0.125	1	0.125	1						
S_3	0.125	0	0.125	1	0.125	0						
S_4	0.0625	0	0.0625	0	0.0625	0	0.0625	1				
S_5	0.0625	0	0.0625	0	0.0625	1	0.0625	0				
S_6	0.0625	0	0.0625	0	0.0625	0	0.0625	1				
S_7	0.03125	0	0.03125	0	0.03125	0	0.03125	0	0.03125	1		
S_8	0.03125	0	0.03125	0	0.03125	0	0.03125	0		0		

Code.	l_i
1	1
011	3
010	3
0011	4
0010	4
0001	4
00001	5
00000	5

Average length. $L = \sum_{i=1}^{8} P_i L_i$

$= (0.5)(1) + (0.125)(3)(2) + (0.0625)(4)(3)$

$+ (0.03125)(5)(2)$

$= 2.3125$ bits / msg Symbol //

Entropy $H(x) = \sum_{i=1}^{8} P_i \log \frac{1}{P_i}$

$= 0.5 \log \frac{1}{0.5} + 2 \times 0.125 \log \frac{1}{0.125} + 3 \times 0.0625 \log \frac{1}{0.0625}$

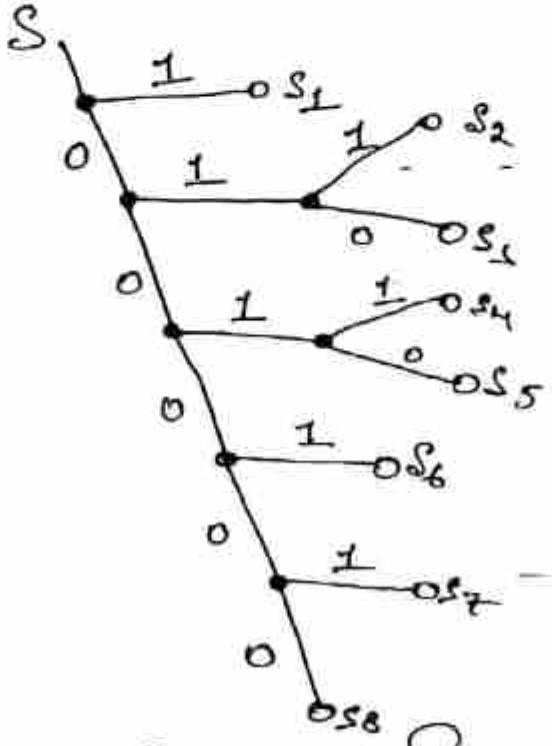
$+ 2 \times 0.03125 \log \frac{1}{0.03125}$

$= 2.3125$ bits / msg Symbol //

Coding efficiency $\eta_c = \frac{H(x)}{L} = \frac{2.2185}{2.2125} \times 100\%$ $\eta_c = 100\%$ //

$R_{\eta_c} = (1 - \eta_c) \times 100\% = 0\%$ //

Code Tree:



* A zero-memory Information Source produces 5 symbols. S_1, S_2, S_3, S_4 and S_5 with probability of 0.42, 0.17, 0.16, 0.13, and 0.12 respectively.

1) Construct Shannon-Fano binary code.
2) Find. Code efficiency & Redundancy.
3) Determine probabilities of "0" and "1" in the code

Sol:

S_1	0.42	1							
S_2	0.17	0	0.17	1	0.17	1			
S_3	0.16	0	0.16	1	0.16	0			
S_4	0.13	0	0.13	0	0.13	1			
S_5	0.12	0	0.12	0	0.12	0			

Code	l_i
1	1
011	3
010	3
001	3
000	3

Average Length $L = \sum_{i=1}^{5} P_i (i) = (0.4)(1) + (0.17+0.16+0.13+0.12)(2)$

$= 2.16$ bits/msg-Symbol //

$H(x) = \sum_{i=1}^{5} P_i \log \frac{1}{P_i} = 0.42 \log \frac{1}{0.42} + 0.17 \log \frac{1}{0.17} + 0.16 \log \frac{1}{0.16}$

$+ 0.13 \log \frac{1}{0.13} + 0.12 \log \frac{1}{0.12}$

$= 2.133$ bits/msg Symbol //

$\eta_c = \left(\frac{H(x)}{L}\right) \times 100 = \left(\frac{2.133}{2.16}\right) \times 100 = 98.75\%$ //

$R_{2c} = (1 - \eta_c) \times 100 = 1.25\%$ //

$P(\text{"0" in the code}) = \frac{1}{2.16}[(1)(0.17)+(2)(0.16)+(2)(0.13)+(3)(0.12)]$

$= 0.5139$ //

$P(\text{"1" in the code}) = \frac{1[(1)(0.42)+(2)(0.17)+(1)(0.16)+(1)(0.13)]}{2.16}$

$= 0.4861$ //

* A Source produces are Symbols $X_1, X_2, X_3 \ldots X_9$ with Respective probabilities of 0.24, 0.23, 0.19, 0.13, 0.08, 0.06, 0.04, 0.02 and 0.01.

1) Construct a Shannon-Fano Ternary Code.
2) Determine the Code efficiency and Redundancy
3) Draw Code Tree
4) Determine the probabilities of 0, 1 and 2 when the Encoding alphabet is $\{0, 1, 2\}$

P.T.O.

1)

X_1	0.24	2						
X_2	0.23	1	0.23	2				
X_3	0.19	1	0.19	1				
X_4	0.13	0	0.13	2				
X_5	0.08	0	0.08	1	0.08	2		
X_6	0.06	0	0.06	1	0.06	1		
X_7	0.04	0	0.04	0	0.04	2		
X_8	0.02	0	0.02	0	0.02	1		
X_9	0.01	0	0.01	0	0.01	0		

Code	L_i
2	1
12	2
11	2
02	2
012	3
011	3
002	3
001	3
000	3

2) Average Length $L = \sum_{i=1}^{9} P_i L_i$

$= (0.24)(1) + (0.23+0.19+0.13)(2) + (0.08+0.06+0.04+0.02+0.01)(3)$

$= 1.97$ Ternary digits / msg Symbol //

Entropy $H_3(S) = \frac{H(S)}{\log_2 3} = \frac{1}{\log_2(3)} \left[\sum_{i=1}^{9} P_i \log \frac{1}{P_i} \right]$

$= \frac{1}{\log_2 3} \left[0.24 \log \frac{1}{0.24} + 0.23 \log \frac{1}{0.23} + 0.19 \log \frac{1}{0.19} + 0.13 \log \frac{1}{0.13} + 0.08 \log \frac{1}{0.08} + 0.06 \log \frac{1}{0.06} + 0.04 \log \frac{1}{0.04} + 0.02 \log \frac{1}{0.02} + 0.01 \log \frac{1}{0.01} \right]$

$H_3(S) = 1.716$ Ternary Units / msg — Symbol

$\therefore$ Code efficiency $= \frac{H_3(S)}{L} \times 100\% = 87.11\%$ // $\therefore R_D = 12.89\%$

3)

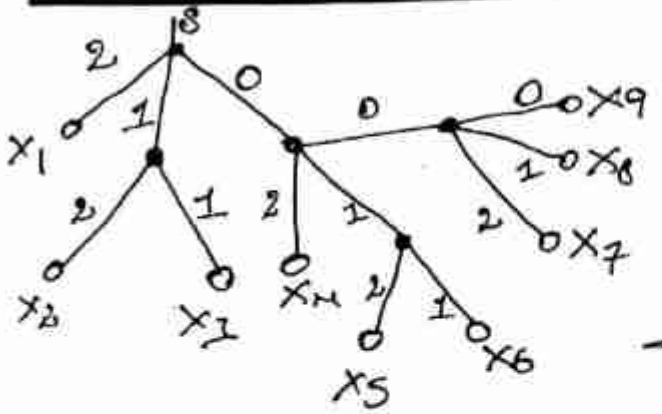


fig: Code-Tree.

4) $P(\text{'0's in the Code}) = \frac{1}{L}\sum_{i=1}^{q}[\text{number of '0's in the Code for } X_i][P_i]$

$= \frac{1}{1.97}[(1)(0.13)+(1)(0.08)+(1)(0.06)+(2)(0.04)+(2)(0.02)+(3)(0.01)]$

$= \frac{0.42}{1.97} = 0.2211$ //

$P(\text{'1's in the Code}) = \frac{1}{L}\sum_{i=1}^{q}[\text{number of '1's in the Code for } X_i][P_i]$

$= \frac{1}{1.97}[(1)(0.23)+(2)(0.19)+(1)(0.08)+(2)(0.06)+(1)(0.02)]$

$= \frac{0.81}{1.97} = 0.4213$ //

$P(\text{'2's in the Code}) = \frac{1}{1.97}[(1)(0.24)+(1)(0.23)+(1)(0.13)+(1)(0.08)+(1)(0.04)]$

$= \frac{0.72}{1.97} = 0.1655$ //

* A Source produces 5 Symbols S_1, S_2, S_3, S_4 and S_5 with respective probabilities of 0.1, 0.3, 0.4, 0.12 and 0.08.
1) Construct Huffman Binary Code.
2) Determine efficiency and Redundancy of the code.
3) Draw Code - Tree.

Sol :- 1)

Source Symbols	P_i	Code.	Source S_a		Source S_b		Source S_c	
			P_i	Code.	P_i	Code	P_i	Code.
S_3	0.4	1	0.4	1	0.4	1	0.6	0
S_2	0.3	00	0.3	00	0.3	00	0.4	1
S_4	0.12	011	0.18	010	0.3	01		
S_1	0.10	0100	0.12	011				
S_5	0.08	0101						

2) Average Length. $L = \sum_{i=1}^{5} P_i l_i$

$= (0.4)(1) + (0.3)(2) + (0.12)(3) + (0.10+0.08)(4)$

$= 2.8$ bits / msg Symbol //

Entropy $H(S) = \sum_{i=1}^{5} P_i \log_2 \frac{1}{P_i}$.

$= 0.4 \log \frac{1}{0.4} + 0.3 \log \frac{1}{0.3} + 0.12 \log \frac{1}{0.12} + 0.10 \log \frac{1}{0.10} + 0.08 \log \frac{1}{0.08}$.

$= 2.041$ bits / msg Symbol //

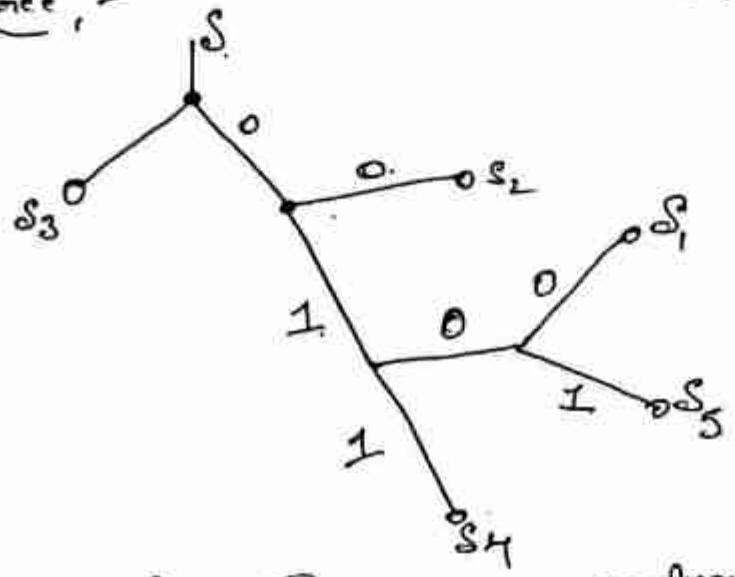
Code efficiency $\eta_c = \left(\frac{H(s)}{L}\right) \times 100\% = \frac{2.041}{2.08} \times 100\%$

$= 98.12\%$

Code Redundancy $= (1 - \eta_c) \times 100\%$

$= 1.88\%$ (or) 1.875

3) Code Tree :-



* Apply Huffman Encoding procedure for the following message ensemble and find Code efficiency and Redundancy. Draw Code Tree.

$[S] = [S_1, S_2, S_3, S_4, S_5, S_6, S_7, S_8]$

$P[S_i] = [0.36, 0.24, 0.12, 0.08, 0.08, 0.07, 0.03, 0.02]$

Encoding alphabet $= [0, 1, 2]$

NOTE: alphabet = 0, 1, 2 → Ternary code

Sol :- $q = r + (r-1)\alpha$

$\alpha = \frac{q-r}{r-1} = \frac{8-3}{3-1} = \frac{5}{2}$ //

Which is not an integer. The next minimum value for 'q' to have an integer value for 'r' is $q = 9$

∴ add one dummy symbols with '0' probabilities for which 'r' becomes

$$r = \frac{9-3}{3-1} = \frac{6}{2} = 3 \rightarrow \text{stages.}$$

Source Symbol	P_i	Code	Source S_a		Source S_b		Source S_c	
			P_i	Code	P_i	Code	P_i	Code
S_1	0.36	1	0.36	1	0.36	1	0.40	0
S_2	0.24	2	0.24	2	0.24	2	0.36	1
S_3	0.12	01	0.12	01	0.20	00	0.24	2
S_4	0.08	02	0.08	02	0.12	01		
S_5	0.08	000	0.08	000	0.08	02		
S_6	0.07	001	0.07	001				
S_7	0.01	0020	0.05	002				
S_8	0.02	0021						
S_9	0	0022	discard.					

$$L^{(2)} = \sum_{i=1}^{6} P_i l_i = (0.36+0.24)(1) + (0.12+0.08)(2) + (0.08+0.07)(3) + (0.01+0.02)(4)$$

$= 1.65$ ternary digits/msg-symbol

$$H_3(z) = \sum_{i=1}^{9} P_i \log_3 \frac{1}{P_i} \quad \text{(OR)} \quad \frac{H(z)}{\log_2(3)}$$

$$H_3(z) = 0.36 \log_3 \frac{1}{0.36} + 0.24 \log_3 \frac{1}{0.24} + 0.12 \log_3 \frac{1}{0.12} + 0.08 \log_3 \frac{1}{0.08} + 0.07 \log_3 \frac{1}{0.07} + 0.01 \log_3 \frac{1}{0.01} + 0.02 \log_3 \frac{1}{0.02}$$

$= 1.398$ ternary units/msg symbol

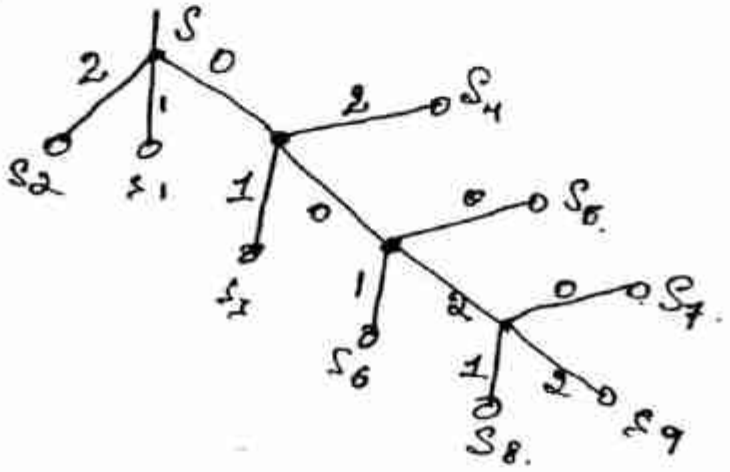
Code efficiency $= \frac{H_1(s)}{L^1} \times 100\%$ (or) $\frac{H_1(s)}{L} \times 100\%$

$= \frac{1.398}{1.05} \times 100\% = 84.76\%$ //

Code Redundancy $R_{\eta_{c(s)}} = (1 - \eta_{(s)}) \times 100\%$

$= 15.24\%$ //

Code-Tree:



* Apply Huffman's Encoding algorithm to the following message. Ensemble. $[x] = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9\}$

$P[s_i] = [0.45, 0.15, 0.1, 0.08, 0.08, 0.06, 0.04, 0.01, 0.01]$

Encoding alphabet $[X] = [0, 1, 2, 3]$

1) Find the Required quaternary Code
2) Finding efficiency and Redundancy.
3) Draw Code - Tree
4) Determine the probabilities of 0, 1, 2 and 3 in the code.

Sol:- Consider the Equation $q = r + (r-1)\alpha$.

$$\alpha = \frac{q-r}{r-1} = \frac{9-4}{4-1} = \frac{5}{3}.$$

which is not an integer. The next minimum value for α is $q = 10$. "q" to have an integer value for "α" i.e add one dummy symbols with "0" probabilities. for which "α" becomes.

$$\alpha = \frac{10-4}{4-1} = \frac{6}{3} = 2 \text{ stages}$$

1)

Source Symbols	P_i	Code.	Source S_a		Source S_b	
			P_i	Code	P_i	Code
s_1	0.45	0	0.45	0	0.45	0
s_2	0.15	2	0.15	2	0.30	1
s_3	0.10	3	0.10	3	0.15	2
s_4	0.08	10	0.08	10	0.10	3
s_5	0.08	11	0.08	11		
s_6	0.06	12	0.08	12		
s_7	0.04	120	0.08	13		
s_8	0.02	122				
s_9	0.01	121				
s_{10}	0	123 ← Discard.				

2) Avg. Length. $L^{(N)}$ (or) $L = \sum_{i=1}^{9} P_i l_i$.

$= (0.45 + 0.15 + 0.10)(1) + (0.08 + 0.08 + 0.06)(2) +$
$(0.04 + 0.01 + 0.01)(3)$. $= 1.18$ quaternary/msg Symbol

Entropy $H_4(x) = \sum_{i=1}^{9} P_i \log_4 \frac{1}{P_i}$ (or) $\frac{H(x)}{\log_2(4)}$.

$= 0.45 \log_4 \frac{1}{0.45} + 0.15 \log_4 \frac{1}{0.15} + 0.10 \log_4 \frac{1}{0.10} +$
$2 \times 0.08 \log_4 \frac{1}{0.08} + 0.06 \log_4 \frac{1}{0.06} + 0.04 \log_4 \frac{1}{0.04}$
$+ 0.01 \log_4 \frac{1}{0.01} + 0.01 \log_4 \frac{1}{0.01}$.

$= 1.2458$ quaternary Units / msg - Symbol

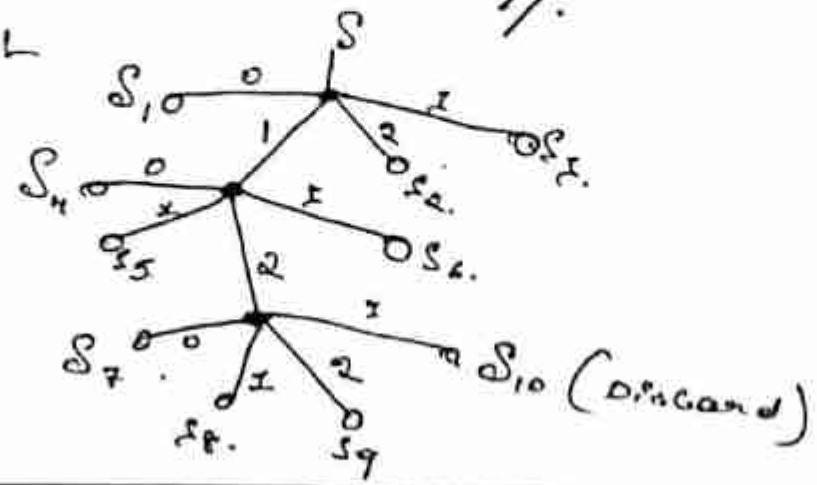
Code efficiency $\eta_{c(4)}$ (or) $\eta_c = \frac{H_4(x)}{L^{(4)}}$ (or) $\frac{H_4(x)}{L}$.

$= \frac{1.2458}{1.18} \times 100\%$

$= 90.28\%$

R_{nc} (or) $R_{nc(4)} = 9.72\%$.

3) Code Tree,



$$\Rightarrow P(\text{'0's in the Code}) = \frac{(0.415)(1) + (0.08)(1) + (0.04)(1)}{1.38}$$

$$= 0.4130$$

$$P(\text{"1"s in the Code}) = \frac{(0.08)(1) + (0.08)(2) + (0.06)(1) + (0.04)(1) + (0.03)(2) + (0.01)(1)}{1.38}$$

$$= 0.2971$$

$$P(\text{"2"s in the Code}) = \frac{(0.15)(1) + (0.04)(1) + (0.03)(1) + (0.01)(2)}{1.38}$$

$$= 0.1739$$

$$P(\text{"3"s in the Code}) = \frac{(0.1)(1) + (0.06)(1)}{1.38}$$

$$= 0.1160$$

NOTE

Check $p(0) + p(1) + p(2) + p(3) = 0.413 + 0.2971 + 0.1739 + 0.116 = 1$

LEMPEL - ZIV ALGORITHM

* LZ algorithm is a dictionary based approach in which a Common dictionary is built at both Encoder and Receiver End. depending upon the String to be Sent.
* The Codeword will be of the form <Index, code> where 'Index' is the Index of the dictionary where the longest match for the input sequence is found, 'Code' is the Codeword for the Symbol Succeeding the Longest match.
* Index "0" is used if the Symbol is Encountered for the first time and not found in the dictionary.
* The dictionary is updated in accordance with the input String.
* Longer the String to be Encoded, better will be the Compression achieved. Exactly the Same process is done at the Receiver End for decoding.

EXAMPLE 1:

Encode the following Information using LZ algorithm

"THIS_IS_HIS_HIT"

Solution:-

Step 1: Dictionary will be Partially Empty. The first Letter is. The input String is "T" which is appearing for the first time. Thus, an Index of "0" followed by the Code "T" is Transmitted. Dictionary is updated.

with. The first entry "T" at index "1"

Step2: Similarly. Since. The. succeeding. Symbols. 'H', 'I', 'S' and. '_'. appear. for. The first time, Index. '0' in The first. Entry 'T' at index '1'

Step3: The next Symbol. in The. Input. String is "I" is. already. present in The dictionary. Hence. The algorithm. Checks. for The next Symbol in the String. i.e 'S' for a better match. But. The. String 'IS' is not present in The. updated. dictionary. Thus it is Encoded. as. <Index(I), Code(S)> and. The String. 'IS' is updated in The dictionary.

Step4: The next Symbol in The Input String is "_" That is present at Location. 5. However, the next Symbol Succeeding in the String. is "H" resulting in '_H' that. is not get updated. in the dictionary. Thus. the String '_H' is Coded. as <Index(_), Code(H)> and "H_" is updated. in the next Location. of The dictionary.

Step5:- The next. Symbol. in The String is "I" That is. present. in. dictionary. The Succeeding Symbol is "S" resulting. in "IS" That is also. present in The dictionary. Thus the algorithm. Takes. The next Symbol '_' for a better match. resulting in 'IS_' That. is not present. Thus. The Coded. Sequence would. be <Index(IS), Code(_)> and. The String 'IS_" is updated in The dictionary.

In above Steps are. repeated. for all further. Symbols in the String and. Corresponding Codewords are. generated.

Table 1 details the dictionary updating procedure for the given input string. The complete encoding procedure is presented in Table 2

Table 1: Dictionary in LZ algorithm.

Index	Symbol
1	T
2	H
3	I
4	S
5	—
6	IS
7	—H
8	IS—
9	HI
10	T—

Table 2: Encoded string.

Symbol	Encoded output
T	<0, code(T)>
H	<0, code(H)>
I	<0, code(I)>
S	<0, code(S)>
—	<0, code(—)>
IS	<3, code(S)>
—H	<5, code(H)>
IS—	<6, code(—)>
HI	<2, code(I)>
T—	<1, code(—)>

ARITHMETIC CODING :

*. Arithmetic Coding is one such technique. Where in a tag is assigned to each sequence of "m" symbols, A binary code is then generated for the obtained tag.

* In order to assure proper decoding, tag assigned to each set of symbols has to be unique. Generally a range of (0, 1) is considered.

*. note that "0" is included in the range but not "1". As there are infinite sample s/o this range, different tags can be assigned to different set of symbols.

KRAFT INEQUALITY [KRAFT – McMILLAN INEQUALITY]

A necessary and sufficient condition for the existence of an instantaneous code with word lengths $l_1, l_2, \ldots, l_q$ is that

$$\sum_{i=1}^{q} r^{-l_i} \leq 1$$

where r = number of different symbols used in the code alphabet X.

l_i = word length in binary digits of the code-word corresponding to i^{th} source symbols.

proof: The word lengths $l_1, l_2, \ldots, l_q$ are arranged in the ascending order so that

$$l_1 \leq l_2 \leq \ldots \leq l_q.$$

* If we have to choose "one-length" "code-words" for all the "q" number of source symbols satisfying prefix property, we can do so only when $q \leq r$.
* If $q > r$, then we have to go for combinations of "r" symbols to form instantaneous code-words.
* Let n_i represent the number of messages encoded into code words of length "i". Then we have

for $i = 1$, $n_1 \leq r$.

* For $i = 2$, for getting an instantaneous code, we must start encoding using $(r - n_1)$ symbols only as

The first digit and the second digit can be any of "r" symbols of the code alphabet.

∴ for i=2 $n_2 \le (r - n_1)$ Xer.

(or) $n_2 \le r^2 - n_1 r$.

Ily. $n_3 \le [(r^2 - n_1 r) - n_2]$ Xer.

$n_3 \le r^3 - n_1 r^2 - n_2 r$.

proceeding this way. we can arrive at

$n_i \le r^i - n_1 r^{(i-1)} - n_2 r^{(i-2)} - \cdots - n_{(i-1)} r$

$\therefore n_i + n_1 r^{(i-1)} + n_2 r^{(i-2)} + \cdots + n_{(i-1)} r \le r^i$

multiplying throughout by r^{-i} we get.

$n_i r^{-i} + n_1 r^{-1} + n_2 r^{-2} + \cdots + n_{(i-1)} r^{-(i-1)} \le 1$.

adopting the above in the reverse way, we get.

$n_i r^{-i} + n_{(i-1)} r^{-(i-1)} + n_{(i-2)} r^{-(i-2)} + \cdots + n_1 r^{-1} \le 1$

(or) $\sum_{n=1}^{i} n_n r^{-n} \le 1$. ——— (1)

Since the actual number of messages n_i has to be integer. we can write equation (1) as.

$\sum_{n=1}^{i} n_n r^{-n} = \underbrace{(r^{-1} + r^{-1} + \cdots + r^{-1})}_{n_1 \text{ terms}} + \underbrace{(r^{-2} + r^{-2} + \cdots + r^{-2})}_{n_2 \text{ terms}} + \cdots$

$\cdots + \underbrace{(r^{-i} + r^{-i} + \cdots + r^{-i})}_{n_i \text{ terms}} \le 1$.

$$\sum_{m=1}^{i} n_m r^{-m} = \underbrace{\sum_{j=1}^{n_1} r^{-1}}_{\text{Length } L_1 \text{ group.}} + \underbrace{\sum_{j=1}^{n_2} r^{-2}}_{\text{Length } l_2 \text{ group.}} + \cdots\cdots\cdots + \underbrace{\sum_{j=1}^{n_q} r^{-q}}_{\text{Length } L_q \text{ group.}} \leq 1$$

Combining all the groups (since $n_1 + n_2 + \cdots + n_i = q$) we can write.

$$\sum_{i=1}^{q} r^{-l_i} \leq 1 \rightarrow \text{proved.}$$

INFORMATION THEORY AND CODING

5TH SEM E&C



JAYANTHDWIJESH H P **M.tech (DECS)**

Assistant Professor – Dept of E&C

B.G.S INSTITUTE OF TECHNOLOGY (B.G.S.I.T)

B.G Nagara, Nagamangala Tq, Mandya District- 571448

FORMULAS FOR REFERENCE

MODULE – 5 (SOME IMPORTANT CYCLIC CODES AND CONVOLUTION CODES)

CHAPTER -1 SOME IMPORTANT CYCLIC CODES

❖ **Reed – Solomon (RS) codes**

Block length : $n = q - 1$

Number of parity check symbols : $n - k = \delta - 1 = 2t$

Number of message digits = Dimension : $k = q - 1 - 2t = q - \delta$

Minimum Distance : $d_{min} = \delta = n - k + 1 = 2t + 1$

Error correcting capability of the code : $t = \frac{n-k}{2}$

The weight spectrum of MDS codes, including RS codes, is known in exact form given by

$$A_w = {}_nC_w \sum_{m=0}^{w-(n-k+1)} (-1)^m \, {}_wC_m \left[q^{w-m-(n-k)} - 1\right],$$

$$w = n - k + 1, n - k + 2, \ldots\ldots n$$

The number of code words having minimum non-zero weight [i.e., nearest neighbours for a specified value of d_{min}] is given by

$$A_{dmin} = {}_nC_{d_{min}} (q - 1)$$

CHAPTER-2 CONVOLUTION CODES

In a general (n, k, m) convolutional encoder, the following notations are used.

n = number of outputs = number of modulo-2 adders (normally)

k = number of input bits entering at any time

m = number of stages of shift register

= number of flip-flops

L = number of bits in the message sequence.

Constraint length = m × n digits

rate efficiency = k/n

TIME DOMAIN APPROACH

❖ **The generator matrix G is given by**

$$G = \begin{bmatrix} g_1^{(1)} g_1^{(2)} g_1^{(3)} & g_2^{(1)} g_2^{(2)} g_2^{(3)} & g_3^{(1)} g_3^{(2)} g_3^{(3)} & \cdots & g_{m+1}^{(1)} g_{m+1}^{(2)} g_{m+1}^{(3)} & 0 & 0 & 0 & \cdots & 000 \\ 0 \; 0 \; 0 & g_1^{(1)} g_1^{(2)} g_1^{(3)} & g_2^{(1)} g_2^{(2)} g_2^{(3)} & \cdots & g_m^{(1)} g_m^{(2)} g_m^{(3)} & g_{m+1}^{(1)} g_{m+1}^{(2)} g_{m+1}^{(3)} & \cdots & 000 \\ 0 \; 0 \; 0 & 0 \; 0 \; 0 & g_1^{(1)} g_1^{(2)} g_1^{(3)} & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

- **Number of rows in [G] =L (one input)**
- **Number of rows in [G] =2L (two inputs)**
- **Number of columns in [G]= n(L+M)**
- **The output of the encoder is given by**

  [C] = [d][G]

TRANSFORM DOMAIN APPROACH

- **The output polynomials for the j(j=1,2,3…) adders are given by**

  $C^j(x) = d(x)\, g^j(x)$ **for j=1, 2, 3….**
- **The output of the encoder is given by**

  $C(X) = C^{(1)}(X^n) + X\, C^{(2)}(X^n) + X^2\, C^{(3)}(X^n) + \ldots\ldots\ldots\ldots + X^{n-1}\, C^{(n)}(X^n)$

JAN-2017[5M].
JAN-2015[5M]
JAN-2014[5M]
July-2013[5M]

MODULE – 5 :-

SOME IMPORTANT CYCLIC CODES

DEC-2011[5M], JUNE-2012[5M], JUNE-July-2009[5M], JUNE-2010[5M], DEC-2010[5M]

Golay Codes :- DEC-2009[5M], July-2011[5M], DEC-2012[5M]

Golay code is a (23, 12) binary code. Capable of correcting upto "3" errors in a block of 23 bits. It is a perfect binary code because it satisfies the Hamming bound with Equality Sign for $t = 3$.

Hamming bound : $2^{n-k} \geq \sum_{i=0}^{t} {}^{n}C_i$

For Golay code consider

$$2^k \cdot \sum_{i=0}^{t} {}^{n}C_i$$

given $n = 23$, $k = 12$, $t = 3$

$$2^{12} \sum_{i=0}^{3} {}^{23}C_i = 2^{12}\left[{}^{23}C_0 + {}^{23}C_1 + {}^{23}C_2 + {}^{23}C_3\right]$$

$$= 2^{12}(2048) = 2^{12} \cdot 2^{11} = 2^{23}$$

$$= 2^n$$ and hence the result.

The generator polynomial obtained from

$$x^{23} + 1 = (x+1)\, g_1(x)\, g_2(x)$$

Assume where $g_1(x) = 1 + x^2 + x^4 + x^5 + x^6 + x^{10} + x^{11}$

$g_2(x) = 1 + x + x^5 + x^6 + x^7 + x^9 + x^{11}$

* The Encoder can be implemented using Shift Register using either $g_1(x)$ or $g_2(x)$ as divider polynomials.
* The code has minimum distance $d_{min} = 7$

JUNE-2010[5M], DEC-2010[5M], DEC-2011[5M], DEC-2018[5M], JULY-2013[5M]

BCH Codes [Bose-Chaudhuri-Hocquenghem (BCH)]:-

→ One of the most Important and powerful classes of Linear block codes are BCH Codes, which are Cyclic Codes with a widely variety of parameters.

→ The most Common BCH codes are characterized as follows. Specially, for any +ve Integer "m" (equal to or greater than 3) and "t" [Less than $(2^m-1)/2$] there exists a binary BCH Code with the following parameters:

Block Length: $n = 2^m - 1$.

Number of message bits: $k \geq n - mt$

minimum distance: $d_{min} \geq 2t+1$

→ Each BCH code is a 't'-error correcting code in that it can detect and correct up to 't' random errors per code word.

→ The Hamming Single error correcting codes can be described as BCH Codes.

→ The BCH codes offer flexibility in the choice of code parameters, namely, blocklength and code rate.

→ Further more, at block lengths of a few hundred. (or) Less, the BCH codes are among the best known codes of the same block length and code rate.

CYCLIC REDUNDANCY CHECK CODES:

→ Cyclic codes are extremely well-suited for error detection.

→ We say this for two reasons. first, they can be designed to detect many combinations of likely errors. Second, the implementation of both encoding and error-detecting circuits is practical.

→ It is for these reasons that virtually all error-detecting codes used in practice are of the cyclic code type.

→ A cyclic code used for error detection is referred to as cyclic redundancy check (CRC) codes.

→ A CRC error burst of length "B" is an 'n'-bit received word or a contiguous sequence of "B" bits in which the 1st and last bits and any number of intermediate bits are received in error.

→ Error burst may also include an end-shifted version of a contiguous sequence.

→ In any event, binary (n, k) CRC codes are capable of detecting the following error patterns:

1) All CRC error bursts of length n−k (or less).
2) A fraction of CRC error bursts of length equal to n−k+1; the fraction equals $1-2^{-(n-k+1)}$.
3) A fraction of CRC error bursts of length greater than n−k+1; the fraction equals $1-2^{-(n-k)}$
4) All combinations of $d_{min}-1$ (or less) errors.

5) All error patterns with an odd number of errors if the generator polynomial $g(D)$ for the code has an even number of non-zero coefficients.

MAXIMUM-LENGTH CODES

⇒ For any +ve integer $m \geq 3$, there exists a maximum-length code with the following parameters.

Block length $n = 2^m - 1$.

number of message bits $k = m$.

minimum distance $d_{min} = 2^{m-1}$.

→ maximum length codes are generated by polynomial of the form.

$$g(D) = \frac{1+D^n}{h(D)}$$ //

where $h(D)$ = is any primitive polynomial of degree m.

→ maximum length codes are also referred to as PSEUDO-NOISE (PN) codes //.

→ PSEUDO-NOISE is derived from the fact that these codes have correlation and spectral characteristics that resemble those of a white noise sequence.

//

REED-SOLOMON (RS) CODES :-

The RS Codes are a special case of BCH Codes. The primary q-ary BCH codes have block length $n = q^m - 1$ and the extension field of $GF(q^m)$ is involved for locating roots of generator polynomial $g(x)$. Taking $m=1$, we get $n = q-1$, which gives the standard RS-codes. For the primitive BCH Codes, the generator polynomial is given by.

$$g(x) = LCM\left[\phi_{\alpha^i}(x), \phi_{\alpha^{i+1}}(x) \ldots\ldots \phi_{\alpha^{i+\delta-1}}(x)\right] \quad \text{---(1)}$$

Where.

$\phi_{\alpha^i}(x)$ = minimum polynomial of element α^i

α^i = power of primitive element over $GF(q^m)$

δ = designed distance (Designed value of minimum distance d_{min})

With $m=1$, the minimal polynomials over $GF(q)$ of elements in $GF(q)$ are first degree of the form $(x-\alpha^i)$ and $g(x)$ reduces to.

$$g(x) = (x-\alpha^i)(x-\alpha^{i+1}) \ldots\ldots\ldots (x-\alpha^{i+\delta-2}) \quad \text{---(2)}$$

This is polynomial over $GF(q)$ of degree Exactly $(\delta-1)$.

Thus $(n-k) = \delta - 1$ for RS Codes.

$\therefore n-k = \delta-1$.

The True minimum distance must be.

$d_{min} \geq \delta = n-k+1$. Hence for RS codes.

$d_{min} = n-k+1$.

* RS codes are "maximum distance Seperable" codes (MDS codes).
* for any (n, k) Linear Block code, the minimum distance $d_{min} \leq (n-k+1)$, But for RS code $d_{min} = n-k+1$.
* The value of "k" may be selected as any integer upto $(n-1)$.
* RS Codes Can be shortened Without decreasing there (or) changing $(n-k)$ and thus RS Codes are also MDS.

Thus a t-error Correcting q-ary RS Code with symbol from GF(q) has the following parameters

Block length : $n = q-1$

number of parity check symbols : $n-k = \delta - 1 = 2t$.

number of message digits = Dimension : $k = q-1-2t$

$k = q-\delta$

Minimum Distance : $d_{min} = \delta = n-k+1$

$\delta = 2t+1$. //

Error Correcting Capability of the code : $t = \frac{n-k}{2}$.

The Weight Spectrum of MDS Codes, including RS Codes is known in exact form given by

$$A_w = {}^nC_w \cdot \sum_{m=0}^{w-(n-k+1)} (-1)^m {}^wC_m \left[q^{w-m-(n-k)} - 1 \right],$$

$w = n-k+1, n-k+2, \ldots\ldots n.$

The number of code words having minimum non zero weight is given by

$$A_{dmin} = n C_{dmin} (q-1)^{d_{min}} //$$

* Determine the parameters of a q-ary RS code over GF(256) for a $d_{min} = 33$

Sol :-

The Block Length $n = q-1 = 256-1$
$= 255 //$

Dimension : $k = q - \delta$.
$= q - d_{min}$.
$k = 256 - 33$
$= 223 //$

$d_{min} = \delta$

number of parity check symbols $= n-k$
$= 255 - 223$
$= 32 //$

$\therefore n-k$.

Error Correcting Capability $t = \frac{n-k}{2} = \frac{32}{2} = 16 //$

$\therefore$ the parameters of RS code over GF(256) are

Block Length : $n = 255$
Number of parity check Symbol : $n-k = 32$.
number of message digits : $k = 233$
Minimum Distance : $d_{min} = 33$
Error Correcting Capability : $t = 16 //$

* Determine the parameters of a q-ary R-S code over GF(16) for a $d_{min} = 9$. Also find the total number of code-words in the code and also the nearest neighbours for any code word at a distance of $d_{min} = 9$.

Sol :- The block length $n = q - 1 = 16 - 1 = 15$ //

number of parity check symbols : $n - k = 15 - 7 = 8$ //.

Error. Correcting Capability : $t = \frac{n-k}{2} = \frac{8}{2} = 4$ //.

number of message digits : $k = q - 8$.
$= q - d_{min}$
$= 16 - 9$
$= 7$ //.

minimum. distance : $d_{min} = 2t + 1$
$= 2(4) + 1 = 9$ //.

Ie. parameters of R-S code over GF(16) are given by.

Block length : $n = 15$ //

number of parity check symbols : $n - k = 8$ //

number of message digits : $k = 7$ //.

minimum. distance : $d_{min} = 9$ //.

Error. Correcting Capability : $t = 4$ //.

q : $k + d_{min}$.
$7 + 9 = 16$ //.

The Total number of Code-words in the Code is Calculated by using the A_w Equation. With $n=15$, $q=16$, $k=7$ //

$$A_w = nC_w \sum_{m=0}^{w-(n-k+1)} (-1)^m \, {}_wC_m \left[q^{w-m-(n-k)} - 1 \right), \quad w = n-k+1, n-k+2, \ldots n$$

$$A_w = 15C_w \sum_{m=0}^{w-(15-7+1)} (-1)^m \, {}_wC_m \left[16^{w-m-(15-7)} - 1 \right] \quad w = 9, 10, \ldots 15$$

$$A_w = 15C_w \sum_{m=0}^{w-9} (-1)^m \, {}_wC_m \left[16^{w-m-8} - 1 \right]$$

For $w=9$.

$$A_9 = 15C_9 \, (-1)^{(0)} \, 9C_0 \left[16^{9-0-8} - 1 \right]$$

$$= 15C_9 \, (1)(1) \left[16^1 - 1 \right] = 75075 //$$

$w=10$.

$$A_{10} = 15C_{10} \sum_{m=0}^{1} (-1)^m \, 10C_m \left(16^{2-m} - 1 \right)$$

$$= 15C_{10} \, (-1)^0 \, 10C_0 \left(16^2 - 1 \right) + 15C_{10} \, (-1)^1 \, 10C_1 \left(16^{2-1} - 1 \right)$$

$$= 3003 \times 1 \times 1 \times 255 + 3003 \, (-1)(10)(15)$$

$$= 315315 //$$

$w=11$

$$A_{11} = 15C_{11} \sum_{m=0}^{2} (-1)^m \, 11C_m \left(16^{3-m} - 1 \right)$$

$$= 2886975 //$$

$w=12$

$$A_{12} = 15C_{12} \sum_{m=0}^{3} (-1)^m \, 12C_m \left(16^{4-m} - 1 \right)$$

$$= 13615875 //$$

$w = 13$

$$A_{13} = {}_{15}C_{13} \sum_{m=0}^{4} (-1)^m \cdot {}_{13}C_m \left(16^{5-m} - 1\right) = 47651625 //$$

$w = 14$

$$A_{14} = {}_{15}C_{14} \sum_{m=0}^{5} (-1)^m \, {}_{14}C_m \left(16^{6-m} - 1\right) = 1.0086705 \times 10^8 //$$

$w = 15.$

$$A_{15} = {}_{15}C_{15} \sum_{m=0}^{6} (-1)^m \, {}_{15}C_m \left(16^{7-m} - 1\right) = 1.0195817 \times 10^8 //$$

∴ Total number of Code words. $= A_9 + A_{10} + A_{11} + A_{12} + A_{13} + A_{14} + A_{15}$

$= 75075 + 315315 + 2886995 + 13615875 + 47651625 + 1.0086705 \times 10^8 + 1.0195817 \times 10^8$

$= 2.67 \times 10^8 //$

The nearest neighbours for any Code-word at a distance of $d_{min} = 9$. As given by.

$$A_{d_{min}} = {}_nC_{d_{min}} (q-1)$$

$$A_9 = {}_{15}C_9 (16-1) = 75075 //$$

Application of RS Codes:

1) The RS Codes are a natural means of Coding for M-ary modulation Techniques.
2) Storage devices (Including Compact DISC, Blue-ray DISC, DVD, barcodes, etc).

3) Wireless (or) mobile Communication. (Including Cellular Telephones, interplane links etc)

4) Satellite Communication.

5) Digital television / Digital video Broadcast (DVB)

6) high-Speed modems such as those employing ADSL, XDSL etc

7) VLSI Implementation using RS Codes

8) RS Codes may be used in conjunction with known Interpolation techniques.

DEC-2012[5M], July-2013[5M], JAN-2014[5M] DEC-2010[5M], DEC-2009[5M], JUNE/JULY-2009[5M], JUNE-2010[5M], JULY-2011[5M], DEC-2011[5M], JUNE-2014[5M]

SHORTEND CYCLIC CODES:-

* The Cyclic Codes so far considered have generator polynomials that are divisors of X^n+1.
* In general the polynomial X^n+1 has relatively few divisors and as a result there are usually very few Cyclic Codes of a given length
* To overcome this difficulty, to Construct the useful codes we have to increase the number of parity (n,k) hence Cyclic Codes are often used in shortened form.
* The last "i" Information digits are always taken to be "0"s. These bits are not transmitted: the decoder for the original Cyclic Code can decode the

Shortened. Code - vectors. Simply by padding. The Received (n-1) - tuples. with '0's.

* Hence, given an (n-k) Cyclic. Code, it is always possible to Construct a (n-1, k-1) Shortend cyclic Code.

* The Shortend. Cyclic code is a subset of the Cyclic. Code. from which it was derived. and hence its minimum distances and. Error. Correcting ability is at least as great as that of the original Code.

* Shortend. Cyclic Codes are used. in The Implementation. advantages. & mathematical structure of Cyclic Codes.

DEC-2009 [5M], DEC-2011 [5M]. JUNE-2012 [5M]. JAN-2016 [5M] JUNE/JULY-2018 [5M]

BURST — ERROR. CORRECTING CODES :- JAN-2017 [5M]

* The presence of Impulse noise affects more than one Symbol (or) bit to cause. burst errors.

* Codes used. for. Correcting random errors. are not. Sufficient for Correcting burst errors. Special Codes have been developed. for. Correcting. burst errors.

Burst of Length. "q" :- It is defined. as a. vector. whose non-zero Components are confined to "q" Consecutive digit positions. The first and last of which are non-zero.

A code which is Capable. of Correcting all burst errors of length. "q" (or) less is called. "q"-burst error. Correcting Code and it is said. to have burst - error. Correcting ability q.

The condition for a q-burst error correcting Code (n,k) is that the number of parity bits must at least be 2q.

$$\langle i.e\ n-k \geq 2q \rangle$$

The upper bound on the burst error correcting capability of an (n,k) code is

$$\left\langle q \leq \frac{n-k}{2} \right\rangle //$$

July-2011[5M], JAN-2014[5M]

BURST AND RANDOM ERROR CORRECTING CODES :-

* In practical systems, errors occur neither independently at Random nor in well defined bursts.
* Random error Correcting Codes or Single-burst-error Correcting Codes will be inadequate for combating a mixture of Random and burst errors.
* It is better to design Codes capable of Correcting Random errors and/or Single or multiple bursts.
* the most effective method is the "INTERLACING TECHNIQUE".
* Given a (n,k) cyclic Code, it is possible to Construct a $(\lambda n, \lambda k)$ Cyclic "Interleaved Code" by simply arranging "λ" Code-vectors of the

original Code into 'λ' Rows of a Rectangular array and Transmitting them Column by Column.
* The Resulting Code is Called an interlaced Code with an interlacing degree λ. //

Problem :-
* Consider a (15, 9) Cyclic Code generated by $g(x) = 1+x^3+x^4+x^5+x^6$. This Code has burst error Correcting ability b=3. Find the burst error Correcting efficiency of this Code.
Aug-2011 [4M] Feb-2002 (4M)

Sol :-
The burst-error Correcting efficiency 'Z' of an (n, k) Cyclic Code is defined by

$$Z = \frac{2b}{n-k}$$

Given that b=3 n=15 k=9

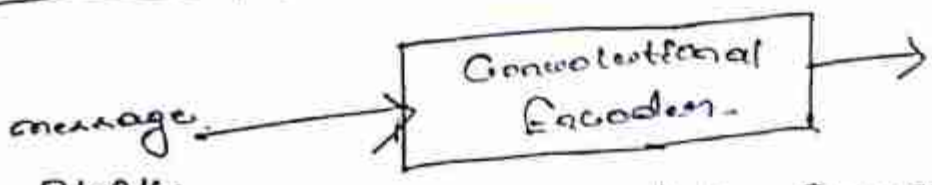
$$Z = \frac{2\times3}{15-9} = 1 \;//\; \Rightarrow 100\% \;//.$$

CONVOLUTIONAL CODES :-

In "block codes" a block of "n" digits generated by the encoder in a particular time unit depends only on the block of "k" input message digits within that time unit.

In Convolutional codes, a block of "n" code digits generated by the encoder in a time unit depends on not only the block of "k" message digits within that timeunit but also on the preceding (m-1) block of message digits (m>1). usually the values of "k" and "n" will be small.

Encoder for Convolutional Codes :-



* A Convolutional Encoder, takes sequences of message digits and generates sequence of code digits.

* A message block consisting of "k" digits is fed into the encoder & the encoder generates a code block consisting of "n" code digits (k<n)

* The "n"-digit code block depends not only on the "k"-digit message block of the same time unit, but also on the previous (m-1) message blocks.

* The code generated by the above encoder is called as (n,k,m) Convolutional code of constraint length "nm" digits and "code efficiency k/n".

The following notations are used from a general (n, k, m) Convolutional Encoder :-

n = number of outputs = number of modulo-2 adders

k = number of i/p bits entering at any time.

m = number of stages of shift register

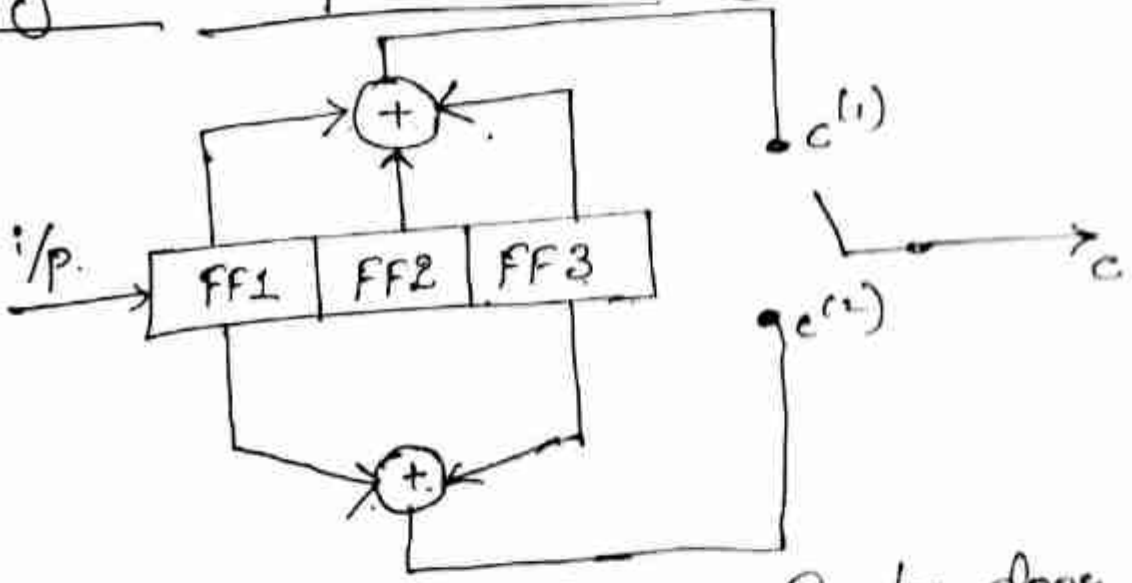
= number of flipflops.

L = number of bits in the message sequence (unencoded)

Length = m x n digits.

Rate efficiency = k/n.

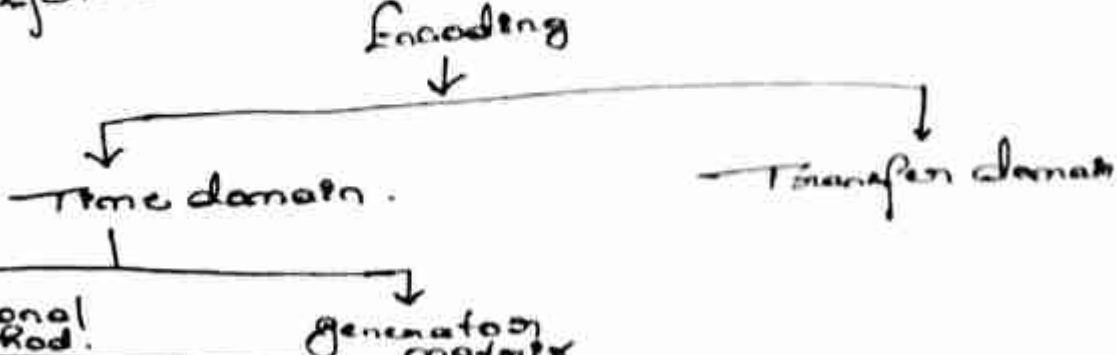
Diagramatic Representation of Convolutional Encoder ;



In Convolution Codes, Encoding can be done in 2 methods

1) Time domain.

2) Transfer domain.



Jn. Dec-2009 [14M], Dec-2012. [10M]

* Consider a (2, 1, 3) Convolution Encoder. find the output sequence for the input message $d = 10111$ using a) Time Domain approach.
b) Transform Domain approach. for $g^{(1)} = 1011$, $g^{(2)} = 1111$.

Sol:- given. $(n, K, m) = (2, 1, 3)$

n = number of modulo-2 adder (or) outputs = 2//

K = number of inputs = 1.

m = number of flipflops = 3//.

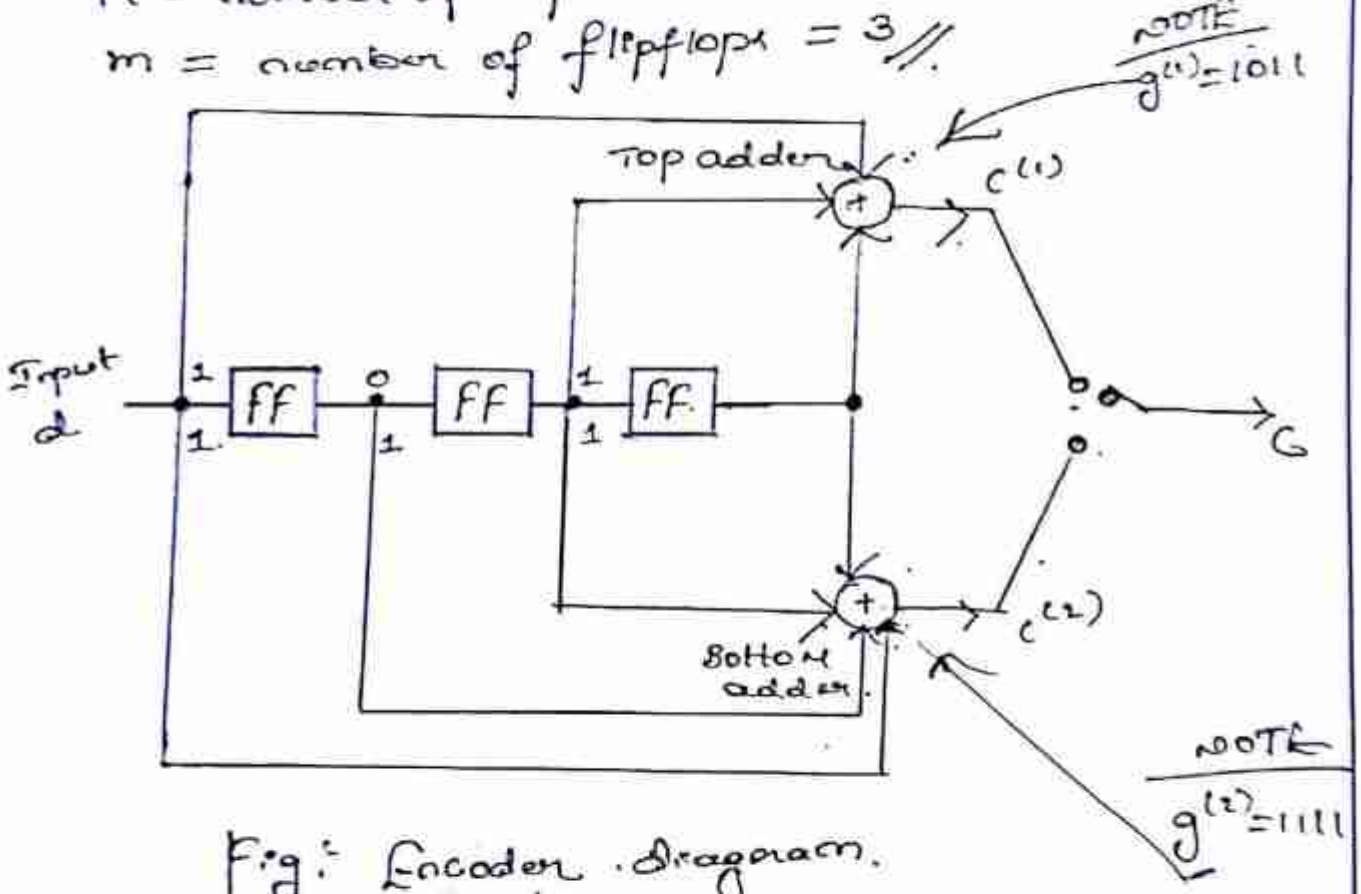


Fig: Encoder diagram.

(a) Time Domain approach:-

* For the Top adder, the generator Sequence is given by

$$g^{(1)} = \begin{bmatrix} g_1^{(1)} & g_2^{(1)} & g_3^{(1)} & g_4^{(1)} \end{bmatrix} = \begin{bmatrix} 1011 \end{bmatrix}_{//}$$

* Wy for the bottom adder. the generator Sequence

$$g^{(2)} = \begin{bmatrix} g_1^{(2)} & g_2^{(2)} & g_3^{(2)} & g_4^{(2)} \end{bmatrix} = \begin{bmatrix} 1111 \end{bmatrix}_{//}$$

* The generator matrix is of the order $L \times [n(L+N)]$

where L = number of bits in the message sequence = 5

$L \times [n(L+N)]$

$5 \times [2(5+3)]$

5×16 //

i.e 5 rows × 16 columns //

root

d = 10111

5 message sequence number.

$$G = \begin{bmatrix} 11 & 01 & 11 & 11 & 00 & 00 & 00 & 00 \\ 00 & 11 & 01 & 11 & 11 & 00 & 00 & 00 \\ 00 & 00 & 11 & 01 & 11 & 11 & 00 & 00 \\ 00 & 00 & 00 & 11 & 01 & 11 & 11 & 00 \\ 00 & 00 & 00 & 00 & 11 & 01 & 11 & 11 \end{bmatrix}$$

root

$g^{(1)} = 1011$ $g^{(2)} = 111$

$u = [11\ 01\ 11\ 11]$

∴ To find the code vector [c] is given by

$$[c] = [d][G]$$

$$= [10111] \begin{bmatrix} 11 & 01 & 11 & 11 & 00 & 00 & 00 & 00 \\ 00 & 11 & 01 & 11 & 11 & 00 & 00 & 00 \\ 00 & 00 & 11 & 01 & 11 & 11 & 00 & 00 \\ 00 & 00 & 00 & 11 & 01 & 11 & 11 & 00 \\ 00 & 00 & 00 & 00 & 11 & 01 & 11 & 11 \end{bmatrix}$$

$$= [11\ 01\ 00\ 01\ 01\ 01\ 00\ 11]$$ //

P.T.O.

b) Transform domain approach:

* We have for the message $d = 10111$ the message polynomial given by

$$d = 10111$$
$$d(x) = 1 + x^2 + x^3 + x^4 \text{ //}$$

* We have $g^{(1)} = 1011$

$$g^{(1)}(x) = 1 + x^2 + x^3 \text{ //}$$

$$g^{(2)} = 1111$$
$$g^{(2)}(x) = 1 + x + x^2 + x^3 \text{ //}$$

* The Top-adder output polynomial is given by.

$$C^{(1)}(x) = d(x)\, g^{(1)}(x) \Rightarrow C^{(1)}(x) = d(x)\, g^{(1)}(x).$$
$$= (1 + x^1 + x^3 + x^4)(1 + x^1 + x^3)$$
$$= 1 + x^2 + x^3 + x^2 + x^4 + x^3 + x^5 + x^4 + x^6 + x^5 + x^7$$
$$= 1 + x^7 \text{ //}$$

* The bottom adder. ∴ output polynomial is given by

$$C^{(2)}(x) = d(x)\, g^{(2)}(x)$$
$$= (1 + x^2 + x^3 + x^4)(1 + x + x^2 + x^3)$$
$$= 1 + x + x^2 + x^3 + x^2 + x^3 + x^4 + x^3 + x^4 + x^5 + x^4 + x^5 + x^6 + x^5 + x^6 + x^7$$
$$= 1 + x + x^3 + x^4 + x^5 + x^7$$

* ∴ The final Encoder output polynomial is given by

$$C(X) = C^{(1)}(X^n) + XC^{(2)}(X^n) + X^2 C^{(3)}(X^n) + \text{-----} + X^{n-1} C^{(n)}(X^n)$$

where n = number of modulo-2 adders = 2.

$$C(X) = C^{(1)}(X^2) + X\, C^{(2)}(X^2)$$

We have $C^{(1)}(X) = 1 + X^7$.

$$C^{(1)}(X^2) = 1 + (X)^{7 \times 2} = 1 + X^{14}$$

$$C^{(2)}(X) = 1 + X + X^3 + X^4 + X^5 + X^7.$$

$$C^{(2)}(X^2) = 1 + (X)^2 + (X^3)^2 + (X^4)^2 + (X^5)^2 + (X^7)^2$$

$$C^{(2)}(X^2) = 1 + X^2 + X^6 + X^8 + X^{10} + X^{14}$$

$$\therefore C(X) = (1 + X^{14}) + X(1 + X^2 + X^6 + X^8 + X^{10} + X^{14})$$

$$= 1 + X^{14} + X + X^3 + X^7 + X^9 + X^{11} + X^{15}$$

Increasing order

$$C(X) = 1 + X + X^3 + X^7 + X^9 + X^{11} + X^{14} + X^{15}$$

* The code word corresponding to the polynomial is

$$C = [11, 01, 00, 01, 01, 01, 00, 11]$$

which is same as obtained using time domain approach

* for the Convolution Encoder shown in below fig the Information Sequence is $d = 10011$. Find the output Sequence using the following two approaches.

(a) Time-domain approach.

(b) Transform-domain approach. June/July-2012 [10M]

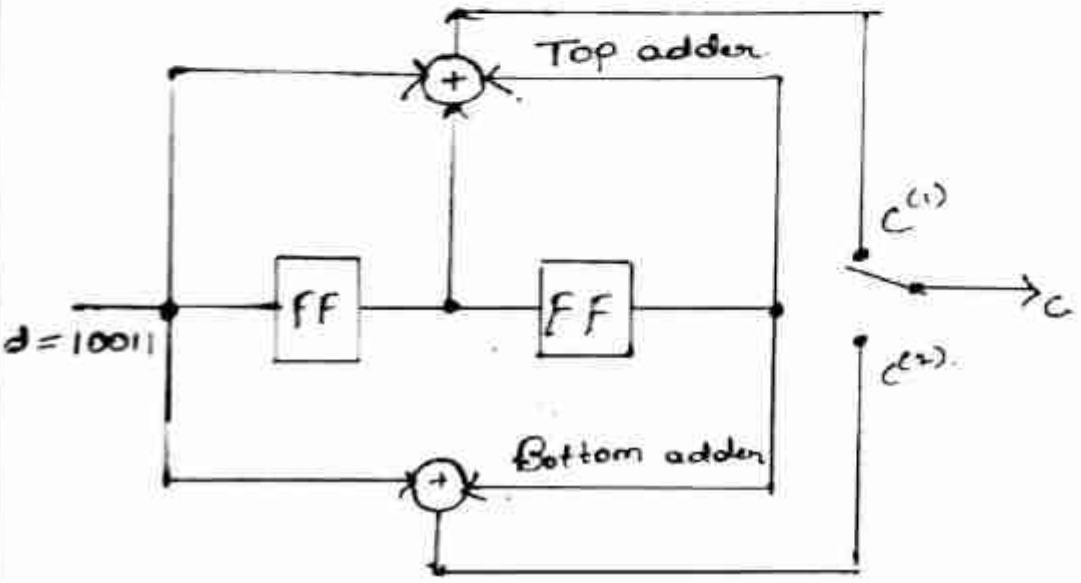


Fig: (2.1.2) Convolution Encoder.

Solution:- given $(n, k, m) = (2, 1, 2)$.

$n = 2, \quad k = 1, \quad m = 2.$

(a) Time-domain approach:

* for the Top adder. The generator Sequence is given by

$$g^{(1)} = \begin{bmatrix} g_1^{(1)} & g_2^{(1)} & g_3^{(1)} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

* for the bottom adder, The generator Sequences is given by.

$$g^{(2)} = \begin{bmatrix} g_1^{(2)} & g_2^{(2)} & g_3^{(2)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} //$$

* The generator matrix of the order $k \times [n(k+m)]$ where k = number of bits in the message sequence = 5 //

$$k \times [n(k+m)] = 5 \times [2(5+2)]$$

$$= 5 \times 14.$$

(5 → rows, 14 → columns)

$$G = \begin{bmatrix} 11 & 10 & 11 & 00 & 00 & 00 & 00 \\ 00 & 11 & 10 & 11 & 00 & 00 & 00 \\ 00 & 00 & 11 & 10 & 11 & 00 & 00 \\ 00 & 00 & 00 & 11 & 10 & 11 & 00 \\ 00 & 00 & 00 & 00 & 11 & 10 & 11 \end{bmatrix}$$

Note:
$g^{(1)} = [1\ 1\ 1]$
$g^{(2)} = [1\ 0\ 1]$
1st row: 11 10 11 00 00 00 00

The output of the Convolution Encoder is given by

$$[o] = [d][G].$$

$$= [10011] \begin{bmatrix} 11 & 10 & 11 & 00 & 00 & 00 & 00 \\ 00 & 11 & 10 & 11 & 00 & 00 & 00 \\ 00 & 00 & 11 & 10 & 11 & 00 & 00 \\ 00 & 00 & 00 & 11 & 10 & 11 & 00 \\ 00 & 00 & 00 & 00 & 11 & 10 & 11 \end{bmatrix}$$

$$= [11, 10, 11, 11, 01, 01, 11] //$$

↳ P.T.O.

b) Transform domain approach

The message $[d] = [1001]$

$d(x) = 1+x^3+x^4$ //

The generator polynomials are given by

$g^1 = [111]$

$g^1(x) = 1+x+x^2$ //

$g^{(2)} = [101]$

$g^1(x) = 1+x^2$ //

The top-adder output polynomial is given by

$$C^{(i)}(x) = d(x)\, g^{(i)}(x)$$

For The Top adder ($i=1$)

$$C^{(1)}(x) = d(x) g^{(1)}(x)$$

$$= (1+x^3+x^4)(1+x+x^2)$$

$$= 1+x+x^2+x^3+x^4+x^5+x^4+x^5+x^6$$

$$= 1+x+x^2+x^3+x^6$$ //

The bottom adder output polynomial is given by

$$C^{(2)}(x) = d(x)\, g^{(2)}(x)$$

$$= (1+x^3+x^4)(1+x^2)$$

$$= 1+x^3+x^4+x^2+x^5+x^6$$

$$= 1+x^2+x^3+x^4+x^5+x^6$$ //

* The final Encoder output polynomial is given by.

$$G(x) = c^{(1)}(x^n) + c^{(2)}(x^n) \quad \text{---} (1)$$

where n = 2 (number of modulo-2 adder)

$$G(x) = c^{(1)}(x^2) + c^{(2)}(x^2)$$

$$c^1x = 1+x+x^2+x^3+x^6.$$

$$c^1(x^2) = 1+x^2+x^4+x^6+x^{12} \quad \text{---} (2)$$

$$c^2(x) = 1+x^2+x^3+x^4+x^5+x^6.$$

$$c^2(x^2) = 1+x^4+x^6+x^8+x^{10}+x^{12} \quad \text{---} (3)$$

eq (2) and (3) Sub into eq (1)

$$G(x) = 1+x^2+x^4+x^6+x^{12}+x[1+x^4+x^6+x^8+x^{10}+x^{12}]$$

$$= 1+x^2+x^4+x^6+x^{12}+x+x^5+x^7+x^9+x^{11}+x^{13}$$

$$G(x) = 1+x+x^2+x^4+x^5+x^6+x^7+x^9+x^{11}+x^{12}+x^{13}$$

The Codeword corresponding to this polynomial is.

$$G = [11\ 10\ 11\ 11\ 01\ 01\ 11]$$

Which is same as obtained using Time-domain approach

* Consider the (3, 1, 2) Convolution Code with $g^{(1)} = (110)$ $g^{(2)} = (101)$ and $g^{(3)} = (111)$

1) Draw the Encoder block diagram
2) Find the generator matrix.
3) Find the Code word corresponding to the Information Sequence (11101) using Time domain and Transform domain approach.

June/July-2011 [10M].
June/July-2012 [10M]

Sol:-

Given $(n, k, m) = (3, 1, 2)$

n = number of modulo-2 adders (or) outputs = 3//

k = number of inputs = 1//

m = number of flipflops = 2//

NOTE:
$g^{(1)} = (110)$ Top adder
$g^{(2)} = (101)$ middle adder
$g^{(3)} = 111$ bottom adder.

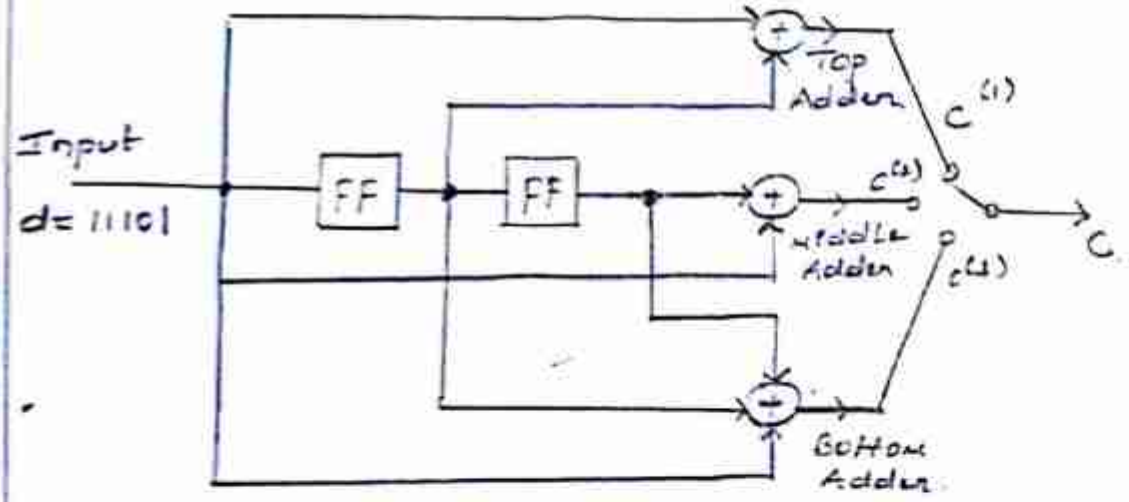


Fig: (3, 1, 2) Convolution Encoder.

Time domain approach:-

* For the Top adder, the generator Sequence is given by

$$g^{(1)} = [g_0^{(1)} \; g_1^{(1)} \; g_2^{(1)} \; \ldots] = (110)$$

* With from the middle adder the generator sequence is given by

$$g^{(2)} = [g_1^{(2)}\ g_2^{(2)}\ g_3^{(2)} \ldots] = [1\ 0\ 1]$$

* From the bottom adder the generator sequence is given by.

$$g^{(3)} = [g_1^{(3)}\ g_2^{(3)}\ g_3^{(3)}] = [1\ 1\ 1]$$

* The generator matrix is of the order $L \times (n(L+m))$ where L = number of bits in the msg. sequence = 5

$$L \times [n(L+m)]$$
$$5 \times (3(5+2))$$
$$5 \times 21 \quad \text{i.e. 5 Rows \& 21 columns}$$

NOTE:

$$G = \begin{bmatrix} g_{(1)}^{(1)} g_{(1)}^{(2)} g_{(1)}^{(3)} & g_2^{(1)} g_2^{(2)} g_2^{(3)} & \cdots & g_{m+1}^{(1)} g_{m+1}^{(2)} g_{m+1}^{(3)} & \cdots & 000 \\ 000 & g_1^{(1)} g_{(1)}^{(2)} g_{(1)}^{(3)} & g_2^{(1)} g_2^{(2)} g_2^{(3)} & \cdots & g_{m+(1)}^{(1)} g_{m+1}^{(2)} g_{m+1}^{(3)} & \cdots & 000 \\ 000 & 000 & g_1^{(1)} g_1^{(2)} g_1^{(3)} & \cdots & \vdots & \cdots & \end{bmatrix}$$

* The generator matrix "G" is given by.

$$[G] = \begin{bmatrix} 111 & 101 & 011 & 000 & 000 & 000 & 000 \\ 000 & 111 & 101 & 011 & 000 & 000 & 000 \\ 000 & 000 & 111 & 101 & 011 & 000 & 000 \\ 000 & 000 & 000 & 111 & 101 & 011 & 000 \\ 000 & 000 & 000 & 000 & 111 & 101 & 011 \end{bmatrix}$$

* The output of the Encoder is given by

$$[c] = [d]\,[G]$$

$$= [1110\,1] \begin{bmatrix} 111 & 101 & 011 & 000 & 000 & 000 & 000 \\ 000 & 111 & 101 & 011 & 000 & 000 & 000 \\ 000 & 000 & 111 & 101 & 011 & 000 & 000 \\ 000 & 000 & 000 & 111 & 101 & 011 & 000 \\ 000 & 000 & 000 & 000 & 111 & 101 & 011 \end{bmatrix}$$

$$= [111,\ 010,\ 001,\ 110,\ 100,\ 101,\ 011]$$ //.

Transform domain approach.

Let us write down the polynomials as given below

We have $[d] = [1110\,1]$

$d(x) = 1+x+x^2+x^4$ //.

$g^{(1)} = [110) \rightarrow g^{(1)}(x) = 1+x$ //

$g^{(2)} = (101) \rightarrow g^{(2)}(x) = 1+x^2$ //

$g^{(3)} = (111) \rightarrow g^{(3)}(x) = 1+x+x^2$ //

The output polynomials for the three adders are given by.

$$c^{(i)}(x) = d(x)\, g^{(i)}(x) \text{ for } i = 1, 2, 3$$

For the Top adder. ($i = 1$).

$$c^{(1)}(x) = d(x)\, g^{(1)}(x)$$

$$= (1+x+x^2+x^4)\,(1+x)$$

$$= 1+x+x^2+x^4+x+x^2+x^3+x^5$$

$$= 1+x^3+x^4+x^5$$

for the middle adder ($l=2$).

$$c^{(2)}(x) = d(x)\, g^{(2)}(x)$$
$$= (1+x+x^2+x^4)(1+x^2)$$
$$= 1+x+x^2+x^4+x^2+x^3+x^4+x^6$$
$$= 1+x+x^3+x^6 \text{ //}$$

for the bottom adder ($l=3$)

$$c^{(3)}(x) = d(x)\, g^{(3)}(x)$$
$$= (1+x+x^2+x^4)(1+x+x^2)$$
$$= 1+x+x^2+x^4+x+x^2+x^3+x^5+x^2+x^3+x^6$$
$$= 1+x^2+x^5+x^6 \text{ //}$$

The output of the encoder is given by.

$$C(x) = c^{(1)}(x^n) + x\, c^{(2)}(x^n) + x^2 c^{(3)}(x^n)$$

$n=3$

$$C(x) = c^{(1)}x^3 + x\, c^{(2)}(x^3) + x^2 c^{(3)}(x^3)$$

$c^{(1)}x = 1+x^3+x^4+x^5$ $\quad c^{(2)}x = 1+x+x^3+x^6$

$c^{(1)}x^3 = 1+x^9+x^{12}+x^{15}$ // $\quad \therefore c^{(2)}x^3 = 1+x^3+x^9+x^{18}$ //.

$c^{(3)}x = 1+x^2+x^5+x^6$

$c^{(3)}x^3 = 1+x^6+x^{15}+x^{18}$ //

$$C(x) = 1+x^9+x^{12}+x^{15} + x(1+x^3+x^9+x^{18}) + x^2(1+x^6+x^{15}+x^{18})$$
$$= 1+x^9+x^{12}+x^{15}+x+x^4+x^{10}+x^{19}+x^2+x^8+x^{17}+x^{20}$$
$$= 1+x+x^2+x^4+x^8+x^9+x^{10}+x^{12}+x^{15}+x^{17}+x^{19}+x^{20}$$

The output of the encoder is

$$C = [111, 010, 001, 110, 100, 101, 011]$$

* For the Convolution Encoder shown in below fig
1) find the impulse response and hence calculate the output produced by the Information Sequence 10111
2) Write the generation polynomials of the Encoder and recompute the output for the input (1) and compare with that of (1).

DEC-2012 [8M]

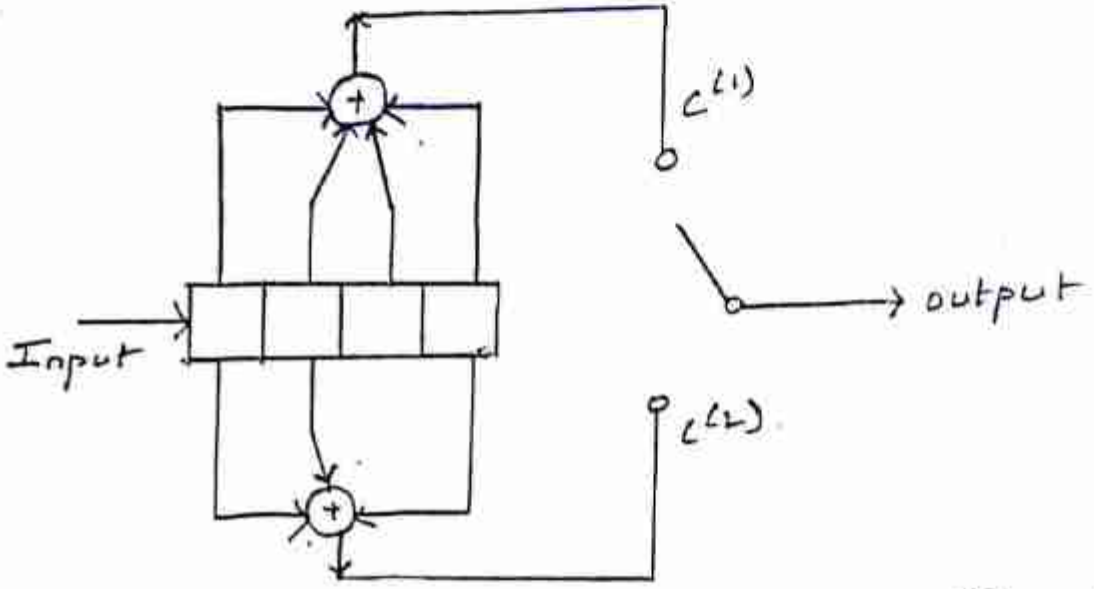


Fig: a (2, 1, 4) Convolutional Encoder.

Sol :- ∴ given $(n, K, M) = (2, 1, 4)$

$n = 2 \quad K = 1 \quad m = 4$ //

∴ number rows in $[G] = K = 5$ //

∴ number of Columns $[G] = n(K+M)$
$= 2(5+4) = 18$ //

NOTE
Inf^n sequen.
10111
$= 5$
K //

(5×18) matrix
↳ Columns //
Rows

Time domain approach :-

* For the Top adder, the generator sequences are given by

$$g^{(1)} = \begin{bmatrix} g_1^{(1)} & g_2^{(1)} & g_3^{(1)} & g_4^{(1)} & g_5^{(1)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

* For the bottom adder, the generator sequence is given by

$$g^{(2)} = \begin{bmatrix} g_1^{(2)} & g_2^{(2)} & g_3^{(2)} & g_4^{(2)} & g_5^{(2)} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

The generator polynomial is given by.

$$[G] = \begin{bmatrix} 00 & 11 & 11 & 10 & 11 & 00 & 00 & 00 & 00 \\ 00 & 00 & 11 & 11 & 10 & 11 & 00 & 00 & 00 \\ 00 & 00 & 00 & 11 & 11 & 10 & 11 & 00 & 00 \\ 00 & 00 & 00 & 00 & 11 & 11 & 10 & 11 & 00 \\ 00 & 00 & 00 & 00 & 00 & 11 & 11 & 10 & 11 \end{bmatrix}$$

The encoder output is given by $[c] = [d][G]$

$$[c] = [1\ 0\ 1\ 1\ 1] \begin{bmatrix} 00 & 11 & 11 & 10 & 11 & 00 & 00 & 00 & 00 \\ 00 & 00 & 11 & 11 & 10 & 11 & 00 & 00 & 00 \\ 00 & 00 & 00 & 11 & 11 & 10 & 11 & 00 & 00 \\ 00 & 00 & 00 & 00 & 11 & 11 & 10 & 11 & 00 \\ 00 & 00 & 00 & 00 & 00 & 11 & 11 & 10 & 11 \end{bmatrix}$$

$$= [00\ 11\ 11\ 01\ 11\ 10\ 10\ 01\ 11] \ //.$$

P.T.O.

Transform domain approach :-

The message. $[d] = [10111]$

$d(x) = 1+x^2+x^3+x^4$ //.

$g^{(1)} = [01111] \Rightarrow g^{(1)}(x) = x+x^2+x^3+x^4$ //

$g^{(2)} = [01101] \Rightarrow g^{(2)}(x) = x+x^2+x^4$ //.

The output polynomial for the top adder is.

$$c^{(1)}(x) = d(x)\,g^{(1)}(x).$$
$$= (1+x^2+x^3+x^4)(x+x^2+x^3+x^4)$$
$$c^{(1)}(x) = x+x^2+x^3+x^4+x^3+x^4+x^5+x^6+x^4+x^5+x^6+x^7+x^5+x^6+x^7+x^8.$$
$$= x+x^2+x^4+x^5+x^6+x^8$$ //

The output polynomial for the bottom adder is

$$c^{(2)}(x) = d(x)\,g^{(2)}(x)$$
$$= (1+x^2+x^3+x^4)\cdot(x+x^2+x^4)$$
$$= x+x^2+x^4+x^3+x^4+x^6+x^4+x^5+x^7+x^5+x^6+x^8$$
$$= x+x^2+x^3+x^4+x^7+x^8$$ //

The output of the Encoder is given by

$$C(x) = C^{(1)}(x^2) + x\,C^{(2)}(x^2)$$

$$c'(x) = x+x^2+x^4+x^5+x^6+x^8$$

$$\therefore c'(x^2) = x^2+x^4+x^8+x^{10}+x^{12}+x^{16}$$ //.

$c^{(2)}(x) = x + x^2 + x^3 + x^4 + x^7 + x^8$

$c^{(2)}(x^2) = x^2 + x^4 + x^6 + x^8 + x^{14} + x^{16}$ //

$C(x) = c^{(1)}(x^2) + x\ c^{(2)}(x^2)$

$= x^2 + x^4 + x^8 + x^{10} + x^{12} + x^{16} + x(x^2 + x^4 + x^6 + x^8 + x^{14} + x^{16})$

$= x^2 + x^3 + x^4 + x^5 + x^7 + x^8 + x^9 + x^{10} + x^{12} + x^{15} + x^{16} + x^{17}$

$C = [00, 11, 11, 01, 11, 10, 10, 01, 11]$ //

* For the (3, 2, 1) Convolution Encoder shown in below figure find Codeword "C" for Input Sequence of $d^{(1)} = 101$ and $d^{(2)} = 110$ using

a) Time-domain approach (using generator matrix)

b) Transform-domain approach by constructing Transfer function matrix. DEC-2016/JAN-2017 [10M]

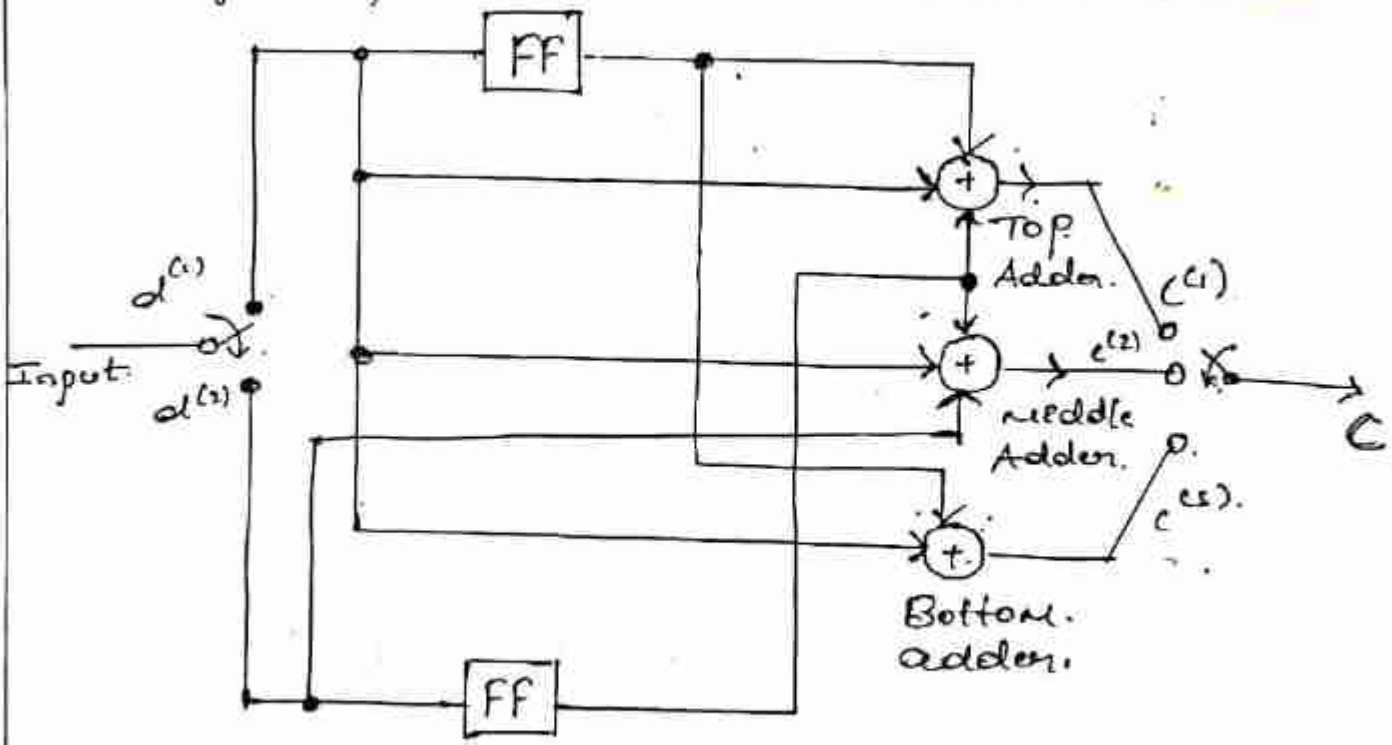


Fig:- a(3, 2, 1) Convolution Encoder.

Time domain approach:-

Sol :- Given $(n, k, m) = (3, 2, 1)$

$n=3 \quad k=2 \quad m=1$ //

The generator sequences for the encoder as shown in the above figure.

$g_1^{(1)} = [g_{11}^{(1)} \; g_{12}^{(1)}] = [1, 1]$ //

$g_1^{(2)} = [g_{11}^{(2)} \; g_{12}^{(2)}] = [1, 0]$ //

$g_1^{(3)} = [g_{11}^{(3)} \; g_{12}^{(3)}] = [11]$ //

$g_2^{(1)} = [g_{21}^{(1)} \; g_{22}^{(2)}] = [01]$ //

$g_2^{(3)} = [g_{21}^{(3)} \; g_{22}^{(3)}] = [0, 0]$ //

$g_2^{(2)} = [g_{21}^{(2)} \; g_{22}^{(2)}] = [1, 1]$ //

ROOTS $g^{(1)} = 101$, $g^{(2)} = 101$ 2D //

The generator matrix will now have $L+L = 2L$

$= 2 \times 3$

$= 6$ //

number of Rows $= 6$ //

number of Columns in $[G] = n(L+m)$

$= 2(5+1) = 18$ //

The generator matrix is given by.

$$[G] = \begin{bmatrix} 111 & 101 & 000 & 000 \\ 010 & 110 & 000 & 000 \\ 000 & 111 & 101 & 000 \\ 000 & 010 & 110 & 000 \\ 000 & 000 & 111 & 101 \\ 000 & 000 & 010 & 110 \end{bmatrix}$$ //

ROOTS

$g_1^{(1)} =$

$g_1^{(2)} =$

$g_1^{(3)} =$ 101 111

$g_2^{(1)} = 01$

$g_2^{(3)} = 11$

$g_2^{(2)} = 10$

$g_2 =$ 010 110

The output of the given Convolutional Encoder is given by

$$[c] = [d][G]$$

$$d^{(1)} = [101] \qquad d^{(2)} = [110]$$

$$d^{(1)} = \begin{bmatrix} d_1^{(1)} & d_2^{(1)} & d_3^{(1)} \end{bmatrix} = [101] //$$

$$d^{(2)} = \begin{bmatrix} d_1^{(2)} & d_2^{(2)} & d_3^{(2)} \end{bmatrix} = [110] //$$

where, $[d] = \begin{bmatrix} d_1^{(1)} d_1^{(2)} & d_2^{(1)} d_2^{(2)} & d_3^{(1)} d_3^{(2)} \end{bmatrix}$

$$= [11 \quad 01 \quad 10] //$$

$$[c] = [d]\,[G]$$

$$= [11\,01\,10] \begin{bmatrix} 111 & 101 & 000 & 000 \\ 010 & 110 & 000 & 000 \\ 000 & 111 & 101 & 000 \\ 000 & 010 & 110 & 000 \\ 000 & 000 & 111 & 101 \\ 000 & 000 & 010 & 110 \end{bmatrix}$$

$$= [101, 001, 001, 101] //$$

P.T.O.

Transform domain :-

$g_1^{(1)} = [1\ 1] \rightarrow g_1^{(1)}(x) = 1+x$ //

$g_1^{(2)} = [1\ 0] \rightarrow g_1^{(2)}(x) = 1$ //

$g_1^{(3)} = [1\ 1] \rightarrow g_1^{(3)}(x) = 1+x$ //

$g_2^{(1)} = [0\ 1] \rightarrow g_2^{(1)}(x) = x$ //

$g_2^{(2)} = [1\ 1] \rightarrow g_2^{(2)}(x) = 1+x$ //

$g_2^{(3)} = [0\ 0] \rightarrow g_2^{(3)}(x) = 0$ //

The generator polynomial matrix also called "TRANSFER FUNCTION MATRIX" is given by

$$G(x) = \begin{bmatrix} g_1^{(1)}(x) & g_1^{(2)}(x) & g_1^{(3)}(x) \\ g_2^{(1)}(x) & g_2^{(2)}(x) & g_2^{(3)}(x) \end{bmatrix}$$

$$G[x] = \begin{bmatrix} 1+x & 1 & 1+x \\ x & 1+x & 0 \end{bmatrix}$$

The Information Sequence (or) message polynomial are given by

$d^{(1)} = [1\ 0\ 1] \rightarrow d^{(1)}(x) = 1+x^2$ //

$d^{(2)} = [1\ 1\ 0] \rightarrow d^{(2)}(x) = 1+x$ //

$\therefore d(x) = [d^{(1)}(x),\ d^{(2)}(x)] = [(1+x^2),\ (1+x)]$

The Code polynomial at the output of the adder are then given by

$$C_1(x) = [c^{(1)}(x) \quad c^{(2)}(x) \quad c^{(3)}(x)]$$

$$C(x) = d(x)\, G_1[x]$$

NOTE: $c^{(1)}(x)\, c^{(2)}(x) \cdots c^{(n)}(x) = d(x)\, G_1(x)$

$$= [(1+x^2)\ (1+x)] \begin{bmatrix} 1+x & 1 & 1+x \\ x & 1+x & 0 \end{bmatrix}$$

$$= [(1+x^2)(1+x)+(1+x)(x) \ , \ (1+x^2)(1)+(1+x)(1+x) \ , \ (1+x)(1+x^2)+(0)(1+x)]$$

$$= [1+x+x^2+x^3+x+x^2 \ , \ 1+x^2+1+x+x+x^2 \ , \ 1+x^2+x+x^3]$$

$$C[x] = [1+x^3 \ , \ 0 \ , \ 1+x^2+x+x^3] \ //$$

or $C[x] = [c^{(1)}(x) \ , \ c^{(2)}(x) \ , \ c^{(3)}(x)]$

where $c^{(1)}(x) = 1+x^3$ // $c^{(2)}(x) = 0$ // $c^{(3)}(x) = 1+x^2+x+x^3$ //

The output code word is given by

$$C(x) = [c^{(1)}(x^n) + x\, c^{(2)}(x^n) + x^2\, c^{(3)}(x^n)]$$

Let $n = 3$ (o/p's)

$$C(x) = [c^{(1)}(x^3) + x\, c^{(2)}(x^3) + x^2 c^{(3)}(x^3)]$$

$$C(x) = \left[c^{(1)}(x^3) + x\, c^{(2)}(x^3) + x^2 c^{(3)}(x^3) \right]$$

$c^1(x) = 1+x^3$ $\quad c^{(2)}(x) = 0$ $\quad c^{(3)}(x) = 1+x^2+x^4+x^7$

$\therefore c^1(x^3) = 1+x^9$ // $\quad \therefore c^{(2)}(x^3) = 0$ // $\quad c^{(3)}(x) = 1+x^6+x^{12}+x^{21}$ //

$$C(x) = 1+x^9 + x[0] + x^2\left[1+x^6+x^{12}+x^{21}\right]$$

$$= 1+x^9 + x^2+x^8+x^{14}+x^{23}$$

$$= 1+x^2+x^8+x^9+x^{14}+x^{23}$$

The Code word at the output of the Encoder is

$C = [101\ 001\ 001\ 101]$ // which is same as before //

STATE TABLE, STATE TRANSITION TABLE, STATE DIAGRAM, CODE TREE :-

1) Consider the binary Convolutional Encoder shown in fig (below). Draw the State Table, State Transition Table, State diagram and the corresponding Code Tree, using the Code Tree, find the Encoded sequence for the message (10111), verify the output sequence so obtained using Transform domain approach. DEC-2010 [10M]

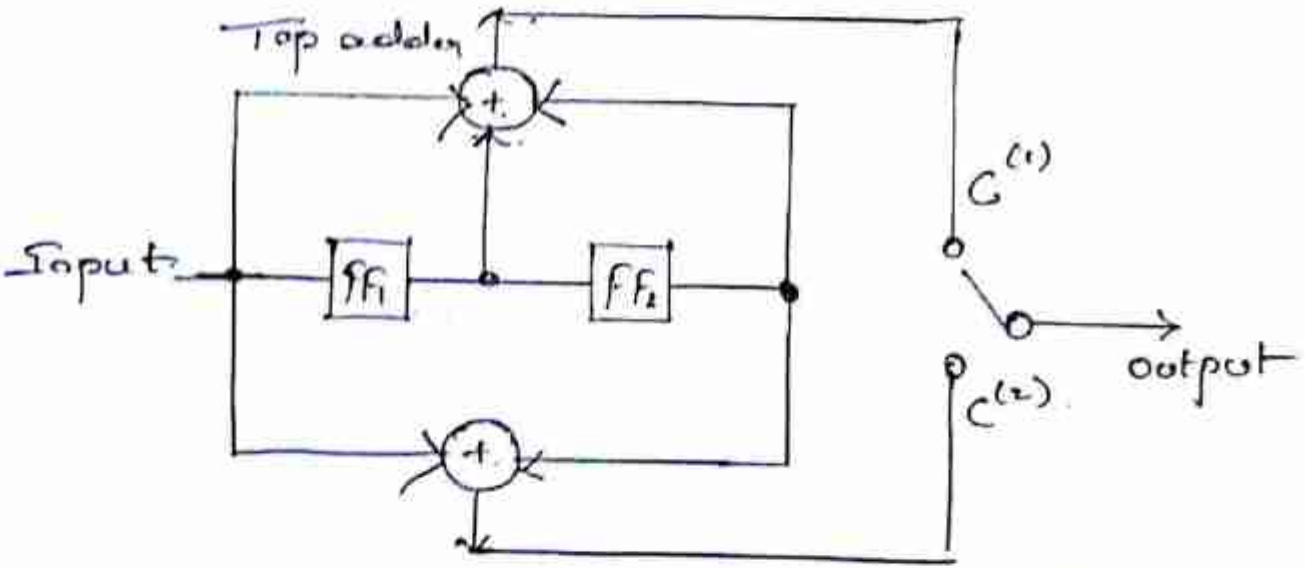


Fig :- a (2, 1, 2) Convolutional Encoder.

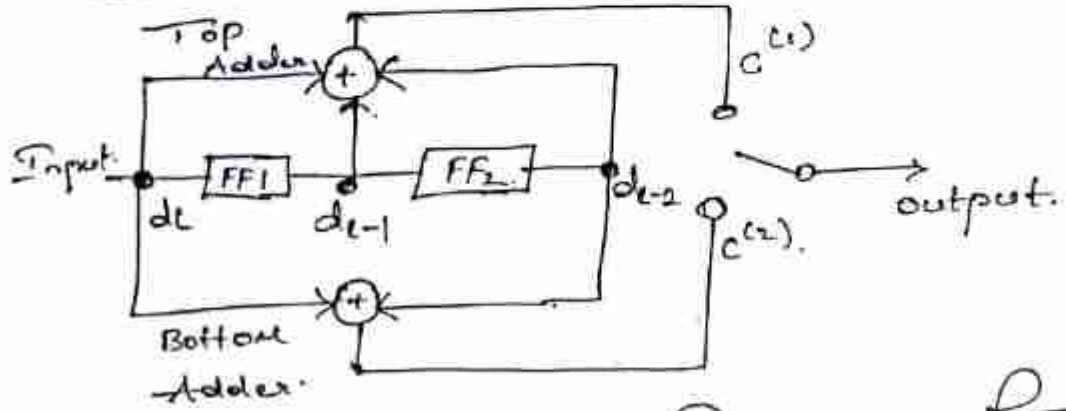
Sol :-

State Table :-

There are 2-flipflops in the Shift register hence There are $2^2 = 4$ States. Let the States be S_0, S_1, S_2, and S_3 as represented as shown in before Table

State.	S_0	S_1	S_2	S_3
Binary description	00	10	01	11

State Transition Table :-



From the above figure. The outputs of the two modulo-2 adder. (Top and bottom adder) $C^{(1)}$ and $C^{(2)}$ Can be represented as o/p Sequence given by.

$C^{(1)} = d_k + d_{k-2} + d_{k-2}$ for the Top adder

$C^{(2)} = d_k + d_{k-2}$ for the bottom adder.

present State.	Binary description	Input	Next State.	Binary description.	d_k	d_{k-1}	d_{k-2}	Output $C^{(1)}$	$C^{(2)}$
S_0	00	0	S_0	00	0	0	0	0	0
		1	S_1	10	1	0	0	1	1
S_1	10	0	S_2	01	0	1	0	1	0
		1	S_3	11	1	1	0	0	1
S_2	01	0	S_0	00	0	0	1	1	1
		1	S_1	10	1	0	1	0	0
S_3	11	0	S_2	01	0	1	1	0	1
		1	S_3	11	1	1	1	1	0

Table :- State Transition Table.

State diagram:-

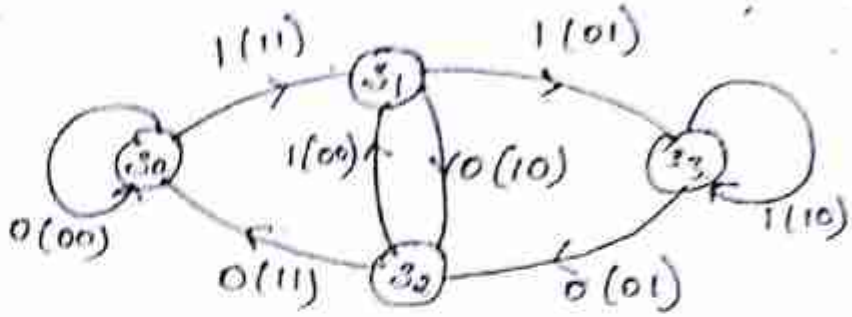


Fig: State diagram.

Code Tree:-

The State diagram can be drawn as a Code Tree.

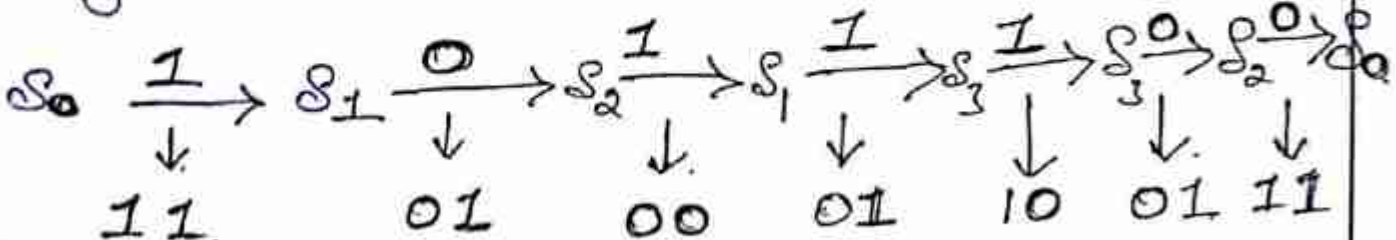
Rules:-

1) Start with State S_0.

2) If input = 0 move up the Tree.
If input = 1 move down the Tree.
The vertical line "→" move to node.
The horizontal line "↓" to Branch.

3) For every Transition, Corresponding to every bit, the States are represented as vertical lines (nodes) and the Corresponding bit are written on horizontal lines (Branches).

So given [D] = 10111.



Code-Length is $n(L+M)$

$2(5+2)$

$= 14$ bits //

note: $D = 10111$, $L = 5$ //

We will Augment 2 additional zero input bits, in order to get the Complete Code, so as to Clear the Shift Register

$\therefore [D] = [11\ 10\ 00\ 01\ 10\ 01\ 11]$

If we Decoder it:

note: see state diagram

$S_0 \xrightarrow{11} S_1 \xrightarrow{10} S_2 \xrightarrow{00} S_1 \xrightarrow{01} S_1 \xrightarrow{10} S_2 \xrightarrow{01} S_2 \xrightarrow{11} S_0$

1 0 1 1 1 0 0

discard last "m" bits.

Decoded bit is $= [10111]$ //

Decoding operation is done by pairing the Encoded Convolution o/p into groups of $n = 2$ bits //

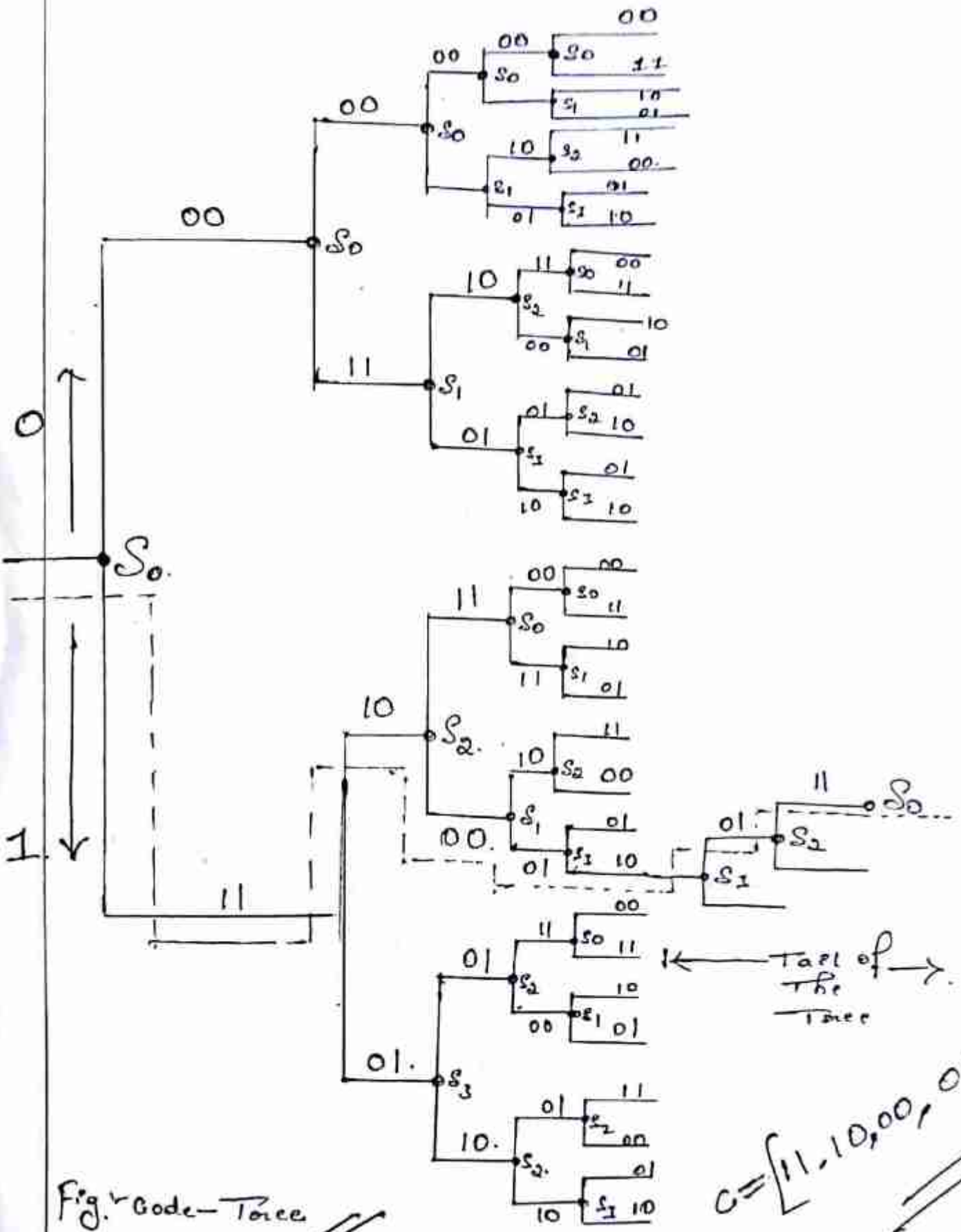


Fig:- Code-Tree

Verification Through Transform Domain Approach

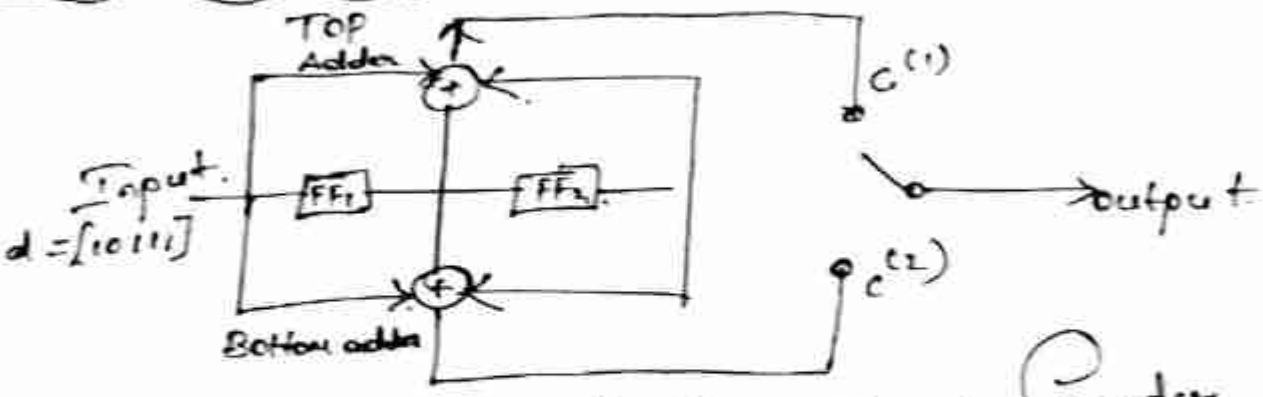


Fig: (2, 1, 2) Convolutional Encoder.

Soln: $g^{(1)} = [1\ 1\ 1]$ $\quad g^{(1)}(x) = 1+x+x^2$

$g^{(2)} = [1\ 0\ 1]$ $\quad g^{(2)}(x) = 1+x^2$

$d = [1011]$ $\quad d(x) = 1+x^2+x^3+x^4$

For the Top adder:

$$c^{(1)}(x) = d(x)g^{(1)}(x)$$
$$= (1+x^2+x^3+x^4)(1+x+x^2)$$
$$= 1+x+x^2+x^3+x^3+x^4+x^4+x^5+x^4+x^5+x^6$$
$$= 1+x+x^4+x^6$$

For the bottom adder:

$$c^{(2)}(x) = d(x)g^{(2)}(x)$$
$$= (1+x^2+x^3+x^4)(1+x^2)$$
$$= 1+x^2+x^3+x^4+x^2+x^4+x^5+x^6$$
$$= 1+x^3+x^5+x^6$$

P.T.O.

The output of the encoder is given by.

$$C(x) = C^{(1)}(x^n) + xC^{(2)}(x^n)$$

$n = 2$.

$$C(x) = C^{(1)}(x^2) + x\,C^{(2)}(x^2)$$

$C^1(x) = 1+x+x^4+x^6$ $\quad C^{(2)}(x) = 1+x^3+x^5+x^6$

$\therefore C^1(x^2) = 1+x^2+x^8+x^{12}$ // $\quad C^{(2)}(x^2) = 1+x^6+x^{10}+x^{12}$ //

$$C(x) = C^{(1)}x^2 + xC^{(2)}(x^2)$$

$$= 1+x^2+x^8+x^{12} + x\left[1+x^6+x^{10}+x^{12}\right]$$

$$= 1+x^2+x^8+x^{12}+x+x^7+x^{11}+x^{13}$$

$$= 1+x+x^2+x^7+x^8+x^{11}+x^{12}+x^{13}$$ //

$\therefore C = [11, 10, 00, 01, 10, 01, 11]$

↑ Which is same as before //

* Consider the Convolutional Encoder shown in below. fig. The Code is Systematic.

1) Draw the State diagram.
2) Draw the Code tree.
3) find the Encoder output produced by the msg Sequence 10 111
4) verify the output using time-domain approach.

JUNE-2008 [8M]
DEC-JAN 2007 [12M]

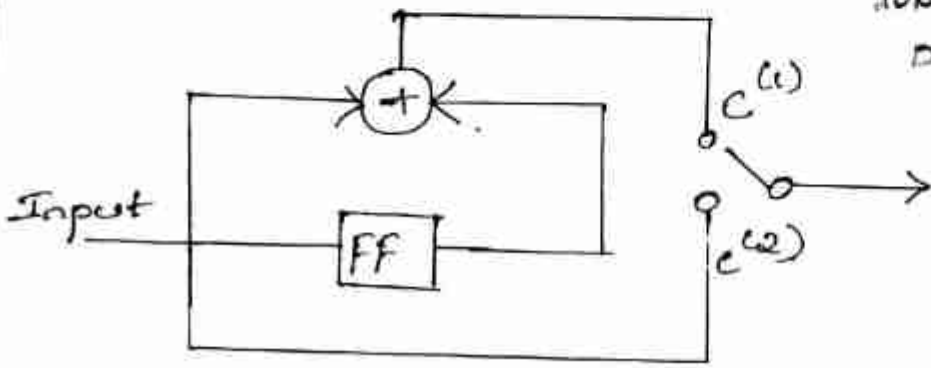


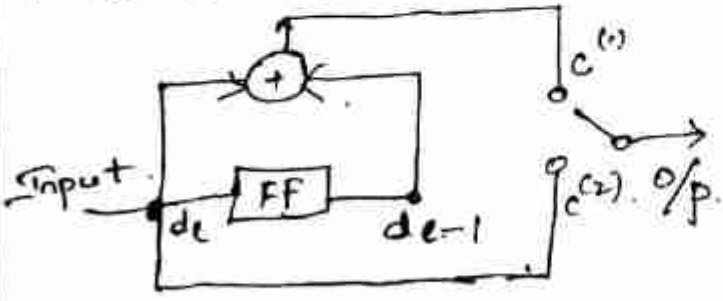
Fig:- A (2, 1, 1) Convolutional Encoder.

Sol:- State Table:-

Since there is only one flip-flop, we have only 2 States. Let the two States be represented by $S_0 \rightarrow 0$ & $S_1 \rightarrow 1$

State.	S_0	S_1
Binary Description	0	1.

State Transition Table:-



The output of the Encoder are given by

$c^{(1)} = d_k + d_{k-1}$

and $c^{(2)} = d_k$ //

present State	Binary Description	Input	Next State	Binary Description	d_i	d_{i-1}	output $c^{(1)}$	$c^{(2)}$
S_0	0	0	S_0	0	0	0	0	0
		1	S_1	1	1	0	1	1
S_1	1	0	S_0	0	0	1	1	0
		1	S_1	1	1	1	0	1

Table: State Transition Table

State diagram:-

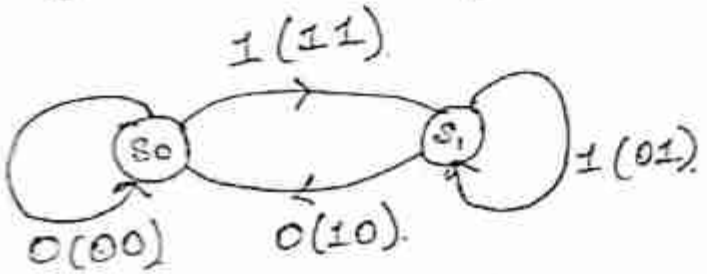
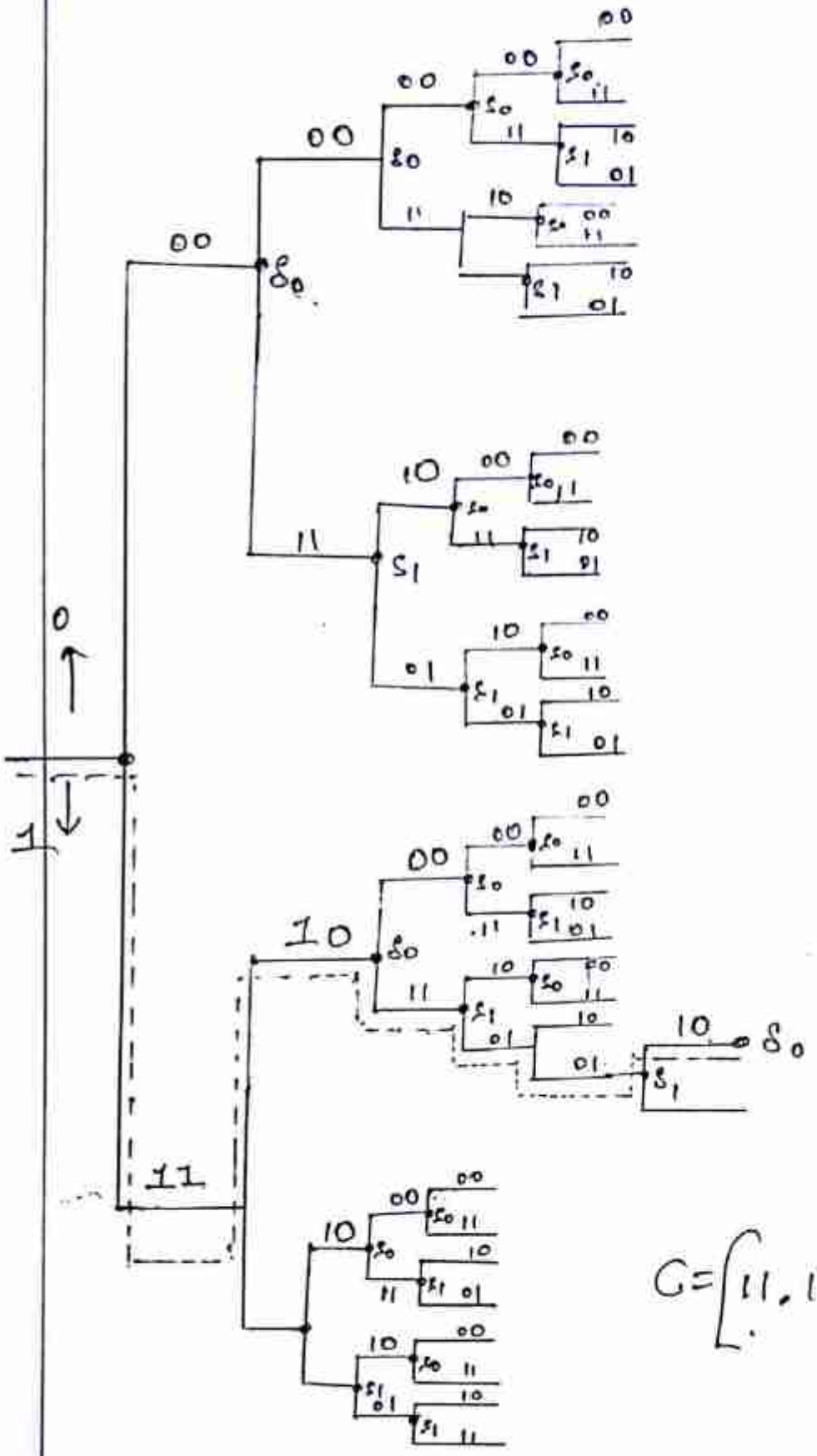


Fig:- State diagram.

Given. $[D] = [1011]$

P.T.O.



$C = [11, 10, 11, 01, 01, 10]$

Fig:- Code Tree.

Verification :-

$g^{(1)} = [1\ 1]$ $g^{(2)} = [1\ 0]$

$n(L+N) = 2(5+1) = 12$ Columns //.

$L = 5$ - Rows //.

$$G = \begin{bmatrix} 11 & 10 & 00 & 00 & 00 & 00 \\ 00 & 11 & 10 & 00 & 00 & 00 \\ 00 & 00 & 11 & 10 & 00 & 00 \\ 00 & 00 & 00 & 11 & 10 & 00 \\ 00 & 00 & 00 & 00 & 11 & 10 \end{bmatrix}$$ //.

$C = d\,[G]$.

$$= [1\ 0\ 1\ 1\ 1] \begin{bmatrix} 11 & 10 & 00 & 00 & 00 & 00 \\ 00 & 11 & 10 & 00 & 00 & 00 \\ 00 & 00 & 11 & 10 & 00 & 00 \\ 00 & 00 & 00 & 11 & 10 & 00 \\ 00 & 00 & 00 & 00 & 11 & 10 \end{bmatrix}$$

$= [11, 10, 11, 01, 01, 10]$ which is same as before //.

* Figure below. Shows a (2, 1, 3) Convolutional Encoder
1) Draw the State diagram
2) Draw the Code Tree.
3) Find the Encoder output produced by the message Sequence 11101 by traversing through the code tree

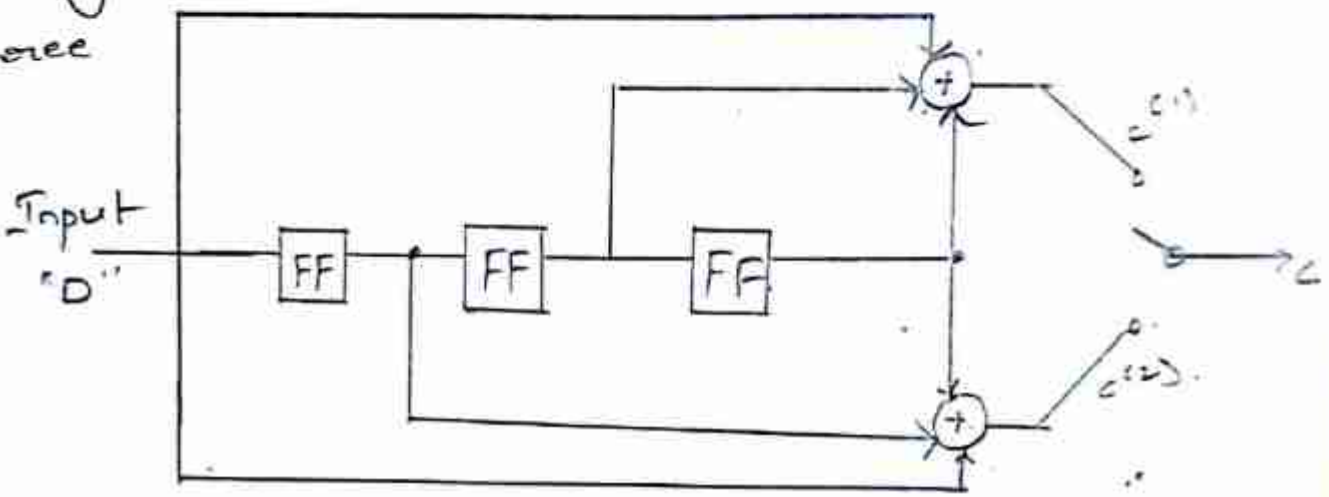


Fig:- a (2,1,3) Convolutional Encoder.

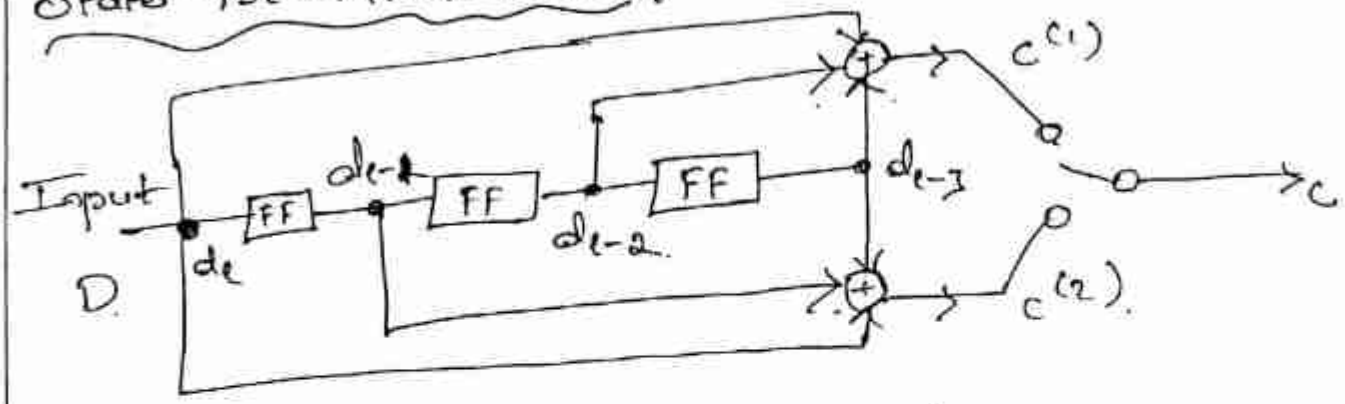
June/July-2014 [14M]

Sol :- State Table :-

There are 3 flip-flops in the Shift Register and hence there are $2^3 = 8$ States. Let the States S_0 to S_7 be represented as shown in below table

State.	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Binary Description	000	100	010	110	001	101	011	111

State Transition Table :-



$$c^{(1)} = [1011] = d_k + d_{k-2} + d_{k-3} //$$

$$c^{(2)} = [1101] = d_k + d_{k-1} + d_{k-3} //$$

present State	Binary description	Input	Next state	Binary description	d_k	d_{k-1}	d_{k-2}	d_{k-3}	output $c^{(1)}$	output $c^{(2)}$
S_0	000	0	S_0	000	0	0	0	0	0	0
		1	S_1	100	1	0	0	0	1	1
S_1	100	0	S_2	010	0	1	0	0	0	1
		1	S_3	110	1	1	0	0	1	0
S_2	010	0	S_4	001	0	0	1	0	1	0
		1	S_5	101	1	0	1	0	0	1
S_3	110	0	S_6	011	0	1	1	0	1	1
		1	S_7	111	1	1	1	0	0	0
S_4	001	0	S_0	000	0	0	0	1	1	1
		1	S_1	100	1	0	0	1	0	0
S_5	101	0	S_2	010	0	1	0	1	1	0
		1	S_3	110	1	1	0	1	0	1
S_6	011	0	S_4	001	0	0	1	1	0	1
		1	S_5	101	1	0	1	1	1	0
S_7	111	0	S_6	011	0	1	1	1	0	0
		1	S_7	111	1	1	1	1	1	1

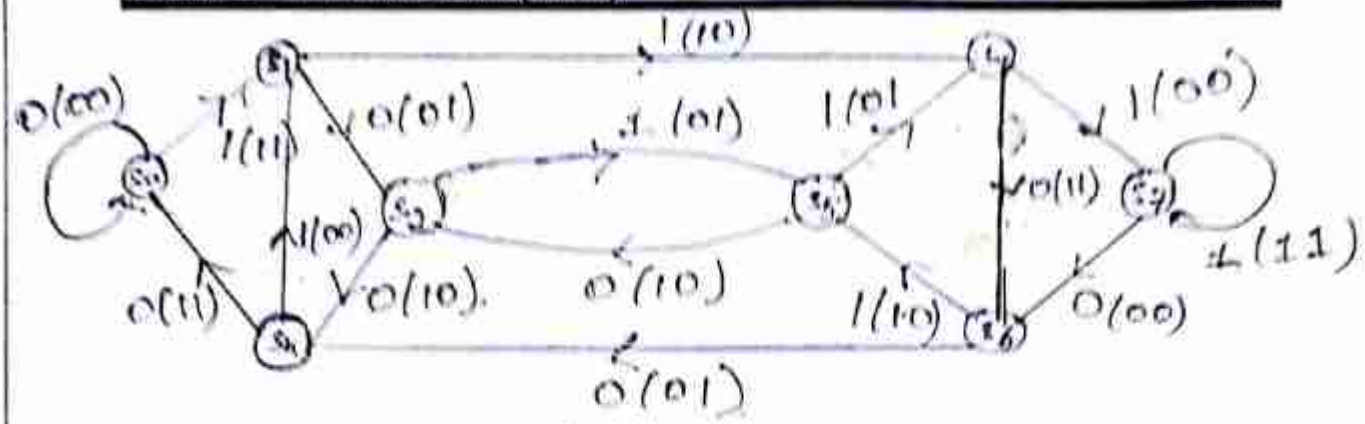


Fig: State diagram.

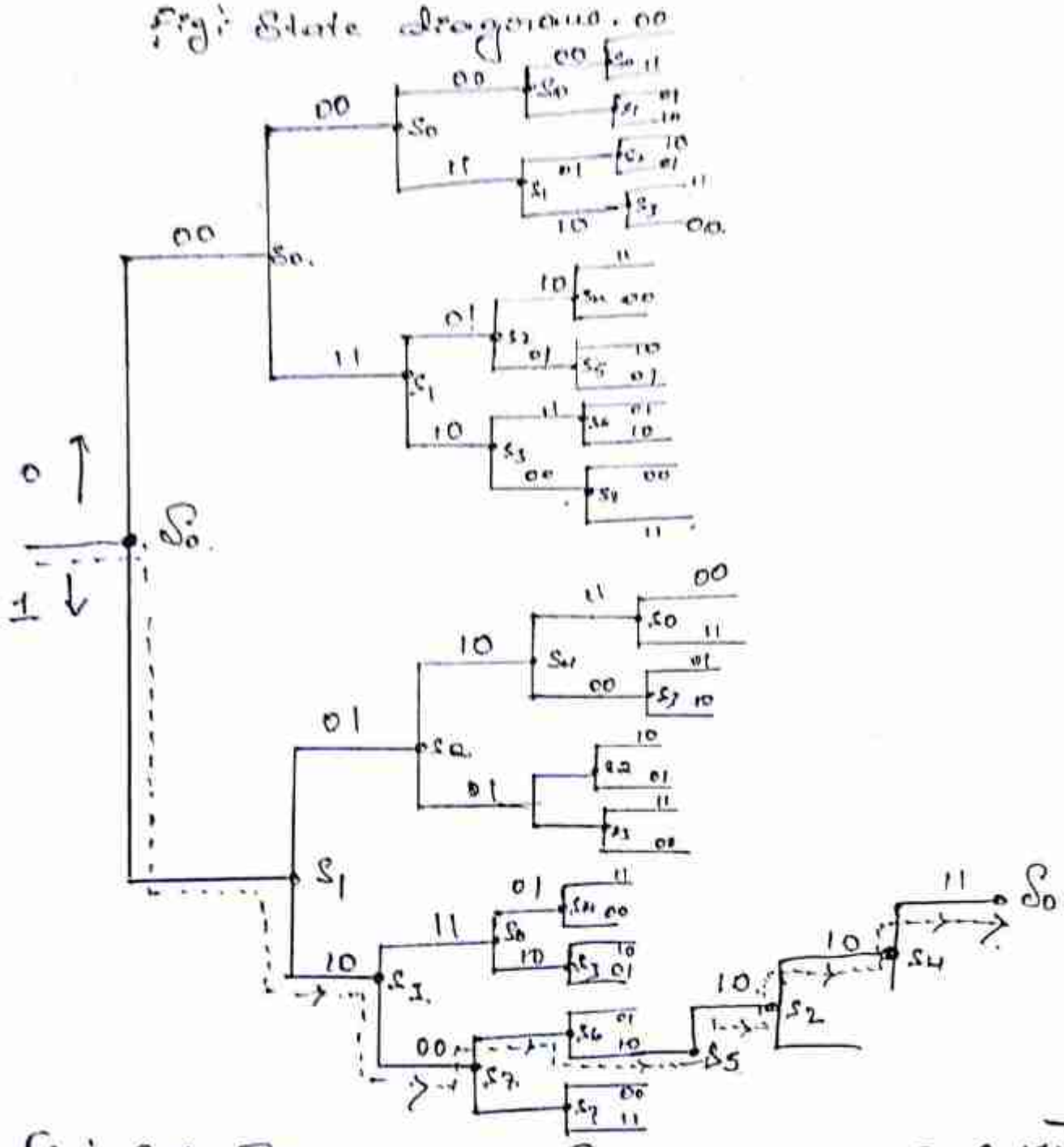


Fig: Code-Tree. $G = \{11, 10, 00, 00, 10, 10, 11\}$

Verification:- $g^{(1)} = 1011$ $\quad g^{(2)} = [1\ 1\ 0\ 1]$

$L \times [n(L+M)]$ $\quad (n, k, M)$

$5 \times [2(5+3)]$ $\quad d = \underbrace{1110}_{L=5}1$

5×16 //

$$G = \begin{bmatrix} 11\ 01\ 10\ 11\ 00\ 00\ 00\ 00 \\ 00\ 11\ 01\ 10\ 11\ 00\ 00\ 00 \\ 00\ 00\ 11\ 01\ 10\ 11\ 00\ 00 \\ 00\ 00\ 00\ 11\ 01\ 10\ 11\ 00 \\ 00\ 00\ 00\ 00\ 11\ 01\ 10\ 11 \end{bmatrix}$$

$$[C] = [d].[G] = [1\ 1\ 1\ 0\ 1] \begin{bmatrix} 11\ 01\ 10\ 11\ 00\ 00\ 00\ 00 \\ 00\ 11\ 01\ 10\ 11\ 00\ 00\ 00 \\ 00\ 00\ 11\ 01\ 10\ 11\ 00\ 00 \\ 00\ 00\ 00\ 11\ 01\ 10\ 11\ 00 \\ 00\ 00\ 00\ 00\ 11\ 01\ 10\ 11 \end{bmatrix}$$

$$[C] = [11, 10, 00, 00, 10, 10, 10, 11]$$

Which is same as before //

* Consider a (3, 1, 2) Convolutional Encoder with $g^{(1)} = 110$ $g^{(2)} = 101$ and $g^{(3)} = 111$

1) Draw the Encoder block diagram.
2) Draw State table
3) Draw State transition table.
4) Draw State diagram.
5) Find the Encoder output by traversing through the State diagram for input message Sequence of (11101)
6) Draw Code Trellis and obtain the output of the Encoder for the same input Sequence of (11101)

DEC 2006 - 20 M]

Sol :-

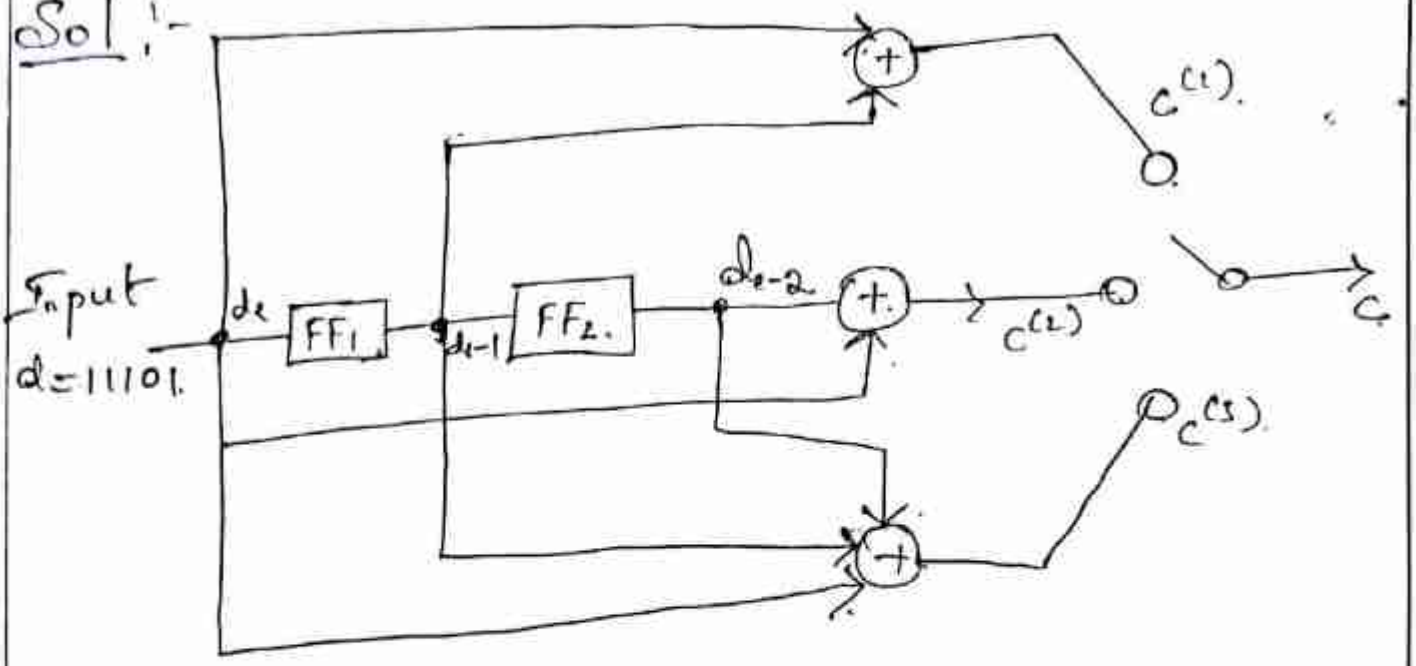


Fig: Convolutional Encoder.

State table :- There are 2 flip-flops and hence there are $2^2 = 4$ States. Let these States be represented by S_0, S_1, S_2 and S_3.

State	S_0	S_1	S_2	S_3
Binary Description	00	10	01	11

State Transition Table :-

From the above fig. The outputs of the three modulo-2 adders "$C^{(1)}$", $C^{(2)}$, & $C^{(3)}$ can be represented as. output sequences given by.

$$C^{(1)} = d_t + d_{t-1} \qquad C^{(2)} = d_t + d_{t-2}.$$

$$C^{(3)} = d_t + d_{t-1} + d_{t-2}.$$

present State	Binary Description	Input	next State	Binary Description	d_t	d_{t-1}	d_{t-2}	Output $C^{(1)}$	$C^{(2)}$	$C^{(3)}$
S_0	00	0	S_0	00	0	0	0	0	0	0
		1	S_1	10	1	0	0	1	1	1
S_1	10	0	S_2	01	0	1	0	1	0	1
		1	S_3	11	1	1	0	0	1	0
S_2	01	0	S_0	00	0	0	1	0	1	1
		1	S_1	10	1	0	1	1	0	0
S_3	11	0	S_2	01	0	1	1	1	1	0
		1	S_3	11	1	1	1	0	0	1

Table :- State Transition Table //.

State diagram:-

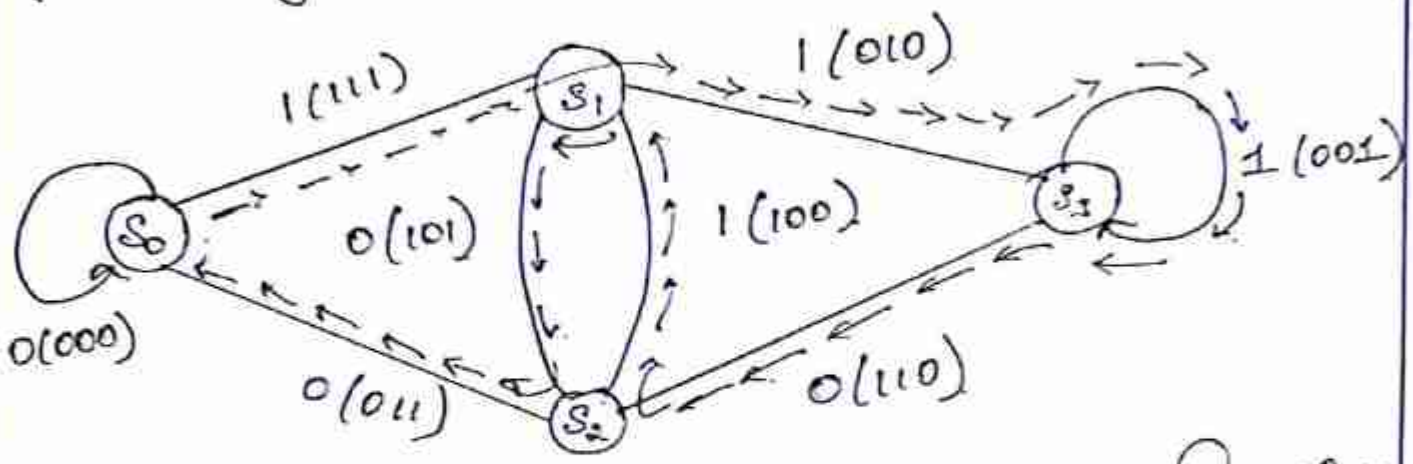


Fig:- State diagram.

By Traversing through the State diagram following the path $S_0 \to S_1 \to S_3 \to S_2 \to S_1 \to S_2 \to S_0$.

.. producing an output of

$C \to [111, 010, 001, 110, 100, 101, 011]$

Code Trellis:-

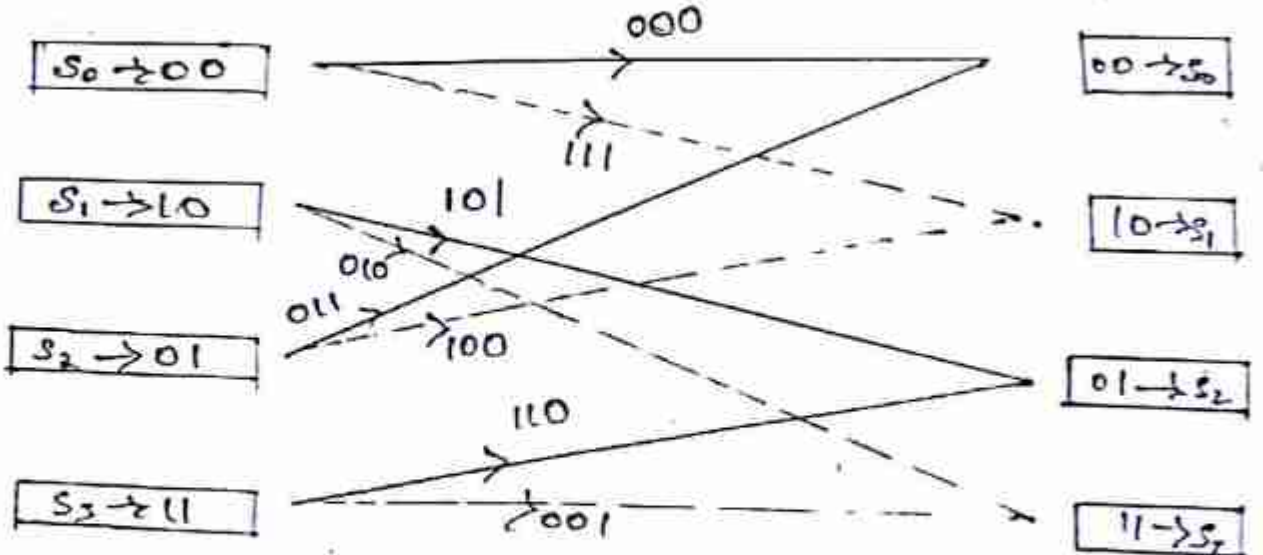


Fig:- Code Trellis:- Single Transition in a single Time Unit.

The Transition from one State to the other is given by the Sequence.

$S_0 \to S_1 \to S_3 \to S_7 \to S_6 \to S_5 \to S_2 \to S_0$

111 010 001 110 100 101 011.

The Transition from S_0 to S_1 produces an output 111, The Transition from S_1 to S_3 produces 010. and so on. We observe that the output Sequence S_0 obtained for same as obtained in "State diagram o/p"

Verification:- $g^{(1)} = 110$ // $g^{(2)} = 101$ // $g^{(3)} = 111$ //

$d = 11101$ //

NOTE
2.1.2
n k H

The generator matrix has $L \times n (L+M)$
$5 \times 3 (5+2)$
5×21 //

$$G = \begin{bmatrix} 111 & 101 & 011 & 000 & 000 & 000 & 000 \\ 000 & 111 & 101 & 011 & 000 & 000 & 000 \\ 000 & 000 & 111 & 101 & 011 & 000 & 000 \\ 000 & 000 & 000 & 111 & 101 & 011 & 000 \\ 000 & 000 & 000 & 000 & 111 & 101 & 011 \end{bmatrix}$$

$$C = D G_n = [11101] \begin{bmatrix} 111 & 101 & 011 & 000 & 000 & 000 & 000 \\ 000 & 111 & 101 & 011 & 000 & 000 & 000 \\ 000 & 000 & 111 & 101 & 011 & 000 & 000 \\ 000 & 000 & 000 & 111 & 101 & 011 & 000 \\ 000 & 000 & 000 & 000 & 111 & 101 & 011 \end{bmatrix}$$

$C = [111, 010, 001, 110, 100, 101, 011]$ //

* For the Convolutional Encoder shown in below.

1) Draw State Table. 2) Draw State Transition Table

3) Draw State diagram.

4) Find Encoder output for the message Sequence (10011) by Traversing through the State diagram.

5) Draw Code Trellis for one time unit and obtain the output for the same message Sequence (10011) using the Code trellis.

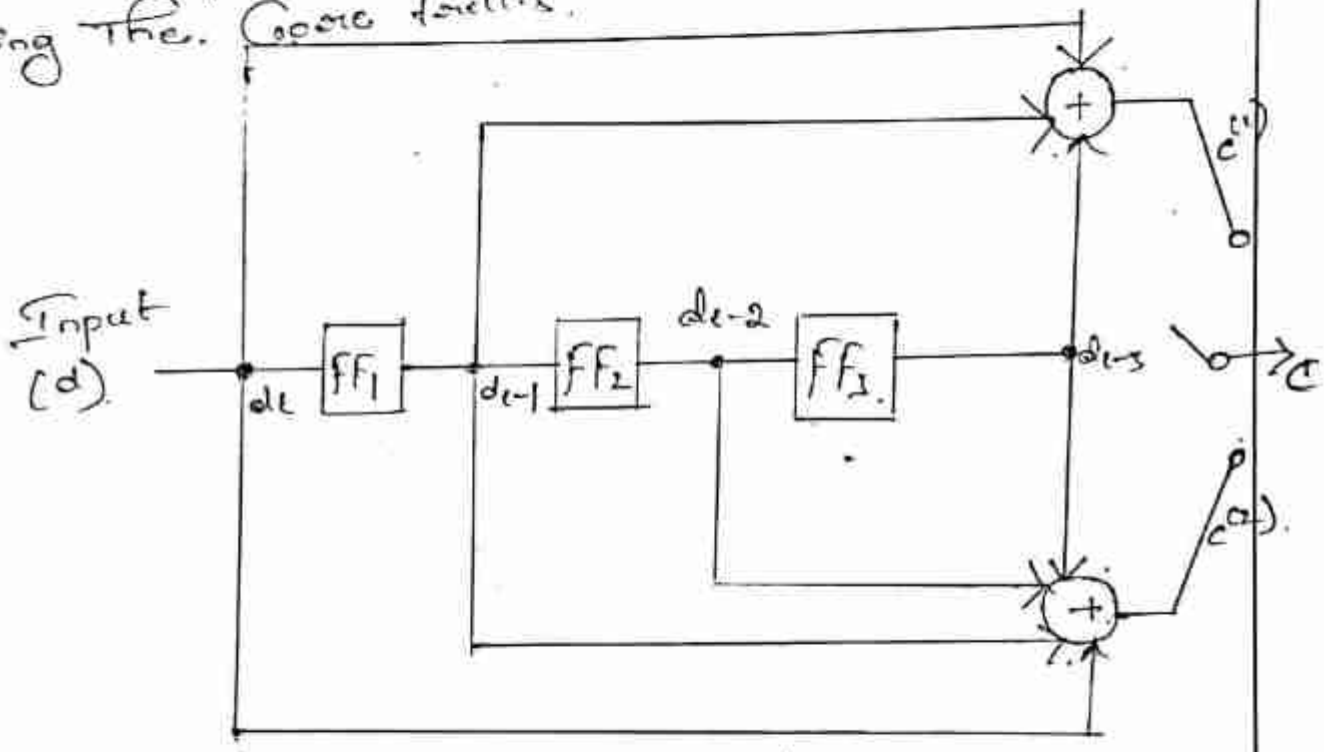


Fig:- Convolutional Encoder.

Sol:-

State Table:- There are 3-flipflops and hence there are $2^3 = 8$ States. Let the States be $S_0, S_1, \ldots, S_7$. These States and their binary description are as shown in below.

State	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Binary Description	000	100	010	110	001	101	011	111

Table : State Table.

State Transition Table :

$$c^{(1)} = d_e + d_{e-1} + d_{e-3}$$
$$c^{(2)} = d_e + d_{e-1} + d_{e-2} + d_{e-3}$$

present State	Binary Description	I/P	next State	Binary Description	d_e	d_{e-1}	d_{e-2}	d_{e-3}	output $c^{(1)}$	$c^{(2)}$
S_0	000	0	S_0	000	0	0	0	0	0	0
		1	S_1	100	1	0	0	0	1	1
S_1	100	0	S_2	010	0	1	0	0	1	1
		1	S_3	110	1	1	0	0	0	0
S_2	010	0	S_4	001	0	0	1	0	0	1
		1	S_5	101	1	0	1	0	1	0
S_3	110	0	S_6	011	0	1	1	0	1	0
		1	S_7	111	1	1	1	0	0	1
S_4	001	0	S_0	000	0	0	0	1	1	1
		1	S_1	100	1	0	0	1	0	0
S_5	101	0	S_2	010	0	1	0	1	0	0
		1	S_3	110	1	1	0	1	1	1
S_6	011	0	S_4	001	0	0	1	1	1	0
		1	S_5	101	1	0	1	1	0	1
S_7	111	0	S_6	011	0	1	1	1	0	1
		1	S_7	111	1	1	1	1	1	0

State diagram:-

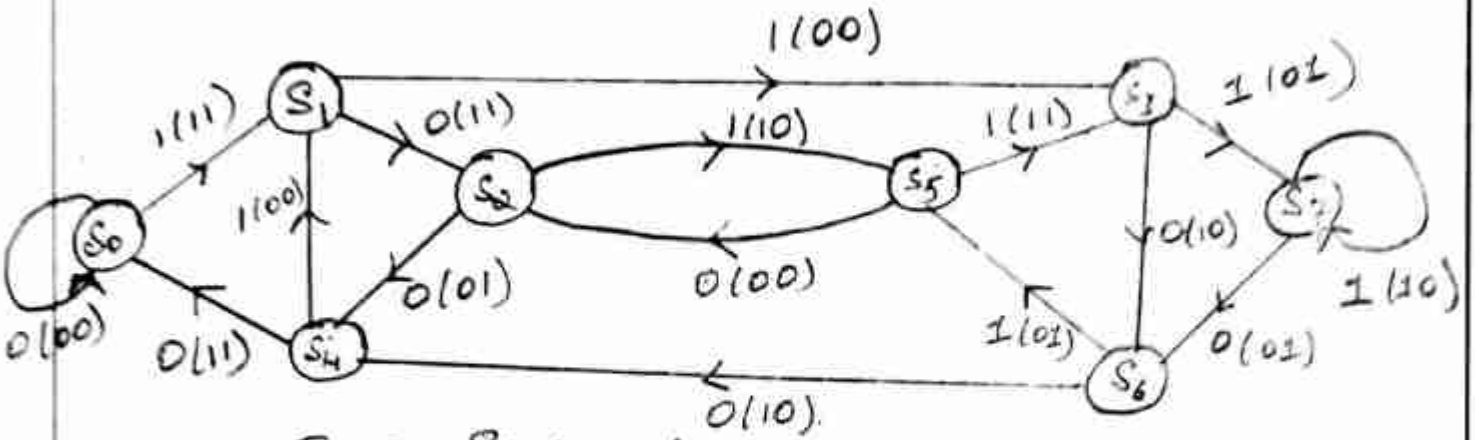


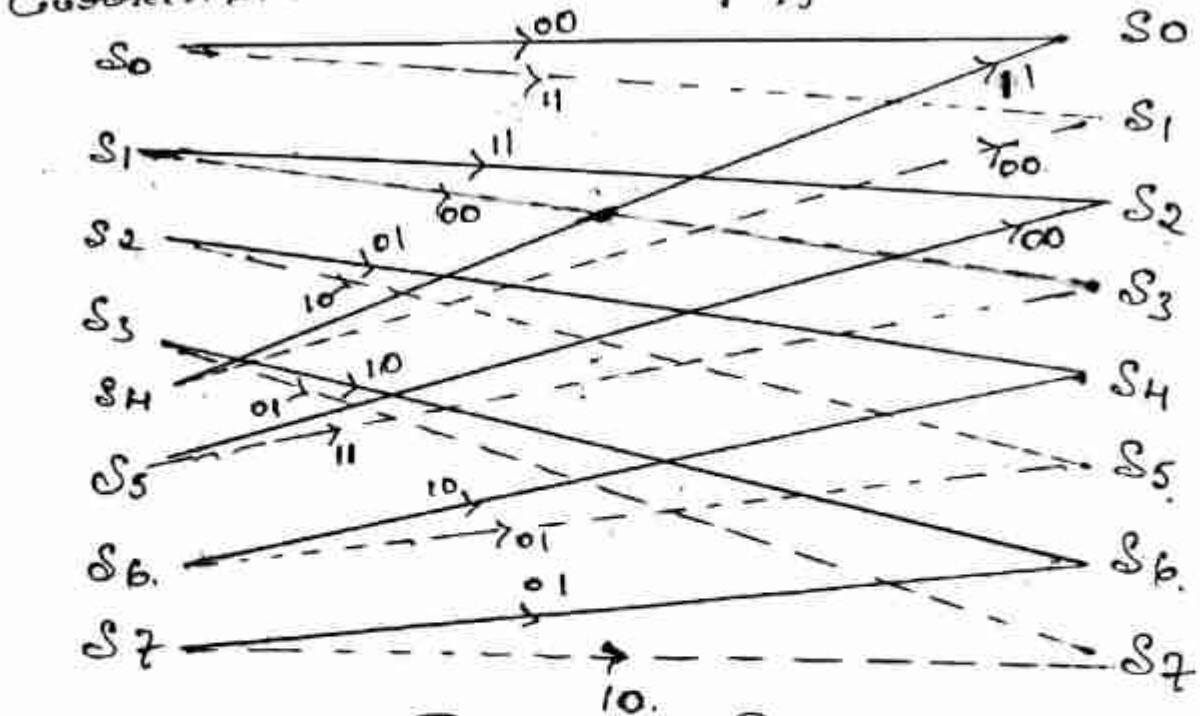
Fig:- State diagram.

Total Code length. is $n(L+M) = 2(5+3) = 16$ //

$S_0 \xrightarrow{1} S_1 \xrightarrow{0} S_2 \xrightarrow{0} S_4 \xrightarrow{1} S_1 \xrightarrow{1} S_3 \xrightarrow{0} S_6 \xrightarrow{0} S_5 \xrightarrow{0} S_0$

11 11 01 00 00 10 10 11 //

Code Trellis:-



From fig above. The Transition from one state to other given by the sequence

$S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_4 \rightarrow S_1 \rightarrow S_3 \rightarrow S_6 \rightarrow S_5 \rightarrow S_0$

11 11 01 00 00 10 10 11

Verification :- $g^{(1)} = [1101]$ $g^{(2)} = [1111]$

message Sequence $D = [10011]$ //

The generator matrix has $n = 5$ rows.

$n(n+m) = 2(5+3) = 16$ //

ROOT (from the figure)

(2 1 3)

$n \times m$

$$G = \begin{bmatrix} 11 & 11 & 01 & 11 & 00 & 00 & 00 & 00 \\ 00 & 11 & 11 & 01 & 11 & 00 & 00 & 00 \\ 00 & 00 & 11 & 11 & 01 & 11 & 00 & 00 \\ 00 & 00 & 00 & 11 & 11 & 01 & 11 & 00 \\ 00 & 00 & 00 & 00 & 11 & 11 & 01 & 11 \end{bmatrix}$$

$C = d[G]$

$= [10011] \begin{bmatrix} 11 & 11 & 01 & 11 & 00 & 00 & 00 & 00 \\ 00 & 11 & 11 & 01 & 11 & 00 & 00 & 00 \\ 00 & 00 & 11 & 11 & 01 & 11 & 00 & 00 \\ 00 & 00 & 00 & 11 & 11 & 01 & 11 & 00 \\ 00 & 00 & 00 & 00 & 11 & 11 & 01 & 11 \end{bmatrix}$

$= [11, 11, 01, 00, 00, 10, 10, 11]$ //

TRELLIS :

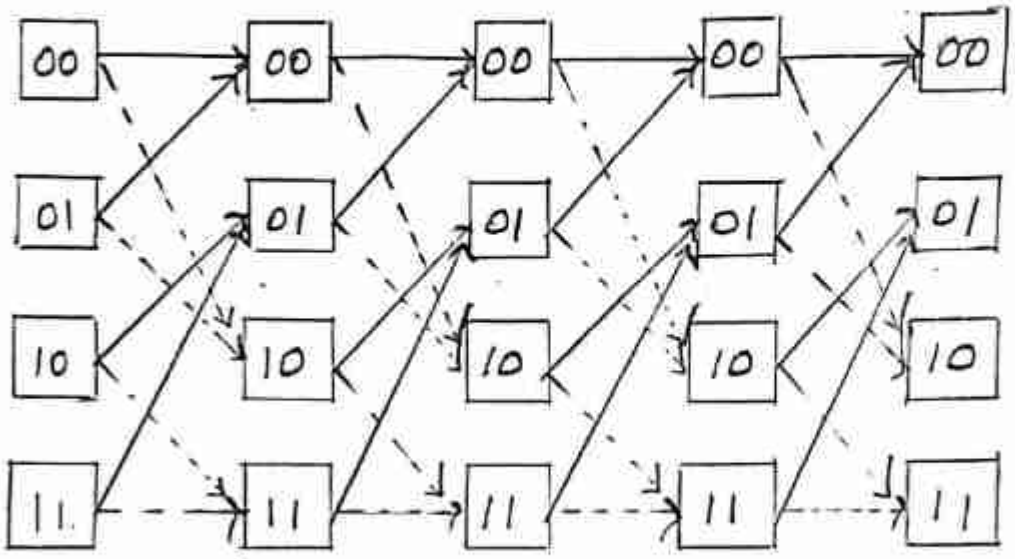


Fig:- Convolutional Code Trellis diagram.

* A trellis is a graph whose nodes are ordered into vertical slices (time), and with each node at each time connected to at least one node at an earlier time and at least one node at a later time.
* The earliest and latest times in the trellis have only one node.
* Trellises are used in encoders and decoders for Communication Theory and Cryptography.
* They are also the central datatype used in Baum-Welch algorithm for Hidden Markov models.

STATE DIAGRAMS:-

* Likely any other Sequential Circuit, a Convolution Encoder. Can be characterized by its State diagram.

* The State table for a (2,1,2) Convolution Encoder Can be written as follows: As There are two memory elements, There will be four possible States.

1. $S_0 = 00$
2. $S_1 = 01$
3. $S_2 = 10$
4. $S_3 = 11$

The State table is as shown in below.

The Corresponding State diagram. shown in below.

* Although State diagrams is one of the Easiest ways of Representing The Encoder it will not Consider the time instant into account. In other words we Cannot decide The present state of The system for a given instant.

Ex:-

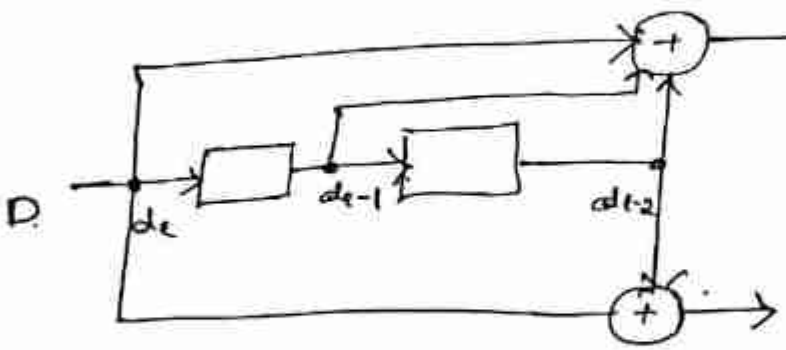


Fig:- (2,1,2) Convolutional encoder.

outputs:-

$C^{(1)} = d_t + d_{t-1} + d_{t-2}$ //

$C^{(2)} = d_t + d_{t-2}$ //

present state	Binary description	I/p	next state	Binary description	d_t	d_{t-1}	d_{t-2}	output $c^{(1)}$	output $c^{(2)}$
S_0	00	0	S_0	00	0	0	0	0	0
		1	S_1	10	1	0	0	1	1
S_1	10	0	S_2	01	0	1	0	1	0
		1	S_3	11	1	1	0	0	1
S_2	01	0	S_0	00	0	0	1	1	1
		1	S_1	10	1	0	1	0	0
S_3	11	0	S_2	01	0	1	1	0	1
		1	S_3	11	1	1	1	1	0

Table :- State Table for $(2, 1, 2)$ Code

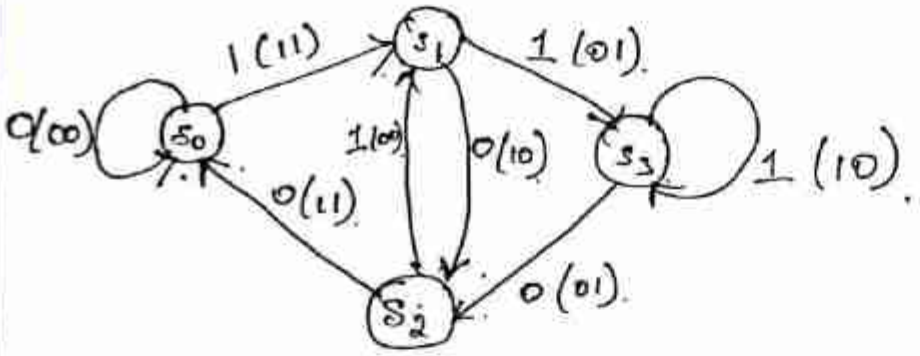


Fig: State diagram //

VITERBI ALGORITHM :-

* The Viterbi algorithm is a maximum - likelihood decoder which is optimum for a White Gaussian noise Channel.
* For an L-bit message sequence and an Encoder of memory M. The algorithm proceeds as follows (assuming that the Encoder is initially in the all-zero state at $i=0$):

Step 1 :- Starting at level (i.e time unit) $i=M$, Compute the metric for the single path entering each state of the Encoder. Store the path (survivor) and its metric for each state.

Step 2 :- Increment the level "i" by 1. Compute the metric for all the path entering each state by adding the metric of the incoming branches to the metric of the connecting survivor from the previous time unit. For each state, identify the path with the lowest metric as the survivor of step 2. Store the survivor and its metric.

Step 3 :- If level $i < L+M$, repeat step 2.
otherwise stop.

Note That the memory is $M = K-1$, where "K" is the Constraint Length of the Code.

* The great advantage of the viterbi algorithm is that the number of operations performed on decoding L bits is $L.2^{n(K-1)}$ which is linear in L.
* the number of operations performed per decoded bit is an exponential function of the Constraint length K. The exponential dependence on "K" limits the utilization of the viterbi algorithm as a practical decoding Technique to relatively Short Constraint - length Codes (Typically in the range of 7 to 11). //

INFORMATION THEORY AND CODING

5TH SEM E&C



JAYANTHDWIJESH H P **M.tech (DECS)**

Assistant Professor – Dept of E&C

B.G.S INSTITUTE OF TECHNOLOGY (B.G.S.I.T)

B.G Nagara, Nagamangala Tq, Mandya District- 571448

FORMULAS FOR REFERENCE

MODULE –3 (INFORMATION CHANNELS)

- The entropy of input symbol is given by

  H (A) = $\sum_{i=1}^{r} P(a_i)\log(\frac{1}{P(a_i)})$ bits/ message-symbol

- The entropy of output symbol is given by

  H (B) = $\sum_{j=1}^{s} P(b_j)\log(\frac{1}{P(b_j)})$ bits/ message-symbol

- The joint entropy is given by

  H (A, B) = $\sum_{i=1}^{r}\sum_{j=1}^{s} P(a_i, b_j)\log(\frac{1}{P(a_i,b_j)})$ bits/ message-symbol

- The equivocation H(A/B) is given by

  H (A / B) = $\sum_{i=1}^{r}\sum_{j=1}^{s} P(a_i, b_j)\log(\frac{1}{P(a_i/b_j)})$ bits/ message-symbol

  Since P(a_i/b_j) = $\frac{P(a_i,b_j)}{P(b_j)}$

  And another formula

  H (A/B) = H (A, B) – H (B) bits/ message-symbol

- The equivocation H(B/A) is given by

  H (B / A) = $\sum_{i=1}^{r}\sum_{j=1}^{s} P(a_i, b_j)\log(\frac{1}{P(b_j/a_i)})$ bits/ message-symbol

  Since P(b_j/ a_i) = $\frac{P(a_i,b_j)}{P(a_i)}$

  And another formula

  H (B/A) = H (A, B) – H (A) bits/ message-symbol

- The mutual information is given by

  I (A, B) = H (A) – H (A/B) bits/ message-symbol

- The channel capacity is given by

  C = (log s – h) r_s bits/ sec

  Since h= H $\left(\frac{B}{A}\right)$ = $\sum_{j=1}^{s} P_j\log(\frac{1}{P_j})$ bits/ message-symbol

- The capacity of the channel is given by

  C=B log_2 [1 + S/N]

- Source efficiency η_S is given by

  η_S = $\frac{H(X)}{H(X)max}$ x 100%

- **Source efficiency R_{η_S} is given by**

$$R_{\eta_S} = 1 - \eta_S$$

- **Channel efficiency is given by**

$$\eta_{ch} = \frac{I(X,Y)}{C} \times 100\%$$

- **Channel redundancy is given by**

Channel redundancy = (1- η_{ch}) x 100%

- **Estimation of channel capacity by MUROGA'S method**

channel with m × m noise matrix is done as below :

$$P[y_j/x_i] = \begin{bmatrix} p_{11} & p_{12} & \cdots\cdots & p_{1m} \\ p_{21} & p_{22} & \cdots\cdots & p_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ p_{m1} & p_{m2} & \cdots\cdots & p_{mm} \end{bmatrix}$$

The quantities $Q_1, Q_2, \ldots, Q_m$ are found out using

$$\begin{bmatrix} p_{11} & p_{12} & \cdots\cdots & p_{1m} \\ p_{21} & p_{22} & \cdots\cdots & p_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ p_{m1} & p_{m2} & \cdots\cdots & p_{mm} \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_m \end{bmatrix} = \begin{bmatrix} p_{11}\log p_{11} + p_{12}\log p_{12} + \cdots\cdots + p_{1m}\log p_{1m} \\ p_{21}\log p_{21} + p_{22}\log p_{22} + \cdots\cdots + p_{2m}\log p_{2m} \\ \vdots \quad \vdots \quad \vdots \\ p_{m1}\log p_{m1} + p_{m2}\log p_{m2} + \cdots\cdots + p_{mm}\log p_{mm} \end{bmatrix}$$

Then the channel capacity 'C' is given by

$$C = \log [2^{Q_1} + 2^{Q_2} + \ldots\ldots + 2^{Q_m}] \text{ bits/message-symbol.}$$

MODULE -3

INFORMATION CHANNELS

COMMUNICATION CHANNELS OR DISCRETE DATA COMMUNICATION CHANNEL OR BLOCK DIAGRAM OF COMMUNICATION SYSTEM.

DEC-2010 [HM], JUNE-JULY-2011 [ON]

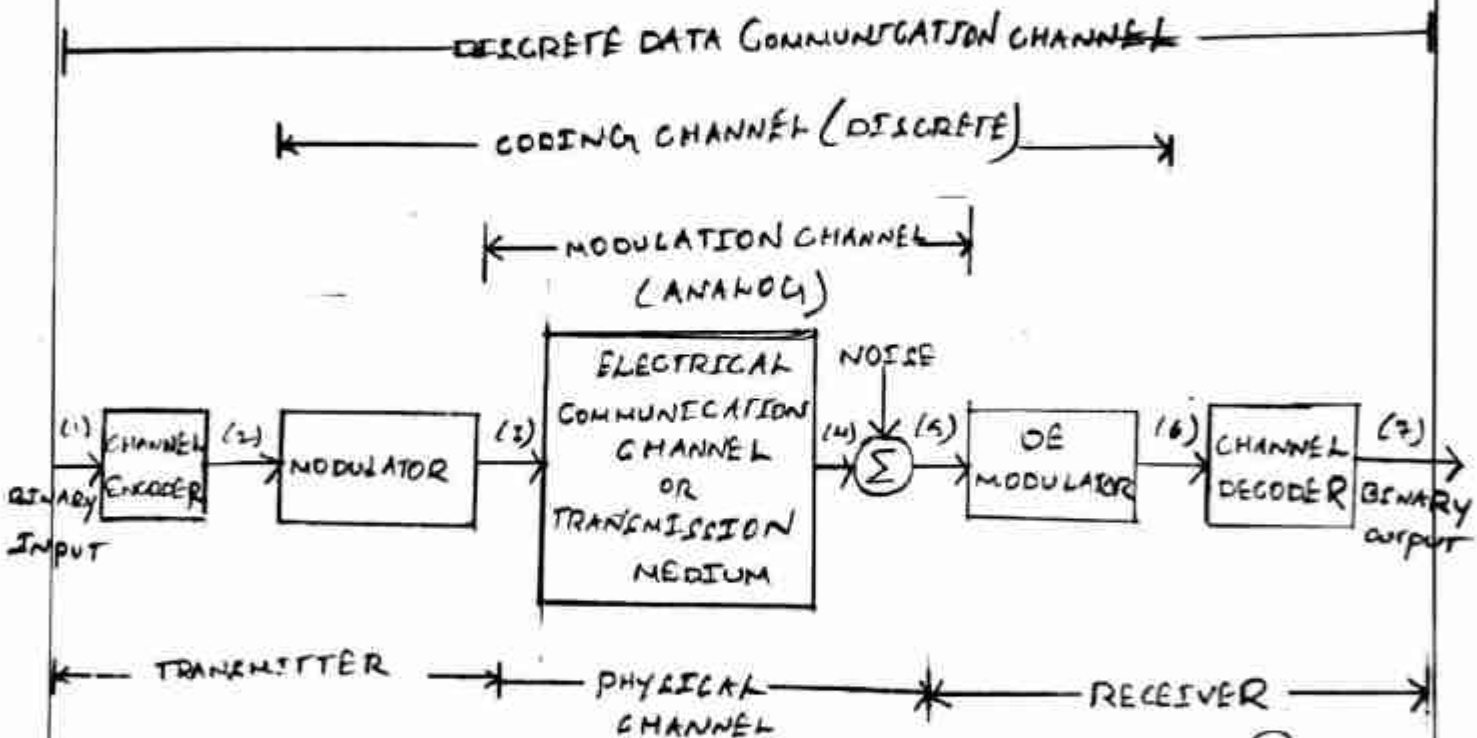


Fig: Block diagram of Communication System.

* A practical Communication System can be divided into a Transmitter, physical channel and a Receiver.
* Transmitter consists of an Encoder and a modulator, while the Receiver consists of demodulator and a decoder.

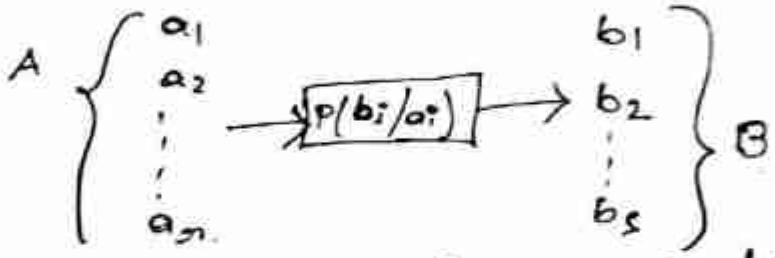
* B/w points 5 and 6 in the System shown there is a structure often referred to as a Coding Channel that accepts a sequence of symbols at its input and produces a sequence of symbol at it's output.
* The Communication Channel b/w 3 and 5 in the System provides Electrical Connection b/w the Transmitter and Receiver.
* The I/p and o/p are analog electrical waveforms. This portion of the Channel is Called a Continuous (or) modulation Channel.
* The Channel Corrupts the Signal Statistically due to various types of additive & multiplicative noise.
* An Important Characteristic of a data Communication System is The "Channel Capacity" which represents the maximum rate at which data is Transferred. Across the Channel, with an arbitrarily Small probability of error

DISCRETE COMMUNICATION CHANNEL

The input to the Channel is a Symbol belonging to an alphabet "A" with "r" Symbols. The output of the Channel is a Symbol belonging to Some other alphabet "B" with "s" Symbols.
Due to Errors in the Channel, the output Symbol may be different from the input Symbol during any Symbol Interval. Errors are mainly due to noise in the analog portion of the Channel. //

REPRESENTATION OF A CHANNEL :-

A Communication Channel may be represented by a set of input alphabet $A = \{a_1, a_2 \dots a_r\}$ Consisting of 'r' Symbols, a set of output alphabet $B = \{b_1, b_2, \dots b_s\}$ Consisting of "S" Symbols and a set of Conditional probabilities $p(b_j/a_i)$ with $i = 1, 2, 3 \dots r$ and $j = 1, 2, \dots S$ as shown below



The Conditional probabilities existance is due to the presence of noise in the Channel we have a different number of Symbol 'S' at the Receiver from 'r' Symbols at the Transmitter.

∴ we have 'rXS' number of Conditional probabilities which are represented in a "matrix" form with all the input Symbols represented row-wise and output Symbols column wise. Such a matrix Called "Channel matrix (or) noise matrix"

$$P(b_j/a_i) \text{ (or) } P[B/A] = \begin{array}{c} \\ a_1 \\ a_2 \\ \vdots \\ a_r \end{array} \begin{array}{c} \begin{array}{cccc} b_1 & b_2 & \dots & b_s \end{array} \\ \begin{bmatrix} p(b_1/a_1) & p(b_2/a_1) & \dots & p(b_s/a_1) \\ p(b_1/a_2) & p(b_2/a_2) & \dots & p(b_s/a_2) \\ \vdots & \vdots & & \vdots \\ p(b_1/a_r) & p(b_2/a_r) & \dots & p(b_s/a_r) \end{bmatrix} \end{array}$$

A Channel Can be represented with Conditional probabilities relating input and output symbols and the diagram showing these relation is known as Channel/noise diagram.

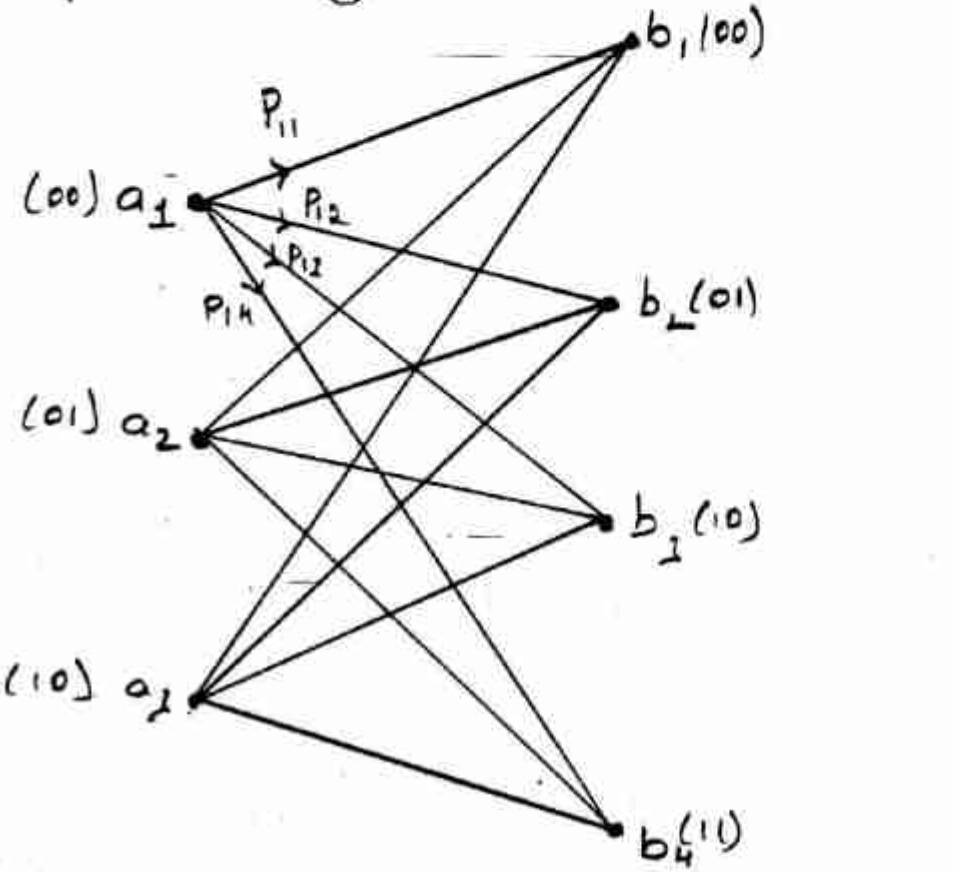


Fig: noise diagram.

In the above Channel, the i/p symbol a_1 being Transmitted having a Codeword "00" when these 2 binary symbols are Transmitted over the Channel, there is a Chance that these 2 symbols are not affected and it is received as it is at the Receiver.

This is the Conditional probability that Symbol by is Received given that a_1 is transmitted at the Transmitter denoted by $p(b/a_i)$ this Can be written as.

$P_{11} = P(b_1/a_1)$ — (1)

but at the 2nd. Transmission to a_2 there is a Chance of "00" getting Converted to 01. This Can be Considered as

$P_{12} = P(b_2/a_1)$ — (2)

$P_{13} = P(b_3/a_1)$ — (3)

$P_{14} = P(b_4/a_1)$ — (4)

But we know that Sum of all probabilities = 1.

$P_{11} + P_{12} + P_{13} + P_{14} = 1$ — (5)

Using (1) to (4) Substituting in to eq (5)

$P(b_1/a_1) + P(b_2/a_1) + P(b_3/a_1) + P(b_4/a_1) = 1$ — (6)

Generalizing Equation (6) for 'S' output Symbols and 'n' input Symbols

$P(b_1/a_1) + P(b_2/a_1) - - - - - - - - + P(b_s/a_1) = 1.$

Considering Left. Summation Sign.

$$\sum_{i=1}^{S} P(b_i/a_1) = 1$$

but it must hold good for all "n" input Symbols

$$\sum_{j=1}^{S} P(b_i/a_j) = 1 \quad //$$ for $i = 1, 2 - - - n$

∴ "THE SUM OF ALL THE ELEMENTS IN ANY ROW OF THE CHANNEL MATRIX IS EQUAL TO UNITY"

Let us Suppose that the probabilities of the input Symbols namely $P(a_1), P(a_2) - - - - P(a_n)$ are also known. Such that

$P(a_1) + P(a_2) + - - - - + P(a_n) = 1.$

$$\therefore \sum_{i=1}^{n} P(a_i) = 1 \quad //$$

Knowing the Channel matrix elements $p(b_j/a_i)$ for all "i" and "j" and the input probabilities $p(a_i)$ for all "i", the probabilities of the output symbols $p(b_j)$ for all "j" can be found using the "Theorem of Total probability"

$$p(b_1) = p(b_1/a_1)p(a_1) + p(b_1/a_2)p(a_2) + \cdots + p(b_1/a_m)p(a_m)$$

$$p(b_2) = p(b_2/a_1)p(a_1) + p(b_2/a_2)p(a_2) + \cdots + p(b_2/a_m)p(a_m)$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$p(b_j) = p(b_j/a_1)p(a_1) + p(b_j/a_2)p(a_2) + \cdots + p(b_j/a_m)p(a_m)$$

Knowing the input probabilities $p(a_i)$. Channel matrix element $p(b_j/a_i)$ and the output probabilities $p(b_j)$. The "INPUT CONDITIONAL PROBABILITIES $p(a_i/b_j)$" can be found by using "Baye's Rule.

$$p(a_i/b_j) = \frac{p(b_j/a_i)\,p(a_i)}{p(b_j)}$$

JOINT PROBABILITY. [05] JOINT PROBABILITY MATRIX:

If "x_i" are the inputs and "y_i" are the outputs. Then the joint probability is given by

$$P(x_i, y_i) = P\left(\frac{y_i}{x_i}\right) \cdot P(x_i) = P\left(\frac{x_i}{y_i}\right) \cdot P(y_i)$$

and

$$\left[P\left(\frac{y_i}{x_i}\right)\right] = \begin{bmatrix} P\left(\frac{y_1}{x_1}\right) & P\left(\frac{y_2}{x_1}\right) & \cdots & P\left(\frac{y_n}{x_1}\right) \\ P\left(\frac{y_1}{x_2}\right) & P\left(\frac{y_2}{x_2}\right) & \cdots & P\left(\frac{y_n}{x_2}\right) \\ P\left(\frac{y_1}{x_m}\right) & P\left(\frac{y_2}{x_m}\right) & \cdots & P\left(\frac{y_n}{x_m}\right) \end{bmatrix}$$

multiply 1st row of $P(y_i/x_i)$ by $P(x_1)$, Second row by $P(x_2)$ and so on we get.

$$\left[P\left(\frac{y_i}{x_i}\right)\right] \cdot P(x_i) = \begin{bmatrix} P\left(\frac{y_1}{x_1}\right)P(x_1) & P\left(\frac{y_2}{x_1}\right)P(x_1) & \cdots & P\left(\frac{y_n}{x_1}\right)P(x_1) \\ P\left(\frac{y_1}{x_2}\right)P(x_2) & P\left(\frac{y_2}{x_2}\right)P(x_2) & \cdots & P\left(\frac{y_n}{x_2}\right)P(x_2) \\ \vdots & & & \\ P\left(\frac{y_1}{x_m}\right)P(x_m) & P\left(\frac{y_2}{x_m}\right)P(x_m) & \cdots & P\left(\frac{y_n}{x_m}\right)P(x_m) \end{bmatrix}$$

But we know that $P\left(\frac{y_i}{x_i}\right) \cdot P(x_i) = P(x_i, y_i)$ Thus we get.

$$[P(x_i, y_i)] = \begin{array}{c} \\ i=1 \\ i=2 \\ \vdots \\ i=m \end{array} \begin{array}{c} \begin{array}{cccc} j=1 & j=2 & \cdots & j=n \end{array} \\ \begin{bmatrix} P(x_1, y_1) & P(x_1, y_2) & \cdots & P(x_1, y_n) \\ P(x_2, y_1) & P(x_2, y_2) & \cdots & P(x_2, y_n) \\ \vdots & \vdots & & \vdots \\ P(x_m, y_1) & P(x_m, y_2) & \cdots & P(x_m, y_n) \end{bmatrix} \end{array}$$

This is the joint probability matrix (JPM) which has the following properties.

1. $\sum_{i=1}^{m} P(x_i, y_j) = P(y_j)$ That is the sum of all the elements of the j^{th} column of JPM gives the probability of the j^{th} output — probability of all output symbols.

2. $P(x_i) = \sum_{j=1}^{n} P\left(\frac{x_i}{y_j}\right) \cdot P(y_j)$ $i = 1, 2, 3 \ldots m$ i.e.

$$P(x_i) = \sum_{j=1}^{n} P(x_i, y_j), \quad i = 1, 2, 3 \ldots m$$

Since $P(x_i/y_j) \cdot P(y_j) = P(x_i, y_j)$ (i.e probability of the i^{th} input can be obtained by adding all the elements in the i^{th} row of JPM — probability of i/p symbols).

3. $\sum_{i=1}^{m} P(x_i) = 1$

Hence $\sum_{i=1}^{m} \sum_{j=1}^{n} P(x_i, y_j) = 1$

i.e the sum of all the elements of JPM is equal to unity. //

* In a Communication System, a Transmitter has 3 input Symbols. $A = \{a_1, a_2, a_3\}$ and Receiver also has 3 output Symbols. $B = \{b_1, b_2, b_3\}$. The matrix given below shows JPM with some marginal probabilities.

a_i \ b_j	b_1	b_2	b_3
a_1	$\frac{1}{12}$	*	$\frac{5}{36}$
a_2	$\frac{5}{36}$	$\frac{1}{9}$	$\frac{5}{36}$
a_3	*	$\frac{1}{6}$	*
$p(b_j)$	$\frac{1}{3}$	$\frac{14}{36}$	*

1) Find the missing probabilities (*) in the Table
2) Find $p(b_3/a_1)$ and $p(a_1/b_3)$
3) Are the Events a_1 and b_3 Statistically Independent? Why?

Sol:-

1) Consider:

given $p(b_1) + p(b_2) + \cdots + p(b_s) = 1$

$$\sum_{i=1}^{S} p(b_i) = 1$$

$$\therefore \sum_{i=1}^{S} p(b_i) = p(b_1) + p(b_2) + p(b_3) = 1$$

[$S = 3$ [output Symbol (b_1, b_2, b_3)]]

$$\sum_{i=1}^{3} p(b_i) = p(b_1) + p(b_2) + p(b_3) = 1$$

$$\frac{1}{3} + \frac{14}{36} + p(b_3) = 1$$

$$p(b_3) = 1 - \frac{1}{3} - \frac{14}{36} = \frac{5}{18}$$

(4)

* $P(b_1)$ [1st column] is given by.

$P(b_1) = P(a_1, b_1) + P(a_2, b_1) + P(a_3, b_1)$

$\frac{1}{3} = \frac{1}{12} + \frac{5}{36} + P(a_3, b_1)$.

$P(a_3, b_1) = \frac{1}{3} - \frac{1}{12} - \frac{5}{36} = \frac{1}{9}$ //.

* $P(b_2)$ [2nd column] is given by

$P(b_2) = P[a_1 b_2] + P(a_2, b_2] + P(a_3, b_2)$

$\frac{14}{36} = P(a_1, b_2) + \frac{1}{9} + \frac{1}{6}$.

$P(a_1, b_2] = \frac{14}{36} - \frac{1}{9} + \frac{1}{6} = \frac{1}{9}$ //.

* $P(b_3]$ [3rd column] is given by.

$P(b_3) = P(a_1, b_3) + P(a_2, b_3) + P(a_3, b_3)$

$\frac{5}{18} = \frac{5}{36} + \frac{5}{36} + P(a_3, b_3)$.

$P(a_3, b_3) = \frac{5}{18} - \frac{5}{36} - \frac{5}{36} = 0$ //.

2) $P(b_3/a_1)$ is given by.

* $P(b_3/a_1) = \frac{P(a_1 \cap b_3)}{P(a_1)} = \frac{P(a_1, b_3)}{P(a_1)}$ — (1)

$\therefore P(a_1) = P(a_1, b_1) + P(a_1, b_2) + P(a_1, b_3)$

$= \frac{1}{12} + \frac{1}{9} + \frac{5}{36} = \frac{1}{3}$ // — (2)

and $P(a_1, b_3) = \frac{5}{36}$ // ← (3)

NOTE.



eq (3) & (5) Sub in to eq (1).

$$P(b_2/a_1) = \frac{P(a_1 b_2)}{P(a_1)} = \frac{(5/36)}{(1/2)} = \frac{5}{10}//$$

* $P(a_1/b_2) = \frac{P(a_1 b_2)}{P(b_2)} = \frac{(5/36)}{(5/18)}$ → from eq (4) above.

$$= 1/2//$$

3) a_1 and b_1 to be statistically Independent

$$P(a_1 \cap b_1) = P(a_1, b_1) = P(a_1) = P(b_1).$$

Consider $P(a_1) \cdot P(b_1) = (1/3)(1/3) = 1/9//$

$$P(a_1, b_1) = 1/12//$$

$$P(a_1, b_1) \neq P(a_1) P(b_1) //$$

SYSTEM ENTROPIES :- (OR) ENTROPY FUNCTION AND EQUIVOCATION :-

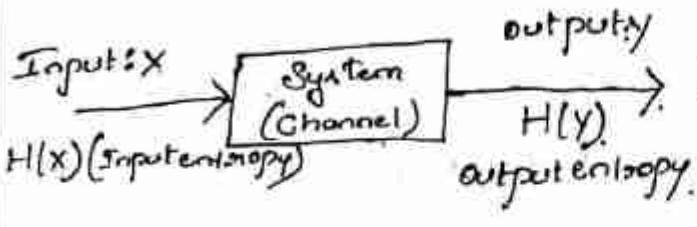


Fig :- System entropies.



Fig :- System entropies.

The input X and output Y of a channel are source with their own entropies - H(X) and H(Y) as shown in above fig.

1) PRIORI ENTROPY (OR) INPUT ENTROPY :-

The entropy of the input symbols $a_1, a_2, \ldots a_n$ before the channel is called as "priori entropy" Therefore transmission is defined as. denoted by H(A) given by

$$H(A) = \sum_{i=1}^{n} p(a_i) \log \frac{1}{p(a_i)} \quad \text{bits/msg-symbol.}$$

(or)

$$H(X) = \sum_{i=1}^{m} p(x_i) \log \frac{1}{p(x_i)} \quad \text{bits/message-symbol}$$

2) OUTPUT ENTROPY :-

The output entropy H(A) is given by.

$$H(A) = \sum_{i=1}^{s} p(b_i) \log \frac{1}{p(b_i)} \quad \text{bits/message-symbol}$$

(or)

$$H(Y) = \sum_{i=1}^{s} p(Y_i) \log \frac{1}{p(Y_i)} \quad \text{bits/message symbol}$$

3) POSTERIORI (CONDITIONAL) ENTROPY :-

The Entropy of the input Symbols $a_1, a_2 \ldots\ldots a_n$ as after Transmission and Reception of a particular output Symbol "b_i" is defined as "posteriori Conditional Entropy" denoted by $H(A/b_i)$ given by.

$$H(A/b_i) = \sum_{i=1}^{n} P(a_i/b_i) \log \frac{1}{P(a_i/b_i)} \quad \text{bits/msg-Symbol}$$

4) EQUIVOCATION :-

Equivocation is defined as the average value of all the Conditional Entropies when "i" is varied from 1 to S. It is denoted by $H(A/B)$ is given by.

$$H(A/B) = E[H(A/b_i)]$$

$$= \sum_{i=1}^{S} P(b_i) H(A/b_i) \quad \text{---(1)}$$

$\therefore$ Since $H(A/b_i) = \sum_{i=1}^{n} P(a_i/b_i) \log \frac{1}{P(a_i/b_i)}$ ---(2)

eq (2) Sub into eq (1)

$$H(A/B) = \sum_{i=1}^{S} P(b_i) \sum_{i=1}^{n} P(a_i/b_i) \log \frac{1}{P(a_i/b_i)}$$

$$H(A/B) = \sum_{i=1}^{n} \sum_{i=1}^{S} P(b_i) \cdot P(a_i/b_i) \log \frac{1}{P(a_i/b_i)}$$

where $P(b_i) P(a_i/b_i) = P(a_i \cdot b_i)$

$$H(A/B) = \sum_{i=1}^{n} \sum_{i=1}^{S} P(a_i \cdot b_i) \log \frac{1}{P(a_i/b_i)} \quad \text{bits/m-Symbol}$$

Interchanging A and B' is given by.

$$H(B/A) = \sum_{i=1}^{n} \sum_{j=1}^{s} P(a_i, b_j) \log \frac{1}{P(b_j/a_i)} \text{ bit/msg symbol}$$

W.K.T.

$$H(X/Y) = \sum_{i=1}^{n} \sum_{j=1}^{s} P(x_i, y_j) \log \frac{1}{P(x_i/y_j)} \text{ bit/msg symbol}$$

$$H(Y/X) = \sum_{i=1}^{n} \sum_{j=1}^{s} P(x_i, y_j) \log \frac{1}{P(y_j/x_i)} \text{ bits/msg symbol}$$

5) MUTUAL INFORMATION:

$$H(A) = \sum_{i=1}^{n} P(a_i) \log \frac{1}{P(a_i)} \text{ bit/message-symbol.} \quad \text{— (1)}$$

$$H(A/B) = \sum_{i=1}^{n} \sum_{j=1}^{s} P(a_i, b_j) \log \frac{1}{P(a_i/b_j)} \text{ bits/message symbol} \quad \text{— (2)}$$

"Mutual Information (eq) Transinformation" of the Channel denoted by $I(A, B)$.

$$I(A, B) = H(A) - H(A/B) \quad \text{— (3)}$$

eq (1) and (2) Sub into eq (3)

$$I(A, B) = \sum_{i=1}^{n} P(a_i) \log \frac{1}{P(a_i)} - \sum_{i=1}^{n} \sum_{j=1}^{s} P(a_i, b_j) \log \frac{1}{P(a_i/b_j)} \quad \text{— (4)}$$

Since $\sum_{j=1}^{s} P(b_j/a_i) = 1$ — (5)

multiplying 1st term of equation (4) by 1 and, substituting the above expression (5) in place of 1. we get

$$I(A,B) = \sum_{i=1}^{m} p(a_i)\left[\sum_{j=1}^{s} p(b_j/a_i)\right]\log\frac{1}{p(a_i)} - \sum_{i=1}^{m}\sum_{j=1}^{s} p(a_i b_j)\log\frac{1}{p(a_i/b_j)}$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{s} p(a_i)\,p(b_j/a_i)\log\frac{1}{p(a_i)} - \sum_{i=1}^{m}\sum_{j=1}^{s} p(a_i,b_j)\log\frac{1}{p(a_i/b_j)}$$

where $p(a_i,b_j) = p(a_i)p(b_j/a_i)$.

$$I(A,B) = \sum_{i=1}^{m}\sum_{j=1}^{s} p(a_i,b_j)\log\frac{1}{p(a_i)} + \sum_{i=1}^{m}\sum_{j=1}^{s} p(a_i/b_j)\log p(a_i/b_j)$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{s} p(a_i,b_j)\left[\log\frac{1}{p(a_i)} + \log p(a_i/b_j)\right]$$

Since: $p(a_i/b_j) = \dfrac{p(a_i,b_j)}{p(b_j)}$

$$= \sum_{i=1}^{m}\sum_{j=1}^{s} p(a_i b_j)\left[\log\frac{1}{p(a_i)} + \log\frac{p(a_i,b_j)}{p(b_j)}\right]$$

$$I(A,B) = \sum_{i=1}^{m}\sum_{j=1}^{s} p(a_i,b_j)\log\frac{p(a_i b_j)}{p(a_i)p(b_j)} \text{ bits/message-symbol} \quad (6)$$

On interchanging A and B

$$I(B,A) = \sum_{i=1}^{m}\sum_{j=1}^{s} p(a_i,b_j)\log\frac{p(a_i,b_j)}{p(a_i)p(b_j)} \text{ bits/msg-symbol} \quad (7)$$

NOTE: no changes. both are same.

Jayanthdwijesh h p. BE (E&C), M.tech (DECS)
Assistant Lecturer - Dept of E&C
BGSIT, b.g Nagara, Nagamangala Tq, Mandya.

eq (6) and (8) are identical

$$I(B,A) = I(A,B) \quad \text{---} (8)$$

from eq (3)

$$I(A,B) = H(A) - H(A/B)$$

$$\text{i.e. } I(B,A) = H(B) - H(B/A) \quad \text{---} (9)$$

from eq (2)

$$I(B,A) = I(A,B)$$

eq (3) & (9) Sub. in to eq (8)

$$H(B) - H(B/A) = H(A) - H(A/B)$$

$$H(B) + H(A/B) = H(A) + H(B/A) \quad \text{---} (10)$$

The L.H.S of eq (10) is

$$H(B) + H(A/B) = H(B,A) \quad \text{---} (11)$$

and R.H.S is

$$H(A) + H(B/A) = H(A,B) \quad \text{---} (12)$$

$\therefore H(B,A) = H(A,B)$ is defined as the joint entropy

The expression for joint entropy can be obtained by using eq (axioms) (1) and $H(B/A)$ in equation (2)

$$H(A,B) = H(A) + H(B/A)$$

$$= \sum_{i=1}^{n} p(a_i) \log \frac{1}{p(a_i)} + \sum_{i=1}^{n} \sum_{j=1}^{m} p(a_i b_j) \log \frac{1}{p(b_j/a_i)}$$

Since, $\sum_{i=1}^{s} p(b_i/a_i) = 1$.

$$H(A,B) = \sum_{i=1}^{m} p(a_i)\left[\sum_{i=1}^{s} p(b_i/a_i)\log\frac{1}{p(a_i)}\right] + \sum_{i=1}^{m}\sum_{i=1}^{s} p(a_i,b_i)\log\frac{1}{p(b_i/a_i)}$$

$$= \sum_{i=1}^{m}\sum_{i=1}^{s} p(a_i)p(b_i/a_i)\log\frac{1}{p(a_i)} + \sum_{i=1}^{m}\sum_{i=1}^{s} p(a_i,b_i)\log\frac{1}{p(b_i/a_i)}$$

Since, $p(a_i)p(b_i/a_i) = p(a_i b_i) =$ joint probability

$$H(A,B) = \sum_{i=1}^{m}\sum_{i=1}^{s} p(a_i b_i)\log\frac{1}{p(a_i)} + \sum_{i=1}^{m}\sum_{i=1}^{s} p(a_i b_i)\log\frac{1}{p(b_i/a_i)}$$

$$= \sum_{i=1}^{m}\sum_{i=1}^{s} p(a_i b_i)\left[\log\frac{1}{p(a_i)} + \log\frac{1}{p(b_i/a_i)}\right]$$

$$= \sum_{i=1}^{m}\sum_{i=1}^{s} p(a_i,b_i)\log\frac{1}{p(a_i)p(b_i/a_i)}$$

where $p(a_i)p(b_i/a_i) = p(a_i,b_i)$

$$H(A,B) = \sum_{i=1}^{m}\sum_{i=1}^{s} p(a_i b_i)\log\frac{1}{p(a_i,b_i)}$$ bits/msg symbol //

Consider eq no (9)

$$H(B/A) = H(B) - I(B,A)$$

$$H(B/A) = H(B) - I(A,B) \quad (13)$$

eq no (12) Sub into eq (13)

$$H(A)+H(B) - I(A,B) = H(A,B)$$

$$I(A,B) = H(A) + H(B) - H(A,B)$$

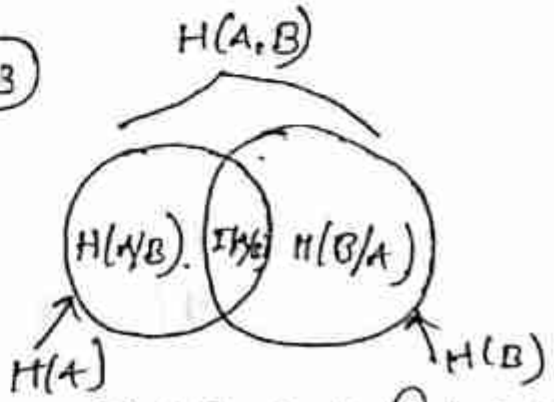


Fig:- Illustrating Entropy relations.

PROPERTIES OF MUTUAL INFORMATION.

June-2010 [8M]

property-1:-

The mutual Information of a channel is symmetric.

i.e $I(A,B) = I(B,A)$

property-2:-

The mutual Information is always non-negative.

i.e $I(A,B) \geq 0$.

proof:- $I(A,B)$ is given by

$$I(A,B) = \sum_{i=1}^{m} \sum_{j=1}^{n} p(a_i b_j) \log_2 \frac{p(a_i, b_j)}{p(a_i)\, p(b_j)}$$

$$I(A,B) = \log e \sum_{i=1}^{m} \sum_{j=1}^{n} p(a_i b_j) \ln \frac{p(a_i b_j)}{p(a_i)\, p(b_j)} \quad \text{---(1)}$$

from the inequality we know that.

$$\ln \frac{1}{x} \geq 1 - x$$

putting $x = \dfrac{p(a_i)\, p(b_j)}{p(a_i, b_j)}$

we get $\ln \dfrac{p(a_i b_j)}{p(a_i)\, p(b_j)} \geq \left[1 - \dfrac{p(a_i)\, p(b_j)}{p(a_i, b_j)}\right]$

multiplying both side of above Equation by $p(a_i, b_j)$ and Taking double Summation. we get.

$$\sum_{i=1}^{n}\sum_{j=1}^{s} P(a_i,b_j)\ln\frac{P(a_i,b_j)}{P(a_i)P(b_j)} \geq \sum_{i=1}^{n}\sum_{j=1}^{s} P(a_i,b_j)\left[1-\frac{P(a_i)P(b_j)}{P(a_i,b_j)}\right]$$

Multiply both side by $\log_e$ we get

$$\log_e \sum_{i=1}^{n}\sum_{j=1}^{s} P(a_i,b_j)\ln\frac{P(a_i,b_j)}{P(a_i)P(b_j)} \geq \log_e\left[\sum_{i=1}^{n}\sum_{j=1}^{s} P(a_i,b_j) - \sum_{i=1}^{n}\sum_{j=1}^{s} P(a_i)P(b_j)\right]$$

The L.H.S of the above equations is $I(A,B)$ from Eq (1)

$$I(A,B) \geq \log_e\left[\sum_{i=1}^{n}\sum_{j=1}^{s} P(a_i,b_j) - \sum_{i=1}^{n} P(a_i)\sum_{j=1}^{s} P(b_j)\right]$$

we have $\sum_{i=1}^{n}\sum_{j=1}^{s} P(a_i,b_j)$ = Sum of all the elements of JPM

$= 1$ //

$\sum_{i=1}^{n} P(a_i) = 1$ // and $\sum_{j=1}^{s} P(b_j) = 1$ //

$\therefore I(A,B) \geq \log_e[1-(1)(1)]$

$I(A,B) \geq 0$ // (proved)

Property-3 :-

The mutual Information of a Channel may be expressed in terms of the Entropy of the Channel o/p as

$$I(A,B) = H(B) - H(B/A)$$ //

property-4:- The mutual Information is related to the joint Entropy of the channel by

$$I(A,B) = H(A) + H(B) - H(A,B)$$ //

S.T. $H(X,Y) = H(X/Y) + H(Y)$ 2001-Aug 2002-Feb

Proof: $H(X/Y)$ is given by

$$H(X/Y) = \sum_{i=1}^{n}\sum_{j=1}^{s} P(x_i,y_j) \log \frac{1}{P(x_i/y_j)} \text{ bits/message symbol.} \quad -(1)$$

& $H(Y)$ is given by

$$H(Y) = \sum_{j=1}^{s} P(y_j) \log \frac{1}{P(y_j)} \text{ bits/message symbol} \quad -(2)$$

& $H(X,Y)$ is given by

$$H(X,Y) = \sum_{i=1}^{n}\sum_{j=1}^{s} P(x_i,y_j) \log \frac{1}{P(x_i,y_j)}$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{s} P(x_i,y_j) \log \frac{1}{P(x_i/y_j)P(y_j)}$$

[Since $P(x_i,y_j) = P(x_i/y_j)P(y_j)$]

$$H(X,Y) = \sum_{i=1}^{n}\sum_{j=1}^{s} P(x_i,y_j) \log \frac{1}{P(x_i/y_j)} + \sum_{i=1}^{n}\sum_{j=1}^{s} P(x_i,y_j) \log \frac{1}{P(y_j)}$$

from eqn (1)

$$H(X,Y) = H(X/Y) + \sum_{j=1}^{s}\left[\sum_{i=1}^{n} P(x_i,y_j)\right] \log \frac{1}{P(y_j)}$$

$$= H(X/Y) + \sum_{j=1}^{s} P(y_j) \log \frac{1}{P(y_j)}$$

$\therefore H(X,Y) = H(X/Y) + H(Y)$ proved.

* A Transmitter has an alphabet consisting of 5 letters $\{a_1, a_2, a_3, a_4, a_5\}$ and the Receiver has an alphabet of four letters $\{b_1, b_2, b_3, b_4\}$. The joint probabilities of the system are given below.

$$P(A,B) = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{array} \begin{array}{c} \begin{array}{cccc} b_1 & b_2 & b_3 & b_4 \end{array} \\ \begin{bmatrix} 0.25 & 0 & 0 & 0 \\ 0.10 & 0.30 & 0 & 0 \\ 0 & 0.05 & 0.10 & 0 \\ 0 & 0 & 0.05 & 0.1 \\ 0 & 0 & 0.05 & 0 \end{bmatrix} \end{array}$$

Compute different Entropies of this Channel.

July-2011 [10M]

Sol :-

From property-1 of IpM, addition of elements of IpM column wise. Result is probability of o/p symbols.

$\therefore p(b_1) = 0.25 + 0.10 + 0 + 0 + 0 = 0.35$ //.

$p(b_2) = 0 + 0.30 + 0.05 + 0 + 0 = 0.35$ //.

$p(b_3) = 0 + 0 + 0.10 + 0.05 + 0.05 = 0.2$ //.

$p(b_4) = 0 + 0 + 0 + 0.1 + 0 = 0.1$ //.

From property-2 of IpM, addition of element of IpM row wise. Result is probability of i/p symbols.

$p(a_1) = 0.25 + 0 + 0 + 0 = 0.25$ //

$p(a_2) = 0.10 + 0.10 + 0 + 0 = 0.40$ //.

$p(a_3) = 0 + 0.05 + 0.10 + 0 = 0.15$ //

$p(a_4) = 0 + 0 + 0.05 + 0.1 = 0.15$ //.

$p(a_5) = 0 + 0 + 0.05 + 0 = 0.05$ //.

$$H(A) = -\sum_{i=1}^{N} p(a_i) \log_2 p(a_i)$$

(or)

$$= \sum_{i=1}^{5} p(a_i) \log_2 \frac{1}{p(a_i)}$$

$$= 0.05 \log \frac{1}{0.05} + 0.4 \log \frac{1}{0.4} + 0.15 \log \frac{1}{0.15} + 0.15 \log \frac{1}{0.15} + 0.05 \log \frac{1}{0.05}$$

$$= 2.065 \text{ (or) } 2.066 \text{ bits/msg. symbol} //$$

$$H(B) = \sum_{i=1}^{3} p(b_i) \log_2 \frac{1}{p(b_i)}$$

$$= 0.35 \log \frac{1}{0.35} + 0.35 \log \frac{1}{0.35} + 0.2 \log \frac{1}{0.2} + 0.1 \log \frac{1}{0.1}$$

$$= 1.852 \text{ bits/msg-symbol} //$$

The joint entropy is given by.

$$H(A,B) = -\sum_{i} \sum_{j} p(x_i, y_j) \log_2 p(x_i, y_j)$$

(or)

$$H(A,B) = \sum_{i=1}^{5} \sum_{j=1}^{4} p(a_i, b_j) \log \frac{1}{p(a_i, b_j)}$$



$$H(A,B) = 0.05 \log \frac{1}{0.05} + 0.1 \log \frac{1}{0.1} + 0.2 \log \frac{1}{0.2} + 0.05 \log \frac{1}{0.05} + 0.1 \log \frac{1}{0.1} + 0.05 \log \frac{1}{0.05} + 0.05 \log \frac{1}{0.05} + 0.1 \log \frac{1}{0.1} = 2.666 \text{ bits/msg symbol.}$$

The Equivocation H(B/A) is found using equation as

$$H(B/A) = H(A,B) - H(A)$$
$$= 2.666 - 2.066 = 0.6 \text{ bits/msg symbol}$$

The other Equivocation $H(A/B) = H(A,B) - H(B)$

$$= 2.666 - 1.857 = 0.809 \text{ bits/msg symbol}$$

The mutual Information $I(A,B)$ is found from the Equation as.

$$I(A,B) = H(A) - H(A/B)$$
$$= 2.066 - 0.809 = 1.257 \text{ bits/msg symbol}$$

Interchange A and B we get

$$I(A,B) = H(B) - H(B/A)$$
$$= 1.857 - 0.6 = 1.257 \text{ bits/msg symbol}$$

* A Transmitter Transmits 5 Symbols a_1, a_2, a_3, a_4, a_5 with probabilities 0.2, 0.3, 0.2, 0.1 and 0.2. Given the Channel matrix $p(B/A)$. Calculate 1) $H(B)$ 2) $H(A,B)$.

$$p(B/A) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1/4 & 3/4 & 0 & 0 \\ 0 & 1/3 & 2/3 & 0 \\ 0 & 0 & 1/3 & 2/3 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Sol:- NOTE:

1st Change $p(B/A) \longrightarrow p(B,A)$.

Using the formula.

$$p(a_i b_i) = p(a_i) p(b_i/a_i)$$

(or) $p(A,B) = p(a_i) p(b_i/a_i)$

The JPM may now be Constructed by multiplying.

1st row element by $p(a_1) = 0.2 = 1/5$ //

2nd row by $p(a_2) = 0.3 = 3/10$ //, 3rd row by $p(a_3) = 0.2 = 1/5$ //

4th row by $p(a_4) = 0.1 = \frac{1}{10}$ //, 5th row by $p(a_5) = 0.2 = 1/5$ //

$$p(A,B) = \begin{matrix} \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{matrix} \begin{matrix} p(a_1)p(b_1/a_1) & \cdots & p(a_1)p(b_4/a_1) \end{matrix} \begin{bmatrix} 1\times 1/5 & 0\times 1/5 & 0\times 1/5 & 0\times 1/5 \\ 3/10\times 1/4 & \frac{3}{10}\times\frac{1}{4} & 3/10\times 0 & 3/10\times 0 \\ 1/5\times 0 & 1/5\times 1/3 & 1/5\times 2/3 & 1/5\times 0 \\ 1/10\times 0 & 1/10\times 0 & 1/10\times 1/3 & \frac{1}{10}\times 2/3 \\ 0\times 1/5 & 0\times 1/5 & 1\times 1/5 & 0\times 1/5 \end{bmatrix}$$

$$= \begin{bmatrix} 1/5 & 0 & 0 & 0 \\ 3/40 & 9/40 & 0 & 0 \\ 0 & 1/15 & 2/15 & 0 \\ 0 & 0 & 1/30 & 1/15 \\ 0 & 0 & 1/5 & 0 \end{bmatrix}$$

From property-1 of JPM, addition of elements of JPM column wise result in probability of output symbols.

$P(b_1) = \frac{1}{6} + \frac{3}{110} + 0 + 0 + 0 = \frac{11}{110}$ //

$P(b_2) = \frac{9}{110} + \frac{1}{15} + 0 + 0 + 0 = \frac{9}{30}$ //

$P(b_3) = 0 + 0 + \frac{2}{15} + \frac{1}{10} + \frac{1}{3} = \frac{11}{30}$ //

$P(b_4) = 0 + 0 + 0 + \frac{1}{15} + 0 = \frac{1}{15}$ //

1) H(B) or Entropy of the o/p symbols is given by

$$H(B) = \sum_{i=1}^{n} P(b_i) \log \frac{1}{P(b_i)} \text{ bits/msg symbol}$$ //

$$= \sum_{i=1}^{4} P(b_i) \log \frac{1}{P(b_i)}$$

$$= \frac{11}{110} \log \frac{110}{11} + \frac{7}{30} \log \frac{30}{7} + \frac{11}{30} \log \frac{30}{11} + \frac{1}{15} \log 15 = 1.822 \text{ bits/msg-symbol}$$ //

2) H(A, B) is called Joint Entropy is given by

$$H(A,B) = \sum_{i=1}^{3} \sum_{j=1}^{5} p(a_i, b_j) \log \left(\frac{1}{p(a_i, b_j)} \right)$$

$$= \frac{1}{5} \log(5) + \frac{3}{110} \log \frac{110}{3} + \frac{9}{110} \log \frac{110}{9} + \frac{1}{15} \log 15 + \frac{2}{15} \log \frac{15}{2} + \frac{1}{30} \log 30 + \frac{1}{3} \log 3 + \frac{1}{15} \log 15$$

$H(A,B) = 2.7653$ bits/msg symbol //

* For the JPM given below, Compute Individually $H(X)$, $H(Y)$, $H(X,Y)$, $H(X/Y)$, $H(Y/X)$ and $I(X,Y)$. Verify the relationship among these Entropies.

$$P(X,Y) = \begin{bmatrix} 0.05 & 0 & 0.20 & 0.05 \\ 0 & 0.10 & 0.10 & 0 \\ 0 & 0 & 0.20 & 0.10 \\ 0.05 & 0.05 & 0 & 0.10 \end{bmatrix}$$

July/Aug-2008 [10M], Dec-2008 [10M], Jan-09 [10M], Jan-2014 [10M].

Sol:-

$$P(X,Y) = \begin{array}{c} \\ X_1 \\ X_2 \\ X_3 \\ X_4 \end{array} \begin{array}{c} \begin{array}{cccc} Y_1 & Y_2 & Y_3 & Y_4 \end{array} \\ \begin{bmatrix} 0.05 & 0 & 0.20 & 0.05 \\ 0 & 0.10 & 0.10 & 0 \\ 0 & 0 & 0.20 & 0.10 \\ 0.05 & 0.05 & 0 & 0.10 \end{bmatrix} \end{array}$$

$P(Y_1) = 0.05 + 0.05 = 0.10$.

$P(Y_2) = 0.10 + 0.05 = 0.15$

$P(Y_3) = 0.2 + 0.1 + 0.2 = 0.50$

$P(Y_4) = 0.05 + 0.1 + 0.1 = 0.25$

NOTE ← Column wise

$P(X_1) = 0.05 + 0.2 + 0.05 = 0.30$.

$P(X_2) = 0.10 + 0.10 = 0.20$

$P(X_3) = 0.20 + 0.10 = 0.30$.

$P(X_4) = 0.05 + 0.05 + 0.1 = 0.20$.

NOTE ← Row wise

The Entropy of Input Symbols is given by

$$H(X) = \sum_{i=1}^{4} P(x_i) \log \frac{1}{P(x_i)}$$

$$= \left(0.3 \log \frac{1}{0.1}\right)(2) + \left(0.2 \log \frac{1}{0.2}\right)(2)$$

$$= 1.971 \text{ bits/msg-Symbol}$$

The Entropy of the o/p Symbols is given by

$$H(Y) = \sum_{i=1}^{4} P(y_i) \log \frac{1}{P(y_i)}$$

$$= 0.1 \log \frac{1}{0.1} + 0.15 \log \frac{1}{0.15} + 0.5 \log \frac{1}{0.5} + 0.25 \log \frac{1}{0.25}$$

$$= 1.743 \text{ bits/msg-Symbol}$$

The joint Entropy is given by

$$H(X,Y) = \sum_{i=1}^{4} \sum_{j=1}^{4} P(x_i, y_j) \log \frac{1}{P(x_i, y_j)}$$

$$= \left(0.05 \log \frac{1}{0.05}\right)(4) + \left(0.10 \log \frac{1}{0.10}\right)(4) + \left(0.2 \log \frac{1}{0.2}\right)(2)$$

$$= 3.122 \text{ bits/msg-Symbol}$$

* The equation $H(X/Y)$ is given by.

$$H(X/Y) = \sum_{i=1}^{n} \sum_{j=1}^{s} P(x_i, y_j) \log \frac{1}{P(x_i/y_j)} \quad \text{bits/msg-Symbol}$$

where $P(x_i/y_i) = \dfrac{P(x_i, y_j)}{P(y_j)}$

$$P(X/Y) = \begin{bmatrix} \frac{P(x_1,y_1)}{P(y_1)} & \frac{P(x_2,y_2)}{P(y_2)} & \frac{P(x_3,y_3)}{P(y_3)} & \frac{P(x_3,y_4)}{P(y_4)} \\ \frac{P(x_2,y_1)}{P(y_1)} & \frac{P(x_2,y_2)}{P(y_2)} & \frac{P(x_2,y_3)}{P(y_3)} & \frac{P(x_2,y_4)}{P(y_4)} \\ \frac{P(x_3,y_1)}{P(y_1)} & \frac{P(x_3,y_2)}{P(y_2)} & \frac{P(x_3,y_3)}{P(y_3)} & \frac{P(x_3,y_4)}{P(y_4)} \\ \frac{P(x_4,y_1)}{P(y_1)} & \frac{P(x_4,y_2)}{P(y_2)} & \frac{P(x_4,y_3)}{P(y_3)} & \frac{P(x_4,y_4)}{P(y_4)} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{0.05}{0.10} & 0 & \frac{0.2}{0.5} & \frac{0.05}{0.25} \\ 0 & \frac{0.10}{0.15} & \frac{0.1}{0.5} & 0 \\ 0 & 0 & \frac{0.2}{0.5} & \frac{0.10}{0.25} \\ \frac{0.05}{0.10} & \frac{0.05}{0.15} & 0 & \frac{0.10}{0.25} \end{bmatrix}$$

$$P(X/Y) = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \end{array} \\ \begin{bmatrix} y_2 & 0 & 2/5 & 1/5 \\ 0 & 1/2 & 1/5 & 0 \\ 0 & 0 & 2/5 & 2/5 \\ y_2 & y_1 & 0 & 2/5 \end{bmatrix} \end{array}$$

$$H(X/Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} P(x_i, y_j) \log \frac{1}{P(x_i/y_j)} \quad \text{bits/msg Symbol}$$

$$= 0.05 \log 1/y_2 + 0.05 \log 1/y_2 + 0.1 \log 3/2 + 0.05 \log \frac{1}{1/3}$$

$$+ 0.0 \log \frac{1}{2/3} + 0.1 \log 1/y_5 + 0.0 \log \frac{1}{2/3} + 0.1 \log 1/2/5$$

$$+ 0.1 \log 1/0/5 = 1.379 \text{ bits/msg Symbol} //$$

* $H(Y/X)$ is given by.

$$H(Y/X) = \sum_{i=1}^{n} \sum_{j=1}^{m} P(x_i, y_j) \log \frac{1}{P(y_j/x_i)} \quad \text{bits/msg - Symbol}$$

Consider. $P(y_j/x_i) = \dfrac{P(x_i y_j)}{P(x_i)}$

$$P(Y/X) = \begin{bmatrix} \frac{P(x_1y_1)}{P(x_1)} & \frac{P(x_1y_2)}{P(x_2)} & \frac{P(x_1y_3)}{P(x_3)} & \frac{P(x_1y_4)}{P(x_4)} \\ \frac{P(x_2y_1)}{P(x_1)} & \frac{P(x_2y_2)}{P(x_2)} & \frac{P(x_2y_3)}{P(x_3)} & \frac{P(x_2y_4)}{P(x_4)} \\ \frac{P(x_3y_1)}{P(x_1)} & \frac{P(x_3y_2)}{P(x_2)} & \frac{P(x_3y_3)}{P(x_3)} & \frac{P(x_3y_4)}{P(x_4)} \\ \frac{P(x_4y_1)}{P(x_1)} & \frac{P(x_4y_2)}{P(x_2)} & \frac{P(x_4y_3)}{P(x_3)} & \frac{P(x_4y_4)}{P(x_4)} \end{bmatrix}$$

$$P(Y/X) = \begin{bmatrix} \frac{0.05}{0.10} & 0 & \frac{0.20}{0.30} & \frac{0.05}{0.30} \\ 0 & \frac{0.10}{0.20} & \frac{0.10}{0.20} & 0 \\ 0 & 0 & \frac{0.20}{0.30} & \frac{0.10}{0.30} \\ \frac{0.05}{0.20} & \frac{0.05}{0.20} & 0 & \frac{0.10}{0.20} \end{bmatrix}$$

$$P(Y/X) = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \end{array} \\ \begin{bmatrix} 1/6 & 0 & 1/3 & 1/6 \\ 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 2/3 & 1/3 \\ 1/4 & 1/4 & 0 & 1/2 \end{bmatrix} \end{array}$$

$$H(Y/X) = \sum_{i=1}^{m} \sum_{j=1}^{n} P(x_i, y_j) \log \frac{1}{P(y_j/x_i)} \text{ bits/msg symbol}$$

$$= 0.05 \log 1/1/6 + 0.0 \log 1/2/3 + 0.05 \log 1/1/6 + 0.1 \log 1/1/2$$
$$+ 0.1 \log 1/1/2 + 0.2 \log 1/2/3 + 0.1 \log 1/1/3 + 0.05 \log 1/1/4$$
$$+ 0.05 \log 1/1/4 + 0.1 \log 1/1/2$$
$$= 1.151 \text{ bits/msg-symbol} //.$$

Verification :-

$$H(Y/X) = H(X,Y) - H(X)$$
$$= 3.122 - 1.971$$
$$= 1.151 \text{ bits/msg symbol} //.$$

$$H(X/Y) = H(X,Y) - H(Y)$$
$$= 3.122 - 1.743$$
$$= 1.379 \text{ bits/msg-symbol} //.$$

The mutual Information $I(X,Y)$ is given by

$$I(X,Y) = H(X) - H(X/Y) = 1.971 - 1.379$$
$$= 0.592 \text{ bits/msg-symbol}$$

Verification: $I(X,Y) = H(Y) - H(Y/X)$

$$= 1.741 - 1.151$$
$$= 0.592 \text{ bits/msg-symbol}$$

Verified //

SHANNON – HARTLEY LAW (OR) SHANNON – HARTLEY THEOREM (OR) SHANNON'S INFORMATION CAPACITY THEOREM (OR) CHANNEL CAPACITY THEOREM :-

JULY/AUG-2005 [6M], JAN/FEB-2006 [6M], JAN/FEB-2007 [6M]. JULY/AUG-2009 [8M],

When Shannon's Theorem of Channel Capacity is applied specifically to a channel in which the noise is gaussian in nature is known as Shannon Hartley Theorem. It is also called Information Capacity Theorem.

Statement of the Theorem

Consider a band limited gaussian channel operating in the presence of AWGN. shown in below fig.

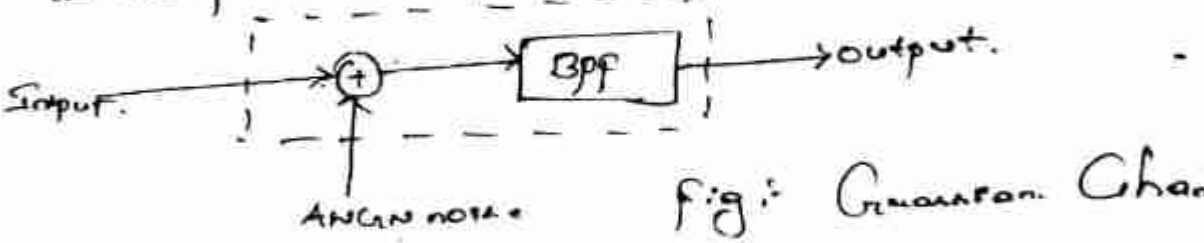


fig: Gaussian Channel.

The Channel capacity of a white bandlimited gaussian channel is.

$$C = B \log_2\left(1 + \frac{S}{N}\right) \text{ bits/sec}$$

Here B → is the Channel bandwidth.

S → is the signal power

N → is the total noise power within the Channel bandwidth.

* Assume that the quantization level of the signal be seperated by $\alpha = \lambda\sigma$. If 'λ' is chosen to have a large value, a smaller error probability could be achieved. Further if each message is represented with a voltage level, we will have 'M' distinct levels for 'M' possible messages.

Then the average signal power can be written as.

$$S = \frac{M^2 - 1}{12} (\lambda\sigma)^2$$

* The number of levels for 'M' given average signal power is therefore

$$M = \left[1 + \frac{12}{\lambda^2} \frac{S}{N}\right]^{1/2}$$

* where $N = \sigma^2$ is the noise power, Assuming that all messages are equally likely, they would carry equal amount of Info'n with an entropy of 'H' is given by

$$H = \log_2 M = \frac{1}{2} \log_2 \left[1 + \frac{12}{\lambda^2} \frac{S}{N}\right] \text{ bits/msg}$$

* We now approximate $\tau = 0.5/B$ so that rate of Information can be written as

$$R = \frac{1}{\tau} = 2B \text{ messages/s}$$

* The rate of Information 'R' and substituting for 'H' is.

$$R = rH = B \log_2 \left[1 + \frac{12}{\lambda^2} \frac{S}{N}\right]$$ //

[OR]

CHANNEL CAPACITY THEOREM [SHANNON – HARTLEY LAW]

Statement of Shannon Hartley Law:-

Shannon Hartley law is also called Shannon's Third Theorem. States that the Capacity of a band-limited Gaussian Channel with AWGN is given by

$$C = B \log\left[1 + S/N\right] \text{ bits/sec.}$$

where B = Channel bandwidth in Hz

S = Signal power in watts.

N = noise power in watts $= \eta B$

where the two sided power Spectral density of noise is $(\eta/2)$ watts/Hz.

Proof:- The Channel Capacity "C" is given by

$$C = [H(Y) - H(N)]_{max.} \quad \text{— (1)}$$

If the noise is additive, white and Gaussian having a power "N" in a bandwidth of "B" Hz.

$$\therefore H(N)_{max.} = B \log_2 TTeN \text{ bits/sec} \quad \text{— (2)}$$

When the i/p signal is limited to an average power "S" over the same bandwidth of "B" Hz. and when the signal at the Receiver $Y = X + N$ with X and N being independent.

∴ The Received Signal power is nothing but variance given by. $\sigma_Y^2 = (S+N)_{\#}$ — (3).

The entropy will be maximum if the Signal is Gaussian and the maximum entropy is given by

$$H(Y)_{max} = B \log 2\pi e \sigma_Y^2 \text{ bits/sec.} \quad — (4)$$

Using eq. (3) in (4) we get

$$H(Y)_{max} = B \log 2\pi e (S+N) \text{ bits/sec} \quad — (5)$$

Using eq. (2) and (5) in Eq. (1) we get

$$C = B \log 2\pi e (S+N) - B \log 2\pi e N.$$

$$= B \log \frac{2\pi e (S+N)}{2\pi e N}$$

$$= B \log \left[\frac{S+N}{N}\right] \text{ bits/sec}$$

(or)

$$C = B \log \left[\frac{S}{N} + 1\right] \text{ bits/sec} //.$$

Alternate proof:-

* If "S" is the Signal power and "N" the noise power. Then the net voltage level in the case of additive noise is given by $\sqrt{S+N}$ and the noise voltage level by $\sqrt{N}$.

* Since the voltage levels close to $\sqrt{N}$ cannot be clearly distinguished from noise, we have according to Sampling theorem $\sqrt{S+N}/\sqrt{N}$ number of distinguishable levels, Assume all these levels to be equiprobable,

∴ The probability 'P_i' is given by

$$P_i = \frac{1}{(\sqrt{S+N}/\sqrt{N})} = \frac{1}{q} \quad \text{for all 'i' number of levels.}$$

∴ The maximum Information Content (avg) is given by

$$I = \log_2(q) = \log_2 \frac{\sqrt{S+N}}{\sqrt{N}} \text{ bits/level} \quad \text{---(1)}$$

If "t" Represents the time taken to Transmit a Signal corresponding to i^{th} level, Then according to Nyquist rate, if it is limited to the bandwidth "B" by

$$"t" = \frac{1}{2B} \text{ Secs/level}$$

* If "n" is the maximum number of level sno Second.

Then $n = \frac{1}{t} = 2B$ number of levels/sec ---(2)

* The maximum rate of Information Transmission which is the Channel Capacity "C" is given by

$$C = nI \text{ bits/sec} \quad \text{---(3)}$$

Eq (1) and (2) Sub into eq (3)

$$C = 2B \log_2 \frac{\sqrt{S+N}}{\sqrt{N}} \quad \text{(or)} \quad = B \log\left(1+\frac{S}{N}\right) \text{bits/sec}$$

CHANNEL CAPACITY FOR CHANNELS OF INFINITE BANDWIDTH

OR PROVE THAT $\lim_{B\to\infty} C = 1.44 S/\eta$. July/Aug-2004 [6M], Aug-2003 [6M], Feb-2003 [6M], July-2001 [6M]. Aug-2002 [6M] JAN-FEB 2006 [3M], Aug-2008 [4M], DEC-2012 [6M].

The Shanon-Hartley Theorem. Suggests That a Channel with Infinite bandwidth will have Infinite Capacity.

Considering the noise power Spectral density as $\eta/2$ i.e. Total noise power becomes $N = \eta B$

$$C = B \log_2(1 + S/N)$$

$$= B \log_2(1 + S/\eta B)$$

$$= \frac{S}{\eta}\left(\frac{\eta}{S}\right)(B) \log_2\left(1 + \frac{S}{\eta B}\right)$$

$$C = \frac{S}{\eta} \log_2\left(1 + \frac{S}{\eta B}\right)^{\frac{\eta B}{S}}$$

We know That. $\lim_{x\to 0}(1+x)^{1/x} = e$

Hence as $x \to S/\eta B$. OR $1/x = \eta B/S$, As $B \to \infty$ The Capacity of the Channel becomes.

$$C_\infty = \lim_{B\to\infty} C = \frac{S}{\eta} \log_2(1+x)^{1/x}$$

$$= \frac{S}{\eta} \log_2(e) = \frac{S}{\eta} 1.44 //$$

OR $C = S/N_0\ 1.44$ //

* For an AWGN Channel with 4kHz bandwidth and noise power spectral density ($\eta/2$) = 10^{-12} W/Hz. The signal power required at the receiver is 0.1 mW. Calculate the capacity of the channel

Dr G-2.010[HM]

Sol :- given. B = 4KHz = 4000 Hz //.

$\eta/2$ (or) $N_0/2 = 10^{-12}$ watts/Hz //, S = 0.1 mW (or) 0.1×10^{-3} //

∴ noise power = $N = N_0 B$

$= 2 \times 10^{-12} \times 4000$ Hz

$N = 8 \times 10^{-9}$ watts //

NOTE

$N_0/2 = 10^{-12}$ W/Hz

$N_0 = 10^{-12} \times 2$ W/Hz

∴ Channel Capacity "C" = $B \log_2 [1 + S/N]$

$C = 4000 \log_2 \left[1 + \frac{0.1 \times 10^{-3}}{8 \times 10^{-9}}\right]$

$C = 54.43$ bits/sec //

(or) $C = 54.43$ K bits/sec //

* Calculate the capacity of a gaussian channel with a bandwidth with a bandwidth of 1 MHz and (S/N) ratio of 30 dB

July/Aug-2008 [5M]

Sol :- $B = 1$ MHz $= 10^6$ Hz //.

* $10 \log_{10} (S/N) = 30$ dB

$\log_{10} (S/N) = \frac{30}{10} = 3$

∴ $(S/N) = 10^{(3)}$ //

* Channel Capacity $C = B \log_2 (1 + S/N) = 10^6 \frac{\log (1+10^3)}{\log (2)}$

$C = 9.967$ M bits/sec //

* GRT Terminal is used to Enter alphanumeric data into a Computer. The GRT is connected through a voice-grade Telephone line having usable bandwidth of 3KHz and an output (S/N) of 10dB. Assume that the terminal has 128 Characters and data is sent in an independent manner with Equal probability.

(a). Find the average information per character.

(b) Find Capacity of the Channel.

(c) Find the maximum rate at which data can be sent from terminal to the Computer without error.

June-July-2011[6M], Jan/Feb-2002[10M]

Sol:-

1) Since all the character are Equiprobable. the average Information Content per Character is

$H = H_{max} = \log_2(Q) = \log_2(128)$.

$\therefore H = 7$ bits/character //

2) Given $B = 3KHz = 3000$ Hz //.

$10\log_{10}(S/N) = (S/N)dB$

$10\log_{10}(S/N) = 10dB$

$\log_{10}(S/N) = 1$

$S/N = 10^1$.

$\therefore S/N = 10$ //.

NOTE

dB to Ratio Conversion

∴ The Channel Capacity is given by

$$C = B \log_2 (1 + S/N)$$
$$= 3000 \log_2 (1 + 10)$$
$$C = 10378.295 \text{ bits/sec} //$$

3) Average Information rate $= R_s = r_s H$.

∴ Error free Transmission. We must have

$$R_s = C.$$

Since $r_s H < C$.

$$r_s < C/H$$

$$r_s < \frac{10378.295}{7}$$

$$r_s < 1482.61 \text{ Symbol/sec} //$$

* Calculate the Capacity of Lowpass Channel with a usable bandwidth of 3000Hz and $S/N = 10^3$ at the Channel output. Assume the Channel noise to be Gaussian.

So 1 ∴ $C = B \log_2 (1 + S/N)$.

$$= (3000) \log_2 (1 + 1000)$$
$$= 30,000 \text{ bits/sec} //$$

* A voice grade channel of the telephone network has the bandwidth of 3.4KHz. Calculate.

1) The channel capacity for a SNR of 30dB

2) The minimum SNR required to support information transmission at the rate of 4800 bits/sec.

June/July - 2009 [6M], July - 2012 [6M]

July/Aug - 2006 [4M]

Sol: Given: Channel bandwidth B = 3.4KHz

$$10\log_{10}(S/N) = (S/N)dB$$

$$\log_{10}(S/N) = \frac{30dB}{10}$$

$$\log_{10}(S/N) = 3dB$$

$$S/N = 10^3 dB = 1000$$

1) Channel capacity "C" $= B\log_2(1+S/N)$

$$= 3400\log_2(1+1000)$$

$$C = 33.888 \text{ K bits/sec}$$

2) Given $C = 4800$ bits/sec

$$C = B\log_2(1+S/N)$$

$$4800 = 3400\log_2(1+S/N)$$

$$\log_2(1+S/N) = \frac{4800}{3400} = 1.4117$$

$$(1+S/N) = 2^{(1.4117)} = 2.6605$$

$$\left(1 + S/N\right) = 2.6605$$

$$S/N = 2.6605 - 1 = 1.6605 //$$

(or) $$\frac{S}{N} \text{ in dB} = 10 \log_{10}(1.66)$$

$$S/N = 2.2 \text{dB} //$$

Ex: ** Dec-2012 [6M]

* prove that the infinite bandwidth Signal Energy to noise ratio E_b/η @ E/η approaches a limiting value

Sol:- for infinite bandwidth Signal Energy to noise ration E_b/η approaches a limiting value.

$$\left(E_b/\eta\right)_\infty = \lim_{B \to \infty}\left(E_b/\eta\right) = \lim_{B \to \infty}\left[\frac{2^{C/B} - 1}{C/B}\right]$$

where $E_b = 2^{C/B} - 1$ //

$\eta = C/B$ //

let $C/B = x$ as $B \to \infty$, $x \to 0$

$$\left(E_b/\eta\right)_\infty = \lim_{x \to 0}\left[\frac{2^x - 1}{x}\right] \quad \text{---(1.)}$$

---(2.)

∴ Assume let $y = 2^x$

Taking ln on both side

$$\ln y = x \ln(2)$$

diff w.r.t "x"

$$\frac{1}{y}\frac{dy}{dx} = \ln(2)$$

$$\int_y \frac{dy}{dx} = \ln(2)$$

$$\frac{dy}{dx} = \ln(2)\, dy \quad \text{—— (3)}$$

eq (2) Sub into eq (3)

$$\frac{dy}{dx} = 2^x \ln(2) \quad \text{—— (a.)}$$

eq (1) Differentiating both numerator and denominator of the RHS of Equating 1 with respect to "x" we get

$$\left(\frac{E_0}{\eta}\right)_\infty = \lim_{x\to 0}\left[\frac{\frac{d}{dx}(2^x - 1)}{\frac{d}{dx}(x)}\right]$$

$$= \lim_{x\to 0}\left[\frac{\frac{d}{dx}(2^x) - \frac{d}{dx}(1)}{\frac{d(x)}{dx}}\right] \quad \text{—— (4.)}$$

Eq (2) Sub into eq (4)

$$= \lim_{x\to 0}\left[\frac{\frac{dy}{dx} - \frac{d(1)}{dx}}{\frac{d(x)}{dx}}\right]$$

$$= \lim_{x\to 0}\left[\frac{\frac{dy}{dx}}{\frac{1}{1}}\right] \quad \text{—— (5)}$$

Eq (a) Sub into eq (5)

$$= \lim_{x\to 0}\left[2^x \ln(2)\right].$$

$$\left(\frac{E_b}{\eta}\right)_\infty = 2^{(0)} \ln(2) = \ln(2) = 0.693$$

(or) $\left(E_b/\eta\right)_\infty$ in dB $= 10\log_{10}(0.693)$

$= -1.6$ dB

U.4.4.

IMP [DEC-2012 [6M]]

* A Black and white Television picture may be viewed as consisting of approximately 3×10^5 elements each one of which may occupy one of 10 distinct brightness levels with equal probability. Assume.

(a) the rate of Transmission. i.e 30 picture frames per second and.

(b) the Signal-to-noise ratio is 30dB.

Using the Channel Capacity theorem. Calculate the minimum bandwidth required to support the Transmission of the resultant video signal.

Sol:- Given number of elements/picture frame $= 3\times10^5$

number of brightness levels $= 10$.

$\therefore$ number of different frames possible $= 10^{3\times10^5}$ frames.

Since all the levels are Equiprobable, the maximum average Information Content per frame is given by

$$I = \log_2 10^{3\times10^5} \text{ bits/frame}$$

$$= 3\times10^5 \log_2(10) \text{ bits/frame}$$

$$I = 9.96\times10^5 \text{ bits/frame}$$

The maximum Rate of Information is given by

$$R_{s\,max} = r_s \, I$$

$$= (30 \text{ frames/sec})(9.96 \times 10^6 \text{ bits/frame})$$

$$R_{s(max)} = 29.88 \times 10^6 \text{ bits/sec} //$$

According to Shannon's 2nd Theorem $R_{s\,max}$ is equal to Channel Capacity "C" and according to Shannon-Hartley law

$$C = B \log_2 (1 + S/N)$$

given $10 \log_{10} (S/N) = (S/N)_{dB}$

$$10 \log_{10} (S/N) = 30 \text{ dB}$$

$$\log_{10} (S/N) = 3 \text{ dB}$$

$$S/N = 10^3$$

$$= 1000 //$$

$$C = B \log_2 (1 + S/N)$$

Since $C = R_{s(max)} = 29.88 \times 10^6$ bits/sec.

$$29.88 \times 10^6 = B \log_2 (1 + 1000)$$

$$29.88 \times 10^6 = B (9.96722)$$

$$B = \frac{29.88 \times 10^6}{9.96722} = 2.99 \text{ MHz} //$$

(OR)

$$= 3 \text{ MHz} //$$

* Alphanumeric data are entered into a computer from a remote terminal through a voice-grade telephone channel. The channel has a bandwidth of 3HKHz, and output signal-to-noise ratio of 20dB. The terminal has a total of 128 symbols. Assume that the symbols are equiprobable and successive transmissions are statistically independent.

(a) Calculate Channel Capacity.
(b) Find the average information content persymbol in long transmissions.
(c) Calculate the maximum symbol rate for which errors free transmission over the channel is possible.

Sol:- Given. $B = 3HKHz = 3H00Hz$

$$10\log_{10}(S/N) = (S/N)_{dB}$$

$$10\log_{10}(S/N) = (20.dB)$$

$$\log_{10}(S/N) = 2$$

$$S/N = 10^2.$$

$$S/N = 100 //.$$

(a). Channel Capacity.

$$C = B\log_2(1+S/N)$$

$$C. = B\log_2.(1+100.) = 3H00\frac{\log.(1+100.)}{\log(2).}$$

$$= 22637.9.\ bits/sec.//.$$

b) Average Information Content per Character, i.e.,

$$H = H_{max} = \log_2(q) = \log_2(128) = 7 \text{ bits/character}$$

c) Average Information rate $R_s = r_s H$.

For error-free Transmission, we must have $R_s < C$

$$R_s < C.$$

$$\therefore r_s H < C.$$

$$r_s < \frac{C}{H}$$

$$r_s < \frac{22637}{7}$$

$$r_s < 3233 \text{ Symbols/sec}$$

I.M.

* An Analog Signal has a 4KHz bandwidth. The Signal is Sampled at 2.5 times the Nyquist rate and each Sample quantized into 256 equally likely levels. Assume that the Successive Samples are Statistically Independent.

1) Find the Information rate of this Source.

2) Can the output of this Source be transmitted without errors over a Gaussian Channel of bandwidth 50KHz and (S/N) ratio of 20dB?

3) If the output of this Source is to be Transmitted without errors over an analog Channel having (S/N) of 10dB. Compute the bandwidth requirement of the Channel.

Aug-2004 [10M]

Sol:- Given $B = 4KHz = 4000Hz$

$\therefore$ Nyquist rate $= 2B = 8000Hz$

∴ Sampling rate f_s = (2.5) (Nyquist rate)
= (2.5) (8000)
= 20,000 Samples/sec //

Since all the 256 quantization levels are equally likely the maximum Information Content is.

$I = \log_2(q) = \log_2(256) = 8$ bits/sample //

1) Information Rate $R_s = f_s I$
= (20000 Sample/sec) (8 bits/sample)
= 1,60,000 bits/sec //

2) From Shannon–Hartley law.

$C = B \log(1 + S/N)$

given $10 \log_{10}(S/N) = (S/N)_{dB}$

$10 \log_{10}(S/N) = 20$ dB

$\log_{10}(S/N) = 2$

$S/N = 10^2 = 100$ //

B = 50 KHz = 50000 Hz.

$C = 50000 \log_2(1 + 100)$
= 332.91 × 10³ bits/sec //

3) given $10 \log_{10}(S/N) = 10$ dB

$\log_{10}(S/N) = 1$

$S/N = 10^1$
= 10 //

$C = B \log(1+S/N) = R_s$

$B = \frac{R_s}{\log(1+S/N)} = \frac{160\times10^3}{\log(1+10)} = 46.25\text{KHz}$ //

Jan ***

* A Gaussian channel has a 10MHz bandwidth. If (S/N) ratio is 100, Calculate the channel capacity and the maximum Information rate.

Sol :- $B = 10\text{MHz}$ // $S/N = 100$.

$C = B\log_2(1+S/N) = 10\times10^6 \log_2(1+100)$

$= 10\times10^6 \log_2(101)$

$= 66.58\times10^6$ bits/sec //

maximum Information rate $R_{max} = C_\infty = (S/N) B \log_2(e)$

$R_{max} = (100)(10\times10^6)\log_2 e$.

$= 1.44\times10^9$ bits/sec //

V.V.V.Imp.

* A Gaussian channel has a 10MHz bandwidth and a two sided noise power spectral density $(\eta/2)$ of 10^{-14} watt/Hz. The Signal power at the Receiver has to be maintained at level less than (or) Equal to $\frac{1}{100}$ of a milliwatt. Calculate the Capacity of the channel.

Sol :- given $B = 10\text{MHz} = 10\times10^6\text{Hz}$. $\eta/2 = 10^{-14}$ watt/Hz

$\therefore \eta = 2\times10^{-14}$ watt/Hz

$\therefore$ Noise power $N = \eta B = (2\times10^{-14})(10\times10^6)$

$= 2\times10^{-7}$ watts //

Signal power. $S \leq \frac{1}{100}$ mW.

$S \leq 1 \times 10^{-5}$ watts //

$$\left(\frac{S}{N}\right)_{max} = \frac{10^{-5}}{2 \times 10^{-7}} = 50 //$$

∴ Channel Capacity $C = B \log_2 (1 + S/N)$

$= 10 \times 10^6 \log_2 (1+50)$

$= 56.724$ M bits/sec //

Ex:

* A friend of yours says he can design a system for transmitting the output of a microcomputer to a line printer operating at a speed of 30 lines/minute over a voice grade telephone line with a bandwidth of 3.5 kHz and $(S/N) = 30$ dB. Assume that the line printer needs 8 bits of data per character and prints out 80 characters per line. What would you believe him?

Sol: Given $B = 3.5$ kHz //

$10 \log_{10} (S/N) = (S/N)$ dB

$\log_{10} (S/N) = \frac{30 \text{ dB}}{10}$

$S/N = 10^3 = 1000 //$

∴ Channel Capacity $= B \log (1 + S/N)$

$= 3500 \log (1 + 1000)$

$= 34885.292$ bits/sec //

Given. $r_s = 30$ lines / minute

$= \frac{30}{60} = 0.5$ lines/sec //

Rate of Transmission. $= Hr_s$

$R_t = (8 \text{ bits/character})(80 \text{ characters/line})(0.5 \text{ line/sec})$

$R_t = 320$ bits/sec //

Since $R_t << C$ //

*. A friend of yours says that the Con. design a system for Transmitting the output of a microcomputer to a line printer operating at a speed of 30 lines/minute over a voice grade telephone line with a bandwidth of 4 KHz and $(S/N) = 30$ dB. Assume that the line printer needs 8 bits of data per character and prints out 80 characters per line. Would you believe him?

Sol :- Given $B = 4$ KHz //

$10 \log_{10} S/N = (30)$ dB

$\log_{10}(S/N) = 3$

$S/N = 10^3 = 1000$

∴ Channel Capacity $= B \log(1 + S/N)$

$= 4000 \log(1 + 1000)$

$C = 26632.85$ bits/sec //

given $r_s = 30$ lines/minute

$= \frac{30}{60} = 0.5$ lines/sec //

∴ Rate of Transmission R_t = 1 Hz.

R_t = (8 bits/character) (80 characters/line) (0.5 lines/sec)

R_t = 320 bits/sec //.

$R_t \ll C$

---

* An analog signal having bandwidth of 5KHz is sampled at twice the Nyquist rate with each sample quantized into one of 128 equally likely levels.

(a). What is the Information rate of this source?

(b). Can the output of this source be transmitted without error over an AWGN channel with bandwidth of 12KHz and SNR of 22dB?

(c) Find SNR required for error free transmission for part (a).

(d) Find bandwidth required for an AWGN channel for error-free transmission of this source if SNR happens to be 22dB.

Sol :- B = 5KHz //

$f_s = 2f_m = 2(2B) = 2(2 \times 5k)$ //.

f_s = 20K Samples/sec //.

∴ no of quantization levels = 128

(a). Information rate $R = rH$.

r = number of Samples/sec = 20 K-Samples/sec.

$H = \log_2(q) = \log_2(128) = 7$ bits/sample //

$R = (20\text{K-Samples/sec})(7 \text{ bits/sample})$

$R = 140 \cdot \text{KBPS}$ //

(b). $B_1 = 12\text{KHz}$ // $SNR_1 = 22\text{dB}$

$$10\log_{10}\left(S_1/N_1\right) = \left(S_1/N_1\right)\text{dB}$$

$$\log_{10}\left(S_1/N_1\right) = \frac{22}{10} = 2.2.$$

$$S_1/N_1 = 10^{2.2} = 158.489 //$$

$$C_1 = B_1 \log\left(1 + S_1/N_1\right) = 12\text{K} \log\left(1 + 158.489\right)$$

$$= 87808 \text{ bits/sec.}$$

$$= 87.808 \text{ KBPS} //$$

(c). SNR for error-free transmission is

$C_2 = R = 140\text{KBPS}$ $\quad B_2 = 12\text{KHz}$

$$140\text{KBPS} = B_2 \log_2\left(1 + S_2/N_2\right)$$

$$140\text{K} = 12\text{K} \cdot \log_2\left(1 + S_2/N_2\right)$$

$$11.666 = \log_2\left(1 + S_2/N_2\right)$$

$$2^{11.666} = 1 + S_2/N_2$$

$$3250.49 - 1 = S_2/N_2.$$

$$3250 = S_2/N_2$$ //

d) $SNR_3 = 22\,dB \quad \left(\frac{S_1}{N_1}\right)_{dB} = 10\log_{10}\left(S_1/N_1\right)$

$$22.db = 10\log_{10}\left(S_1/N_1\right)$$

$$2.2 = \log_{10}\left(S_1/N_1\right)$$

$$10^{2.2} = S_1/N_1$$

$$158.489 = S_3/N_1 //$$

$$C_3 = R = 140\,KBps = B_3 \log_2\left(1 + S_1/N_1\right)$$

$$B_3 = \frac{140 \times 10^3}{\log_2(1+158.489)}$$

$$= 19.133\,kHz //$$

* A Communication Channel has a bandwidth of 6KHz and a Signal to noise ratio of 63.

(a) Determine the Capacity of the Channel.

(b) Determine the new bandwidth needed if the Signal-to-noise ratio is reduced to 31 keeping the channel Capacity same as in (a)

(c) Determine the factor by which the Signal power has to be increased if the bandwidth is reduced to 3.6KHz from 6KHz //

Sol: $B_1 = 6KHz //$ $\quad \frac{S_1}{N_1} = 63 //$

(a) Channel Capacity $C_1 = B_1 \log_2(1 + S_1/N_1)$

$= 6\times10^3 \log_2(1+63)$

$= 36\times10^3$ bps //

b) $C_2 = C_1 = 36\times10^3$ bps //

$\frac{S_2}{N_2} = 31$ //

$C_2 = B_2 \log\left(1 + \frac{S_2}{N_2}\right)$

$B_2 = \frac{C_2}{\log\left(1 + \frac{S_2}{N_2}\right)} = \frac{36\times10^3 \text{ bps}}{\log_2(1+31)} = 7.2\text{KHz}$ //

c) given. $B_3 = 3.6\text{KHz}$ $C_3 = C_2 = C_1 = 36\times10^3$ bps.

$C_3 = B_3 \log_2\left(1 + S_3/N_3\right)$.

$36\times10^3 = 3.6\text{K}. \log_2\left(1 + S_3/N_3\right)$

$10 = \log_2\left(1 + S_3/N_3\right)$

$2^{10} = 1 + S_3/N_3$.

$1024 - 1 = S_3/N_3$

$S_3/N_3 = 1023$ //

$\therefore S_3 = 1023\, N_3$ //

given $S_1/N_1 = 63$

$S_1 = 63 N_1$ //

$\frac{S_3}{S_1} = \frac{1023 N_3}{63 N_1} = \frac{1023\,\eta_0 B_3}{63\,\eta_0 B_1} = \frac{1023\times3.6\text{K}}{63\times6\text{K}} = 9.743$ //

SYMMETRIC / UNIFORM CHANNEL:

A Channel is said to be Symmetric (or) uniform Channel if the second and subsequent rows of the Channel matrix contain the same elements as that of 1st row but in a different order.

CHANNEL CAPACITY OF SYMMETRIC / UNIFORM CHANNEL:

The Channel matrix of a Symmetric / uniform Channel can be written as [for 'r' input symbols and "s" output symbols]

$$P(B/A) = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_r \end{array} \begin{array}{c} \begin{array}{ccccc} b_1 & b_2 & b_3 & \cdots\cdots & b_s \end{array} \\ \begin{bmatrix} P_1 & P_2 & P_3 & \cdots\cdots & P_s \\ P_2 & P_3 & P_4 & \cdots\cdots & P_1 \\ P_3 & P_4 & P_5 & \cdots\cdots & P_2 \\ \vdots & \vdots & \vdots & & \vdots \\ P_r & P_{r+1} & P_{r+2} & \cdots\cdots & P_{r-1} \end{bmatrix} \end{array}$$

where $P_1, P_2, P_3, P_4 \cdots\cdots P_{s-4}, P_{s-3}, P_{s-2}, P_{s-1}, P_s$ are the conditional probabilities $P(b_j/a_i)$ whose permutations and combinations are present in other rows and. The sum of all the elements of any row equal to one is given by $\sum_{j=1}^{s} P(b_j/a_i) = 1$.

But all the "n" number of rows have the same elements. $P_1, P_2, P_3, P_4 \cdots P_{s-4}, P_{s-3}, P_{s-2}, P_{s-1}, P_s$ Hence, the above equation becomes.

$$\sum_{i=1}^{s} P_i = 1.$$

The equation $H(B/A)$ is given by

$$H(B/A) = \sum_{i=1}^{n} \sum_{j=1}^{s} p(a_i, b_j) \log \frac{1}{p(b_j/a_i)}$$

$$= -\sum_{i=1}^{n} \sum_{j=1}^{s} p(a_i)\, p(b_j/a_i) \log \frac{1}{p(b_j/a_i)}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{s} p(a_i)\, P_j \log \frac{1}{P_j}$$

$$H(B/A) = \left[\sum_{i=1}^{n} p(a_i)\right] \left[\sum_{j=1}^{s} P_j \log \frac{1}{P_j}\right]$$

Since $\sum_{i=1}^{n} p(a_i) = 1$ and.

let $\sum_{j=1}^{s} P_j \log \frac{1}{P_j} = h$ = a constant which remains the same for all rows since all the rows constant. of same elements as that of the 1st row.

$\therefore H(B/A) = [1][h]$

$$H(B/A) = h = \sum_{j=1}^{s} P_j \log \frac{1}{P_j}$$

The mutual Information $I(A, B)$ is given by.

$$I(A, B) = H(B) - H(B/A)$$

$$= H(B) - h \;//$$

The Channel Capacity as defined by

$$C = \max [I(A,B)] \, r_s$$

r_s = 1 message-Symbol/sec

$$C = \max [I(A,B)]$$

$$= \max [H(B) - h] \quad \text{--- (1)}$$

$$C = \max [H(B)] - h$$ Since "h" is a Constant //

But H(B), the Entropy of the output Symbols becomes maximum if and only when all the Received Symbols, become equiprobable and since there are 'S' number of output Symbols, we have.

$$H(B)_{max} = \max [H(B)] = \log S \quad \text{--- (2)}$$

Eq (2) Sub into eq (1).

$$C = \log S - h \; //$$

* for. The Channel matrix shown. find the Channel capacity.

$$P(b_j/a_i) = \begin{matrix} \\ a_1 \\ a_2 \\ a_3 \end{matrix} \begin{matrix} b_1 & b_2 & b_3 \\ \end{matrix} \begin{bmatrix} 1/2 & 1/3 & 1/6 \\ 1/3 & 1/6 & 1/2 \\ 1/6 & 1/2 & 1/3 \end{bmatrix}$$

Sol:-

The Equivocation 'h' is given by -

$$h = \sum_{i=1}^{3} P_i \log \frac{1}{P_i}$$

$$= 1/2 \log(2) + 1/3 \log(3) + 1/6 \log 6$$

$$= 1.4591 \text{ bits/msg Symbol}$$

∴ The Channel Capacity is given by (total number of Symbols)

Assume $r_s = 1$ msg-symbol/sec, $S = 3$

$$C = (\log_2 S - h)\, r_s \text{ bits/sec} = \log(3) - 1.4591$$

$$= 0.1258 \text{ bits/sec}$$

* for The Channel matrix given below, compute the Channel Capacity.

$$P(Y/X) = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.2 & 0.6 & 0.2 \\ 0.2 & 0.2 & 0.6 \end{bmatrix} \text{ with } r_s = 1000 \text{ Symbol/sec.}$$

Jan/Feb-2006 [4M], July-2013 (6M), Dec-2010 (6M)

Sol:- $h = \sum_{i=1}^{3} P_i \log \frac{1}{P_i} = 3 \times 0.2 \log \frac{1}{0.2} + 0.6 \log \frac{1}{0.6}$

$$= 1.371 \text{ bits/msg-Symbol}$$

$\therefore$ Channel Capacity $C = (\log S - h)$ bits/sec.

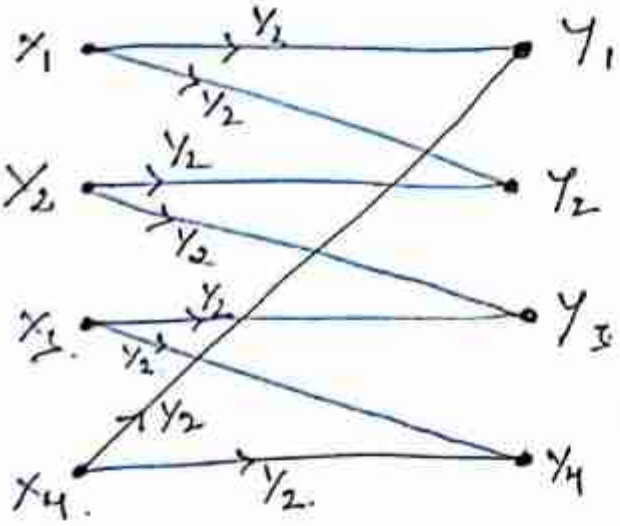
$= (\log(3) - 1.371)(1000)$ bits/sec

$= 2114$ bits/sec //

Note :-

Given S = number of output symbols $= 3$ //

* Determine the Capacity of the channel shown in below fig.



Sol :- The channel matrix $P(Y/X)$ can be written straightaway by looking at the channel diagram.

$$P(Y/X) = \begin{array}{c} \\ X_1 \\ X_2 \\ X_3 \\ X_4 \end{array} \begin{array}{c} \begin{array}{cccc} Y_1 & Y_2 & Y_3 & Y_4 \end{array} \\ \begin{bmatrix} 1/2 & 1/2 & 0 & 0 \\ 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 1/2 & 1/2 \\ 1/2 & 0 & 0 & 1/2 \end{bmatrix} \end{array}$$

" By looking at the Channel matrix, we can conclude that the given Channel is a Symmetric/uniform Channel since the 2nd, 3rd and Hth row of the matrix consists of the same elements as the 1st row elements.

$$\bar{h} = H(Y/X) = \sum_{i=1}^{4} p_i \log \frac{1}{p_i}$$

$$= \tfrac{1}{2}\log(2) + \tfrac{1}{4}\log(4) + 0 + 0 = 1 \text{ bit/msg symbol}$$

where r_s = 1 msg symbol/sec

$\therefore C = (\log_2 s - \bar{h})\, r_s$. Since there are 4 columns in the channel matrix $s = 4$

$= \log_2(4) - 1$

$= 2 - 1 = 1$ bit/msg Symbol //

* A Channel has the following Characteristics.

$$P(Y/X) = \begin{array}{c} \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \end{array} \\ \begin{bmatrix} 1/3 & 1/3 & 1/6 & 1/6 \\ 1/6 & 1/6 & 1/3 & 1/3 \end{bmatrix} \end{array}$$

Find $H(X)$, $H(Y)$, $H(X,Y)$ and Channel Capacity if r_s = 1000 Symbols/sec

July/Aug-2002 [6M].

Sol: Let us assume that the input probabilities.

$$p(x_1) = p(x_2) = 1/2.$$

$$P(Y/X) = \begin{array}{c} \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \end{array} \\ \begin{bmatrix} 1/3 & 1/3 & 1/6 & 1/6 \\ 1/6 & 1/6 & 1/3 & 1/3 \end{bmatrix} \end{array}$$

← NOTE: First Convert $P(Y/X)$ into $P(X,Y)$.

formula :- $P(x_i, y_i) = P(y_i/x_i)\, P(x_i)$

$P(x_1, y_1) = P(y_1/x_1) P(x_1) = (1/3)(1/2) = 1/6$ //

$P(x_1,y_2) = P(y_2/x_1)\,P(x_1) = 1/3 \times 1/2 = 1/6$ //

$P(x_1,y_3) = P(y_3/x_1)\,P(x_1) = 1/6 \times 1/2 = 1/12$ //

$P(x_1,y_4) = P(y_4/x_1)\,P(x_1) = 1/6 \times 1/2 = 1/12$ //

$P(x_2,y_1) = P(y_1/x_2)\,P(x_2) = 1/6 \times 1/2 = 1/12$ //

$P(x_2,y_2) = P(y_2/x_2)\,P(x_2) = 1/6 \times 1/2 = 1/12$ //

$P(x_2,y_3) = P(y_3/x_2)\,P(x_2) = 1/3 \times 1/2 = 1/6$ //

$P(x_2,y_4) = P(y_4/x_2)\,P(x_2) = 1/3 \times 1/2 = 1/6$ //

$P(y_1)$:- The Jpm can be constructed below

$$P(X,Y) = \begin{matrix} & y_1 & y_2 & y_3 & y_4 \\ x_1 & 1/6 & 1/6 & 1/12 & 1/12 \\ x_2 & 1/12 & 1/12 & 1/6 & 1/6 \end{matrix}$$

The o/p probabilities are equal i.e. given by.

$P(y_1) = 1/6 + 1/12 = 1/4$ // $P(y_2) = 1/6 + 1/12 = 1/4$ //

$P(y_3) = 1/12 + 1/6 = 1/4$ // $P(y_4) = 1/12 + 1/6 = 1/4$ //

* $H(x)$ is given by

$$H(x) = \sum_{i=1}^{2} P(x_i) \log \frac{1}{P(x_i)} = \sum_{i=1}^{2} P(x_i) \log \frac{1}{P(x_i)}.$$

$= 1/2 \log(2) + 1/2 \log(2) = 1$ bits/msg-symbol //

* $H(y)$ is given by

$$H(y) = \sum_{i=1}^{n} P(y_i) \log \frac{1}{P(y_i)} = \frac{1}{4}\log(4) + \frac{1}{4}\log(4) + \frac{1}{4}\log(4) + \frac{1}{4}\log(4)$$

$= 2$ bits/msg-symbol //

* The joint entropy $H(X,Y)$ is given by

$$H(X,Y) = \sum_{i=1}^{m} \sum_{j=1}^{n} P(x_i, y_j) \log \frac{1}{P(x_i, y_j)}$$

$$= \sum_{i=1}^{2} \sum_{j=1}^{4} P(x_i y_j) \log \frac{1}{P(x_i, y_j)}$$

$$= (1/6 \log 6)(4) + (1/12 \log 12)(4) = 2.9183 \text{ bits/m-symbol}$$

* The i/p probabilities are assumed to be equal. The mutual Information maximizes and becomes the Channel Capacity is given by.

$$C = (\log_2 s - h) \times r_s \text{ bits/sec.}$$

Let $\log_2 s = \max H(y)$

$$C = [\max H(y) - h] \times r_s \text{ bits/sec.}$$

Since 'h' is Constant

$$\therefore C = [\max [H(y) - h]] \times r_s \text{ bits/sec}$$

Where $H(y) - h = I(x,y)$

$$C = [I(x,y)] \times r_s \text{ bits/sec.}$$

when $I(x,y) = H(x) + H(y) - H(x,y)$

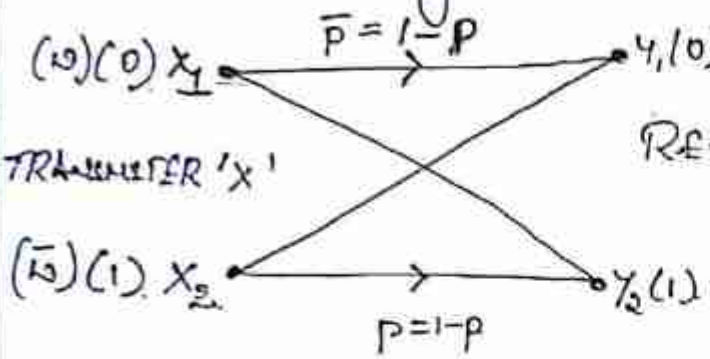
$$I(x,y) = 1 + 2 - 2.9183 = 0.0817 \text{ bits/m-sym}$$

$$\therefore C = [0.0817] \times 1000 \text{ bits/sec.}$$

$$= 81.7 \text{ bits/sec} //$$

BINARY SYMMETRIC CHANNELS (BSC)

BSC is one of the most Commonly used and Leadely used Channels whose Channel diagram is shown in below fig.



Let $P(X_1) = p$ and $P(X_2) = 1-p = \bar{p}$ (say)

p = probability of error.

= probability of Reception of "1" when "0" is Transmitted

= probability of Reception of '0' when "1" is Transmitted

In a BSC the Symbol X_1 is Encoded as "0" and X_2 as "1".

$$\therefore P(Y/X) = \begin{matrix} X_1 \\ X_2 \end{matrix} \begin{bmatrix} P(Y_1/X_1) & P(Y_2/X_1) \\ P(Y_1/X_2) & P(Y_2/X_2) \end{bmatrix} = \begin{bmatrix} \bar{p} & p \\ p & \bar{p} \end{bmatrix} \quad \text{---(1)}$$

$\therefore$ eq (1) is Symmetric matrix hence the Channel is a Symmetric Channel.

* Symmetric/uniform Channel. The Equivocation is given by.

$$H(Y/X) = h = \sum_{i=1}^{2} P_i \log \frac{1}{P_i}$$

$$h = \bar{p} \log \frac{1}{\bar{p}} + p \log \frac{1}{p} \quad \text{---(2)}$$

To find $H(Y)$:-

$$H(Y) = \sum_{i=1}^{2} p(y_i) \log \frac{1}{p(y_i)} = \sum_{i=1}^{2} p(y_i) \log \frac{1}{p(y_i)}$$

$$= p(y_1) \log \frac{1}{p(y_1)} + p(y_2) \log \frac{1}{p(y_2)} \quad \text{— (3)}$$

$\therefore p(y_1)$ & $p(y_2)$ is given by.

$$p(y_1) = p(y_1/x_1)\, p(x_1) + p(y_2/x_1)\, p(x_2) = \bar{p}w + p\bar{w} \quad \text{— (4)}$$

$$p(y_2) = p(y_2/x_1)\, p(x_1) + p(y_2/x_2)\, p(x_2) = pw + \bar{p}\bar{w} \quad \text{— (5)}$$

eq (4) and (5) Sub into eq (3)

$$H(Y) = (\bar{p}w + p\bar{w}) \log \frac{1}{(\bar{p}w + p\bar{w})} + (pw + \bar{p}\bar{w}) \log \frac{1}{(\bar{p}w + \bar{p}\bar{w})} \quad \text{— (6)}$$

* The mutual Information of a BSC is given by.

$$I(Y,X) = I(X,Y) = H(Y) - H(Y/X)$$

$$I(X,Y) = \left[(\bar{p}w + p\bar{w}) \log \frac{1}{(\bar{p}w + p\bar{w})} + (pw + \bar{p}\bar{w}) \log \frac{1}{(\bar{p}w + \bar{p}\bar{w})} \right]$$

$$- \left[\bar{p} \log \frac{1}{\bar{p}} + p \log \frac{1}{p} \right]$$

using eq. (6) and (2)

* In BSC if a Symmetric Channel, the Channel Capacity with $r_s = 1$ message-symbol/sec is found by using eq as.

$$C = \log S - h \text{ bits/sec}$$

$$= \log(2) - h = \frac{\log(2)}{\log(2)} - h = 1 - h \quad \text{— (7)}$$

//

(6.7)

$$C = 1 - \left[\bar{p}\log\frac{1}{\bar{p}} + p\log\frac{1}{p}\right] \text{ bits/sec} \quad \text{---(7)}$$

Consider a. eq no (6.)

$$H(y) = (\bar{p}w + p\bar{w})\log\frac{1}{(\bar{p}w + p\bar{w})} + (pw + \bar{p}\bar{w})\log\frac{1}{(pw + \bar{p}\bar{w})}$$

where $w = \bar{w} = 1/2$

$$= \tfrac{1}{2}(p+\bar{p})\log\frac{1}{\frac{1}{2}(p+\bar{p})} + \tfrac{1}{2}(p+\bar{p})\log\frac{1}{\frac{1}{2}(p+\bar{p})}$$

$= \frac{1}{2}\log(2) + \frac{1}{2}\log(2)$ where $p+\bar{p} = 1$

$H(y) = 1$ //

∴ The mutual Information $I(x,y)$ becomes.

$$I(x,y) = H(y) - H(y/x)$$

$$= 1 - H(y/x)$$

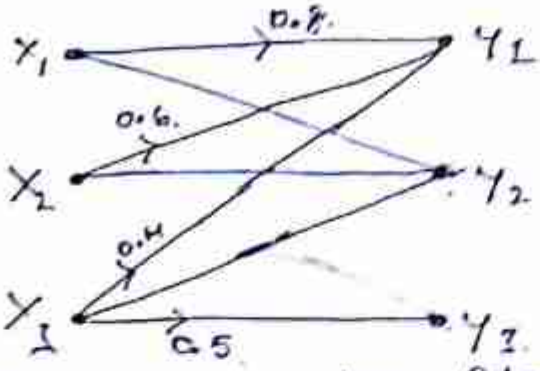
Sin. $H(y/x) = h$

$$I(x,y) = 1 - h$$

$$= 1 - \left[\bar{p}\log\frac{1}{\bar{p}} + p\log\frac{1}{p}\right] \quad \text{---(9.)}$$

∴ Eq. (7) and (8) are Identical //

* For the channel shown in below fig. write the channel matrix.



Sol:- For the Communication channel, the sum of all outgoing probabilities from any i/p must be 1. The missing probabilities can be calculated as given.

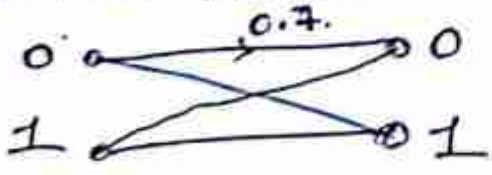
$$P\left(\frac{y_2}{x_1}\right) = 1-0.8 = 0.2 \quad P\left(\frac{y_1}{x_2}\right) = 1-0.6 = 0.4$$

$$P\left(\frac{y_2}{x_3}\right) = 1-(0.4+0.5) = 0.1$$

The channel matrix is given by.

$$P(y/x) = \begin{matrix} \\ x_1 \\ x_2 \\ x_3 \end{matrix} \begin{matrix} \begin{matrix} y_1 & y_2 & y_3 \end{matrix} \\ \begin{bmatrix} 0.8 & 0.2 & 0 \\ 0.6 & 0.4 & 0 \\ 0.4 & 0.1 & 0.5 \end{bmatrix} \end{matrix}$$

* For the BSC shown in below fig. write the channel matrix



Sol:- The channel of BSC $P(y/x) = P(y_1)$.

$\therefore P(y_1) = 0.7$ //

$P(1/0) = 1 - 0.7 = 0.3$ //

$P(0/1) = 1 - 0.7 = 0.3$ //

Channel matrix $= P(Y/X) =$

	0	1
0	0.7	0.3
1	0.3	0.7

1st * For the channel matrix given, find the missing entries and also draw the corresponding channel diagram.

Channel matrix $= P(Y/X) =$

	y_1	y_2	y_3
x_1	0.8	—	0.2
x_2	—	0.6	0.2
x_3	0.2	0.3	—

Sol: The sum of all the elements of a row of a channel matrix is equal to unity.

$\therefore P(y_1/x_1) + P(y_2/x_1) + P(y_3/x_1) = 1$

$P(y_2/x_1) = 1 - P(y_1/x_1) - P(y_3/x_1)$

$= 1 - 0.8 - 0.2 = 0$ //

$P(y_1/x_2) = 1 - P(y_2/x_2) - P(y_3/x_2) = 1 - 0.6 - 0.2 = 0.2$ //

$P(y_3/x_3) = 1 - P(y_1/x_3) - P(y_2/x_3) = 1 - 0.2 - 0.3 = 0.5$ //

$\therefore P(Y/X) =$

	y_1	y_2	y_3
x_1	0.8	0	0.2
x_2	0.2	0.6	0.2
x_3	0.2	0.3	0.5

//

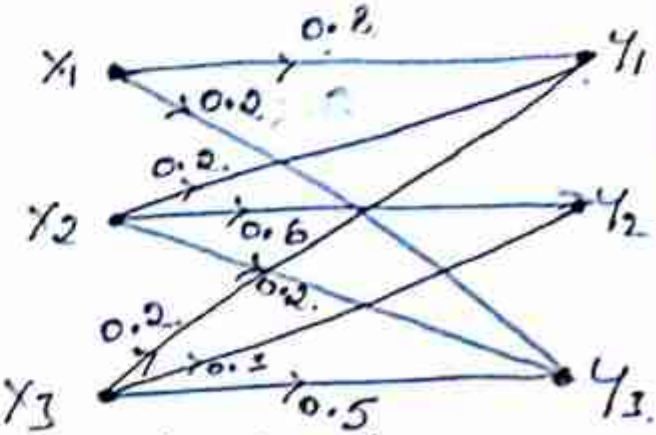


Fig:- Channel diagram.

* A binary Symmetric Channel has the following noise matrix with Source probabilities of $P(x_1) = 2/3$ and $P(x_2) = 1/3$

$$P(Y/X) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} \begin{matrix} y_1 & y_2 \end{matrix} \\ \begin{bmatrix} 2/4 & 2/4 \\ 1/4 & 3/4 \end{bmatrix} \end{matrix}$$

1) Determine $H(X)$, $H(Y)$, $H(X,Y)$, $H(Y/X)$, $H(X/Y)$ and $I(X,Y)$
2) Find the Channel Capacity "C"
3) Find the Channel efficiency and Redundancy.

Jan-2015[10M], Jan-2014[10M], Jan-2012[8M]

Sol:

$$P(Y/X) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} \begin{matrix} y_1 & y_2 \end{matrix} \\ \begin{bmatrix} 1/4 & 3/4 \\ 3/4 & 1/4 \end{bmatrix} \end{matrix}$$

note

$$\begin{bmatrix} 1/4 & 3/4 \\ 3/4 & 1/4 \end{bmatrix} = \begin{bmatrix} P(y_1/x_1) & P(y_2/x_1) \\ P(y_1/x_2) & P(y_2/x_2) \end{bmatrix}$$

1) H(X) is given by.

$$H(X) = \sum_{i=1}^{m} P(x_i) \log \frac{1}{P(x_i)} = P(x_1) \log \frac{1}{P(x_1)} + P(x_2) \log \frac{1}{P(x_2)}$$

$$= 2/3 \log 3/2 + 1/3 \log(3) = 0.9182 \text{ bits/message symbol}$$

* H(Y) is given by.

$$H(Y) = \sum_{i=1}^{n} P(y_i) \log \frac{1}{P(y_i)} = P(y_1) \log \frac{1}{P(y_1)} + P(y_2) \log \frac{1}{P(y_2)}$$

$$\therefore \begin{bmatrix} P(y_1) \\ P(y_2) \end{bmatrix} = \begin{bmatrix} P(x_1), P(x_2) \end{bmatrix} \begin{bmatrix} P(y_1/x_1) & P(y_2/x_1) \\ P(y_1/x_2) & P(y_2/x_2) \end{bmatrix}$$

$$= \begin{bmatrix} P(x_1) P(y_1/x_1) + P(x_2) P(y_1/x_2) \\ P(x_1) P(y_2/x_1) + P(x_2) P(y_2/x_2) \end{bmatrix}$$

$$\begin{bmatrix} P(y_1) \\ P(y_2) \end{bmatrix} = \begin{bmatrix} (2/3)(3/4) + (1/3)(1/4) \\ (2/3)(1/4) + (1/3)(3/4) \end{bmatrix} = \begin{bmatrix} 7/12 \\ 5/12 \end{bmatrix}$$

$\therefore P(y_1) = 7/12$ // $P(y_2) = 5/12$ //

$$H(Y) = P(Y_1)\log\frac{1}{P(Y_1)} + P(Y_2)\log\frac{1}{P(Y_2)}$$

$$= \frac{7}{12}\log\frac{12}{7} + \frac{5}{12}\log\frac{12}{5} = 0.9799 \text{ bits/msg-symbol}$$

$$* \quad H(Y/X) = \sum_{i=1}^{2}\sum_{j=1}^{3} P(X_i Y_j)\log\frac{1}{P(Y_j/X_i)} \text{ bits/msg symbol}$$

$$\therefore P(Y_j/X_i) = \frac{P(X_i, Y_j)}{P(X_i)}$$

$$P(X_i Y_j) = P(Y_j/X_i)\,P(X_i)$$

$$P(X,Y) = \begin{matrix} & Y_1 & Y_2 \\ X_1 & 1/2 & 1/6 \\ X_2 & 1/12 & 1/4 \end{matrix}$$

$$P(X_1Y_1) = P(Y_1/X_1)\,P(X_1) = (3/4)(2/3) = 1/2$$

$$P(X_1Y_2) = P(X_1)\,P(Y_2/X_1) = (2/3)(1/4) = 1/6$$

$$P(X_2Y_1) = P(Y_2)\,P(Y_1/X_2) = (1/3)(1/4) = 1/12$$

$$P(X_2Y_2) = P(X_2)\,P(Y_2/X_2) = (1/3)(3/4) = 1/4$$

$$H(Y/X) = P(X_1Y_1)\log\frac{1}{P(Y_1/X_1)} + P(X_1Y_2)\log\frac{1}{P(Y_2/X_1)} + P(X_2Y_1)\log\frac{1}{P(Y_1/X_2)}$$

$$+ P(X_2Y_2)\log\frac{1}{P(X_2/Y_2)}$$

$$= \frac{1}{2}\log\frac{4}{3} + \frac{1}{6}\log(4) + 1/12\log(4) + 1/4\log(4/3)$$

$$= 1/2 \times 0.4150 + 1/6(2) + (1/12)(2) + 1/4 \times 0.4150$$

$$= 0.8113 \text{ bits/msg. symbol}$$

* The joint entropy $H(X,Y)$ is given by, bits/msg symbol

$$H(X,Y) = \sum_{i=1}^{2}\sum_{j=1}^{3} P(X_i Y_j)\log\frac{1}{P(X_i Y_j)}$$

$$= 1/2\log(2) + 1/6\log(6) + 1/12\log(12) + 1/4\log(4)$$

$$= 1.7296 \text{ bits/msg symbol}$$

(or) $H(X,Y) = H(X) + H(Y/X)$

$= 0.9183 + 0.8113$

$= 1.7296$ bits/msg symbol //

* $H(X/Y)$ is given by.

* $H(X/Y) = \sum_{i=1}^{m} \sum_{j=1}^{n} P(x_i, y_j) \log \frac{1}{P(x_i/y_j)}$ bits/msg symbol

$\therefore P(x_i/y_j) = \frac{P(x_i, y_j)}{P(y_j)}$

(or)

$H(X/Y) = H(X,Y) - H(Y)$

$= 1.7296 - 0.9799 = 0.7497$ bits/msg symbo

* $I(X,Y)$ is given by

$I(X,Y) = H(Y) - H(Y/X)$ (or) $I(X,Y) = H(X) - H(X/Y)$

$= 0.9799 - 0.8113$ $= 0.9183 - 0.7497$

$= 0.1686$ bits/msg symbo $= 0.1686$ bits/msg symbo

2) The Channel Capacity "C" is given by.

"C" $= (\log_2 S - h)\, r_s$ bits/msg symbol. NOTE: $\rightarrow (4.142)$

where S = number of o/p symbols $= 2$ //

assume $r_s = 1$ msg-symbol/sec.

"C" $= (\log_2(2) - h) \times 1$

"C" $= 1 - h$ NOTE: $\therefore h = H(Y/X)$

$= 1 - H(Y/X)$

$C = 1 - 0.8113 = 0.1887$ bits/msg-symbol //

$\therefore$ Channel efficiency $= \dfrac{I(X,Y)}{C}$

$$\eta_{ch} = \frac{0.1686}{0.1887} = 89.35\% \text{ //}$$

Channel Redundancy $R_{\eta ch} = (1 - \eta_{ch}) \times 100\%$

$= (1 - 0.8935) \times 100\%$

$= 0.1065 \times 100\%$

$= 10.65\%$ //

* A message source produces 2 independent symbols "A" and "B" with probabilities $P(A) = 0.4$ and $P(B) = 0.6$. Calculate the efficiency of the source and hence its redundancy. If the symbols are received as with "A" in every 1000 symbols as error. Calculate the transmission rate of the system.

Sol:- It is given that the transmitter "X" produces two symbols "A" and "B" with probabilities 0.4 and 0.6 respectively.

The entropy of the source "X" is given by

$$H(X) = P(A) \log \frac{1}{P(A)} + P(B) \log \frac{1}{P(B)}$$

$$= 0.4 \log \frac{1}{0.4} + 0.6 \log \frac{1}{0.6}$$

$$= 0.97095 \text{ bits/message-symbol}$$

$$H(X)_{max} = \log_2(2) = \log_2(2) = 1 \text{ bit/message-symbol}$$ //

* Source efficiency $\eta_x = \frac{H(x)}{H(x)_{max}} = \frac{0.97105}{1} = 97.095\%$ //

* Source Redundancy $R_{\eta_x} = (1-\eta_x) \times 100\%$
$= 2.905\%$ //

If 'p' is given that p = probability of error.
$p = \frac{4}{100} = 0.04$ //

and
$\bar{p}$ (or) $1-p = 1-0.04$
$= 0.96$ //

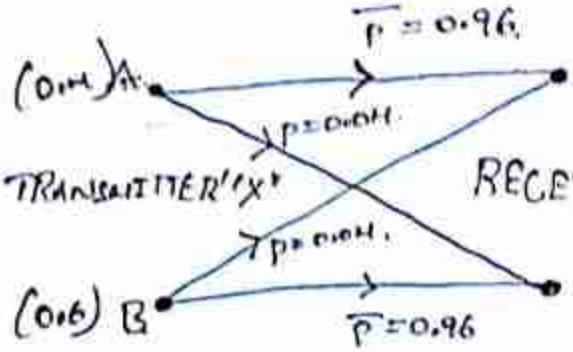


Fig: Channel diagram.

Given. $m = p(A) = 0.4$ // and. $\bar{m} = p(B) = 0.6$ //

* The mutual Information $I(X,Y)$ of a BSC is given by

$$I(X,Y) = \left[(\bar{p}m + p\bar{m}) \log \frac{1}{(\bar{p}m + p\bar{m})} + (pm + \bar{p}\bar{m})\right] - \left[\bar{p} \log \frac{1}{\bar{p}} + p \log \frac{1}{p}\right]$$

$(\bar{p}m + p\bar{m}) = (0.96)(0.4) + (0.04)(0.6) = 0.408$ //

$(\bar{p}m + p\bar{m}) = (0.04)(0.4) + (0.96)(0.6) = 0.592$ //

$$I(X,Y) = \left[(0.408) \log \frac{1}{(0.408)} + (0.592) \log \frac{1}{(0.592)}\right] - \left[0.04 \log \frac{1}{0.04} + 0.96 \log \frac{1}{0.96}\right]$$

$I(X,Y) = 0.7331$ bits/m-Symbol //

The Transmission Rate R_t of the system is given by

$$R_t = I(X,Y)\, r_s$$

$$= (0.9333)(100)$$

$$R_t = 73.31 \text{ bits/sec.}$$

Where $r_s = 100$ Symbol/sec (assumed).

* A binary channel has the following Characteristics

$$P(Y/X) = \begin{bmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{bmatrix}$$

If input symbols are Transmitted with probs $3/4$ and $1/4$ Respectively, find Entropies $H(X)$, $H(X,Y)$ and $H(Y/X)$

Feb-2002 [5M].

Sol: $P[Y/X] = \begin{bmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{bmatrix}$

given $P[Y/X] = \begin{bmatrix} \bar{p} & p \\ p & \bar{p} \end{bmatrix}$

where $\bar{p} = 2/3$, $p = 1/3$ (Considering a first column only)

and $X_1 = p = 3/4$, $X_2 = \bar{p} = 1/4$.

* $H(X)$ is given by

$$H(X) = \sum_{i=1}^{2} P(x_i) \log \frac{1}{P(x_i)} = \sum_{i=1}^{2} P(x_i) \log \frac{1}{P(x_i)}$$

$$= p \log \frac{1}{p} + \bar{p} \log \frac{1}{\bar{p}}$$

$$H(X) = 3/4 \log 4/3 + 1/4 \log(4) = 0.8113 \text{ bits/M-Symbols}$$

* $H(Y/X)$ is given by

$$H(Y/X) = \sum_{i=1}^{m} \sum_{j=1}^{n} p(x_i, y_j) \log \frac{1}{p(y_j/x_i)}$$

(or)

$$H(Y/X) = h = \overline{p} \log \frac{1}{\overline{p}} + p \log \frac{1}{p}$$

$$= 2/3 \log 3/2 + 1/3 \log (3)$$

$$= 0.9183 \text{ bits/m-symbol}$$

The joint entropy $H(X,Y)$ is given by

$$H(X,Y) = H(X) + H(Y/X)$$

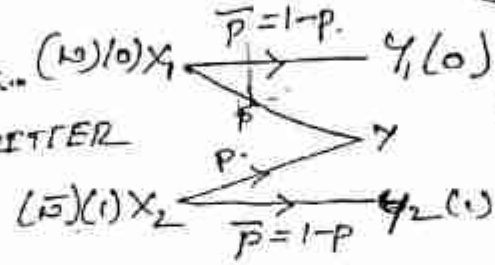
$$= 0.8113 + 0.9183 = 1.72.96 \text{ bits/m-symbol}$$

BINARY ERASURE CHANNEL (BEC) — JAN-2014 [10M]

BEC is one of the most Important Channel used in Digital Communication.

If a Symbol is Received in error, it is just marked as error and discarded & a Retransmission is requested from the Transmitter using a Reverse Channel, till a Correct Symbol is Received at the output. This ensures 100% Correct data Recovery. Since the error is totally erased in this Channel it is Called "Binary Erasure Channel".

Channel Diagram:



TRANSMITTER "X"

RECEIVER "Y"

Ex: So we have $p[Y/X] = p(y_i/x_i) =$

	Y_1	Y	Y_2
X_1	$\bar{p}$	p	0
X_2	0	p	$\bar{p}$

Let $p(x_1)=w$, $p(x_2)=\bar{w}$. Then $w+\bar{w}=1$ and $p+\bar{p}=1$

In order to Characterize the Channel find $H(Y/X)$

We know $H(Y/X) = h = \sum_{i=1}^{s} P_i \log \frac{1}{P_i}$

$$H(Y/X) = h = \bar{p}\log\frac{1}{\bar{p}} + p\log\frac{1}{p}$$

* $H(X) = \sum_{i=1}^{n} p(x_i)\log\frac{1}{p(x_i)}$

$$= \sum_{i=1}^{2} p(x_i)\log\frac{1}{p(x_i)} = w\log\frac{1}{w} + \bar{w}\log\frac{1}{\bar{w}}$$

* $P(x_i, y_j)$ is given by.

$$P(x_i, y_j) = P(x_i) P(y_j/x_i)$$

let $P(x_1) = w$. $P(x_2) = \bar{w}$

$$P(x, y) = P(x_i, y_j) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} \begin{matrix} y_1 & y & y_2 \end{matrix} \\ \begin{bmatrix} p\bar{w} & pw & 0 \\ 0 & p\bar{w} & \bar{p}\bar{w} \end{bmatrix} \end{matrix}$$

* The probabilities of o/p symbols is given by

$P(y_1) = \bar{p}w$ // $P(y) = pw + p\bar{w} = p(w + \bar{w}) = p$ //

$P(y_2) = \bar{p}\bar{w}$ //

* $P(x_i/y_j)$ is given by

$$P(x_i/y_j) = \frac{P(x_i, y_j)}{P(y_j)}$$

$$P(x/y) = P(x_i/y_j) = \begin{bmatrix} (p\bar{w})/(\bar{p}\bar{w}) & (pw/p) & 0 \\ 0 & (p\bar{w})/p & (\bar{p}\bar{w})/(\bar{p}\bar{w}) \end{bmatrix}$$

$$P(x/y) = P(x_i/y_j) = \begin{bmatrix} 1 & w & 0 \\ 0 & \bar{w} & 1 \end{bmatrix}$$ //

* The Equivocation $H(x/y)$ is given as.

$$H(x/y) = \sum_{i=1}^{2} \sum_{j=1}^{2} P(x_i y_j) \log \frac{1}{P(x_i/y_j)}$$

$$= \bar{p}w \log 1 + pw \log \frac{1}{w} + p\bar{w} \log \frac{1}{\bar{w}} + \bar{p}\bar{w} \log (1)$$

$$H(x/y) = p\left[w \log \frac{1}{w} + \bar{w} \log \frac{1}{\bar{w}}\right]$$

The mutual Information of BEC is given by.

$$I(X,Y) = H(X) - H(X/Y)$$

$$= p\log\frac{1}{p} + \bar{p}\log\frac{1}{\bar{p}} - p\left[p\log\frac{1}{p} + \bar{p}\log\frac{1}{\bar{p}}\right]$$

$$= \left[1-p\right]\left[p\log\frac{1}{p} + \bar{p}\log\frac{1}{\bar{p}}\right]$$

$$I(X,Y) = \bar{p}\,H(X)$$ //

The Channel Capacity of a BEC is given by

$$C = \max\left[I(X,Y)\right]$$

$$= \max\left[\bar{p}\,H(X)\right]$$

$$= \bar{p}\max\left[H(X)\right]$$

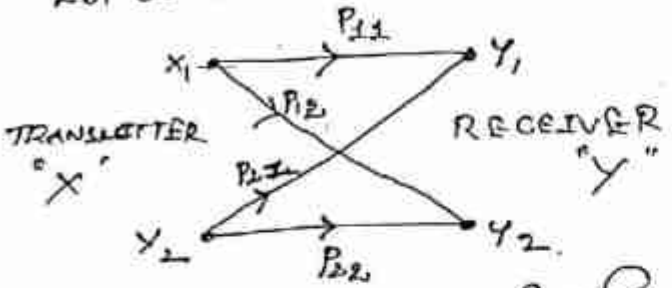
$$= \bar{p}\,H(X)_{max}$$

$$= \bar{p}\log 2. \quad \left[\text{since } H(x)_{max} = \log_2 = \log 2\right]$$

$$C = \bar{p}$$ //

ESTIMATION OF CHANNEL CAPACITY BY MUROGA'S METHOD :-

Let us Consider a binary channel as shown in below fig



The channel matrix of the binary channel shown is given by

$$P(y_j/x_i) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} \begin{matrix} y_1 & y_2 \end{matrix} \\ \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \end{matrix}$$

Two Quantities Q_1 and Q_2 are found out using the matrix Equation.

$$\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = \begin{bmatrix} P_{11}\log P_{11} + P_{12}\log P_{12} \\ P_{21}\log P_{21} + P_{22}\log P_{22} \end{bmatrix} \quad \text{— (1)}$$

Expanding we get

$$P_{11}Q_1 + P_{12}Q_2 = P_{11}\log P_{11} + P_{12}\log P_{12} \quad \text{— (2)}$$

$$P_{21}Q_1 + P_{22}Q_2 = P_{21}\log P_{21} + P_{22}\log P_{22}$$

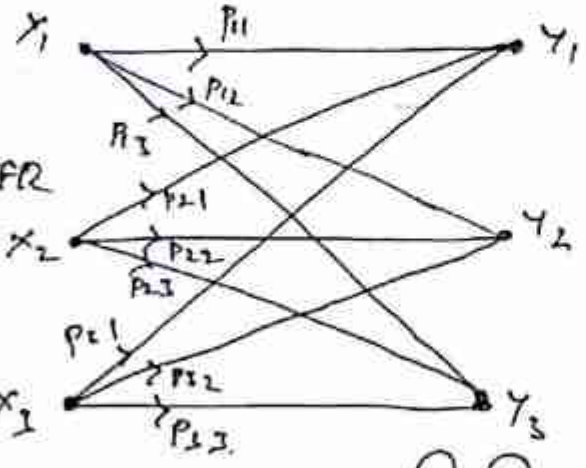
Eq (2) above are Simultaneous Equations which are Solved to get Q_1 and Q_2

The channel Capacity "C" of the binary channel is then calculated using the relationship.

$$C = \log\left[2^{Q_1} + 2^{Q_2}\right] \text{ bits/message-symbol} //$$

P.T.O.

This method can be extended to a channel having 3X3 channel matrix given in figure (below)



TRANSMITTER 'X'

RECEIVER 'Y'

$r_s = 10,000$ / sec.

The channel matrix of the channel of above fig is given by

$$P(Y_i/X_j) = \begin{matrix} & Y_1 & Y_2 & Y_3 \\ X_1 & P_{11} & P_{12} & P_{13} \\ X_2 & P_{21} & P_{22} & P_{23} \\ X_3 & P_{31} & P_{32} & P_{33} \end{matrix}$$

Three Quantities Q_1, Q_2 and Q_3 are found out using matrix equation

$$\begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \end{bmatrix} = \begin{bmatrix} P_{11}\log P_{11} + P_{12}\log P_{12} + P_{13}\log P_{13} \\ P_{21}\log P_{21} + P_{22}\log P_{22} + P_{23}\log P_{23} \\ P_{31}\log P_{31} + P_{32}\log P_{32} + P_{33}\log P_{33} \end{bmatrix}$$

The channel capacity "C" can then be found out by

$$C = \log\left[2^{Q_1} + 2^{Q_2} + 2^{Q_3}\right] \text{ bits/msg - symbol}$$

p.T.O.

The generalisation of the above method for a channel with m×n noise matrix is done as below:

$$P(Y_i/X_i) = \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1m} \\ P_{21} & P_{22} & \cdots & P_{2m} \\ \vdots & \vdots & & \vdots \\ P_{m1} & P_{m2} & \cdots & P_{mm} \end{bmatrix}$$

The Quantities $Q_1, Q_2, Q_3 \cdots Q_m$ are found out using.

$$\begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1m} \\ P_{21} & P_{22} & \cdots & P_{2m} \\ \vdots & & & \vdots \\ P_{m1} & P_{m2} & \cdots & P_{mm} \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_m \end{bmatrix} = - \begin{bmatrix} P_{11}\log P_{11} + P_{12}\log P_{12} \cdots\cdots + P_{1m}\log P_{1m} \\ P_{21}\log P_{21} + P_{22}\log P_{22} \cdots\cdots + P_{2m}\log P_{2m} \\ \vdots \\ P_{m1}\log P_{m1} + P_{m2}\log P_{m2} \cdots\cdots + P_{mm}\log P_{mm} \end{bmatrix}$$

Then the channel capacity 'C' is given by.

$$C = \log\left[2^{Q_1} + 2^{Q_2} + \cdots 2^{Q_m}\right] \text{ bits/msg-symbol}$$

1) Find the Capacity of the discrete Channel shown in below fig.

July/Aug-2002 [8M], Jan/Feb-2005. [6M], Dec-2012. [5M]

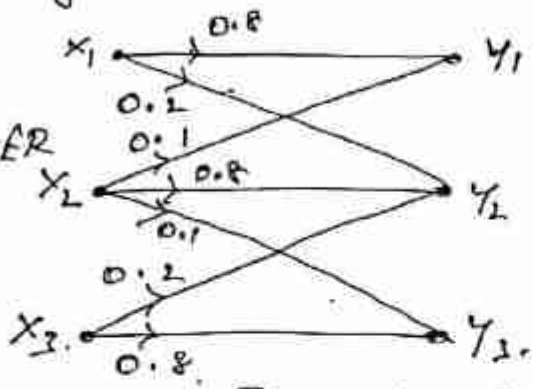


Fig: Channel diagram.

Sol:- The Channel matrix can be constructed as

$$P(Y_i/X_i) = \begin{array}{c} \\ X_1 \\ X_2 \\ X_3 \end{array} \begin{array}{c} \begin{array}{ccc} Y_1 & Y_2 & Y_3 \end{array} \\ \begin{bmatrix} 0.8 & 0.2 & 0.0 \\ 0.1 & 0.8 & 0.1 \\ 0.0 & 0.2 & 0.8 \end{bmatrix} \end{array}$$

Muroga's method. can be used for finding the Channel Capacity

$$\begin{bmatrix} 0.8 & 0.2 & 0.0 \\ 0.1 & 0.8 & 0.1 \\ 0.0 & 0.2 & 0.8 \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \end{bmatrix} = \begin{bmatrix} 0.8\log 0.8 + 0.2\log 0.2 + 0.0\log(0.0) \\ 0.1\log 0.1 + 0.8\log 0.8 + 0.1\log 0.1 \\ 0.0\log 0.0 + 0.2\log 0.2 + 0.8\log 0.8 \end{bmatrix}$$

$$0.8Q_1 + 0.2Q_2 = -0.722 \quad \text{---(1)}$$

$$0.1Q_1 + 0.8Q_2 + 0.1Q_3 = -0.922 \quad \text{---(2)}$$

$$0.2Q_2 + 0.8Q_3 = -0.722 \quad \text{---(3)}$$

ordering eq (1) throughout by 8 we get.

$$0.1Q_1 + 0.025Q_2 = -0.09025$$

$$0.1Q_1 = -0.025Q_2 - 0.09025 \quad \text{—(4)}$$

eq (4) sub into eq (2)

$$0.1Q_1 + 0.8Q_2 + 0.1Q_3 = -0.922.$$

$$-0.025Q_2 - 0.09025 + 0.8Q_2 + 0.1Q_3 = -0.922$$

$$0.775Q_2 + 0.1Q_3 = -0.83175 \quad \text{—(5)}$$

multiplying by 8 we get

$$6.2Q_2 + 0.8Q_3 = -6.654 \quad \text{—(6)}$$

Subst eq (3) from eq (6) we get.

$$0.2Q_2 + 0.8Q_3 = -0.722$$

$$6.2Q_2 + 0.8Q_3 = -6.654$$

$$6Q_2 = -5.932$$

$$Q_2 = 0.9887$$

Using the value of Q_2 in eq (6) we get

$$6.2Q_2 + 0.8Q_3 = -6.654$$

$$Q_3 = \frac{-6.65 - 6.2Q_2}{0.8}$$

$$= \frac{-6.65 - 6.2(0.9887)}{0.8}$$

$$Q_3 = -0.6553$$

using this value of Q_2 in eq (1)

$$0.8Q_1 + 0.2Q_2 = -0.722$$

$$Q_1 = \frac{-0.722 - 0.2Q_2}{0.8}$$

$$= \frac{-0.722 - 0.2(-0.9887)}{0.8}$$

$$= -0.6553$$

The Channel Capacity is given by

$$C = \log_2\left[2^{Q_1} + 2^{Q_2} + 2^{Q_3}\right]$$

$$= \log_2\left[2^{(-0.6553)} + 2^{(-0.9887)} + 2^{(-0.6553)}\right]$$

$$= 0.8269 \text{ bits/msg symbol}$$

The Channel Capacity in bits/sec is given by

$$C = (0.8269)m_s$$

$$= (0.8269)(10000)$$

$$= 8269 \text{ bits/sec}$$

* Find the mutual Infn and the Channel Capacity of the Channel shown in fig (below) given $p(x_1)=0.6$ and $p(x_2)=0.4$

Jan-2015 [10M]

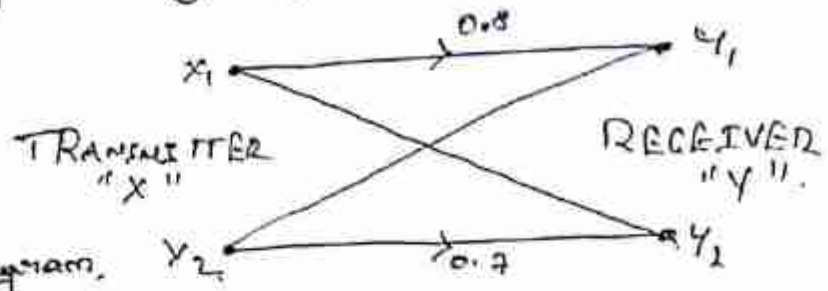


fig: Channel diagram.

Sol:- Channel matrix is constructed as below.

$$P(Y/X) = P(Y_j/X_i) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} y_1 & y_2 \\ \end{matrix} \begin{bmatrix} 0.8 & 0.2 \\ 0.1 & 0.9 \end{bmatrix}$$

∴ Since the channel is Non-Symmetric Channel.. the mutual Information $I(X,Y)$ is found using the formula

$$I(X,Y) = H(Y) - H(Y/X)$$

To find $H(Y/X)$:-

The Jpm is constructed by using $P(X_i,Y_i) = P(X_i)P(Y_j/X_i)$ which is given by.

$$P(X_i,Y_j) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} y_1 & y_2 \\ \end{matrix} \begin{bmatrix} 0.8\times0.6 & 0.2\times0.6 \\ 0.1\times0.4 & 0.9\times0.4 \end{bmatrix}$$

$$= \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} y_1 & y_2 \\ \end{matrix} \begin{bmatrix} 0.48 & 0.12 \\ 0.12 & 0.28 \end{bmatrix}$$ //

$$H(Y/X) = \sum_{i=1}^{2} \sum_{j=1}^{2} p(x_i, y_j) \log \frac{1}{p(y_j/x_i)}$$

$$= 0.48 \log \frac{1}{0.8} + 0.12 \log \frac{1}{0.3} + 0.12 \log \frac{1}{0.2} + 0.28 \log \frac{1}{0.7}$$

$$= 0.7857 \text{ bits/msg-symbol}$$

NOTE

$$P[x_i, y_j] = \begin{matrix} & y_1 & y_2 \\ x_1 & 0.48 & 0.12 \\ x_2 & 0.12 & 0.28 \end{matrix}$$

$$P[y_j/x_i] = \begin{matrix} & y_1 & y_2 \\ x_1 & 0.8 & 0.2 \\ x_2 & 0.3 & 0.7 \end{matrix}$$

$$H(Y) = \sum_{j=1}^{2} p(y_j) \log \frac{1}{p(y_j)}$$

$$= p(y_1) \log \frac{1}{p(y_1)} + p(y_2) \log \frac{1}{p(y_2)}$$

$p(y_1) = 0.48 + 0.12 = 0.6$

$p(y_2) = 0.12 + 0.28 = 0.4$

$$H(Y) = 0.6 \log \frac{1}{0.6} + 0.4 \log \frac{1}{0.4}$$

$$= 0.971 \text{ bits/msg-symbol}$$

The mutual Information $I(X, Y)$ is

$$I(X, Y) = H(Y) - H(Y/X)$$

$$= 0.971 - 0.7857$$

$$= 0.1853 \text{ bits/msg-symbol}$$

To find Channel Capacity:-

To find Channel Capacity, Muroga's method can be used.

$$\begin{bmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = \begin{bmatrix} 0.8 \log 0.8 + 0.2 \log 0.2 \\ 0.3 \log 0.3 + 0.7 \log 0.7 \end{bmatrix} = \begin{bmatrix} -0.722 \\ -0.8813 \end{bmatrix}$$

$0.8Q_1 + 0.2Q_2 = -0.722$ — (1)

$0.2Q_1 + 0.7Q_2 = -0.8811$ — (2)

multiply eq (1) by 1 and eq (2) by 4. and then Subtracting eq.

$0.8Q_1 + 0.6Q_2 = -2.166$

$2.4Q_1 + 5.6Q_2 = -7.0504$

(-) (-) (+)

$-5Q_2 = 4.8844$

$Q_2 = \frac{4.8844}{-5} = -0.9769$ — (3)

eq (3) Subs into eq (1)

$0.8Q_1 + (0.2)(-0.9769) = -0.722$

$Q_1 = \frac{-0.722 + 0.19538}{0.8} = -0.65827$

The channel Capacity "C" is given by

$C = \log_2 [2^{Q_1} + 2^{Q_2}]$

$= \log_2 \left[2^{(-0.65827)} + 2^{(-0.9769)} \right]$

$= 0.1912$ bits/msg - symbol

* The noise characteristics of a channel is shown in below fig. Find the channel capacity. If it noise symmetric channel, Recompute the channel capacity.

July/Aug -2003, JAN-2014 [10M]

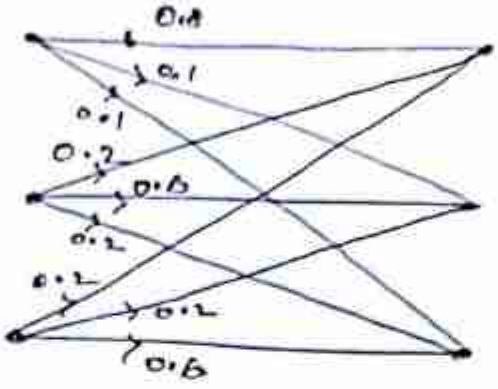


Fig Channel diagram.

Sol:- The channel matrix can be constructed as.

$$P(Y/X) = \begin{matrix} \\ X_1 \\ X_2 \\ X_3 \end{matrix} \begin{matrix} \begin{matrix} Y_1 & Y_2 & Y_3 \end{matrix} \\ \begin{bmatrix} 0.8 & 0.1 & 0.1 \\ 0.2 & 0.6 & 0.2 \\ 0.2 & 0.2 & 0.6 \end{bmatrix} \end{matrix}$$

muroga's method is used for finding the channel capacity is given by

$$\begin{bmatrix} 0.8 & 0.1 & 0.1 \\ 0.2 & 0.6 & 0.2 \\ 0.2 & 0.2 & 0.6 \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \end{bmatrix} = \begin{bmatrix} 0.8\log(0.8) + 0.1\log 0.1 + 0.1\log 0.1 \\ 0.2\log 0.2 + 0.6\log 0.6 + 0.2\log 0.2 \\ 0.2\log 0.2 + 0.2\log 0.2 + 0.6\log 0.6 \end{bmatrix}$$

$$\begin{bmatrix} 0.8Q_1 + 0.1Q_2 + 0.1Q_3 \\ 0.2Q_1 + 0.6Q_2 + 0.2Q_3 \\ 0.2Q_1 + 0.2Q_2 + 0.6Q_3 \end{bmatrix} = \begin{bmatrix} -0.922 \\ -1.371 \\ -1.371 \end{bmatrix}$$

$0.8Q_1 + 0.1Q_2 + 0.1Q_3 = -0.922$ — (1)

$0.2Q_1 + 0.6Q_2 + 0.2Q_3 = -1.371$ — (2)

$0.2Q_1 + 0.2Q_2 + 0.6Q_3 = -1.371$ — (3)

Subtracting eq (2) and (3) we get.

$0.2Q_1 + 0.6Q_2 + 0.2Q_3 = -1.371$

$0.2Q_1 + 0.2Q_2 + 0.6Q_3 = -1.371$

$0.4Q_2 - 0.4Q_3 = 0$

$Q_2 = Q_3$ // — (4)

eq (4) Sub into eq (1) and eq (2)

$0.8Q_1 + 0.1Q_2 + 0.1Q_3 = -0.922$

$0.8Q_1 + 0.1Q_2 + 0.1Q_2 = -0.922$

$0.8Q_1 + 0.2Q_2 = -0.922$ // — (5)

$0.2Q_1 + 0.6Q_2 + 0.2Q_3 = -1.371$

$0.2Q_1 + 0.6Q_2 + 0.2Q_2 = -1.371$

$0.2Q_1 + 0.8Q_2 = -1.371$ // — (6)

multiply eq (6) by 4.

$[0.2Q_1 + 0.8Q_2 = -1.371] \times 4$

$0.8Q_1 + 3.2Q_2 = -5.484$ — (7)

Subtracting eq (5) from eq (7) we get

$0.8Q_1 + 0.2Q_2 = -0.922$

$0.8Q_1 + 3.2Q_2 = -5.484$

$3Q_2 = -4.562$

$Q_2 = -1.5202 = Q_3$ //

- from eq (1)

$0.8Q_1 + 0.1(-1.5207) + (0.1)(-1.5207) = -0.932$

$Q_1 = -0.7723$ //

The Channel Capacity is given by.

$$C = \log_2\left[2^{B_1} + 2^{B_2} + 2^{Q_1}\right] \text{ bits/message-symbol}$$

$$= \log_2\left[2^{(-0.7723)} + 2^{(-1.5207)} + 2^{(-1.5207)}\right]$$

$$= 0.159 \text{ bits/message-symbol}$$

If it were a symmetric channel, then the channel capacity is given by.

$$C = \log_2(3) - h$$

$$= \log_2(3) - \sum_{i=1}^{3} P_i \log\frac{1}{P_i}$$

If the 2nd row and 3rd row elements are same. or 0.8, 0.1 and 0.1 in some other order. then

$$C = \log_2(3) - \left[0.8\log\frac{1}{0.8} + 0.1\log\frac{1}{0.1} + 0.1\log\frac{1}{0.1}\right]$$

$$= 0.663 \text{ bits/message-symbol}$$ //

If the 1st row element are same as 2nd and 3rd row elements in some other. Then.

$$C = \log_2(3) - \left[0.2\log\frac{1}{0.2} + 0.6\log\frac{1}{0.6} + 0.2\log\frac{1}{0.2}\right]$$

$$C = 0.214 \text{ bits/message-symbol}$$ //

* Show. That the Capacity of a discrete memoryless Channel Shown in fig (below) is

$C = \log_2 \left[\frac{2(1+\beta)}{\beta}\right]$ bps/ where $\beta = 2^{\alpha}$ and $\alpha = [p\log_2 p + q\log_2 q]$

Symbol rate $r_s = 1$ bit/symbol

$P(x=0) = P(x=1) = p$

$P(x=1) = P(x=2) = q$.

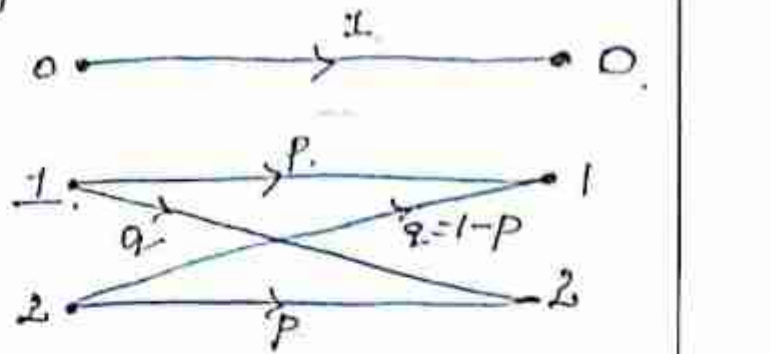


Fig :- Channel diagram.

Sol :- The Channel matrix for the given channel is given by

$$P(Y/X) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & p & q & 0 \\ 0 & q & p & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

By Muroga's method. The Channel Capacity Can be determined

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & p & q & 0 \\ 0 & q & p & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{bmatrix} = \begin{bmatrix} 1\log 1 \\ p\log p + q\log q \\ p\log p + q\log q \\ 1\log 1 \end{bmatrix}$$

$$\begin{bmatrix} Q_1 \\ pQ_2 + qQ_3 \\ qQ_2 + pQ_3 \\ Q_4 \end{bmatrix} = \begin{bmatrix} 0 \\ -\alpha \\ -\alpha \\ 0 \end{bmatrix} //$$

$\therefore Q_1 = 0 \quad pQ_2 + pQ_3 = -\alpha // \quad qQ_2 + pQ_3 = -\alpha // \quad Q_4 = 0$

$Q_1 = 0$.

$p Q_2 + q Q_3 = -\alpha$ — (1)

$q Q_2 + p Q_3 = -\alpha$ — (2)

$Q_4 = 0$.

eq (1) & (2) Equating we get

$p(Q_2) + Q_3 \cdot q = +q Q_2 + p Q_3$

$p Q_2 - q Q_2 = q Q_3 + p Q_3$

$Q_2 (p-q) = Q_3 (p-q)$

$Q_2 = Q_3$ // — (3)

eq (3) Sub in to eq (1)

$p Q_2 + q Q_3 = -\alpha$

$p Q_2 + q Q_2 = -\alpha$

$(p+q) Q_2 = -\alpha$.

where $(p+q=1)$ then $Q_2 = -\alpha$ // — (4)

eq (4) Sub in to eq (3)

$Q_3 = -\alpha$ //

Channel Capacity is given by

$$C = \log \left[2^{Q_1} + 2^{Q_2} + 2^{Q_3} + 2^{Q_4} \right]$$

$$= \log \left[2^0 + 2^{-\alpha} + 2^{-\alpha} + 2^0 \right]$$

$$= \log \left[1 + \frac{1}{2^\alpha} + \frac{1}{2^\alpha} + 1 \right]$$

$$= \log \left[2 + \frac{2}{2^\alpha} \right]$$

Since $\beta = 2\alpha$.

$$C = \log \left[2 + 2/\beta \right] = \log_2 \left[\frac{2\beta+2}{\beta} \right] = \log_2 \left[\frac{2(\beta+1)}{\beta} \right]$$ proved //

*. For the given channel matrix, Compute the mutual Information. $I(X,Y)$ with $p(x_1)=1/2$ and $p(x_2)=1/2$

$$P(Y/X) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix}\begin{matrix} \begin{matrix} y_1 & y_2 & y_3 \end{matrix} \\ \begin{bmatrix} 2/3 & 1/3 & 0 \\ 0 & 1/6 & 5/6 \end{bmatrix} \end{matrix}$$

Sol: The mutual Inf^n $I(X,Y)$ is found using the formula

$$I(X,Y) = H(Y) - H(Y/X) \quad \text{---(1)}$$

$\because P(x_1)=1/2$ and $P(x_2)=1/2$

$$P(x_i, y_i) = P(x_i)\,P(y_i/x_i)$$

$$P(x_i, y_i) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix}\begin{matrix} \begin{matrix} y_1 & y_2 & y_3 \end{matrix} \\ \begin{bmatrix} 2/3 \times 1/2 & 1/3 \times 1/2 & 0 \times 1/2 \\ 0 \times 1/2 & 1/6 \times 1/2 & 5/6 \times 1/2 \end{bmatrix} \end{matrix}$$

$$= \begin{bmatrix} 1/3 & 1/6 & 0 \\ 0 & 1/12 & 5/12 \end{bmatrix} //$$

$$H(Y/X) = \sum_{i=1}^{2}\sum_{j=1}^{3} P(x_i, y_j)\log\frac{1}{P(y_j/x_i)}$$

$$= \frac{1}{3}\log\left(\frac{3}{2}\right) + \frac{1}{6}\log\left(\frac{3}{1}\right) + \frac{1}{12}\log\left(\frac{6}{1}\right) + \frac{5}{12}\log\left(\frac{6}{5}\right)$$

$$= 0.9842 \text{ bits/msg symbol} //$$

$y_1 = 1/3 + 0 = 1/3$

$y_2 = 1/6 + 1/12 = 1/4$

$y_3 = 5/12 //$

$$H(Y) = \sum_{j=1}^{3} P(y_j)\log\frac{1}{P(y_j)}$$

$$= \frac{1}{2}\log 3 + \frac{1}{4}\log(4) + \frac{5}{12}\log\frac{12}{5}$$

$$= 1.55486 \text{ bits/message symbol} //$$

$I(X,Y) = H(Y) - H(Y/X)$

$= 1.5546 - 0.7842$

$= 0.7704$ bits/msg-symbol //

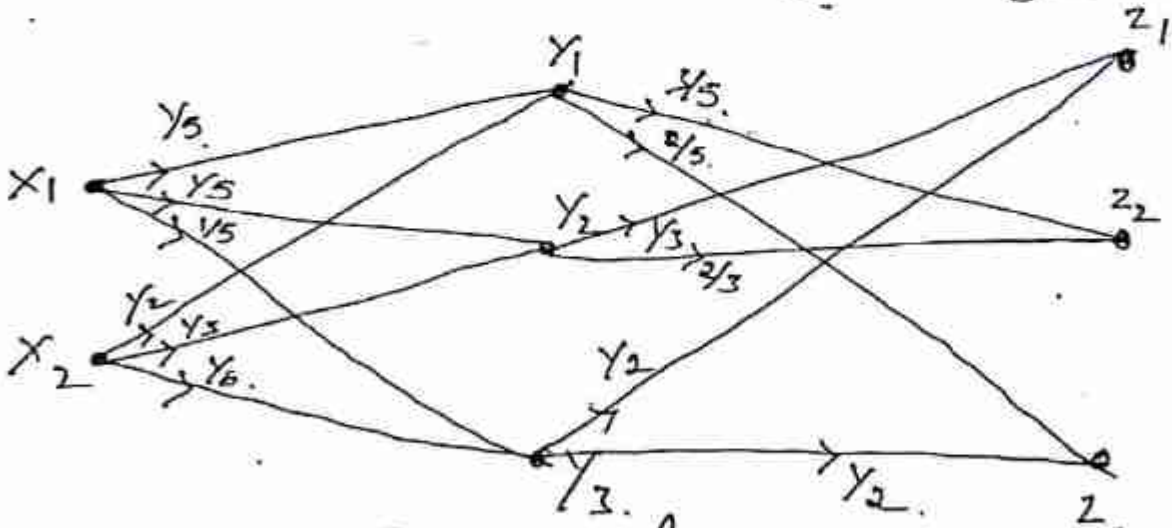
* Two noisy channels are cascaded. whose channel matrices are given by

$$P[Y/X] = \begin{bmatrix} 1/5 & 1/5 & 1/5 \\ 1/2 & 1/3 & 1/6 \end{bmatrix} \quad P[Z/Y] = \begin{bmatrix} 0 & 3/5 & 2/5 \\ 1/3 & 2/3 & 0 \\ 1/2 & 0 & 1/2 \end{bmatrix}$$

with $P(X_1) = P(X_2) = 1/2$. Find the overall mutual information $I(X,Z)$.

Sol:

$$P[Y/X] = \begin{matrix} \\ X_1 \\ X_2 \end{matrix} \begin{matrix} \begin{matrix} Y_1 & Y_2 & Y_3 \end{matrix} \\ \begin{bmatrix} 1/5 & 1/5 & 1/5 \\ 1/2 & 1/3 & 1/6 \end{bmatrix} \end{matrix} \quad P[Z/Y] = \begin{matrix} \\ Y_1 \\ Y_2 \\ Y_3 \end{matrix} \begin{matrix} \begin{matrix} Z_1 & Z_2 & Z_3 \end{matrix} \\ \begin{bmatrix} 0 & 3/5 & 2/5 \\ 1/3 & 2/3 & 0 \\ 1/2 & 0 & 1/2 \end{bmatrix} \end{matrix}$$



From the cascaded channel diagram

$P(Z_1/X_1) = (1/5)(1/3) + (3/5)(1/2) = \frac{1}{15} + \frac{3}{10} = 11/30$ //

$P(Z_1/X_2) = (1/3)(1/3) + (1/6)(1/2) = \frac{1}{9} + \frac{1}{12} = 7/36$ //

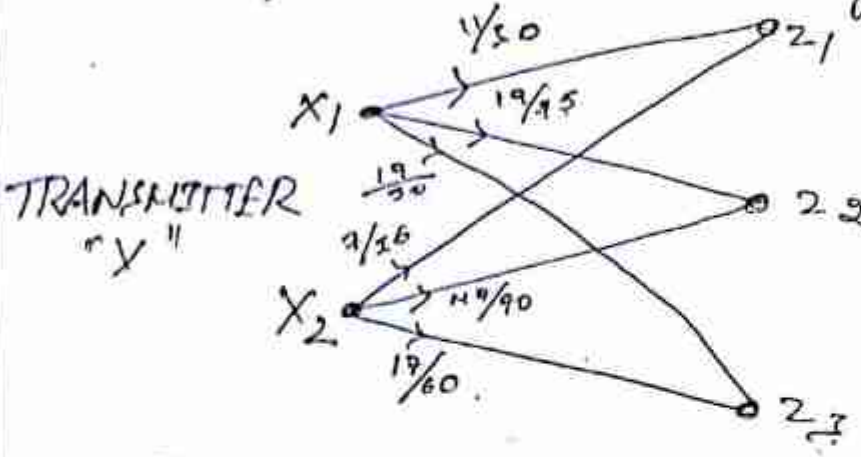
$P(z_1/x_1) = (1/5)(3/5) + (1/5)(2/5) = 19/75$ //

$P(z_2/x_1) = (1/5)(3/5) + (1/5)(2/5) = 47/90$ //

$P(z_3/x_1) = (1/5)(3/5) + (3/5)(1/2) = 19/50$ //

$P(z_3/x_2) = (1/2)(3/5) + (1/6)(1/2) = 17/60$ //

The Cascaded Channel diagram can now be redrawn as an equivalent Channel diagram as.



The Channel matrix is given by

$$P(z/x) \text{ or } P(z_k/x_i) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix} \begin{matrix} z_1 & z_2 & z_3 \end{matrix} \begin{bmatrix} 14/50 & 19/75 & 19/50 \\ 9/16 & 49/90 & 19/60. \end{bmatrix}$$

The mutual Information $I(X,Z)$ is given by

$$I(X,Z) = H(Z) - H(Z/X)$$

To find $H(Z/X)$:

$$P(x_i, z_k) = P(x_i)\, P(z_k/x_i)$$

$\therefore P(x_1) = 1/2 = P(x_2)$

$$P(X,Z_K) = \begin{matrix} \\ X_1 \\ X_2 \end{matrix} \begin{matrix} Z_1 & Z_2 & Z_3 \end{matrix} \begin{bmatrix} 11/30 \times 1/2 & 19/75 \times 1/2 & 19/50 \times 1/2 \\ 7/36 \times 1/2 & 47/90 \times 1/2 & 19/60 \times 1/2 \end{bmatrix}$$

$$= \begin{bmatrix} 11/60 & 19/150 & 19/100 \\ 7/22 & 47/180 & 17/120 \end{bmatrix}$$

$$H(Z/X) = \sum_{j=1}^{2} \sum_{K=1}^{2} P(X_j, Z_K) \log \frac{1}{P(Z_K/I_K)}$$

$$= \frac{11}{60} \log \frac{50}{11} + \frac{19}{150} \log \frac{75}{19} + \frac{19}{100} \log \frac{50}{19} + \frac{7}{22} \log \frac{36}{7}$$

$$+ \frac{47}{180} \log \frac{90}{47} + \frac{17}{120} \log \frac{60}{17}$$

$$= 1.5137 \text{ bits/message-symbol} //$$

To find $H(Z)$.

$$P(Z_1) = \frac{11}{60} + \frac{7}{22} = \frac{101}{360} //$$

$$P(Z_2) = \frac{19}{120} + \frac{47}{180} = \frac{349}{900} // \quad P(Z_3) = \frac{19}{100} + \frac{17}{120} = \frac{199}{600} //$$

$$H(Z) = \sum_{K=1}^{3} P(Z_K) \log \frac{1}{P(Z_K)}$$

$$= \frac{101}{360} \log \frac{360}{101} + \frac{349}{900} \log \frac{900}{349} + \frac{199}{600} \log \frac{600}{199}$$

$$= 1.5725 \text{ bits/message-symbol} //$$

∴ The mutual Information.

$$I(X,Z) = H(Z) - H(Z/X)$$

$$= 1.5725 - 1.5137$$

$$= 0.0588 \text{ bits/message-symbol} //$$

* Determine the rate of Transmission of Information through a channel whose noise characteristics are shown in the below given $P(X_1) = P(X_2) = 1/2$. Assume $r_s = 10,000$ symbol/sec.

Jan/Feb-2003 [12 M].

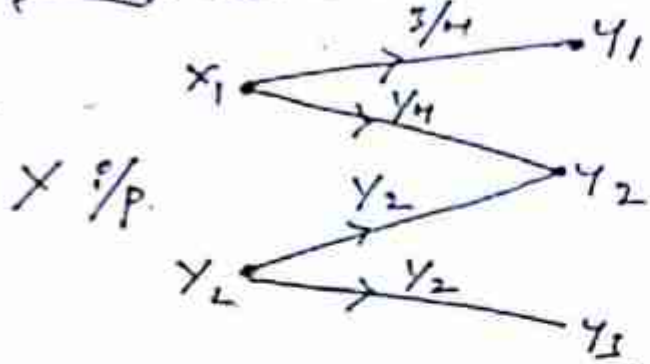


Fig:- Channel diagram.

Sol:- The channel matrix for the given channel is given by

$$P(Y/X) = \begin{matrix} \\ X_1 \\ X_2 \end{matrix} \begin{matrix} \begin{matrix} Y_1 & Y_2 & Y_3 \end{matrix} \\ \begin{bmatrix} 3/4 & 1/4 & 0 \\ 0 & 1/2 & 1/2 \end{bmatrix} \end{matrix}$$

The mutual Information is found using the formula.

$$I(X,Y) = H(Y) - H(Y/X)$$

To find $H(Y/X)$:-

$$P(X_1) = P(X_2) = 1/2$$

$$P(X,Y) = \begin{matrix} \\ X_1 \\ X_2 \end{matrix} \begin{matrix} \begin{matrix} Y_1 & Y_2 & Y_3 \end{matrix} \\ \begin{bmatrix} 3/4 \times 1/2 & 1/4 \times 1/2 & 0 \\ 0 & 1/2 \times 1/2 & 1/2 \times 1/2 \end{bmatrix} \end{matrix} = \begin{bmatrix} 3/8 & 1/8 & 0 \\ 0 & 1/4 & 1/4 \end{bmatrix}$$

$$H(Y/X) = \sum_{i=1}^{3}\sum_{j=1}^{3} p(x_i, y_j) \log \frac{1}{p(y_j/x_i)}$$

$$= \tfrac{3}{8}\log\tfrac{4}{3} + \tfrac{1}{8}\log(4) + \tfrac{1}{4}\log(2) + \tfrac{1}{4}\log(2)$$

$$= 0.90564 \text{ bits/message-symbol}$$

To find $H(Y)$:

$p(y_1) = \tfrac{3}{8}$ // $p(y_2) = \tfrac{1}{8} + \tfrac{1}{4} = \tfrac{3}{8}$ // $p(y_3) = \tfrac{1}{4}$ //

$$H(Y) = \sum_{j=1}^{3} p(y_j) \log \frac{1}{p(y_j)} \quad \text{--- (2)} \left[\tfrac{3}{8}\log\tfrac{8}{3}\right] + \tfrac{1}{4} + \tfrac{1}{4}\log 4$$

$$= 1.56128 \text{ bits/msg-symbol}$$

The mutual Information is given by

$$I(X,Y) = H(Y) - H(Y/X)$$

$$= 1.56128 - 0.90564$$

$$= 0.65564 \text{ bits/message-symbol}$$

The Rate of transmission of Information through the Channel is given by

$$R_t = I(X,Y)\, r_s$$

$$= (0.65564)(10000)$$

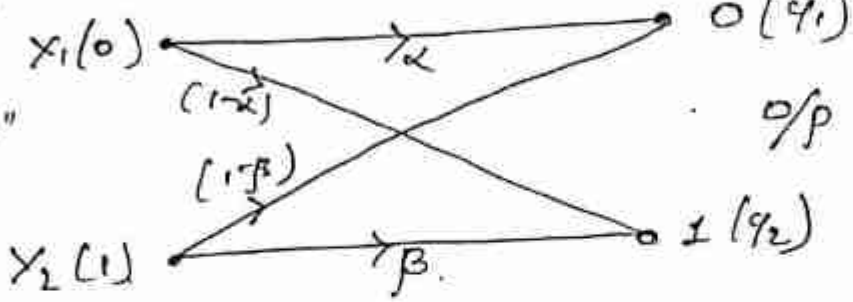
$$= 6556.4 \text{ bits/sec}$$

* A Non-Symmetric binary Channel is given as below fig. 1) find. H(X), H(Y), H(X/Y) and H(Y/X) given.

$P(X=0) = 1/4$, $P(X=1) = 3/4$, $\alpha = 0.95$, $\beta = 0.9$

B) find. The Capacity of the binary Symmetric Channel if $\alpha = \beta = 0.95$.

July/Aug-2004 [8M], Jan-2017 [8M], July-2018 [10M], July-2009 [10M]



Sol :- The Channel matrix is given by

$$P[Y/X] = \begin{matrix} \\ X_1 \\ X_2 \end{matrix} \begin{matrix} \begin{matrix} Y_1 & Y_2 \end{matrix} \\ \begin{bmatrix} \alpha & 1-\alpha \\ 1-\beta & \beta \end{bmatrix} \end{matrix}$$

$$= \begin{matrix} \\ X_1 \\ X_2 \end{matrix} \begin{matrix} \begin{matrix} Y_1 & Y_2 \end{matrix} \\ \begin{bmatrix} 0.95 & 0.05 \\ 0.1 & 0.9 \end{bmatrix} \end{matrix}$$

1) The JPM is Constructed using $P(X_i, Y_j) = P(X_i) P(Y_j/X_i)$

given $P(X=0) = 1/4 = P(X_1)$

$P(X=1) = P(X_2) = 3/4$ //

$$P(X,Y) = \begin{matrix} \\ x_1 \\ x_2 \end{matrix}\begin{matrix} \begin{matrix} y_1 & y_2 \end{matrix} \\ \begin{bmatrix} 0.25 \times 1/4 & 0.25 \times 1/4 \\ 0.1 \times 3/4 & 0.9 \times 3/4 \end{bmatrix} \end{matrix}$$

$$= \begin{matrix} \\ x_1 \\ x_2 \end{matrix}\begin{matrix} \begin{matrix} y_1 & y_2 \end{matrix} \\ \begin{bmatrix} 0.1875 & 0.0625 \\ 0.075 & 0.675 \end{bmatrix} \end{matrix}$$

The Entropy of i/p symbol is given by

* $H(X) = \sum_{i=1}^{2} P(x_i) \log \frac{1}{P(x_i)} = \frac{1}{4}\log(4) + \frac{3}{4}\log\frac{4}{3}$

$= 0.8113$ bits/message - symbol

$P(y_1) = 0.1875 + 0.075 = 0.2625$

$P(y_2) = 0.0625 + 0.675 = 0.7375$

* $H(Y) = \sum_{i=1}^{2} P(y_i) \log \frac{1}{P(y_i)}$

$= 0.2625 \log \frac{1}{0.2625} + 0.7375 \log \frac{1}{0.7375}$

$= 0.8305$ bits/message - symbol

* $H(Y/X) = \sum_{i=1}^{2}\sum_{j=1}^{2} P(x_i, y_j) \log \frac{1}{P(y_j/x_i)}$

$= 0.1875 \log \frac{1}{0.75} + 0.0625 \log \frac{1}{0.25} + 0.075 \log \frac{1}{0.1}$

$+ 0.675 \log \frac{1}{0.9}$

$= 0.5546$ bits/message - symbol

$$H(X/Y) = H(X) + H(Y/X) - H(Y)$$

$$= 0.8113 + 0.5516 - 0.8205$$

$$= 0.5124 \text{ bits/msg-symbol}$$ //

2) $\alpha = \beta = 0.75$

$$C = 1 - \left[p \log \frac{1}{p} + p \log \frac{1}{p} \right]$$

$$= 1 - \left[\alpha \log \frac{1}{\alpha} + (1-\alpha) \log \frac{1}{(1-\alpha)} \right]$$

$$= 1 - \left[0.75 \log \frac{1}{0.75} + 0.25 \log \frac{1}{0.25} \right]$$

$$= 0.1887 \text{ bits/msg-symbol}$$ //

CONTINUOUS CHANNELS :-

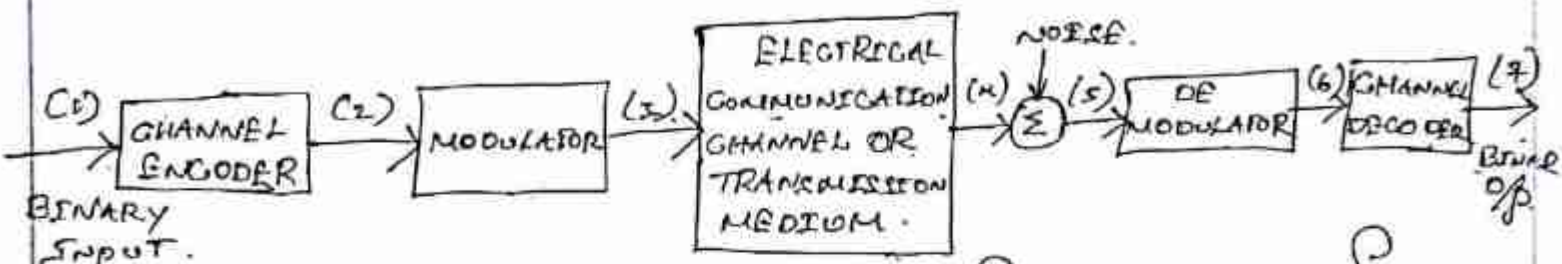
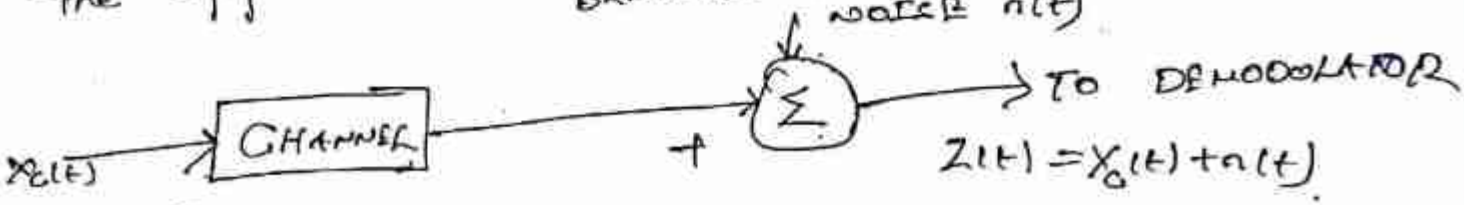


Fig:- Block diagram of Communication System

* The Communication Channel b/w points (3) and (5) in above figure is analog (or) Continuous in nature.
* The i/p Signals are Continuous functions of time and the function of the Channel is to produce at it's output the electrical waveform presented at its input.
* the Channel modifies the input waveform in a random fashion due to additive noise.
* Additive noise Considered here is assumed to be purely Gaussian.
* modulation and demodulation techniques are designed, with the primary objective of reducing the effects of Gaussian noise.



(from the modulator)

fig:- Analog portion of Communication Channel

Above fig:-

→ The i/p. to the channel is an. Random process $X_c(t)$ which consists of the collection of all the waveforms generated by the modulator.

→ The bandwidth of $X_c(t)$ and the channel is assumed to B Hz.

→ The additive noise at the channel output is zero mean, band limited white Gaussian noise $n(t)$.

→ The Capacity of the portion of the channel is found by maximizing the rate of information. Transmission with respect to the distribution of $X_c(t)$.

ENTROPY OF CONTINUOUS SIGNALS & DIFFERENTIAL ENTROPY:-

Various entropies in the discrete case are defined based on the concept of statistical averages of a discrete set. In a similar way, we can define entropies for a continuous channel also in terms of statistical averages of a source probability distribution.

The entropy of a discrete information source is given by

$$H(x) = \sum_{k=1}^{N} P_k \log_2 \frac{1}{P_k} \text{ bits/sym}$$

Consider a Continuous Random variable X with a probability density function $f(x)$. Then the summation of the above Equation has to be replaced by an integration and the Expression for the Entropy of a Continuous Signal is given by

$$H(X) = \int_{-\infty}^{+\infty} f(x) \log_2 \frac{1}{f(x)} dx$$

It is also Called a different Entropy.

MUTUAL INFORMATION AND CAPACITY OF A CONTINUOUS NOISY CHANNEL :-

Joint Entropies for Continuous Systems is given by

$$H(X,Y) = -\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y) \log_2 f(x,y)\, dx\, dy$$

$$H(X/Y) = -\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y) \log_2 f(x/y)\, dx\, dy.$$

Hly the mutual Information $I(X;Y)$ is given by

$$I(X;Y) = -\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y) \log_2 \frac{f(x/y)}{f(x)}\, dx\, dy.$$

$$= -\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y) \log_2 \frac{f(y/x)}{f(y)}\, dx\, dy.$$

and the Channel Capacity is given by

$$C = \max [I(X;Y)]$$

It can be seen that $I(X;Y) \geq 0$, meaning that mutual Information of a Continuous Channel is always

Perfect transmission : If the channel allows transmission of R_s symbol per second, Then channel capacity rate C_s becomes.

$$C_s = R_s C$$

INFORMATION THEORY AND CODING

5TH SEM E&C



JAYANTHDWIJESH H P **M.tech (DECS)**

Assistant Professor – Dept of E&C

B.G.S INSTITUTE OF TECHNOLOGY (B.G.S.I.T)

B.G Nagara, Nagamangala Tq, Mandya District- 571448

FORMULAS FOR REFERENCE

MODULE – 4 (ERROR CONTROL CODING AND BINARY CYCLIC CODES)

CHAPTER -1 ERROR CONTROL CODING

- The code vector is given by

  [C]= [D] [G]

- The Message vector is given by

  2^k

- The Parity check matrix is given by

  [H]= [P^T : I_{n-k}] or [I_{n-k} : P^T]

- The generator matrix is given by

  [G]= [I_K: P] or [P : I_K]

- The syndrome is given by

  S=RH^T or rH^T

- The corrected code vector is given by

  C=R+E

The single error correcting (n, k) Hamming code has the following parameters

- Code length: n≤ 2^{n-k-1}
- Number of message bits : k≤ n-log_2 (n+1)
- Number of parity check bits: (n-k)
- Error correcting capability: t = $\frac{d_{min} - 1}{2}$
- Error correcting capability: $d_{min} - 1$
- H^T is given by

  H^T = [$\frac{P}{I_{n-k}}$] or [$\frac{d_{k\,X\,(n-k)}}{I_{n-k}}$]

HAMMING BOUND (perfect code or not)

- $2^{n-k} \geq \sum_{i=0}^{t} {}^{n}_{i}C$

- **The Non systematic cyclic code vector is given by**

  V(X) =D(X) g(X)

- **The systematic cyclic :**

  The remainder polynomial R(X) is given by

  $$R(X) = \frac{D(X)\, X^{n-k}}{g(X)}$$

  The code vector is given by

  [V] = [R][D]

- **The parity check polynomial is given by**

  h (x) =x^7 +1 / g(x)

- **The code vector is given by**

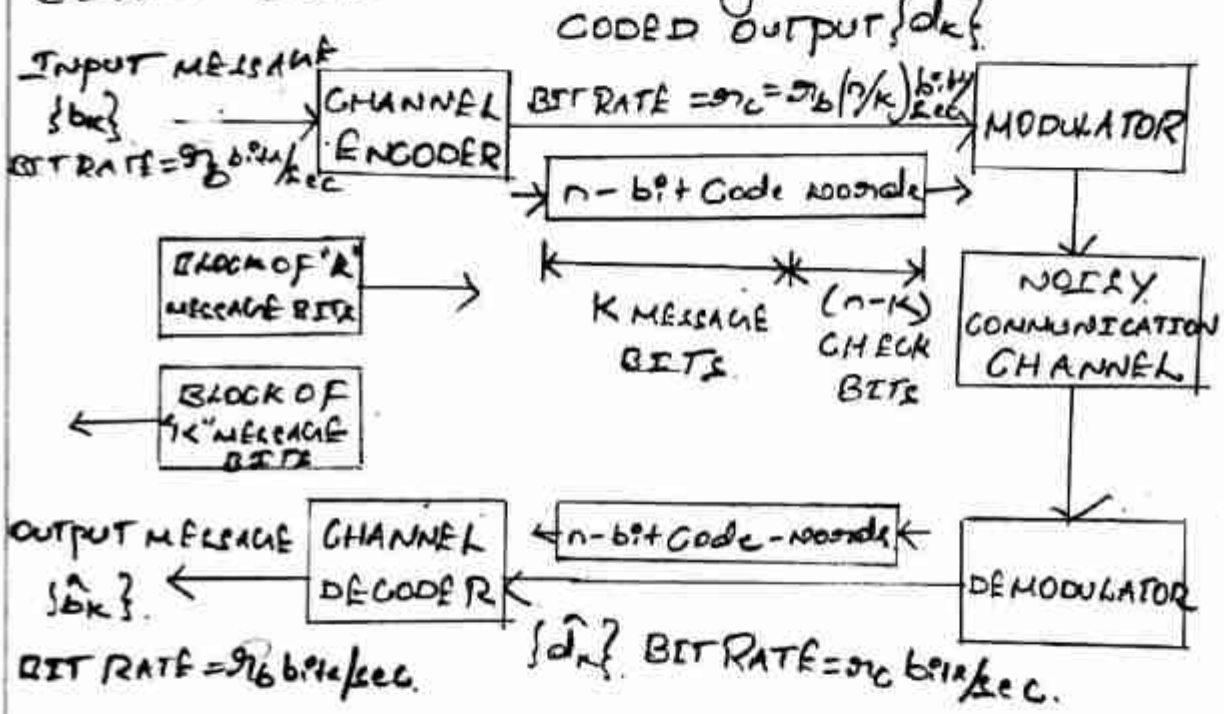
  [V] = [R][D]

MODULE - 4

INTRODUCTION TO ERROR CONTROL CODING

ERROR CONTROL CODING:-

Block diagram of Communication System Employing Error Control Coding:



*The above block diagram gives us a digital Communication System, employing error control coding. The main functional blocks are the channel encoder, Channel decoder, modulator and demodulator and the noisy Communication Channel with a capacity 'C' bits/sec.

* The Source generates a message block $\{b_k\}$ at a rate of r_b bits/sec and feeds it to the Channel Encoder.
* The Channel encoder, then adds $(n-k)$ number of redundant bits to these k-bit messages to form n-bit Code words. Thus $(n-k)$ number of additional bits also called "Check bits" do not Convey any Information but helps Channel decoder to detect and Correct errors.
* The bit rate of the coded output block $\{d_k\}$ will be $r_c = r_b (n/k)$ bits/sec. This is the rate at which the modem operates, to produce a message block $\{\hat{d}_k\}$ at the Receiver.
* the Channel decoder then decodes this message to get back the Information block $\{\hat{b}_k\}$ at the Receiver.
* The Information block $\{\hat{b}_k\}$ usually differs from the Transmitted block $\{b_k\}$. Then probability of error

$$P_e = P\{\hat{b}_k \neq b_k\}$$

* Then the probability of error which depends on bit rate r_c is defined as

$$P_c = P\{\hat{d}_k \neq d_k\}$$

* It is Required to design an Error Control Coding Scheme Such that the overall probability of error is less than the desired value.

METHOD OF CONTROLLING ERRORS : DEC-2013/2014 (06M)

There are two different methods available for Controlling errors in a Communication System.

1) Forward acting error Correction method :-

The method of Controlling errors at the Receiver through attempts to Correct noise induced errors is called the forward acting error Correction method.

2) Error detection method :-

* In this method the decoder examines the demodulator output, accepts the Received Sequences if it matches with a valid message Sequence.
* If not the decoder discards the Received Sequence and notifies the Transmitter regarding the error and requests for retransmission of the message till the Correct Sequence is Received. Thus the decoder attempts to detect errors, but does not attempt to Correct them.

* The disadvantages of error detection method are, the Requirement of reverse Channel and slow down of the effective rate of data Transmission.

TYPES OF ERRORS :- June/July-2010 (6M)

1) Random errors :-

The Transmission errors that occur due to the presence of White Gaussian noise are referred to as "Random errors".

Sources of Gaussian noise include thermal and shot noise in the Transmitting and Receiving Equipment, thermal noise in the Channel and Radiation picked up by the Receiving antenna.

2) BURST Error:-

Impulse noise is Characterized by long quiet Intervals, followed by high amplitude noise bursts.

Examples of Impulse noise are noise that arises due to lightning, Switching transient, man-made noise etc. When such noise bursts occur, they effect more than one Symbol and the error Caused is called "Burst Error".

Imp V.V.I. TYPES OF CODES :- June/July-2010 [6M].

1) Block Codes:-

Block Code Consists of (n-k) number of check bits (redundant bits) being added to "k" number of Information bits to form "n" bits Code words.

These (n-k) number of check bits are "derived from "k" Information bits". At the Receiver, the check bits are used to detect and Correct errors which may occur in the Entire "n"-bit Code words.

2) Convolutional Codes:-

In this Code, the check bits are Continuously interleaved with Information bits, these check bits will help to Correct errors not only in that particular block but also in other blocks as well.

LINEAR BLOCK CODES :-

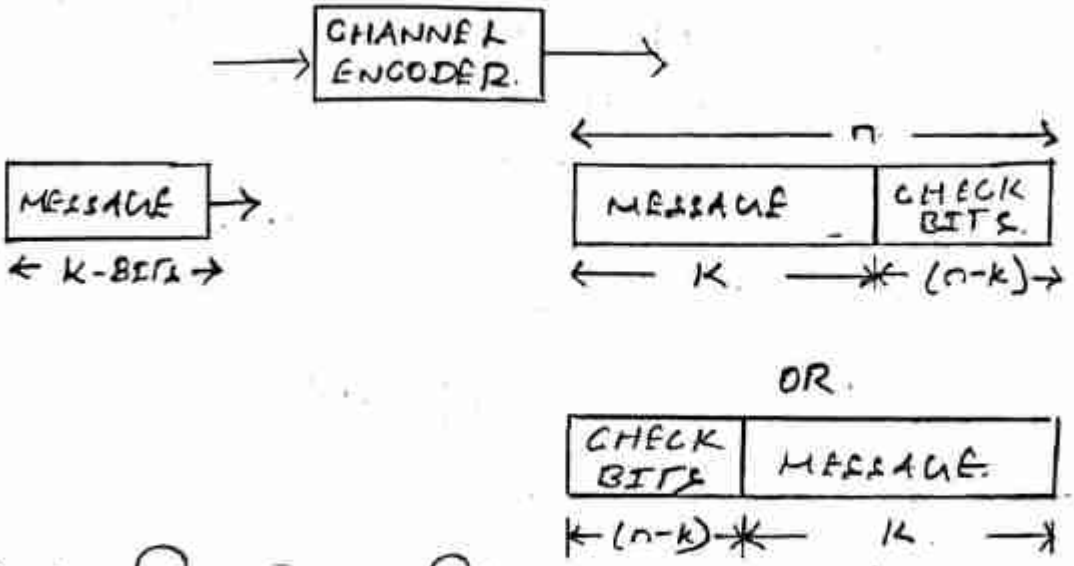


Fig :- The formation of Linear block code.

* In Channel Encoder, a block of "k" message bits is encoded into a block of "n" bits. by adding (n-k) number of check bits. as shown in figure (above).
* Where n>k., such a code formed is called (n, k) block code.
* A (n, k) block code is said to be a (n, k) "linear block code." if it satisfies the condition given below.

→ Let C_1 and C_2 be any 2 code-words belonging to a set of (n, k) block code. if $C_1 \oplus C_2$ is also a 'n'-bit code word belonging to the same set of (n, k) block code, then such a block code is called (n, k) linear block code.

→ A (n, k) linear block is said to be "Systematic" if the k-message bit appear either at the "beginning" of the code-word (or) at the end of the code-word as shown in above figure.

V.T.U. I Sem July-2011 (6M)

MATRIX DESCRIPTION OF LINEAR BLOCK CODES:

* Let the message block of K-bits (codewords) be represented as a "row vector" (or) "K-tuple" called "message-vector" given by.

$$[D] = \{d_1, d_2 \cdots d_k\} \quad \text{---(1)}$$

where $d_1, d_2 \cdots d_k$ are either 0's and 1's.

* The Channel Encoder Systematically adds (n-k) number of check-bits to form a (n,k) linear block code. Then the 2^k Code-vector can be represented by

$$C = \{c_1, c_2 \cdots c_n\} \quad \text{---(2)}$$

* In a Systematic linear block code the message bits appear at the beginning of the Code vector.

$$c_i = d_i \text{ for all } i = 1, 2, \cdots k. \quad \text{---(3)}$$

The Remaining (n-k) bits are check bits, Hence eq (2) and (3) can be Combined as.

$$[C] = \{\underbrace{c_1\ c_2 \cdots c_k}_{\text{k-message bits}}, \underbrace{c_{k+1}, c_{k+2}, \cdots c_n}_{(n-k)\text{ check bits.}}\} \quad \text{---(4)}$$

These (n-k) number of check bits $c_{k+1}, c_{k+2} \cdots c_n$ are derived from k-message bits using a predetermined rule as below:

$$c_{k+1} = p_{11}d_1 + p_{21}d_2 + \cdots + p_{k1}d_k \quad \text{---(5)}$$
$$c_{k+2} = p_{12}d_1 + p_{22}d_2 + \cdots + p_{k2}d_k$$
$$\vdots$$
$$c_n = p_{1,n-k}d_1 + p_{2,n-k}d_2 + \cdots + p_{k,n-k}d_k$$

where P_{11} P_{21} P_{12} P_{22} ... are either 0's and 1's and the addition above is performed using modulo-2 arithmetic.

modulo-2 addition
0+0=0
0+1=1
1+0=1
1+1=0

Table 1

modulo-2 multiplication
0.0=0
0.1=0
1.0=0
1.1=1

Table 2

modulo-2 subtraction
0-0=0
0-1=1
1-0=1
1-1=0

→ both a borrow of "1"

Table 3

* In a modulo-2 arithmetic, the addition of two binary digits is actually the "EXCLUSIVE-OR" operation of those two binary digits as given in following table (1). Table (2) shows the modulo-2 multiplication and Table (3) modulo-2 subtraction.
* Comparing table (1) and (2), one can conclude that modulo-2 subtraction is "same" as modulo-2 addition.
* It is possible to combine equation (3) (4) and (5) and express the result in a "matrix form" as,

$$[C_1\ C_2 \cdots C_k\ C_{k+1}\ C_{k+2} \cdots C_n] = [d_1\ d_2 \cdots d_k] \begin{bmatrix} 100\cdots0 & P_{11}\ P_{12} \cdots P_{1,n-k} \\ 010\cdots0 & P_{21}\ P_{22} \cdots P_{2,n-k} \\ \vdots & \\ 000\cdots1 & P_{k1}\ P_{k2} \cdots P_{k,n-k} \end{bmatrix}$$

(k terms) ((n-k) terms)

(07)

$$[C] = [D]\,[G].$$

Where, $[G]$ is called as "GENERATOR MATRIX" of order $(k \times n)$ given by

$$[G] = [I_k \,!\, P]_{(k \times n)}$$

Where I_k = unit matrix of order "k".

$[P]$ = arbitrary matrix. Called "PARITY MATRIX" of order $k \times (n-k)$.

The generator matrix $[G]$ can also be expressed as

$$[G] = [P \,!\, I_k]$$ //

PARITY CHECK MATRIX $[H]$:-

The generator matrix given by.

$$[G] = [I_k \,!\, P] = \begin{bmatrix} 100\cdots0 & P_{11} & P_{12} & \cdots & P_{1,n-k} \\ 010\cdots0 & P_{21} & P_{22} & \cdots & P_{2,n-k} \\ \vdots & & & & \\ 000\cdots1 & P_{k1} & P_{k2} & \cdots & P_{k,n-k} \end{bmatrix} \quad \text{--- (1)}$$

Associated with generator matrix $[G]$ of eq (1), is another matrix. Called "parity check matrix $-H$" given by

$$[H] = [P^T \,!\, I_{n-k}]$$

$$[H] = \begin{bmatrix} P_{11} & P_{21} & \cdots & P_{k1} & 100\cdots0 \\ P_{12} & P_{22} & \cdots & P_{k2} & 010\cdots0 \\ \vdots & & & & \\ P_{1,n-k} & P_{2,n-k} & \cdots & P_{k,n-k} & 000\cdots1 \end{bmatrix}$$

$[H]$ matrix is a $(n-k) \times (n)$ matrix and this matrix is used for error Correction.

If C is a valid code-vector, then prove that $CH^T = 0$ where H^T is the Transpose of the parity check matrix 'H'.

(OR) prove that $CH^T = 0$

June/July-2012[Mkl], June/July-2011(6M), June/July-2010[4M]

Sol:- The i^{th} row of the [G] matrix is given by

$$g_i = [00\ldots1\ldots0\ P_{i1}\ P_{i2}\ \ldots\ P_{ii}\ \ldots\ P_{i,n-k}]$$

(i^{th} element; $(k+i)^{th}$ element.)

The i^{th} row of the [H] matrix is given by.

$$h_i = [P_{1i}\ P_{2i}\ \ldots\ P_{ii}\ \ldots\ P_{ki}\ 00\ldots1\ldots0]$$

(i^{th} element; $(k+i)^{th}$ element)

Consider. $g_i h_i^T = [00\ldots1\ldots0\ P_{i1}\ P_{i2}\ \ldots\ P_{ii}\ \ldots\ P_{i,n-k}] \begin{bmatrix} P_{1i} \\ P_{2i} \\ \vdots \\ P_{ii} \\ \vdots \\ P_{ki} \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$

(i^{th} element; $(k+i)^{th}$ element)

module-2 multiplication of row and column matrix yields.

$$g_i h_j^T = (0)(P_{1j}) + (0)(P_{2j}) + \cdots + (1)(P_{ij}) + \cdots + (0)(P_{kj}) +$$
$$(P_{i1})(0) + (P_{i2})(0) + \cdots + (P_{ij})(1) + \cdots + (P_{i,n-k})(0)$$

$$= P_{ij} + P_{ij} = P_{ij}(1+1) = P_{ij}(0).$$

$$\therefore g_i h_j^T = 0 \;//$$

This Equation is True for Every values of 'i' and 'j' and hence in matrix form, we have.

$$[G][H^T] = 0 \quad \text{——— (1)}$$

pre multiplying both Side of equation (1) by $[D]$, we get

$$[D][G][H]^T = [D][0] = 0 \;//$$

Code vector $[C]$ is given by

$$[C] = [D][G]$$

$$\therefore [C][H]^T = 0$$

$$CH^T = 0 \;//$$

ENCODING CIRCUIT FOR (n, k) LINEAR BLOCK CODES

Consider the relation b/w codeword and Information

$$[C] = [D][G]$$

$$[C_1\ C_2 \cdots C_k\ C_{k+1}\ C_{k+2} \cdots C_n] = [d_1\ d_2 \cdots d_k] \begin{bmatrix} 1\,0 \cdots 0\ P_{11}\ P_{12} \cdots P_{1,n-k} \\ 0\,1 \cdots 0\ P_{21}\ P_{22} \cdots P_{2,n-k} \\ \vdots \\ 0\,0 \cdots 1\ P_{k1}\ P_{k2} \cdots P_{k,n-k} \end{bmatrix}$$

Expanding the above relation we get

$$C_1 = d_1$$
$$C_2 = d_2$$
$$\cdots$$
$$C_k = d_k$$
$$C_{k+1} = P_{11} d_1 + P_{21} d_2 + \cdots + P_{k1} d_k \quad \text{(E1)}$$
$$C_{k+2} = P_{12} d_1 + P_{22} d_2 + \cdots + P_{k2} d_k$$
$$\vdots$$
$$C_n = P_{1,n-k} d_1 + P_{2,n-k} d_2 + \cdots + P_{k,n-k} d_k$$

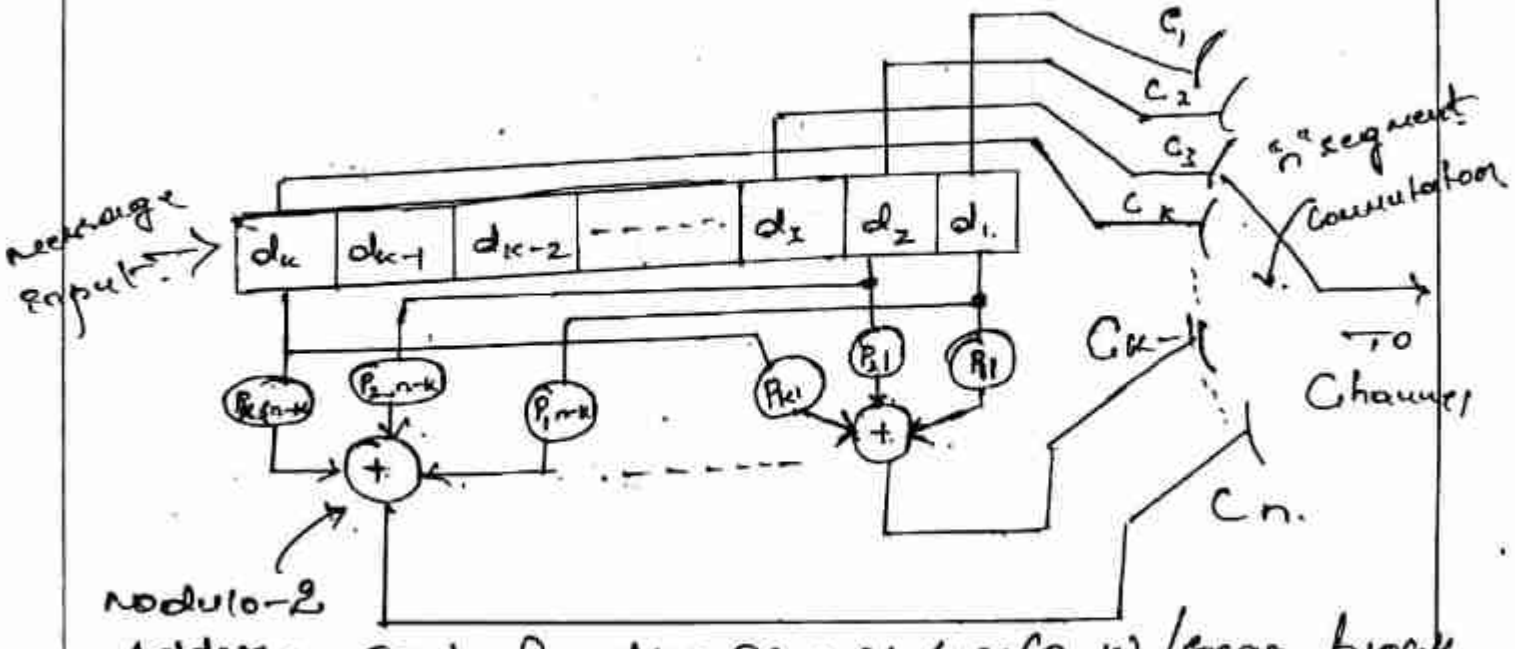


Fig:- Encoding Circuit for (n, k) linear block codes.

∴ The Encoder Circuit can be realized using a logical Circuit Satisfying eq (1). A general encoder Circuit for an (n, k) linear block shown in above.

SYNDROME CALCULATION AND ERROR DETECTION

* Let us Suppose that $C = (c_1, c_2 \cdots c_n)$ be a valid Code-vector transmitted over a noisy Communication Channel belonging to a (n, k) linear block code.
* Let $R = (r_1, r_2 \cdots r_{n-1})$ be the Received vector for the Transmitted Codeword. $C = (c_1 c_2 \cdots c_{n-1})$
* Let "E" be the Error Introduced during Transmission.

$$\therefore R = C + E$$

* Consider the parameter, Syndrome S, for the Received vector "r" given by

$$S = RH^T$$

But $R = C + E$

$$\therefore S = (C+E)H^T = CH^T + EH^T$$

But $CH^T = 0$

$$\therefore S = EH^T //$$

The Syndrome "S" will be a non-zero entity for a non-zero error pattern. A non-zero Syndrome indicates the presence of error during Transmission.

SYNDROME - CALCULATION CIRCUIT :-

Let the Received vector $R = (r_1\ r_2 \ldots r_n)$ the Syndrome vector 'S' is then given by equation.

$$[S] = [S_1, S_2 \ldots S_{n-k}] = RH^T$$

$$[S_1, S_2 \ldots S_{n-k}] = [r_1, r_2 \ldots r_n] \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1,n-k} \\ P_{21} & P_{22} & \cdots & P_{2,n-k} \\ \vdots & & & \\ P_{k1} & P_{k2} & \cdots & P_{k,n-k} \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

on multiplying by using modulo-2 arithmetic we get the Syndrome bits as.

$$S_1 = r_1 P_{11} + r_2 P_{21} + \cdots + r_k P_{k1} + r_{k+1}$$

$$S_2 = r_1 P_{12} + r_2 P_{22} + \cdots + r_k P_{k2} + r_{k+2}$$

$$\vdots$$

$$S_{n-k} = r_1 P_{1,n-k} + r_2 P_{2,n-k} + \cdots + r_k P_{k,n-k} + r_n$$

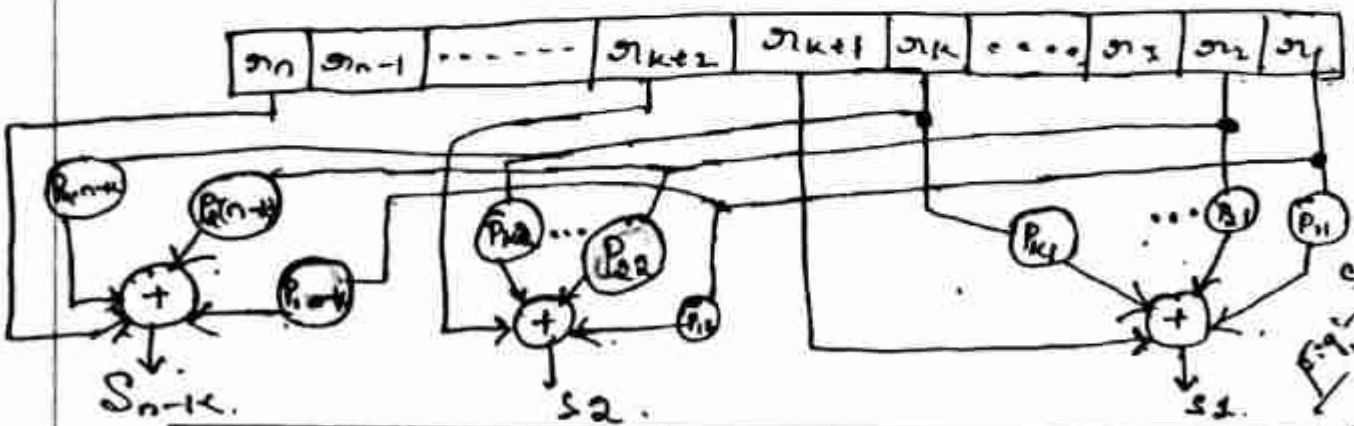


Fig: Syndrome Calculation circuit for (n,k) LBC

DISTANCE PROPERTIES OF A LINEAR BLOCK CODE :- Dec-2010[8M], June/July-2012[6M], July-2014[7M]

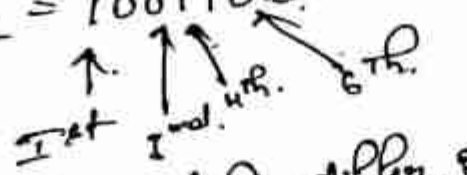
1) Hamming weight: Hamming weight of a code-vector 'C' is defined as the number of non-zero component of 'C'.

Ex:- If $C_1 = 0010110$, Then Hamming weight = number of 1's = 3 //.

2) Hamming distance: Hamming distance b/w any two code-vectors C_1 and C_2 is defined as the number of components in which they differ.

Ex: let $C_1 = 0010110$

$C_2 = 1001100$



The code-vectors C_1 and C_2 differ in 1st, 3rd, 4th and 6th positions.

∴ The Hamming distance b/w C_1 and C_2 is said to be "4".

3) minimum distance [d_{min}] :- minimum distance d_{min} is defined as the smallest Hamming distance b/w any two code vectors in a code.

* For a Systematic (6, 3) linear block code, The parity matrix "P" is given by $[P]=\begin{bmatrix}1&0&1\\0&1&1\\1&1&0\end{bmatrix}$ find all possible Code vectors.

JUNE/JULY-2012 [10M]

JUNE/JULY-2014 [6M]

Sol: Systematic (6, 3) linear code.

Given: $n=6$ $k=3$

Since $k=3$ there are $2^k = 2^3 = 8$ (message vectors)

∴ message vectors given by 000, 001, 010, 011, 100, 101, 110, 111

CODE VECTOR "C" is given by

$[C] = [D][G]$.

where GENERATOR MATRIX "G" is given by

$[G] = [I_k : P]$

$[G] = [I_3 : P] = \begin{bmatrix}1&0&0&:&1&0&1\\0&1&0&:&0&1&1\\0&0&1&:&1&1&0\end{bmatrix}$

$\therefore [C] = [D][G] = [d_1, d_2, d_3]\begin{bmatrix}1&0&0&1&0&1\\0&1&0&0&1&1\\0&0&1&1&1&0\end{bmatrix}$

$[C] = [(d_1(1)+d_2(0)+d_3(0)), (d_1(0)+d_2(1)+d_3(0)), (d_1(0)+d_2(0)+d_3(1)), (d_1(1)+d_2(0)+d_3(1)), (d_1(0)+d_2(1)+d_3(1)), (d_1(1)+d_2(1)+d_3(0))]$

$= [d_1, d_2, d_3, (d_1+d_3), (d_2+d_3), (d_1+d_2)]$

P.T.O

Let $D = (d_1, d_2, d_3) = (0, 0, 0)$

$\therefore [c_a] = [d_1, d_2, d_3, (d_1+d_3), (d_2+d_3), (d_1+d_2)]$

$= [0\ 0\ 0\ 0\ 0\ 0]$ //

Let $D = [d_1, d_2, d_3) = (001)$

$\therefore [c_b] = [d_1, d_2, d_3, (d_1+d_3), (d_2+d_3), (d_1+d_2)]$

$= [0\ 0\ 1\ (0+1)\ (0+1)\ (0+0)]$

$= [0\ 0\ 1\ 1\ 1\ 0]$ //

XOR
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0

Let $[d_1\ d_2\ d_3] = [010]$

$[c_c] = [d_1, d_2, d_3, (d_1+d_3), (d_2+d_3), (d_1+d_2)]$

$= [0\ 1\ 0\ (0+0)\ (1+0)\ (0+1)]$

$= [0\ 1\ 0\ 0\ 1\ 1]$ //

Let $[d_1\ d_2\ d_3] = [011]$

$[c_d] = [d_1, d_2, d_3, (d_1+d_3), (d_2+d_3), (d_1+d_2)]$

$= [0\ 1\ 1\ (0+1)\ (1+1)\ (0+1)]$

$= [0\ 1\ 1\ 0\ 0\ 1]$ //

Let $[d_1\ d_2\ d_3] = (100]$

$\therefore [c_e] = [1\ 0\ 0\ 1\ 0\ 1]$ //

Let $[d_1\ d_2\ d_3] = [101]$

$\therefore [c_f] = [1\ 0\ 1\ 0\ 1\ 1]$ //

Let $[d_1\ d_2\ d_3] = [110]$

$\therefore [c_g] = [1\ 1\ 0\ 1\ 1\ 0]$ //

Let $(d_1, d_2, d_3] = [111]$

$\therefore [c_h] = [1\ 1\ 1\ 0\ 0\ 0]$ //

CODE NAME	MESSAGE -VECTOR	CODE VECTOR FOR (6,3) LINEAR BLOCK CODE
C_a	000	000000
C_b	001	001110
C_c	010	010011
C_d	011	011101
C_e	100	100101
C_f	101	101011
C_g	110	110110
C_h	111	111000.

Table: Code-vectors Table for (6,3) Code //

For Example:-

Consider addition of C_c & C_g in the above Table.

$C_c + C_g = (010011) + (110110)$

$= (100101)$ which is the Code vector "C_e" //

In a Similar way, the above property can be verified for other Combinations to prove that (6,3) Code of above Example is indeed a Linear block Code. //

* For a Systematic (6,3) linear block code, the parity matrix 'P' is given by $[P] = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$. The Received Code vector $R = [110010]$. Detect and Correct the Single error that has occurred due to noise. June/July-2014 [6 M]

Sol:- $[P] = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$

$\therefore [P^T] = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} = [P]$ //

The parity check matrix [H] is given by

$$[H] = [P^T : I_{n-k}]$$

Given $n=6$ and $k=3$ for (6,3) linear block code.

$$[H] = [P^T : I_{6-3}] = [P^T : I_3]$$

$$[H] = \left[\begin{array}{ccc:ccc} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right]$$

$$[H]^T = \left[\begin{array}{ccc} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ \hdashline 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] = \begin{bmatrix} P \\ \hdashline I_{n-k} \end{bmatrix}$$ //

The Syndrome is given by

$$[S] = [S_1 S_2 S_3] = R \cdot H^T = [1\,1\,0\,0\,1\,0] \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$[S] = [(1+0+0+0+1+0), (0+1+0+0+0+0), (1+1+0+0+0+0)]$

$= [1\ 0\ 0]$ // Since $S \neq 0$ it represents an error.

The Syndrome vector $[S] = [100]$ is represented in the 4th row of H^T matrix and hence the 4th bit in the Received vector R counting from left is in error.

∴ The corresponding error-vector is then given by

$$E = [000100]$$

∴ The Corrected Code-vector $[C] = R + E$

$$C = [110010] + [000100]$$

$$= [110110]$$ //

* For the Systematic (6, 3) code, the parity matrix "P" is given by $[P] = \begin{bmatrix} 1 0 1 \\ 0 1 1 \\ 1 1 0 \end{bmatrix}$ The Received vector $R = [r_1, r_2, r_3, r_4, r_5, r_6]$. Construct the corresponding Syndrome Calculation Circuit.

Sol:- $[P] = \begin{bmatrix} 1 0 1 \\ 0 1 1 \\ 1 1 0 \end{bmatrix}$ $\therefore [P]^T = \begin{bmatrix} 1 0 1 \\ 0 1 1 \\ 1 1 0 \end{bmatrix} = [P]$

"PARITY CHECK MATRIX – H" is given by

$$[H] = [P^T : I_{n-k}] = [P^T : I_{6-3}] = [P^T : I_3]$$

$$[H] = \left[\begin{array}{ccc|ccc} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right]$$

$$[H]^T = \left[\begin{array}{ccc} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] = \begin{bmatrix} P \\ \hline I_{n-k} \end{bmatrix} //$$

"THE SYNDROME [S]" is given by

$$[S] = [S_1, S_2, S_3] = [r_1\ r_2\ r_3\ r_4\ r_5\ r_6] \begin{bmatrix} 1 0 1 \\ 0 1 1 \\ 1 1 0 \\ 1 0 0 \\ 0 1 0 \\ 0 0 1 \end{bmatrix}$$

$$= [(r_1 + r_3 + r_4),\ (r_2 + r_3 + r_5),\ (r_1 + r_2 + r_6)] //$$

∴ The Syndrome bits are

$S_1 = r_1 + r_3 + r_4$, $S_2 = r_2 + r_3 + r_5$, $S_3 = r_1 + r_2 + r_6$

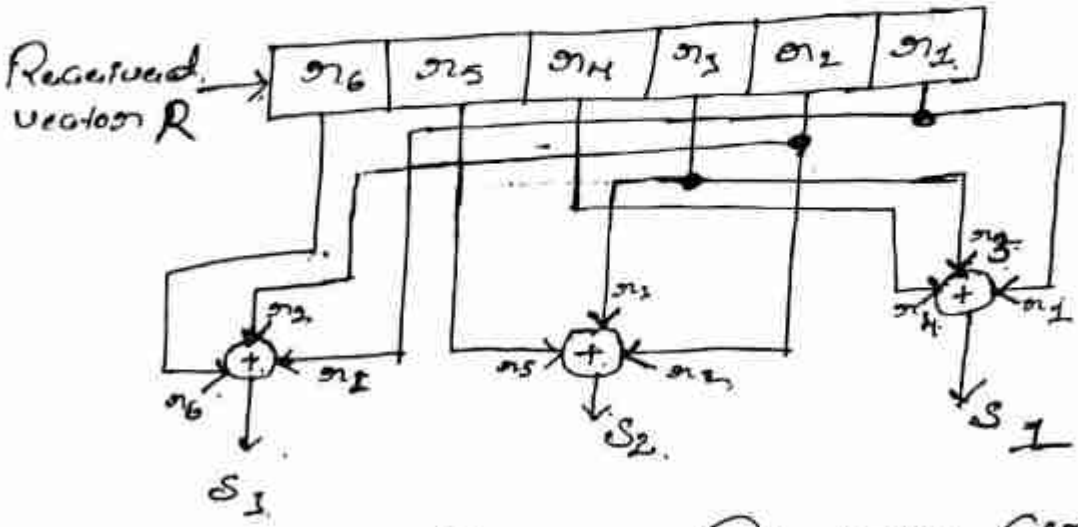


Fig :- Syndrome Calculation Circuit for (6,3) Code of above Example //

* * * IM.

* For a Systematic (7,4) linear block code, the parity matrix 'P' is given by $[P] = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$

1) Find all possible valid Code-vectors.
2) Draw the Corresponding Encoding Circuit.
3) A single error has occurred in each of these received vectors. Detect and Correct those errors.
(a) $R_A = [0111110]$ (b) $R_B = [1011100]$ (c) $R_C = [1010000]$
4) Draw the Syndrome Calculation Circuit.

DEC/Jan-2014 [14M] DEC-2012 [12M]

Sol:-

1) The generator matrix $[G]$ is given by

$$[G] = [I_k \mid P] \quad \text{where } n=7 \text{ and } k=4.$$

$$= [I_4 \mid P]$$

$$= \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right] //$$

∴ The Code vector $[C]$ is given by

$$[C] = [D][G]$$

$2^K = 2^4 = 16$ message vectors given by.

[0000], [0001], [0010], [0011], [0100], [0101], [0110], [0111], [1000], [1001], [1010], [1011], [1100], [1101], [1110], [1111]

4 p.T.O.

$\therefore [C] = [D][G]$ given $D = [d_1\ d_2\ d_3\ d_4]$

$$[C] = [d_1, d_2\ d_3] \begin{bmatrix} 1000 & 111 \\ 0100 & 110 \\ 0010 & 101 \\ 0001 & 011 \end{bmatrix}$$

$$= [d_1, d_2, d_3, d_4, (d_1+d_2+d_3), (d_1+d_2+d_4), (d_1+d_3+d_4)]$$

Let $D = [0000]$ ← Substitute

$\therefore [C] = [0000000]$ //

Let $D = [0001]$

$\therefore [C] = [0001011]$ // OR $[C] = [0001] \begin{bmatrix} 1000 & 111 \\ 0100 & 110 \\ 0010 & 101 \\ 0001 & 011 \end{bmatrix}$

$= [0001011]$ //

Let $D = [0010]$ $\therefore [C] = [0010100]$ //

Let $D = [0011]$ $\therefore [C] = [0011110]$ //

Let $D = [0100]$ $\therefore [C] = [0100110]$ //

Let $D = [0101]$ $\therefore [C] = [0101101]$ //

Let $D = [0110]$ $\therefore [C] = [0110011]$ //

Let $D = [0111]$ $\therefore [C] = [0111000]$ //

Let $D = [1000]$ $\therefore [C] = [1000111]$ //

Let $D = [1001]$ $\therefore [C] = [1001100]$ //

Let $D = [1010]$ $\therefore [C] = [1010010]$ //

Let $D = [1011]$ $\therefore [C] = [1011001]$ //

Let $D = [1100]$ $\therefore [C] = [1100001]$ //

Let $D = [1101]$ $\therefore [C] = [1101010]$ //

Let $D = [1110]$ $\therefore [C] = [1110100]$ //

Let $D = [1111]$ $\therefore [C] = [1111111]$ //

Message vector (D)	Code vector (C)	Message vector (D)	Code vector (C)
0000	0000000	1000	1000111
0001	0001011	1001	1001100
0010	0010101	1010	1010010
0011	0011110	1011	1011001
0100	0100110	1100	1100001
0101	0101101	1101	1101010
0110	0110011	1110	1110100
0111	0111000	1111	1111111

2) Code vector in terms of the message bits are found. Contg.

$$[C] = [D][G] = [d_1\ d_2\ d_3\ d_4] \begin{bmatrix} 1000111 \\ 0100110 \\ 0010101 \\ 0001011 \end{bmatrix}$$

$$= [d_1,\ d_2,\ d_3,\ d_4,\ (d_1+d_2+d_3),\ (d_1+d_2+d_4),\ (d_1+d_3+d_4)]$$

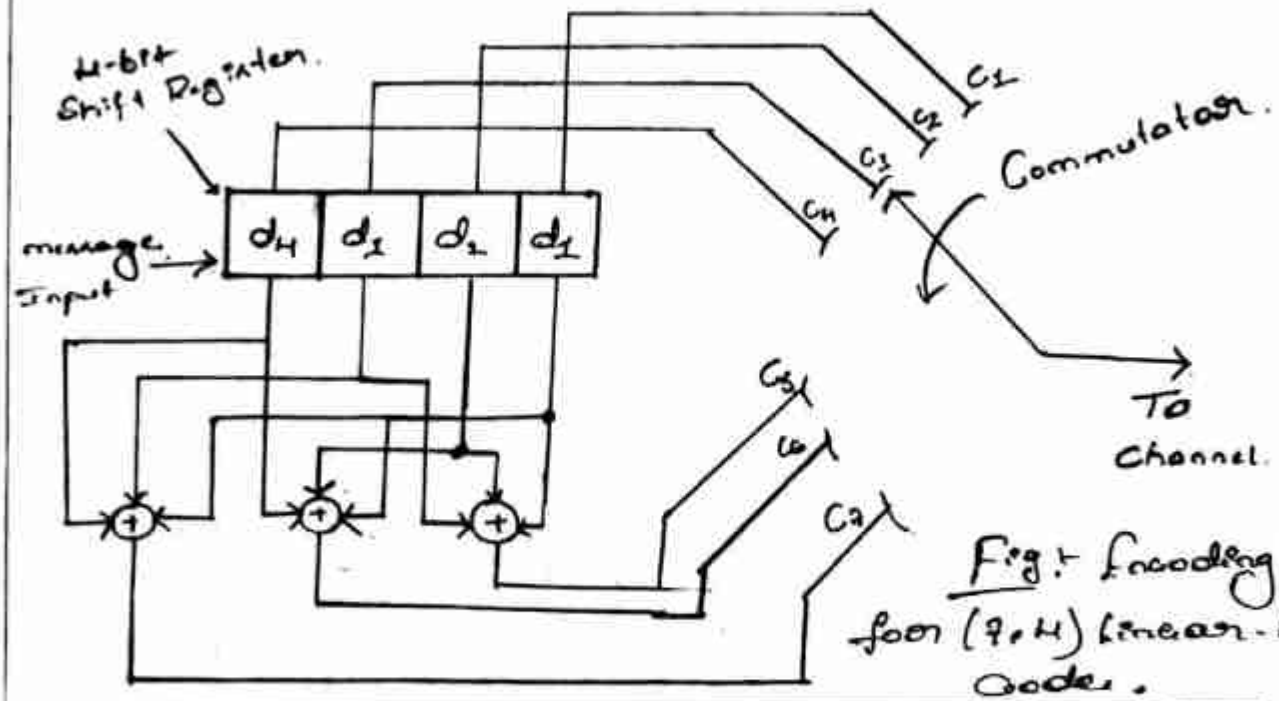


Fig: Encoding Circuit for (7,4) linear-block code.

3) The Syndrome [S] is given by

$$S = RH^T$$

∴ The parity check matrix "H" is given by

$$H = [P^T : I_{n-k}] = [P^T : I_{7-4}] = [P^T : I_3] \text{ //}$$

$$H = \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

NOTE

$$P = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$P^T = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

$$H^T = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ //}$$

$$S = RH^T$$

$$\text{i.e. } S_A = R_A H^T$$

(a) $R_A = [0\,1\,1\,1\,1\,1\,0]$

$$\therefore S_A = [0\,1\,1\,1\,1\,1\,0] \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [1\,1\,0] \text{ //}$$

This Syndrome is located in the 2nd row of H^T matrix.
Hence the 2nd bit Counting from left is in error.

The corresponding error vector is given by.

$E_A = [0100000]$

∴ the Corrected Code-vector $= C_A = R_A + E_A$

$C_A = [0111110] + [0100000]$

$= [0011110]$ // which is the valid Code-vector corresponding to the msg vector 0011 as seen above table.

(b). given $R_B = [1011100]$

$\therefore S_B = R_B H^T$

$= [1011100] \begin{bmatrix} 111 \\ 110 \\ 101 \\ 011 \\ 100 \\ 010 \\ 001 \end{bmatrix} = [101]$ which is located in the 3rd row of H^T //

∴ the error vector $E_B = [0010000]$.

∴ the Corrected Code-vector $= C_B = R_B + E_B$

$C_B = [1011100] + [0010000] = [1001100]$ //

↑ valid Code vector corresponding to the msg vector 1001 as seen from above Table //

(c) given $R_C = [1010000]$

$S_C = R_C H^T = [1010000] \begin{bmatrix} 111 \\ 110 \\ 101 \\ 011 \\ 100 \\ 010 \\ 001 \end{bmatrix} = [010]$.

this is present in the 6th row of H^T. //

∴ The error vector $E_c = [0000010]$

∴ Corrected Code-vector $C_c = R_c + E_c$

$= [1010000] + [0000010]$

$= [1010010]$ //

4) Syndrome Calculation Circuit :-

The Syndrome [S] is given by

$[S] = RH^T$

given. $R = [r_1\ r_2\ r_3\ r_4\ r_5\ r_6\ r_7]$

$$[S] = [r_1\ r_2\ r_3\ r_4\ r_5\ r_6\ r_7] \begin{bmatrix} 1\,1\,1 \\ 1\,1\,0 \\ 1\,0\,1 \\ 0\,1\,1 \\ 1\,0\,0 \\ 0\,1\,0 \\ 0\,0\,1 \end{bmatrix}$$

$S = [S_1\ S_2\ S_3] = [(r_1 + r_2 + r_3 + r_5),\ (r_1 + r_2 + r_4 + r_6),\ (r_1 + r_3 + r_4 + r_7)]$ //

7-bit Shift Register.

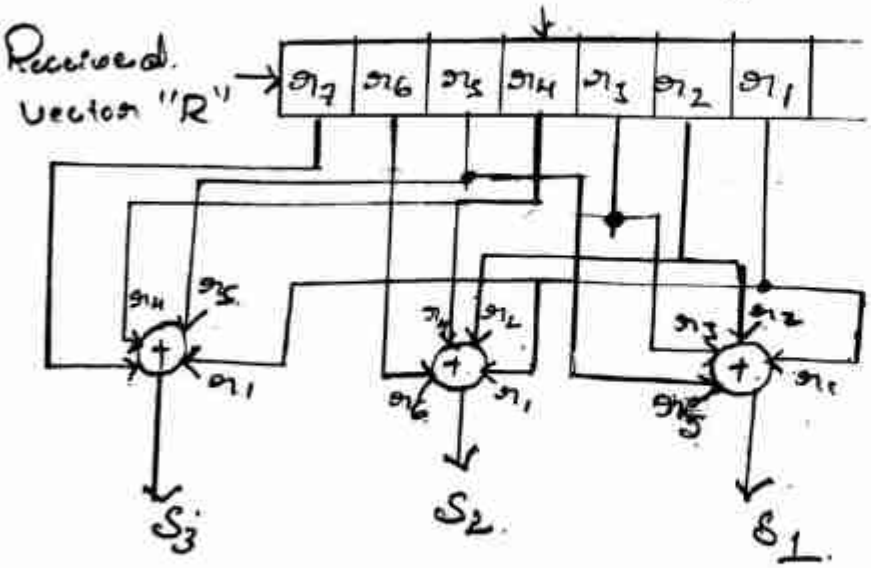


Fig:- Syndrome Calculation Circuit.

* Repetition codes represent Simplest type of linear block codes. The generator matrix of a (5,1) repetition code is given by. $[G] = [1111 \,!\, 1]$.

1) Write its parity check matrix

2) Evaluate the Syndrome for all five possible Single error patterns and also for all ten possible double error patterns

Sol:- given. $n=5$ $k=1$ for (5,1) repetition code.

1) * Generator matrix "G" is given by

$$[G] = [1111 \,!\, 1]$$

$$[G] = [P \,!\, I_k]$$

$$[G] = [P \,!\, I_1] \;//$$

$[P] = [1111]$ // $I_k = 1$ // $P^T = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$ //

* The parity check matrix [H] is given by.

$$[H] = [I_{n-k} \,!\, P^T]$$

$$= [I_{5-1} \,!\, P^T] = [I_4 \,!\, P^T]$$

$$[H] = \left[\begin{array}{cccc|c} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right] \;//$$

$$[H]^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

2) Since. $k=1$, There are only two valid Code-words. which are found using $[c] = [D][G]$.

∴ The message-vector $[D]$ Can be Either $[0]$ (or) $[1]$

When $[D]=0$.

$$[c] = [0][11111] = [00000] //$$

$D=1$

$$[c] = [1][11111] = [11111] //$$

Let the Transmitted vector be $[00000]$. Then there are five Single-error patterns given by $[10000]$, $[01000]$, $[00100]$, $[00010]$, and $[00001]$.

The Syndrome $[S]$ is given by

$$[S] = [R][H^T]$$

given $R_A = [10000]$.

$$\therefore S_A = R_A H^T = [10000] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [1000] //$$

$R_B = [01000]$

$$\therefore S_B = R_B H^T = [01000] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [0100] //$$

$R_C = [00100]$

$$\therefore S_C = R_C H^T = [00100] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [0010] //$$

$R_D = [00010]$

$\therefore S_D = R_D H^T = [00010] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [0001]$ //

$R_E = [00001]$

$\therefore S_E = R_E H^T = [00001] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [1111]$ //

$\therefore$ There are ten double error patterns given by [11000], [10100], [10010], [10001], [01100], [01010], [01001], [00110], [00101] and [00011]. The syndromes for these "10" double error patterns can be similarly found as below.

$R_F = [11000]$, $\therefore S_F = R_F H^T = [11000] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [1100]$ //

$R_G = [10100]$ $\therefore S_G = R_G H^T = [10100] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [1010]$ //

$R_H = [10010]$ $\therefore S_H = R_H H^T = [10010] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [1001]$ //

$R_I = [10001]$ $\therefore S_I = R_I H^T = [10001] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [0111]$ //

$R_J = [01100]$ $\therefore S_J = R_J H^T = [01100] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [0110]$ //

$R_K = [01010] \therefore S_K = R_K H^T = [01010] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [0101]$ //

$R_L = [01001] \therefore S_L = R_L H^T = [01001] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [1011]$ //

$R_M = [00110] \therefore S_M = R_M H^T = [00110] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [0011]$ //

$R_N = [00101] \therefore S_N = R_N H^T = [00101] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [1101]$ //

$R_O = [00011] \therefore S_O = R_O H^T = [00011] \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1111 \end{bmatrix} = [1110]$ //.

* Consider a (6,3) linear code whose generator matrix is $G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$ FEB-2002(10M) DEC-2010(10M)

1) Find all code vectors
2) Find all the hamming weights and distances.
3) Find minimum weight parity check matrix.
4) Draw the encoder circuit for the above codes.

Sol:- For (6,3) code, $k=3$ and $n=6$.

There are $2^k = 2^3 = 8$ message vectors given by (000), (001), (010), (011), (100), (101), (110), and (111).

1) The code vector is given by

$$[C] = [D][G] = [d_1\ d_2\ d_3] \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$[C] = [d_1, d_2, d_3, (d_1+d_2), (d_2+d_3), (d_1+d_3)]$$

$(d_1 d_2 d_3) = (000)$ $[C_a] = [000000]$ //

$(d_1 d_2 d_3) = (001)$ $[C_b] = [001011]$ //

$(d_1 d_2 d_3) = (010)$ $[C_c] = [010110]$ //

$(d_1 d_2 d_3) = (011)$ $[C_d] = [011101]$ //

$(d_1 d_2 d_3) = (100)$ $[C_e] = [100101]$ //

$(d_1 d_2 d_3) = (101)$ $[C_f] = [101110]$ //

$(d_1 d_2 d_3) = (110)$ $[C_g] = [110011]$ //

$(d_1 d_2 d_3) = (111)$ $[C_h] = [111000]$ //

2)

Code name.	message vector.	Code vector	Hamming weights.
C_a	000	000000	0
C_b	001	001011	3
C_c	010	010110	3
C_d	011	011101	4
C_e	100	100101	3
C_f	101	101110	4
C_g	110	110011	4
C_h	111	111000	3

Table: Hamming weight of Code-vector

The distance are given by

$d_{ab} = C_a + C_b = 000000 + 001011 = 001011 = 3$//

$d_{ac} = C_a + C_c = 000000 + 010110 = 010110 = 3$//

$d_{ad} = C_a + C_d = 000000 + 011101 = 011101 = 4$//

$d_{ae} = C_a + C_e = 000000 + 100101 = 100101 = 3$//

$d_{af} = 101110 = 4$// $d_{ag} = 110011 = 4$// $d_{ah} = 111000 = 3$//

$d_{bc} = C_b + C_c = 001011 + 010110 = 011101 = 4$//

$d_{bd} = C_b + C_d = 001011 + 011101 = 010110 = 3$//

$d_{be} = C_b + C_e = 001011 + 100101 = 101110 = 4$//

$d_{bf} = C_b + C_f = 001011 + 101110 = 100101 = 3$//

$d_{bg} = C_b + C_g = 001011 + 110011 = 111000 = 3$//

$d_{bh} = C_b + C_h = 001011 + 111000 = 110011 = 3$//

$d_{cd} = 010110 + 011101 = 001011 = 3//$

$d_{ce} = 010110 + 100101 = 110011 = 4//$

$d_{cf} = C_c + C_f = 010110 + 101110 = 111000 = 3//$

$d_{cg} = C_c + C_g = 010110 + 110011 = 100101 = 3//$

$d_{ch} = C_c + C_h = 010110 + 111000 = 101110 = 4//$

$d_{de} = C_d + C_e = 011101 + 100101 = 111000 = 3//$

$d_{df} = C_d + C_f = 011101 + 101110 = 110011 = 4//$

$d_{dg} = C_d + C_g = 011101 + 110011 = 101110 = 4//$

$d_{dh} = C_d + C_h = 011101 + 111000 = 100101 = 3//$

$d_{ef} = C_e + C_f = 100101 + 101110 = 001011 = 3//$

$d_{eg} = C_e + C_g = 100101 + 110011 = 010110 = 3//$

$d_{eh} = C_e + C_h = 100101 + 111000 = 011101 = 4//$

$d_{fg} = C_f + C_g = 101110 + 110011 = 011101 = 4//$

$d_{fh} = C_f + C_h = 101110 + 111000 = 010110 = 4//$

$d_{gh} = C_g + C_h = 110011 + 111000 = 001011 = 3//$

3) $G_2 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$

$[G_2] = [I_3 : P]$ //

$\therefore I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ // and $P = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$ //

$\therefore P^T = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$ //

The systematic weight parity check matrix is given by

$$[H] = [P^T \mid I_{n-k}] = [P^T \mid I_{6-3}] = [P^T, I_3]$$

NOTE: $n = 6$, $k = 3$

$$\therefore [H] = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

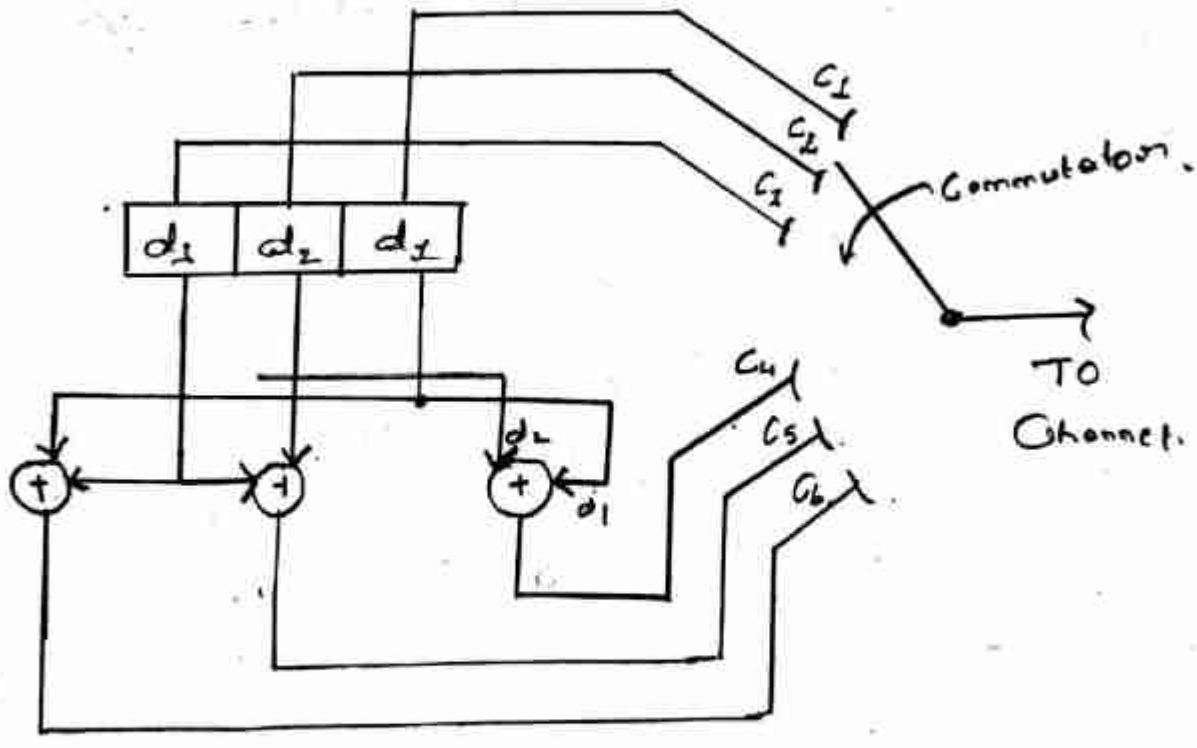
4) Code vector is given by

$$[G] = [D][G_1]$$

Given. $D = d_1\, d_2\, d_3$.

$$[G] = [d_1\, d_2\, d_3] \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$= [d_1,\ d_2,\ d_3,\ (d_1+d_2),\ (d_2+d_3),\ (d_1+d_3)]$$



* Consider the (7,4) LBC whose generation matrix is given below. Find
1) all code words with Hamming weights.
2) parity check matrix

$$G = \begin{bmatrix} 1000 & 101 \\ 0100 & 111 \\ 0010 & 110 \\ 0001 & 011 \end{bmatrix}$$

JUNE/JULY-2010

Sol :-

1) The code vectors. $[C] = [D][G]$

given $[D] = d_1 d_2 d_3 d_4$.

$$[C] = [d_1 d_2 d_3 d_4] \begin{bmatrix} 1000 & 101 \\ 0100 & 111 \\ 0010 & 110 \\ 0001 & 011 \end{bmatrix}$$

$$[C] = [d_1, d_2, d_3, d_4, (d_1+d_2+d_3), (d_2+d_3+d_4), (d_1+d_3+d_4)]$$

Since $k=3$ there are $2^k = 2^4 = 16$ message vectors given by (0000), (0001), (0010), (0011), (0100), (0101), (0110), (0111), (1000), (1001), (1010), (1011), (1100), (1101), (1110), (1111).

Given $[D] = [d_1 d_2 d_3] = [0000]$

$\therefore C_a = [D][G]$

$= [0000000]$

given $[D] = [d_1 d_2 d_3] = [0001]$

$\therefore C_b = [0001011]$

given $[D] = [d_1 d_2 d_3] = [0010]$

$\therefore C_c = [0010110]$

Similarly

Name.	message	Code vector.	Hamming weight.
C_a	0 0 0 0	0 0 0 0 0 0 0 0.	0
C_b	0 0 0 1	0 0 0 1 0 1 1	3
C_c	0 0 1 0	0 0 1 0 1 1 0	3
C_d	0 0 1 1	0 0 1 1 1 0 1	4
C_e	0 1 0 0	0 1 0 0 1 1 1	4
C_f	0 1 0 1	0 1 0 1 1 0 0	3
C_g	0 1 1 0	0 1 1 0 0 0 1	3
C_h	0 1 1 1	0 1 1 1 0 1 0	4
C_i	1 0 0 0	1 0 0 0 1 0 1	3
C_j	1 0 0 1	1 0 0 1 1 1 0	4
C_k	1 0 1 0	1 0 1 0 0 1 1	4
C_l	1 0 1 1	1 0 1 1 0 0 0	3
C_m	1 1 0 0	1 1 0 0 0 1 0	3
C_n	1 1 0 1	1 1 0 1 0 0 1	4
C_o	1 1 1 0	1 1 1 0 1 0 0	4
C_p	1 1 1 1	1 1 1 1 1 1 1 1	7.

Table : Hamming weight of Code-vectors.

2)

$$G = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

$$= [I_K \mid P]$$

$$= [I_4 \mid P] \quad //.$$

given

$$I_k = I_4 = \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix}.$$

and $P = \begin{bmatrix} 101 \\ 111 \\ 110 \\ 011 \end{bmatrix}$ and $P^T = \begin{bmatrix} 1110 \\ 0111 \\ 1101 \end{bmatrix}$ //.

*. Considered a (6, 3) linear block code whose generator matrix is given below.

$$G = \begin{bmatrix} 100111 \\ 010110 \\ 001101 \end{bmatrix}$$

Find. 1) all code words.
2) all Hamming weights and distance.
3) parity check matrix.
4) Draw Encoder Circuit.

JUNE/JULY-2011 [14M]

Sol :- given. $n=6$. $k=3$ //.

Since $k=3$ There are $2^k = 2^3 = 8$ (message vector)

∴ message vectors given by 000, 001, 010, 011, 100, 101, 110, 111

1) CODE VECTOR "C" is given by.

$$[C] = [D][G]$$

given $[D] = [d_1\ d_2\ d_3]$

$$[C] = [d_1\ d_2\ d_3] \begin{bmatrix} 100111 \\ 010110 \\ 001101 \end{bmatrix}$$

$\therefore [C] = [d_1, d_2, d_3, (d_1+d_2+d_3), (d_1+d_2), (d_1+d_3)]$

given. $[d_1\ d_2\ d_3] = [000]$

$\therefore C_a = [000000]$ //

given $[d_1\ d_2\ d_3] = [001]$

$C_b = [001.101]$ //

114

Code words	message vector.	Code vector.	Hamming weight
C_a.	000	000000	0
C_b	001	001101	3
C_c	010	010110	3
C_d	011.	011011	4.
C_e	100	100111	4
C_f.	101.	101010	3.
C_g.	110	110001	3
C_h.	111.	111111.	4.

Table : Code vector. Table for. (7, 4) Hamming Code.

2) The distances are given by.

$d_{ab} = C_a + C_b = 000000 + 001101 = 001101 = 3$ //

$d_{ac} = C_a + C_c = 000000 + 010110 = 010110 = 3$ //

$d_{ad} = C_a + C_d = 000000 + 011011 = 011011 = 4$ //

114. $d_{ae} = C_a + C_e = 100111 = 4$ //

$d_{af} = C_a + C_f = 101010 = 3$ //

$d_{ag} = C_a + C_g = 110001 = 3$ //

$d_{ah} = C_a + C_h = 111111 = 6$ //

$d_{bc} = C_b + C_c = 011011 = 4$ //

$d_{bd} = C_b + C_d = 010110 = 3$ //

$d_{be} = C_b + C_e = 100010 = 3$ //

$d_{bf} = C_b + C_f = 100111 = 4$ //

$d_{bg} = C_b + C_g = 111100 = 4$ //

$d_{bh} = C_b + C_h = 110010 = 3$ //

$d_{cd} = C_c + C_d = 001101 = 3$ //

$d_{ce} = C_c + C_e = 110001 = 3$ //

$d_{cf} = C_c + C_f = 111010 = 4$ //

$d_{cg} = C_c + C_g = 100111 = 4$ //

$d_{ch} = C_c + C_h = 101001 = 3$ //

$d_{de} = C_d + C_e = 111100 = 4$ //

$d_{df} = C_d + C_f = 110001 = 3$ //

$d_{dg} = C_d + C_g = 101010 = 3$ //

$d_{dh} = C_d + C_h = 100100 = 2$ //

$d_{ef} = C_e + C_f = 001101 = 3$ //

$d_{eg} = C_e + C_g = 010110 = 3$ //

$d_{eh} = C_e + C_h = 011000 = 2$ //

$d_{fg} = C_f + C_g = 011011 = 4$ //

$d_{fh} = C_f + C_h = 010101 = 3$ //

$d_{gh} = C_g + C_h = 001110 = 3$ //

3) parity check matrix $[H]$ is given by

$$[H] = [P^T : I_{n-k}]$$

$$= [P^T : I_3]$$

$$[H] = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

NOTE

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

I_3 P

$$P = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \quad P^T = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

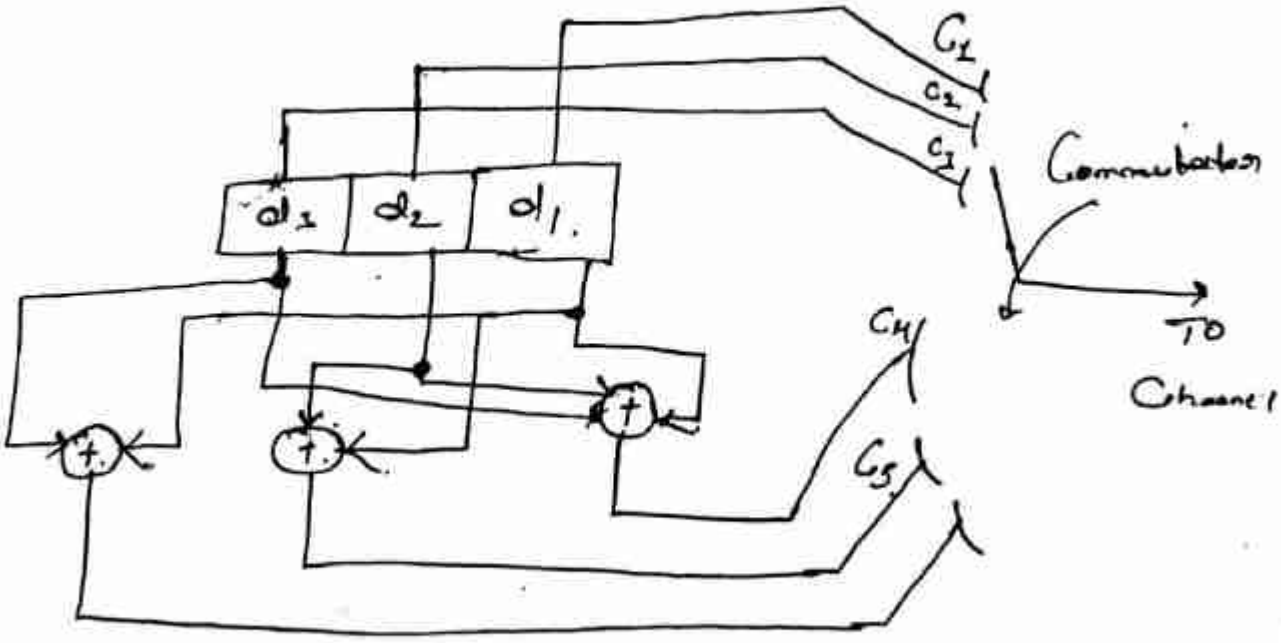
4) Code vectors $[C]$ is given by

$$[C] = [D] \cdot [G]$$

given $[D] = [d_1\ d_2\ d_3]$

$$[C] = [d_1\ d_2\ d_3] \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

$$= d_1\ d_2\ d_3\ (d_1+d_2+d_3),\ (d_1+d_2),\ (d_1+d_3)$$



* The parity Check bits of a (7,4) Hamming Code are generated by $C_5 = d_1 + d_3 + d_4$

$C_6 = d_2 + d_3 + d_4$

$C_7 = d_2 + d_1 + d_4$.

Where d_1, d_2, d_3 and d_4 are the message bits.

(a) find the generator matrix [G] and parity check matrix [H] for this Code.

(b) prove that $GH^T = 0$

(c). The (n,k) linear block Code so obtained has a "dual" Code. This dual Code is an (n, n-k) Code having a generator matrix "H" and parity check matrix G. Determine the Eight Code vector of the "dual Code" for the (7,4) Hamming Code discussed above

(d) find the minimum distance of the dual Code. determined in part (c)

June/July-2016 [10M]

Sol :-

(a) The Code vectors are given by

$[C] = [C_1, C_2, C_3, C_4, C_5, C_6, C_7]$

$= [d_1, d_2, d_3, d_4, (d_1+d_3+d_4), (d_1+d_2+d_3), (d_2+d_3+d_4)]$

$$[C] = \left[\begin{array}{cccc:ccc} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]$$

$[C] = [D].[G]$ //

∴ The generator matrix [G] is given by

$$[G] = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

$$= [I_K : P] = [I_4 : P] \text{ //}$$

where $I_K = I_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ //, $P = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$ and $P^T = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$ //

The parity check matrix "H" is given by.

$$[H] = [P^T : I_{n-k}]$$

$$= [P^T : I_{7-4}] = [P^T : I_3]$$

$$[H] = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$ //.

and $H^T = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ //.

b) $[G][H^T] = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$ //

∴ $[G][H^T] = [0]$ //

(c) Dual Code:-

The message vector length for the $(n, n-k)$ dual code is given by $n-k = 7-4 = 3$ bits.

$\therefore (n, n-k) = [7, 3]$

given $n = 7$ and $n-k = 3$

Since $n-k = 3$ there are $2^{n-k} = 2^3 = 8$ message vector message vector is given by 000, 001, 010, 011, 100, 101, 110, 111

The Code vector $[c]$ is given by.

$$[c] = [D][H]$$

given $[D] = d_1' d_2' d_3'$

$$[c] = [d_1' d_2' d_3'] \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

$$= \left[(d_1' + d_2'), (d_2' + d_3'), (d_1' + d_2' + d_3'), (d_1' + d_3'), d_1', d_2', d_3' \right]$$

proof:

Codevector $[c]$ is given by $[c] = [D][G]$ → linear block code.

$[c] = [D][H]$ ↑ dual code method.

Ex:-

given $(d_1' d_2' d_3') = (111)$

$\therefore C = [0\ 0\ 1\ 0\ 1\ 1\ 1]$

given $(d_1' d_2' d_3') = (110)$

$\therefore C = [0\ 1\ 0\ 1\ 1\ 1\ 0]$

Message vector.	Code-vector.	Hamming weight (H_w)	message vector.	Code vector.	Hamming weight (H_w)
000	0000000	0	100	1011100	4
001	0111001	4	101	1100101	4
010	1110010	4	110	0101110	4
011	1001011	4.	111	0010111	4

Table: Code-vector table for dual Code.

(d) The minimum distance of the dual Code is Equal to the minimum Hamming weight.

$$d_{min} = H_{w_{min}} = 4 //$$

* The parity check bits of a (8,4) block code are generated by.

$C_5 = d_1 + d_2 + d_4$.

$C_6 = d_1 + d_3 + d_4$.

$C_7 = d_1 + d_2 + d_4$

$C_8 = d_2 + d_3 + d_4$.

where d_1 d_2 d_3 and d_4 are the message bits.

a) find the generator matrix and parity check matrix for this code.

b) find the minimum weight of this code.

c) Show that it is capable of correcting all single error patterns and capable of detecting all double errors by preparing the Syndrome table for them.

Jan/Feb-2003 (10M) Jan/Feb-2006 (14M) Jan-2017 [10M]

Sol :-

The code vector [C] is given by.

$$[C] = [D]\,[G]$$

$$C_1\ C_2\ C_3\ C_4\ C_5\ C_6\ C_7\ C_8$$

Since $[C] = d_1\ d_2\ d_3\ d_4\ (d_1+d_2+d_4)\ (d_1+d_3+d_4)\ (d_1+d_2+d_3)\ (d_2+d_3+d_4)$

$$= [d_1\ d_2\ d_3\ d_4]$$

$$= (d_1\ d_2\ d_3\ d_4)\begin{bmatrix} 1&0&0&0 & 1&1&1&0 \\ 0&1&0&0 & 1&1&1&1 \\ 0&0&1&0 & 1&0&1&1 \\ 0&0&0&1 & 1&1&0&1 \end{bmatrix}$$

$$[C] = [D]\,[G]$$ //

a) generator matrix is given by

$$[G] = \left[\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{array}\right]$$

$$[G] = [I_K \,|\, P] = [I_4 \,|\, P] \quad //$$

where $P = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$ $\therefore P^T = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$ //

The parity check matrix is given by.

$$[H] = [P^T \,|\, I_{n-k}] = [P^T \,|\, I_H]$$

$$[H] = \left[\begin{array}{cccc|cccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right] \quad //$$

$$\therefore H^T = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

b) Code vector $[C]$ is given by.

$$[C] = [D][G] = (d_1 d_2 d_3 d_4) \left[\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{array}\right]$$

$$= d_1, d_2 d_3 d_4 (d_1+d_2+d_3)(d_1+d_2+d_3)(d_1+d_2+d_4)(d_2+d_3+d_4)$$

Since $K=3$ There are $2^K = 2^4 = 16$ message vectors given by 0000 ---------- 1111 //

Ex:- $(d_1 d_2 d_3 d_4) = (0000)$

$\therefore C_a = (0000.0000)$

$(d_1 d_2 d_3 d_4) = (0001)$

$C_b = (00011011)$ //

message vector	Code vector.	Hamming weight	message vector	Code vector.	Hamming weight
0000	00000000	0	1000	10001110	4
0001	00011011	4	1001	10010101	4
0010	00100111	4	1010	10101001	4
0011	00111100	4	1011	10110010	4
0100	01001101	4	1100	11000011	4
0101	01010110	4	1101	11011001	5
0110	01101010	4	1110	11100101	5
0111	01110001	4	1111	11111111	5

The minimum distance of the dual code is equal to the minimum Hamming weight

$\therefore d_{min} = H_{wmin} = [4]$ //

NOTE
0 → replace

c) Single error pattern.

Example: Syndrome 'S' is given by

$[S] = RH^T$.

given. $R = [10000000]$ //

$\therefore S = [10000000] \begin{bmatrix} 1110 \\ 1101 \\ 0111 \\ 1011 \\ 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix} = [1110]$ // Ans.

Given $R = [01000000]$

$\therefore S = [01000000] \begin{bmatrix} 1110 \\ 1101 \\ 0111 \\ 1011 \\ 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix} = [1101]$ //

Hly.

Single error pattern.	Syndrome
1000 0000	1110
0100 0000	1101
0010 0000	0111
0001 0000	1011
0000 1000	1000
0000 0100	0100
0000 0010	0010
0000 0001	0001

Table: Syndrome table for Single error pattern.

P.T.O

Double error pattern,

Example: given $R = [11000000]$

$$[s] = [11000000] \begin{bmatrix} 1110 \\ 1101 \\ 0111 \\ 1011 \\ 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix} = [0011]$$

given $R = [10100000]$

$$[s] = [10100000] \begin{bmatrix} 1110 \\ 1101 \\ 0111 \\ 1011 \\ 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix} = 1001$$

Ily.

Double-error pattern	Syndrome	Double-error pattern	Syndrome
11000000	0011	00101000	1111
10100000	1001	00100100	0011
10010000	1001	00100010	0101
10001000	0110	00100001	0110
10000100	1010	00011000	0011
10000010	1100	00010100	1111
10000001	1111	00010010	1001
01100000	1010	00010001	1010
01010000	0110	00001100	1100
01001000	0101	00001010	1010
01000100	1001	00001001	1001
01000010	1111	00000110	0110
01000001	1100	00000101	0101
00111000	1100	00000011	0011

Table: Syndrome table for double-error pattern.

* The parity check bits of a (8,4) LBC are given by

$V_5 = u_1 + u_2 + u_4$

$V_6 = u_1 + u_2 + u_3$

$V_7 = u_1 + u_3 + u_4$

$V_8 = u_2 + u_3 + u_4$.

Where $u_1 u_2 u_3 u_4$ are data bits.

(1) Find generator and parity check matrix.

(2) Find all possible Code vectors

(3) Find the minimum Hamming weight

Sol:-

The Codevector $[v]$ is given by $[v] = [u]\,[G]$.

Since $[v] = v_1\ v_2\ v_3\ v_4\ v_5\ v_6\ v_7\ v_8$

$[v] = u_1\ u_2\ u_3\ u_4\ (u_1+u_2+u_4)\ (u_1+u_2+u_3)\ (u_1+u_3+u_4)\ (u_2+u_3+u_4)$

$$[v] = u_1\ u_2\ u_3\ u_4 \begin{bmatrix} 1&0&0&0&1&1&1&0 \\ 0&1&0&0&1&1&0&1 \\ 0&0&1&0&0&1&1&1 \\ 0&0&0&1&1&0&1&1 \end{bmatrix}$$

$\therefore [v] = [u]\,[G]$ //

where $[u] = u_1 u_2 u_3 u_4$ and $[G] = \begin{bmatrix} 1&0&0&0&1&1&1&0 \\ 0&1&0&0&1&1&0&1 \\ 0&0&1&0&0&1&1&1 \\ 0&0&0&1&1&0&1&1 \end{bmatrix}$ //

Note

Code vectors is given by

generator matrix (IIC) book $[G] = [P]\,[I_4]$ or

$\Rightarrow [V] = [U]\,[G]$

IIC (Shivaprakash) book.

1) * Generator matrix $[G]$ is.

$$[G] = \begin{bmatrix} 1000 & 1110 \\ 0100 & 1101 \\ 0010 & 0111 \\ 0001 & 1011 \end{bmatrix}$$

$$\therefore [G] = [I_k \mid P] = [I_4 \mid P]$$

where $I_4 = \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix}$ and $P = \begin{bmatrix} 1110 \\ 1101 \\ 0111 \\ 1011 \end{bmatrix}$, $P^T = \begin{bmatrix} 1101 \\ 1110 \\ 1011 \\ 0111 \end{bmatrix}$ //

* parity check matrix is given by

$$[H] = [P^T \mid I_{n-k}] = [P^T \mid I_{8-4}]$$

$$[H] = \begin{bmatrix} 1101 & 1000 \\ 1110 & 0100 \\ 1011 & 0010 \\ 0111 & 0001 \end{bmatrix}$$ //

2) * Code vector $[V]$ is given by.

$$[V] = [U][G]$$

$$= [u_1 u_2 u_3 u_4] \begin{bmatrix} 1000 & 1110 \\ 0100 & 1101 \\ 0010 & 0111 \\ 0001 & 1011 \end{bmatrix}$$

$$= [u_1 u_2 u_3 u_4 \; (u_1+u_2+u_4) \; (u_1+u_2+u_3) \; (u_1+u_3+u_4) \; (u_2+u_3+u_4)]$$ //

Since $k=3$ there are $2^k = 2^4 = 16$ message vectors given by 0000 ——— 1111.

Eg: given $(b_1 b_2 b_3 b_4) = [0000]$

$\therefore [v] = [0000.0000]$ //

$[v_1 v_2 v_3 v_4] = [0001]$

$\therefore [v] = [000110111]$ //

Ily.

Message vectors.	Codevectors.	Hamming weight.
0000	00000000	-
0001	00011011	4
0010	00101111	4
0011	00111100	4
0100	01001101	4
0101	01010110	4
0110	01100001	4
0111	01111010	4
1000	10001110	4
1001	10010101	4
1010	10101001	4
1011	10110010	4
1100	11000011	4
1101	11011000	4
1110	11100100	4
1111	11111111	8

3) $d_{min} = H_{min} = 4$ //

* In a linear block code the Syndrome is given by.

$S = r_1 + r_2 + r_3 + r_5$

$S_2 = r_1 + r_2 + r_4 + r_6$

$S_3 = r_1 + r_3 + r_4 + r_7$.

1) Find the parity check matrix [H]
2) Draw the Encoder circuit.
3) Find the Codeword for all input Sequences
4) How many errors it can detect and Correct?
5) What is the Syndrome for the Received data 1011011?

Dec-2012 [10M] June/July-2013 [14M]

Dec-2010/June-2014 [12M]

Sol:-

1) Syndrome 'S' is given by

$S = rH^T$ (or) $RH^T = [r_1 r_2 r_3 r_4 r_5 r_6 r_7]$

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$\therefore [s] = [s_1 s_2 s_3] = [(r_1 + r_2 + r_3 + r_5), (r_1 + r_2 + r_4 + r_6), (r_1 + r_3 + r_4 + r_7)]$ //

Therefore $H^T = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$ //

2) We have.

$$H = \left[\begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

$$H = [P^T \mid I_{n-k}]$$

$$\therefore P^T = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \quad \therefore P = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Generator matrix 'G' is given by

$$[G] = [I_K \mid P] = \left[\begin{array}{c|ccc} & 1 & 1 & 1 \\ I_K & 1 & 1 & 0 \\ & 1 & 0 & 1 \\ & 0 & 1 & 1 \end{array}\right]$$

It is clear that "P" is H matrix means Identity matrix is H Comes H matrix.

$\therefore I_K = I_H$ & $K = H$ (number of msg bits)

$$[G] = [I_K \mid P] = [I_H \mid P] \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

Since K=4. There are $2^K = 2^4 = 16$ message vectors given by (0000) ---------- (1111).

$$[C] = [D][G]$$

$$= (d_1\ d_2\ d_3\ d_4) \begin{bmatrix} 1000 & 111 \\ 0100 & 110 \\ 0010 & 101 \\ 0001 & 011 \end{bmatrix}$$

$$= [d_1,\ d_2\ d_3\ d_4\ (d_1+d_2+d_3)\ (d_1+d_2+d_4)\ (d_1+d_3+d_4)] \quad //$$

given $(d_1\ d_2\ d_3\ d_4) = (0000)$

$\therefore C = [0000000]$ //

given $(d_1\ d_2\ d_3\ d_4) = (0001)$

$\therefore C = [0001011]$ //

⋮

given $(d_1\ d_2\ d_3\ d_4) = (1111)$

$\therefore C = [1111111]$ //

message vectors.	Code Vectors.	Hamming Weight	MSG vectors	Code vectors.	Hamming Weight
0000	0000 000	—	1000	1000 111	4
0001	0001 011	3	1001	1001 100	3
0010	0010 101	3	1010	1010 010	3
0011	0011 110	4	1011	1011 001	4
0100	0100 110	3	1100	1100 001	3
0101	0101 101	4	1101	1101 010	4
0110	0110 011	4	1110	1110 100	4
0111	0111 000	3	1111	1111 111	7

3) Code vector [C] is given by

$$[C] = [D][G]$$

$$[C] = [d_1\ d_2\ d_3\ d_4] \begin{bmatrix} 1000 & 111 \\ 0100 & 110 \\ 0010 & 101 \\ 0001 & 011 \end{bmatrix}$$

Note:
It is clear that.
number of msg bits
$k = 3$ //.

$$[C] = [d_1\ d_2\ d_3\ d_4\ (d_1+d_2+d_3)\ (d_1+d_2+d_4)]$$

$\therefore C_1 = d_1,\ C_2 = d_2,\ C_3 = d_3,\ C_4 = d_4,\ C_5 = d_1+d_2+d_3$

$C_6 = d_1+d_2+d_3,\ C_7 = d_1+d_2+d_4.$

↑
It is clear that the length of the Codeword is "7"
so that $n = 7$ //.

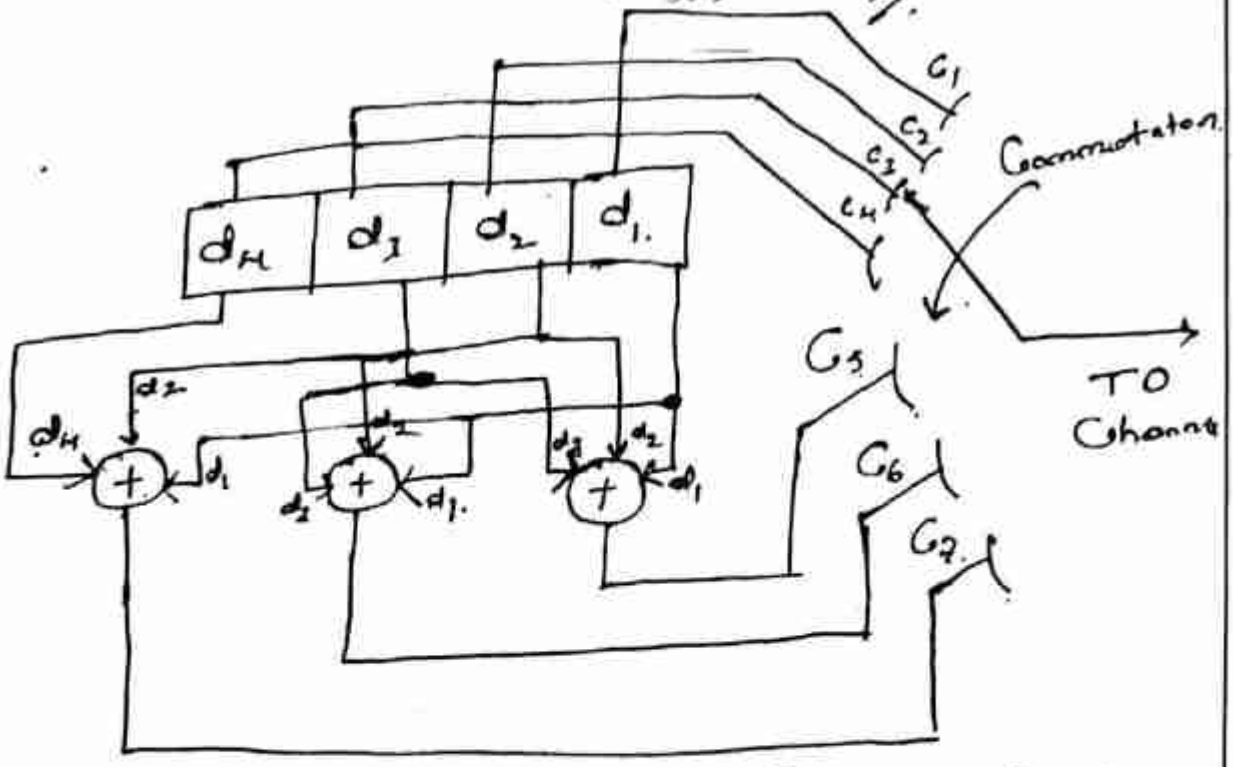


Fig:- Encoder Circuit for the given (7, 4) LBC

4) Since $d_{min} = (HD)_{min}$.

$= 3$ //

$\therefore$ Error Correcting Capability $t = \frac{d_{min} - 1}{2}$

$= \frac{3-1}{2} = 1$ bit error

5) Syndrome is given as.

$$S = R H^T = [1011011] \begin{bmatrix} 111 \\ 110 \\ 101 \\ 011 \\ 100 \\ 010 \\ 001 \end{bmatrix} = [010]$$

Error vector $E = [0000010]$ //

Correct vector $C = R + E$

$= [1011011] + [0000010]$

$= 1011001$ //

*. The generator matrix of a linear block code is given by. $[G] = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$

1) Find all possible valid code vectors
2) Find parity check matrix.
3) Find minimum distance.
4) Draw Encoding Circuit
5) Draw Syndrome Calculation Circuit

Sol:- $[G] = \left[\begin{array}{ccc|cccc} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{array}\right]$

$[G] = [I_K \mid P]$

$= [I_3 \mid P]$ //

no of message bits $K = 3$ //

1) code vector is given by

$[C] = [D][G]$

$= [d_1 d_2 d_3] \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$

$[C] = [d_1\ d_2\ d_3\ (d_1+d_3)\ (d_1+d_2+d_3)\ (d_1+d_2)\ (d_2+d_3)]$

$\therefore C_1 = d_1,\ C_2 = d_2,\ C_3 = d_3,\ C_4 = (d_1+d_3),\ C_5 = (d_1+d_2+d_3)$
$C_6 = (d_1+d_2),\ C_7 = (d_2+d_3)$

↑

It is clear that, the code length 'n' = 7 //

Since k=3 There are $2^k = 8$ message vectors given by (000), (001), (010), (011), (100), (101), (110) and (111)

Message vector	Cod. vector	Hamming weight	Msg vector	Code vector	Hamming weight
000	0000000	0	101	1010011	4
001	0011101	4	110	1101001	4
010	0100111	4	111	1110100	4
011	0111010	4			
100	1001110	4			

2). parity check matrix [H] is given by

$$[H] = [P^T \mid I_{n-k}] = [P^T \mid I_H]$$

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

$$G = [I_K \mid P]$$

where $P = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$ $\quad P^T = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$

$$\therefore [H] = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad H^T = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3). From the above Table. We observe that The minimum Hamming weight H_N of a non-zero code vector is 4

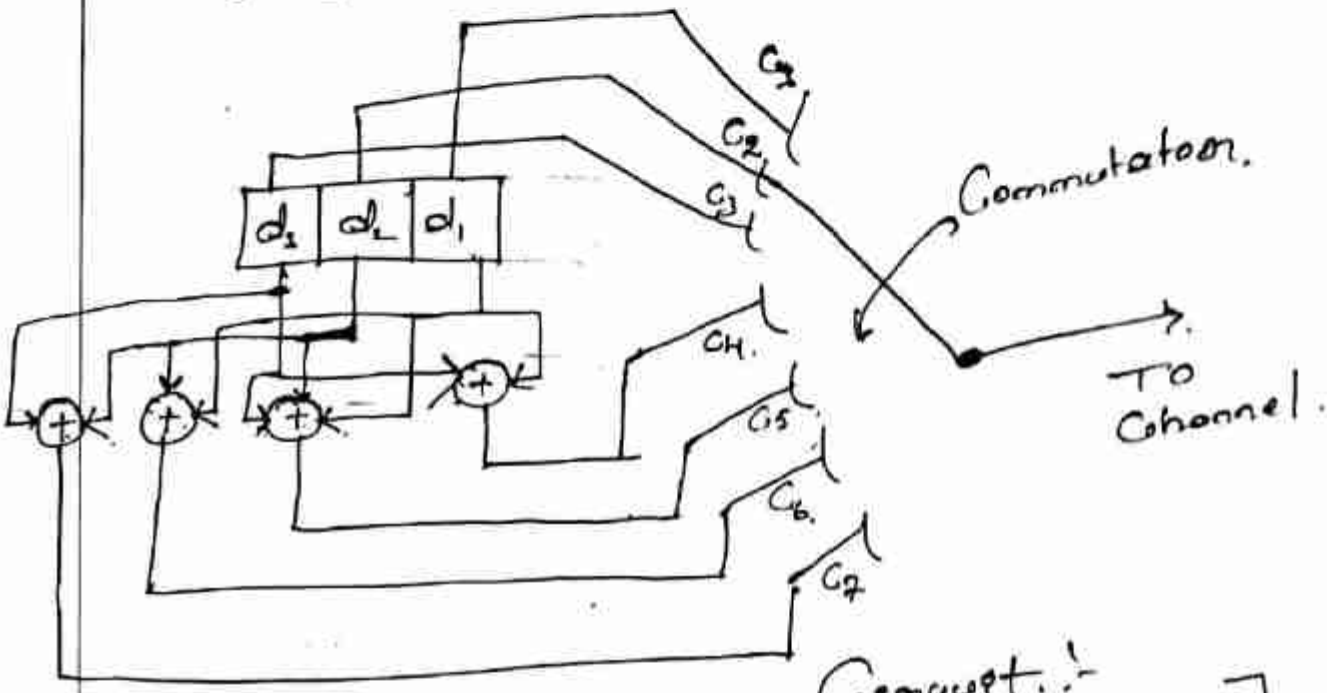
$$d_{min} = H_{N(min)} = 4$$

4) Encoding Circuit :-

$$[C] = [D][G]$$

$$= [d_1 d_2 d_3] \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

$$[C] = [d_1 \; d_2 \; d_3 \; (d_1+d_3) \; (d_1+d_2+d_3) \; (d_1+d_2) \; (d_2+d_3)] \; //$$



5) Syndrome Calculation Circuit :-

$$S = RH^T = [r_1 r_2 r_3 r_4 r_5 r_6 r_7] \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$S = [S_1 S_2 S_3 S_4] = [(r_1+r_3+r_4) \; (r_1+r_2+r_3+r_5) \; (r_1+r_2+r_6) \; (r_2+r_3+r_7)] \; //$$

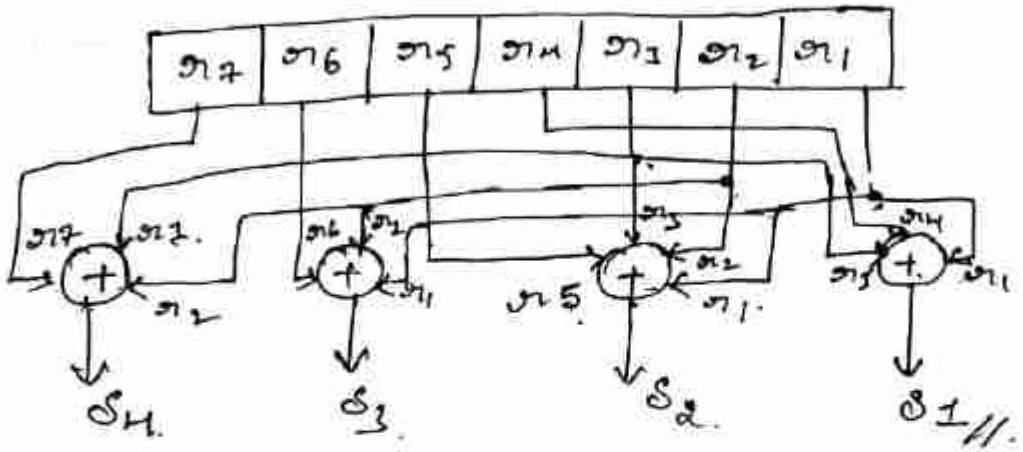


Fig: Syndrome calculation Circuit //

* Consider the (7,4) LBC whose generation matrix is given below. find

1) all code words 2) parity check matrix,
3) The minimum weight and distance to the code

$$G = \begin{bmatrix} 1000 & 101 \\ 0100 & 111 \\ 0010 & 110 \\ 0001 & 011 \end{bmatrix}$$

June-July - 2010 (10M)

Sol:- Code vector [C] is given by.

$$[C] = [D][G]$$

$$= [d_1\ d_2\ d_3\ d_4] \begin{bmatrix} 1000 & 101 \\ 0100 & 111 \\ 0010 & 110 \\ 0001 & 011 \end{bmatrix}$$

$$[C] = [d_1\ d_2\ d_3\ d_4.\ (d_1+d_2+d_3)\ (d_2+d_3+d_4)\ (d_1+d_2+d_4)] //$$

Since $k=4$ There are $2^k = 2^4 = 16$ message vectors give by 0000 ---------- 1111

Name.	message.	Codevectors.	Hamming weight
C_a	0 0 0 0	0000000	—
C_b	0 0 0 1	0001011	3
C_c	0 0 1 0	0010110	3
C_d	0 0 1 1	0011101	4
C_e	0 1 0 0	0100111	4
C_f	0 1 0 1	0101100	3
C_g	0 1 1 0	0110001	3
C_h	0 1 1 1	0111010	4
C_i	1 0 0 0	1000101	3
C_j	1 0 0 1	1001110	4
C_k	1 0 1 0	1010011	4
C_l	1 0 1 1	1011000	3
C_m	1 1 0 0	1100010	3
C_n	1 1 0 1	1101001	4
C_o	1 1 1 0	1110100	4
C_p	1 1 1 1	1111111	7

2) parity check matrix "H" is given by

$$H = [P^T \mid I_{n-k}] = [P^T \mid I_3] = \left[\begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

3) $d_{min} = (Hw)_{min}$

$= 3$ //.

* For a systematic (6,3) linear block code

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

1) Find all code vectors.
2) Draw Encoder Circuit for above Code.
3) Find minimum Hamming weight.

DEC-2010[4M]

Sol: 1) Code vector [C] is given by

$$[C] = [D][G]$$

$$= [d_1 d_2 d_3] \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$[C] = [d_1 \; d_2 \; d_3 \; (d_1+d_2) \; (d_2+d_3) \; (d_1+d_2)] \; //$$

Since K=3 There are $2^3 = 8$ message vectors. given by 000 ------ 111

Q) :-

given $(d_1 d_2 d_3) = (000)$

$\therefore [C_a) = (000000)$ //

given $(d_1 d_2 d_3) = (111)$

$\therefore C_h = (111000)$ //

Code. name.	Message vector.	Code vector	Hamming weight
C_a	0 0 0	000.000	
C_b	0 0 1	001110.	3
C_c	0 1 0	010011	3
C_d	0 1 1	011101	4
C_e	1 0 0	100.101	3
C_f	1 0 1	101011	3
C_g	1 1 0	110110	4
C_h	1 1 1	111000	3

2) $[c] = [D][G]$

$$[c] = [d_1\ d_2\ d_3] \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} = [d_1, d_2, d_3, d_1+d_2, d_2+d_3, d_1+d_2]$$

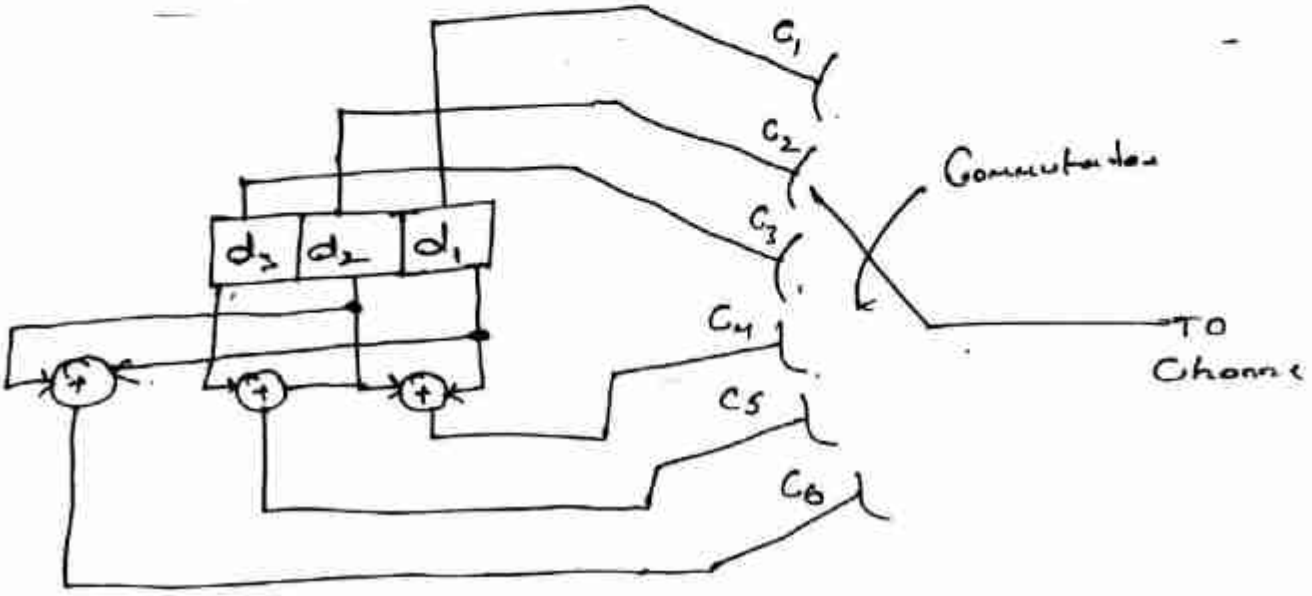


Fig: Encoder Circuit

3) $Data = HW_{(n,k)} = 3//$

* Considered a (6, 3) linear block code whose generator matrix is given below.

$$G = \begin{bmatrix} 100 & 111 \\ 010 & 110 \\ 001 & 101 \end{bmatrix}$$

Find. 1) all code words.
2) all hamming weights and distance
3) parity check matrix.
4) Draw Encoder Circuit

June/July-2004(8M), July/Aug-2008(8M), June/July-2011 [14M]

So 1 :-

For (6, 3) code. k = 3. Thus there are $2^3 = 8$ message vectors given by. 000, 001, 010, 011, 100, 101, 110, 111.

1) Codewords [C] is given by.

$$[C] = [D][G] = [d_1 d_2 d_3] \begin{bmatrix} 100 & 111 \\ 010 & 110 \\ 001 & 101 \end{bmatrix}$$

$$= [d_1, d_2, d_3, d_1+d_2+d_3, d_1+d_2, d_1+d_3]$$

Code name	message vectors.	Codevectors.	Hamming weight
C_a	000	000000	0
C_b	001	001101	3
C_c	010	010110	3
C_d	011	011011	4
C_e	100	100111	4
C_f	101	101010	3
C_g	110	110001	3
C_h.	111	111100.	4.

2) Destances. Examples:-

The destances are given. $d_{ab} = C_a + C_b$

$000000 + 001101$

$= 001101 = 3$ //

Thus $d_{ac} = 3$, $d_{ad} = 4$, $d_{bc} = 4$ $d_{ba} = 3$ //

$d_{ad} = 3$ // $d_{ac} = 3$ // $d_{ef} = 3$ // $d_{fg} = 4$ // $d_{gh} = 3$ // e.t.c //

$$G_1 = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} = [I_K : P] = [I_3 : P] \text{ //}$$

$$I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ //} \quad P = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \text{ //} \quad P^T = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \text{ //}$$

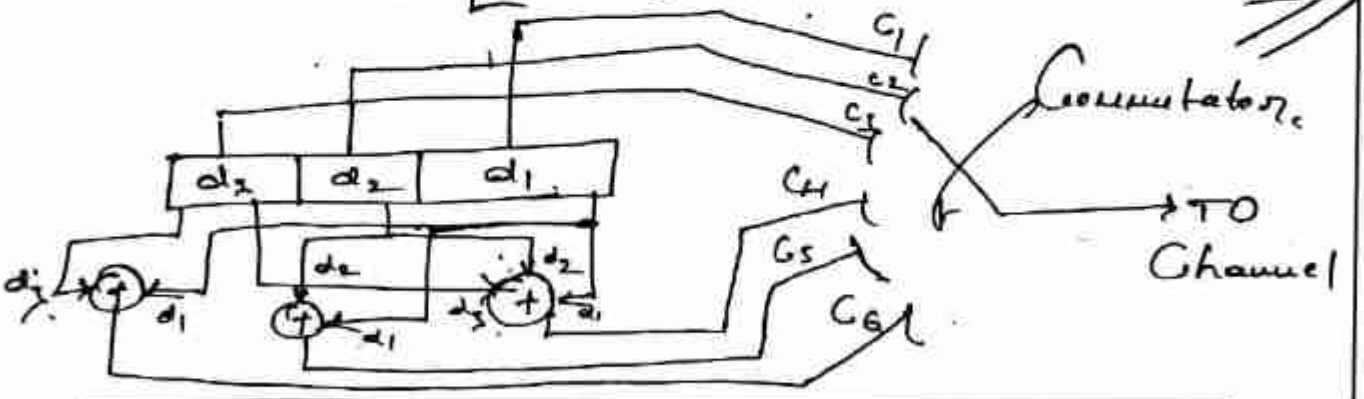
3) The parity check matrix is given by

$$[H] = [P^T : I_{n-k}]$$

$$[H] = [P^T : I_3] = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \text{ //}$$

4) $C = [D][G_1] = [d_1\ d_2\ d_3] \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$

$= [d_1\ d_2\ d_3\ d_1+d_2+d_3\ d_1+d_2\ d_1+d_3]$ //



* Given the generator matrix for an linear block code.

$$G = \begin{bmatrix} 1110 & 1000 \\ 1001 & 1100 \\ 1100 & 0110 \\ 0110 & 0011 \end{bmatrix}$$

1) Write the "G" in the Systematic form.
2) find n and k values
3) find all Codewords.
4) find dmin.
5) find the error detecting and Correcting Capability of the Code

Sol :-

1) $$G = \begin{bmatrix} 1110 & 1000 \\ 1001 & 1100 \\ 1100 & 0110 \\ 0110 & 0011 \end{bmatrix}$$ ← First of all the given generator matrix is not in the systematic form thus first we need to reduce the given "G" matrix to the form $[I_k : P]$ by elementary row operation.

NOTE

$r_2 \leftarrow r_2 + r_1$.

$$G = \begin{bmatrix} 1110 & 1000 \\ 0111 & 0100 \\ 1100 & 0110 \\ 0110 & 0011 \end{bmatrix}$$

$r_1 \leftarrow r_3 + r_2$.

$$\begin{bmatrix} 11101000 \\ 01110100 \\ 10111001 \\ 01100011 \end{bmatrix}$$

Finally $x_4 \leftarrow x_4 + x_1$ given

$$G = \begin{bmatrix} 1110 & 1000 \\ 0111 & 0100 \\ 1011 & 0010 \\ 1101 & 0001 \end{bmatrix}$$

Altogether NOTE
In book: $[G] = [I_K : P]$
Sharpe book OR: $[G] = [P : I_K]$
ITC book formula

$$G = [P : I_K] \text{ // or } [I_K : P] \text{ //}$$

where $P = \begin{bmatrix} 1110 \\ 0111 \\ 1011 \\ 1101 \end{bmatrix}$ // $I_K \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix}$ // $P^T = \begin{bmatrix} 1011 \\ 1101 \\ 1110 \\ 0111 \end{bmatrix}$ //

In XXT

2) $$G = \begin{bmatrix} 1110 & 1000 \\ 0111 & 0100 \\ 1011 & 0010 \\ 1101 & 0001 \end{bmatrix}$$

As there are Eight Columns and four rows in "G"
$n = 8$ and $K = 4$, It is (8, 4) linear block Code.

3) Code vector is given by.

$$[C] = [D][G].$$

given $[D] = d_1 d_2 d_3 d_4$.

$$[C] = [d_1 d_2 d_3 d_4] \begin{bmatrix} 1110 & 1000 \\ 0111 & 0100 \\ 1011 & 0010 \\ 1101 & 0001 \end{bmatrix}$$

$$[C] = [(d_1+d_3+d_4)\ (d_1+d_2+d_4)\ (d_1+d_2+d_3)\ (d_2+d_3+d_4)\ d_1,\ d_2,\ d_3,\ d_4] \text{ //}$$

For (8,4) Code $k=4$ Then there are $2^4 = 16$ message vectors given by (0000) - - - - - - (1111) //

Ex.1: given $(d_1 d_2 d_3 d_4) = (0000)$

$\therefore [C_0] = 00000000$ //

given $(d_1 d_2 d_3 d_4) = (0001)$

$\therefore [C_1] = 11010001$ //

given $(d_1 d_2 d_3 d_4) = (0010)$

$\therefore [C_2] = (10110010)$ //

Msg message vector.	Code vector	Hamming weight
0 0 0 0	0000 0000	- - - -
0 0 0 1	1101 0001	4
0 0 1 0	1011 0010	4
0 0 1 1	0110 0011	4
0 1 0 0	0111 0100	4
0 1 0 1	1010 0101	4
0 1 1 0	1100 0110	4
0 1 1 1	0001 0111	4
1 0 0 0	1101 1000	4
1 0 0 1	0011 1001	4
1 0 1 0	0101 1010	4
1 0 1 1	1000 1011	4
1 1 0 0	1001 1100	4
1 1 0 1	0100 1101	4
1 1 1 0	0010 1110	4
1 1 1 1	1111 1111	8.

4) $d_{min} = H_{min} = 4$ //

5) Error detecting capability $= d_{min} - 1 = 3$ //

Error Correcting Capability $t = \frac{d_{min} - 1}{2} = 1$ bit error //

* Obtain all possible code vectors for a (7,4) LBC in its Systematic form for the generation matrix.

$$G = \begin{bmatrix} 1 1 0 0 0 1 1 \\ 0 1 0 0 1 0 1 \\ 1 0 1 0 1 0 1 \\ 0 0 0 1 1 1 1 \end{bmatrix}$$

Sol:- First of all the given generation matrix is not in the Systematic form. Thus first we need to reduce the given "G" matrix to the form $[I_k : P]$ by elementary row operations.

$$G = \begin{bmatrix} 1 1 0 0 0 1 1 \\ 0 1 0 0 1 0 1 \\ 1 0 1 0 1 0 1 \\ 0 0 0 1 1 1 1 \end{bmatrix}$$

$r_1 \rightarrow r_1 + r_2 \Rightarrow \begin{bmatrix} 1 0 0 0 1 1 0 \\ 0 1 0 0 1 0 1 \\ 1 0 1 0 1 0 1 \\ 0 0 0 1 1 1 1 \end{bmatrix}$

$r_3 \rightarrow r_3 + r_1$

$G \Rightarrow \begin{bmatrix} 1 0 0 0 1 1 0 \\ 0 1 0 0 1 0 1 \\ 0 0 1 0 0 1 1 \\ 0 0 0 1 1 1 1 \end{bmatrix}$ //

$$G = [I_k : P]$$

$$I_H = \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix} \text{ //} \quad P = \begin{bmatrix} 110 \\ 101 \\ 011 \\ 111 \end{bmatrix} \text{ //} \quad P^T = \begin{bmatrix} 1 1 0 1 \\ 1 0 1 1 \\ 0 1 1 1 \end{bmatrix} \text{ //}$$

* Code vector [C] is given by.

$$[C] = [D][G]$$

$$= (d_1 d_2 d_3 d_4) \begin{bmatrix} 1000110 \\ 0100101 \\ 0010011 \\ 0001111 \end{bmatrix}$$

$$[C] = [\, d_1 \; d_2 \; d_3 \; d_4 \; (d_1+d_2+d_4)(d_1+d_3+d_4)(d_2+d_3+d_4) \,]$$

For (7,4) code K=4 Thus There are $2^4 = 16$ message vector.

given by 0000 ---------- 1111 //

Eg:- given $(d_1 d_2 d_3 d_4) = (0000)$

$[C_0] = [0000000]$ //

given $(d_1 d_2 d_3 d_4) = (0001)$

$[C_0] = [0001111]$ //

11y

Msg vector	Code vector	msg vector	Code vector
0000	0000000	1000	1000110
0001	0001111	1001	1001001
0010	0010011	1010	1010101
0011	0011100	1011	1011010
0100	0100101	1100	1100011
0101	0101010	1101	1101100
0110	0110110	1110	1110000
0111	0111001	1111	1111111

Hamming Bound

If an (n, k) linear block code is capable of correcting t-bit errors. Then it is necessary to have the number of syndromes to be at least equal to the total number of correctable error patterns. That is.

$$2^{n-k} \geq \sum_{i=0}^{t} {}^{n}C_i$$

The above inequality is called Hamming bound. A code for which the Hamming bound turns out to be an equality is called a perfect code.

Eg:- $n=7$, $k=4$ and $t=1$.

$$2^{n-k} = 2^{7-4} = 8//$$

$$\sum_{i=0}^{1} {}^{n}C_i = {}^{7}C_0 + {}^{7}C_1 = 1+7 = 8//$$

* Determine whether the following codes are perfect & not

(a) (15, 11), $t=1$

(b) (6, 3), $t=1$.

Sol:- 1) $2^{n-k} = 2^{15-11} = 2^4 = 16//$

$$\sum_{i=0}^{1} {}^{15}C_i = {}^{15}C_0 + {}^{15}C_1 = 16//$$

Since $2^{n-k} = \sum_{i=0}^{t} {}^{n}C_i$ Therefore (15, 11) is a perfect code.

2) $2^{n-k} = 8//$

$$\sum_{i=0}^{t} {}^{n}C_i = \sum_{i=0}^{1} {}^{6}C_i = {}^{6}C_0 + {}^{6}C_1 = 1+6 = 7//$$

Since $2^{n-k} \neq \sum_{i=0}^{t} {}^{n}C_i$ Therefore (6, 3) is not perfect code.

* Test hamming bound of (7,4) hamming code and. Show that it is perfect code.

Sol:- Check bit = (7-4) = 3 // W.K.T $d_{min} = 3$ //

Error detecting Capability = $d_{min} - 1$

$= 3 - 1 = 2$ //

Error Correcting Capability of the code is.

$$t = \frac{d_{min} - 1}{2} = \frac{3-1}{2} = 1$$ //

Consider $2^{n-k} \geq \sum_{i=0}^{t} {}^{n}C_i$ $t = 1$ //

$2^{n-k} = 2^{7-4} = 2^3 = 8$ //

$\sum_{i=0}^{t} {}^{n}C_i = \sum_{i=0}^{1} {}^{7}C_i = {}^{7}C_0 + {}^{7}C_1 = 1 + 7 = 8$ //

∴ We Can Conclude that (7,4) Hamming Code is a perfect code //

ERROR DETECTION AND ERROR CORRECTION CAPABILITIES OF LINEAR BLOCK CODES:-

Theorem 1:- The minimum distance of linear block code is equal to the minimum Hamming weight of a non-zero code vector.

Proof:- By definition.

$$d_{min} = \min \left[d(C_i, C_j) \right], \forall i, j \text{ and } i \neq j$$

We also have

$$d(C_i, C_j) = H_w(C_i \oplus C_j)$$

$$\therefore d_{min} = \min \left\{ H_w(C_i \oplus C_j) \right\}$$

now. $C_p \oplus C_j = C_K$ will also be a codeword. Though not equal to zero vector as $C_i \neq C_j$.

$\therefore d_{min} = \min \{ H_w(C_K) \}$ $\forall$ K. except C_K is an all-zero codeword.

Theorem 2:- For every code vector of weight 'l' in an Linear block code, there exist 'l' columns in 'H' matrix, whose sum is zero,

proof:- Consider the parity check matrix 'H' as

$$H = [h_0 \; h_1 \; h_2 \; \cdots \; h_{n-1}]$$

where $h_0, h_1, \ldots h_{n-1}$ are columns of "H" with respect to $CH^T = 0$ for valid Codeword C:

$$C \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ \vdots \\ h_{n-1} \end{bmatrix} = 0. \qquad \text{——— (1)}$$

Consider the weight of "C" as "l" That is it has "l" non zero elements say $C_{i1}, C_{i2}, \ldots C_{il} = 1$

∴ eq (1) becomes.

$$C_{i1} h_{i1} + C_{i2} h_{i2} + \cdots + C_{il} h_{il} = 0$$

$$h_{i1} + h_{i2} + \cdots h_{il} = 0$$

or $C_{i1} = C_{i2} = \cdots = C_{il} = 1$. There exist "l" columns in the "H" matrix whose sum is zero.

There are a few corollaries for Theorem 2.

Corollary 2 (a) :- An Linear block code is said to have a dmin of "d" if there are no fewer than d columns in "H" matrix whose sum is zero.

Corollary 2 (b) :- If there are "d" columns in "H" matrix whose sum is zero and no fewer than "d" columns whose sum is zero then the linear block code is said to have a minimum distance of "d"

Theorem 3: An (n, k) Linear block code is capable of correcting up to $[(d_{min}-1)/2]$ number of error bits for a minimum distance d_{min}.

A Linear block code with a minimum distance d_{min} can detect upto $(d_{min}-1)$ errors in each code-vector and can correct upto $[(d_{min}-1)/2]$ errors where $[(d_{min}-1)/2]$ denotes the largest integer no greater than $(d_{min}-1)/2$.

Proof:- Let "t" be the number of error bits the code can correct.

* Let "r" be the Received vector for the transmitted vector "c". Let "c^1" be any other code vector. Then the following condition has to be satisfied.

$$d(c, r) + d(c^1, r) \geq d(c, c^1)$$

If an error pattern of t-bit error occurs then

$$d(c, r) = t.$$

$$d(c, c^1) \geq d_{min}.$$

$$d(c^1, r) \geq d_{min} - t.$$

The decoder will properly decode the Received vector "r" as "c" if and only if the Received vector "r" is closer to "c" than any other vector "c'" in the code, that is $d(c, r) < d(c', r) \; \forall \; c' \neq c$ in code.

Substituting $d(c,r) = t$ and $d(c',r) \geq d_{min} - t$ we get

$$t < d_{min} - t = 2t < d_{min} \Rightarrow t < \frac{d_{min}}{2} \Rightarrow t \leq \frac{d_{min}-1}{2}.$$

∴ The Linear block code can properly correct t-bits of errors where.

$$t \leq \frac{d_{min}-1}{2}$$

$$t = \left\lfloor \frac{d_{min}-1}{2} \right\rfloor$$ //

The XXX.

SINGLE ERROR CORRECTING HAMMING CODES :-

* The parity check matrix $[H]$ is given by

$$H = [P^T : I_{n-k}]$$

$$H^T = \begin{bmatrix} P \\ \hline I_{n-k} \end{bmatrix} \quad \text{---} (1)$$

Eq (1) : It is clear that there are $(n-k)$ number of columns. Therefore. Each row in H^T has $(n-k)$ number of entries Each of which could be a "0" or 1. Thus we can have 2^{n-k} number of distinct rows. But a row of "0"s cannot be used as this represents the syndrome of no error. Thus we are left with $[2^{n-k}-1]$ number of distinct rows. From equation (1), we observe that H^T has "n" number of rows.

Thus the Condition for all the errors of H^T to be distinct is that

$$2^{n-k} - 1 \geq n.$$

$$\therefore 2^{n-k} \geq n+1$$

$$n-k \geq \log_2(n+1)$$

$$\text{(or)} \quad k \leq n - \log_2(n+1) \text{ //}$$

Thus the Single error Correcting (n,k) Hamming Code has the following parameters:

Code length : $n \leq 2^{n-k} - 1$

number of message bits : $k \leq n - \log_2(n+1)$

number of parity check bits : $(n-k)$

Error Correcting Capability $t = \frac{d_{min} - 1}{2}$.

* Design (n, k) Hamming Code with a minimum distance of $d_{min} = 3$ and a message length of 4 bits.

Sol:

Number of message bits $= k = 4$ //

Code length $n \leq 2^{n-k} - 1$

$n \leq 2^{n-4} - 1$ //

By trial and error, the least value of "n" which satisfies the inequality is found to be 7 //

$\therefore (n, k) = (7, 4)$

$n = 7$ and $k = 4$ //

Hamming Code parity check matrix is given by

$$H^T = \left[\frac{P}{I_{n-k}}\right] = \left[\frac{P}{I_{7-4}}\right] = \left[\frac{P}{I_3}\right] //$$

$$\therefore H^T = \left[\frac{P}{\begin{matrix}1&0&0\\0&1&0\\0&0&1\end{matrix}}\right] //$$

Since $k = 4$, there are $2^{2-k} = 2^{7-4} = 2^3 = 8$ // Combinations of 3-bit numbers namely (000), (001), (010), (011), (100), (101), (110), (111)

NOTE

Where $n = 0, 1, 2, 3$

n not possible because

$2^{0-4} - 1 = -0.93$ ← -ve value

$2^{1-4} - 1 = -0.875$ ← -ve value

$2^{2-4} - 1 = -0.75$ ← -ve.

$2^{3-4} - 1 = -0.5$ ← -ve.

$2^{4-4} - 1 = 0$ ← $n > k$ not satisfied

$2^{5-4} - 1 = 1$ //

$n > k$ condition is not satisfied

$2^{6-4} - 1 = 3$ //

$n > k$ condition is not satisfied

Where $n = 7$

$2^{7-4} - 1 = 7$ //

$n > k$ Condition is satisfied //

$\therefore 7 > 4$ condition satisfied.

* Out of these 8 combinations, 000 cannot be used and also 001, 010 and 100 cannot be used as they are already present as the rows of unit matrix $I_{n-k} = I_3$
* Thus we are left with only four combinations. 011, 101, 110 and 111 to be chosen as the rows of $[P]$

Let us choose a parity matrix.

$$[P] = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\therefore H^T = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} P \\ \hline I_{n-k} \end{bmatrix} //.$$

NOTE:
000 - negative
→ 001
→ 010
011
→ 100
101
110
111
P → parity matrix

they are already present as the rows of unit matrix $I_{n-k} = I_3$

$$H^T = \begin{bmatrix} P \\ \hline 100 \\ 010 \\ 001 \end{bmatrix}$$

∴ The parity check matrix "H" also called "Hamming matrix" is given by.

$$H = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} //$$

The generator matrix $[G]$ is given by.

$$[G] = [I_K \mid P]$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} //.$$

Code-vector [C] is given by

$$[C] = [D][G]$$

given $[D] = [d_1\ d_2\ d_3\ d_4]$

$$[C] = [d_1\ d_2\ d_3\ d_4] \begin{bmatrix} 1000\,011 \\ 0100\,101 \\ 0010\,110 \\ 0001\,111 \end{bmatrix}$$

$$= [d_1, d_2, d_3, d_4, (d_2+d_3+d_4), (d_1+d_3+d_4), (d_1+d_2+d_3)]$$ //

$K=4$. Thus there are $2^4 = 16$ message vectors given by

(0000), (0001), (0010), (0011), (0100), (0101), (0110), (0111)
(1000), (1001), (1010), (1011), (1100), (1101), (1110), (1111)

Example:-

let $(d_1\ d_2\ d_3\ d_4) = (0000)$.

$$C_a = (0000) \begin{bmatrix} 1000\,011 \\ 0100\,101 \\ 0010\,110 \\ 0001\,111 \end{bmatrix} = (0000\,000)$$ //

let $(d_1\ d_2\ d_3\ d_4) = (0001)$.

$$C_b = (0001) \begin{bmatrix} 1000\,011 \\ 0100\,101 \\ 0010\,110 \\ 0001\,111 \end{bmatrix} = (0001111)$$ //

Message vector (D)	Code vector (C)	message vector (D)	Code vector (C)
0000	0000000	1000	1000011
0001	0001111	1001	1001100
0010	0010110	1010	1010101
0011	0011001	1011	1011010
0100	0100101	1100	1100110
0101	0101010	1101	1101001
0110	0110011	1110	1110000
0111	0111100	1111	1111111

Table: Code-vectors Table for (7,4) Hamming Code.

Error Correction:

Since $d_{min} = 3$, The error Correcting Capability of this Code is.

$$t = \frac{d_{min} - 1}{2} = \frac{3-1}{2} = 1 \text{ //}$$

Ex:- Let the Received vector with single error be given by. $R = [1111001]$

$\therefore$ Syndrome $S = RH^T = [1111001] \begin{bmatrix} 011 \\ 101 \\ 110 \\ 111 \\ 100 \\ 010 \\ 001 \end{bmatrix}^T$

$= \underline{110}$ which is present in 3rd row of H^T.

Hence the error-vector $E = [0010000]$.

$\therefore$ The corrected code-vector. $C = R + E$.

$C = [1111001] + [0010000]$.

$= [1101001]$ //

* Design a Single error Correcting code with a message block Size of 11 and Show that by an Example that it can correct Single error.

AUCh-2001 [6M]

Sol: given $k = 11$.

- Code length : $n \leq 2^{n-k} - 1$.

Where $n = 15$

Code length : $n \leq 2^{15-1} - 1 = 15$ //

*There are totally $2^{n-k} = 2^{15-11} = 2^4 = 16$ //

*16 Combinations of 4 bit numbers namely 0000, 0001, 0010, ---- 1111

out of these 16 Combinations, 0000 Cannot be used and also 0001, 0010, 0100, 1000 Cannot be used as they are already present as the rows of unit matrix. $I_{n-k} = I_3$.

$$H^T = \begin{bmatrix} P_{k \times (n-k)} \\ I_{n-k} \end{bmatrix} = \begin{bmatrix} P_{(11 \times 4)} \\ I_{15-11} \end{bmatrix} = \begin{bmatrix} P_{(11 \times 4)} \\ I_4 \end{bmatrix}$$

$$H^T = \begin{bmatrix} P_{(11 \times 4)} \\ 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix}$$ //

Identity matrix (already present)

Identity matrix value (already present)

Identity matrix (already present)

neglate

(0000), 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111

" ___ " → Indicates parity matrix 'P' values

(already present) Identity matrix.

NOTE

Where $n = 0$ ------ 10

$n \leq 2^{n-k} - 1$ = -ve. error.

$\therefore n > k$ Condition is not Satisfied.

Where $n = 11$

$n \leq 2^{n-k} - 1 = 2^{11-11} - 1 = 0$ //

$\therefore n > k$ is not satisfied

Where $n = 12$.

$n \geq 2^{n-k} - 1 = 2^{12-11} - 1 = 1$

$\therefore n > k$ Condition is not Satisfied

Where $n = 13$

$n \geq 2^{n-k} - 1 = 2^{13-11} - 1 = 3$ //

$\therefore n > k$ Condition is not Satisfied

Where $n = 14$

$n \geq 2^{n-k} - 1 = 2^{14-11} - 1 = 7$ //.

$\therefore n > k$ Condition is not satisfied.

Where $n = 15$

$n \geq 2^{n-k} - 1 = 2^{15-11} - 1 = 15$ //

$n > k$ Condition is Satisfied.

$\rightarrow 15 > 11$ Satisfied.

$$H^T = \begin{bmatrix} P_{(1\times4)} \\ \hline 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix} = \begin{bmatrix} 0011 \\ 0101 \\ 0110 \\ 0111 \\ 1001 \\ 1010 \\ 1011 \\ 1100 \\ 1101 \\ 1110 \\ 1111 \\ \hline 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix}$$

∴ The [H] matrix is then given by

$$[H] = \left[\begin{array}{cccccccccccccc} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{array}\right]$$

The generator matrix [G] is then given by

$$[G] = [I_k \mid P_{k\times(n-k)}] = [I_{11} \mid P_{11\times4}]$$

$$= \left[\begin{array}{cccccccccccc|cccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

All the 2^11 code-vectors can be found out using the relationship

$$[C] = [D][G] \; //$$

Example to Show That it is a Single error Correcting Code:

Let us Consider the all-zero code vector and let us Suppose that an error has occurred in the penultimate bit of the Received Sequence given as

$$[R] = [000\ 000\ 000\ 000\ 010]$$

↓

Single error occurred due to noise

The Syndrome vector $[S]$ is then Calculated using.

$$[S] = [S_1 S_2 S_3 S_4] = [R][H]^T$$

$$= [000\ 000\ 000\ 000\ 010] \cdot \begin{bmatrix} 0011 \\ 0101 \\ 0110 \\ 0111 \\ 1001 \\ 1010 \\ 1011 \\ 1100 \\ 1101 \\ 1110 \\ 1111 \\ \hline 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix}$$

$$[S] = [0010] \; //$$

* Find the generator matrix "G" and parity check matrix 'H' for a linear block code with minimum distance three and a message block size of eight bits.

Feb-2002 [10M], Dec-2012 [4M]

Sol :- $n \leq 2^{n-k} - 1$

It is given that $k = 8$.

$n \leq 2^{n-8} - 1$.

Where $n = 12$.

$n \geq 2^{12-8} - 1 = 15$ //

$\rightarrow n > k$ Condition is satisfied.

$12 > 8$ //

$(n, k) = (12, 8)$ //

Note

Where $n = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11$

$n > k$ Condition is not satisfied.

Where $n = 12$.

$n \geq 2^{n-k} - 1 \geq 2^{12-8} - 1$

$= 16 - 1 = 15$ //

$\rightarrow n > k$ Condition is satisfied.

$12 > 8$.

The Transpose of [H] matrix is given by

$$H^T = \left[\frac{P}{I_{n-k}}\right] \text{ or } \left[\frac{P_{k \times (n-k)}}{I_{n-k}}\right] = \left[\frac{P_{8 \times 4}}{I_4}\right] = \left[\frac{P_{(8 \times 4)}}{\begin{matrix}1000\\0100\\0010\\0001\end{matrix}}\right]$$ //

There are totally $2^{n-k} = 2^{12-8} = 2^4 = 16$ // Combinations of 4 bits

numbers → 0000 (identity) 0001 0010 0011 0100 (identity) 0101 0110 0111

1000 (identity) 1001 1010 1011 1100 1101 1110 1111

↑ Consider a first 8 bits only because "P" is 8 × 4 matrix

$$H^T = \left[\frac{P_{8\times4}}{I_4}\right] = \begin{bmatrix} 0011 \\ 0101 \\ 0110 \\ 0111 \\ 1001 \\ 1010 \\ 1011 \\ 1100 \\ \hline 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix} //$$

The [H] matrix is then given by

$$[H] = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} //$$

The generator matrix [G] is given by

$$[G] = \left[I_K \,!\, P_{K\times(n-K)}\right] = \left[I_8 \,!\, P_{8\times4}\right].$$

$$[G] = \left[\begin{array}{cccccccc|cccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{array}\right] //$$

TABLE LOOKUP DECODING (SYNDROME DECODING) USING THE STANDARD ARRAY:-

Let $C_i = C_1, C_2, C_3 \ldots C_{2^k}$ be the 2^k distinct code vectors of a (n, k) linear block code.

Let 'E' be any error pattern.

$\therefore E_i = E + C_i$ — (1) where $i = 1, 2, \ldots 2^k$.

where the set of vectors E_i is called the "Co-SET" of the code.

Multiplying eq — (1) by H^T on both the sides.

$E_i H^T = EH^T + C_i H^T$

— where $C_i H^T = 0$

$$[E_i H^T = EH^T] //$$

Each element in the standard array is distinct.

Steps to Construct the Standard Array:-

Step no 1:

The 2^k valid code-vectors of the code are placed in a row. with all zero code vector as the 1st element.

Step no 2:-

From the remaining $(2^n - 2^k)$ n-tuples, an n-tuple E_2 is chosen and is placed below the all zero code-vector. The second row can now be formed by placing $(E_2 + C_i)$, $i = 2, 3, \ldots 2^k$ under C_i.

Step no 3 :-
Now an unused n-tuple E_3 is taken and the 3rd row is completed as given in step number 2.

Step no 4 :-
The process is continued till all the n-tuples are used.

The Resulting array for a (n, k) linear block code is shown in below table.

C_1 = all 0's	C_2	C_3	$\cdots C_{2^k}$
E_2	C_2+E_2	C_3+E_2	$\cdots C_{2^k}+E_2$
E_3	C_2+E_3	C_3+E_3	$\cdots C_{2^k}+E_3$
⋮	⋮		
$E_{2^{n-k}}$	$C_2+E_{2^{n-k}}$	$C_3+E_{2^{n-k}}$	$\cdots C_{2^k}+E_{2^{n-k}}$

Table : A Standard array for an (n, k) Linear block code.

Properties of Standard Array :-

property -1 :-
Each element in the standard array is distinct and hence different columns of the array are disjoint.

property -2 :- The first entry of each row is called the coset leader and it represent a correctable error pattern for a particular code.

Property - 3 :-

All the 2^k n-tuples of a co-set have the same Syndrome and the Syndrome of different Co-sets are different. //

With the above properties of standard array in mind, the following procedure can be used for decoding :-

1) The Syndrome RH^T is computed for the Received vector R and let $RH^T = S$ //.

2) The Co-set Leader E_i is located which has a Syndrome $S = E_i H^T$. Then E_i will be the error pattern caused by the noisy channel.

3) The Corrected Code C is obtained from $C = R + E_i$ - This scheme will correct 2^{n-k} most likely error patterns introduced by the channel.

GENERAL DECODING CIRCUIT FOR (n, k) LBC

The implementation of the decoding circuit, the Standard array can be regarded as the truth Table of "n" - switching functions.

The Truth table Consists of 2 Columns

① The Syndrome Column.

② Co-set Leader Column

$$c_1 = f_1(s_1, s_2, \ldots s_{n-k})$$
$$c_2 = f_2(s_1, s_2, \ldots s_{n-k})$$
$$\vdots$$
$$c_n = f_n(s_1, s_2, \ldots s_{n-k})$$

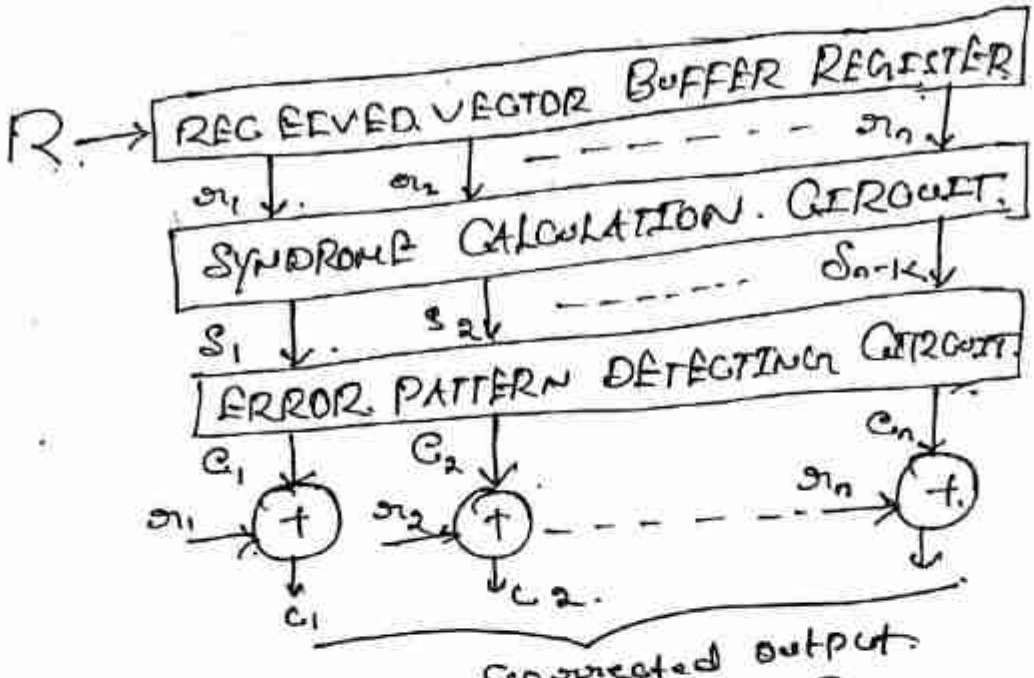


Fig: General decoding Circuit for (n, k) linear block code.

Where, $s_1, s_2 \ldots s_{n-k}$ are Syndrome digits

$c_1 c_2 \ldots c_n$ are error digits of Coset Leader.

*. Construct the Standard array for a (6,3) linear block code with the parity matrix

$$\begin{bmatrix} 101 \\ 011 \\ 110 \end{bmatrix}$$

using the same decode as the Received vectors.

a) 110011 b) 100001 c) 110111 d) 011000

Sol:- let $P = \begin{bmatrix} 101 \\ 011 \\ 110 \end{bmatrix}$ //

given $n=6$ and $k=3$ for (6,3) Linear block code.

Since $k=3$ there are $2^k = 2^3 = 8$ message vectors. given by 000, 001, 010, 011, 100, 101, 110, 111.

Code vector $[C]$ is given by.

$$[C] = [D][G]$$

$$[G] = [I_k \mid P] = [I_3 \mid P]$$

$$[G] = \begin{bmatrix} 100 \mid 101 \\ 010 \mid 011 \\ 001 \mid 110 \end{bmatrix} //$$

$$[C] = [d_1\ d_2\ d_3] \begin{bmatrix} 100 & 101 \\ 010 & 011 \\ 001 & 110 \end{bmatrix}$$

$$= [d_1, d_2, d_3, d_1+d_3, d_2+d_3, d_1+d_2] //$$

Ex:- $(d_1 d_2 d_3) = (000)$

$C_a = (000000)$ //

$(d_1 d_2 d_3) = (001)$.

$C_b = (001110)$ //

P.T.O

Code name	message vector	Code vector
C_a	000	0000.00
C_b	001	001110
C_c	010	010011
C_d	011	011101
C_e	100	100101
C_f	101	101011
C_g	110	110110
C_h	111	111000.

$$P = \begin{bmatrix} 101 \\ 011 \\ 110 \end{bmatrix} \therefore P^T = \begin{bmatrix} 101 \\ 011 \\ 110 \end{bmatrix}$$

$$H = [P^T : I_{n-k}] = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H^T = \begin{bmatrix} 101 \\ 011 \\ 110 \\ 100 \\ 010 \\ 001 \end{bmatrix} = \begin{bmatrix} P \\ I_{n-k} \end{bmatrix}$$

Syndrome	Co-set leaders							
000	000000	001110	010011	011101	100101	101011	110110	111000
101	100000	101110	110011	111101	000101	001011	010110	011000
011	010000	011110	000011	001101	110101	111011	100110	101000
110	001000	000110	011011	010101	101101	100011	111110	110000
100	000100	001010	010111	011001	100001	101111	110010	111100
010	000010	001100	010001	011111	100111	101001	110100	111010
001	000001	001111	010010	011100	100100	101010	110111	111001
111	110000	111110	110011	000101	010101	011011	000110	011010

1) Let the Received vector $R_A = 110011$

$$S_A = R_A H^T = 110011 \begin{bmatrix} 101 \\ 011 \\ 110 \\ 100 \\ 010 \\ 001 \end{bmatrix} = 001$$

$\therefore$ Error vector $E_A = 000001$

Corrected Code vector $C_A = R_A + E_A = 110011 + 000001 = 110010$

2) $R_B = 100001$

$$S_B = R_B H^T = [100001]\begin{bmatrix}101\\011\\110\\100\\010\\001\end{bmatrix} = 100$$

Error vector $E_B = [000100]$

$\therefore C_B = R_B + E_B = [100001] + [000100] = 100101$

3) $R_C = 110111$

$$S_C = R_C H^T = 110111\begin{bmatrix}101\\011\\110\\100\\010\\001\end{bmatrix} = 001$$

$\therefore E_C = [000001]$

$C_C = R_B + E_C = [110111] + [000001]$

$= [110110]$

4) $R_D = 011000$

$$S_D = R_D H^T = 011000\begin{bmatrix}101\\011\\110\\100\\010\\001\end{bmatrix} = 101$$

$E_D = [100000]$

$C_D = R_D + E_D = [011000] + [100000]$

$= 111000$

*. Construct the Standard array for a (7,4) linear block code with the parity matrix "P" is given by

$$[P] = \begin{bmatrix} 1 1 1 \\ 1 1 0 \\ 1 0 1 \\ 0 1 1 \end{bmatrix}$$

Draw the Complete error Correcting Circuit (decoding Circuit) for the (7,4) linear block code //

Sol: * The generator matrix [G] is given by

$$[G] = [I_K : P] = [I_4 : P]$$

$$= \begin{bmatrix} 1000 & 111 \\ 0100 & 110 \\ 0010 & 101 \\ 0001 & 011 \end{bmatrix} //$$

* The Code vector [C] is given by

$$[C] = [D][G]$$

$$= (d_1 d_2 d_3 d_4) \begin{bmatrix} 1000 & 111 \\ 0100 & 110 \\ 0010 & 101 \\ 0001 & 011 \end{bmatrix}$$

$$[C] = [d_1 d_2 d_3 d_4 . (d_1+d_2+d_3) (d_1+d_2+d_4) (d_1+d_3+d_4)]$$

Since K=4. There are $2^K = 2^4 = 16$ message vectors is given by. (0000) - - - - - - - - - - - - - - - (1111).

Ex: given $[d_1 d_2 d_3 d_4] = [0000]$.

$$[C_a] = [0000000] //$$

given $[d_1 d_2 d_3 d_4] = [0001]$

$$[C_b] = [0001011] //$$

given $[d_1 d_2 d_3 d_4] = [0011]$

$$[C_c] = [0011110] //$$

message vector (d)	Code Vectors	message vector	Code Vector (c)
0 0 0 0	0 0 0 0 0 0 0	1 0 0 0	1 0 0 0 1 1 1
0 0 0 1	0 0 0 1 0 1 1	1 0 0 1	1 0 0 1 1 0 0
0 0 1 0	0 0 1 0 1 0 1	1 0 1 0	1 0 1 0 0 1 0
0 0 1 1	0 0 1 1 1 1 0	1 0 1 1	1 0 1 1 0 0 1
0 1 0 0	0 1 0 0 1 1 0	1 1 0 0	1 1 0 0 0 0 1
0 1 0 1	0 1 0 1 1 0 1	1 1 0 1	1 1 0 1 0 1 0
0 1 1 0	0 1 1 0 0 1 1	1 1 1 0	1 1 1 0 1 0 0
0 1 1 1	0 1 1 1 0 0 0	1 1 1 1	1 1 1 1 1 1 1

* The parity check matrix "H" is given by.

$$H = [P^T \mid I_{n-k}] = [P^T \mid I_3]$$

$$H = \begin{bmatrix} 1 1 1 0 & 1 0 0 \\ 1 1 0 1 & 0 1 0 \\ 1 0 1 1 & 0 0 1 \end{bmatrix} \quad \therefore H^T = \begin{bmatrix} 1 1 1 \\ 1 1 0 \\ 1 0 1 \\ 0 1 1 \\ 1 0 0 \\ 0 1 0 \\ 0 0 1 \end{bmatrix}$$

Syndrome	Co-set Leader					
0 0 0	0 0 0 0 0 0 0	0 0 0 1 0 1 1	0 0 1 0 1 0 1	0 0 1 1 1 1 0	- - - - - - -	1 1 1 1 1 1 1
1 1 1	1 0 0 0 0 0 0	1 0 0 1 0 1 1	1 0 1 0 1 0 1	1 0 1 1 1 1 0		0 1 1 1 1 1 1
1 1 0	0 1 0 0 0 0 0	0 1 0 1 0 1 1	0 1 1 0 1 0 1	0 1 1 1 1 1 0		1 0 1 1 1 1 1
1 0 1	0 0 1 0 0 0 0	0 0 1 1 0 1 1	0 0 0 0 1 0 1	0 0 0 1 1 1 0		1 1 0 1 1 1 1
0 1 1	0 0 0 1 0 0 0	0 0 0 0 0 1 1	0 0 1 1 1 0 1	0 0 1 0 1 1 0		1 1 1 0 1 1 1
1 0 0	0 0 0 0 1 0 0	0 0 0 1 1 1 1	0 0 1 0 0 0 1	0 0 1 1 0 1 0		1 1 1 1 0 1 1
0 1 0	0 0 0 0 0 1 0	0 0 0 1 0 0 1	0 0 1 0 1 1 1	0 0 1 1 1 0 0		1 1 1 1 1 0 1
0 0 1	0 0 0 0 0 0 1	0 0 0 1 0 1 0	0 0 1 0 1 0 0	0 0 1 1 1 1 1		1 1 1 1 1 1 0
↑ ↑ ↑ $S_1 S_2 S_3$	↑ ↑ ↑ ↑ ↑ ↑ ↑ $e_1 e_2 e_3 e_4 e_5 e_6 e_7$					

H^T

Table: Standard array for (7,4) Code.

From the above Table the error bits corresponding to the coset leader can be expressed in terms of Syndrome bits are given below.

$C_1 = S_1 S_2 S_3$ $C_4 = \overline{S_1} S_2 S_3$ $C_7 = \overline{S_1}\,\overline{S_2} S_3$

$C_2 = \overline{S_1} S_2 \overline{S_3}$ $C_5 = S_1 \overline{S_2} \overline{S_3}$

$C_3 = S_1 \overline{S_2} S_3$ $C_6 = \overline{S_1} S_2 \overline{S_3}$

The Syndrome [S] is given by.

$$[S] = RH^T = \begin{bmatrix} r_1 r_2 r_3 r_4 r_5 r_6 r_7 \end{bmatrix} \begin{bmatrix} 111 \\ 110 \\ 101 \\ 011 \\ 100 \\ 010 \\ 001 \end{bmatrix}$$

$$= \begin{bmatrix} (r_1 + r_2 + r_3 + r_5) & (r_1 + r_2 + r_4 + r_6) & (r_1 + r_3 + r_4 + r_7) \end{bmatrix}$$

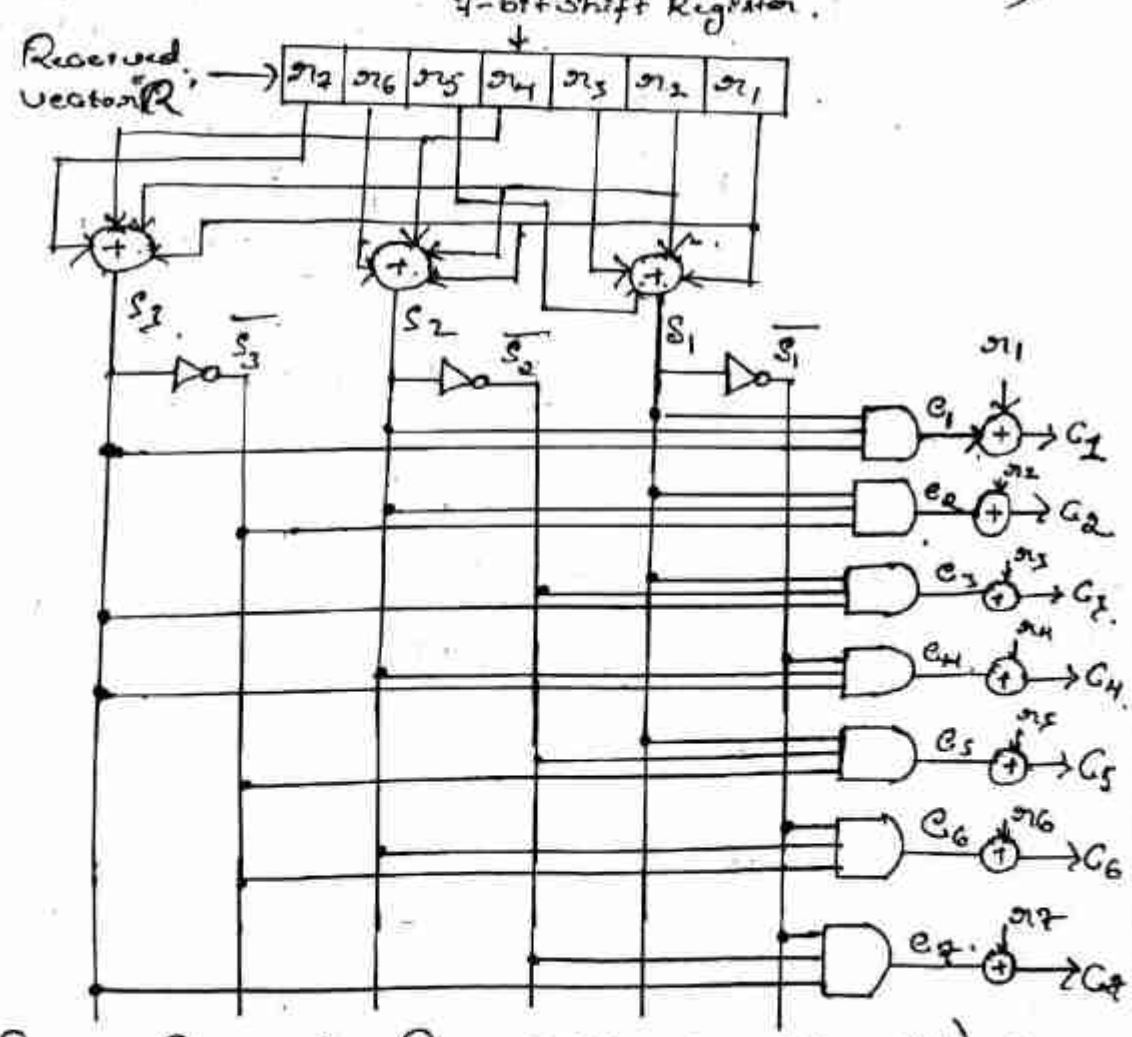


Fig: Error Correcting Circuit (decoding circuit)

* Design a (4, 2) linear block code.

1) Find the generator matrix for the code vector set
2) Find the parity check matrix.
3) Choose the code vectors to be in Systematic form, with the goal of maximizing dmin.
4) Enter the Sixteen 4 – tuples into a Standard array
5) What are the error - detecting and error Correcting Capabilities of the code?
6) make a Syndrome table for the Correctable error patterns.
7) Draw the Encoding Circuit
8) Draw the Syndrome Calculating Circuit

Sol :-

* For (4, 2) linear block code, we have n=4 and k=2
* There are totally $2^{n-k} = 2^{4-2} = 2^2 = 4$ Combinations of 2 bit numbers namely 00, 01, 10, 11 //.
* The parity matrix "P" is the order of k X (n-k)
  = 2 X (2) = 2 X 2 given by.

$$P = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} //$$

* The unit matrix rows 01 and 10 Cannot be used as the rows of P and also a row of 00 which represents Syndrome of no error, Cannot be used, thus leaves us with only Choice of "11" to be used.
* But the parity matrix must have two distinct rows of 2 binary digits each. Hence we have no other Choice than to accept either "10" (or) "01" as the second row. Let us Choose "10" as the 2nd row of P

1) * The generator matrix $[G]$ is given by

$$[G] = [I_K \mid P] = [I_2 \mid P] = \begin{bmatrix} 1 & 0 & : & 1 & 1 \\ 0 & 1 & : & 1 & 0 \end{bmatrix} ///$$

2) * The parity check matrix $[H]$ is given by

$$[H] = [P^T \mid I_{n-k}] = [P^T \mid I_{4-2}] = [P^T \mid I_2]$$

$$[H] = \begin{bmatrix} 1 & 1 & : & 1 & 0 \\ 1 & 0 & : & 0 & 1 \end{bmatrix} ///$$

3) * The Systematic Code-vectors are found from.

$$[C] = [D][G].$$

Since $K=2$. Then there are $2^2 = 4$ message vectors given by 00, 01, 10, 11.

$$C_A = [D_A][G] = [00] \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} = [0000] ///$$

$$C_B = [D_B][G] = [01] \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = [0110] ///$$

$$C_C = [D_C][G] = [10] \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = [1011] ///$$

$$C_B = [D_D][G] = [11] \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = [1101] ///$$

4) $H = \begin{bmatrix} 1 & 1 & : & 1 & 0 \\ 1 & 0 & : & 0 & 1 \end{bmatrix}$ $\therefore H^T = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} ///$

Syndrome	Co-set leader			
00	0000.	0110	1011	1101
Rows of H^T 11	1000	1110	0011	0101
Rows of H^T 10	0100	0010	1111	1001
Rows of H^T 10	0010	0100	1001	1111
Rows of H^T 01	0001	0111	1010	1100.

Table: Standard array for (4, 2) linear block code

5) Code vector [C] is given by.

message vector.	Code-vector.	Hamming weight $[H_W]$
00	0000	0
01	0110	2
10	1011	3
11	1101.	3

∴ for a linear block code $d_{min} = H_{W\,min}$

$d_{min} = 2$ //

∴ Error detecting Capability $= d_{min} - 1$

$= 2 - 1 = 1$ //.

∴ Error Correcting Capability $t = \frac{d_{min} - 1}{2} = \frac{2-1}{2} = \frac{1}{2}$ //

The Syndrome [S] is given by

$S = RH^T$

Where $R = [1111]$ // (or) given $R = [1111]$ //

$$S = [1111] \begin{bmatrix} 11 \\ 10 \\ 10 \\ 01 \end{bmatrix} = [10]$$ //

∴ $S = 10$. which is present in the 2nd or row 1 or 3rd row of H^T. Thus the error vector may be respectively [0100] (or) [0010]. Since we have the ambiguity, we can conclude that the given (4, 2) linear block code cannot correct any errors.

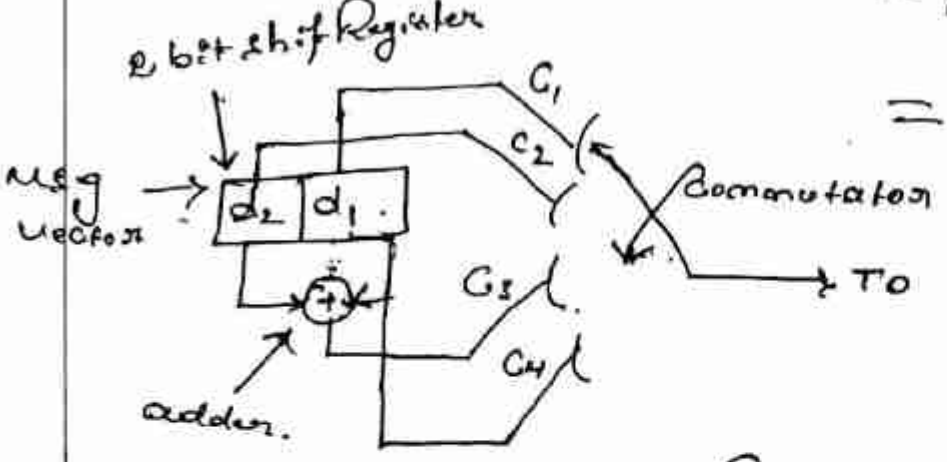
However. If Can detect a Single error. Then. Can be Confirmed by looking at the Syndrome for any Received vector. If the Syndrome is a non-zero vector Then an error is detected.

6) The Syndrome table for the Correctable error patterns, is shown in below table. Leaving out the two errors Corresponding to Syndrome 10 in below Table.

Syndrome	Coset Leader.			
00	0000	0110	1011	1101
11	1000	1110	0011	0101
01	0100	0111	1010	1100

7) - Encoding Circuit:

$$[C] = [D][G]$$
$$= [d_1 d_2] \begin{bmatrix} 1011 \\ 0110 \end{bmatrix}$$
$$= [d_1, d_2, (d_1+d_2), d_1]$$



8) Syndrome Calculation Circuit :-

$$S = RH^T = [r_1 r_2 r_3 r_4] \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} = [(r_1+r_2+r_3), (r_1+r_4)]$$

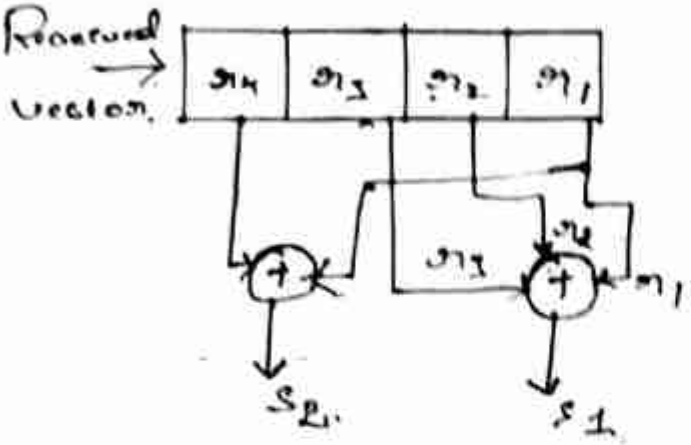


Fig: Syndrome Calculation Circuit for (4, 2) linear block code.

* For a (5, 2) LBC, the generator matrix is of the form $[I_k \mid P]$ where $[P]$ is given by

$$[P] = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

Find. 1) generator matrix 2) parity check matrix
3) all possible valid Code-vector.
4) dmin and error-Correcting Capability of the code

Sol: -1) Generator matrix is given by

$$[G] = [I_k \mid P] = [I_2 \mid P]$$

$$[G] = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$ ///

2) parity check matrix $[H]$ is given by

$$[H] = [P^T : I_{n-k}] = [P^T : I_3]$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} //$$

3) Code vectors $[c]$ is given by

$$[c] = [D][G] = [d_1 d_2] \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$= [d_1 \; d_2 \; (d_1 \oplus d_2) \; (d_1 \oplus d_2) \; d_1] //$$

$2^k = 2^2 = 4$ message vectors.

message vectors is given by 00, 01, 10, 11

message vector	Code vectors.	Hamming weight.
00	00000	0
01	01110	3
10	10111	4
11	11001	3

4) $d_{min} = H_{min} = 3$ //

Error correcting Capability $t = \dfrac{d_{min} - 1}{2}$

$= \dfrac{3-1}{2} = 1$ //

BINARY CYCLIC CODES & (OR) CYCLIC CODES

Cyclic Codes :-

Cyclic Codes are an important subclass of Linear block Codes which satisfy the additional structural requirement so as to have an ease of Implementation.

Cyclic Codes are developed under the strong structure of Galois fields, coding efficient encoding and decoding algorithms. These codes can be easily implemented using shift registers and feedback connections.

ALGEBRAIC STRUCTURE OF CYCLIC CODES :-

Definition of Cyclic Code :-

A (n, k) linear block code "C" is said to be a cyclic code if every cyclic shifts of the code is also a code vector of "C"

For Example :- Let $C_1 = 0111001$ be a code vector of C. If $C_2 = 1011100$ is also a code vector of C, then it is called as "Cyclic Code".

Similarly $C_3 = 0101110$, $C_4 = 0010111$ e.t.c. will be also be code vector of C.

In general, Let the n-tuple be represented by

$$V = (v_0\ v_1\ v_2 \cdots v_{n-1}) \quad \longrightarrow (1)$$

If "V" belongs to a Cyclic Code then.

$$V^{(1)} = (v_{n-1}\ v_0\ v_1\ v_2 \cdots v_{n-2})$$
$$V^{(2)} = (v_{n-2}\ v_{n-1}\ v_0\ v_1\ v_2 \cdots v_{n-3})$$
$$\vdots$$
$$V^{(i)} = (v_{n-i}\ v_{n-i+1} \cdots v_0\ v_1\ v_2 \cdots v_{n-i-1}) \quad \text{--- (2)}$$

obtained by shifting "V" cyclically successively are also Code vectors of "C".

This property of cyclic code allows us to treat the elements of each code vector as the coefficients of a polynomial of degree $(n-1)$.

Eq (1) Can now be represented as a polynomial given by

$$V(X) = v_0 + v_1 X + v_2 X^2 + \cdots + v_{n-1} X^{n-1}.$$

Illy Set of eq (2) are also represented as polynomial given by

$$V^1(X) = v_{n-1} + v_0 X + v_1 X^2 + \cdots + v_{n-2} X^{n-1}$$
$$V^2(X) = v_{n-2} + v_{n-1} X + v_0 X^2 + \cdots + v_{n-3} X^{n-1}.$$
$$\vdots$$
$$V^{(i)}(X) = v_{n-i} + v_{n-i+1} X + v_{n-i+2} X^2 + \cdots + v_{n-i-1} X^{n-1}$$

These coefficient $v_0, v_1, \cdots v_{n-1}$ belong to a binary field with the following rules of addition and multiplication

$0+0=0$	$0 \cdot 0 = 0$
$0+1=1$	$0 \cdot 1 = 0$
$1+0=0$	$1 \cdot 0 = 0$
$1+1=0$	$1 \cdot 1 = 1$
Modulo-2 addition.	Modulo-2 multiplication.

Modulo-2 algebra:-

Let us discuss addition, Subtraction, Multiplication and division of polynomials with suitable examples

Addition: The quantity $X+X$ can be written as.

$$X+X = X(1+1) = X.0 = 0 \text{ //}$$

Since $1+1=0$ in modulo-2 addition.

Similarly $X^2+X^2 = X^2(1+1) = X^2.0 = 0$

and $X^3+X^3 = X^3(1+1) = X^3.0 = 0$ and so on.

Subtraction: of polynomials is same as addition. in modulo-2 algebra.

Multiplication: The quantity $X.X = X^2$, $X^2.X = X^3$ and so on.. Let us now Consider multiplication of two polynomials. //

PROPERTIES OF CYCLIC CODES:- JUNE-JULY-2016 [6M]

1. For a (n,k) Cyclic Codes there exists a generator polynomial of degree $(n-k)$ and it is given by.

$$g(x) = g_0 + g_1 X + g_2 X^2 + \cdots\cdots + g_{n-k} X^{n-k}.$$

2. The generator polynomial $g(x)$ of a (n,k) Cyclic Code is a factor of X^n+1

$$\text{i.e } X^n+1 = g(x).h(x)$$

where $h(x)$ is another polynomial of degree "k" called "parity Check polynomial".

3. If $g(x)$ is a polynomial of degree $(n-k)$ and is a factor of X^n+1 then it generates the (n,k) cyclic code.

4. The code vector polynomial $V(X)$ can be found by using

$$V(X) = D(X)g(X)$$

where $D(X)$ = message vector polynomial of degree "k"

$$\therefore D(X) = d_0 + d_1X + d_2X^2 + \cdots + d_{k-1}X^{k-1}$$

5. Cyclic codes in Systematic format can be obtained by performing the division $\left(\frac{X^{n-k}D(X)}{g(X)}\right)$ and appending the Remainder of this division before the data.

$$\text{i.e } V(X) = R(X) + D(X)$$

In a linear (n,k) code there are 2^k messages and codewords

GALOIS FIELDS :-

Consider the set (0,1) together with modulo-2 addition and multiplication shown in below tables (1) and (2). Such a field is usually called a "binary field" and is denoted by "GALOIS FIELD (2)" or GF(2)

MODULO-2 addition
0+0 = 0
0+1 = 1
1+0 = 1
1+1 = 0

Table (1).

MODULO-2 multiplication
0·0 = 0
0·1 = 0
1·0 = 0
1·1 = 0

Table (2).

JUNE-JULY 2010 [10M], DEC-2013/JAN2014 [10M], DEC2014/JAN2015 [10M]

ENCODING OF CYCLIC CODES USING (n-k) SHIFT REGISTERS (OR) ENCODING USING (n-k) BIT SHIFT REGISTER :-

An (n, k) Cyclic Code with Systematic form can be obtained by performing the following steps.

1. Multiply the message polynomial D(X) by X^{n-k}
2. Divide $X^{n-k} D(X)$ by g(X) and obtain the Remainder R(X)
3. The Code can be obtained as.

$$V(X) = R(X) + X^{n-k} D(X)$$

The above procedures can be accomplished using an, (n-k) stage register with feedback Connection based on the Coefficients of the generator polynomial. So we have

$$g(X) = g_0 + g_1 X + g_2 X^2 + \cdots + g_{n-k} X^{n-k}$$

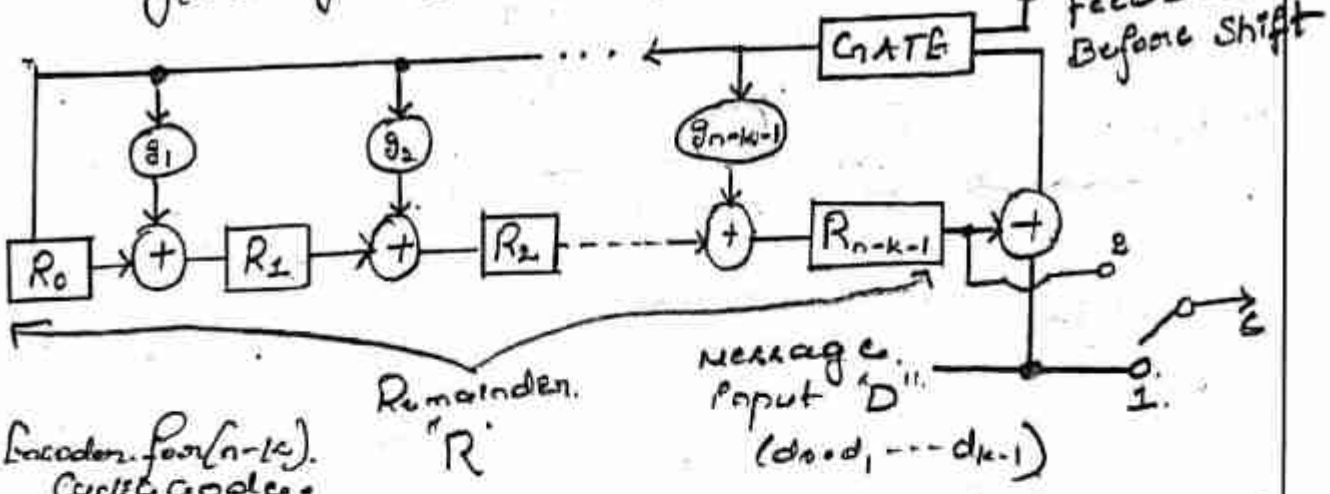


Fig :- Encoder for (n-k) Cyclic Codes.

Symbols :

- R → Flipflops that make up a Shift register.
- + → modulo-2 adders.
- g_i → A closed path if $g_i = 1$ and open if $g_i = 0$
- Gate → AND gate.

now. $g_0 = g_{n-k} = 1$ always. Therefore.

$$g(x) = 1 + g_1 x + g_2 x^2 + \cdots + \underbrace{g_{n-k}}_{=1} x^{n-k}$$

$$\therefore g(x) = 1 + g_1 x + g_2 x^2 + \cdots + x^{n-k}.$$

A general Circuit for Encoding an (n, k) Cyclic Code using an $(n-k)$ Stage linear feedback shift register as shown in the above figure. The Encoding procedure using feedback shift registers of $(n-k)$ stages involves the following steps:

Step 1 : With GATE turned ON, Shift the message bits to the shift Register.

Step 2 : once all bits are shifted. The Remainder will be obtained in the shift registers that can be taken out by turning the GATE OFF.

The pre-multiplication of x^{n-k} to $u(x)$ is performed by shifting the input from the right end.

PRIMITIVE POLYNOMIAL:

The algebra of GF (2^m) can be derived using a polynomial of degree "m" and with binary coefficients and using a new variable "α" called "primitive element", such that $p(\alpha) = 0$. When $p(x)$ is irreducible, then $p(x)$ is said to be "primitive polynomial".

* Find the product of polynomials $f_1(x) = (x+1)$ and $f_2(x) = x^3+x+1$ using modulo-2 algebra

Sol: $f_1(x).f_2(x) = (x+1)(x^3+x+1)$

$= x^4+x^2+x+x^3+x+1$

$= x^4+x^3+x^2+x+x+1$

$= x^4+x^3+x^2+x(1+1)+1$

$= x^4+x^3+x^2+x(0)+1$

$= x^4+x^3+x^2+1$ //

* Multiply $f_1(x) = 1+x+x^3$ and $f_2(x) = 1+x+x^2+x^4$ using modulo-2 algebra.

$f_1(x)\,f_2(x) = (1+x+x^3)(1+x+x^2+x^4)$

$= 1+x+x^2+x^4+x+x^2+x^3+x^5+x^3+x^4$

$+x^5+x^7$

$= 1+x+x+x^2+x^2+x^3+x^3+x^4+x^4+x^5+x^5+x^7$

$= 1+x[1+1]+x^2[1+1]+x^3[1+1]+x^4[1+1]+x^5[1+1]+x^7$

$= 1+0+0+0+0+0+x^7 = 1+x^7$ //

* Divide $f_2(x) = x^6+x^5+x^2$ by $f_1(x) = x^3+x+1$ using modulo-2 algebra.

P.T.O.

Sol :- $x^3+x+1 \;)\; x^6+x^5+x^2 \;(x^3+x^2+x$ ← Q(x)

$x^6+x^4+x^3$

$x^5+x^4+x^3+x^2$

$x^5+x^3+x^2$

x^4

x^4+x^2+x

x^2+x ← R(x)

The quantities $Q(x) = x+x^2+x^3$ is called "QUOTIENT POLYNOMIAL" and $R(x) = x+x^2$ is called "REMAINDER POLYNOMIAL".

* If $f(x) = x^4+x+1$, then show that $[f(x)]^2 = f(x^2)$ in modulo-2 algebra.

Sol :- $[f(x)]^2 = [x^4+x+1]^2$

$= (x^4+x+1)(x^4+x+1)$

$= x^8+x^5+x^4+x^5+x^2+x+x^4+x+1$

$= x^8+x^5[1+1]+x^4[1+1]+x^2+x[1+1]+1$

$[f(x)]^2 = x^8+x^2+1$ ——— (1)

Consider $f(x) = x^4+x+1$.

$f(x^2) = (x^2)^4+x^2+1 = x^8+x^2+1$ ——— (2)

Comparing eq (1) and (2) $[f(x)]^2 = f(x^2)$

* For the (7,4) Single-error Correcting Cyclic Code, $D(x) = d_0 + d_1x + d_2x^2 + d_3x^3$ and $x^n + 1 = x^7 + 1 = (1+x+x^3)(1+x+x^2+x^4)$. Using the generator polynomial $g(x) = 1+x+x^3$, find all the 16 code vectors of the cyclic code, both for Non-Systematic and Systematic form.

DEC-2002 [10M]. 2006 [12M]. DEC 2016/JAN 2017 [10M]
DEC 2014/JAN 2015 [10M]

Sol :-

1. Non-Systematic Cyclic Code

* The non-Systematic Cyclic Code - vector polynomial $V(x)$ Can be found by using.

$$V(x) = D(x)\, g(x).$$

Eg :-

* Let us Consider a message vector. $= (0000)$

Assume, $D = (d_0\, d_1\, d_2\, d_3)$

* The message polynomial is.

$$D(x) = d_0 + d_1x + d_2x^2 + d_3x^3$$
$$= 0 + 0 + 0 + 0 = 0 //$$

$\therefore V(x) = D(x)\, g(x).$

$V(x) = 0 \cdot (1+x+x^3) = 0 //$

$\therefore [V] = 0000000 //$

Consider $D = (d_0\, d_1\, d_2\, d_3) = (0001)$.

$\therefore D(x) = 0 + 0 + 0 + (1)x^3 = x^3 //$

$V(x) = D(x)\, g(x)$

$[V(x)] = x^3 (1+x+x^3) = x^3 + x^4 + x^6$

$\therefore V = 0001101 //$

Consider. $D = (d_0 d_1 d_2 d_3) = 0010.$

$\therefore D(x) = 0 + 0 + x^2 = x^2$ //

$V(x) = D(x) g(x)$.

$= x^2[1 + x + x^3] = x^2 + x^3 + x^5$

$\therefore V = 0011010$ //

Now. $D = (d_0 d_1 d_2 d_3) = (1001)$

$D(x) = 1 + 0 + 0 + x^3 = 1 + x^3$ //

$V(x) = D(x) g(x) = (1+x^3)(1+x+x^3)$

$= 1 + x + x^3 + x^3 + x^4 + x^6$

$= 1 + x + x^4 + x^6$

$V = 1100101$ //

$D = (d_0 d_1 d_2 d_3) = (1111)$

$D(x) = 1 + x + x^2 + x^3$.

$V(x) = D(x) g(x)$

$= (1 + x + x^2 + x^3)(1 + x + x^3)$

$= 1 + x + x^2 + x^3 + x + x^2 + x^3 + x^4 + x^3 + x^4 + x^5 + x^6$.

$V(x) = 1 + x^3 + x^6$.

$\therefore V = 1001001$ //.

message (D)	codevector (V)	message	Codevector	message	Codevector
0000	0000000	0110	0101110	1100	1011100
0001	0001101	0111	0100011	1101	1010001
0010	0011010	1000	1101000	1110	1000110
0011	0010111	1001	1100101	1111	1001011
0100	0110100	1010	1110010		
0101	0111001	1011	1111111		

Table: Non-Systematic Code-vectors.

2. Systematic Cyclic code

The Remainder polynomial $R(x)$ is given by.

$$R(x) = \frac{x^{n-k} D(x)}{g(x)}$$

Example. Let $D = [0000]$.

$$D(x) = 0+0+0+0 = 0 //$$

$$x^{n-k} D(x) = 0 //$$

$$g(x) \overline{)\ D(x.} \quad (0$$

$$0 //$$

The code vector is given by.

$$[v] = [R][0]$$

$$= .000.0000 //.$$

Let $D = [0001]$.

$$D(x) = x^3 //$$

$$x^{n-k} D(x) = x^{7-4} D(x) = x^3(x^3) = x^6 //.$$

$$g(x) \overline{)\ D(x)} \quad \text{↳}$$

$$x^3+x+1 \overline{)\ x^6 \quad (\ x^3+x+1}$$

$$\underline{x^6+x^4+x^3}$$

$$x^4+x^3$$

$$\underline{x^4+x^2+x}$$

$$x^3+x^2+x$$

$$\underline{x^3+x+1}$$

$$x^2+1 // \leftarrow R(x).$$

$\therefore R(x) = 1+x^2$.

$[R] = 101$ //

The Code vector is given by

$[v] = [R][D] = 1010001$ //.

Let $D = [0010]$.

$D(x) = x^2$ //.

$x^{n-k} D(x) = x^3(x^2) = x^5$ //

$g(x) \overline{)\, D(x)}$ (.

$x^3+x+1 \overline{)\, x^5}$ (x^2+1

$x^5+x^3+x^2$

x^3+x^2.

x^3+x+1.

x^2+x+1 // ← $R(x)$.

$\therefore R(x) = x^2+x+1$. $\therefore R = [111]$ //

The Code vector $[v]$ is given by

$[v] = [R][D]$

$= 1110010$ //.

Illy

message (D)	Code-vector (v)	message	Code-vector	message	Code-vector
0000	0000000	0110	1000110	1011	1001011
0001	1010001	0111	0010111	1100	1011100
0010	1110010	1000	1101000	1101	0001101
0011	0100011	1001	0111001	1110	0101110
0100	0110100	1010	0011010	1111	1111111.
0101	1100101				

Table: Systematic Code vectors.

* For a (7, 4) binary cyclic code the generator polynomial is given by $g(x) = 1+X+X^3$ find the generator matrix and parity check matrix.

DEC-2010 [10M]

Sol:- $n = 7$ (code length) (columns) $K = 4$ (message bits) (ROWS).

- $g(x) = 1+x+x^3$.

$g(x) = 1 + (1)x + (1)x^3 + (0)x^4 + (0)x^5 + (0)x^6$.

$[g] = 1\ 101\ 000$. //

14 $xg(x) = x(1+x+x^3)$

$= x + x^2 + x^4$

The code-vector corresponding to $xg(x)$ is 0110100

$x^2g(x) = x^2(1+x+x^3)$

$= x^2 + x^3 + x^5$ → code is 0011010

$x^3g(x) = x^3(1+x+x^3)$

$= x^3 + x^4 + x^6$ → code is 0001101

The generator matrix [G] is given by.

$$[G] = \begin{bmatrix} 1\ 1\ 0\ 1\ 0\ 0\ 0 \\ 0\ 1\ 1\ 0\ 1\ 0\ 0 \\ 0\ 0\ 1\ 1\ 0\ 1\ 0 \\ 0\ 0\ 0\ 1\ 1\ 0\ 1 \end{bmatrix}$$

NOTE:
→ $g(x)$ code.
→ $xg(x)$ code.
→ $x^2g(x)$ code.
→ $x^3g(x)$ code

NOTE: actually [G] is non-systematic form. To arrange the systematic form

$R_3 \leftarrow R_3 + R_1$

$$\therefore [G] = \begin{bmatrix} 1101000 \\ 0110100 \\ 1110010 \\ 0001101 \end{bmatrix}$$

$R_4 \leftarrow R_4 + R_2 + R_1$

$$\therefore [G] = \begin{bmatrix} 1101000 \\ 0110100 \\ 1110010 \\ 1010001 \end{bmatrix}$$

$[G] = [P \mid I_4]$ form.

The Systematic Cyclic Code vector for given by

$$[V] = [D][G]$$

$$= [d_0 d_1 d_2 d_3] \begin{bmatrix} 1101000 \\ 0110100 \\ 1110010 \\ 1010001 \end{bmatrix}$$

$$[V] = [(d_0 + d_2 + d_3), (d_0 + d_1 + d_2), (d_1 + d_2 + d_3), d_0, d_1, d_2, d_3]$$

parity check matrix "H" is given by.

$$x^n + 1 = g(x)\, h(x)$$

$$\therefore h(x) = \frac{x^n + 1}{g(x)}$$

$$h(x) = \frac{x^7 + 1}{g(x)}$$

$g(x) \overline{)\, x^7 + 1\ (}$

$x^3 + x + 1 \overline{)\, x^7 + 1\ (\ x^4 + x^2 + x + 1}$ ← $h(x)$

$x^7 + x^5 + x^4$

$x^5 + x^4 + 1$

$x^5 + x^3 + x^2$

$x^4 + x^3 + x^2 + 1$

$x^4 + x^2 + x$

$x^3 + x + 1$

$x^3 + x + 1$

0 //

$\therefore$ the parity check polynomial $h(x) = 1 + x + x^2 + x^4$

we have $h(x) = 1 + x + x^2 + x^4$

$h(x^{-1}) = 1 + \frac{1}{x} + \frac{1}{x^2} + \frac{1}{x^4}$

$x^4 h(x^{-1}) = x^4 + x^3 + x^2 + x + 1$

$x^4 h(x^{-1}) = x^4 + x^3 + x^2 + 1$.

(6m)

$= 1 + x^2 + x^3 + x^4$. → code is 1011100

$x^5 h(x^{-1}) = x^5 + \frac{x^5}{x} + \frac{x^5}{x^2} + \frac{x^5}{x^4}$

$= x^5 + x^4 + x^3 + x$

(6m)

$= x + x^3 + x^4 + x^5$ → code is 0101110

$x^6 h(x^{-1}) = x^6 + \frac{x^6}{x} + \frac{x^6}{x^2} + \frac{x^6}{x^4}$.

$= x^6 + x^5 + x^4 + x^2$.

(6m) $= x^2 + x^4 + x^5 + x^6$ → code is 0010111

The parity check matrix "H" which is a $(n-k) \times n$ = $(7-4) \times 7$ = 3×7 matrix is given by

$$[H] = \begin{bmatrix} 1011100 \\ 0101110 \\ 0010111 \end{bmatrix}$$

NOT E. — non systematic form.

$r_1 \leftarrow r_1 + r_3$

$$[H] = \left[\begin{array}{ccc|cccc} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array}\right]$$

$\therefore [H] = [I_{n-k} \mid P^T]$

$= [I_3 \mid P^T]$

* Design an Encoder for the (7,4) binary cyclic code generated by $g(x) = 1 + x + x^3$ and verify it's operation using the message vectors (1001) and (1011).

Sol:- DEC-2012 [10M] 2017 [10M], JUNE/JULY 2011 [8M]

In general the generator polynomial is represented as,

$$g(x) = g_0 + g_1 x + g_2 x^2 + \cdots + g_{n-k} x^{n-k} \quad \text{---(1)}$$

In the problem (7,4) cyclic code.

$$g(x) = 1 + x + x^3 \quad \text{---(2)}$$

∴ Comparing the coefficient in eq (1) and (2) we get

$g_0 = 1$ $g_1 = 1$ $g_2 = 0$ and $g_3 = 1$.

∴ To write the Encoding Circuit we have $(n-k) \rightarrow (7-4) = 3$ flip flop i.e. R_0, R_1 and R_2. and 2 modulo-2 adder.

i.e. The last generation needed is $g_{n-k-1} = g_{7-4-1} = g_2$

∴ 2nd modulo.

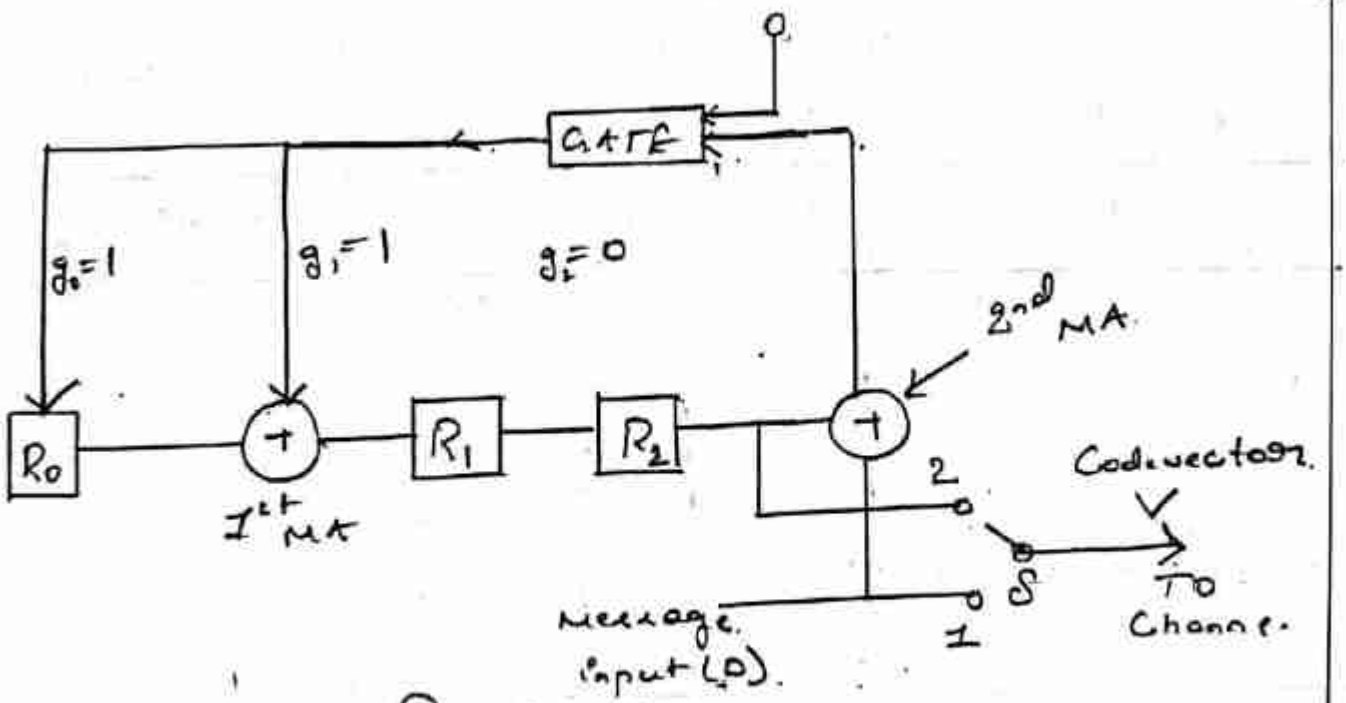


Fig:- Encoder for (7,4) Cyclic Code

1) For the message. D = [1001]. The Shift Register Contents.

Number of Shifts.	Input D	Shift Register Contents. R_0 R_1 R_2	Remainder bits → R.
Initialization → Switch "S" is in position-1 and gate is turned ON		0 0 0	—
1	1	1 1 0	—
2	0	0 1 1	—
3	0	1 1 1	—
4	1	0 1 1	—
Switch "S" moves to position-2 and gate is turned OFF			
5	X	0 0 1	1
6	X	0 0 0	1
7	X	0 0 0	0

The code vector. $[v] = [R][D] = 0111001$ //.

2) For the message D = [1011]

Number of Shifts.	Input D	Shift Register Contents R_0 R_1 R_2	Remainder bits → R.
Initialization → Switch "S" is in position-1 and gate is turned ON.		0 0 0	
1	1	1 1 0	
2	1	1 0 1	
3	0	1 0 0	
4	1	1 0 0	
Switch "S" moves to position-2 and gate is turned OFF			
5	X	0 1 0	0
6	X	0 0 1	0
7	X	0 0 0	1

The code vector. $[v] = [R][D] = 1001011$.

Verification:- 1) for $D=[1011]$

$\therefore D(x) = 1+x^2+x^3$

$g(x) = 1+x+x^3$

The remainder polynomial $R(x)$ given by.

$$R(x) = \frac{x^{n-k} D(x)}{g(x)} = \frac{x^{7-4} D(x)}{g(x)} = \frac{x^3(1+x^2+x^3)}{1+x+x^3}$$

$$R(x) = \frac{x^3+x^5+x^6}{1+x+x^3}$$

$$x^3+x+1 \;)\; x^6+x^5+x^3 \;(\; x^3+x^2+x+1$$

$x^6+x^4+x^3$

x^5+x^4

$x^5+x^3+x^2$

$x^4+x^3+x^2$

x^4+x^2+x

x^3+x

x^3+x+1

1 ← $R(x)$

$\therefore$ Code vector $[v] = [1001011] = RD$

(R. D.)

2) for $D=[1001]$

$\therefore D(x) = 1+x^3$ // $\quad g(x) = 1+x+x^3$

$$R(x) = \frac{x^{n-k} D(x)}{g(x)} = \frac{x^3(1+x^3)}{1+x+x^3} = \frac{x^3+x^6}{1+x+x^3}$$

$$x^3+x+1 \;)\; x^6+x^3 \;(\; x^3+x$$

$x^6+x^4+x^3$

x^4

x^4+x^2+x

x^2+x // ← $R(x)=011$

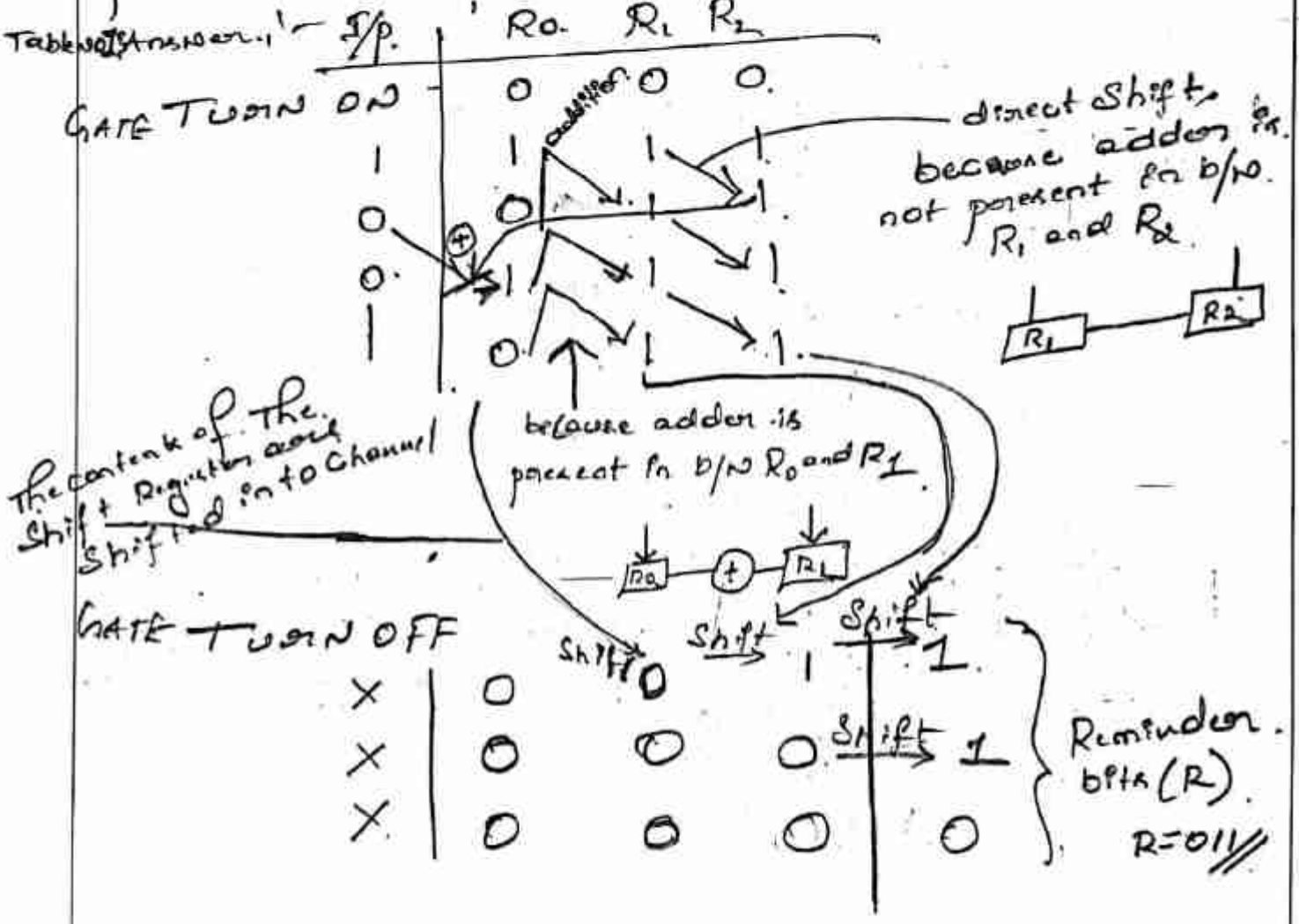
Codevector $[v] = [Q][D]$

$= 0111001$ //

NOTE :-

Ex:- Look at the Table number one for the procedure problem.

Table solution :-



* Consider a (15, 11) Cyclic Code generated by $g(x) = 1+X+X^4$. Devise a feed back shift Register Encoder circuit. Illustrate the Encoding procedure with the message vector 10010110111 by listing the state of the Register.

Sol :-

In general the generator polynomial is represented as,

$$g(x) = g_0 + g_1X + g_2X^2 + \cdots + g_{n-k}X^{n-k} \quad \text{---(1)}$$

In the problem given, for (15, 11) Cyclic code.

$$g(x) = 1+X+X^4. \quad \text{---(2)}$$

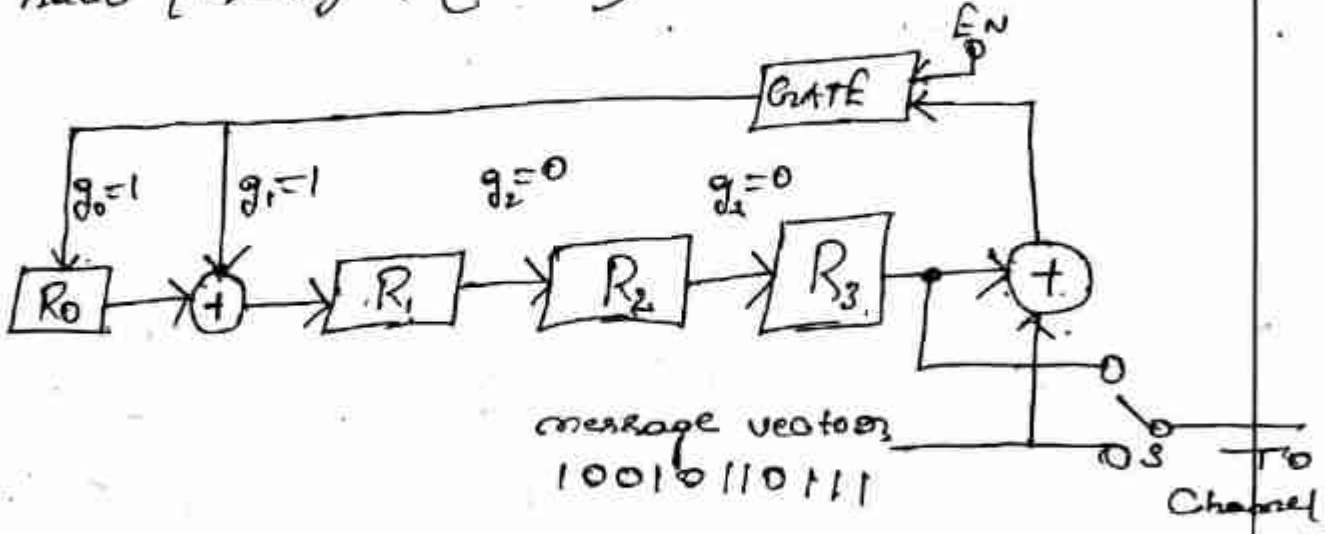
Compare eq (1) and (2)

$g_0 = 1 \quad g_1 = 1 \quad g_2 = 0 \quad g_3 = 0 \quad g_4 = 1.$

$R(n-k-1) = R(15-11-1) = R_3$

∴ To locate the Encoding Circuit.

We have $(n-k) \rightarrow (15-11) = 4$ flipflops i.e $R_0 R_1 R_2$ and R_3.



Number of Shifts.	Input "D"	Shift Register Contents R_0 R_1 R_2 R_3	Remainder bits R
Initialization → Switch "S" is in position -1 and gate is turned ON.		0 0 0 0	
1	1	1 1 0 0	
2	1	1 0 1 0	
3	1	1 0 0 1	
4	0	1 0 0 0	
5	1	1 0 0 0	
6	1	1 0 0 0	
7	0	0 1 0 0	
8	1	1 1 1 0	
9	0	0 1 1 1	
10	0	1 1 1 1	
11	1	0 1 1 1	
Switch "S" moves to position -2 and gate is turned off.			
12	X	0 0 1 1	1
13	X	0 0 0 1	1
14	X	0 0 0 0	1
15	X	0 0 0 0	0

Table:- Encoding process of a (15, 11) cyclic code.

The Code vector. $[v] = [R][D] = 011110010111011$.

Verification:- $g(x) = 1+x+x^4$.

$D = 10010110111$.

$D(x) = 1+x^3+x^5+x^6+x^8+x^9+x^{10}$.

$$R(x) = \frac{x^{n-k} D(x)}{g(x)} = \frac{x^{15-11}}{g(x)} = x^4\left(\frac{1+x^3+x^5+x^6+x^8+x^9+x^{10}}{1+x+x^4}\right)$$

$$= \frac{x^4+x^7+x^9+x^{10}+x^{12}+x^{13}+x^{14}}{1+x+x^4}.$$

$x^4+x+1 \;)\; x^{14}+x^{13}+x^{12}+x^{10}+x^9+x^7+x^4 \;(\; x^{10}+x^9+x^8+x^7+x^6+x^5+x^3+x$

$x^{14}+x^{11}+x^{10}$

$x^{13}+x^{12}+x^{11}+x^9+x^7+x^4$

$x^{13}+x^{10}+x^9$

$x^{12}+x^{11}+x^{10}+x^7+x^4$

$x^{12}+x^9+x^8$

$x^{11}+x^{10}+x^9+x^8+x^7+x^4$

$x^{11}+x^8+x^7$

$x^{10}+x^9+x^4$

$x^{10}+x^7+x^6$

$x^9+x^7+x^6+x^4$

$x^9+x^6+x^5$

$x^7+x^5+x^4$

$x^7+x^4+x^3$

x^5+x^3

x^5+x^2+x

$x^2+x+0 \leftarrow R(x)$

$R = 0111$

Codeword $(v) = [R][D]$

$v = 0111\,1001\,0111\,0111$

* A Linear Hamming code is described by a generator polynomial $g(D) = 1+D+D^3$

1) Determine the generator matrix "G" and parity check matrix
2) Design an Encoder Circuit.

June – 2010 [12 M]

Sol :-

1) $g(D) = 1+D+D^3 \rightarrow$ code is [1101000]
$D\,g(D) = D+D^2+D^4 \rightarrow$ code is [0110100]
$D^2 g(D) = D^2+D^3+D^5 \rightarrow$ code is [0011010]
$D^3 g(D) = D^3+D^4+D^6 \rightarrow$ code is [0001101]

The generator matrix [G] is given by.

$$[G] = \begin{bmatrix} 1101000 \\ 0110100 \\ 0011010 \\ 0001101 \end{bmatrix}$$

$R_3 \leftarrow R_3 + R_4$

$$[G] = \begin{bmatrix} 1101000 \\ 0110100 \\ 1110010 \\ 0001101 \end{bmatrix}$$

$R_4 \leftarrow R_4 + R_1$

$$[G] = \begin{bmatrix} 1101000 \\ 0110100 \\ 1110010 \\ 1100101 \end{bmatrix}$$

$R_4 \leftarrow R_4 + R_2$

$$G = \begin{bmatrix} 1101000 \\ 0110100 \\ 1110010 \\ 1010001 \end{bmatrix}$$

where $n = 7$ (Columns) $\quad k = 4$ (rows)

$$G = \left[P \,!\, I_k \right] = \left[P \,!\, I_4 \right] //.$$

where $P = \begin{bmatrix} 110 \\ 011 \\ 111 \\ 101 \end{bmatrix} \quad \therefore P^T = \begin{bmatrix} 1011 \\ 1110 \\ 0111 \end{bmatrix}$

The parity check matrix $[H]$ is given by

$$[H] = \left[I_{n-k} \,!\, P^T \right] = \left[I_{7-4} \,!\, P^T \right] = \left[I_3 \,!\, P^T \right]$$

$$[H] = \begin{bmatrix} 100 & 101 \\ 010 & 1110 \\ 001 & 0111 \end{bmatrix} //.$$

2) In general the generator polynomial is represented as,

$$g(D) = g_0 + g_1(D) + g_2 D^2 + \cdots + g_{n-k} D^{n-k} \quad \text{---(1)}$$

In the problem given.. for (7, 4) Linear Hamming code.

$$g(D) = 1 + D + D^3 \quad \text{---(2)}$$

Compare eq (1) and (2)

$g_0 = 1 \quad g_1 = 1 \quad g_2 = 0 \quad g_3 = 1$

To write the Encoding Circuit

we have $(n-k) \rightarrow (7-4) = 3$ flipflops i.e R_0 R_1 and R_2

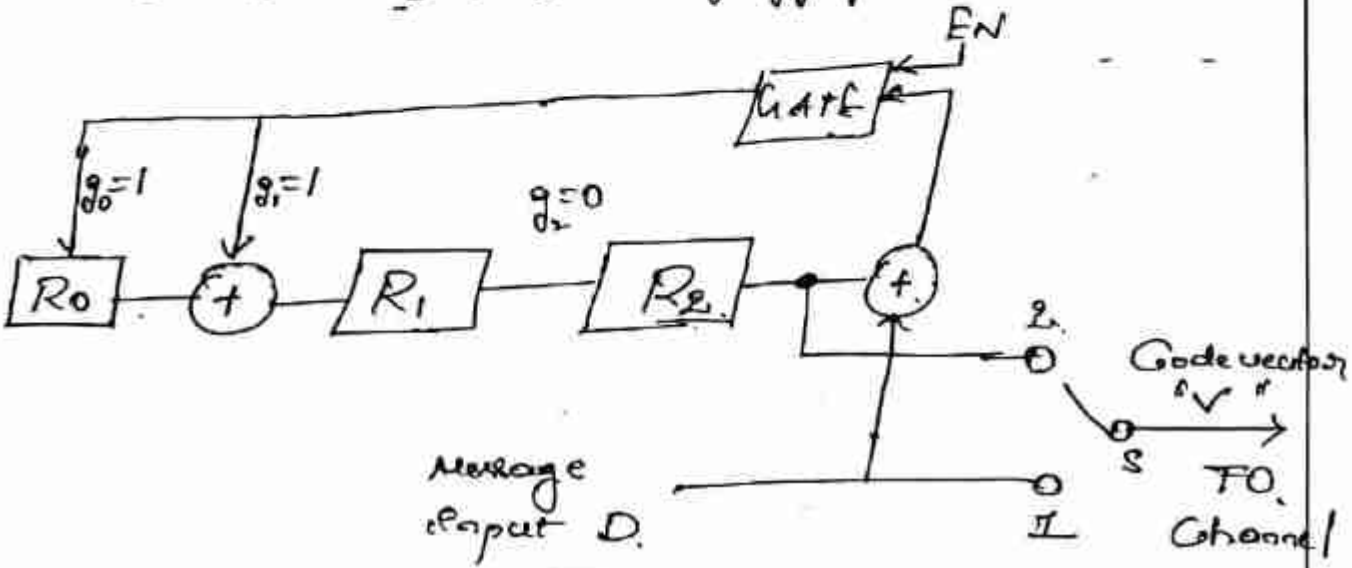


Fig:- Encoder Circuit.

2010

* A. (15, 5) Linear Cyclic code has a generator polynomial

$$g(x) = 1+x+x^2+x^4+x^5+x^8+x^{10}$$

a) Draw the block diagram of the Encoder.

b) Find the Code polynomial for the message $1+x^3+x^4$ by listing the states of the registers with each binary message input.

c) Verify the validity of the Code vector so. formed. by using division method.

2010 [6M]

Sol :- In general the generator polynomial is represented as

a) $$g(x) = g_0 + g_1 x + g_2 x^2 + \cdots + g_{n-k} x^{n-k} \quad \text{---(1)}$$

In the problem given .. for (15, 5) Cyclic Code

$$g(x) = 1+x+x^2+x^4+x^5+x^8+x^{10} \quad \text{---(2)}$$

Compare eq (1) and (2)

$g_0=1$ $g_1=1$ $g_2=1$ $g_3=0$ $g_4=1$ $g_5=1$ $g_6=0$ $g_7=0$

$g_8=1$ $g_9=0$ $g_{10}=1$.

To locate the Encoding Circuit

We have $(n-k) \rightarrow (15-5) = 10$ flipflops i.e. $R(n-k-1) \rightarrow R(15-5-1)$

$\Rightarrow R_0\ R_1\ R_2\ R_3\ R_4\ R_5\ R_6\ R_7\ R_8$ and R_9 //

P.T.O.

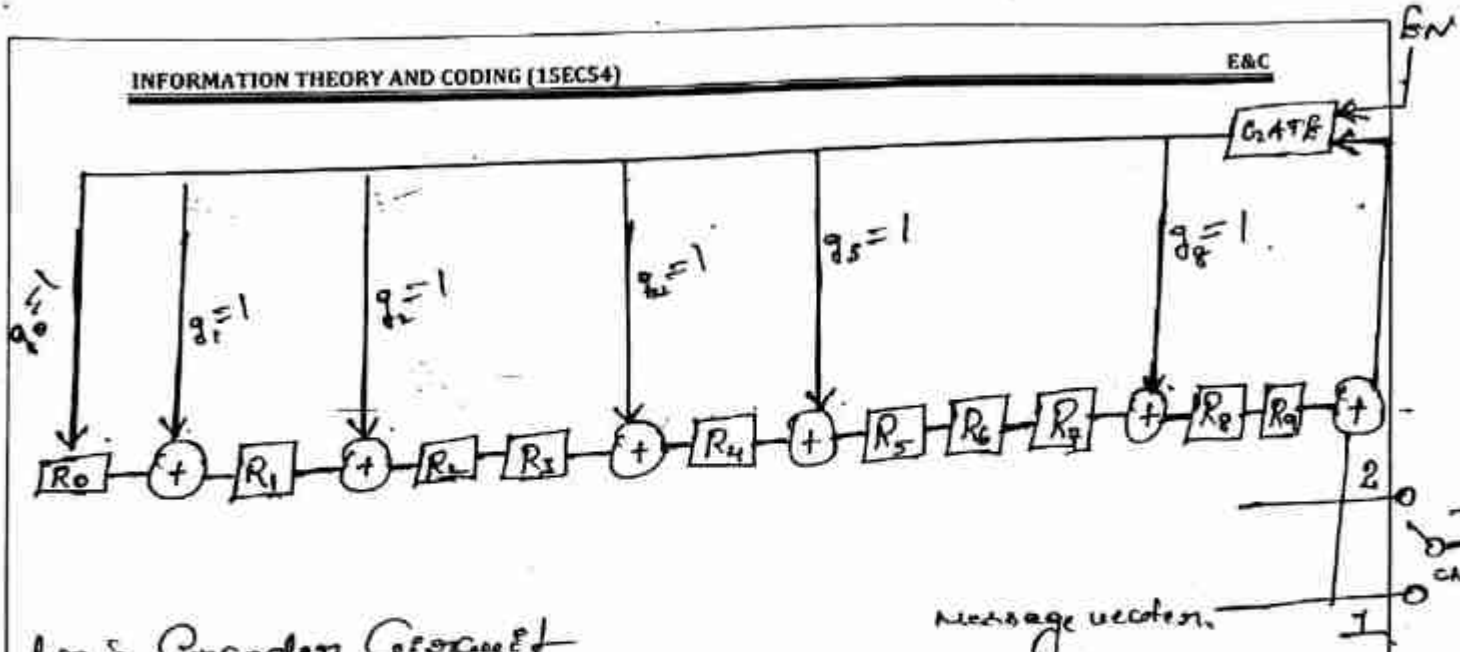


fig:- Encoder Circuit

b) Let message input of $D(x) = 1+x^3+x^4$

$D = .10011.$

Number of shift	Input D	Shift Register Content R_0 R_1 R_2 R_3 R_4 R_5 R_6 R_7 R_8 R_9	Remainder R
Initialization.		0 0 0 0 0 0 0 0 0 0	
GATE is turned ON		1 1 1 0 1 1 0 0 1 0	
1	1	1 0 0 1 1 0 1 0 1 1	
2	1	1 0 1 0 0 0 0 1 1 1	
3	0	1 0 1 1 1 1 0 0 0 1	
4	0	0 1 0 1 1 1 1 0 0 0	
5	1	0 0 1 0 1 1 1 1 0 0	0
Turn OFF GATE.		0 0 0 1 0 1 1 1 1 0	0
6	x	0 0 0 0 1 0 1 1 1 1	0
7	x	0 0 0 0 0 1 0 1 1 1	1
8	x	0 0 0 0 0 0 1 0 1 1	1
9	x	0 0 0 0 0 0 0 1 0 1	1
10	x	0 0 0 0 0 0 0 0 1 0	0
11	x	0 0 0 0 0 0 0 0 0 1	1
12	x	0 0 0 0 0 0 0 0 0 0	1
13	x	0 0 0 0 0 0 0 0 0 0	0
14	x	0 0 0 0 0 0 0 0 0 0	1

Code vector. $V = RD$.

Code vector. $= 0101111000100111$

e.g. Code vector polynomial is $V(x) = x + x^3 + x^4 + x^5 + x^6 + x^{10} + x^{13} + x^{14}$

c) Division method. (verification)

$$R(x) = \frac{x^{n-k} D(x)}{g(x)} = \frac{x^{10}(x^4 + x^3 + 1)}{x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1} = \frac{x^{14} + x^{13} + 1}{g(x)}$$

$$\begin{array}{r|l}
x^{10}+x^8+x^5+x^4+x^2+x+1 \;\big)\; x^{14}+x^{13}+x^{10} & \big(\; x^4 + x^3 + x^2 \\
x^{14}+x^{12}+x^9+x^8+x^6+x^5+x^4 & \\
\hline
x^{13}+x^{12}+x^{10}+x^9+x^8+x^6+x^5+x^4 & \\
x^{13}+x^{11}+x^8+x^7+x^5+x^4+x^3 & \\
\hline
x^{12}+x^{11}+x^{10}+x^9+x^7+x^6+x^3 & \\
x^{12}+x^{10}+x^7+x^6+x^4+x^3+x^2 & \\
\hline
x^{11}+x^9+x^4+x^2 & \\
x^{11}+x^9+x^6+x^5+x^3+x^2+x & \\
\hline
x^6+x^5+x^4+x^3+x & \leftarrow R(x).
\end{array}$$

Remainder polynomial $R(x) = x + x^3 + x^4 + x^5 + x^6$.

Code vector $V = RD$.

$= 0101111000100111$

e.g. Code vector polynomial $= R(x) + x^{10} D(x)$

$V(x) = x + x^3 + x^4 + x^5 + x^6 + x^{10}(1 + x^3 + x^4)$

$= x + x^3 + x^4 + x^5 + x^6 + x^{10} + x^{13} + x^{14}$

* Consider the (15,11) cyclic code generated by $g(x) = 1+X+X^4$.

(a) Draw the feedback register encoding circuit for this cyclic code.

(b) Illustrate the encoding procedure with the message vector 0110100 1011 by listing the states of the register with each input

(c) Verify the code polynomial by using the division method.

Sol:- The generator polynomial is given by

$$g(x) = 1+X+X^4$$
$$= g_0+g_1X+g_2X^2+g_3X^3+g_4X^4$$

Comparing $g_0=1$ $g_1=1$ $g_2=0$ $g_3=0$ $g_4=1$

To write the encoding circuit

We have $(n-k) \rightarrow (15-11) = 4$ flip flops i.e R_0, R_1, R_2 & R_3. //

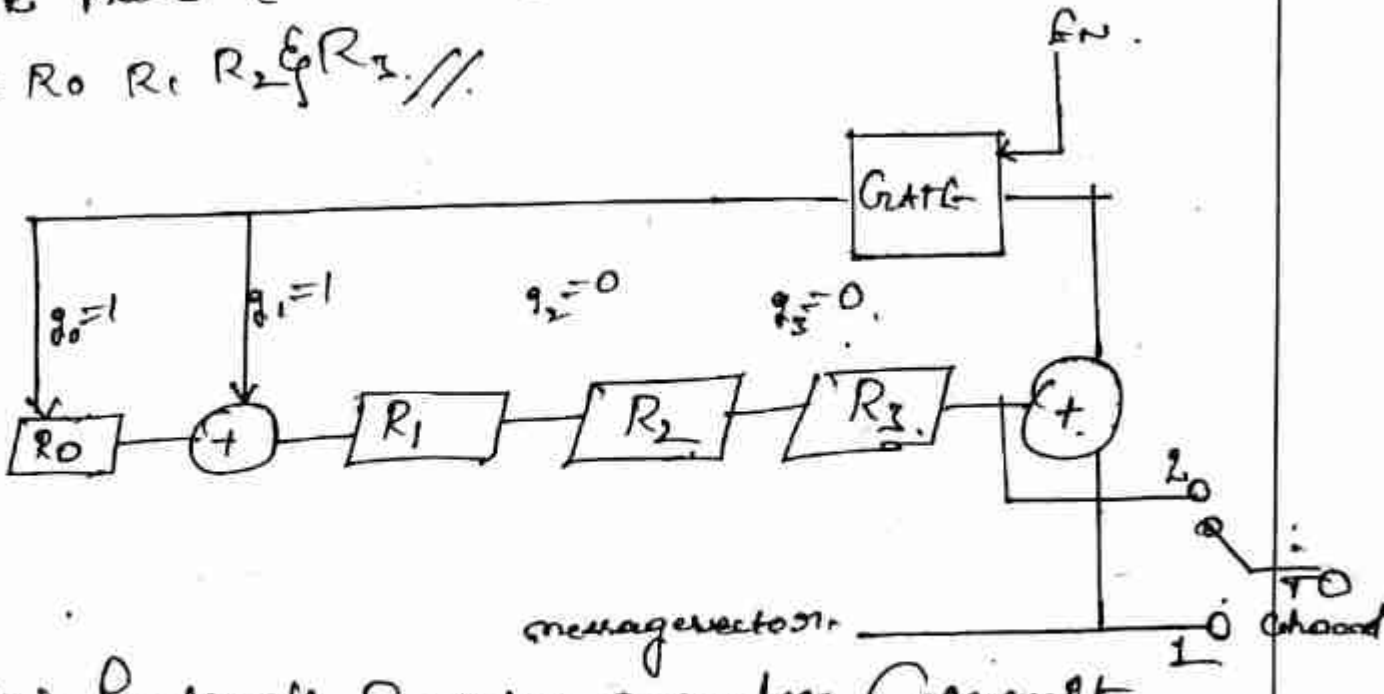


fig:- Feedback Register encoder Circuit

b) message vector $D = 0110\,1001\,011$

Number of shifts	Input D_0	Shift Register Contents R_0	R_1	R_2	R_3	Remainder R
Initialization		0	0	0	0	
Gate Turned ON		1	1	0	0	
1	1	1	0	1	0	
2	1	0	1	0	1	
3	0	0	0	1	0	
4	1	0	0	0	1	
5	0	0	0	0	0	
6	0	1	1	0	0	
7	1	1	0	1	0	
8	0	0	1	0	1	
9	1	0	0	1	0	
10	1	1	1	0	1	
11	0	1	0	1	0	
Switch S moved to position '2' and gate is turned off.						
	X	0	1	0	1	0
	X	0	0	1	0	1
	X	0	0	0	1	0
	X	0	0	0	0	1

∴ Code vector $[v] = [R]\,[D]$

$= 1010\,0110\,1001\,011$

NOTE.

Message D = 01101001011 $\Rightarrow D(x) = (x + x^2 + x^4 + x^7 + x^9 + x^{10})$

c) Verification using division method.

$$R(x) = \frac{x^{n-k} D(x)}{g(x)} = \frac{x^4 (D(x))}{g(x)} = \frac{x^5 + x^6 + x^8 + x^{11} + x^{13} + x^{14}}{1 + x + x^4}$$

$x^4 + x + 1$) $x^{14} + x^{13} + x^{11} + x^8 + x^6 + x^5$ ($x^{10} + x^9 + x^5 + x^4 + x + 1$

$x^{14} + x^{11} + x^{10}$

$x^{13} + x^{10} + x^8 + x^6 + x^5$

$x^{13} + x^{10} + x^9$

$x^9 + x^8 + x^6 + x^5$

$x^9 + x^6 + x^5$

x^8

$x^8 + x^5 + x^4$

$x^5 + x^4$

$x^5 + x^2 + x$

$x^4 + x^2 + x$

$x^4 + x + 1$

$x^2 + 1$ // ← R(x)

Code-vector polynomial $= R(x) + x^{n-k} D(x)$

$V(x) = 1 + x^2 + x^4 (x + x^2 + x^4 + x^7 + x^9 + x^{10})$

$= 1 + x^2 + x^5 + x^6 + x^8 + x^{11} + x^{13} + x^{14}$ //

Code vector "V" = R D (15,11)

= 1010 01101001011 n=15 o/p

R D

Code length

* Consider a (15, 11) Cyclic code generated using $g(x) = 1 + x + x^4$.

a) Design a feedback register Encoder for the same

b) Illustrate the Encoding procedure with the Message vector [11001101011] by listing the States of the register alongwith right most bit as the Current bit.

June/July-2012 [10M]. Jan/July-2014

Sol:- The generator polynomial is given by

a) $g(x) = 1 + x + x^4$.

$= g_0 + g_1 x + g_2 x^2 + g_3 x^3 + g_4 x^4$.

$g_0 = 1 \quad g_1 = 1 \quad g_2 = 0 \quad g_3 = 0 \quad g_4 = 1$

To realize the Encoding Circuit we have $(n-k) \rightarrow (15-11) = 4$ flipflops i.e $R_{(n-k-1)} = R_{(15-11-1)} = R_3$ i.e (R_0, R_1, R_2, R_3).

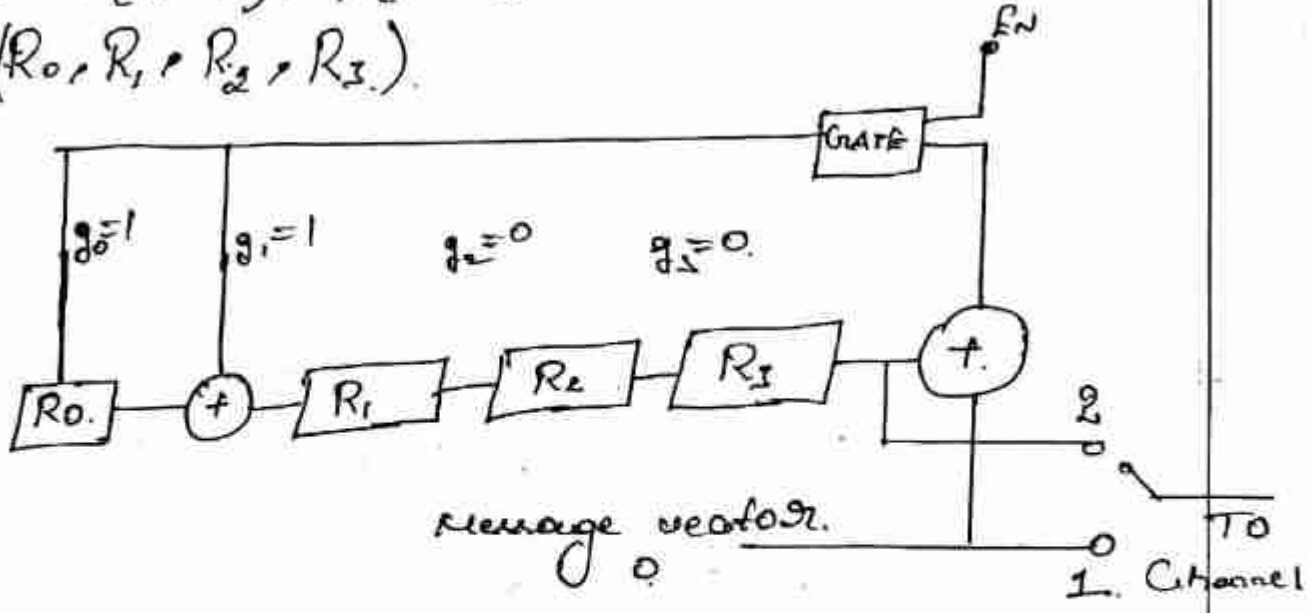


Fig:- Encoder Circuit

b)

No of shifts	Input D.	Shift Register. R_0	R_1	R_2	R_3	Remainder (R).
Initialisation.		0	0	0	0	
1	1	1	1	0	0	
2	1	1	0	1	0	
3	0	0	1	0	1	
4	1	0	0	1	0	
5	0	0	0	0	1	
6	1	0	0	0	0	
7	1	1	1	0	0	
8	0	0	1	1	0	
9	0	0	0	1	1	
10	1	0	0	0	1	
11	1	0	0	0	0	
GATE Turnoff						
12	X	0	0	0	0	0
13	X	0	0	0	0	0
14	X	0	0	0	0	0
15	X	0	0	0	0	0

Code vector $[V] = [R][D]$

$= 0000.1100110101 1.$

MQP Im.

* The generator polynomial of a (15, 7) Cyclic Code is $g(x) = 1+X^4+X^6+X^7+X^8$. Find the Code-word in Systematic form for the message (01010101) clearly Explaining all the steps. In particular list the States of Registers on each step of Code Computation. Verify your Answer by direct hand Calculation.

JULY/AUG - 2013 [7 M]

Sol :-

The generator polynomial is given by

$$g(x) = 1+X^4+X^6+X^7+X^8.$$

$g_0=1$ $g_1=0$ $g_2=0$ $g_3=0$ $g_4=1$ $g_5=0$ $g_6=1$ $g_7=1$

To locate the Encoding Circuit

we have $(n-k) \rightarrow (15-7) = 8$ flip flops.

i.e $R_{(n-k-1)} = R_{(15-7-1)} = R_7$ (i.e $(R_0$ to $R_7)$ Shift Registers)

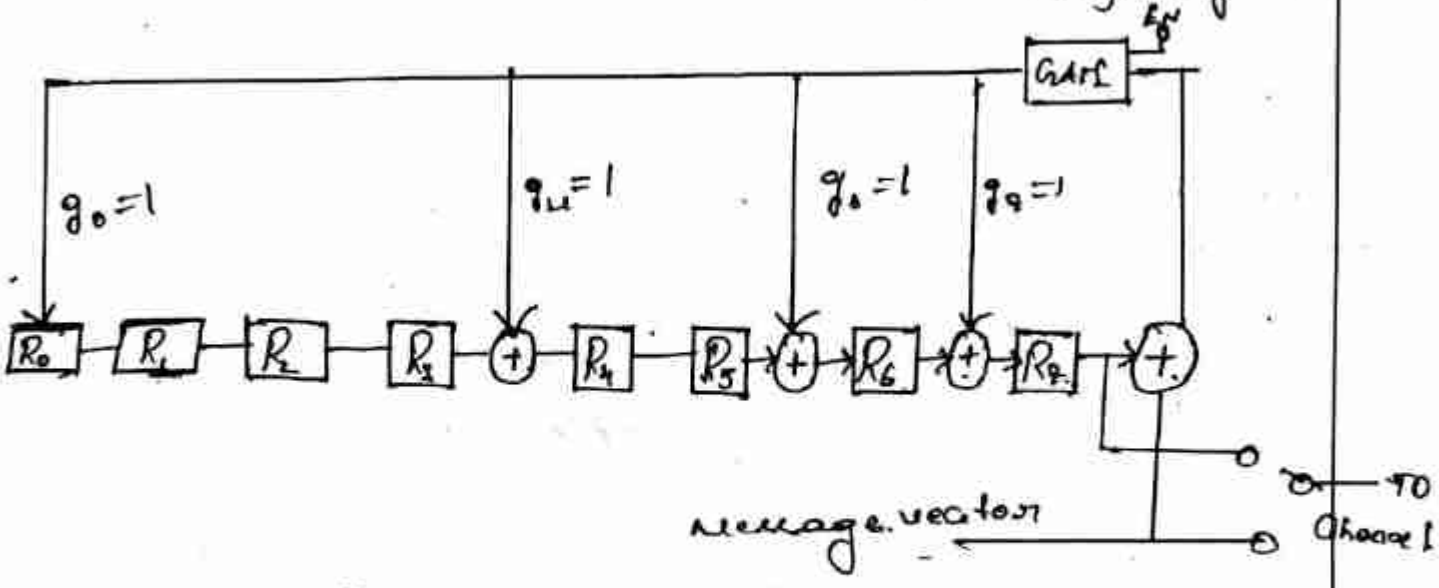


Fig: Encoder Circuit

Number of Shifts.	Input D.	Shift Register Contents R_0	R_1	R_2	R_3	R_4	R_5	R_6	R_7
Initialization		0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	1	0	1	1
3	0	1	1	0	0	1	1	1	0
4	1	1	1	1	0	1	1	0	0
5	0	0	1	1	1	0	1	1	0
6	1	1	0	1	1	0	0	0	
7.	0.	0	1	0	1	1	0	0	0

Remainder R.

Code vector $[V] = [R]\,[D]$

$= [01011000\ 0101010]$

Coefficient of $R(x)$ — message vector //

Direct hand Calculation:-

The Remainder polynomial $R(x)$ is obtained by dividing $X^{n-k} D(x)$ by $g(x)$

We have $[D] = [0101010]$

$D(x) = X + X^3 + X^5$

$X^{n-k} D(x) = X^{15-7}(X + X^3 + X^5)$

$= X^{13} + X^{11} + X^9$ //

Division of $X^{n-k} D(X)$ by $g(X)$ is shown as below.

$$X^8+X^7+X^6+X^4+1 \;\big)\; X^{13}+X^{11}+X^9 \;\big($$

$X^{13}+X^{12}+X^{11}+X^9+X^5$

$X^{12}+X^5$

$X^{12}+X^{11}+X^{10}+X^8+X^4$

$X^{11}+X^{10}+X^8+X^5+X^4$

$X^{11}+X^{10}+X^9+X^7+X^3$

$X^9+X^8+X^7+X^5+X^4+X^3$

$X^9+X^8+X^7+X^5+X$

X^4+X^3+X

$\therefore$ Remainder polynomial $R(X) = X+X^3+X^4$

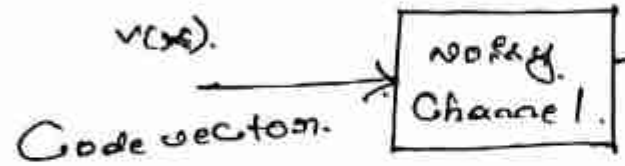
$R = [01011000]$.

Code vector V is given by

$[V] = [010110000101010]$ //

ERROR DETECTION AND CORRECTION

The Codevector "V" is Transmitted over a noisy Communication Channel.



$$[Z] = [V] + [E] \quad \text{---} (1)$$

Where. 'E' Represents the error Introduced by the Channel.

To prove that $S(x) \propto E(x)$: Syndrome $S(x)$

Consider eq (1). Writing as polynomial function in 'x'

$$Z(x) = v(x) + E(x) \quad \text{---} (2)$$

$\div$ throughout equation (2) by $g(x)$

$$\frac{Z(x)}{g(x)} = \frac{V(x)}{g(x)} + \frac{E(x)}{g(x)}$$

But $V(x) = D(x) \cdot g(x)$

$$\therefore \frac{Z(x)}{g(x)} = D(x) + \frac{E(x)}{g(x)} \quad \text{---} (3)$$

But The Syndrome. Can also be obtained as The Remainder of $\frac{Z(x)}{g(x)}$ division. Thus Can be represented in the form of an Equation

$$Z(x) = q(x) * g(x) + S(x)$$

$\div$ $g(x)$

$$\frac{Z(x)}{g(x)} = q(x) + \frac{S(x)}{g(x)} \quad \text{---} (4)$$

Comparing eq (3) and (4)

$$D(x) + \frac{E(x)}{g(x)} = q(x) + \frac{S(x)}{g(x)}$$

$$\frac{E(x)}{g(x)} = [q(x) - D(x)] + \frac{S(x)}{g(x)}$$

$$E(x) = g(x)[q(x) - D(x)] + S(x)$$

But we know modulo addition and subtraction is same

$$\therefore E(x) = g(x)[q(x) + D(x)] + S(x) \quad (5)$$

Equation (5) it is clear that the error polynomial is directly proportional to Syndrome polynomial $S(x)$

hence Syndrome can be measure of channel Induced Errors //.

SYNDROME CALCULATION CIRCUIT

The Syndrome Calculation Circuit performs the division of the Received vectors by the generator and Stores the Remainder in the flip flops of a shift Register.

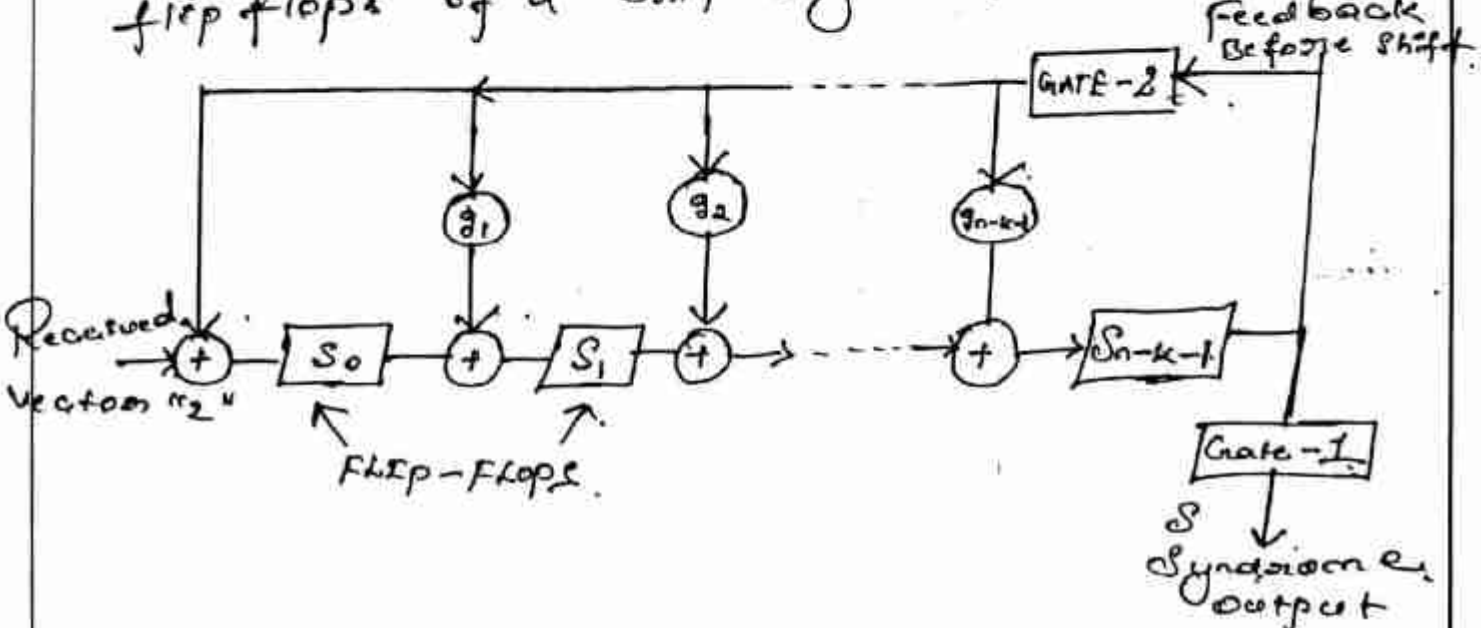


Fig :- A (n-k) Syndrome Calculation Circuit For a (n, k) Cyclic code.

The Syndrome Calculation are Carried out as below

1) The Register is first initialized. The with gate-2 turned ON and gate-1 OFF, The Received vector "Z" is entered into Shift Register.

2) After the Entire received vector is shifted into the Register, The Content of the Register will be Syndrome. NOW gate-2 is turned "off" gate-1 ON and the Syndrome vector is Shifted out of the Register. The Circuit is ready for processing the next Received vector.

GENERAL DECODER CIRCUIT FOR (n, k) CYCLIC CODES OR DECODING OF CYCLIC CODES

JUNE/JULY 2010 [10M], DEC 2013/2014 [10M], DEC 2014/JAN 2013 [10M]

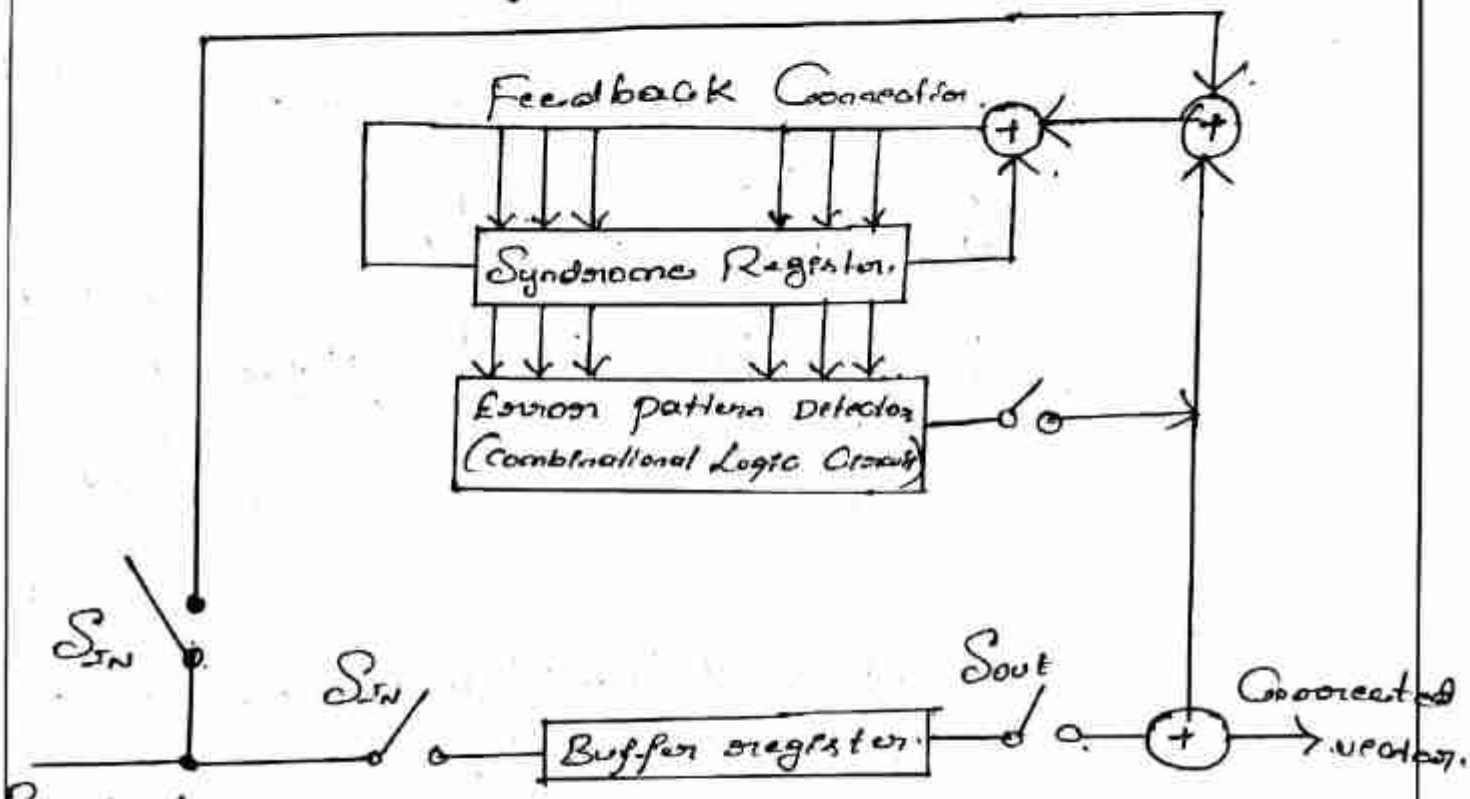


Fig :- General form of decoder for Cyclic Codes.

Error Correction procedure Consists of the following Steps.

STEP 1 :-

The Received vector is Shifted into the buffer register and the Syndrome Register.

STEP-2 :-

→ After the Syndrome for the Received vector is Calculated and placed in the Syndrome register, the Contents of the Syndrome Register are fed into the detector.

→ The detector is a Combinational Logic Circuit designed to output a "1" if and only if the Syndrome in the Syndrome register corresponds to a Correctable error pattern with an error at the highest order position.

→ If the detector o/p is "1" then the Received digit at the Right most Stage of the buffer register is assumed to be erroneous and hence is Corrected.

→ If the detector o/p is '0' then the Received digit at the Right most Stage of the buffer register is assumed to be Correct.

→ The detector o/p is the Estimated error value for digit coming out of the buffer register.

STEP-3 :-

→ The 1st Received digit is shifted out of the buffer and the same time the Syndrome register is shifted right once.

→ If the Received digit is in error, the detector o/p will be '1' which is used for Correcting the 1st Received digit.

→ The output of the detector is also fed into the Syndrome register to modify the syndrome.

Then Results in a new Syndrome. Corresponding to the altered received vector shifted to the Right by one place.

STEP-4 :-

→ The new Syndrome is now used to check whether or not the Second Received digit now at the Right-most stage of the buffer register is an erroneous digit.

→ If so it is Corrected a new Syndrome is Calculated as in Step 3 and the procedure is repeated.

STEP-5 :-

The decoder operates on the Received vector digit by digit until the Entire received vector is shifted out of the buffer.

* For a (7,4) cyclic code. The Received vector is 0100101 and the generator polynomial is $g(x) = 1+x+x^3$. Draw the Syndrome Calculation Circuit and Correct the Single error in the Received vector.

Sol:-

The generator polynomial is given by

$$g(x) = 1+x+x^3 = g_0 + g_1 x + g_2 x^2 + g_3 x^3.$$

∴ The coefficients are given by.

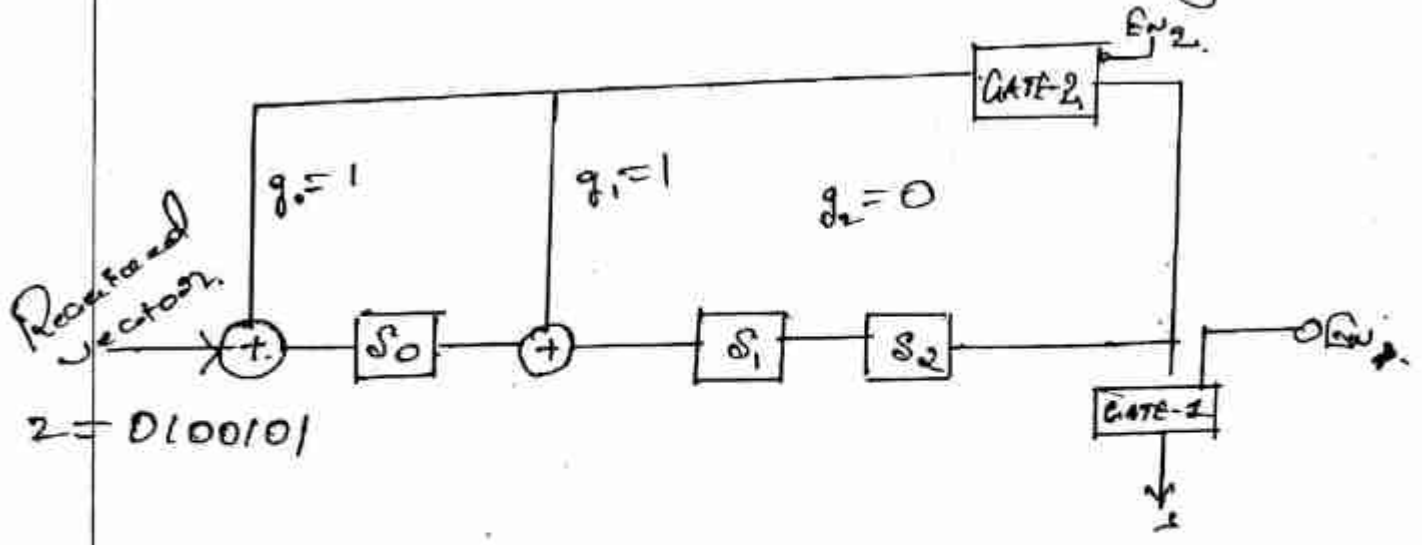
$g_0 = 1 \quad g_1 = 1 \quad g_2 = 0 \quad g_3 = 1$

To write the Syndrome Circuit

we have $(n-k) = (7-4) = 3$ flip flops

∴ $(n-k-1)$ to $S_{n-k-1} = S_{7-4-1} = S_2$ (i.e S_0 to S_2)

Syndrome bits



Z = 0100101

Fig: Syndrome Circuit

Number of Shifts.	Input $z(x)$	Shift Register S_0	S_1	S_2	Comments
Initialization. gate-1 off and gate-2 on.		0	0	0	Shift Register Content are Cleared.
1	1	1	0	0	
2	0	0	1	0	
3	1	1	0	1	
4	0	1	0	0	
5	0	0	1	0	
6	1	1	0	1	
7	0	1	0	0	← Indicates error
8	0	0	1	0	
9	0	0	0	1	
10	0	1	1	0	
11	0	0	1	1	
12	0	1	1	1	
13	0	1	0	1	
14.	0	1	0	0	← end of shift operation

Error vector. $E = 1000000$.

Corrected vector $V = Z + E$

$= 0100101 + 1000000$

$V = 1100101$ //

Verification:- $h(x) = \frac{x^7+1}{g(x)} = \frac{x^7+1}{1+x+x^3}$ //.

$x^3+x+1 \;)\; x^7+1 \;(\; x^4+x^2+x+1$

$x^7+x^5+x^4$

x^5+x^4+1

$x^5+x^3+x^2$

$x^4+x^3+x^2+1$

x^4+x^2+x

x^3+x+1

x^3+x+1

0 //.

$h(x) = x^4+x^2+x+1$ //

$h(x^{-1}) = \frac{1}{x^4}+\frac{1}{x^2}+\frac{1}{x}+1$

$x^4 h(x^{-1}) = 1+x^2+x^3+x^4$

$= 1011101$ //

$x^5 h(x^{-1}) = x+x^3+x^4+x^5$

$= 0101110$

$x^6 h(x^{-1}) = x^2+x^4+x^5+x^6$

$= 0010111$ //.

$$H = \begin{bmatrix} 1011100 \\ 0101110 \\ 0010111 \end{bmatrix}$$

$R_1 \leftarrow R_1 + R_3$

$$H = \begin{bmatrix} 1001011 \\ 0101110 \\ 0010111 \end{bmatrix} \quad \therefore \quad H^T = \begin{bmatrix} 100 \\ 010 \\ 001 \\ 110 \\ 011 \\ 111 \\ 101 \end{bmatrix}$$ //.

$[S] = 2H^T$

$= 0100101 \begin{bmatrix} 100 \\ 010 \\ 001 \\ 110 \\ 011 \\ 111 \\ 101 \end{bmatrix}$.

$[S] = 100$ // error which is same as before and hence verified.

IA 2012 [14M]

* The expurgated $(n, k-1)$ Hamming code is obtained from the original (n, k) Hamming code by discarding some of the code vectors. Let $g(x)$ denote the generator polynomial of the original Hamming code. The most common expurgated Hamming code is the one generated by

$$g_1(x) = (1+x)\, g(x)$$

where $(1+x)$ is a factor of $1+x^n$.

Consider the $(7, 4)$ Hamming code generated by

$$g(x) = 1+x^2+x^3$$

a) Construct the "8" code vectors for the expurgated $(7, 3)$ Hamming code, assuming a systematic format. Hence show that the minimum distance of the code is 4.

b) Determine the generator matrix G_1 and parity check matrix H_1 of the expurgated Hamming code.

c) Devise the encoder for the expurgated Hamming code and list the shift register contents in a tabular fashion for the message 011. Verify the code vector so obtained using $[v] = [D][G_1]$.

d) Devise the syndrome calculator for the expurgated Hamming code. Hence determine the syndrome for the received vector 0111110. Also correct the error, if any, in that received vector.

Sol:- a) The generator matrix is given by.

$g_1(x) = (1+x)\, g(x)$.

$= (1+x)(1+x^2+x^3)$

$= 1+x^2+x^3+x+x^3+x^4$

$g_1(x) = 1+x+x^2+x^4$ → code ex. 1110100.

$x g_1(x) = x+x^2+x^3+x^5$ → code is 0111010

$x^2 g_1(x) = x^2+x^3+x^4+x^6$ → code is 0011101

b) The generator matrix G_1 is given by.

$$[G_1] = \begin{bmatrix} 1 1 1 0 1 0 0 \\ 0 1 1 1 0 1 0 \\ 0 0 1 1 1 0 1 \end{bmatrix}$$

NOTE: Expurgated $(n, k-1)$ Hamming code → k equal to "k".

→ 3 Rows = k
Columns = n.

$R_3 \leftarrow R+R_2$

$$= \begin{bmatrix} 1 1 1 0 1 0 0 \\ 0 1 1 1 0 1 0 \\ 1 0 1 0 0 0 1 \end{bmatrix}$$

$[G_1] = [P \,!\, I_3] = [P \,!\, I_{k-1}]$ → equal to k

The parity-check matrix H_1 is given by

$H_1 = [I_{n-(k-1)} \,!\, P^T]$

Means equal to "k"

$H_1 = [I_4 \,!\, P^T]$ //

$\therefore P = \begin{bmatrix} 1 1 1 0 \\ 0 1 1 1 \\ 1 1 0 1 \end{bmatrix}$ // $\quad P^T = \begin{bmatrix} 1 0 1 \\ 1 1 1 \\ 1 1 0 \\ 0 1 1 \end{bmatrix}$ //

$$H_1 = \begin{bmatrix} 1000 & 101 \\ 0100 & 111 \\ 0010 & 110 \\ 0001 & 011 \end{bmatrix}$$

The Code-vectors are generated using,

$$[v] = [D][G_1]$$

$$= [d_0\ d_1\ d_2] \begin{bmatrix} 1110 & 100 \\ 0111 & 010 \\ 1101 & 001 \end{bmatrix}$$

$$= [(d_0+d_2), (d_0+d_1+d_2), (d_1+d_2), (d_2+d_3), d_0, d_1, d_2]$$

Message vector. [D]	Code vector.	Hamming weight (Hw)
000	0000000	0
001	1101001	4
010	0111010	4
011	1010011	4
100	1110100	4
101	0011101	4
110	1001110	4
111	0100111	4.

Table :- Code vector table for (9, 3) expurgated Hamming Code.

c) Encoder for Expurgated Hamming Code.

The generator matrix for (7,3) expurgated Hamming code is given by

$$g_1(x) = 1+x+x^2+x^4$$

The coefficients of generator polynomial are given by

$g_0=1$ $g_1=1$ $g_2=1$ $g_3=0$ $g_4=1$

We have $(n-k) = 4$ // Shift Registers. (i.e $R_0\ R_1\ R_2\ R_3$)

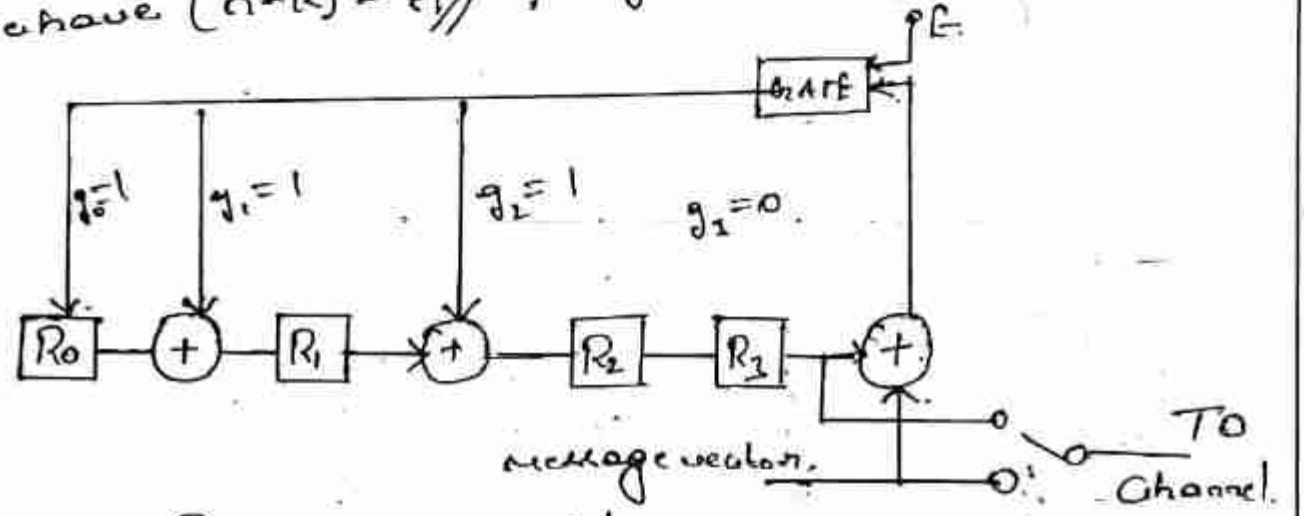


Fig:- Encoder Circuit

Number of Shifts.	Input D	Shift Register Contents R_0 R_1 R_2 R_3	Remainder R
Initialize gate is ON.		0 0 0 0	
	1	1 1 1 0	
	1	1 0 1 1	
	0	1 0 0 1	
gate is turned OFF		0 1 0 0	1
	X	0 0 1 0	0
	X	0 0 0 1	0
	X	0 0 0 0	1

Code vector $[v] = [R][D]$

$= 1010011$ //

d) The generator polynomial is given by

$g(x) = 1+x+x^2+x^4$

$g_0=1 \quad g_1=1 \quad g_2=1 \quad g_3=0 \quad g_4=1$

To write the Syndrome Circuit

We have $(n-k) = (7-3) = 4$ flipflops.

$\therefore (n-k-1)$ to $0 \Rightarrow S_{n-k-1} = S_{7-3-1} = S_3$ (i.e $S_0 - S_3$)

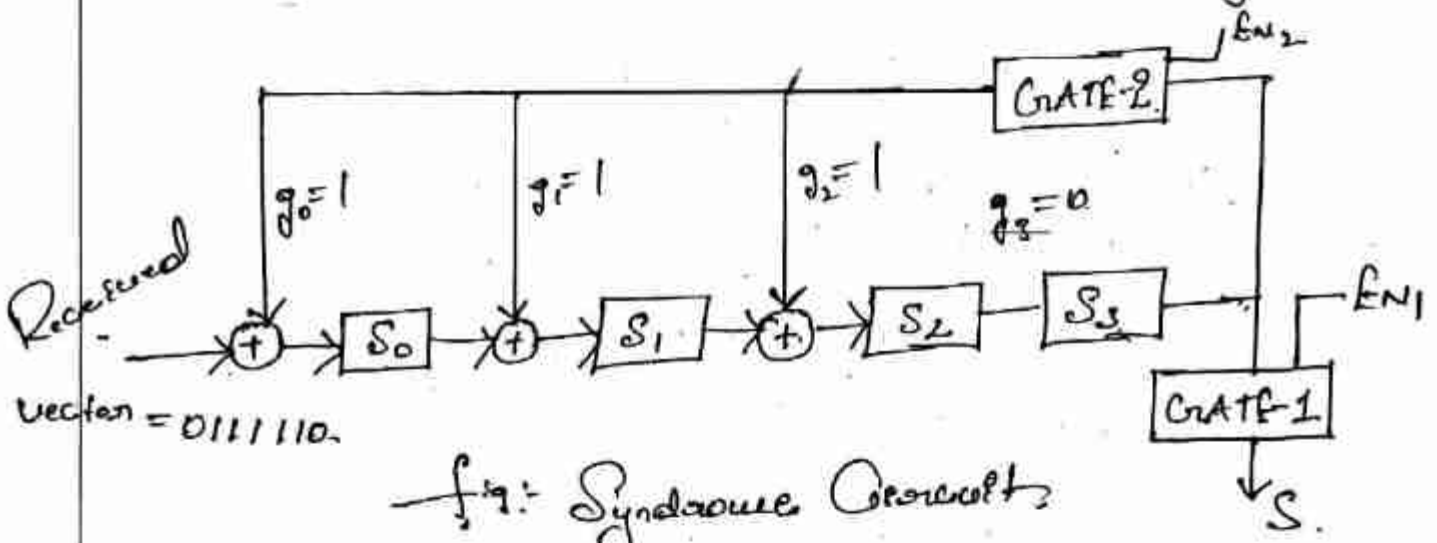


fig: Syndrome Circuit

number of shifts	Input z(x)	Shift Register Contents S_0 S_1 S_2 S_3	Comments
Initialization gate-1 OFF and gate-2 ON.		0 0 0 0	
1	0	0 0 0 0	
2	1	1 0 0 0	
3	1	1 1 0 0	
4	1	1 1 1 0	
5	1	1 1 1 1	
6	1	0 0 0 1	

7	0	1 1 1 0	← Indicates error
8	0	0 1 1 1	
9	0	1 1 0 1	
10	0	1 0 0 0	← end of shifting operation

Received vector $z \rightarrow 0\,1\,1\,1\;1\,1\,1\,0$

Error vector $e = 0\,0\,0\,0\,1\,0\,0$

Correct vector $v = z + e$

$= 01111110 + 00000100$

$= 0111010$ //

Syndrome verification

$$h(x) = \frac{x^{n+1}}{g(x)} = \frac{x^7+1}{g(x)} = \frac{x^7+1}{1+x+x^2+x^4}$$

$x^4+x^2+x+1 \;)\; x^7+1 \;(\; x^3+x+1$

$x^7+x^5+x^4+x^3$

$x^5+x^4+x^3+1$

$x^5+x^3+x^2+x$

x^4+x^2+x+1

x^4+x^2+x+1

$0\ 0\ 0\ 0$ //

$h(x) = x^3+x+1$.

$h(x^{-1}) = \frac{1}{x^3}+\frac{1}{x}+1$

$x^3 h(x^{-1}) = 1+x^2+x^3$ //

1011000 //

$x^4 h(x^{-1}) = x+x^3+x^4$

0101100 //

$x^5 h(x^{-1}) = x^2+x^4+x^5$

0010110 //

$x^6 h(x^{-1}) = x^3+x^5+x^6$

$= 0001011$ //

$$H = \begin{vmatrix} 1011000 \\ 0101100 \\ 0010110 \\ 0001011 \end{vmatrix}$$

$R_1 \leftarrow R_1+R_3+R_4$

$$= \begin{vmatrix} 1000101 \\ 0101100 \\ 0010110 \\ 0001011 \end{vmatrix}$$

$R_2 \leftarrow R_2+R_4$

$$H = \begin{vmatrix} 1000101 \\ 0100111 \\ 0010110 \\ 0001011 \end{vmatrix}$$

$$H^T = \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1110 \\ 0111 \\ 1101 \end{bmatrix}$$

$$S = [z][H^T]$$

$$= 0\,1\,1\,1\,1\,1\,0 \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \\ 1110 \\ 0111 \\ 1101 \end{bmatrix} = [1\,1\,1\,0] //$$

Error vector $= 0\,0\,0\,0\,1\,0\,0$

Correct $v = [z] + E$

Code vector $= 0\,1\,1\,1\,1\,1\,0 + 0\,0\,0\,0\,1\,0\,0$

$v = 0\,1\,1\,1\,0\,1\,0$ //

* For a (7,4) Cyclic Code. The Received vector, Z(x) is 1110101 and the generator polynomial is $g(x) = 1+x+x^3$. Draw the Syndrome Calculation Circuit and Correct the Single error in the Received vector.

DEC-2012 [10M]

Sol:- The generator polynomial is given by.

$$g(x) = 1+x+x^3 = g_0 + g_1 x + g_2 x^2 + g_3 x^3.$$

∴ The Coefficient are given by.

$g_0 = 1$, $g_1 = 1$, $g_2 = 0$, and $g_3 = 1$.

To write the Syndrome Circuit:

We have $(n-k) = (7-4) = 3$ flipflops.

∴ $(n-k-1)$ (or) $S_{n-k-1} = S_{7-4-1} = S_2$ (i.e S_0, S_1, S_2)

↓ Syndrome bit

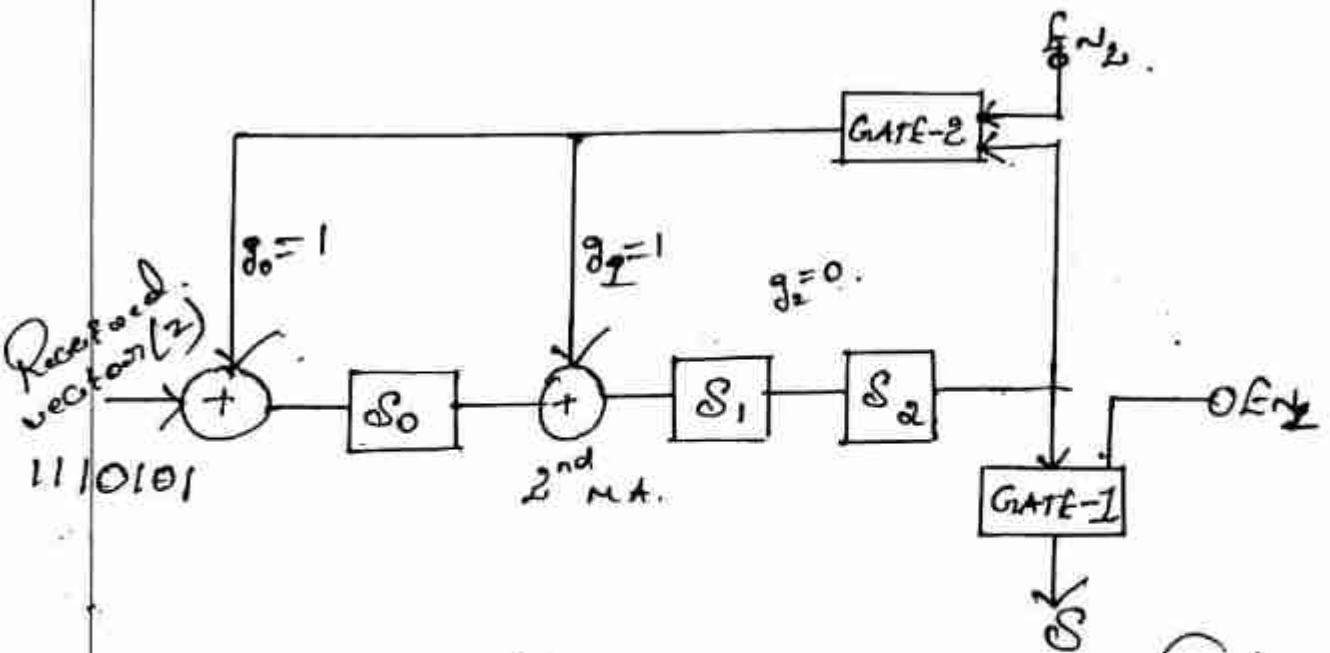


Fig:- Syndrome Calculation Circuit

METHOD -I :

number of shifts.	Input z.	Shift Register Content S_0	S_1	S_2	Comments
Initialization gate-1 OFF and gate-2 ON.		0	0	0	Shift Register Contents are Cleared.
1	1	1	0	0	
2	0	0	1	0	
3	1	1	0	1	
4	0	1	0	0	
5	1	1	1	0	
6	1	1	1	1	
7	1	0	0	1	← Indicates error.

Knowing the Syndrome $S_0 S_1 S_2 \rightarrow 001$ and H^T Matrix, the error can be corrected analytically as shown below.

∴ The H^T matrix is written as.

$$H^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

The Syndrome $S_0 S_1 S_2 \rightarrow 001$ is present in 3rd row.

∴ The Received vector $Z = 1110101$.

∴ The Error vector 'E' = 0010000 //

The corrected vector = $Z + E = 1110101 + 0010000 = 1100101$ //

METHOD-II :-

number of Shifts.	Input (2)	Shift Register Contents. S_0	S_1	S_2	Comments.
Initialization.		0	0	0	
1	1	1	0	0	
2	0	0	1	0	
3	1	1	0	1	
4	0	1	0	0	
5	1	1	1	0	
6	1	1	1	1	
7	1	0	0	1	← Indicates Error
8	0	1	1	0	
9	0	0	1	1	
10	0	1	1	1	
11	0	1	0	1	
12	0	1	0	0	← End of shifting operation

Syndrome. $S_0 S_1 S_2 \rightarrow 001$ //

$Z = 1110101$ and $E = 0010000$

$V = Z + E$

$= 1110101 + 0010000$

$= 1100101$

//

Verification: $h(x) = \frac{x^7+1}{g(x)} = \frac{x^7+1}{g(x)} = \frac{x^7+1}{1+x+x^3}$

$x^3+x+1 \;)\; x^7+1 \;(x^4+x^2+x+1$ ← h(x)

$x^7+x^5+x^4$

x^5+x^4+1

$x^5+x^3+x^2$

$x^4+x^3+x^2+1$

x^4+x^2+x

x^3+x+1

x^3+x+1

0 //

$h(x) = x^4+x^2+x+1$

$Y_{h(x)} = Y_{x^4} + Y_{x^2} + Y_x + 1$

$x^4 h(x^{-1}) = 1 + x^2 + x^3 + x^4$

$10111 0.$

$x^5 h(x^{-1}) = x + x^3 + x^4 + x^5$

$= 0101110$

$x^6 h(x^{-1}) = x^2 + x^4 + x^5 + x^6$

$= 0010111$

$$H = \begin{bmatrix} 1011100 \\ 0101110 \\ 0010111 \end{bmatrix}$$

$R_1 \leftarrow R_1 + R_3$

$$H = \begin{bmatrix} 1001011 \\ 0101110 \\ 0010111 \end{bmatrix}$$

$$H^T = \begin{bmatrix} 100 \\ 010 \\ 001 \\ 110 \\ 011 \\ 111 \\ 101 \end{bmatrix}$$

$S = ZH^T$

$= 1110101 \begin{bmatrix} 100 \\ 010 \\ 001 \\ 110 \\ 011 \\ 111 \\ 101 \end{bmatrix}$

$= 001$ (error) which is same as before and hence verified.

* A (15, 5) Linear Cyclic Code has a generator polynomial. $g(x) = 1+x+x^2+x^4+x^5+x^8+x^{10}$

a) Draw the block diagram of an Encoder and Syndrome Calculator for this Code.

b) Find the Code polynomial for the message polynomial $D(x) = 1+x^2+x^4$ in Systematic form.

c) Is $v(x) = 1+x^4+x^6+x^8+x^{14}$ a code polynomial?

Sol:

(a) The generator polynomial is given by

$$g(x) = 1+x+x^2+x^4+x^5+x^8+x^{10}$$

The coefficient are then given by

$g_0=1$ $g_1=1$ $g_2=1$ $g_3=0$ $g_4=1$ $g_5=1$ $g_6=0$ $g_7=0$ $g_8=1$ $g_9=0$.

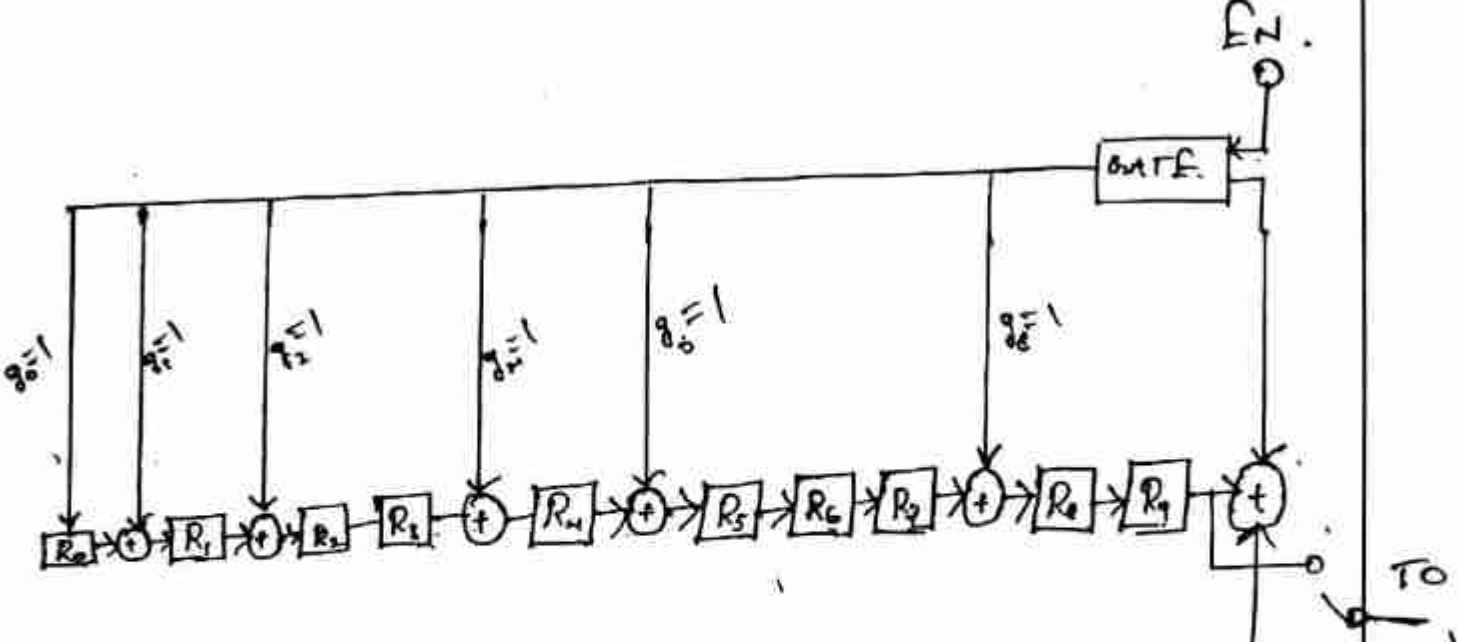


Fig: Encoder for (15, 5) Cyclic Code.

b) $D(x) = 1+x^2+x^4 \rightarrow 10101$ //

Number of Shifts	Input D	Shift Register Contents R_0 R_1 R_2 R_3 R_4 R_5 R_6 R_7 R_8 R_9	Remainder bits - Q
Initialization		0 0 0 0 0 0 0 0 0 0	—
1	1	1 1 1 0 1 1 0 0 1 0	—
2	0	0 1 1 1 0 1 1 0 0 1	—
3	1	0 0 1 1 1 0 1 1 0 0	—
4	0	0 0 0 1 1 1 0 1 1 0	—
5	1	1 1 1 0 0 0 1 0 0 1	—

Code vector $[v] = R^T D$

$= 1110001000\ 10101$ //

* The generator polynomial is given by
$g(x) = 1+x+x^2+x^4+x^5+x^8+x^{10}$
* The coefficient are given by.
$g_0=1\ g_1=1\ g_2=1\ g_3=0\ g_4=1\ g_5=1\ g_6=0\ g_7=0\ g_8=1\ g_9=0$

To realize the syndrome Circuit
We have $(n-k) = (15-5) = 10$ flip flops
$\therefore (n-k)$ (for) $S_{n-k-1} = S_{15-5-1} = S_9$ (i.e S_0 to S_9)

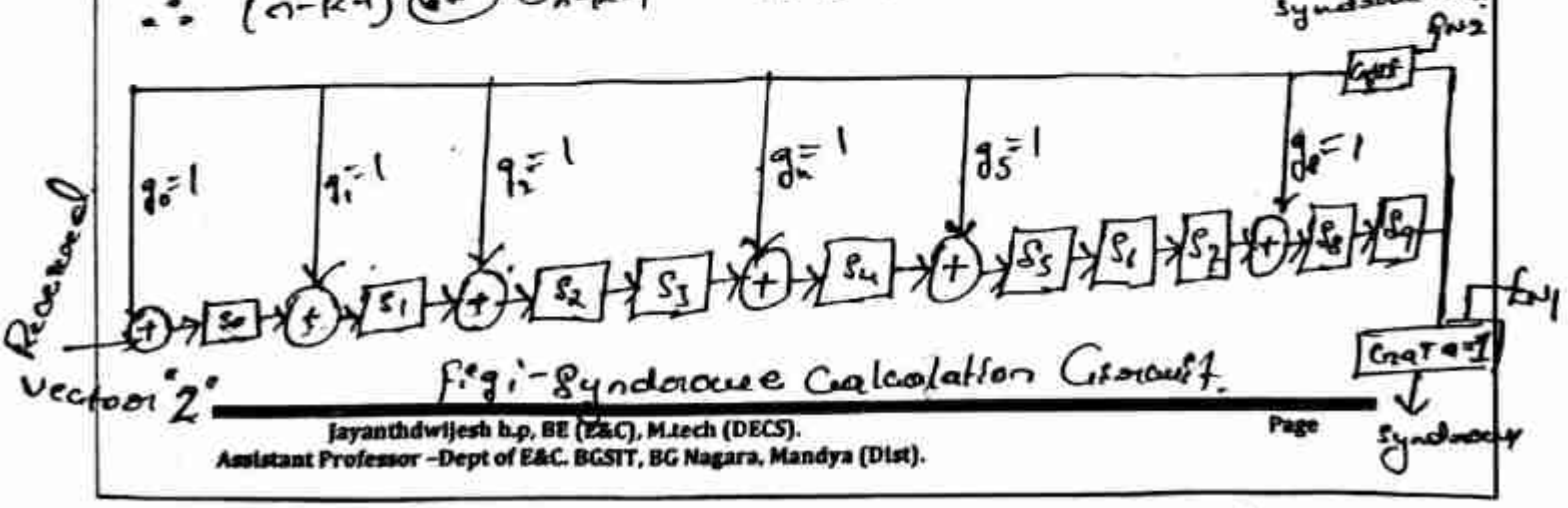


Fig:- Syndrome Calculation Circuit.

c) $V(x) = 1+x^4+x^6+x^8+x^{14}$ a code polynomial?

$g(x) \overline{)V(x)}$

$x^{10}+x^8+x^5+x^4+x^3+x+1 \;\overline{)\; x^{14}+x^8+x^6+x^4+1 \;}\; x^4+x^2$

$x^{14}+x^{12}+x^9+x^8+x^6+x^5+1$

$x^{12}+x^9+x^5+1$

$x^{12}+x^{10}+x^7+x^6+x^4+x^3+x^2$

$x^{10}+x^9+x^7+x^6+x^5+x^4+x^3+x^2$

$x^{10}+x^8+x^5+x^4+x^2+x+1$

$x^9+x^8+x^7+x^6+x^3+x+1.$

$R(x) \longrightarrow$

The Remainder polynomial $R(x)$ is a non-zero polynomial. Hence the given polynomial is not a code polynomial.

NOTE

If Remainder $R(x)=0$ ↑ Code polynomial

If $R(x)$ is not zero ↑ not a Code polynomial (non-zero polynomial)

* The generator polynomial for a (15, 9) cyclic code is $g(x) = 1+x^4+x^6+x^7+x^8$.

1) Find the code vector in systematic form for the message $D(x) = x^2+x^3+x^4$.

2) Assume that the first and last bits of the code vector $V(x)$ for $D(x) = x^2+x^3+x^4$ suffer transmission errors. Find the syndrome of $V(x)$.

Sol:-

1) $D(x) = x^2+x^3+x^4 = 0011100$ //

$g(x) = 1+x^4+x^6+x^7+x^8$.

$$R(x) = \frac{x^{n-k} D(x)}{g(x)} = \frac{x^8(x^2+x^3+x^4)}{g(x)} = \frac{x^{10}+x^{11}+x^{12}}{1+x^4+x^6+x^7+x^8}$$

$x^8+x^7+x^6+x^4+1 \;)\; x^{12}+x^{11}+x^{10} \;(\; x^4+1$

$x^{12}+x^{11}+x^{10}+x^8+x^4$

x^8+x^4

$x^8+x^7+x^6+x^4+1$

x^7+x^6+1

$R(x) = 1+x^6+x^7$

$= 1+(0)(x)+(0)x^2+(0)x^3+(0)x^4+(0)(x^5)+(1)(x^6)+(1)x^7$

$= [10000011]$ //.

The Codeword vector $[V]$ is given by

$[V] = [R]\,[D]$

$= 10000011\;0011100$ //

2) When the first and Last bit of the code-vector 'V' are in errors, Then the Received Code-vector is given by

$$[V] = [R][D]$$

$$= [1000011\ 0011100]$$

(Last (1) Rever.)

$$\therefore [Z] = [0000011\ 0011101]$$

(First (0) Rever.)

$$Z(X) = X^6 + X^7 + X^{10} + X^{11} + X^{12} + X^{14}$$

The Syndrome $S(X)$ of the Received vector $Z(X)$ is obtained as the Remainder resulting from division of $Z(X)$ by $g(X)$ as given below

$$g(X) \overline{)\ Z(X)}$$

$$X^8 + X^7 + X^6 + X^4 + 1 \overline{)\ X^{14} + X^{12} + X^{11} + X^{10} + X^7 + X^6}\ (X^6$$

$$X^{14} + X^{13} + X^{12} + X^{10} + X^6$$

$$X^{13} + X^{11} + X^7$$

$$X^{13} + X^{12} + X^{11} + X^9 + X^5$$

$$X^{12} + X^9 + X^7 + X^5$$

$$X^{12} + X^{11} + X^{10} + X^8 + X^4$$

$$X^{11} + X^{10} + X^9 + X^8 + X^7 + X^5 + X^4$$

$$X^{11} + X^{10} + X^9 + X^7 + X^3$$

$$X^8 + X^5 + X^4 + X^3$$

$$X^8 + X^5 + X^4 + 1$$

$$X^7 + X^6 + X^5 + X^3 + 1$$

$\therefore$ Syndrome $S(X) = 1 + X^3 + X^5 + X^6 + X^7$.

$\therefore$ Syndrome vector $[S]$ which is a $(n-k) = (15-7) = 8$ bit vector is given by

$$[S] = 10010111$$

* Given $n \leq 7$. Identify (n, k) values of the Cyclic Codes generated by the following generator polynomials.

1) $g(x) = 1 + x^2 + x^3$ 2) $g(x) = 1 + x + x^2 + x^4$ 3) $g(x) = 1 + x^2 + x^3 + x^4$

July/Aug – 2003 [5M]

Sol:

From property – 1 of Cyclic codes we observe that the degree of generator polynomial of a Cyclic Code is given by $(n-k)$.

NOTE: $g(x) = 1 + x + x^3$

1) $(n-k) = 3$ // for $n = 7$, we have $k = 4$. and we get $(7, 4)$ Cyclic code.

NOTE: For $n = 6$ it is not possible to generate $(6, 3)$ cyclic code, since $x^6 + 1$ is not divisible by the given polynomial $g(x)$. Similar argument holds good for $n < 6$.

NOTE: $g(x) = 1 + x + x^2 + x^4$

2) Given $(n-k) = 4$ // for $n = 7$ we have $k = 3$ and we get $(7, 3)$ Cyclic Code // [This is because $x^7 + 1$ is divisible by $g(x) = 1 + x + x^2 + x^4$] //

3) Given $(n-k) = 4$ // for $n = 7$ we have $k = 3$. But the given polynomial is not a factor of $x^n + 1 = x^7 + 1$ as shown below.

$$x^4+x^3+x^2+x+1 \;\big)\; x^7+1 \;\big(\; x^3+x^2$$

$$x^7+x^6+x^5+x^4+x^3$$

$$x^6+x^5+x^4+x^3+1$$

$$x^6+x^5+x^4+x^3+x^2$$

$$x^2+1 \longleftarrow \text{Remainder}$$

Since the Remainder is not zero, the given polynomial $g(x)$ is not a factor of x^7+1 and it cannot generate any cyclic code. //

* The generator polynomial of a (15,7) cyclic code is given by

a) Draw the Syndrome Calculation Circuit

b) Find the Syndrome of the Received polynomial

$Z(X) = 1+X+X^3+X^6+X^8+X^9+X^{11}+X^{14}$

by testing the States of the Registers used in Syndrome Calculation Circuit

c) Verify the Syndromes obtained in (b) by using direct hand Calculation.

Sol :- $g(X) = 1+X^4+X^6+X^7+X^8$

$g_0=1$ $g_1=0$ $g_2=0$ $g_3=0$ $g_4=1$ $g_5=0$ $g_6=1$ $g_7=1$ and $g_8=1$.

To locate the Syndrome Calculation Circuit we have $(n-k) = 15-7 = 8$ flip flops.

$\therefore f_{(n-k-1)} = f_{(15-7-1)} = S_7$ i.e $(S_0 \cdots S_7)$. //

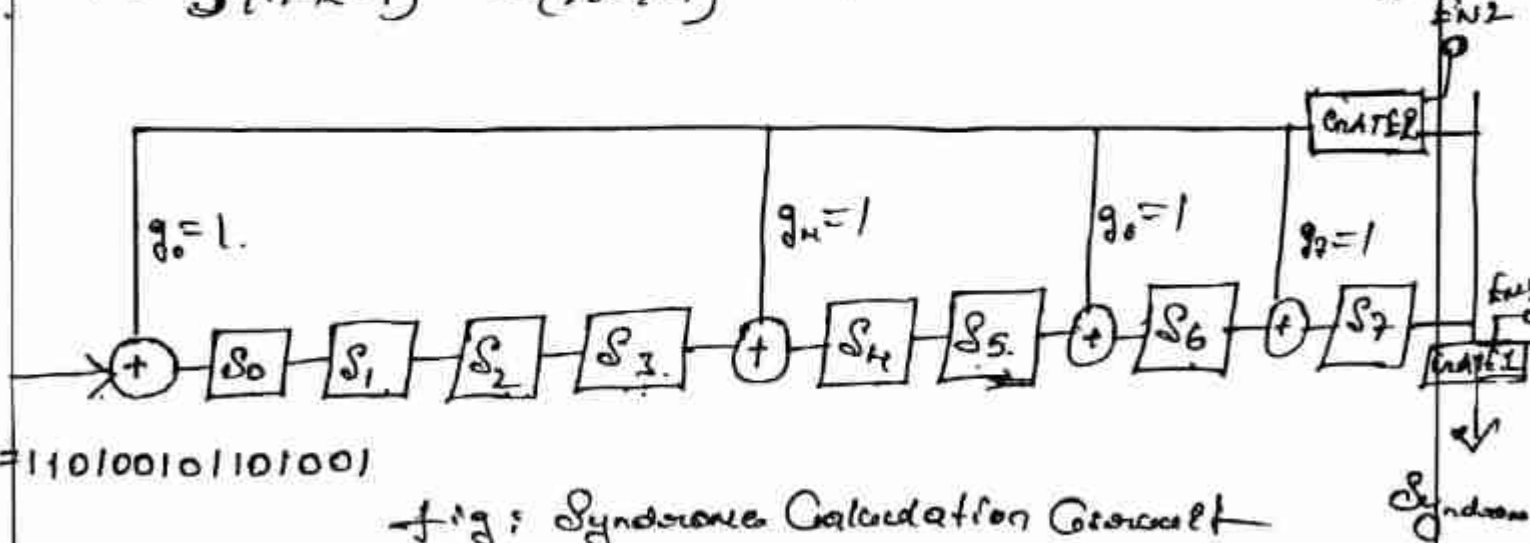


fig: Syndrome Calculation Circuit

b)

no of Shifts	Input Z.	Shift Register Content							
		S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Initialization		0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0
4	1	1	0	0	1	0	0	0	0
5	0	0	1	0	0	1	0	0	0
6	1	1	0	1	0	0	1	0	0
7	1	1	1	0	1	0	0	1	0
8	0	0	1	1	0	1	0	0	1
9	1	0	0	1	1	0	1	0	1
10	0	1	0	0	1	1	0	1	0
11	0	0	1	0	0	1	1	0	0
12	1	1	0	1	0	0	1	1	0
13	0	1	1	0	1	0	0	1	1
14	1	0	1	1	0	1	0	0	1
15.	1	0	0	1	1	0	1	0	0

The Syndrome for the Received vector $Z(x)$ is $S \rightarrow 00111000$

$S(x) = x^2 + x^3 + x^4$ is the Syndrome polynomial //

P.T.O.

c) Consider $Z(x)$ being divided by $g(x)$ as given below

$$x^8+x^7+x^6+x^4+1 \;\big)\; x^{14}+x^{11}+x^9+x^8+x^6+x^3+x+1 \;\big($$

$$x^{14}+x^{13}+x^{12}+x^{10}+x^6$$

$$x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^8+x^3+x+1$$

$$x^{13}+x^{12}+x^{11}+x^9+x^5$$

$$x^{10}+x^8+x^5+x^3+x+1.$$

$$x^{10}+x^9+x^8+x^6+x^2$$

$$x^9+x^6+x^5+x^3+x^2+x+1$$

$$x^9+x^8+x^7+x^5+x$$

$$x^8+x^7+x^6+x^3+x^2+x+1$$

$$x^8+x^7+x^6+x^4+1$$

$$x^4+x^3+x^2$$

$S(x) = x^2+x^3+x^4$, which is same as before.

P.T.O.

* Consider the (15,11) Cyclic Code generated by $g(x) = 1+X+X^4$

(a) Draw the Syndrome calculation Circuit for the code

(b) Find the Syndrome for the Received vector $Z(X) = 000111100100001$ by using the State of the Register based on the Syndrome calculation Circuit with each Input

(c) verify the Syndrome obtained in (b) by using direct method.

Sol:-

$g(x) = 1+X+X^4$

$g_0 = 1 \quad g_1 = 1 \quad g_2 = 0 \quad g_3 = 0 \quad g_4 = 1$

To locate the Syndrome Calculation Circuit we have $(n-k) = 15-11 = 4$ flip flops.

$\therefore S_{(n-k-1)} = S_{(15-11-1)} = S_3$ i.e $(S_0 \text{ ---- } S_3)$

(a)

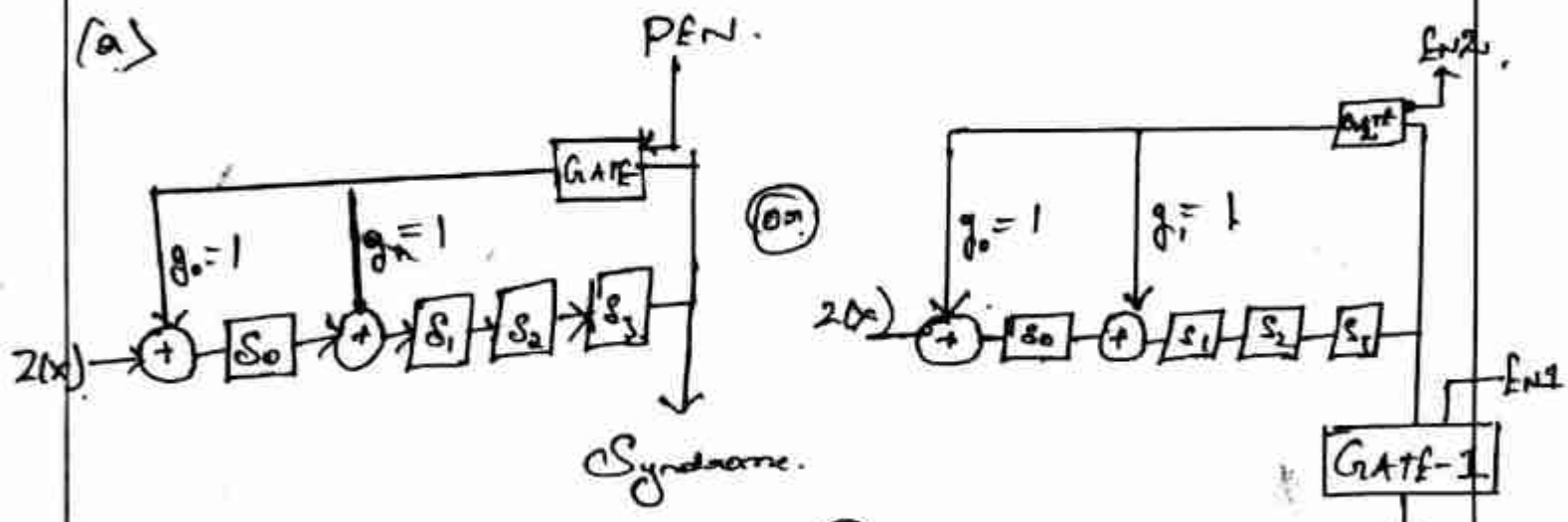


Fig: Syndrome Circuit

Number of Shifts.		Shift Register Contents.			
		S_0	S_1	S_2	S_3
Initialization		0	0	0	0
1	1	1	0	0	0
2	0	0	1	0	0
3	0	0	0	1	0
4	0	0	0	0	1
5	0	1	1	0	0
6	1	1	1	1	0
7	0	0	1	1	1
8	0	1	1	1	1
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	1	1
13	0	1	0	0	1
14	0	1	0	0	0
15	0	0	1	0	0

The Syndrome $S = 0100$.

$\therefore$ Syndrome polynomial is $S(X) = X$ //

L.p.t.o.

c) verification using division method.

Given $z \rightarrow 000111100100001$

$z(x) = x^3 + x^4 + x^5 + x^6 + x^9 + x^{14}$

x^4+x+1) $x^{14} + x^9 + x^6 + x^5 + x^4 + x^3$

$x^{14} + x^{11} + x^{10}$

$x^{11} + x^{10} + x^9 + x^6 + x^5 + x^4 + x^3$

$x^{11} + x^8 + x^7$

$x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3$

$x^{10} + x^7 + x^6$

$x^9 + x^8 + x^5 + x^4 + x^3$

$x^9 + x^6 + x^5$

$x^8 + x^6 + x^4 + x^3$

$x^8 + x^5 + x^4$

$x^6 + x^5 + x^3$

$x^6 + x^3 + x^2$

$x^5 + x^2$

$x^5 + x^2 + x$

$x \;//\; \longrightarrow s(x)$

∴ Syndrome polynomial $(s(x)) = x$ which is same as before and hence verified.

Verilog HDL 18EC55

Module 3b

Verilog constructs to Gates

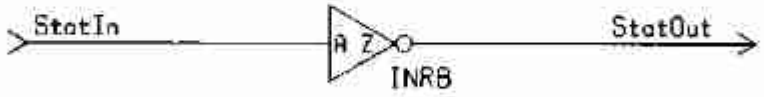
Continuous Assignment statement

A continuous assignment statement represents, in hardware, logic that is derived from the expression on the right-hand-side of the assignment statement driving the net that appears on the left-hand-side of the assignment. The target of a continuous assignment is always a net driven by combinational logic.

```
module Continuous (StatIn, StatOut);
  input StatIn;
  output StatOut;

  assign StatOut = ~ StatIn; // Continuous assignment.
endmodule
```

```
// Synthesized netlist is shown in Figure 2-1.
```



The continuous assignment statement describes an inverter that has its input connected to *StatIn* and whose output is *StatOut*. Delays, if any, specified in a continuous assignment statement are usually ignored by a synthesis system. For example, in the continuous assignment:

```
assign #2 EffectiveAB = DriverA | DriverB;
```

the delay #2 is ignored for synthesis.

Procedural Assignment statement

A procedural assignment statement represents, in hardware, logic that is derived from the expression on the right-hand-side of the assignment statement driving the variable that appears on the left-hand-side of the assignment. Note that procedural assignments can appear only within an always statement[1].

There are two kinds of procedural assignment statements:

i. Blocking

ii. Non-blocking

Blocking Procedural assignment

```
module Blocking (Preset, Count);
  input [0:2] Preset;
  output [3:0] Count;
  reg [3:0] Count;

  always @ (Preset)
    Count = Preset + 1;
    // Blocking procedural assignment.
endmodule
// Synthesized netlist is shown in Figure 2-2.
```

The blocking procedural assignment statement describes an adder that takes *Preset* and the integer 1 as inputs and places the result in the variable *Count*.

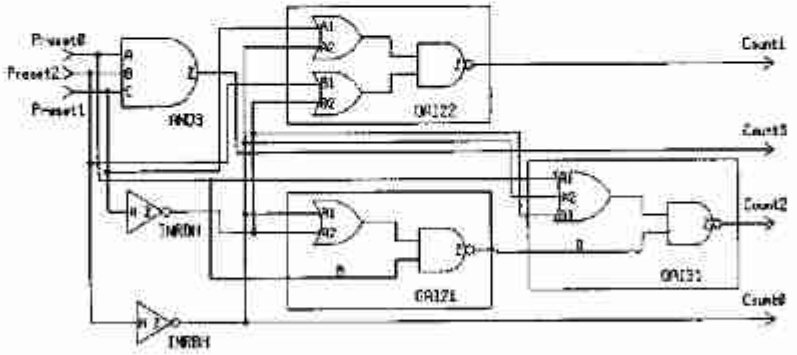


Figure 2-2 Combinational circuit from blocking procedural assignment.

Non-blocking procedural assignment

```
module NonBlocking (RegA, Mask, RegB);
  input [3:0] RegA, Mask;
  output [3:0] RegB;
  reg [3:0] RegB;

  always @ (RegA or Mask)
    RegB <= RegA & Mask;
    // Non-blocking procedural assignment.
endmodule
```

The blocking or non-blocking nature of an assignment does not cause any change to the combinational logic generated from the assignment statement itself, but affects the use of the resultant value later on.

A good recommendation to follow is to use blocking assignments for modeling combinational logic and to use non-blocking assignments for modeling sequential logic; more on this is described in later sections.


```
// Synthesized netlist is shown in Figure 2-3.
```

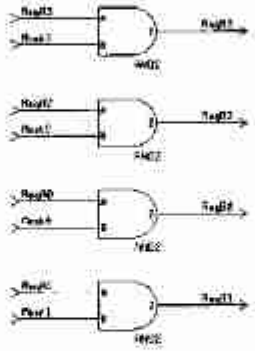


Figure 2-3 Combinational logic from non-blocking procedural assignment.

Blocking vs. Non-blocking assignments

There is a difference between how non-blocking and blocking assignments are treated for synthesis because of the language semantic difference. In a blocking assignment, the assignment to the left-hand-side target completes before the next statement in the sequential block is executed (there is no difference between blocking and non-blocking if only one statement is present in the always statement). In a non-blocking assignment, the assignment to the left-hand-side target is scheduled for the end of the simulation cycle (assignment does not occur immediately) before the next statement is executed. Let us look at an example.


```
module FlagBits (ClockB, Strobe, Xflag, Mask,
                 RightShift, SelectFirst, CheckStop);
  input ClockB, Strobe, Xflag, Mask;
  output RightShift, SelectFirst, CheckStop;
  reg RightShift, SelectFirst, CheckStop;

  always @ (negedge ClockB)
  begin
    RightShift = RightShift & Strobe;  // Blocking
    SelectFirst <= RightShift | Xflag; // Non-blocking
    CheckStop <= SelectFirst ^ Mask;   // Non-blocking
  end
endmodule
```

The always statement has a sequential block with three assignment statements, the first one is a blocking procedural assignment and the next two are non-blocking assignments. Since all these assignments occur under the control of a clock edge, falling-edge-triggered flip-flops are synthe-

Blocking vs. Non-blocking assignments

sized for *RightShift*, *SelectFirst* and *CheckStop*. However, the difference is in the way the information is connected to the data input of these flip-flops. Because the assignment to *RightShift* is a blocking assignment, the new value of *RightShift* is available for use in the second assignment. This implies that the data input of the *RightShift* flip-flop should be used to gate into the data input of the *SelectFirst* flip-flop. Because *SelectFirst* is a non-blocking assignment, the use of *SelectFirst* in the third assignment refers to the old value of *SelectFirst*, not to the value being assigned in the second statement. Consequently, it is the flip-flop output of *SelectFirst* (old value) that feeds the data input of the *CheckStop* flip-flop. The difference explained can also be confirmed in the synthesized netlist shown in Figure 2-56.


```
// Synthesized netlist is shown in Figure 2-56.
```

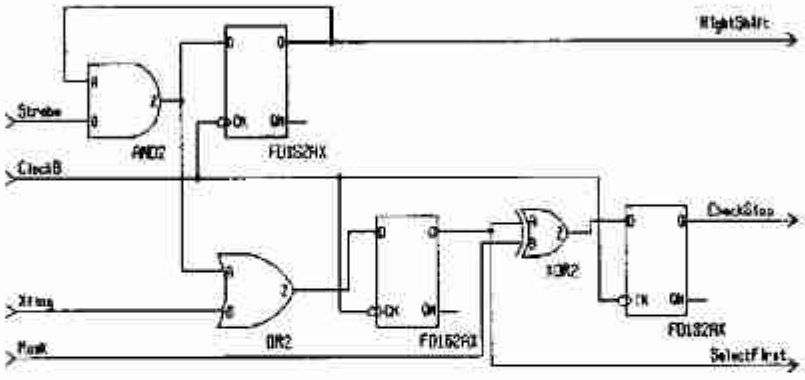


Figure 2-56 Non-blocking vs blocking procedural assignment.

```
module NonBlockingExample (ClockZ, Merge, ER, Xmit,
                           FDDI, Claim);
  input ClockZ, Merge, ER, Xmit, FDDI;
  output Claim;
  reg Claim;
  reg FCR;

  always @ (posedge ClockZ)
  begin
    FCR <= ER | Xmit;            // Assignment 1.

      if (Merge)
        Claim <= FCR & FDDI;     // Assignment 2.
      else
        Claim <= FDDI;
    end
  endmodule
```



```
// Synthesized netlist is shown in Figure 2-57.
```

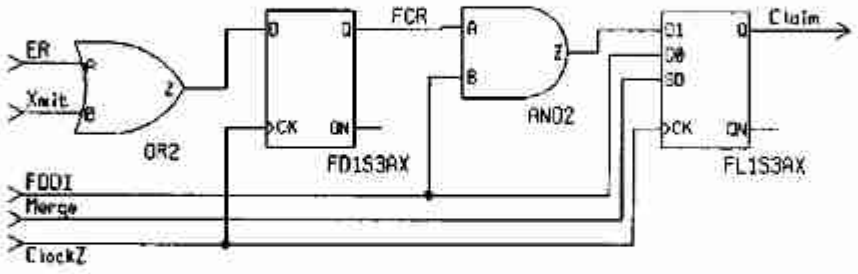


Figure 2-57 Non-blocking assignments.

There are three assignment statements in this example. Statements within the sequential block execute sequentially. However, the target of a non-blocking assignment is always assigned a value in the future (at the end of the current simulation time). Therefore in keeping with the language semantics of a non-blocking assignment, the use of *FCR* in assignment 2 is the old value of *FCR* and not the value assigned in assignment 1. Thus in the synthesized netlist, the output of the flip-flop for *FCR* is used to feed into the logic for *Claim*.


```
module BlockingExample (ClockZ, Merge, ER, Xmit, FDDI,
                        Claim);
  input ClockZ, Merge, ER, Xmit, FDDI;
  output Claim;
  reg Claim;

  reg FCR;

  always @ (posedge ClockZ)
  begin
    FCR = ER | Xmit;            // Assignment 1.
    if (Merge)
      Claim = FCR & FDDI;       // Assignment 2.
    else
      Claim = FDDI;
  end
endmodule
```

```
// Synthesized netlist is shown in Figure 2-58.
```

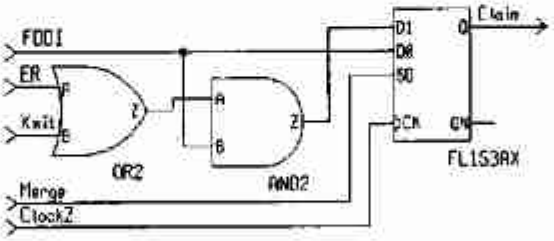


Figure 2-58 Blocking assignments.

In this case, the assignment to *FCR* must complete first before assignment 2 is done. Thus to mimic the language semantics, the right-hand-side expression of assignment 1 must be used to form the logic for the data for *Claim*. This is shown in the synthesized netlist. In addition, no flip-flop is inferred for *FCR* since its value is assigned and then used; the value of *FCR* does not have to be saved between different iterations of the always statement.

Chapter 5 explains the rationale behind the recommendation for using non-blocking assignments for sequential logic and for using blocking assignments for combinational logic; this is to prevent any possibility of a functional mismatch between the Verilog HDL model and its synthesized netlist. Note that this is only a recommendation; in many cases, it may be perfectly okay to use either of the two assignments, as long as the semantic differences are understood.

Target of Assignment

The target of a procedural assignment is synthesized into a wire, a flip-flop, or a latch, depending on the context under which the assignment appears in the Verilog HDL model. For example, if the previously described non-blocking procedural assignment statement appeared, say, under the control of a clock as shown in the following example, then the target is synthesized as a flip-flop.

```
module Target (Clk, RegA, RegB, Mask);
  input Clk;
  input [3:0] RegA, Mask;
  output [3:0] RegB;
  reg [3:0] RegB;

  always @ (posedge Clk)
    RegB <= RegA & Mask;
endmodule
```

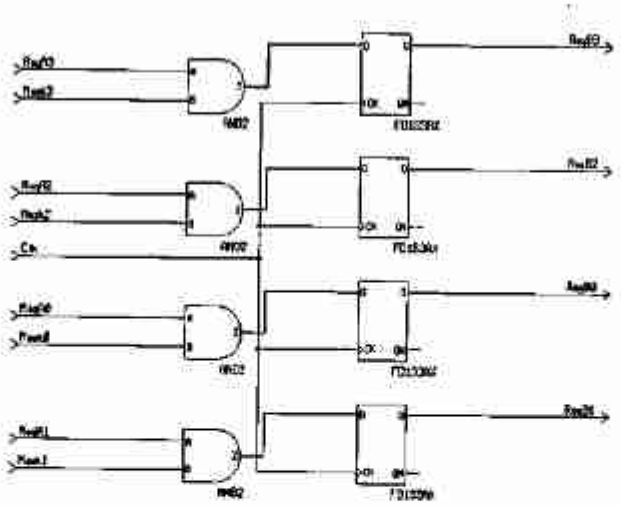



Figure 2-4 Target of an assignment is a flip-flop.

Assignment Restrictions

Any kind of delay, delay control or intra-statement delay, specified in a procedural assignment (blocking or non-blocking) is ignored by a synthesis system. This can potentially lead to a functional mismatch between the design model and its synthesized netlist.

```
#5 RegB <= RegA & Mask;
  // Delay control #5 is ignored.
RegB = #2 RegA & Mask;
  // Intra-statement delay #2 is ignored.
```


There is another restriction on using both blocking and non-blocking assignments in a single model for synthesis.

```
A target cannot be assigned using a blocking assignment
and a non-blocking assignment.
```

What this means is that if a target is assigned using a blocking (or a non-blocking) assignment, then the same target can only be assigned again using a blocking (or a non-blocking) assignment. Here is an example.

```
Count = Preset + 1;
. . .
Count <= Mask; // This is illegal since Count is
      // previously assigned using a blocking assignment.
```

Logical operators

The logical operators get directly mapped onto primitive logic gates in hardware. Here is a model of a full-adder using continuous assignment statements.

```
module FullAdder (A, B, CarryIn, Sum, CarryOut);
  input A, B, CarryIn;
  output Sum, CarryOut;

  assign Sum = (A ^ B) ^ CarryIn;
  assign CarryOut = (A & B) | (B & CarryIn) |
                    (A & CarryIn);
endmodule
```

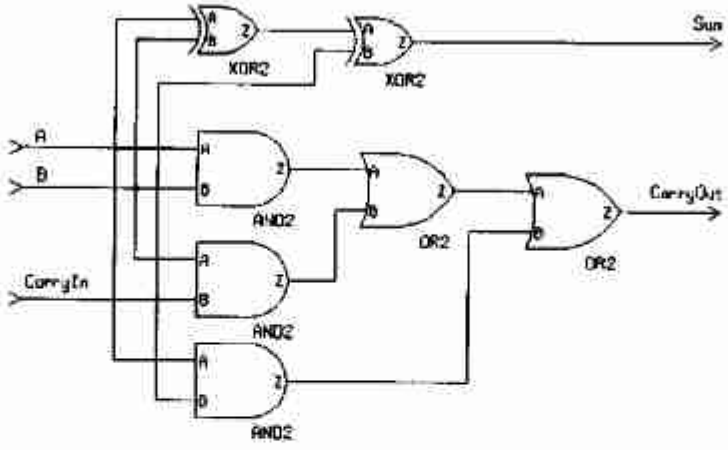



Figure 2-5 Logical operators map to primitive logic gates.

Arithmetic Operators

In Verilog HDL, a reg type is interpreted as an unsigned number and an integer type is interpreted as a signed number in 2's complement form with the rightmost bit as the least significant bit. Thus, to synthesize an unsigned arithmetic operator, the reg type is used. To get a signed arithmetic operator, the integer type is used.

The net type is interpreted as unsigned numbers.

```
module UnsignedAdder (Arb, Bet, Lot);
  input [2:0] Arb, Bet;
  output [2:0] Lot;

  assign Lot = Arb + Bet;
endmodule
```

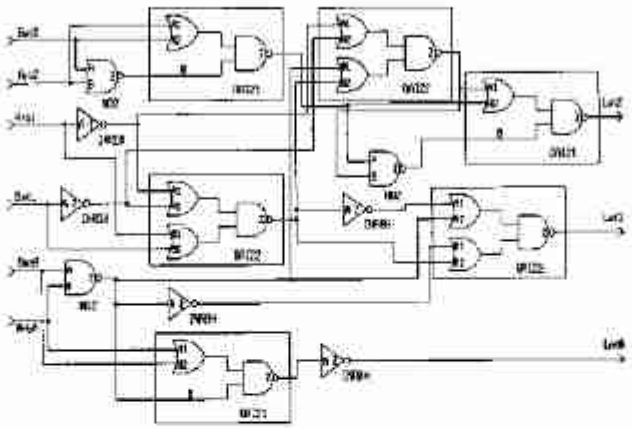



Figure 2-6 A 3-bit adder.

In this example, a 3-bit adder is being modeled. The number system for the operands is unsigned since they are of the net type. The leftmost bit is the most significant bit.

Here is an example in which the operands are signed numbers. This is achieved by using the integer type.

```
module SignedAdder (Arb, Bet, Lot);
  input [1:0] Arb, Bet;
  output [2:0] Lot;
  reg [2:0] Lot;

  always @ (Arb or Bet)
  begin: LABEL_A
    // A sequential block requires a label if local
    // declarations are present.
    integer ArbInt, BetInt;

  ArbInt = - Arb; // Store negative number just to show
       //that the + is operating on signed operands.
  BetInt = Bet;
  Lot = ArbInt + BetInt;
 end
endmodule
```

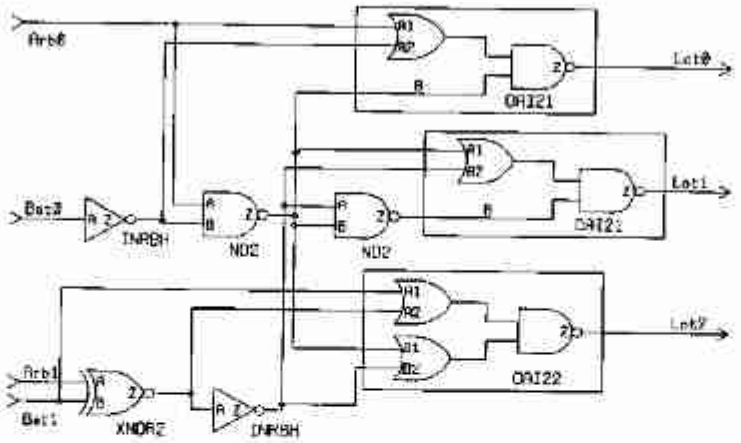



Figure 2-7 Signed adder.

Note that the adder logic with signed operands is the same as that with unsigned operands since the signed values are represented in 2's complement form.

Relational Operators

The relational operators supported for synthesis are:

```
>, <, <=, >=
```

Relational operators can be modeled similar to arithmetic operators. In this case, the logic produced from synthesis is different depending on whether unsigned or signed numbers are being compared. If variables of a reg type or a net type are compared, an unsigned relational operator is synthesized. If integer variables are compared, then a signed relational operator is synthesized. Here is an example of a relational operator that is used with unsigned numbers

```
module GreaterThan (A, B, Z);
  input [3:0] A, B;
  output Z;

  assign Z = A[1:0] > B[3:2];
  // Variables A and B are of net type.
endmodule
```

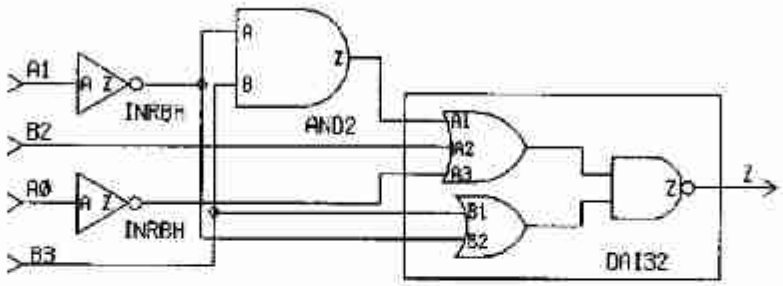



Figure 2-8 Unsigned ">" relational operator.

Equality operators

The equality operators supported for synthesis are:

==, !=

The operators === (case equality) and !== (case inequality) are not supported for synthesis.

Equality operators are modeled similar to arithmetic operators in terms of whether signed or unsigned comparison is to be made. Here is an example that uses signed numbers. Note that in this case, the operands of the equality operator are of integer type because values of this type represent signed numbers.


```
module NotEquals (A, B, Z);
  input [0:3] A, B;
  output Z;                    // Synthesized netlist is shown in Figure 2-10.
  reg Z;

  always @ (A or B)
  begin: DF_LABEL
    integer IntA, IntB;

    IntA = A;
    IntB = B;
    Z = IntA != IntB;
  end
endmodule
```

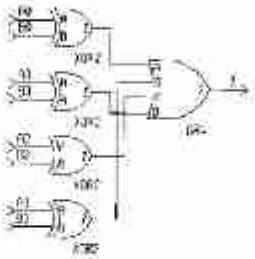


Figure 2-10 Signed "!=" relational operator.

Shift operators

Verilog HDL synthesis supports the left shift (<<) and the right shift (>>) operators. The vacated bits are filled with 0. The right operand, which is the amount of shift, may either be a constant or a variable. In both cases, combinational logic is produced. When shifting by a constant, simple rewiring is performed. When shifting by a variable, a general-purpose shifter is synthesized. This is shown in the following examples.

```
module ConstantShift (DataMux, Address);
  input [0:3] DataMux;
  output [0:5] Address;

  assign Address = (~ DataMux) << 2;
endmodule
// Synthesized netlist is shown in Figure 2-11.
```

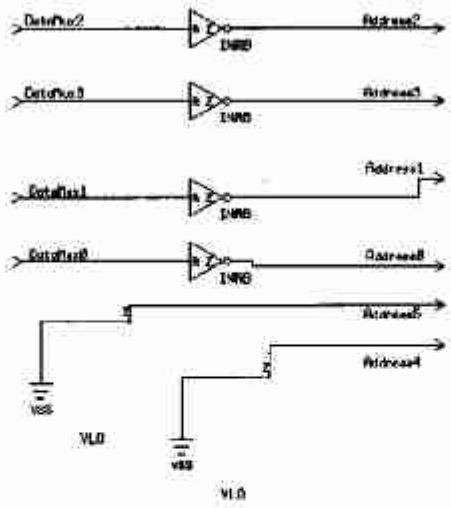



Figure 2-11 Constant shift.

As per Verilog HDL rules, when performing the left shift operation in module *ConstantShift*, the shifted bits from *DataMux* are not discarded but simply move into the higher order bits of *Address*. If *Address* were the same size as *DataMux*, then the high-order bits get shifted out and discarded.

Vector operations

This example shows that vector operands can be used in expressions. The four bits of *A* are and'ed with the four bits of *B*, the result of which is or'ed with the four bits of *C*. The result is assigned (starting with the rightmost bit) to the target net *RFile*.

```
module VectorOperations (A, B, C, RFile);
  input [3:0] A, B, C;
  output [3:0] RFile;

  assign RFile = (A & B) | C;
endmodule
// Synthesized netlist is shown in Figure 2-13.
```

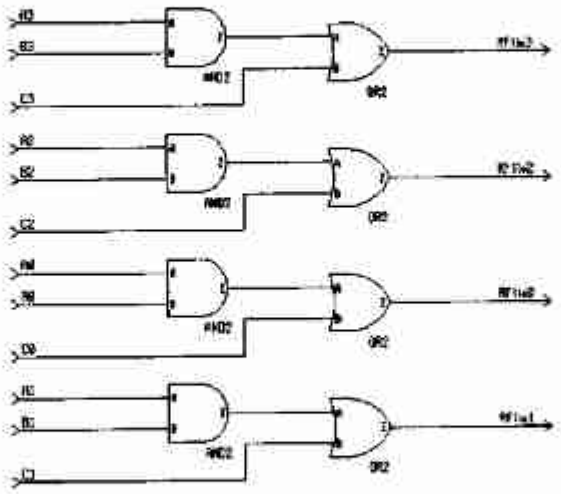


Figure 2-13 Vector operations.

Here is another example where the operands of a logical operator are vectors. In such a case, a series of logic gates to cover the range of the vector are produced.

```
module VectorOperands (Bi, Stdy, Tap);
  input [0:3] Bi, Stdy;
  output [0:3] Tap;

  assign Tap = Bi ^ Stdy;
endmodule
// Synthesized netlist is shown in Figure 2-14.
```

Four exclusive-or gates are synthesized since each operand in the right-hand-side is of size 4.

In the above examples on continuous assignments, there is a one-to-one correlation between a continuous assignment statement and its synthesized logic. This is because a continuous assignment implicitly describes the structure.

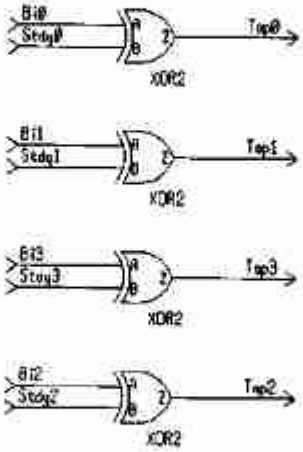


Figure 2-14 A bank of logic gates.

Part-selects

Operations using part-selects can be used in a model. Here is an example.

```
module PartSelect (A, C, ZCat);
  input [3:0] A, C;
  output [3:0] ZCat;

  assign ZCat[2:0] = {A[2], C[3:2]};
endmodule
// Synthesized netlist is shown in Figure 2-15.
```

ZCat[2:0] and *C*[3:2] are examples of part-selects. Non-constant part-selects are not supported in Verilog HDL.

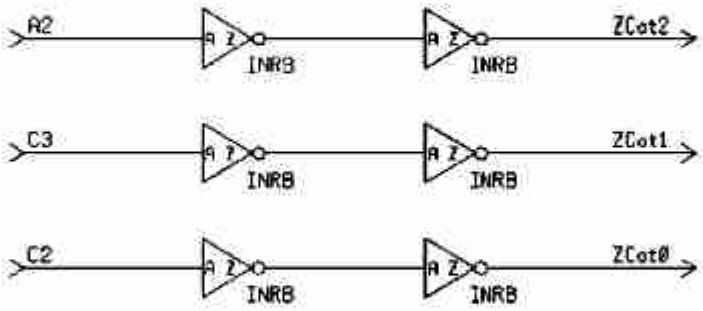


Figure 2-15 Part-select example.

Bit-selects

A bit-select can be a constant index or a non-constant index.

Constant Index

Here is an example that uses constant values for bit-select indices.

```
module ConstantIndex (A, C, Reg_File, ZCat);
  input [3:0] A, C;
  input [3:0] Reg_File;
  output [3:0] ZCat;

  assign ZCat[3:1] = {A[2], C[3:2]};
  assign ZCat[0] = Reg_File[3];
endmodule
// Synthesized netlist is shown in Figure 2-16.
```

A[2], *ZCat*[0] and *Reg_File*[3] are examples of bit-selects. The concatenation operator, { }, is used to generate a bigger array.

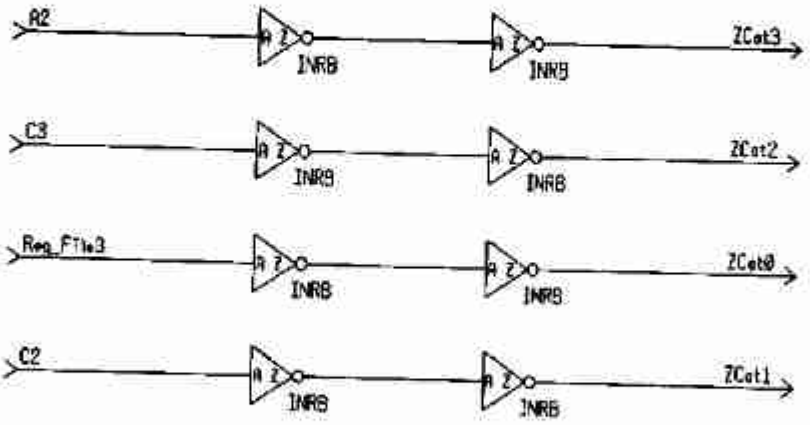


Figure 2-16 Constant bit-select.

Non-constant Index in Expression

It is possible to use a non-constant as an index in a bit-select as shown in the following model.

```
module NonComputeRight (Data, Index, Dout);
  input [0:3] Data;
  input [1:2] Index;
  output Dout;

  assign Dout = Data [Index];
endmodule
// Synthesized netlist is shown in Figure 2-17.
```

In this case, a multiplexer is generated as shown in the synthesized netlist.

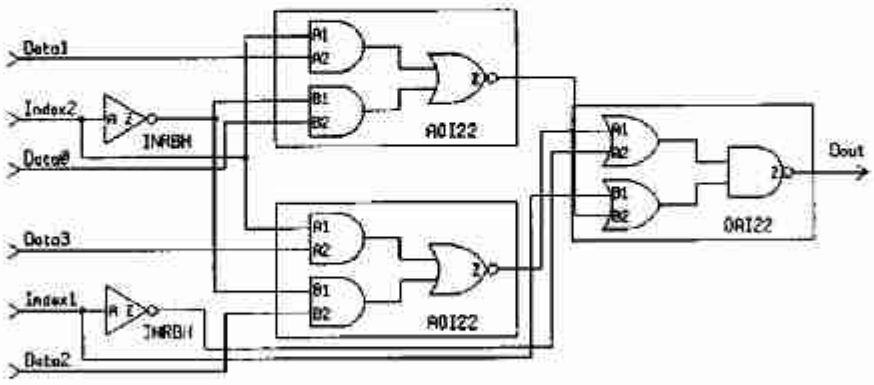


Figure 2-17 Non-constant bit-select generates a multiplexer.

Conditional Expression

A conditional expression selects between two expressions according to the value of a condition.

```
<condition> ? <expression1> : <expression2>
```

If the condition is true, select the first expression, else select the second. Here is an example.

```
module ConditionalExpression (StartXM, ShiftVal,
                              Reset, StopXM);
  input StartXM, ShiftVal, Reset;
  output StopXM;
```

```
    assign StopXM = (! Reset) ? StartXM ^ ShiftVal :
                                  StartXM | ShiftVal;
endmodule
// Synthesized netlist is shown in Figure 2-19.
```

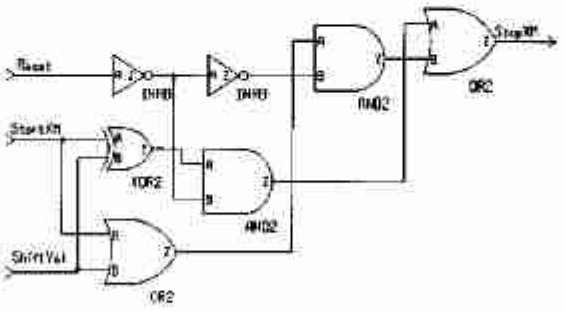


Figure 2-19 Logic generated from a conditional expression.

Always statement

An always statement is used to model the procedural behavior of a circuit. Here is an example of an always statement that contains procedural assignment statements.

```
module EvenParity (A, B, C, D, Z);
  input A, B, C, D;
  output Z;
  reg Z, Temp1, Temp2;

  always @(A or B or C or D)
  begin
    Temp1 = A ^ B;
    Temp2 = C ^ D;
    Z = Temp1 ^ Temp2;
    // Note that the temporaries are really not
    // required. They are used here to illustrate the
    // sequential behavior of the statements within
    // the sequential block.
  end
```

```
endmodule
// Synthesized netlist is shown in Figure 2-20.
```

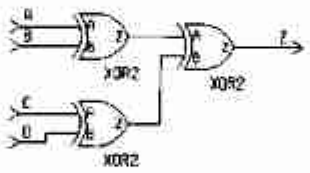


Figure 2-20 Procedural assignment statements.

All variables whose values are read in the always statement must appear in the event list (the parenthesized list following the "@" symbol); otherwise the functionality of the synthesized netlist may not match that of the design model. Here is a simple example that illustrates this point.


```
module AndBehavior (Z, A, B);
  input A, B;
  output Z;
  reg Z;

  always @ (B)
    Z = A & B;
endmodule
// Synthesized netlist is shown in Figure 2-21.
```

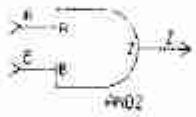


Figure 2-21 Incomplete event list.

The semantics of the always statement specifies that whenever an event occurs on *B*, the assignment is to be executed and *Z* gets a value. If any events occur on *A*, this has no impact on the value of Z. However, the synthesized netlist of the above module, shown in Figure 2-21, shows an and gate. Here any time *A* or *B* changes, the value of *Z* is updated. Hence a functional mismatch occurs. A synthesis system usually would issue a warning about such missing variables in the event list.

A good practice is to include all variables read in the always statement in the event list; this is true only when modeling combinational logic. When modeling sequential logic, a different kind of event list is required; this is described later.

A variable declared within an always statement holds a temporary value and does not necessarily infer a unique wire in hardware as the following example shows.

```
module VariablesAreTemporaries (A, B, C, D, Z);
  input A, B, C, D;
  output Z;
  reg Z;

  always @ (A or B or C or D)
  begin: VAR_LABEL
    integer T1, T2;

    T1 = A & B;
    T2 = C & D;
    T1 = T1 | T2;
    Z = ~ T1;
  end
endmodule
```

```
// Synthesized netlist is shown in Figure 2-22.
```

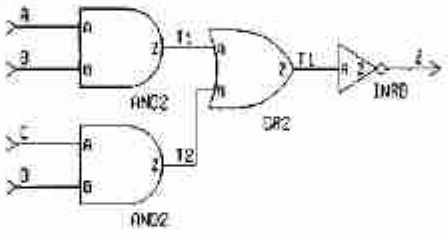


Figure 2-22 One variable can represent many wires.

In the synthesized netlist, the output of the *AND2* gate is the variable *T1*; so is the output of the *OR2* gate. In this example, each assignment to the integer variable infers a unique wire.

If statement

An `if` statement represents logic that is conditionally controlled. Here is an example.

```
module SelectOneOf (A, B, Z);
  input [1:0] A, B;
  output [1:0] Z;
  reg [1:0] Z;

  always @ (A or B)
    if (A > B)
      Z = A;
    else
      Z = B;
endmodule
```

```
// Synthesized netlist is shown in Figure 2-23.
```

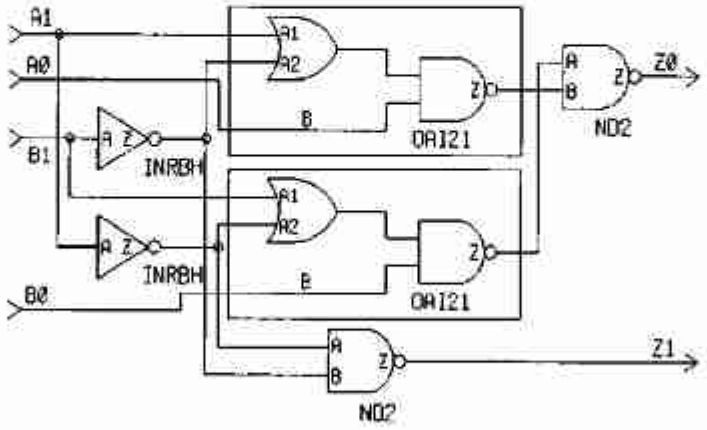


Figure 2-23 Logic derived from an `if` statement.

Case statement

```
module ALU (Op, A, B, Z);
  input [1:2] Op;
  input [0:1] A, B;
  output [0:1] Z;
  reg [0:1] Z;

  parameter ADD = 'b00, SUB = 'b01, MUL = 'b10,
            DIV = 'b11;

  always @(Op or A or B)
    case (Op)
      ADD  :  Z = A + B;
      SUB  :  Z = A - B;
      MUL  :  Z = A * B;
      DIV  :  Z = A / B; // The A/B operation may not be
                         // supported by some synthesis tools.
    endcase
endmodule
```

A case statement behaves like a nested `if` statement, that is, the value of the case expression (*Op*) is checked with the first case item (*ADD*), if it does not match, the second case item (*SUB*) is checked and so on. The equivalent `if` statement for the above case statement is shown next.

```
if (Op == ADD)
  Z = A + B;
else if (Op == SUB)
  Z = A - B;
else if (Op == MUL)
  Z = A * B;
else if (Op == DIV)
  Z = A / B;
```

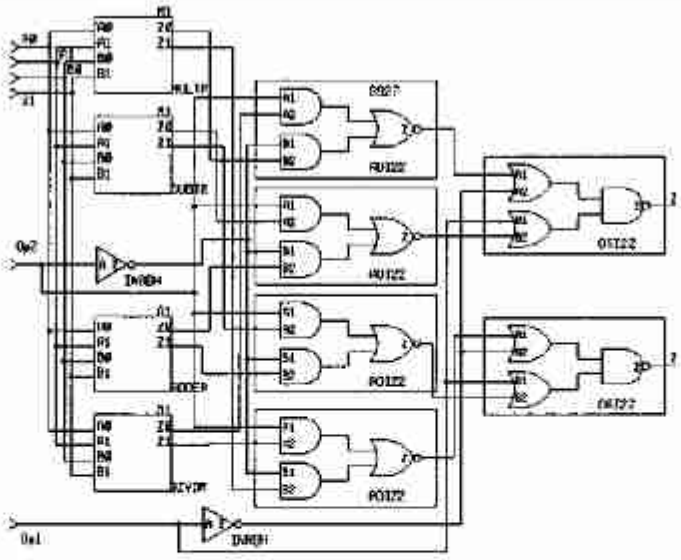


Figure 2-26 A 2-bit ALU.

Casex

In a casex statement, the values x and z (? for a z is ok too) in a case item expression are considered as don't care values. These values, for synthesis purposes, cannot appear as part of the case expression. Here is an example of a casex statement used to model a priority encoder.

```
module PriorityEncoder (Select, BitPosition);
  input [5:1] Select;
  output [2:0] BitPosition;
  reg [2:0] BitPosition;

  always @ (Select)
    casex (Select)
    5'bxxxx1 : BitPosition = 1;
    5'bxxx1x : BitPosition = 2;
    5'bxx1xx : BitPosition = 3;
    5'bx1xxx : BitPosition = 4;
    5'b1xxxx : BitPosition = 5;
    default : BitPosition = 0;
  endcase
endmodule
```

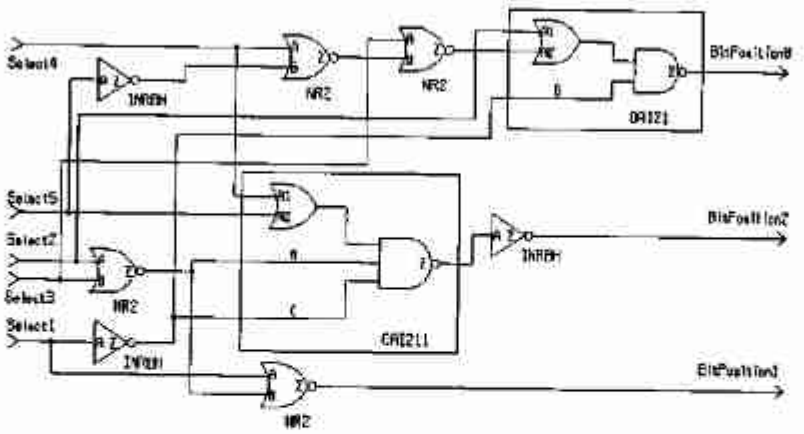



Figure 2-32 A priority encoder using casex statement.

Casez

In a casez statement, the value z is considered as a don't-care when it appears in a case item expression. The ? character can also be used alternatively for the character z. Values z and x are not allowed in a case expression. Additionally, value x cannot appear in a case item expression.

```
module CasezExample (ProgramCounter, DoCommand);
 input [0:3] ProgramCounter;
 output [0:1] DoCommand;
 reg [0:1] DoCommand;

 always @ (ProgramCounter)
  casez (ProgramCounter)
    4'b???1 : DoCommand = 0;
    4'b??10 : DoCommand = 1;
    4'b?100 : DoCommand = 2;
    4'b1000 : DoCommand = 3;
    default : DoCommand = 0;
  endcase
endmodule
```

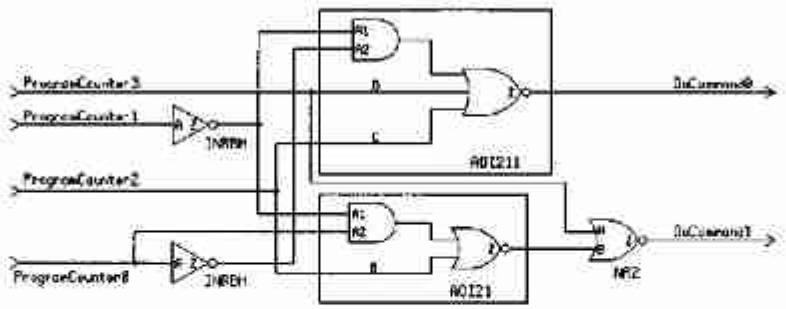



Figure 2-31 Casez statement example.

The casez statement is equivalent to the following `if` statement (note that the `?` character in a case item denotes a don't-care value).

```
if (ProgramCounter [3])
  DoCommand = 0;
else if (ProgramCounter [2:3] == 2'b10)
  DoCommand = 1;
else if (ProgramCounter [1:3] == 3'b100)
  DoCommand = 2;
else if (ProgramCounter [0:3] == 4'b1000)
  DoCommand = 3;
else
  DoCommand = 0;
```


Inferring latches

A latch can be inferred by using an incompletely specified `if` statement or a case statement, that is, if a variable is not assigned a value in all branches of an `if` statement or a case statement, a latch is inferred for that variable. Here is an example.

```
module LatchExample (ClockA, CurrentState, NextState);
  input ClockA;
  input [3:0] CurrentState;
  output [3:0] NextState;
  reg [3:0] NextState;

  always @ (ClockA or CurrentState)
    if (ClockA)
      NextState = CurrentState;
  endmodule
```

```
// Synthesized netlist is shown in Figure 2-39.
```

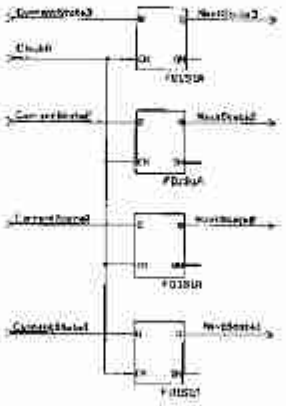


Figure 2-39 An incompletely specified condition infers a latch.

The variable *NextState* is assigned a value only when *ClockA* is 1. If *ClockA* is 0, *NextState* retains its previous value, thus inferring a latch.

Loop statement

There are four kinds of loop statements in Verilog HDL.

- while-loop
- for-loop
- forever-loop
- repeat-loop

The for-loop statement is the one typically supported for synthesis. A for-loop is implemented by unrolling the for-loop, that is, all statements within the for-loop are replicated, once for each value of the for-loop index. This puts a restriction on the for-loop bounds, which must therefore evaluate to constants. Here is an example of a for-loop statement.

```
module DeMultiplexer (Address, Line);
  input [1:0] Address;
  output [3:0] Line;
  reg [3:0] Line;

  integer J;

  always @ (Address)
    for (J = 3; J >= 0; J = J - 1)
      if (Address == J)
        Line[J] = 1;
      else
        Line[J] = 0;
endmodule
```



```
// Synthesized netlist is shown in Figure 2-44.
```

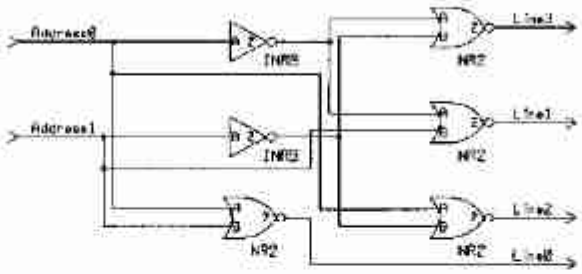


Figure 2-44 A for-loop example

When the for-loop is expanded, the following four IF statements are obtained.

```
if (Address == 3) Line[3] = 1; else Line[3] = 0;
if (Address == 2) Line[2] = 1; else Line[2] = 0;
if (Address == 1) Line[1] = 1; else Line[1] = 0;
if (Address == 0) Line[0] = 1; else Line[0] = 0;
```

Modeling Flip-flops

A flip-flop is inferred from a variable when it is assigned a value in a special form of always statement. This always statement is of the form:

```
always @ (<clock_event>)
  <statement>
```

where *<clock_event>* is one of:

```
posedge <clock_name>
negedge <clock_name>
```

The semantics of the always statement implies that all statements in *<statement>* are to be executed only when a rising edge or a falling edge of clock occurs. We shall call this special always statement as a *clocked always statement*.

When modeling sequential logic, it is recommended that a non-blocking procedural assignment be used for a variable that is assigned in a clocked always statement and its value used outside of the always statement; this is to prevent any possibility of functional mismatch between the design model and its synthesized netlist. Such a target of a non-blocking assignment that appears in a clocked always statement accurately models the behavior of a sequential element.

```
module PickOne (A, B, Clock, Control, Zee);
  input A, B, Clock, Control;
  output Zee;
  reg Zee;

  always @ (negedge Clock)
    if (Control)
      Zee <= A;
    else
      Zee <= B;
endmodule
```



```
// Synthesized netlist is shown in Figure 2-45.
```

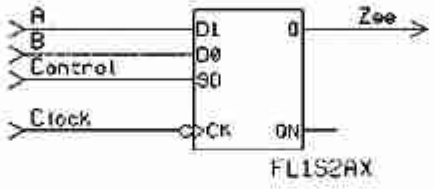


Figure 2-45 Sequential logic synthesized using a special always statement.

The assignment to output *Zee* occurs only at the falling edge of the clock. Variable *Zee* is inferred to be a falling-edge-triggered flip-flop. The flip-flop shown in the figure has a data-select, that is, the input *Control* selects either *A* or *B* as the data for the flip-flop.

Functions

A function call represents combinational logic since a function call is part of an expression in Verilog HDL. A function call is synthesized by expanding the function call into in-line code. Any local variable declared within the function is treated as a pure temporary; such a variable gets synthesized as a wire.

```
module FunctionCall (XBC, DataIn);
 output XBC;
 input [0:5] DataIn;

 function [0:2] CountOnes;
  input [0:5] A;
  integer K;
  begin
   CountOnes = 0;

   for (K = 0; K <= 5; K = K + 1)
    if (A[K])
     CountOnes = CountOnes + 1;
  end
 endfunction
```



```
  // If number of ones in DataIn is greater than 2,
  // return 1 in XBC.
  assign XBC = CountOnes (DataIn) > 2;
endmodule
// Synthesized netlist is shown in Figure 2-59.
```

After in-line expansion of the function call and further in-line expansion of the for-loop statement, the following code is obtained.

```
CountOnes = 0;
if (DataIn[0]) CountOnes = CountOnes + 1;
if (DataIn[1]) CountOnes = CountOnes + 1;
if (DataIn[2]) CountOnes = CountOnes + 1;
if (DataIn[3]) CountOnes = CountOnes + 1;
if (DataIn[4]) CountOnes = CountOnes + 1;
```

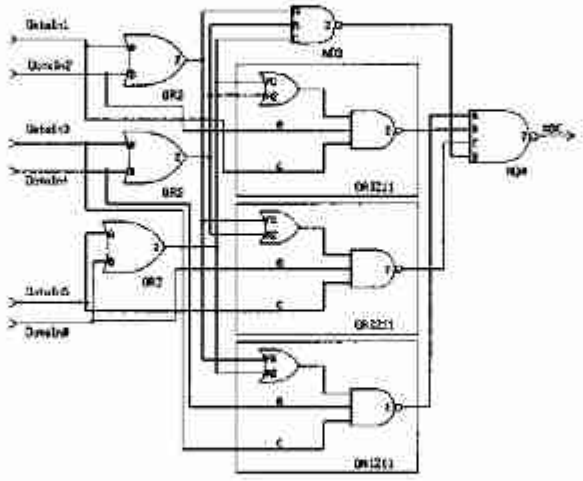


Figure 2-59 A function call example.

```
if (DataIn[5]) CountOnes = CountOnes - 1;
XOC = CountOnes > 2;
```


Tasks

A task call can represent either combinational logic or sequential logic depending on the context under which the task call occurs. By this, we mean that the output parameters of a task call may imply memory depending on the context in which they are assigned. For example, if a task call occurs in a clocked always statement (always statement with a clock event), then an output parameter in a task call may be synthesized as a flip-flop; this is determined by using the flip-flop inference rules. A synthesis system implements a task call by expanding the task call in-line with the rest of the code; in effect, no separate hierarchy for the task call is maintained.

Here is an example of a task call that represents pure combinational logic.

```
module CombTask (ShA, ShB, ShCarryIn, ShSum,
                 ShCarryOut);
  input [0:2] ShA, ShB;
  input ShCarryIn;
  output [0:2] ShSum;
  output ShCarryOut;
  reg [0:3] TempCarry;

  task AddOneBitWithCarry;
    input A, B, CarryIn;
    output Sum, CarryOut;
    begin
      Sum = A ^ B ^ CarryIn;
      CarryOut = A & B & CarryIn;
    end
  endtask
```



```
  always @ (ShA or ShB or ShCarryIn)
  begin: EXAMPLE
    integer J;

    TempCarry[0] = ShCarryIn;

    for (J = 0; J < 3; J = J + 1)
      AddOneBitWithCarry (ShA[J], ShB[J],
        TempCarry[J], ShSum[J], TempCarry[J+1]);
  end

  assign ShCarryOut = TempCarry[3];
endmodule
```

After in-line expansion of the task call and the for-loop statement by the synthesis tool, the following code is obtained.

```
TempCarry[0] = ShCarryIn;
ShSum[0] = ShA[0] ^ ShB[0] ^ TempCarry[0];
TempCarry[1] = ShA [0] & ShB[0] & TempCarry[0];
ShSum[1] = ShA[1] ^ ShB[1] ^ TempCarry[1];
TempCarry[2] = ShA[1] & ShB[1] & TempCarry[1];
ShSum[2] = ShA[2] ^ ShB[2] ^ TempCarry[2];
```


// Synthesized netlist is shown in Figure 2-60.

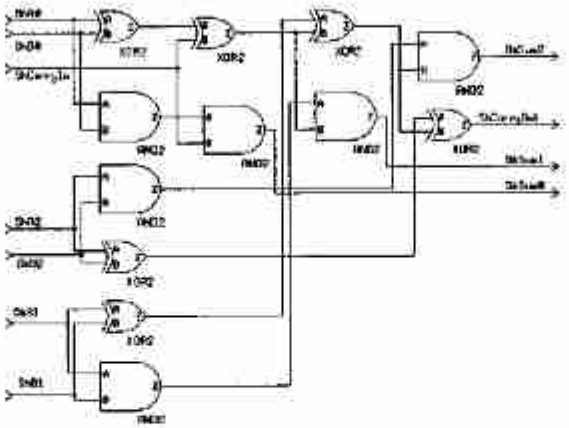


Figure 2-60 A task call example: combinational logic

```
TempCarry[3] = ShA[2] & ShB[2] & TempCarry[2];
ShCarryOut = TempCarry[3];
```

Using values x and z

Verilog HDL has two non-logical values: x (unknown) and z (high-impedance). In this section, we specify the domain under which these values can be used for synthesis. Use caution when using these values in a synthesis model as they can potentially cause a functional mismatch between the design model and the synthesized netlist.

The Value x

The value x can be assigned to any variable in an assignment statement. In such a case, x is treated as a don't-care for synthesis purposes. A synthesis system may intelligently select either a logic-0 or a logic-1 for the value x that leads to optimal logic.

```
Reset = 'bx;  // Assign a don't-care value to Reset.
  // Synthesis system will automatically select
  // logic-0 or logic-1.
```


When value x is used in a case item of a case statement (not casex, casez), the branch corresponding to that case item is considered never to execute for synthesis purposes.

```
case (In)
  2'bx : Out = In;     // This branch will never occur for
                       // synthesis.
  default : Out = ~ In;
endcase
```

Thus a functional mismatch may occur; a synthesis tool may report a warning in such a case. Avoid using x in a case item of a case statement (not casex, casez).

The Value z

The value z is used to generate a three-state gate. The value z can be assigned to a variable in an assignment statement; however for synthesis, such an assignment must occur under the control of a condition, either in an `if` statement, or in a case statement. Here is an example.

```
module ThreeState (Ready, DataInA, DataInB, Select1);
  input Ready, DataInA, DataInB;
  output Select1;
  reg Select1;

  always @ (Ready or DataInA or DataInB)
    if (Ready)
      Select1 = 1'bz;
    else
      Select1 = DataInA & DataInB;
endmodule
```



```
// Synthesized netlist is shown in Figure 2-62.
```

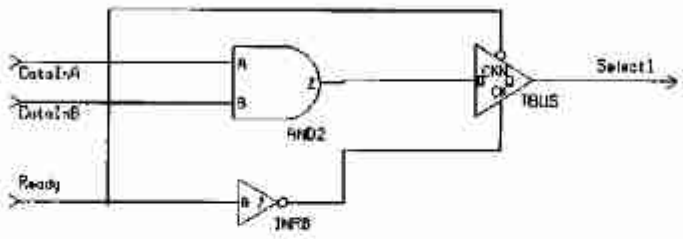


Figure 2-62 A conditional z value assignment produces a three-state gate.

A three-state gate can also be obtained by assigning the value z in a conditional expression, as shown in the next example.

Gate level modeling

Gate level primitives can be instantiated in a model using gate instantiation. The following gate level primitives are supported for synthesis.

and, nand, or, nor, not, xor, xnor, buf,
bufif0, bufif1, notif0, notif1

```
module GateLevel (A, B, Ctrl, Zbus);
  input A, B, Ctrl;
  output Zbus;
  // Not necessary to declare nets AndOut and OrOut.
  // The instance names, A1, O1, etc. are also optional
  // but are recommended for simulation debugging.
  and A1 (AndOut, A, B); // First terminal is output,
                         // other two are inputs.
  or O1 (OrOut, A, B);
  bufif0 B1 (Zbus, AndOut, Ctrl);  // First terminal is
       // output, second terminal is input, and third
       // terminal is control.
  bufif0 B2 (Zbus, OrOut, ! Ctrl);
endmodule
```


Synthesizing a gate primitive simply generates logic based on the gate behavior, which eventually gets mapped to the target technology. Synthesizing any of the last four listed primitives (three-state gate primitives), synthesizes a three-state gate in the appropriate target technology with additional combinational logic to support the behavior of the three-state gate. Here is an example that drives the and of two inputs onto a bus if control is 1, else it drives the or of the two inputs.

```
// Synthesized netlist is shown in Figure 2-66.
```

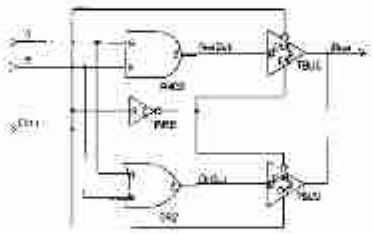


Figure 2-66 Gate instantiations.

Module Instantiation Statement

A module instantiation statement can be written within a module declaration. A synthesis system treats such a module instance as a black box and does not take further action, that is, the module instance appears in the synthesized netlist as if it were a primitive component. Here is an example of a full-adder module that contains one module instantiation statement. Notice that in the synthesized netlist, shown in Figure 2-67, the module *MyXor* appears just as it is described in the top level module *FullAdderMix*.


```
module FullAdderMix (A, B, CarryIn, Sum, CarryOut);
  input A, B, CarryIn;
  output Sum, CarryOut;
  wire Sft;                    // Sft need not be declared.

  MyXor X1 (.In0(A), .In1(B), .Out(Sft));

  assign CarryOut = A & B & CarryIn;
  assign Sum = Sft ^ CarryIn;
endmodule
```

```
// Synthesized netlist is shown in Figure 2-67.
```

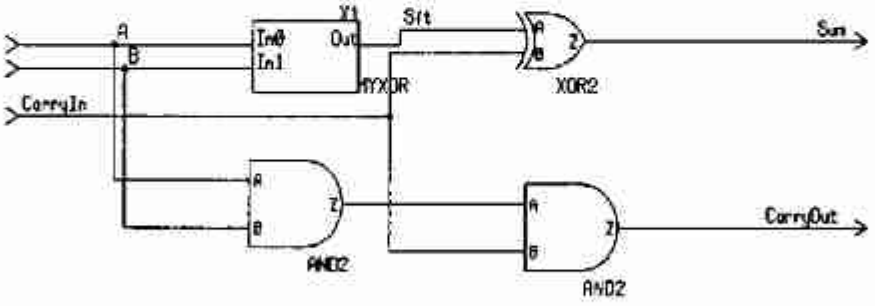


Figure 2-67 A module instance mixed with behavior.

Parameterized Designs

Parameters in Verilog HDL provide a powerful mechanism to model parameterized designs. Here is a simple example of an *N*-bit register.

```
module NbitRegister (Data, Clock, Q);
  parameter N = 3;
  input [N-1:0] Data;
  input Clock;
  output [N-1:0] Q;
  reg [N-1:0] Q;

  always @ (negedge Clock)
    Q <= Data;
endmodule
```

The module *NbitRegister* when synthesized produces a 3-bit register. The module is a parameterized module since the size of the register has been specified using a parameter which can be modified easily or overwritten by instantiating it from another module. For example, if a 4-bit register is required, one way is to just change N in the module *NbitRegister* to 4 and resynthesize. The other alternative is to instantiate *NbitRegister* in a different module and specify a new value for N; this approach has the advantage that the parameterized module *NbitRegister* does not have to be modified. Here is a module that instantiates two 2-bit registers. The new parameter values, that is, the new values for N, are specified using the # symbol.


```
// Synthesized netlist is shown in Figure 2-71.
```

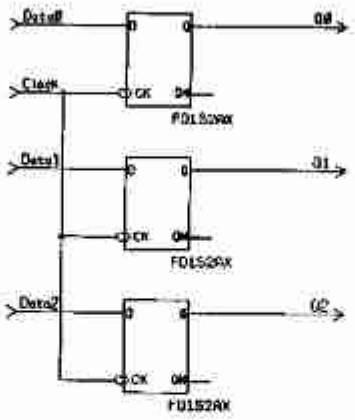


Figure 2-71 A parameterized register.

Overview of Digital Design with Verilog HDL

Evolution of CAD:

In early days digital circuits were designed with vacuum tubes and transistor. Then integrated circuits chips were invented which consists of logic gates embed on them. As technology advances from SSI (Small Scale Integration), MSI (Medium Scale Integration), LSI (Large Scale Integration), designers could implement thousands of gates on a single chip. So the testing of circuits and designing became complicated hence Electronic Design Automation (EDA) techniques to verify functionality of building blocks were one.

The advances in semiconductor technology continue to increase the power and complexity of digital systems with the invent of VLSI (very Large Scale Integration) with more than 10000 transistors. Because of the complexity of circuit, breadboard design became impossible and gave rise to computer aided techniques to design and verify VLSI digital circuits. These computer aided programs and tools allow us to design, do automatic placement and routing and Abe to develop hierarchical based development and hence prototype development by downloading of programmable chips (like - ASIC, FPGA, CPLD) before fabrication.

Emergence of HDLs:

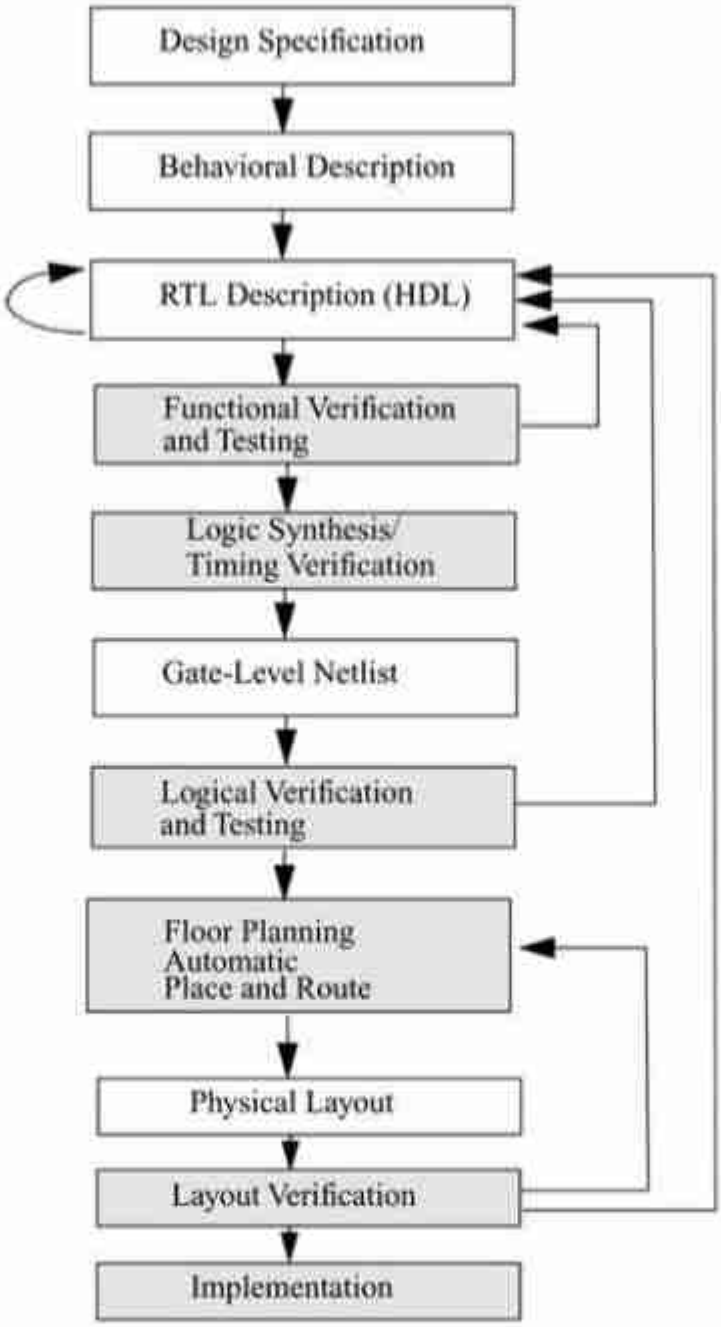
In the field of digital design, the complexity in designing a circuit gave birth to standard languages to describe digital circuits (ie. Hardware Description Languages - HDL). HDL is a Computer Aided design (CAD) tool for the modern design and synthesis of digital systems. HDLs were been used to model hardware elements very concurrently. Verilog HDL and VHDL are most popular HDLs.

In initial days of HDL, designing and verification were done using tool but synthesis (ie translation of RTL to schematic circuit) used to be done manually which become tediously as technology advances. Later tool is automated to generate the schematic of RTL developed.

Digital circuits are described at Registers Transfer Level (RTL) by using HDL. Then logic synthesis tool will generate details of gates and interconnection to implement circuits. This synthesised result can be used for fabrication by having placement and routing details. Verify functionality using simulation. HDLs are used for system-level design - simulation of system boards, interconnect buses, FPGAs and PALs. Verilog HDL is a IEEE standard - IEEE 1364-2001.

Note: RTL - designer has to specify how the data flows between registers and how the design processes the data.

Typical HDL flow:



A typical design flow (HDL flow) for designing VLSI IC circuits is as shown in figure below.

The design flow In any design, specifications are written first. Specifications describe abstractly the functionality, interface, and overall architecture of the digital circuit to be designed. At this point, the architects do not need to think about how they will implement this circuit.A behavioral description is

then created to analyze the design in terms of functionality, performance, and compliance to standards, and other high-level issues. Behavioral descriptions are often written with HDLs.

New EDA tools have emerged to simulate behavioral descriptions of circuits. These tools combine the powerful concepts from HDLs and object oriented languages such as C++. These tools can be used instead of writing behavioral descriptions in Verilog HDL. The behavioral description is manually converted to an RTL description in an HDL. The designer has to describe the data flow that will implement the desired digital circuit. From this point onward, the design process is done with the assistance of EDA tools.

Logic synthesis tools convert the RTL description to a gate-level net list. Logic synthesis tools ensure that the gate-level net list meets timing, area, and power specifications.

A gate-level net list is a description of the circuit in terms of gates and connections between them. The gate-level netlist is input to an Automatic Place and Route tool, which creates a layout.

The layout is verified and then fabricated on a chip.

Thus, most digital design activity is concentrated on manually optimizing the RTL description of the circuit. After the RTL description is frozen, EDA tools are available to assist the designer in further processes. Designing at the RTL level has shrunk the design cycle times from years to a few months. It is also possible to do many design iterations in a short period of time.

Behavioral synthesis tools have begun to emerge recently. These tools can create RTL descriptions from a behavioral or algorithmic description of the circuit. As these tools mature, digital circuit design will become similar to high-level computer programming. Designers will simply implement the algorithm in an HDL at a very abstract level. EDA tools will help the designer convert the behavioral description to a final IC chip.

Why Verilog HDLs?

HDLs have many advantages that helps in developing large digital circuits reaching the optimised circuit design.

- Designs can be described at a very abstract level by use of HDLs. Designers can write their RTL description without choosing a specific fabrication technology. Logic synthesis tools can automatically convert the design to any fabrication technology. If a new technology emerges, designers do not need to redesign their circuit. They simply input the RTL description to the logic synthesis tool and create a new gate-level netlist, using the new fabrication technology. The logic synthesis tool will optimize the circuit in area and timing for the new technology.
- By describing designs in HDLs, functional verification of the design can be done early in the design cycle. Since designers work at the RTL level, they can optimize and modify the RTL description until it meets the desired functionality. Most design bugs are eliminated at this

point. This cuts down design cycle time significantly because the probability of hitting a functional bug at a later time in the gate-level netlist or physical layout is minimized.

- Designing with HDLs is similar to computer programming. A textual description with comments is an easier way to develop and debug circuits. This also provides a concise representation of the design, compared to gate-level schematics. Gate-level schematics are almost incomprehensible for very complex designs.
- Verilog HDL is a general-purpose hardware description language that is easy to learn and easy to use. It is similar in syntax to the C programming language. Designers with C programming experience will find it easy to learn Verilog HDL.
- Verilog HDL allows different levels of abstraction to be mixed in the same model. Thus, a designer can define a hardware model in terms of switches, gates, RTL, or behavioral code. Also, a designer needs to learn only one language for stimulus and hierarchical design.
- Most popular logic synthesis tools support Verilog HDL. This makes it the language of choice for designers.
- All fabrication vendors provide Verilog HDL libraries for postlogic synthesis simulation. Thus, designing a chip in Verilog HDL allows the widest choice of vendors.
- The Programming Language Interface (PLI) is a powerful feature that allows the user to write custom C code to interact with the internal data structures of Verilog. Designers can customize a Verilog HDL simulator to their needs with the PLI.

Trends in HDLs

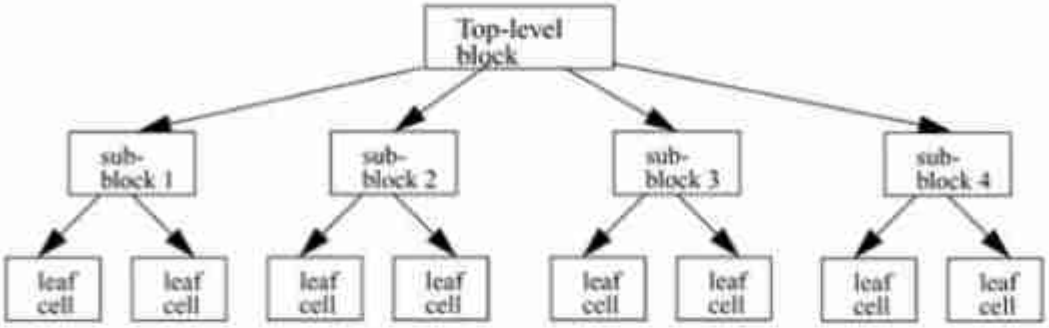
Increase in speed and complexity go digital circuits will complicate the designer job, but EDA tools make the job easy for designer. Designer has to do high level abstraction designing and need to take care of functionality of the design and EDA tools take care of implementation, and can achieve a almost optimum design.

Digital circuits are designed in HDL at an RTL level, so that logic synthesis tools can create gate level net lists. Behavioral synthesis allowed designers to directly design in terms of algorithms and the behavior of the circuit EDA tool is then used to translate and optimise at each phase of design. Verilog HDL is also used widely for verification. Formal verification uses mathematical techniques to verify the correctness of Verilog HDL descriptions and to establish equivalency between RTL and gate level net lists.Assertion checking is done to check the transition and important parts of a design.

Design Methodologies:

There are two types of design methodologies: Top down and bottom-up.

Top-down design methodology:

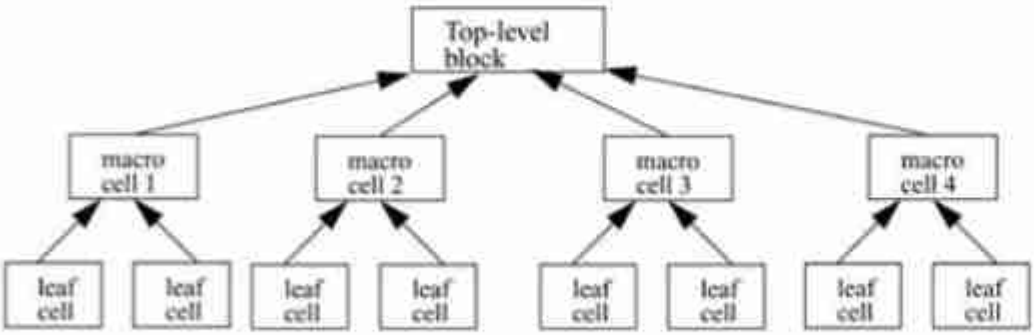


This designing approach allows early testing, easy change of different technologies, a well structures system design and offers many other advantages.

In this method, top-level block is defined and sub-blocks necessary to build the top-level block are identified.

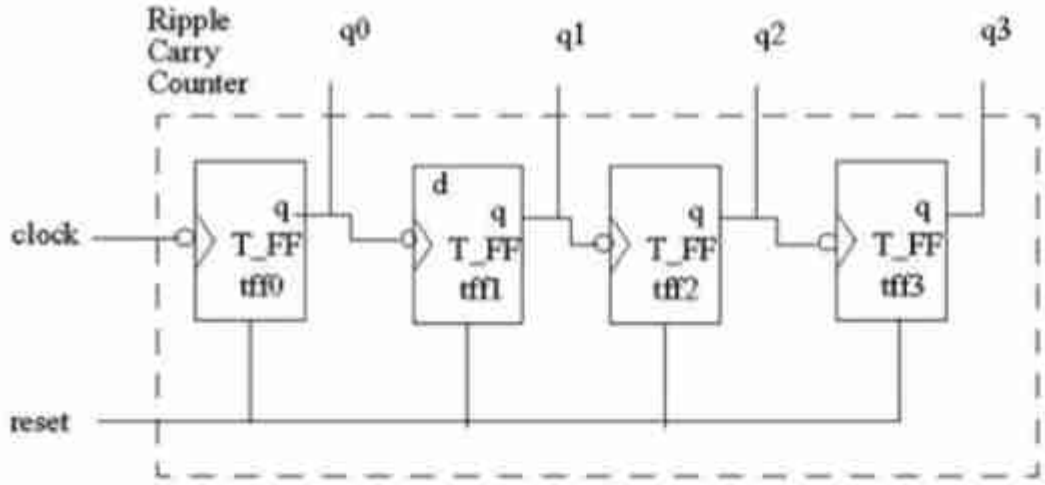
We further subdivide, sub-blocks until cells cannot be further divided, we call these cells as leaf cells.

Bottom-up design methodology:



We first identify the available building blocks and try to build bigger cells out of these, and continue process until we reach the top-level block of the design. Most of the time, the combination of these two design methodologies are used to design. Logic designers decide the structure of design and break up the functionality into blocks and sub blocks. And designer will design a optimized circuit for leaf cell and using these will design top level design.

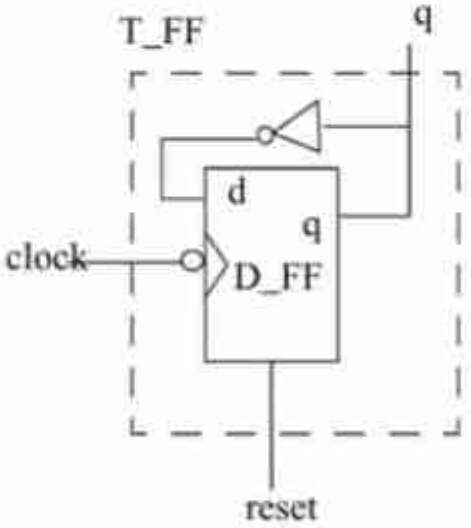
Illustration of hierarchical modelling concepts:

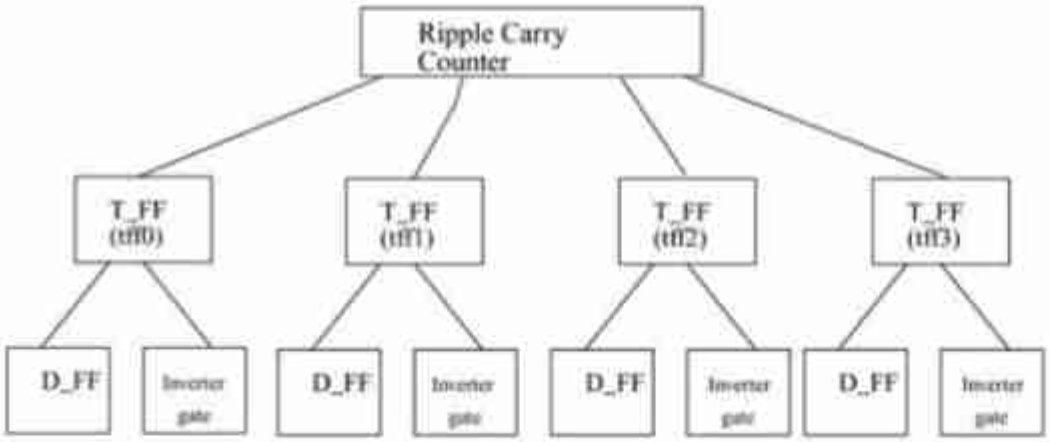


Example 1. 4 bit Ripple carry adder.

It is a circuit used to add two, 4 bit numbers can be designed using a adder that can add two, 1 bit numbers. The design is as shown in figure, using the concept of sub dividing the blocks. hence 4 bit Ripple carry Vader is built in a hierarchical fashion by using building blocks as follows.

reset	q_n	q_{n+1}
1	1	0
1	0	0
0	0	1
0	1	0
0	0	0





Differences between modules and module instances

We shall now study what module in verily and what is all about module instantiation.

Module:

Verilog provides the concept of a module. A module is the basic building block in Verilog. A module can be an element or a collection of lower-level design blocks. Typically, elements are grouped into modules to provide common functionality that is used at many places in the design. A module provides the necessary functionality to the higher-level block through its port interface (inputs and outputs), but hides the internal implementation. This allows the designer to modify module internals without affecting the rest of the design. In Verilog, a module is declared by the keyword module. A corresponding keyword endmodule must appear at the end of the module definition.

Each module must have a module_name, which is the identifier for the module, and a module_terminal_list, which describes the input and output terminals of the module.

Ripple carry counter, T_FF, D_FF are examples of modules.

module <module_name> (<module_terminal_list>);

...

<module internals>

...

...

endmodule

Example:

module T_FF (q, clock, reset);

.

.

<functionality of T-flipflop>

.

.

endmodule

Verilog is both a behavioral and a structural language.

Internals of each module can be defined at four levels of abstraction, depending on the needs of the design. The module behaves identically with the external environment irrespective of the level of abstraction at which the module is described. The internals of the module are hidden from the environment.

Thus, the level of abstraction to describe a module can be changed without any change in the environment.

- Behavioral or algorithmic level

This is the highest level of abstraction provided by Verilog HDL. A module can be implemented in terms of the desired design algorithm without concern for the hardware implementation details. Designing at this level is very similar to C programming.

- Dataflow level

At this level, the module is designed by specifying the data flow. The designer is aware of how data flows between hardware registers and how the data is processed in the design.

- Gate level

The module is implemented in terms of logic gates and interconnections between these gates. Design at this level is similar to describing a design in terms of a gate-level logic diagram.

- Switch level

This is the lowest level of abstraction provided by Verilog. A module can be implemented in terms of switches, storage nodes, and the interconnections between them. Design at this level requires knowledge of switch-level implementation details.

Verilog allows the designer to mix and match all four levels of abstractions in a design.

Module Instances:

A module provides a template from which you can create actual objects. When a module is invoked, Verilog creates a unique object from the template. Each object has its own name, variables, parameters, and I/O interface. The process of creating objects from a module template is called instantiation, and the objects are called instances. In Example 2, the top-level block creates four instances from the T-flipflop (T_FF) template. Each T_FF instantiates a D_FF and an inverter gate. Each instance must be given a unique name. Note that // is used to denote single-line comments.

Example: Module Instantiation

```
// Define the top-level module called ripple carry
// counter. It instantiates 4 T-flipflops. Interconnections are // shown in Section 2.2, 4-bit Ripple Carry Counter.
module ripple_carry_counter(q, clk, reset);
output [3:0] q; //I/O signals and vector declarations
        //will be explained later.
input clk, reset; //I/O signals will be explained later.
//Four instances of the module T_FF are created. Each has a unique
//name.Each instance is passed a set of signals. Notice, that
//each instance is a copy of the module T_FF.
T_FF tff0(q[0],clk, reset);
T_FF tff1(q[1],q[0], reset);
T_FF tff2(q[2],q[1], reset);
T_FF tff3(q[3],q[2], reset);
endmodule
// Define the module T_FF. It instantiates a D-flipflop. We assumed
// that module D-flipflop is defined elsewhere in the design. Refer
// to Figure 2-4 for interconnections.
module T_FF(q, clk, reset);
//Declarations to be explained later
output q;
input clk, reset;
wire d;
D_FF dff0(q, d, clk, reset); // Instantiate D_FF. Call it dff0.
not n1(d, q); // not gate is a Verilog primitive. Explained later.
endmodule
```

In Verilog, it is illegal to nest modules. One module definition cannot contain another module definition within the module and endmodule statements. Instead, a module definition can incorporate copies of other modules by instantiating them.

Module definitions simply specify how the module will work, its internals, and its interface. Modules must be instantiated for use in the design.

Example 2-2 Illegal Module Nesting

```
// Define the top-level module called ripple carry counter.
// It is illegal to define the module T_FF inside this module.
```

```
module ripple_carry_counter(q, clk, reset);
output [3:0] q;
input clk, reset;
  module T_FF(q, clock, reset);// ILLEGAL MODULE NESTING
  ...
  <module T_FF internals>
  ...
  endmodule // END OF ILLEGAL MODULE NESTING
endmodule
```

Parts of a simulation:

Once a design block is completed, it must be tested. The functionality of the design block can be tested by applying stimulus and checking results. We call such a block the stimulus block. It is good practice to keep the stimulus and design blocks separate. The stimulus block can be written in Verilog. A separate language is not required to describe stimulus. The stimulus block is also commonly called a test bench. Different test benches can be used to thoroughly test the design block.

Two styles of stimulus application are possible. In the first style, the stimulus block instantiates the design block and directly drives the signals in the design block. In Figure, the stimulus block becomes the top-level block. It manipulates signals clk and reset, and it checks and displays output signal q. The second style of applying stimulus is to instantiate both the stimulus and design blocks in a top-level dummy module. The stimulus block interacts with the design block only through the interface. This style of applying stimulus is shown in Figure . The stimulus module drives the signals d_clk and d_reset, which are connected to the signals clk and reset in the design block. It also checks and displays signal c_q, which is connected to the signal q in the design block. The function of top-level block is simply to instantiate the design and stimulus blocks.

Design Block

We use a top-down design methodology. First, we write the Verilog description of the top-level design block

Example 2-3 Ripple Carry Counter Top Block

```
module ripple_carry_counter(q, clk, reset);
output [3:0] q;
input clk, reset;
//4 instances of the module T_FF are created.
T_FF tff0(q[0],clk, reset);
```

```
T_FF tff1(q[1],q[0], reset);
T_FF tff2(q[2],q[1], reset);
T_FF tff3(q[3],q[2], reset);
endmodule
```

In the above module, four instances of the module T_FF (T-flipflop) are used. Therefore, we must now define (Example 2-4) the internals of the module T_FF, which was shown in Figure 2-4.

Example 2-4 Flipflop T_FF

```
module T_FF(q, clk, reset);
output q;
input clk, reset;
wire d;
D_FF dff0(q, d, clk, reset);
not n1(d, q); // not is a Verilog-provided primitive. case sensitive
endmodule
```

Since T_FF instantiates D_FF, we must now define (Example 2-5) the internals of 35

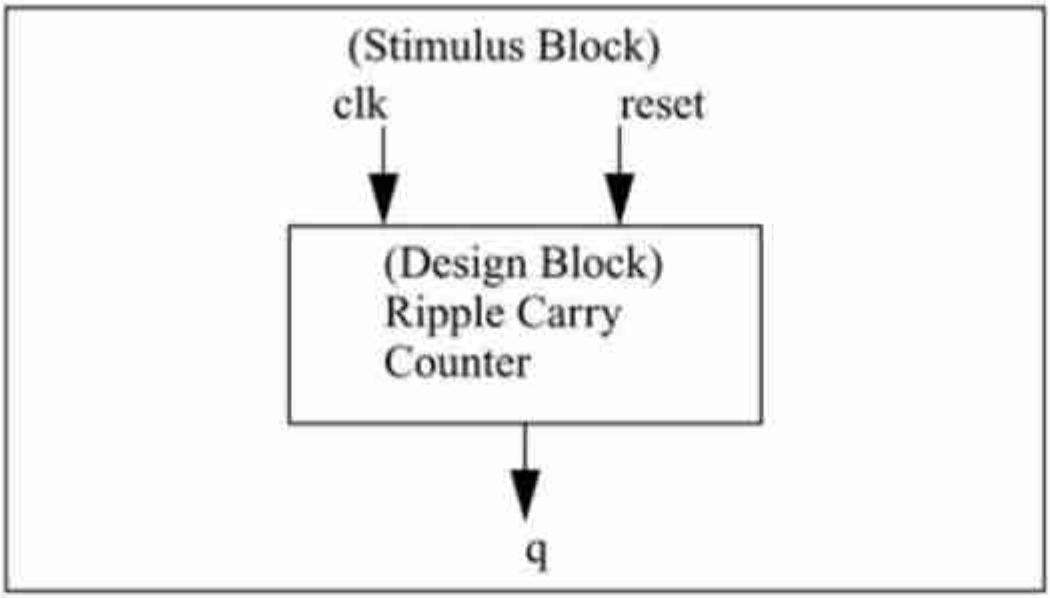
---

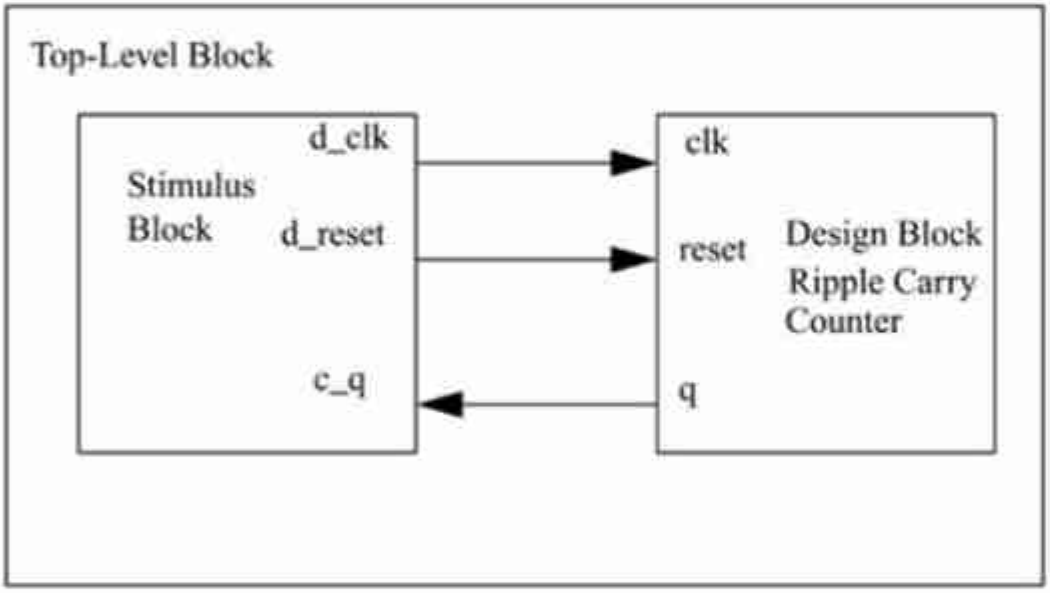
module D_FF. We assume asynchronous reset for the D_FFF.

Example 2-5 Flipflop D_F

```
// module D_FF with synchronous reset
module D_FF(q, d, clk, reset);
output q;
input d, clk, reset;
reg q;
// Lots of new constructs.  Ignore the functionality of the
// constructs.
// Concentrate on how the design block is built in a top-down fashion.
always @(posedge reset or negedge clk)
if (reset)
   q <= 1'b0;
else
   q <= d;
endmodule
```

All modules have been defined down to the lowest-level leaf cells in the design methodology. The design block is now complete.

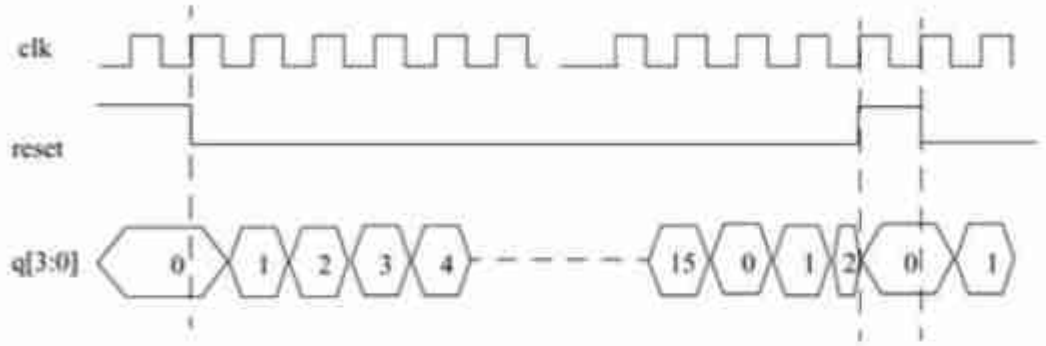


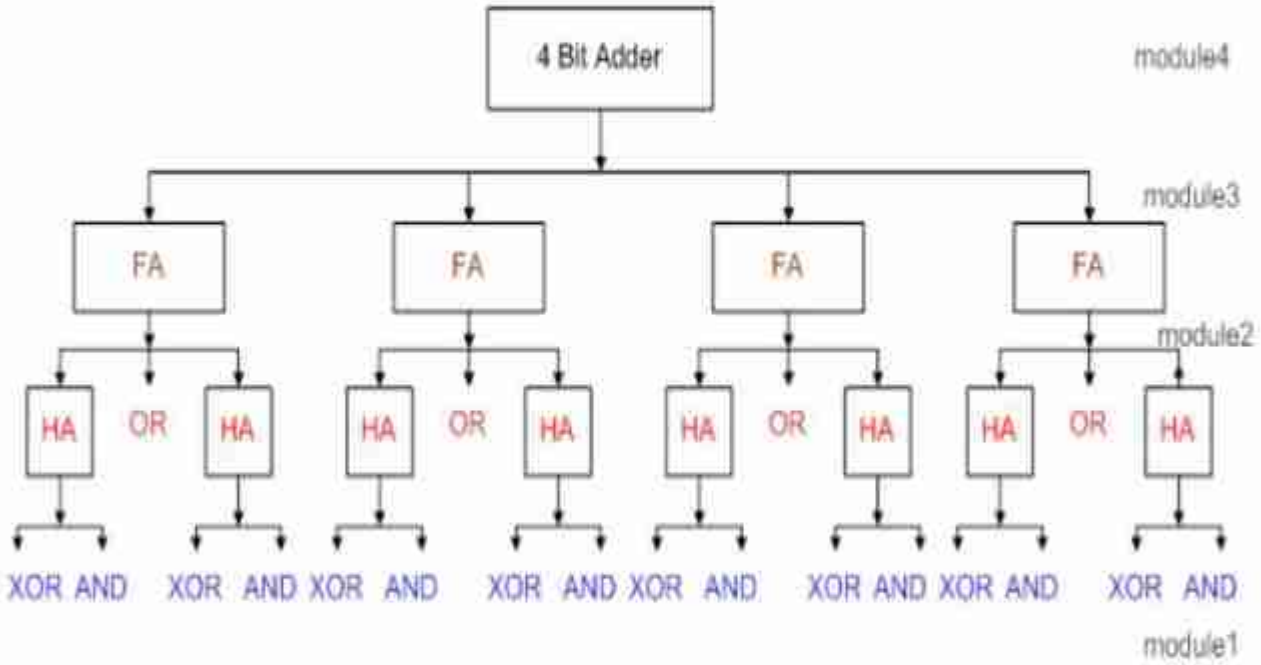


We must now write the stimulus block to check if the ripple carry counter design is functioning correctly. In this case, we must control the signals clk and reset so that the regular function of the ripple carry counter and the asynchronous reset mechanism are both tested. We use the waveforms shown in Figure 2-8 to test the design. Waveforms for clk, reset, and 4-bit output q are shown. The cycle time for clk is 10 units; the reset signal stays up from time 0 to 15 and then goes up again from time 195 to 205. Output q counts from 0 to 15.

We are now ready to write the stimulus block (see Example 2-6) that will create the above waveforms. We will use the stimulus style shown in Figure 2-6. Do not worry about the Verilog syntax at this point. Simply concentrate on how the design block is instantiated in the stimulus block.

Example 2-6 Stimulus Block






```
module stimulus;
reg clk;
reg reset;
wire[3:0] q;
// instantiate the design block
ripple_carry_counter r1(q, clk, reset);
// Control the clk signal that drives the design block. Cycle time = 10
initial
  clk = 1'b0; //set clk to 0
always
  #5 clk = ~clk; //toggle clk every 5 time units
```

```
// Control the reset signal that drives the design block
// reset is asserted from 0 to 20 and from 200 to 220.
initial
begin
  reset = 1'b1;
  #15 reset = 1'b0;
  #180 reset = 1'b1;
  #10 reset = 1'b0;
  #20 $finish; //terminate the simulation
end
// Monitor the outputs
initial
  $monitor($time, " Output q = %d", q);
endmodule
```

Once the stimulus block is completed, we are ready to run the simulation and verify the functional correctness of the design block. The output obtained when stimulus and design blocks are simulated is shown in Example 2-7.

Basic Concepts

Lexical conventions, data types, system tasks, compiler directives. (Text1)

Modules and Ports

Module definition, port declaration, connecting ports, hierarchical name referencing. (Text1)

Module definition, port declaration, connecting ports, hierarchical name referencing. (Text1)

Lexical conventions

The basic lexical conventions used by Verilog HDL are similar to those in the C programming language. Verilog contains a stream of tokens. Tokens can be comments, delimiters, numbers, strings, identifiers, and keywords. Verilog HDL is a case-sensitive language. All keywords are in lowercase.

Whitespace

Blank spaces (\b) , tabs (\t) and newlines (\n) comprise the whitespace. Whitespace is ignored by Verilog except when it separates tokens. Whitespace is not ignored in strings.

Comments

Comments can be inserted in the code for readability and documentation. There are two ways to write comments. A one-line comment starts with "//". Verilog skips from that point to the end of line. A multiple-line comment starts with "/*" and ends with "*/". Multiple-line comments cannot be nested. However, one-line comments can be embedded in multiple-line comments.

```
a = b && c; // This is a one-line comment
/* This is a multiple line
   comment */
/* This is /* an illegal */ comment */
/* This is //a legal comment */
```

Operators

Operators are of three types: unary, binary, and ternary. Unary operators precede the operand. Binary operators appear between two operands. Ternary operators have two separate operators that separate three operands.

```
a = ~ b; // ~ is a unary operator. b is the operand
a = b && c; // && is a binary operator. b and c are operands
a = b ? c : d; // ?: is a ternary operator. b, c and d are operands
```

Number Specification

There are two types of number specification in Verilog: sized and unsized.

Sized numbers

Sized numbers are represented as <size> '<base format> <number>.

<size> is written only in decimal and specifies the number of bits in the number. Legal base formats are decimal ('d or 'D), hexadecimal ('h or 'H), binary ('b or 'B) and octal ('o or 'O). The number is specified as consecutive digits from 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f. Only a subset of these digits is legal for a particular base. Uppercase letters are legal for number specification.

4'b1111 // This is a 4-bit binary number

12'habc // This is a 12-bit hexadecimal number

16'd255 // This is a 16-bit decimal number.

Unsized numbers

Numbers that are specified without a <base format> specification are decimal numbers by default. Numbers that are written without a <size> specification have a default number of bits that is simulator- and machine-specific (must be at least 32).

23456 // This is a 32-bit

'hc3 // This is a 32-bit

'o21 // This is a 32-bit

X or Z values

decimal number by default

hexadecimal number

octal number

Verilog has two symbols for unknown and high impedance values. These values are very important for modeling real circuits. An unknown value is denoted by an x. A high impedance value is denoted by z.

12'h13x // This is a 12-bit hex number; 4 least significant bits

unknown

6'hx // This is a 6-bit hex number

32'bz // This is a 32-bit high impedance number

An x or z sets four bits for a number in the hexadecimal base, three bits for a number in the octal base, and one bit for a number in the binary base. If the most significant bit of a number is 0, x, or z, the number is automatically extended to fill the most significant bits, respectively, with 0, x, or z. This makes it easy to assign x or z to whole vector. If the most significant digit is 1, then it is also zero extended.

Negative numbers

Negative numbers can be specified by putting a minus sign before the size for a constant number. Size constants are always positive. It is illegal to have a minus sign between <base format> and <number>. An optional signed specifier can be added for signed arithmetic.

-6'd3 // 8-bit negative number stored as 2's complement of 3

-6'sd3 // Used for performing signed integer math

4'd-2 // Illegal specification

Underscore characters and question marks

An underscore character "_" is allowed anywhere in a number except the first character. Underscore characters are allowed only to improve readability of numbers and are ignored by Verilog. A question mark "?" is the Verilog HDL alternative for z in the context of numbers. The ? is used to enhance readability in the casex and casez statements

Strings

A string is a sequence of characters that are enclosed by double quotes. The restriction on a string is that it must be contained on a single line, that is, without a carriage return. It cannot be on multiple lines. Strings are treated as a sequence of one-byte ASCII values.

"Hello Verilog World" // is a string

"a / b" // is a string

3.1.6 Identifiers and Keywords

Keywords are special identifiers reserved to define the language constructs. Keywords are in lowercase. A list of all keywords in Verilog is contained in Appendix C, List of Keywords, System Tasks, and Compiler Directives.

Identifiers are names given to objects so that they can be referenced in the design. Identifiers are made up of alphanumeric characters, the underscore (_), or the dollar sign ($). Identifiers are case sensitive. Identifiers start with an alphabetic character or an underscore. They cannot start with a digit or a $ sign (The $ sign as the first character is reserved for system tasks, which are explained later in the book).

reg value; // reg is a keyword; value is an identifier

input clk; // input is a keyword, clk is an identifier

3.1.7 Escaped Identifiers

Escaped identifiers begin with the backslash (\ ) character and end with whitespace (space, tab, or newline). All characters between backslash and whitespace are processed literally. Any printable ASCII character can be included in escaped identifiers. Neither the backslash nor the terminating whitespace is considered to be a part of the identifier.

\a+b-c

my_name

Data Types

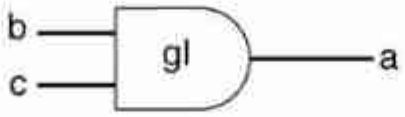
This section discusses the data types used in Verilog.

3.2.1 Value Set

Verilog supports four values and eight strengths to model the functionality of real hardware. The four value levels are listed in Table 3-1.

In addition to logic values, strength levels are often used to resolve conflicts between drivers of different strengths in digital circuits. Value levels 0 and 1 can have the strength levels listed in Table 3-2.

Nets



Nets represent connections between hardware elements. Just as in real circuits, nets have values continuously driven on them by the outputs of devices that they are connected to. In Figure 3-1 net a is connected to the output of and gate g1. Net a will continuously assume the value computed at the output of gate g1, which is b & c.

Nets are declared primarily with the keyword wire. Nets are one-bit values by default unless they are declared explicitly as vectors. The terms wire and net are often used interchangeably. The default value of a net is z (except the trireg net, which defaults to x). Nets get the output value of their drivers. If a net has no driver, it gets the value z.

```
wire a; // Declare net a for the above circuit
wire b,c; // Declare two wires b,c for the above circuit
wire d = 1'b0; // Net d is fixed to logic value 0 at declaration.
```

Registers

Registers represent data storage elements. Registers retain value until another value is placed onto them. Do not confuse the term registers in Verilog with hardware registers built from edge-triggered flipflops in real circuits. In Verilog, the term register merely means a variable that can hold a value. Unlike a net, a register does not need a driver. Verilog registers do not need a clock as hardware registers do. Values of registers can be changed anytime in a simulation by assigning a new value to the register.

Register data types are commonly declared by the keyword reg.

Example 3-1 Example of Register

```
reg reset; // declare a variable reset that can hold its value
initial // this construct will be discussed later
begin
```

```
 reset = 1'b1; //initialize reset to 1 to reset the digital circuit.
 #100 reset = 1'b0; // after 100 time units reset is deasserted.
end
```

Example 3-2 Signed Register Declaration

```
reg signed [63:0] m; // 64 bit signed value
integer i; // 32 bit signed value
```

Vectors

Nets or reg data types can be declared as vectors (multiple bit widths). If bit width is not specified, the default is scalar (1-bit).

```
wire a; // scalar net variable, default
wire [7:0] bus; // 8-bit  bus
wire [31:0] busA,busB,busC; // 3 buses of 32-bit width.
reg clock; // scalar register, default
reg [0:40] virtual_addr; // Vector register, virtual address 41 bits
wide
```

Vectors can be declared at [high# : low#] or [low# : high#], but the left number in the squared brackets is always the most significant bit of the vector. In the example shown above, bit 0 is the most significant bit of vector virtual_addr.

Vector Part Select

For the vector declarations shown above, it is possible to address bits or parts of vectors.

```
busA[7] // bit # 7 of vector busA
bus[2:0] // Three least significant bits of vector bus,
 // using bus[0:2] is illegal because the significant bit should
 // always be on the left of a range specification
virtual_addr[0:1] // Two most significant bits of vector virtual_addr
```

Variable Vector Part Select

Another ability provided in Verilog HDl is to have variable part selects of a vector. This allows part selects to be put in for loops to select various parts of the vector. There are two special part-select operators:

[<starting_bit>+:width] - part-select increments from starting bit [<starting_bit>-:width] - part-select decrements from starting bit

The starting bit of the part select can be varied, but the width has to be constant. The following example shows the use of variable vector part select:

```
reg [255:0] data1; //Little endian notation
```

```
reg [0:255] data2; //Big endian notation
reg [7:0] byte;
//Using a variable part select, one can choose parts
byte = data1[31-:8]; //starting bit = 31, width =8 => data[31:24]
byte = data1[24+:8]; //starting bit = 24, width =8 => data[31:24]
byte = data2[31-:8]; //starting bit = 31, width =8 => data[24:31]
byte = data2[24+:8]; //starting bit = 24, width =8 => data[24:31]
//The starting bit can also be a variable. The width has
//to be constant. Therefore, one can use the variable part select //in a loop to select all bytes of the
vector.
for (j=0; j<=31; j=j+1)
   byte = data1[(j*8)+:8]; //Sequence is [7:0], [15:8]... [255:248]
//Can initialize a part of the vector
data1[(byteNum*8)+:8] = 8'b0; //If byteNum = 1, clear 8 bits [15:8]
```

3.2.5 Integer , Real, and Time Register Data Types

Integer, real, and time register data types are supported in Verilog.

Integer

An integer is a general purpose register data type used for manipulating quantities. Integers are declared by the keyword integer. Although it is possible to use reg as a general-purpose variable, it is more convenient to declare an integer variable for purposes such as counting. The default width for an integer is the host-machine word size, which is implementation-specific but is at least 32 bits. Registers declared as data type reg store values as unsigned quantities, whereas integers store values as signed quantities.

```
integer counter; // general purpose variable used as a counter.
initial
   counter = -1; // A negative one is stored in the counter
```

Real

Real number constants and real register data types are declared with the keyword real. They can be specified in decimal notation (e.g., 3.14) or in scientific notation (e.g., 3e6, which is 3 x 106). Real numbers cannot have a range declaration, and their default value is 0. When a real value is assigned to an integer, the real number is rounded off to the nearest integer.

```
real delta; // Define a real variable called delta
initial
begin
   delta = 4e10; // delta is assigned in scientific notation
```

```
   delta = 2.13; // delta is assigned a value 2.13
end
integer i; // Define an integer i
initial
   i = delta; // i gets the value 2 (rounded value of 2.13)
```

Time

Verilog simulation is done with respect to simulation time. A special time register data type is used in Verilog to store simulation time. A time variable is declared with the keyword time. The width for time register data types is implementation-specific but is at least 64 bits.The system function $time is invoked to get the current simulation time.

```
time save_sim_time; // Define a time variable save_sim_time
initial
   save_sim_time = $time; // Save the current simulation time
```

Arrays

Arrays are allowed in Verilog for reg, integer, time, real, realtime and vector register data types. Multi-dimensional arrays can also be declared with any number of dimensions. Arrays of nets can also be used to connect ports of generated instances. Each element of the array can be used in the same fashion as a scalar or vector net. Arrays are accessed by <array_name>[<subscript>]. For multi-dimensional arrays, indexes need to be provided for each dimension.

```
integer count[0:7]; // An array of 8 count variables
reg bool[31:0]; // Array of 32 one-bit boolean register variables
time chk_point[1:100]; // Array of 100 time checkpoint variables
reg [4:0] port_id[0:7]; // Array of 8 port_ids; each port_id is 5 bits
wide
integer matrix[4:0][0:255]; // Two dimensional array of integers
reg [63:0] array_4d [15:0][7:0][7:0][255:0]; //Four dimensional array wire [7:0] w_array2 [5:0]; //
Declare an array of 8 bit vector wire wire w_array1[7:0][5:0]; // Declare an array of single bit wires
```

It is important not to confuse arrays with net or register vectors. A vector is a single element that is n-bits wide. On the other hand, arrays are multiple elements that are 1-bit or n-bits wide.

Examples of assignments to elements of arrays discussed above are shown below:

```
count[5] = 0; // Reset 5th element of array of count variables
chk_point[100] = 0; // Reset 100th time check point value
port_id[3] = 0; // Reset 3rd element (a 5-bit value) of port_id array.
matrix[1][0] = 33559; // Set value of element indexed by [1][0] to
33559
```

```
array_4d[0][0][0][0][15:0] = 0; //Clear bits 15:0 of the register
                                //accessed by indices [0][0][0][0]
port_id = 0; // Illegal syntax - Attempt to write the entire array
matrix [1] = 0; // Illegal syntax - Attempt to write [1][0]..[1][255]
```

3.2.7 Memories

In digital simulation, one often needs to model register files, RAMs, and ROMs. Memories are modeled in Verilog simply as a one-dimensional array of registers. Each element of the array is known as an element or word and is addressed by a single array index. Each word can be one or more bits. It is important to differentiate between n 1-bit registers and one n-bit register. A particular word in memory is obtained by using the address as a memory array subscript.

```
reg mem1bit[0:1023]; // Memory mem1bit with 1K 1-bit words
reg [7:0] membyte[0:1023]; // Memory membyte with 1K 8-bit words(bytes)
membyte[511] // Fetches 1 byte word whose address is 511.
```

3.2.8 Parameters

Verilog allows constants to be defined in a module by the keyword parameter. Parameters cannot be used as variables. Parameter values for each module instance can be overridden individually at compile time. This allows the module instances to be customized. This aspect is discussed later. Parameter types and sizes can also be defined.

```
parameter port_id = 5; // Defines a constant port_id
parameter cache_line_width = 256; // Constant defines width of cache
line
parameter signed [15:0] WIDTH; // Fixed sign and range for parameter
// WIDTH
```

Strings

Strings can be stored in reg. The width of the register variables must be large enough to hold the string. Each character in the string takes up 8 bits (1 byte). If the width of the register is greater than the size of the string, Verilog fills bits to the left of the string with zeros. If the register width is smaller than the string width, Verilog truncates the leftmost bits of the string. It is always safe to declare a string that is slightly wider than necessary.

```
reg [8*18:1] string_value; // Declare a variable that is 18 bytes wide
initial
  string_value = "Hello Verilog World"; // String can be stored
                                        // in variable
```

Escaped Characters	Character Displayed
\n	newline
\t	tab
%%	%
\\	\
\"	"
\ooo	Character written in 1?3 octal digits

3.3 System Tasks and Compiler Directives

In this section, we introduce two special concepts used in Verilog: system tasks and compiler directives.

3.3.1 System Tasks

Verilog provides standard system tasks for certain routine operations. All system tasks appear in the form $<keyword>. Operations such as displaying on the screen, monitoring values of nets, stopping, and finishing are done by system tasks. We will discuss only the most useful system tasks. Other tasks are listed in Verilog manuals provided by your simulator vendor or in the IEEE Standard Verilog Hardware Description Language specification.

Displaying information

$display is the main system task for displaying values of variables or strings or expressions. This is one of the most useful tasks in Verilog.

Usage: $display(p1, p2, p3,....., pn);

p1, p2, p3,..., pn can be quoted strings or variables or expressions. The format of $display is very similar to printf in C. A $display inserts a newline at the end of the string by default. A $display without any arguments produces a newline.

Monitoring information

Verilog provides a mechanism to monitor a signal when its value changes. This facility is provided by the $monitor task.

Usage: $monitor(p1,p2,p3,....,pn);

Strength Level	Type	Degree
supply	Driving	strongest
strong	Driving	
pull	riving	
large	Storage	
weak	Driving	
medium	Storage	
small	Storage	
highz	High Impedance	weakest

The parameters p1, p2, ... , pn can be variables, signal names, or quoted strings. A format similar to the $display task is used in the $monitor task. $monitor continuously monitors the values of the variables or signals specified in the parameter list and displays all parameters in the list whenever the value of any one variable or signal changes. Unlike $display, $monitor needs to be invoked only once. Only one monitoring list can be active at a time. If there is more than one $monitor statement in your simulation, the last $monitor statement will be the active statement. The earlier $monitor statements will be overridden.

Two tasks are used to switch monitoring on and off.

Compiler Directives

Compiler directives are provided in Verilog. All compiler directives are defined by using the `<keyword> construct. We deal with the two most useful compiler directives.

`define

The `define directive is used to define text macros in Verilog (see Example 3-7). The Verilog compiler substitutes the text of the macro wherever it encounters a `<macro_name>. This is similar to the #define construct in C. The defined constants or text macros are used in the Verilog code by preceding them with a ` (back tick).

`include

The `include directive allows you to include entire contents of a Verilog source file in another Verilog file during compilation. This works similarly to the #include in the C programming language.

Two other directives, `ifdef and `timescale, are used frequently.

Modules

We shall study the internals of modules and concentrate on how modules are defined and instantiated. A module definition always begins with the keyword module. The module name, port list, port declarations, and optional parameters must come first in a module definition. Port list and port declarations are present only if the module has any ports to interact with the external environment.The five components within a module are: variable declarations, dataflow statements, instantiation of lower modules, behavioral blocks, and tasks or functions.

To understand the components of a module shown above, let us consider a simple example of an SR latch, as shown in Figure 4-2.

The SR latch has S and R as the input ports and Q and Qbar as the output ports. The SR latch and its stimulus can be modeled as shown in Example 4-1.

Example 4-1 Components of SR Latch

```
// This example illustrates the different components of a module
// Module name and port list
// SR_latch module
module SR_latch(Q, Qbar, Sbar, Rbar);
//Port declarations
output Q, Qbar;
input Sbar, Rbar;
// Instantiate lower-level modules
// In this case, instantiate Verilog primitive nand gates
// Note, how the wires are connected in a cross-coupled fashion.
nand n1(Q, Sbar, Qbar);
nand n2(Qbar, Rbar, Q);
// endmodule statement
endmodule
// Module name and port list
// Stimulus module
module Top;
// Declarations of wire, reg, and other variables
wire q, qbar;
reg set, reset;
// Instantiate lower-level modules
// In this case, instantiate SR_latch
// Feed inverted set and reset signals to the SR latch
SR_latch m1(q, qbar, ~set, ~reset);
// Behavioral block, initial
initial
begin
  $monitor($time, " set = %b, reset= %b, q= %b\n",set,reset,q);
  set = 0; reset = 0;
  #5 reset = 1;
```

#5 reset = 0;

#5 set = 1; end

// endmodule statement

Endmodule

```
//Display the string in quotes
$display("Hello Verilog World");
-- Hello Verilog World

//Display value of current simulation time 230
$display($time);
-- 230

//Display value of 41-bit virtual address 1fe0000001c at time 200
reg [0:40] virtual_addr;
$display("At time %d virtual address is %h", $time, virtual_addr);
-- At time 200 virtual address is 1fe0000001c

//Display value of port_id 5 in binary
reg [4:0] port_id;
$display("ID of the port is %b", port_id);
-- ID of the port is 00101

//Display x characters
//Display value of 4-bit bus 10xx (signal contention) in binary
reg [3:0] bus;
$display("Bus value is %b", bus);
-- Bus value is 10xx

//Display the hierarchical name of instance p1 instantiated under
//the highest-level module called top. No argument is required. This
//is a useful feature)
$display("This string is displayed from %m level of hierarchy");
-- This string is displayed from top.p1 level of hierarchy


    //Display special characters, newline and %
    $display("This is a \n multiline string with a %% sign");
    -- This is a
    -- multiline string with a % sign

    //Display other special characters


//Monitor time and value of the signals clock and reset
//Clock toggles every 5 time units and reset goes down at 10 time units
initial
begin
    $monitor($time,
             " Value of signals clock = %b reset = %b", clock,reset);
end



// Stop at time 100 in the simulation and examine the results
// Finish the simulation at time 1000.
initial // to be explained later. time = 0
begin
clock = 0;
reset = 1;
#100 $stop; // This will suspend the simulation at time = 100
#900 $finish; // This will terminate the simulation at time = 1000
end
```

```
//define a text macro that defines default word size
//Used as 'WORD_SIZE in the code
`define WORD_SIZE 32

//define an alias. A $stop will be substituted wherever `S appears
`define S $stop;

//define a frequently used text string
`define WORD_REG reg [31:0]
// you can then define a 32-bit register as `WORD_REG reg32;


 // Include the file header.v, which contains declarations in the
 // main verilog file design.v.
 `include header.v
 ...
 ...
 <Verilog code in file design.v>
 ...
 ...
```

Notice the following characteristics about the modules defined above:

- In the SR latch definition above , notice that all components described in Figure 4-1 need not be present in a module. We do not find variable declarations, dataflow (assign) statements, or behavioral blocks (always or initial).

- However, the stimulus block for the SR latch contains module name, wire, reg, and variable declarations, instantiation of lower level modules, behavioral block (initial), and endmodule statement but does not contain port list, port declarations, and data flow (assign) statements.

- Thus, all parts except module, module name, and endmodule are optional and can be mixed and matched as per design needs.

. **Ports**

Ports provide the interface by which a module can communicate with its environment. For example, the input/output pins of an IC chip are its ports. The environment can interact with the module only through its ports. The internals of the module are not visible to the environment. This provides a very powerful flexibility to the designer. The internals of the module can be changed without affecting the environment as long as the interface is not modified. Ports are also referred to as terminals.

4.2.1 List of Ports

A module definition contains an optional list of ports. If the module does not exchange any signals with the environment, there are no ports in the list. Consider a 4-bit full adder that is instantiated inside a top-level module Top. The diagram for the input/output ports is shown in Figure 4-3.

Figure 4-3. I/O Ports for Top and Full Adder

Notice that in the above figure, the module Top is a top-level module. The module fulladd4 is instantiated below Top. The module fulladd4 takes input on ports a, b, and c_in and produces an output on ports sum and c_out. Thus, module fulladd4 performs an addition for its environment. The

module Top is a top-level module in the simulation and does not need to pass signals to or receive signals from the environment. Thus, it does not have a list of ports. The module names and port lists for both module declarations in Verilog are as shown in Example 4-2.

Example 4-2 List of Ports

```
module fulladd4(sum, c_out, a, b, c_in); //Module with a list of ports
module Top; // No list of ports, top-level module in simulation
```

4.2.2 Port Declaration

All ports in the list of ports must be declared in the module. Ports can be declared as follows:

Each port in the port list is defined as input, output, or inout, based on the direction of the port signal.

Example 4-3 Port Declarations

```
module fulladd4(sum, c_out, a, b, c_in);
//Begin port declarations section
output[3:0] sum;
output c_cout;
input [3:0] a, b;
input c_in;
//End port declarations section
...
<module internals>
...
endmodule
```

Note that all port declarations are implicitly declared as wire in Verilog. Thus, if a port is intended to be a wire, it is sufficient to declare it as output, input, or inout. Input or inout ports are normally declared as wires.

Example 4-4 Port Declarations for DFF

```
module DFF(q, d, clk, reset);
output q;
reg q; // Output port q holds value; therefore it is declared as reg. input d, clk, reset;
...
...
endmodule
```

Ports of the type input and inout cannot be declared as reg because reg variables store values and input ports should not store values but simply reflect the changes in the external signals they are connected to.

Example 4-5 ANSI C Style Port Declaration Syntax

```
module fulladd4(output reg [3:0] sum,
                output reg c_out,
                input [3:0] a, b, //wire by default
                input c_in); //wire by default
...
<module internals>
...
endmodule
```

Port Connection Rules

One can visualize a port as consisting of two units, one unit that is internal to the module and another that is external to the module. The internal and external units are connected. There are rules governing port connections when modules are instantiated within other modules. The Verilog simulator complains if any port connection rules are violated. These rules are summarized in Figure 4-4.

Inputs

Internally, input ports must always be of the type net. Externally, the inputs can be connected to a variable which is a reg or a net.

Outputs

Internally, outputs ports can be of the type reg or net. Externally, outputs must always be connected to a net. They cannot be connected to a reg.

Inouts

Internally, inout ports must always be of the type net. Externally, inout ports must always be connected to a net.

Width matching

It is legal to connect internal and external items of different sizes when making inter- module port connections. However, a warning is typically issued that the widths do not match.

Unconnected ports

Verilog allows ports to remain unconnected. For example, certain output ports might be simply for debugging, and you might not be interested in connecting them to the external signals. You can let a port remain unconnected by instantiating a module as shown below.

```
fulladd4 fa0(SUM, , A, B, C_IN); // Output port c_out is unconnected
```

Example of illegal port connection

To illustrate port connection rules, assume that the module fulladd4 in Example 4-3 is instantiated in the stimulus block Top. Example 4-6 shows an illegal port connection.

Example 4-6 Illegal Port Connection

```
module Top;
//Declare connection variables
reg [3:0]A,B;
reg C_IN;
reg [3:0] SUM;
wire C_OUT;
   //Instantiate fulladd4, call it fa0
   fulladd4 fa0(SUM, C_OUT, A, B, C_IN);
   //Illegal connection because output port sum in module fulladd4
   //is connected to a register variable SUM in module Top.
   .
   .
   <stimulus>
   .
   .
endmodule
```

This problem is rectified if the variable SUM is declared as a net (wire).

Connecting Ports to External Signals

There are two methods of making connections between signals specified in the module instantiation and the ports in a module definition. These two methods cannot be mixed. These methods are discussed in the following sections.

Connecting by ordered list

Connecting by ordered list is the most intuitive method for most beginners. The signals to be connected must appear in the module instantiation in the same order as the ports in the port list in the module definition. To connect signals in module Top by ordered list, the Verilog code is shown in Example 4-7. Notice that the external signals SUM, C_OUT, A, B, and C_IN appear in exactly the same order as the ports sum, c_out, a, b, and c_in in module definition of fulladd4.

Example 4-7 Connection by Ordered List

```
module Top;
//Declare connection variables
reg [3:0]A,B;
reg C_IN;
wire [3:0] SUM;
wire C_OUT;
```

```
  //Instantiate fulladd4, call it fa_ordered.
  //Signals are connected to ports in order (by position)
  fulladd4 fa_ordered(SUM, C_OUT, A, B, C_IN);
  ...
  <stimulus>
  ...
endmodule
module fulladd4(sum, c_out, a, b, c_in);
output[3:0] sum;
output c_cout;
input [3:0] a, b;
input c_in;
  ...
  <module internals>
  ...
endmodule
```

Connecting ports by name

For large designs where modules have, say, 50 ports, remembering the order of the ports in the module definition is impractical and error-prone. Verilog provides the capability to connect external signals to ports by the port names, rather than by position.

Hierarchical Names

We described earlier how Verilog supports a hierarchical design methodology. Every module instance, signal, or variable is defined with an identifier. A particular identifier has a unique place in the design hierarchy. Hierarchical name referencing allows us to denote every identifier in the design hierarchy with a unique name.

The top-level module is called the root module because it is not instantiated anywhere. It is the starting point. To assign a unique name to an identifier, start from the top-level module and trace the path along the design hierarchy to the desired identifier.

Continuous assignments, delay specification, expressions, operators, operands, operator types. (Text1)

With gate densities on chips increasing rapidly, dataflow modeling has assumed great importance. No longer can companies devote engineering resources to handcrafting entire designs with gates. Currently, automated tools are used to create a gate-level circuit from a dataflow design description. This process is called logic synthesis. Dataflow modeling has become a popular design approach as logic synthesis tools have become sophisticated.

Continuous Assignments

A continuous assignment is the most basic statement in dataflow modeling, used to drive a value onto a net. This assignment replaces gates in the description of the circuit and describes the circuit at a higher level of abstraction. The assignment statement starts with the keyword assign. The syntax of an assign statement is as follows:

continuous_assign ::= assign [drive_strength] [delay3]
list_of_net_assignments ;
list_of_net_assignments ::= net_assignment { , net_assignment }
net_assignment ::= net_lvalue = expression

Continuous assignments have the following characteristics:

1. The left hand side of an assignment must always be a scalar or vector net or a concatenation of scalar and vector nets. It cannot be a scalar or vector register.
2. Continuous assignments are always active. The assignment expression is evaluated as soon as one of the right-hand-side operands changes and the value is assigned to the left-hand-side net.
3. The operands on the right-hand side can be registers or nets or function calls. Registers or nets can be scalars or vectors.
4. Delay values can be specified for assignments in terms of time units. Delay values are used to control the time when a net is assigned the evaluated value. This feature is similar to specifying delays for gates. It is very useful in modeling timing behavior in real circuits.

Examples of Continuous Assignment

```
// Continuous assign. out is a net. i1 and i2 are nets.
assign out = i1 & i2;
// Continuous assign for vector nets. addr is a 16-bit vector net
```

```
// addr1 and addr2 are 16-bit vector registers.
assign addr[15:0] = addr1_bits[15:0] ^ addr2_bits[15:0];
// Concatenation. Left-hand side is a concatenation of a scalar
// net and a vector net.
assign {c_out, sum[3:0]} = a[3:0] + b[3:0] + c_in;
```

We now discuss a shorthand method of placing a continuous assignment on a net.

Implicit Continuous Assignment

Instead of declaring a net and then writing a continuous assignment on the net, Verilog provides a shortcut by which a continuous assignment can be placed on a net when it is declared. There can be only one implicit declaration assignment per net because a net is declared only once.

In the example below, an implicit continuous assignment is contrasted with a regular continuous assignment.

```
//Regular continuous assignment
wire out;
assign out = in1 & in2;
//Same effect is achieved by an implicit continuous assignment
wire out = in1 & in2;
```

Implicit Net Declaration

If a signal name is used to the left of the continuous assignment, an implicit net declaration will be inferred for that signal name. If the net is connected to a module port, the width of the inferred net is equal to the width of the module port.

```
// Continuous assign. out is a net.
wire i1, i2;
assign out = i1 & i2; //Note that out was not declared as a wire
                      //but an implicit wire declaration for out
                      //is done by the simulator
```

Delays

Delay values control the time between the change in a right-hand-side operand and when the new value is assigned to the left-hand side. Three ways of specifying delays in continuous assignment statements are regular assignment delay, implicit continuous assignment delay, and net declaration delay.

Regular Assignment Delay

The first method is to assign a delay value in a continuous assignment statement. The delay value is specified after the keyword assign. Any change in values of in1 or in2 will result in a delay of 10 time units before recomputation of the expression in1 & in2, and the result will be assigned to out. If in1

or in2 changes value again before 10 time units when the result propagates to out, the values of in1 and in2 at the time of recomputation are considered. This property is called inertial delay. An input pulse that is shorter than the delay of the assignment statement does not propagate to the output.

assign #10 out = in1 & in2; // Delay in a continuous assign

The waveform in Figure 2.1 is generated by simulating the above assign statement. It shows the delay on signal out. Note the following change:

1. When signals in1 and in2 go high at time 20, out goes to a high 10 time units later (time = 30).

2. When in1 goes low at 60, out changes to low at 70.

3. However, in1 changes to high at 80, but it goes down to low before 10 time units have elapsed.

4. Hence, at the time of recomputation, 10 units after time 80, in1 is 0. Thus, out gets the value 0. A pulse of width less than the specified assignment delay is not propagated to the output.

Implicit Continuous Assignment Delay

An equivalent method is to use an implicit continuous assignment to specify both a delay and an assignment on the net.

```
//implicit continuous assignment delay
wire #10 out = in1 & in2;
//same as
wire out;
assign #10 out = in1 & in2;
```

The declaration above has the same effect as defining a wire out and declaring a continuous assignment on out.

Net Declaration Delay

A delay can be specified on a net when it is declared without putting a continuous assignment on the net. If a delay is specified on a net out, then any value change applied to the net out is delayed accordingly. Net declaration delays can also be used in gate-level modeling.

```
//Net Delays
wire # 10 out;
assign out = in1 & in2;
//The above statement has the same effect as the following.
```

```
wire out;
assign #10 out = in1 & in2;
```

Having discussed continuous assignments and delays, let us take a closer look at expressions, operators, and operands that are used inside continuous assignments.

Expressions, Operators, and Operands

Dataflow modeling describes the design in terms of expressions instead of primitive gates. Expressions, operators, and operands form the basis of dataflow modeling.

Expressions

Expressions are constructs that combine operators and operands to produce a result.

```
// Examples of expressions. Combines operands and operators a^b
addr1[20:17] + addr2[20:17]
in1 | in2
```

Operands

Some constructs will take only certain types of operands. Operands can be constants, integers, real numbers, nets, registers, times, bit-select (one bit of vector net or a vector register), part-select (selected bits of the vector net or register vector), and memories or function calls (functions are discussed later).

```
integer count, final_count;
final_count = count + 1;//count is an integer operand
real a, b, c;
c = a - b; //a and b are real operands
reg [15:0] reg1, reg2;
reg [3:0] reg_out;
reg_out = reg1[3:0] ^ reg2[3:0];//reg1[3:0] and reg2[3:0] are
                    //part-select register operands
reg ret_value;
ret_value = calculate_parity(A, B);//calculate_parity is a
                    //function type operand
```

Operators

Operators act on the operands to produce desired results. Verilog provides various types of operators.

```
d1 && d2 // && is an operator on operands d1 and d2
!a[0] // ! is an operator on operand a[0]
B >> 1 // >> is an operator on operands B and 1
```

Operator Types

Verilog provides many different operator types. Operators can be arithmetic, logical, relational, equality, bitwise, reduction, shift, concatenation, or conditional. Some of these operators are similar

to the operators used in the C programming language. Each operator type is denoted by a symbol.

Table 2.1 shows the complete listing of operator symbols classified by category.

Behavioral Modeling

Structured procedures, initial and always, blocking and non-blocking statements, delay control, generate statement, event control, conditional statements, multiway branching, loops, sequential and parallel blocks. (Text1)

With the increasing complexity of digital design, it has become vitally important to make wise design decisions early in a project. Designers need to be able to evaluate the trade- offs of various architectures and algorithms before they decide on the optimum architecture and algorithm to implement in hardware. Thus, architectural evaluation takes place at an algorithmic level where the designers do not necessarily think in terms of logic gates or data flow but in terms of the algorithm they wish to implement in hardware. They are more concerned about the behavior of the algorithm and its performance. Only after the high-level architecture and algorithm are finalized, do designers start focusing on building the digital circuit to implement the algorithm.

Structured Procedures

There are two structured procedure statements in Verilog: always and initial. These statements are the two most basic statements in behavioral modeling. All other behavioral statements can appear only inside these structured procedure statements.

Verilog is a concurrent programming language unlike the C programming language, which is sequential in nature. Activity flows in Verilog run in parallel rather than in sequence. Each always and initial statement represents a separate activity flow in Verilog. Each activity flow starts at simulation time 0.

7.1.1 initial Statement

All statements inside an initial statement constitute an initial block. An initial block starts at time 0, executes exactly once during a simulation, and then does not execute again. If there are multiple initial blocks, each block starts to execute concurrently at time 0. Each block finishes execution independently of other blocks.

Example 7-1 initial Statement

```
module stimulus;
reg x,y, a,b, m;
initial
   m = 1'b0; //single statement; does not need to be grouped
initial
begin
   #5 a = 1'b1; //multiple statements; need to be grouped
   #25 b = 1'b0;
end
initial
begin
#10 x = 1'b0;
   #25 y = 1'b1;
```

```
end
initial
#50 $finish;
endmodule
```

In the above example, the three initial statements start to execute in parallel at time 0. If a delay #<delay> is seen before a statement, the statement is executed <delay> time units after the current simulation time. Thus, the execution sequence of the statements inside the initial blocks will be as follows.

time	statement executed
0	m = 1'b0;
5	a = 1'b1;
10	x = 1'b0;
30	b = 1'b0;
35	y = 1'b1;
50	$finish;

The initial blocks are typically used for initialization, monitoring, waveforms and other processes that must be executed only once during the entire simulation run.

Combined Variable Declaration and Initialization

Variables can be initialized when they are declared. Example 7-2 shows such a declaration.

Example 7-2 Initial Value Assignment

```
//The clock variable is defined first
reg clock;
//The value of clock is set to 0
initial clock = 0;
//Instead of the above method, clock variable
//can be initialized at the time of declaration
//This is allowed only for variables declared
//at module level.
reg clock = 0;
```

Combined Port/Data Declaration and Initialization

The combined port/data declaration can also be combined with an initialization. Example 7-3 shows such a declaration.

Example 7-3 Combined Port/Data Declaration and Variable Initialization

```
module adder (sum, co, a, b, ci);
```

output reg [7:0] sum = 0; //Initialize 8 bit output sum

output reg

input

input

--

--

endmodule

co = 0; //Initialize 1 bit output co

[7:0] a, b;

ci;

Combined ANSI C Style Port Declaration and Initialization

ANSI C style port declaration can also be combined with an initialization. Example 7-4 shows such a declaration.

Example 7-4 Combined ANSI C Port Declaration and Variable Initialization

module adder (output reg [7:0] sum = 0, //Initialize 8 bit output

---

--

--

endmodule

7.1.2 always Statement

output reg

input

input

);

co = 0, //Initialize 1 bit output co

[7:0] a, b,

ci

All behavioral statements inside an always statement constitute an always block. The always statement starts at time 0 and executes the statements in the always block continuously in a looping fashion. This statement is used to model a block of activity that is repeated continuously in a digital circuit. Example 7-5 illustrates one method to model a clock generator in Verilog.

Example 7-5 always Statement

module clock_gen (output reg clock);

//Initialize clock at time zero

initial

```
    clock = 1'b0;
//Toggle clock every half-cycle (time period = 20)
always
#10 clock = ~clock;
initial
    #1000 $finish;
endmodule
```

C programmers might draw an analogy between the always block and an infinite loop. But hardware designers tend to view it as a continuously repeated activity in a digital circuit starting from power on. The activity is stopped only by power off ($finish) or by an interrupt ($stop).

Procedural Assignments

Procedural assignments update values of reg, integer, real, or time variables. The value placed on a variable will remain unchanged until another procedural assignment updates the variable with a different value. These are unlike continuous assignments discussed in Chapter 6, Dataflow Modeling, where one assignment statement can cause the value of the right-hand-side expression to be continuously placed onto the left-hand-side net. The syntax for the simplest form of procedural assignment is shown below.

```
assignment ::= variable_lvalue = [ delay_or_event_control ]
        expression
```

The left-hand side of a procedural assignment <lvalue> can be one of the following:

- A reg, integer, real, or time register variable or a memory element
- A bit select of these variables (e.g., addr[0])
- A part select of these variables (e.g., addr[31:16])
- A concatenation of any of the above

The right-hand side can be any expression that evaluates to a value. In behavioral modeling, all operators listed in Table 6-1 on page 96 can be used in behavioral expressions. There are two types of procedural assignment statements: blocking and nonblocking.

7.2.1 Blocking Assignments

Blocking assignment statements are executed in the order they are specified in a sequential block. A blocking assignment will not block execution of statements that follow in a parallel block. The = operator is used to specify blocking assignments.

Example 7-6 Blocking Statements

```
reg x, y, z;
reg [15:0] reg_a, reg_b;
integer count;
//All behavioral statements must be inside an initial or always block
initial
begin
to end
x = 0; y = 1; z = 1; //Scalar assignments
count = 0; //Assignment to integer variables
reg_a = 16'b0; reg_b = reg_a; //initialize vectors
#15 reg_a[2] = 1'b1; //Bit select assignment with delay
#10 reg_b[15:13] = {x, y, z} //Assign result of concatenation
                    // part select of a vector
count = count + 1; //Assignment to an integer (increment)
```

In Example 7-6, the statement y = 1 is executed only after x = 0 is executed. The behavior in a particular block is sequential in a begin-end block if blocking statements are used, because the statements can execute only in sequence. The statement count = count + 1 is executed last. The simulation times at which the statements are executed are as follows:

- All statements x = 0 through reg_b = reg_a are executed at time 0
    - Statement reg_a[2] = 0 at time = 15
    - Statement reg_b[15:13] = {x, y, z} at time = 25
    - Statement count = count + 1 at time = 25
    - Since there is a delay of 15 and 10 in the preceding statements, count = count + 1 will be executed at time = 25 units

7.2.2 Nonblocking Assignments

Nonblocking assignments allow scheduling of assignments without blocking execution of the statements that follow in a sequential block. A <= operator is used to specify nonblocking assignments. Note that this operator has the same symbol as a relational operator, less_than_equal_to. The operator <= is interpreted as a relational operator in an expression and as an assignment operator in the context of a nonblocking assignment.

Example 7-7 Nonblocking Assignments

```
reg x, y, z;
reg [15:0] reg_a, reg_b;
```

```
integer count;
//All behavioral statements must be inside an initial or always block
initial
begin
    x = 0; y = 1; z = 1; //Scalar assignments
    count = 0; //Assignment to integer variables
    reg_a = 16'b0; reg_b = reg_a; //Initialize vectors
    reg_a[2] <= #15 1'b1; //Bit select assignment with delay
    reg_b[15:13] <= #10 {x, y, z}; //Assign result of concatenation
                        //to part select of a vector
    count <= count + 1; //Assignment to an integer (increment)
end
```

In this example, the statements x = 0 through reg_b = reg_a are executed sequentially at time 0. Then the three nonblocking assignments are processed at the same simulation time.

1. reg_a[2] = 0 is scheduled to execute after 15 units (i.e., time = 15)
2. reg_b[15:13] = {x, y, z} is scheduled to execute after 10 time units (i.e., time = 10)
3. count = count + 1 is scheduled to be executed without any delay (i.e., time = 0)

Delay Controls

Various behavioral timing control constructs are available in Verilog. Timing controls provide a way to specify the simulation time at which procedural statements will execute. There are three methods of timing control: delay-based timing control, event-based timing control, and level-sensitive timing control.

7.3.1 Delay-Based Timing Control

Delay-based timing control in an expression specifies the time duration between when the statement is encountered and when it is executed. We used delay-based timing control statements when writing few modules in the preceding chapters but did not explain them in detail.

```
delay3 ::= # delay_value | # ( delay_value [ , delay_value [ ,
        delay_value ] ] )
delay2 ::= # delay_value | # ( delay_value [ , delay_value ] )
delay_value ::=
      unsigned_number
    | parameter_identifier
    | specparam_identifier
```

| mintypmax_expression

Delay-based timing control can be specified by a number, identifier, or a mintypmax_expression. There are three types of delay control for procedural assignments: regular delay control, intra-assignment delay control, and zero delay control.

Regular delay control

Regular delay control is used when a non-zero delay is specified to the left of a procedural assignment. Usage of regular delay control is shown in Example 7-10.

Example 7-10 Regular Delay Control

```
//define parameters
parameter latency = 20;
parameter delta = 2;
//define register variables
reg x, y, z, p, q;
initial
begin
    x = 0; // no delay control
    #10 y = 1; // delay control with a number. Delay execution of
                    // y = 1 by 10 units
    #latency z = 0; // Delay control with identifier. Delay of 20
units
y. end
#(latency + delta) p = 1; // Delay control with expression
#y x = x + 1; // Delay control with identifier. Take value of
#(4:5:6) q = 0; // Minimum, typical and maximum delay values.
          //Discussed in gate-level modeling chapter.
```

In Example 7-10, the execution of a procedural assignment is delayed by the number specified by the delay control. For begin-end groups, delay is always relative to time when the statement is encountered. Thus, y =1 is executed 10 units after it is encountered in the activity flow.

Intra-assignment delay control

Instead of specifying delay control to the left of the assignment, it is possible to assign a delay to the right of the assignment operator. Such delay specification alters the flow of activity in a different manner. Example 7-11 shows the contrast between intra-assignment delays and regular delays.

Example 7-11 Intra-assignment Delays

```
//define register variables
reg x, y, z;
```

```
//intra assignment delays
initial
begin
    x = 0; z = 0;
end
y = #5 x + z; //Take value of x and z at the time=0, evaluate //x + z and then wait 5 time units to
assign value
//to y.
//Equivalent method with temporary variables and regular delay control
initial
begin
    x = 0; z = 0;
z
temp_xz = x + z;
#5 y = temp_xz; //Take value of x + z at the current time and
    //store it in a temporary variable. Even though x and
    //might change between 0 and 5,
end
//the value assigned to y at time 5 is unaffected.
```

Note the difference between intra-assignment delays and regular delays. Regular delays defer the execution of the entire assignment. Intra-assignment delays compute the right- hand-side expression at the current time and defer the assignment of the computed value to the left-hand-side variable. Intra-assignment delays are like using regular delays with a temporary variable to store the current value of a right-hand-side expression.

Zero delay control

Procedural statements in different always-initial blocks may be evaluated at the same simulation time. The order of execution of these statements in different always-initial blocks is nondeterministic. Zero delay control is a method to ensure that a statement is executed last, after all other statements in that simulation time are executed. This is used to eliminate race conditions. However, if there are multiple zero delay statements, the order between them is nondeterministic. Example 7-12 illustrates zero delay control.

Example 7-12 Zero Delay Control

```
initial
begin
x = 0;
```

```
y = 0; end
initial
begin
   #0 x = 1; //zero delay control
#0 y = 1; end
```

In Example 7-12, four statements?x = 0, y = 0, x = 1, y = 1?are to be executed at simulation time 0. However, since x = 1 and y = 1 have #0, they will be executed last. Thus, at the end of time 0, x will have value 1 and y will have value 1. The order in which x = 1 and y = 1 are executed is not deterministic.

The above example was used as an illustration. However, using #0 is not a recommended practice.

Conditional Statements

Conditional statements are used for making decisions based upon certain conditions. These conditions are used to decide whether or not a statement should be executed. Keywords if and else are used for conditional statements. There are three types of conditional statements. Usage of conditional statements is shown below.

```
//Type 1 conditional statement. No else statement.
//Statement executes or does not execute.
if (<expression>) true_statement ;
//Type 2 conditional statement. One else statement
//Either true_statement or false_statement is evaluated
if (<expression>) true_statement ; else false_statement ;
//Type 3 conditional statement. Nested if-else-if.
//Choice of multiple statements. Only one is executed.
if (<expression1>) true_statement1 ;
else if (<expression2>) true_statement2 ;
else if (<expression3>) true_statement3 ;
else default_statement ;
```

The <expression> is evaluated. If it is true (1 or a non-zero value), the true_statement is executed. However, if it is false (zero) or ambiguous (x), the false_statement is executed. The <expression> can contain any operators mentioned in Table 6-1 on page 96. Each true_statement or false_statement can be a single statement or a block of multiple statements. A block must be grouped, typically by using keywords begin and end. A single statement need not be grouped.

Example 7-18 Conditional Statement Examples

```
//Type 1 statements
if(!lock) buffer = data;
```

```
if(enable) out = in;
//Type 2 statements
if (number_queued < MAX_Q_DEPTH)
begin
end else
data_queue = data;
number_queued = number_queued + 1;
$display("Queue Full. Try again");
//Type 3 statements
//Execute statements based on ALU control signal.
if (alu_control == 0)
y = x + z;
else if(alu_control == 1)
    y = x - z;
else if(alu_control == 2)
    y = x * z;
else
$display("Invalid ALU control signal");
```

Multiway Branching

The nested if-else-if can become unwieldy if there are too many alternatives. A shortcut to achieve the same result is to use the case statement.

7.5.1 case Statement

The keywords case, endcase, and default are used in the case statement..

```
case (expression)
  alternative1: statement1;
  alternative2: statement2;
  alternative3: statement3;
...
    ...
  default: default_statement;
endcase
```

Each of statement1, statement2 , default_statement can be a single statement or a block of multiple statements. A block of multiple statements must be grouped by keywords begin and end. For the first alternative that matches, the corresponding statement or block is executed. If none of the alternatives matches, the default_statement is executed. The default_statement is optional. Placing of multiple

default statements in one case statement is not allowed. The case statements can be nested. The following Verilog code implements the type 3 conditional statement in Example 7-18.

```
//Execute statements based on the ALU control signal
reg [1:0] alu_control;
...
...
case (alu_control)
  2'd0 : y = x + z;
  2'd1 : y = x - z;
  2'd2 : y = x * z;
  default : $display("Invalid ALU control signal");
endcase
```

The case statement can also act like a many-to-one multiplexer. The I/O ports are unchanged. Notice that an 8-to-1 or 16-to-1 multiplexer can also be easily implemented by case statements.

Example 7-19 4-to-1 Multiplexer with Case Statement

```
module mux4_to_1 (out, i0, i1, i2, i3, s1, s0);
// Port declarations from the I/O diagram
output out;
input i0, i1, i2, i3;
input s1, s0;
reg out;
always @(s1 or s0 or i0 or i1 or i2 or i3)
case ({s1, s0}) //Switch based on concatenation of control signals
      2'd0 : out = i0;
      2'd1 : out = i1;
      2'd2 : out = i2;
      2'd3 : out = i3;
      default: $display("Invalid control signals");
endcase
endmodule
```

The case statement compares 0, 1, x, and z values in the expression and the alternative bit for bit. If the expression and the alternative are of unequal bit width, they are zero filled to match the bit width of the widest of the expression and the alternative.

In Example 7- 20, we will define a 1-to-4 demultiplexer for which outputs are completely specified, that is, definitive results are provided even for x and z values on the select signal.

Example 7-20 Case Statement with x and z

```
module demultiplexer1_to_4 (out0, out1, out2, out3, in, s1, s0);
// Port declarations from the I/O diagram
output out0, out1, out2, out3;
reg out0, out1, out2, out3;
input in;
input s1, s0;
always @(s1 or s0 or in)
case ({s1, s0}) //Switch based on control signals
   2'b00 : begin out0 = in; out1 = 1'bz; out2 = 1'bz; out3 =
1'bz; end
2'b01 : begin out0 = 1'bz; out1 = in; out2 = 1'bz; out3 = 1'bz; end
   2'b10 : begin out0 = 1'bz; out1 = 1'bz; out2 = in; out3 =
1'bz; end
   2'b11 : begin out0 = 1'bz; out1 = 1'bz; out2 = 1'bz; out3 =
in; end
   //Account for unknown signals on select. If any select signal is x
   //then outputs are x. If any select signal is z, outputs are z.
   //If one is x and the other is z, x gets higher priority.
   2'bx0, 2'bx1, 2'bxz, 2'bxx, 2'b0x, 2'b1x, 2'bzx :
begin
         out0 = 1'bx; out1 = 1'bx; out2 = 1'bx; out3 = 1'bx;
      end
   2'bz0, 2'bz1, 2'bzz, 2'b0z, 2'b1z :
      begin
         out0 = 1'bz; out1 = 1'bz; out2 = 1'bz; out3 = 1'bz;
      end
   default: $display("Unspecified control signals");
endcase
endmodule
```

In the demultiplexer shown above, multiple input signal combinations such as 2'bz0, 2'bz1, 2,bzz, 2'b0z, and 2'b1z that cause the same block to be executed are put together with a comma (,) symbol.

7.5.2 casex, casez Keywords

There are two variations of the case statement. They are denoted by keywords, casex and casez.

- casez treats all z values in the case alternatives or the case expression as don't cares. All bit positions with z can also represented by ? in that position.

- casex treats all x and z values in the case item or the case expression as don't cares. The use of casex and casez allows comparison of only non-x or -z positions in the case expression and the case alternatives. The use of casez is similar. Only one bit is considered to determine the next state and the other bits are ignored.

Example 7-21 casex Use

reg [3:0] encoding;
- integer state;
-

casex (encoding) //logic value x represents a don't care bit.
- 4'b1xxx : next_state = 3;
- 4'bx1xx : next_state = 2;
- 4'bxx1x : next_state = 1;
-

4'bxxx1 : next_state = 0;
- default : next_state = 0;
- endcase
-

Thus, an input encoding = 4'b10xz would cause next_state = 3 to be executed.

Loops

There are four types of looping statements in Verilog: while, for, repeat, and forever. The syntax of these loops is very similar to the syntax of loops in the C programming language. All looping statements can appear only inside an initial or always block. Loops may contain delay expressions.

7.6.1 While Loop

The keyword while is used to specify this loop. The while loop executes until the while- expression is not true. If the loop is entered when the while-expression is not true, the loop is not executed at all. If multiple statements are to be executed in the loop, they must be grouped typically using keywords begin and end.

Example 7-22 While Loop

//Illustration 1: Increment count from 0 to 127. Exit at count 128.

//Display the count variable.

integer count;

```
initial
begin
end
count = 0;
while (count < 128) //Execute loop till count is 127.
                    //exit at count 128
begin
      $display("Count = %d", count);
      count = count + 1;
end
//Illustration 2: Find the first bit with a value 1 in flag (vector
variable)
'define TRUE 1'b1';
'define FALSE 1'b0;
reg [15:0] flag;
integer i; //integer to keep count
reg continue;
initial
begin
 flag = 16'b 0010_0000_0000_0000;
 i = 0;
 continue = 'TRUE;
 while((i < 16) && continue ) //Multiple conditions using operators.
 begin
  if (flag[i])
  begin
   $display("Encountered a TRUE bit at element number %d", i);
   continue = 'FALSE;
  end
i = i + 1; end
end
```

7.6.2 For Loop

The keyword for is used to specify this loop. The for loop contains three parts:

- An initial condition
- A check to see if the terminating condition is true

- A procedural assignment to change value of the control variable

The counter described in Example 7-22 can be coded as a for loop (Example 7-23). The initialization condition and the incrementing procedural assignment are included in the for loop and do not need to be specified separately. Thus, the for loop provides a more compact loop structure than the while loop. Note, however, that the while loop is more general-purpose than the for loop. The for loop cannot be used in place of the while loop in all situations.

Example 7-23 For Loop

```
integer count;
initial
for ( count=0; count < 128; count = count + 1)
        $display("Count = %d", count);
```

for loops can also be used to initialize an array or memory, as shown below.

```
//Initialize array elements
 'define MAX_STATES 32
integer state [0: 'MAX_STATES-1]; //Integer array state with elements
0:31
  integer i;
initial
  begin

     for(i = 0; i < 32; i = i + 2) //initialize all even locations with 0 state[i] = 0;

for(i = 1; i < 32; i = i + 2) //initialize all odd locations with 1 state[i] = 1;
end
```

for loops are generally used when there is a fixed beginning and end to the loop. If the loop is simply looping on a certain condition, it is better to use the while loop.

7.6.3 Repeat Loop

The keyword repeat is used for this loop. The repeat construct executes the loop a fixed number of times. A repeat construct cannot be used to loop on a general logical expression. A while loop is used for that purpose. A repeat construct must contain a number, which can be a constant, a variable or a signal value.

Example 7-24 Repeat Loop

```
//Illustration 1 : increment and display count from 0 to 127
integer count;
initial
```

```
begin
  count = 0;
  repeat(128)
  begin
     $display("Count = %d", count);
     count = count + 1;
  end
end
//Illustration 2 : Data buffer module example
//After it receives a data_start signal.
//Reads data for next 8 cycles.
module data_buffer(data_start, data, clock);
parameter cycles = 8;
input data_start;
input [15:0] data;
input clock;
reg [15:0] buffer [0:7];
integer i;
always @(posedge clock)
begin
 if(data_start) //data start signal is true
 begin
  i = 0;
  repeat(cycles) //Store data at the posedge of next 8 clock
//cycles
  begin
    @(posedge clock) buffer[i] = data; //waits till next
i = i + 1; end
end end
endmodule
```

7.6.4 Forever loop

The keyword forever is used to express this loop. The loop does not contain any expression and executes forever until the $finish task is encountered. The loop is equivalent to a while loop with an expression that always evaluates to true, e.g., while (1). A forever loop can be exited by use of the disable statement. A forever loop is typically used in conjunction with timing control constructs

Example 7-25 Forever Loop

```
//Example 1: Clock generation
//Use forever loop instead of always block
reg clock;
initial
begin
end
// posedge to latch data
clock = 1'b0;
forever #10 clock = ~clock; //Clock with period of 20 units
//Example 2: Synchronize two register values at every positive edge of
//clock
reg clock;
reg x, y;
initial
    forever @(posedge clock) x = y;
```

7.7 Sequential and Parallel Blocks

Block statements are used to group multiple statements to act together as one. In previous examples, we used keywords begin and end to group multiple statements. Thus, we used sequential blocks where the statements in the block execute one after another.

7.7.1 Block Types

There are two types of blocks: sequential blocks and parallel blocks.

Sequential blocks

The keywords begin and end are used to group statements into sequential blocks. Sequential blocks have the following characteristics:

- The statements in a sequential block are processed in the order they are specified. A statement is executed only after its preceding statement completes execution (except for nonblocking assignments with intra-assignment timing control).

- If delay or event control is specified, it is relative to the simulation time when the previous statement in the block completed execution.

  We have used numerous examples of sequential blocks in this book. Two more examples of sequential blocks are given in Example 7-26. Statements in the sequential block execute in order. In Illustration 1, the final values are x = 0, y= 1, z = 1, w = 2 at simulation time 0. In

Illustration 2, the final values are the same except that the simulation time is 35 at the end of the block.

Example 7-26 Sequential Blocks:

```
//Illustration 1: Sequential block without delay
reg x, y;
reg [1:0] z, w;
initial
begin
x = 1'b0;
y = 1'b1;
z = {x, y};
w = {y, x};
end
//Illustration 2: Sequential blocks with delay.
reg x, y;
reg [1:0] z, w;
initial
begin
end
x = 1'b0; //completes at simulation time 0
#5 y = 1'b1; //completes at simulation time 5
#10 z = {x, y}; //completes at simulation time 15
#20 w = {y, x}; //completes at simulation time 35
```

Parallel blocks

Parallel blocks, specified by keywords fork and join, provide interesting simulation features. Parallel blocks have the following characteristics:

- Statements in a parallel block are executed concurrently.
- Ordering of statements is controlled by the delay or event control assigned to each statement.
- If delay or event control is specified, it is relative to the time the block was entered.

Example 7-27 Parallel Blocks:

```
//Example 1: Parallel blocks with delay.
reg x, y;
```

```
reg [1:0] z, w;
    initial fork
    join
```

```
x = 1'b0; //completes at simulation time 0
#5 y = 1'b1; //completes at simulation time 5
#10 z = {x, y}; //completes at simulation time 10
#20 w = {y, x}; //completes at simulation time 20
```

Parallel blocks provide a mechanism to execute statements in parallel. Race conditions have been deliberately introduced in this example. All statements start at simulation time 0. The order in which the statements will execute is not known. Variables z and w will get values 1 and 2 if x = 1'b0 and y = 1'b1 execute first. Variables z and w will get values 2'bxx and 2'bxx if x = 1'b0 and y = 1'b1 execute last. Thus, the result of z and w is nondeterministic and dependent on the simulator implementation.

```
//Parallel blocks with deliberate race condition
reg x, y;
reg [1:0] z, w;
initial fork
join
x = 1'b0;
y = 1'b1;
z = {x, y};
w = {y, x};
```

The keyword fork can be viewed as splitting a single flow into independent flows. The keyword join can be seen as joining the independent flows back into a single flow. Independent flows operate concurrently.

7.7.2 Special Features of Blocks

We discuss three special features available with block statements: nested blocks, named blocks, and disabling of named blocks.

Nested blocks

Blocks can be nested. Sequential and parallel blocks can be mixed, as shown in Example 7-28.

Example 7-28 Nested Blocks

```
//Nested blocks
initial
begin
x = 1'b0;
```

```
end
fork
join
#20 w = {y, x};
#5 y = 1'b1;
#10 z = {x, y};
```

Named blocks

Blocks can be given names.

• Local variables can be declared for the named block.

• Named blocks are a part of the design hierarchy. Variables in a named block can be accessed by using hierarchical name referencing.

• Named blocks can be disabled, i.e., their execution can be stopped. Example 7-29 shows naming of blocks and hierarchical naming of blocks.

Example 7-29 Named Blocks

```
//Named blocks
module top;
initial
begin: block1 //sequential block named block1
integer i; //integer i is static and local to block1
... ... end
// can be accessed by hierarchical name, top.block1.i
initial
fork: block2 //parallel block named block2
reg i; // register i is static and local to block2
... ... join
// can be accessed by hierarchical name, top.block2.i
```

Disabling named blocks

The keyword disable provides a way to terminate the execution of a named block. disable can be used to get out of loops, handle error conditions, or control execution of pieces of code, based on a control signal. Disabling a block causes the execution control to be passed to the statement immediately succeeding the block. The difference is that a break statement can break the current loop only, whereas the keyword disable allows disabling of any named block in the design.

Consider the illustration in Example 7-22 on page 142, which looks for the first true bit in the flag. The while loop can be recoded, using the disable statement as shown in Example 7-30. The disable statement terminates the while loop as soon as a true bit is seen.

Example 7-30 Disabling Named Blocks

```
//Illustration: Find the first bit with a value 1 in flag (vector
//variable)
reg [15:0] flag;
integer i; //integer to keep count
initial
begin
 flag = 16'b 0010_0000_0000_0000;
 i = 0;
 begin: block1 //The main block inside while is named block1
 while(i < 16)
  begin
    if (flag[i])
    begin
       $display("Encountered a TRUE bit at element number %d", i);
       disable block1; //disable block1 because you found true bit.
    end
i = i + 1; end
end end
```

Gate-Level Modeling

Modeling using basic Verilog gate primitives, description of and/or and buf/not type gates, rise, fall and turn-off delays, min, max, and typical delays. (Text1)

Dataflow Modeling

Continuous assignments, delay specification, expressions, operators, operands, operator types. (Text1)

Gate-Level Modeling

Modeling using basic Verilog gate primitives, description of and/or and buf/not type gates, rise, fall and turn-off delays, min, max, and typical delays.

Modeling using basic Verilog gate primitives:

Most digital design is now done at gate level or higher levels of abstraction. At gate level, the circuit is described in terms of gates (e.g., and, nand). Hardware design at this level is intuitive for a user with a basic knowledge of digital logic design because it is possible to see a one-to-one correspondence between the logic circuit diagram and the Verilog description.

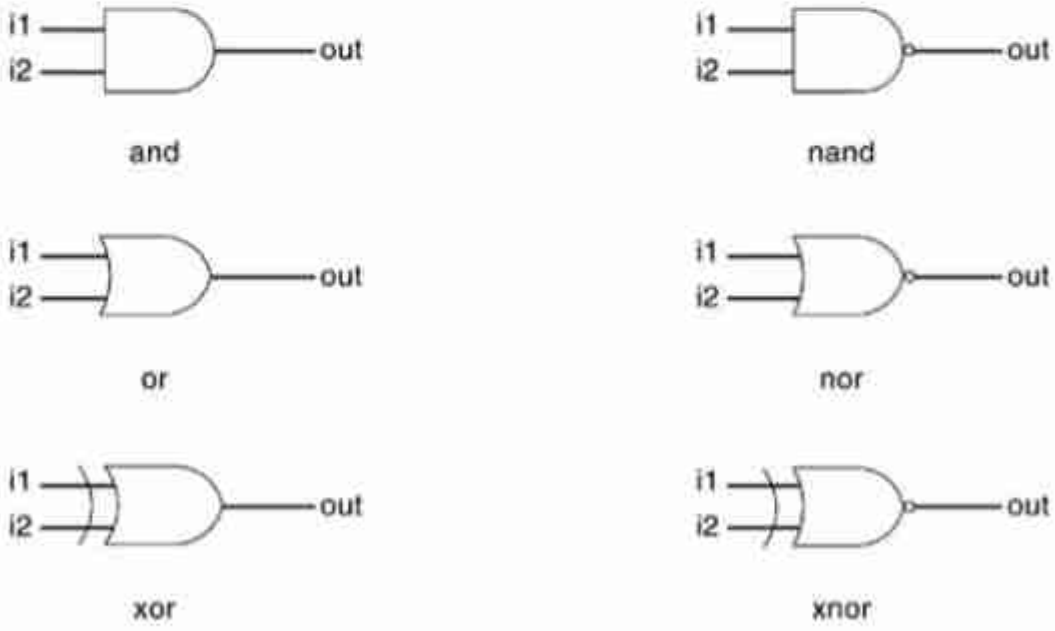
Description of Gate Types:

A logic circuit can be designed by use of logic gates. Verilog supports basic logic gates as predefined primitives. These primitives are instantiated like modules except that they are predefined in Verilog and do not need a module definition. All logic circuits can be designed by using basic gates. There are two classes of basic gates: and/or gates and buf/not gates.

And/Or Gates

And/or gates have one scalar output and multiple scalar inputs. The first terminal in the list of gate terminals is an output and the other terminals are inputs. The output of a gate is evaluated as soon as one of the inputs changes. The and/or gates available in Verilog are shown below:

and, nand, or, nor, xor, xnor.



The corresponding logic symbols for these gates are shown in Figure 1.1. We consider gates with two inputs. The output terminal is denoted by out. Input terminals are denoted by i1 and i2.

These gates are instantiated to build logic circuits in Verilog. Examples of gate instantiations are shown below. In Example 1.1, for all instances, OUT is connected to the output out, and IN1 and IN2

are connected to the two inputs i1 and i2 of the gate primitives. Note that the instance name does not need to be specified for primitives. This lets the designer instantiate hundreds of gates without giving them a name.

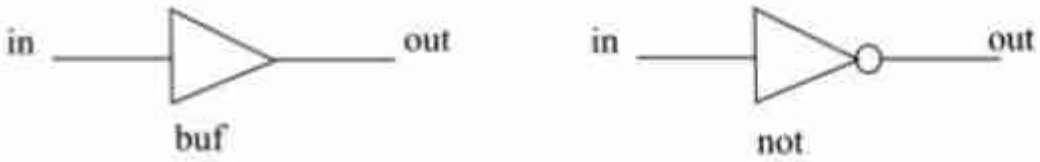
Example 1.1 Gate Instantiation of And/Or Gates

```
wire OUT, IN1, IN2;
// basic gate instantiations.
and a1(OUT, IN1, IN2);
nand na1(OUT, IN1, IN2);
or or1(OUT, IN1, IN2);
nor nor1(OUT, IN1, IN2);
xor x1(OUT, IN1, IN2);
xnor nx1(OUT, IN1, IN2);
// More than two inputs; 3 input nand gate
nand na1_3inp(OUT, IN1, IN2, IN3);
// gate instantiation without instance name
and (OUT, IN1, IN2); // legal gate instantiation
```

The truth tables for these gates define how outputs for the gates are computed from the inputs. Truth tables are defined assuming two inputs. The truth tables for these gates are shown in Table 1.1. Outputs of gates with more than two inputs are computed by applying the truth table iteratively.

Buf/Not Gates

Buf/not gates have one scalar input and one or more scalar outputs. The last terminal in the port list is connected to the input. Other terminals are connected to the outputs. We will discuss gates that have one input and one output.



Two basic buf/not gate primitives are provided in Verilog.

buf not

The symbols for these logic gates are shown in Figure 1.2.

These gates are instantiated in Verilog as shown Example 1.2. Notice that these gates can have multiple outputs but exactly one input, which is the last terminal in the port list.

Example 1.2 Gate Instantiations of Buf/Not Gates

```
// basic gate instantiations.
```

```
buf b1(OUT1, IN);
not n1(OUT1, IN);
// More than two outputs
buf b1_2out(OUT1, OUT2, IN);
// gate instantiation without instance name
not (OUT1, IN); // legal gate instantiation
```

The truth tables for these gates are very simple. Truth tables for gates with one input and one output are shown in Table 1.2.

and	i1: 0	1	x	z
i2: 0	0	0	0	0
1	0	1	x	x
x	0	x	x	x
z	0	x	x	x

nand	i1: 0	1	x	z
i2: 0	1	1	1	1
1	1	0	x	x
x	1	x	x	x
z	1	x	x	x

or	i1: 0	1	x	z
i2: 0	0	1	x	x
1	1	1	1	1
x	x	1	x	x
z	x	1	x	x

nor	i1: 0	1	x	z
i2: 0	1	0	x	x
1	0	0	0	0
x	x	0	x	x
z	x	0	x	x

xor	i1: 0	1	x	z
i2: 0	0	1	x	x
1	1	0	x	x
x	x	x	x	x
z	x	x	x	x

xnor	i1: 0	1	x	z
i2: 0	1	0	x	x
1	0	1	x	x
x	x	x	x	x
z	x	x	x	x

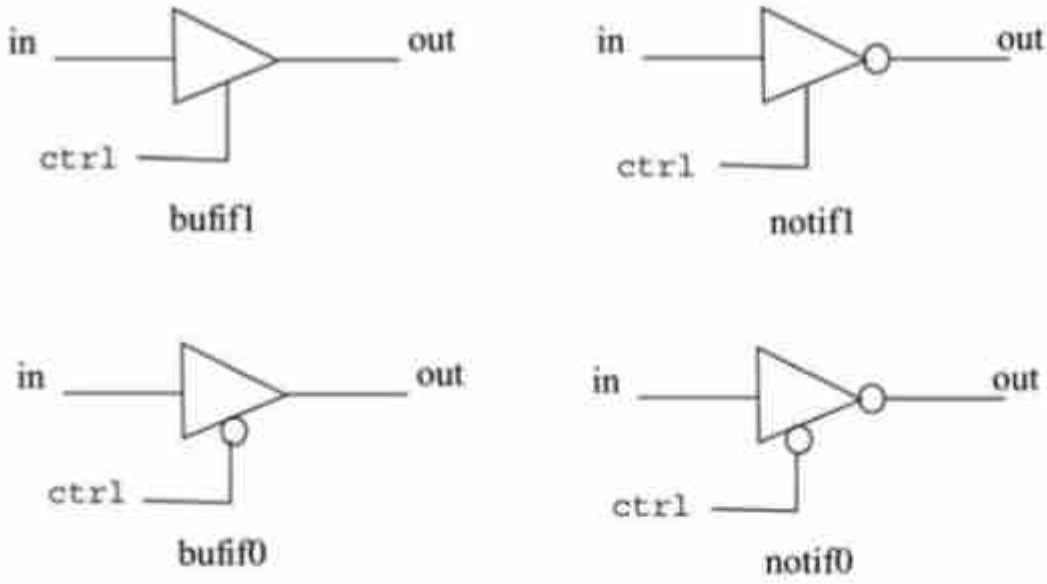
Bufif/notif

Gates with an additional control signal on buf and not gates are also available.

bufif1 notif1

bufif0 notif0

These gates propagate only if their control signal is asserted. They propagate z if their control signal is deasserted. Symbols for bufif/notif are shown in Figure 1.3.



The truth tables for these gates are shown in Table 1.3. These gates are used when a signal is to be driven only when the control signal is asserted. Such a situation is applicable when multiple drivers drive the signal.

bufif1	ctrl 0	1	x	z
in 0	z	0	L	L
1	z	1	H	H
x	z	x	x	x
z	z	x	x	x

bufif0	ctrl 0	1	x	z
in 0	0	z	L	L
1	1	z	H	H
x	x	z	x	x
z	x	z	x	x

notif1	ctrl 0	1	x	z
in 0	z	1	H	H
1	z	0	L	L
x	z	x	x	x
z	z	x	x	x

notif0	ctrl 0	1	x	z
in 0	1	z	H	H
1	0	z	L	L
x	x	z	x	x
z	x	z	x	x

Example 5-3 Gate Instantiations of Bufif/Notif Gates

```
//Instantiation of bufif gates.
bufif1 b1 (out, in, ctrl);
bufif0 b0 (out, in, ctrl);
//Instantiation of notif gates
```

```
notif1 n1 (out, in, ctrl);
notif0 n0 (out, in, ctrl);
```

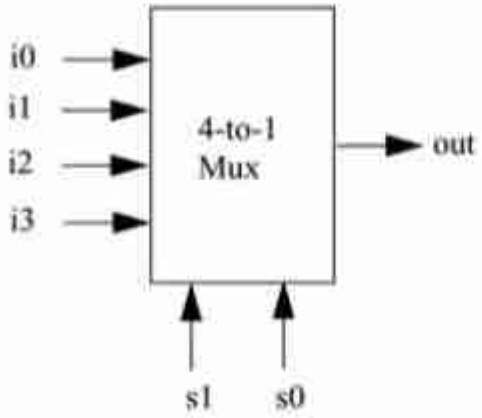
Array of Instances

There are many situations when repetitive instances are required. These instances differ from each other only by the index of the vector to which they are connected. To simplify specification of such instances, Verilog HDL allows an array of primitive instances to be defined. Example 1.4 shows an example of an array of instances.

Example 1.4 Simple Array of Primitive Instances

```
wire [7:0] OUT, IN1, IN2;
// basic gate instantiations.
nand n_gate[7:0](OUT, IN1, IN2);
// This is equivalent to the following 8 instantiations
nand n_gate0(OUT[0], IN1[0], IN2[0]);
nand n_gate1(OUT[1], IN1[1], IN2[1]);
nand n_gate2(OUT[2], IN1[2], IN2[2]);
nand n_gate3(OUT[3], IN1[3], IN2[3]);
nand n_gate4(OUT[4], IN1[4], IN2[4]);
nand n_gate5(OUT[5], IN1[5], IN2[5]);
nand n_gate6(OUT[6], IN1[6], IN2[6]);
nand n_gate7(OUT[7], IN1[7], IN2[7]);
```

Examples



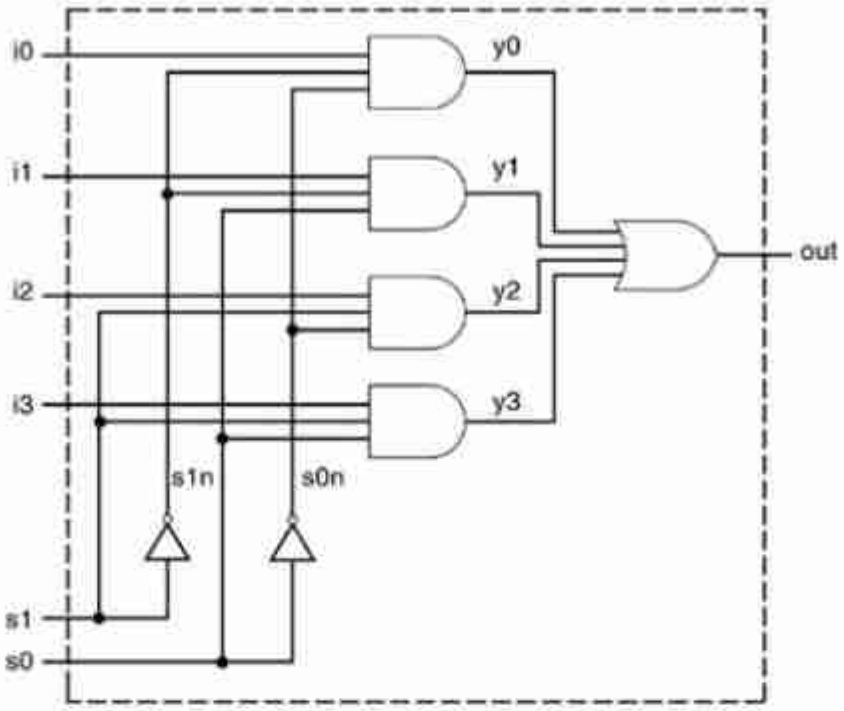
s1	s0	out
0	0	I0
0	1	I1
1	0	I2
1	1	I3

Having understood the various types of gates available in Verilog, we will discuss a real example that illustrates design of gate-level digital circuits.

Gate-level multiplexer

We will design a 4-to-1 multiplexer with 2 select signals. Multiplexers serve a useful purpose in logic design. They can connect two or more sources to a single destination. They can also be used to

implement boolean functions. We will assume for this example that signals s1 and s0 do not get the value x or z.



The logic diagram has a one-to-one correspondence with the Verilog description. The Verilog description for the multiplexer is shown in Example 1.5. Two intermediate nets, s0n and s1n, are created; they are complements of input signals s1 and s0. Internal nets y0, y1, y2, y3 are also required. Note that instance names are not specified for primitive gates, not, and, and or. Instance names are optional for Verilog primitives but are mandatory for instances of user-defined modules.

Example 5-5 Verilog Description of Multiplexer

```
// Module 4-to-1 multiplexer. Port list is taken exactly from
// the I/O diagram.
module mux4_to_1 (out, i0, i1, i2, i3, s1, s0);
// Port declarations from the I/O diagram
output out;
input i0, i1, i2, i3;
input s1, s0;
// Internal wire declarations
wire s1n, s0n;
wire y0, y1, y2, y3;
// Gate instantiations
// Create s1n and s0n signals.
not (s1n, s1);
not (s0n, s0);
```

```
// 3-input and gates instantiated
and (y0, i0, s1n, s0n);
and (y1, i1, s1n, s0);
and (y2, i2, s1, s0n);
and (y3, i3, s1, s0);
// 4-input or gate instantiated
or (out, y0, y1, y2, y3);
endmodule
```

This multiplexer can be tested with the stimulus shown in Example 1.6. The stimulus checks that each combination of select signals connects the appropriate input to the output. The signal OUTPUT is displayed one time unit after it changes. System task $monitor could also be used to display the signals when they change values.

Example 1.6 Stimulus for Multiplexer

```
// Define the stimulus module (no ports)
module stimulus;
// Declare variables to be connected
// to inputs
reg IN0, IN1, IN2, IN3;
reg S1, S0;
// Declare output wire
wire OUTPUT;
// Instantiate the multiplexer
mux4_to_1 mymux(OUTPUT, IN0, IN1, IN2, IN3, S1, S0);
// Stimulate the inputs
// Define the stimulus module (no ports)
initial
begin
 // set input lines
 IN0 = 1; IN1 = 0; IN2 = 1; IN3 = 0;
 #1 $display("IN0= %b, IN1= %b, IN2= %b, IN3= %b\n",IN0,IN1,IN2,IN3);
 // choose IN0
 S1 = 0; S0 = 0;
 #1 $display("S1
 // choose IN1
```

```
 S1 = 0; S0 = 1;
 #1 $display("S1
 // choose IN2
 S1 = 1; S0 = 0;
 #1 $display("S1
 // choose IN3
 S1 = 1; S0 = 1;
 #1 $display("S1
end
endmodule
= %b,
= %b,
= %b,
= %b,
S0 = %b, OUTPUT = %b \n", S1, S0, OUTPUT);
S0 = %b, OUTPUT = %b \n", S1, S0, OUTPUT);
S0 = %b, OUTPUT = %b \n", S1, S0, OUTPUT);
S0 = %b, OUTPUT = %b \n", S1, S0, OUTPUT);
```

The output of the simulation is shown below. Each combination of the select signals is tested.

IN0= 1, IN1= 0, IN2= 1, IN3= 0

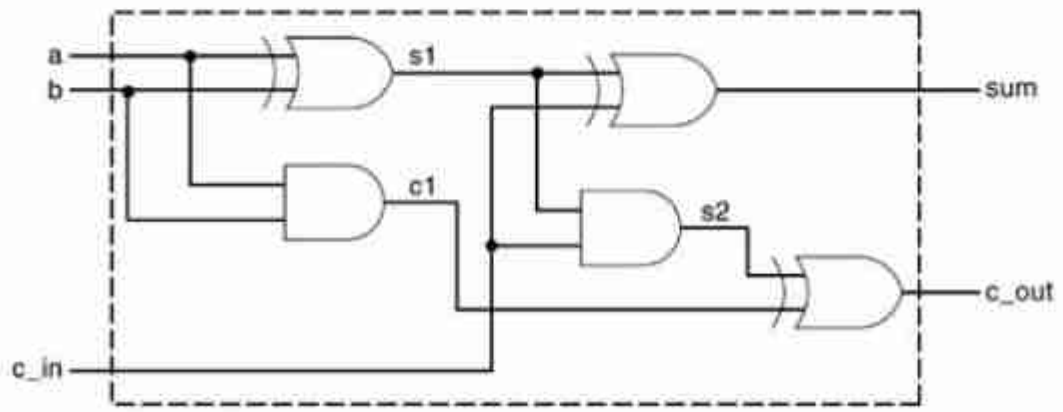
S1 = 0, S0 = 0, OUTPUT = 1

S1 = 0, S0 = 1, OUTPUT = 0

S1 = 1, S0 = 0, OUTPUT = 1

S1 = 1, S0 = 1, OUTPUT = 0

4-bit Ripple Carry Full Adder



We use primitive logic gates, and we apply stimulus to the 4-bit full adder to check functionality .

For the sake of simplicity, we will implement a ripple carry adder. The basic building block is a 1-bit full adder. The mathematical equations for a 1-bit full adder are shown below.

sum = (a $\oplus$b $\oplus$cin) cout=(a b)+cin (a b)

The logic diagram for a 1-bit full adder is shown in Figure 1.6.

This logic diagram for the 1-bit full adder is converted to a Verilog description, shown in Example 1.7.

Example 1.7 Verilog Description for 1-bit Full Adder

```
// Define a 1-bit full adder
module fulladd(sum, c_out, a, b, c_in);
// I/O port declarations
output sum, c_out;
input a, b, c_in;
// Internal nets
wire s1, c1, c2;
// Instantiate logic gate primitives
xor (s1, a, b);
and (c1, a, b);
xor (sum, s1, c_in);
and (c2, s1, c_in);
xor  (c_out, c2, c1);
endmodule
```

A 4-bit ripple carry full adder can be constructed from four 1-bit full adders, as shown in Figure 1.7. Notice that fa0, fa1, fa2, and fa3 are instances of the module fulladd (1-bit full adder).

Example 1.8 Verilog Description for 4-bit Ripple Carry Full Adder

```
// Define a 4-bit full adder
module fulladd4(sum, c_out, a, b, c_in);
// I/O port declarations
output [3:0] sum;
output c_out;
input[3:0] a, b;
input c_in;
// Internal nets
wire c1, c2, c3;
// Instantiate four 1-bit full adders.
```

```
fulladd fa0(sum[0], c1, a[0], b[0], c_in);
fulladd fa1(sum[1], c2, a[1], b[1], c1);
fulladd fa2(sum[2], c3, a[2], b[2], c2);
fulladd fa3(sum[3], c_out, a[3], b[3], c3);
endmodule
```

Example 1.9 Stimulus for 4-bit Ripple Carry Full Adder

```
// Define the stimulus (top level module)
module stimulus;
// Set up variables
reg [3:0] A, B;
reg C_IN;
wire [3:0] SUM;
wire C_OUT;
// Instantiate the 4-bit full adder. call it FA1_4
fulladd4 FA1_4(SUM, C_OUT, A, B, C_IN);
// Set up the monitoring for the signal values
initial
begin
$monitor($time," A= %b, B=%b, C_IN= %b, --- C_OUT= %b, SUM= %b\n", A, B, C_IN, C_OUT,
SUM);
end
// Stimulate inputs
initial
begin
  A = 4'd0; B = 4'd0; C_IN = 1'b0;
  #5 A = 4'd3; B = 4'd4;
  #5 A = 4'd2; B = 4'd5;
  #5 A = 4'd9; B = 4'd9;
  #5 A = 4'd10; B = 4'd15;
  #5 A = 4'd10; B = 4'd5; C_IN = 1'b1;
end
endmodule
```

The output of the simulation is shown below.

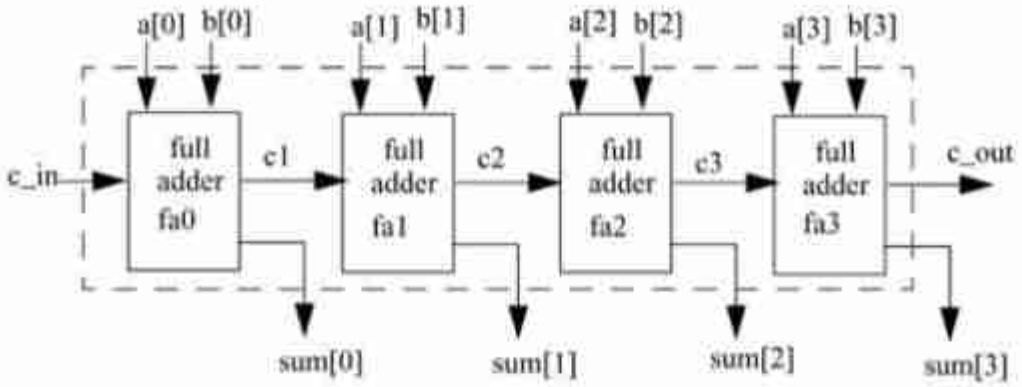
```
0 A= 0000, B=0000, C_IN= 0, --- C_OUT= 0, SUM= 0000
5 A= 0011, B=0100, C_IN= 0, --- C_OUT= 0, SUM= 0111
```

10 A= 0010, B=0101, C_IN= 0, --- C_OUT= 0, SUM= 0111

15 A= 1001, B=1001, C_IN= 0, --- C_OUT= 1, SUM= 0010

20 A= 1010, B=1111, C_IN= 0, --- C_OUT= 1, SUM= 1001

25 A= 1010, B=0101, C_IN= 1,, C_OUT= 1, SUM= 0000



Gate Delays

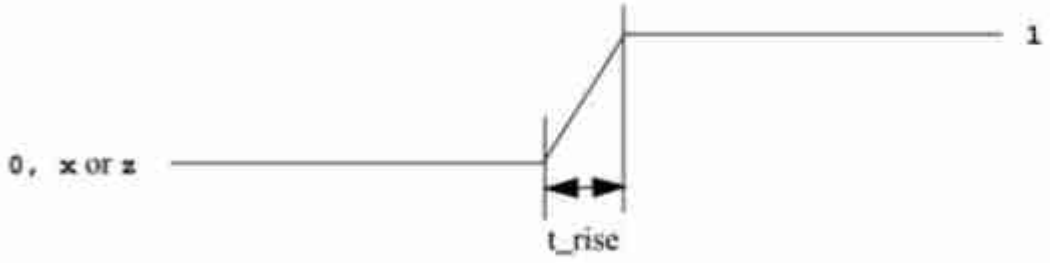
Until now, we described circuits without any delays (i.e., zero delay). In real circuits, logic gates have delays associated with them. Gate delays allow the Verilog user to specify delays through the logic circuits. Pin-to-pin delays can also be specified in Verilog.

Rise, Fall, and Turn-off Delays

There are three types of delays from the inputs to the output of a primitive gate.

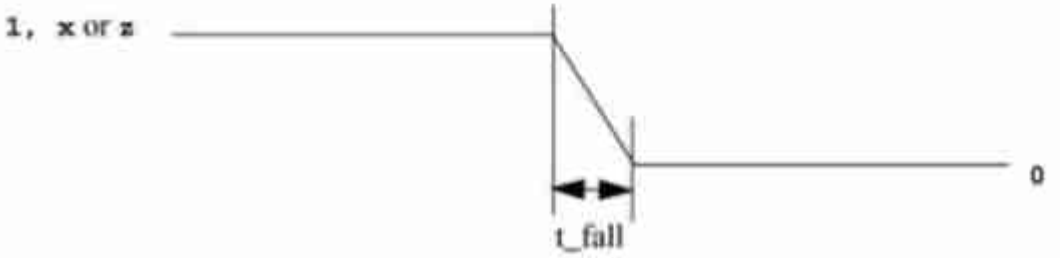
Rise delay

The rise delay is associated with a gate output transition to a 1 from another value.



Fall delay

The fall delay is associated with a gate output transition to a 0 from another value.



Turn-off delay

The turn-off delay is associated with a gate output transition to the high impedance value (z) from another value.

If the value changes to x, the minimum of the three delays is considered.

Three types of delay specifications are allowed. If only one delay is specified, this value is used for all transitions. If two delays are specified, they refer to the rise and fall delay values. The turn-off delay is the minimum of the two delays. If all three delays are specified, they refer to rise, fall, and turn-off delay values. If no delays are specified, the default value is zero. Examples of delay specification are shown in Example 1.10.

Example 1.10 Types of Delay Specification

```
// Delay of delay_time for all transitions
and #(delay_time) a1(out, i1, i2);
// Rise and Fall Delay Specification.
and #(rise_val, fall_val) a2(out, i1, i2);
// Rise, Fall, and Turn-off Delay Specification
bufif0 #(rise_val, fall_val, turnoff_val) b1 (out, in, control);
```

Examples of delay specification are shown below.

```
and #(5) a1(out, i1, i2); //Delay of 5 for all transitions
and #(4,6) a2(out, i1, i2); // Rise = 4, Fall = 6
bufif0 #(3,4,5) b1 (out, in, control); // Rise = 3, Fall = 4, Turn-off =5
```

Min/Typ/Max Values

Verilog provides an additional level of control for each type of delay mentioned above. For each type of delay?rise, fall, and turn-off?three values, min, typ, and max, can be specified. Any one value can be chosen at the start of the simulation. Min/typ/max values are used to model devices whose delays vary within a minimum and maximum range because of the IC fabrication process variations.

Min value

The min value is the minimum delay value that the designer expects the gate to have.

Typ val

The typ value is the typical delay value that the designer expects the gate to have.

Max value

The max value is the maximum delay value that the designer expects the gate to have.

Min, typ, or max values can be chosen at Verilog run time. Method of choosing a min/typ/max value may vary for different simulators or operating systems. (For Verilog- XL , the values are chosen by specifying options +maxdelays, +typdelays, and +mindelays at run time. If no option is specified, the typical delay value is the default). This allows the designers the flexibility of building three delay

values for each transition into their design. The designer can experiment with delay values without modifying the design.

Examples of min, typ, and max value specification for Verilog-XL are shown in Example 1.11.

Example 1.11 Min, Max, and Typical Delay Values

```
// One delay
// if +mindelays, delay= 4
// if +typdelays, delay= 5
// if +maxdelays, delay= 6
and #(4:5:6) a1(out, i1, i2);
// Two delays
// if +mindelays, rise= 3, fall= 5, turn-off = min(3,5)
// if +typdelays, rise= 4, fall= 6, turn-off = min(4,6)
// if +maxdelays, rise= 5, fall= 7, turn-off = min(5,7)
and #(3:4:5, 5:6:7) a2(out, i1, i2);
// Three delays
// if +mindelays, rise= 2 fall= 3 turn-off = 4
// if +typdelays, rise= 3 fall= 4 turn-off = 5
// if +maxdelays, rise= 4 fall= 5 turn-off = 6
and #(2:3:4, 3:4:5, 4:5:6) a3(out, i1,i2);
```

Examples of invoking the Verilog-XL simulator with the command-line options are shown below. Assume that the module with delays is declared in the file test.v.

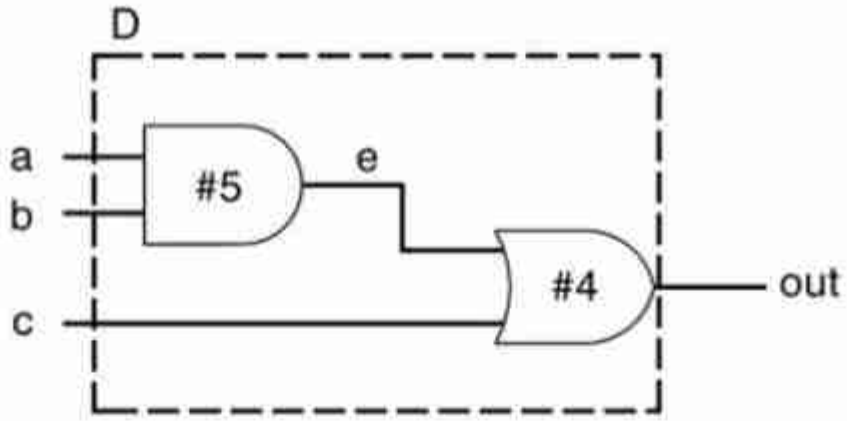
```
//invoke simulation with maximum delay
> verilog test.v +maxdelays
//invoke simulation with minimum delay
> verilog test.v +mindelays
//invoke simulation with typical delay
> verilog test.v +typdelays
```

Delay Example

Let us consider a simple example to illustrate the use of gate delays to model timing in the logic circuits. A simple module called D implements the following logic equations:

out=(a b)+c

The gate-level implementation is shown in Module D (Figure 1.8). The module contains two gates with delays of 5 and 4 time units.



The module D is defined in Verilog as shown in Example 1.12.

Example 1.12 Verilog Definition for Module D with Delay

```
// Define a simple combination module called D
module D (out, a, b, c);
// I/O port declarations
output out;
input a,b,c;
// Internal nets
wire e;
// Instantiate primitive gates to build the circuit
and #(5) a1(e, a, b); //Delay of 5 on gate a1
or  #(4) o1(out, e,c); //Delay of 4 on gate o1
endmodule
```

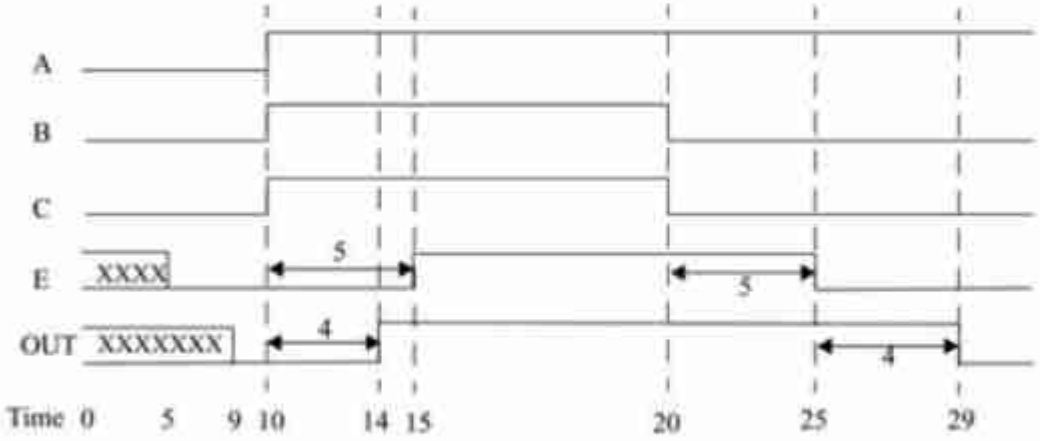
This module is tested by the stimulus file shown in Example 1.13. **Example 1.13 Stimulus for Module D with Delay**

```
// Stimulus (top-level module)
module stimulus;
// Declare variables
reg A, B, C;
wire OUT;
// Instantiate the module D
D d1( OUT, A, B, C);
// Stimulate the inputs. Finish the simulation at 40 time units.
initial
begin
  A= 1'b0; B= 1'b0; C= 1'b0;
```

```
 #10 A= 1'b1; B= 1'b1; C= 1'b1;
 #10 A= 1'b1; B= 1'b0; C= 1'b0;
 #20 $finish;
end
endmodule
```

The waveforms from the simulation are shown in Figure 1.9 to illustrate the effect of specifying delays on gates. The waveforms are not drawn to scale. However, simulation time at each transition is specified below the transition.

1. The outputs E and OUT are initially unknown.
2. At time 10, after A, B, and C all transition to 1, OUT transitions to 1 after a delay of 4 time units and E changes value to 1 after 5 time units.
3. At time 20, B and C transition to 0. E changes value to 0 after 5 time units, and OUT transitions to 0, 4 time units after E changes.



It is a useful exercise to understand how the timing for each transition in the above waveform corresponds to the gate delays shown in Module D.

MODULE 4

SYNTHESIS AND VERIFICATION

Verilog HDL is a hardware description language that can describe hardware not only at the gate level and the register-transfer level (RTL), but also at the algorithmic level. This makes translating a design described in Verilog HDL to logic gates a non-trivial process. It explains how the basics involved in the mapping of a Verilog HDL model to logic gates.

4.1 What is Synthesis?

Synthesis is the process of constructing a gate level netlist from a register transfer level model of a circuit described in Verilog HDL. Figure 4-1 shows such a process. A synthesis system may as an intermediate step, generate a netlist that is comprised of register-transfer level blocks such as flip-flops, arithmetic-logic-units, and multiplexers, interconnected by wires. In such a case, a second program called the RTL module builder is necessary. The purpose of this builder is to build, or acquire from a library of predefined components, each of the required RTL blocks in the user-specified target technology.

BGSIT, ACU 2020-21.

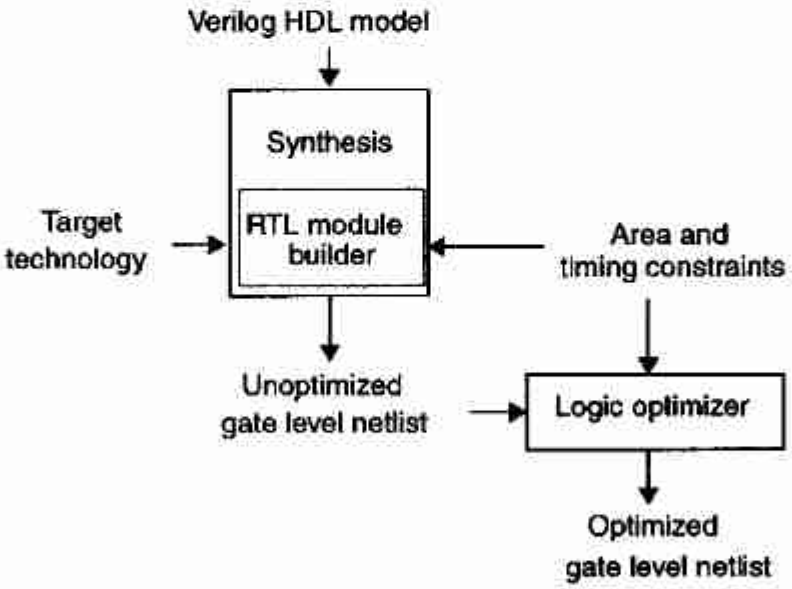


Figure 4.1 The synthesis process.

Having produced a gate level netlist, a logic optimizer reads in the netlist and optimizes the circuit for the user-specified area and timing constraints. These area and timing constraints may also be used by the module builder for appropriate selection or generation of RTL blocks.

Figure 4-2 shows the basic elements of Verilog HDL and the elements used in hardware. A mapping mechanism or a construction mechanism has to be provided that

translates the Verilog HDL elements into their corresponding hardware elements. Questions to ask are:

How does a data type translate to hardware?

How are constants mapped to logic values?

How are statements translated to hardware?

The following sections discuss these mappings in more detail.

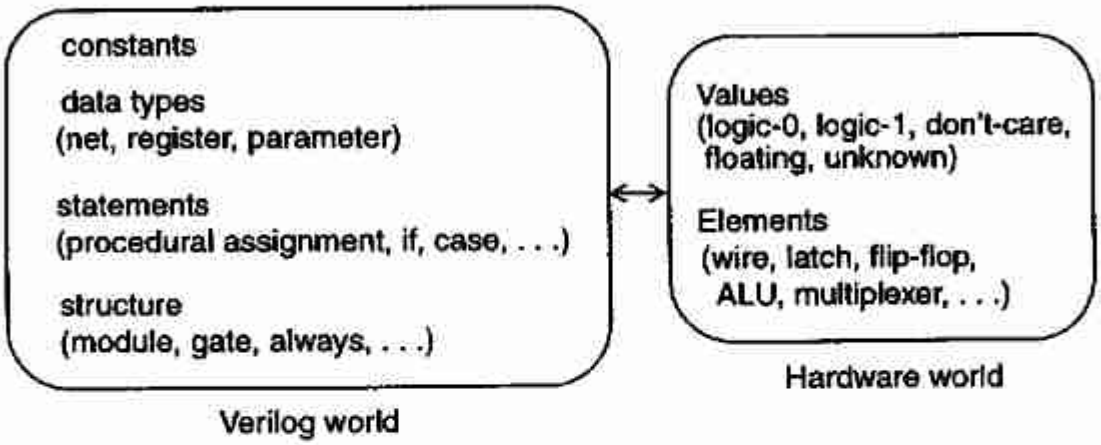


Figure 4-2 The two worlds of synthesis.

4.2 Synthesis in a Design Process

Verilog HDL is a hardware description language that allows a designer to model a circuit at different levels of abstraction, ranging from the gate level, register-transfer level, behavioral level to the algorithmic level. Thus a circuit can be described in many different ways, not all of which may be synthesizable. Compounding this is the fact that Verilog HDL was designed primarily as a simulation language and not as a language for synthesis. Consequently, there are many constructs in Verilog HDL that have no hardware counterpart, for example, the $display system call. Also there is no standardized subset of Verilog HDL for register-transfer level synthesis.

Because of these problems, different synthesis systems support different Verilog HDL subsets for synthesis. Since there is no single object in Verilog HDL that means a latch or a flip-flop, each synthesis system may provide different mechanisms to model a flip-flop or a latch. Each synthesis system therefore defines its own subset of Verilog HDL including its own modeling style.

Figure 4-3 shows a circuit that is described in many different ways using Verilog HDL. A synthesis system that supports synthesis of styles A and B may not support that of style C This implies that typically synthesis models are non-portable across different synthesis systems. Style D may not be synthesizable at all.

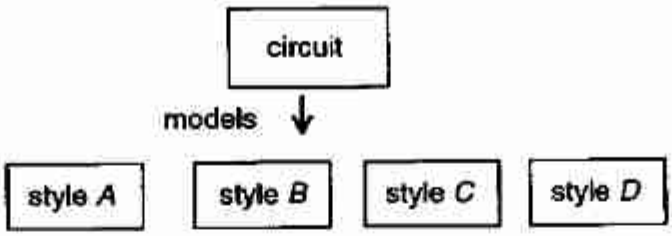


Figure 4-3 Same behavior, different styles.

This limitation creates a severe handicap because now the designer not only has to understand Verilog HDL, but also has to understand the synthesis-specific modeling style before a synthesizable model can be written. The typical design process shown in Figure 4-4 cannot always be followed for Verilog HDL synthesis.

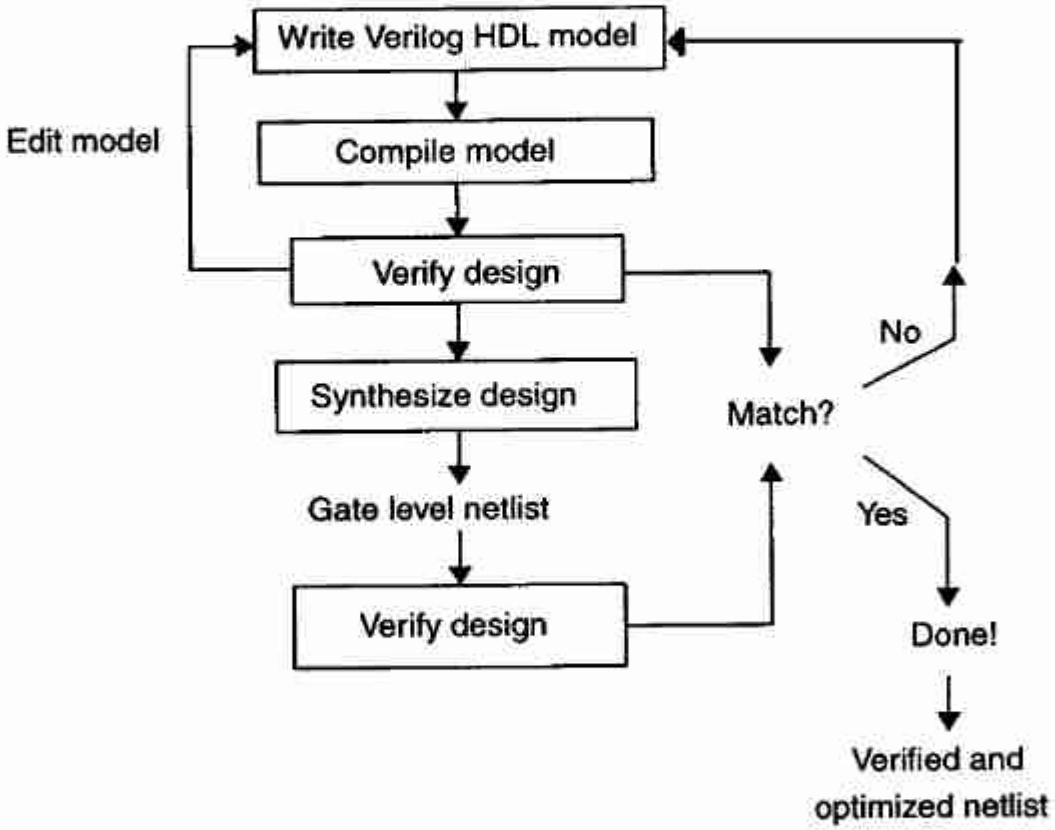


Figure 4-4 Typical design process.

The problem with this design process is that if the Verilog HDL model is written without knowing the synthesis modeling style (this assumes that the model is being written for synthesis; if not, then a non-synthesizable model may be perfectly okay), only during the synthesis phase will the de signer learn about the synthesis-specific modeling restriction and style for synthesis. A model rewrite may be necessary at this point. Also a lot of time may have been wasted in the "Write Verilog HDL model" -> "Compile model" -> "Verify" -> "Edit model" loop. Instead, a more practical design process shown in Figure 4-5 has to be followed for Verilog HDL synthesis. The synthesis methodology checker is

needed to ensure that the model being written is synthesizable. Note that this must be done within the first verification loop. In this way, after the verification results have been checked, a verified synthesizable model exists, which can then be synthesized.

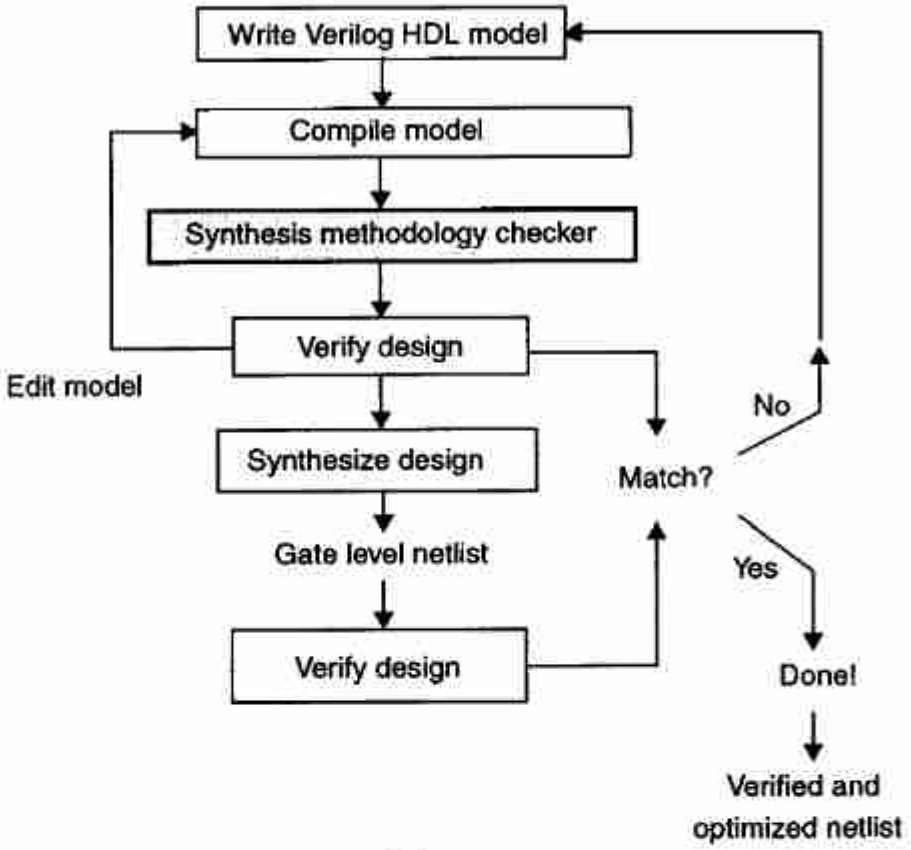


Figure 4-5 New design process.

4.3 Logic Value System

The common values used in modeling hardware are:

- logic-0
- logic-1
- high-impedance
- don't care
- unknown

All these values are defined explicitly in Verilog HDL except for the don't-care value. A synthesis system treats the value x, when it is assigned to a variable, as a don't care value. Here is the mapping between the Verilog HDL values and the hardware modeling values:

BALAJI B S, ASST PROF, L

- 0 <--> logic-0
- 1 <--> logic-1
- z<--> high-impedance
- z<--> don't care (in casex and casez statements)
- x<--> don't care
- x <--> unknown
- X<--> don't care
- x <--> unknown

4.4 Bit-widths

4.4.1 Data Types

In Verilog HDL, a variable belongs to one of the two data types:

i. Net data type

ii. Register data type

Net Data Type

The size of a net is explicitly specified in a net declaration

wire [4:0] Dak; // A 5-bit wire net.

wor Ax; // 1-bit wor net.

When no size is explicitly specified in a net declaration, the default size is one bit.

Here are the different kinds of net data types that are supported for synthesis.

wire wor wand tri supply0 supply1

The wire net is the most commonly used net type. When there are multiple drivers driving a wire net, the outputs of the drivers are shorted together. Here is an example.

BALAJI B S, ASST. PROF, DEPT. ECE, BGSIT - ACU. 2020-21.

```
module WireExample (BpW, Error, Wait, Valid, Clear);
  input Error, Wait, Valid, Clear;
  output BpW;
  wire BpW;
  assign BpW = Error & Wait;
  assign BpW = Valid | Clear;
endmodule
```

//Synthesized netlist is shown in Figure 4-6.

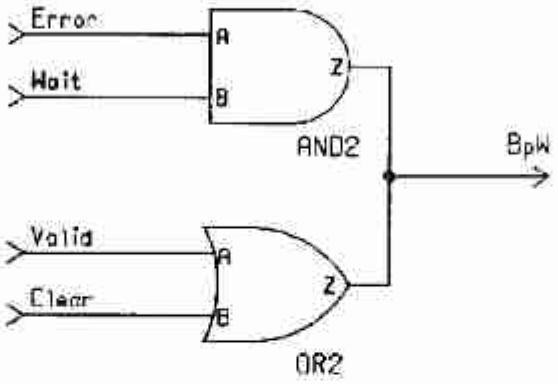


Figure 4-6 Multiple drivers driving a wire net.

The wor nets and wand nets are used when multiple driver resolution needs to be performed using or-logic and and-logic respectively. Upon synthesis, multiple drivers of such a net are connected together by an or gate (for a wor net) and by an and gate (for a wand net). Here is an example that shows this effect.

```
module UsesGates (BpW, BpR, Error, Wait, Clear);
  input Error, Wait, Clear;
  output BpW, BpR;
  wor BpW;
  wand BpR;

  assign BpW = Error & Wait;
  assign BpW = Valid | Clear;

  assign BpR = Error ^ Valid;
  assign BpR = ! Clear;
endmodule
```

//Synthesized netlist is shown in Figure 4-7.

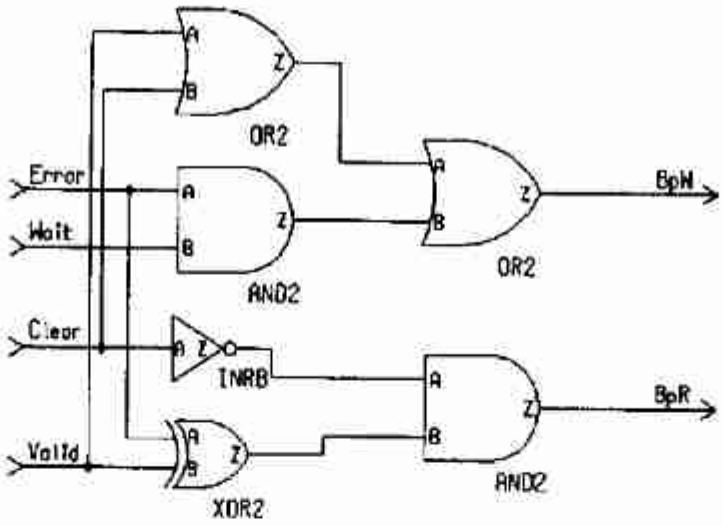


Figure 4-7 Wand net and wor net have multiple drivers.

The tri net synthesizes just like the wire net.

A supply net synthesizes to a wire that is permanently connected to 0 (logic-0), while a supply1 net synthesizes to a wire that is permanently connected to 1 (logic-1).

Register Data Type

The different kinds of register types that are supported for synthesis are:

reg integer

A reg declaration explicitly specifies the size, that is, the corresponding number of bits of the variable in hardware. For example,

```
reg [1:25] Cpt;          // 25-bit variable.
reg Bxr;                 // 1-bit variable.
```

When no size is explicitly specified in a reg declaration, the default is one bit.

For an integer type, the maximum size is 32 bits and the number is assumed to be in 2's complement form. Optionally a synthesis system may perform data flow analysis of the model to determine the maximum size of an integer variable. For example,

```
wire [1:5] Brq, Rbu;
integer Arb;
. . .
Arb = Brq + Rbu;
```

Size of Arb is determined to be 6 bits. An adder of size 6 is sufficient. The leftmost bit is the carry bit.

The register types: time and real, are not supported for synthesis.

4.4.2 Constants

There are three kinds of constants in Verilog HDL: integer, real and string. Real and string constants are not supported for synthesis.

An integer constant can be written in either of the following two forms.

i. Simple decimal

in. Base format

When an integer is written in a simple decimal form, it is interpreted as a signed number. The integer is represented in synthesis as 32 bits in 2's complement form. If an integer is written in the base format form, then the integer is treated as an unsigned number. If a size is explicitly specified for the integer, then the specified size is the number of bits used for the inte ger; if not, 32 bits is used for the size. Here are some examples.

```
30          Signed number, 32 bits
-2          Signed number, 32 bits in 2's complement
2'b10       Size of 2 bits
6'd-4       6-bit unsigned number (-4 is represented in
              2's complement using 6 bits)
'd-10       32-bit unsigned number (-10 is represented in
              2's complement using 32 bits)
```

4.4.3 Parameters

A parameter is a named constant. Since no size is allowed to be specified for a parameter, the size of the parameter is the same as the size of the constant itself.

```
parameter RED = -1, GREEN = 2;
parameter READY = 2'b01, BUSY = 2'b11, EXIT = 2'b10;
```

RED and GREEN are two 32-bit signed constants. READY, BUSY and EXIT is three parameters of size 2 bits each.

4.5 Value Holders for Hardware Modeling

The basic value holders in hardware are:

- wire
- flip-flop (an edge-triggered storage element)
- latch (a level-sensitive storage element)

A variable in Verilog HDL can either be of the net data type or the register data type. For synthesis, a variable of net type maps to a wire in hardware and a variable of the register type maps either to a wire or a storage element (flip-flop or latch) depending on the

context under which the variable is assigned a value. Let us look at a variable of register type in more detail.

In Verilog HDL, a register variable retains its value through the entire simulation run, thus inferring memory. However, this is too general for synthesis. Here is an example of a variable that is used as a temporary and therefore need not be a candidate for a storage element.

```
wire Acr, Bar, Fra;      // A wire is a net type.
reg Trq, Sqp;            // A reg is a register type.
. . .
always @ (Bar or Acr or Fra)
begin[1]
  Trq = Bar & Acr;
  Sqp = Trq | Fra;
end
```

Variable Trq is assigned in the first statement and then used in the right hand-side expression of the second statement. Verilog HDL semantics indicate that Trq retains its value through the entire simulation run. However, it is not necessary to store the value of Trq as a storage element in hardware, since it is assigned and used immediately. Figure 4-8 shows the logic generated.

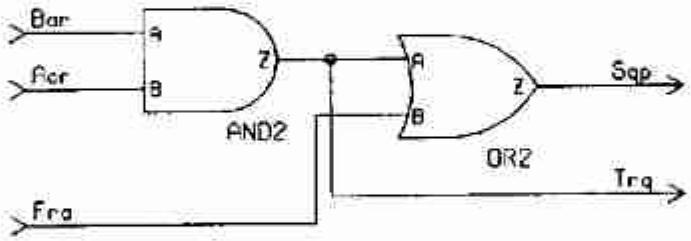


Figure 4-8 Variable Trq is a wire.

Let us look at another example. In this case, variable Trq is used before its assignment.

```
wire Acr, Bar, Fra;
reg Trq, Sqp;
. . .
always @ (Bar or Acr or Fra)
begin
  Sqp = Trq | Fra;
  Trq = Bar & Acr;
end
```

The semantics of this always statement is very clear in Verilog HDL. Whenever an event occurs on Bar, Acr, or Fra (those in the event list), execute the always statement. Since Trq is used before its assignment, Trq has to hold its value during repeated executions of the always statement, thus inferring memory. However, it is not clear how to build a latch for Trq because Trq is not assigned a value under the control of any condition.

BALAJI B S, ASST. PROF, DEPT. ECE, BGSIT – ACU. 2020-21.

A synthesis system may not create a latch in this case and may generate the circuit shown in Figure 4-9. Variable Trq is synthesized to a wire again. However, for functionality to match between the Verilog HDL model and the synthesized netlist, Trq must also be in the event list of the always statement.

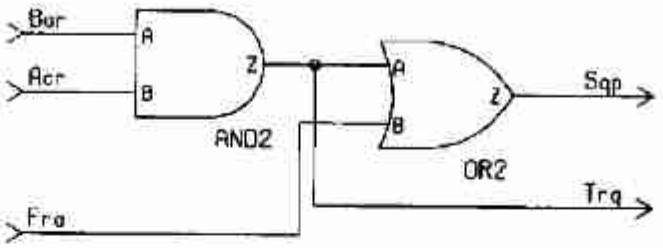


Figure 4-9 No latch for variable Trq.

Here is an example where a latch is inferred for a variable.

```
wire Sat, Ant;
reg Fox, Sout;
. . .
always @ (Sat or Ant)
begin
  if (! Sat)
    Fox = Ant;

  Sout = ! Fox;
end
```

The variable Fox is not assigned in the else-branch of the conditional statement. Consequently, a latch is inferred for Fox since it needs to retain its value when Sat is true. The circuit synthesized in shown in Figure 4-10. How is a flip-flop inferred? It depends on the modeling style being followed and the context under which a variable is assigned a value.

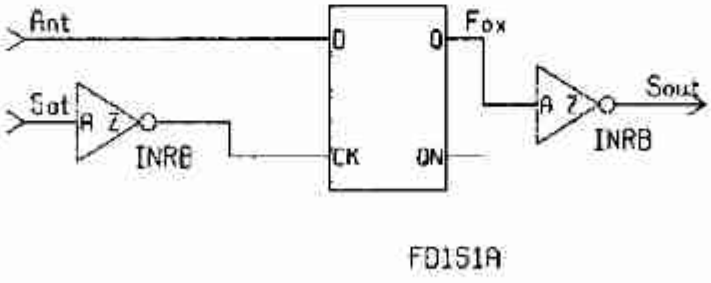


Figure 4-10 Variable Fox is a latch.

This and other examples for flip-flop and latch inferencing are discussed in the next chapter. A memory in hardware can be modeled as an array of flip flops or latches.

4.6 Verification

BALAJI B S, ASST. PROF, DEPT. ECE, BGSIT - ACU. 2020-21.

Having synthesized a Verilog HDL model into a netlist, it is important to verify the functionality of the synthesized netlist to ensure that it still matches the intended functionality. This step is important since a synthesis system may make certain assumptions or interpretations of the Verilog HDL code that may not match those intended by the model writer.

In this chapter, we assume that this verification step is performed using simulation which verifies the functionality between the design model and its synthesized netlist. We illustrate some cases of functional mismatches between the design model and its synthesized netlist that might possibly occur, describe their cause, and provide recommendations for avoiding them.

In this chapter, we assume that the synthesis process produces a synthesized netlist in Verilog HDL as shown in Figure 4-11. A Verilog HDL netlist is a collection of module instances interconnected by nets.

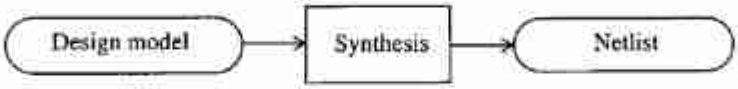


Figure 4-11 A netlist is produced from synthesis.

4.7 A Test Bench

One approach to verifying functionality is to simulate the netlist with the same set of stimulus as used during design model simulation, save the results in a results file and compare to see if the results are identical. This scenario is shown in Figure 4-12.

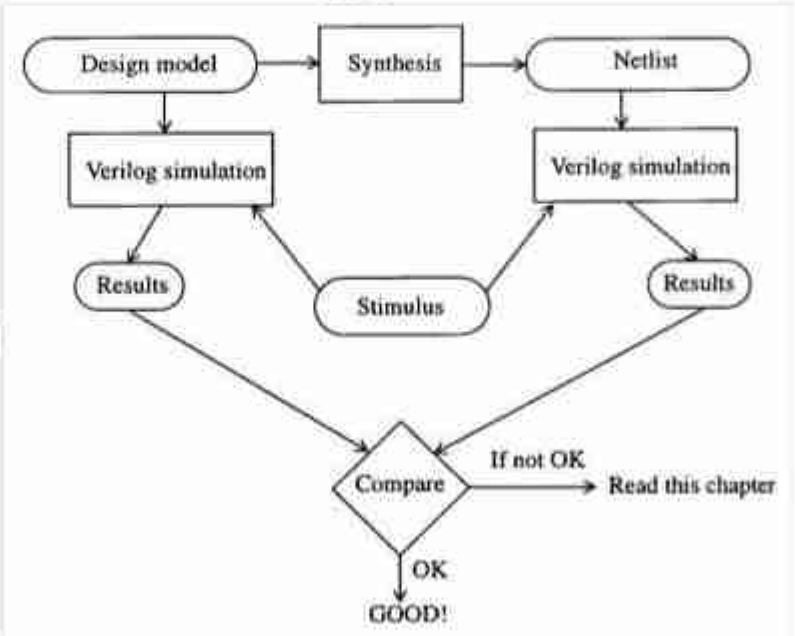


Figure 4-12 Verification by simulation.

Another approach is to write a test bench; a test bench is a model writ ten in Verilog HDL that applies stimulus, compares the output responses, and reports any functional mismatches. Figure 4-13 shows such a scenario A test bench for a full-adder is shown next. The stimulus is read from a vector file "Inputs.vec"; its contents are of the form:

```
100
000
101
011
111
```

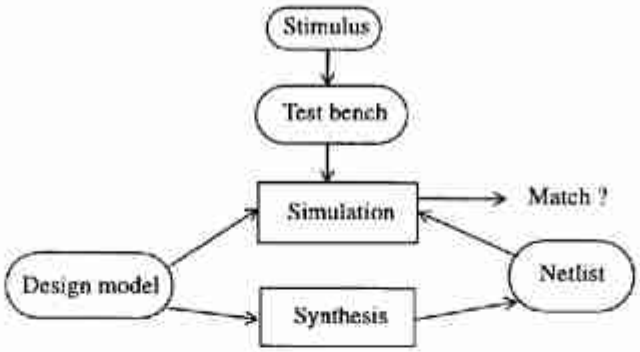


Figure 4-13 Using a common test bench

```
module TestBenchFA;
  parameter WORDS = 5;
  reg [1:3] MemV [1:WORDS];
  reg A, B, Cin;
  wire SumBeh, CoutBeh, SumStr, CoutStr;
  integer J;

  // Instantiate the design module under test:
  FA_RTL F1 (A, B, Cin, SumBeh, CoutBeh);

  // Instantiate the synthesized netlist module:
  FA_Netlist F2 (A, B, Cin, SumStr, CoutStr);

  initial
    begin
      // Read the file with input vectors:
      $readmemb ("Inputs.vec", MemV);

      // Apply each vector to both design module and
      // synthesized netlist module:
      for (J = 1; J <= WORDS; J = J + 1)
      begin
        {A, B, Cin} = MemV[J];
        #5; // Wait for 5 time units for circuit to settle.

        // If output values do not match:
        if ((SumBeh !== SumStr) || (CoutBeh !== CoutStr))
          $display ("*****Mismatch on vector %b ******",
                    MemV[J]);
        else
          $display ("No mismatch on vector %b", MemV[J]);
      end
    end
endmodule
```

This test bench prints all mismatch violations that occur.

In the following sections, we see examples of how mismatches may occur that may be caused due to different interpretations by synthesis (as compared to Verilog HDL language semantics).

4.8 Delays in Assignment Statements

Delays specified in a design model may cause a functional mismatch between the model and the synthesized netlist. Here is an example of an adder model that uses a delay in a continuous assignment, and its synthesized netlist.

```
module Adder (A, B, C);
  input [0:3] A, B;
  output [0:3] C;

  assign #5 C = A + B;
endmodule


// The synthesized netlist is:
module AdderNetList (A0, A1, A2, A3, B0, B1, B2, B3,
                     C0, C1, C2, C3);
  input A0, A1, A2, A3, B0, B1, B2, B3;
  output C0, C1, C2, C3;

  OAI21  C0_1     (C2_1, S248, S310, C0);
  ND2    S310_1   (S248, C2_1, S310);
  XOR2   S248_1   (A0, B0, S248);
  AOI22  C2_2     (S241, S244, A1, B1, C2_1);
  OAI22  C1_1     (S295, S244, S299, S241, C1);
  INRB   S299_1   (S244, S299);
  OAI22  S244_1   (B1, S242, A1, S243, S244);
  INRB   S243_1   (B1, S243);
  INRB   S242_1   (A1, S242);
  INRB   S295_1   (S241, S295);
  OAI22  S241_1   (S291, S237, S238, S239, S241);
  INRB   S291_1   (S240, S291);
  OAI21  C2_1     (S237, S240, S334, C2);
  ND2    S334_1   (S237, S240, S334);
  OAI22  S240_1   (B2, S238, A2, S239, S240);
  INRB   S239_1   (B2, S239);
  INRB   S238_1   (A2, S238);
  ND2    S237_1   (A3, B3, S237);
  OAI22  C3_1     (B3, S235, A3, S236, C3);
  INRB   S236_1   (B3, S236);
  INRB   S235_1   (A3, S235);
endmodule
```

If vectors from a stimulus file were applied, say every 1 ns, and all the module instances in the netlist represent behavioral models with no delay, the results between the design

model and the netlist will be skewed because of the difference in delays. The correct approach in such a case is:

i. Either to delete all delays from the design model,

ii. Or to apply the stimulus with a period greater than 5 ns: a better rule. When delays are present in the models for the library modules, these delays must also be considered in determining the stimulus period.

Recommendation: To avoid delays in a design model from causing functional mismatches, the maximum delay in the model must be computed. The stimulus application time must be greater than this maximum delay.

ACU 2020-21.

4.9 Unconnected ports

It could happen that a synthesized netlist has a module instance with an unconnected input port. Such a case is shown in the following example.

```
module AOI22 (A, B, D, Z);
  input A, B, D;
  output Z;
  reg T1, T2, C;

  always @ (A or B or D)
  begin
    T1 = A & B;
    T2 = C & D;          // C is never assigned a value.
    Z = ! (T1 | T2);
  end
endmodule

// Its synthesized netlist is:
module AOI22_NetList (A, B, D, Z);
  input A, B, D;
  output Z;

  AND2     S0_1    ( A, B, T1_1);
  AND2     S1_1    ( , D, T2_1);   // First port is open.
  OR2      S2_1    ( T1_1, T2_1, T2_0);
  INRB     S3_1    ( T2_0, Z);
endmodule
// Note: A logic optimizer has not yet been used; it
// could potentially remove a redundant gate.
```

Notice that in the synthesized netlist, the first input of the AND2 module instance SI_I is open. During the simulation of module A0122_NetList, the open input takes the value z', whereas the unassigned value of C in module AO122" takes on a default value of x. The fact that different values are used for C during the design model simulation and the synthesized netlist simulation, a potential exists for functional mismatch to occur due to different values being the default in the two different domains.

Recommendation: A good synthesis system will issue warning messages about a value used before being assigned (such as variable C in the module AO122). Pay attention to these warnings.

4.10 Missing Latches

In Module 3, we described rules for inferring latches. We also described an exception to the rule, that is, a variable does not infer a latch if it is used as a temporary. However, there are a few other cases where a variable may not infer a latch, even though it appears from the code sequence that it should.

Let us consider the first case.

```
wire Control, Jrequest;
reg DebugX;
. . .
always @ (Control or Jrequest)
  if (Control)
    DebugX = Jrequest;
  else
    DebugX = DebugX;
```

In this always statement, variable DebugX is assigned in all branches of the if statement. However, data flow analysis reveals that the value of DebugX needs to be saved (since its value is used before an assignment when Control is false). In this case, a synthesis system may produce a warning message about variable DebugX being used before its assignment and also about a potential functional mismatch that may occur between the design model and its synthesized netlist.

Let us reiterate the rules for inferring latches once more:

i. A variable is assigned in a conditional statement (if or case), and

ii. Variable is NOT assigned in all branches of the conditional statement, and

iii. Value of variable needs to be saved between multiple invocations of the always statement.

All the three conditions must be satisfied before a variable is inferred as a latch. In the above always statement, DebugX violates rule (ii). Therefore no latch is produced for DebugX.

Here is another example.

BALAJI B.S, ASST. PROF, DEPT. ECE, BGSIT - ACU. 2020-21.

```
always @ (Control)
begin
  if (Control)
    DebugX = Jrequest;
  else
    DebugX = Bdy;

  Bdy = DebugX;
end
```

In this always statement, it appears that there should be a latch for either DebugX or Bdy. There is no latch for DebugX since it violates rule (ii). There is no latch for Bdy since it violates rule (i). Language semantics however indicate that value for Bdy needs to be saved. A synthesis system in this case may not produce a latch; instead it may issue a warning message about Bdy being used before its assignment and in addition, produce a warning message about a potential for functional mismatch that may occur.

In the following always statement, no latch is produced for DebugX since it violates rule (ii) but a latch is produced for Bdy.

```
always @ (Control)
begin
  if (Control)
    DebugX = Jrequest;
  else
    DebugX = Bdy;

  if (Jrequest)
    Bdy = DebugX;
end
```

What about the following always statement?

```
always @ (Control)
begin
  if (Control)
    DebugX = Jrequest;
  else
    DebugX = Bdy;

  if (Jrequest)
    Bdy = DebugX;
  else
    Bdy = 'b1;
end
```

There are no latches for DebugX and Bdy. However language semantics indicate that Bdy needs to be saved. A synthesis system may not produce a latch; it may generate a warning about the variable being used before its assignment and that there is a potential for a functional mismatch.

4.11 More on Delays

Delays are often ignored by a synthesis system. The fact that they are ignored may simply cause simulation results to differ between the synthesized netlist and the design model. A case in point.

```
LX = #3 'b1;

if (CondA)
  LX = #5 'b0;
  . . .
```

Model simulation shows a value of 1 on LX after 3 ns and the value going to 0 after 5 ns if the condition CondA is true. However, since a synthesis system ignores delays, if CondA is true, the net effect is as if a 0 is assigned to LX and the appropriate hardware gets synthesized to reflect this. Notice that if the synthesized netlist is simulated, the value of LX will not go to 1 if CondA is true.

Recommendation: Avoid inserting delays into a design model that is to be synthesized. If necessary, lump total delays for a variable in one place.

4.12 Event List

Quite often, a synthesis system ignores the event list of an always statement during synthesis. This can lead to functional mismatches if proper care is not taken in modeling. Here is a simple example.

```
always @ (Read)
  Grt = Read & Clock;
```

// Synthesized netlist is shown in Figure 4-14.

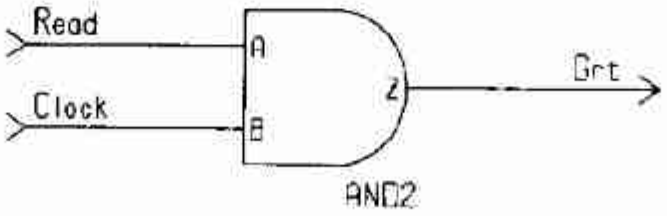


Figure 4-14 Netlist is sensitive to both Read and Clock.

BALAJI S, ASST. PROF, DEPT. ECE, BGSIT, ACU. 2020-21.

The synthesized netlist, as shown in Figure 4-14, evaluates on all changes of Read and Clock, while the always statement executes only on changes to Read.

Here is another example of an always statement with an incomplete event list that may cause functional mismatches.

```
reg Rst;
reg [3:0] Pbus, Treg;
. . .
always @ (Rst)
  if (Rst)
    Treg = 0;
else
  Treg = Pbus;
```

The variable Pbus is not in the event list of the always statement. However in the synthesized netlist, any changes on Pbus will propagate into Treg if the if condition is false. This is not consistent with the design model semantics and thus a functional mismatch occurs.

Recommendation: For an always statement without a clock event (that is, when modeling combinational logic), include all variables read in the always statement in the event list.

4.13 Synthesis Directives

The two synthesis directives we have seen so far, full_case and parallel_case, can potentially cause functional mismatches to occur between the design model and the synthesized netlist. The problem is that these directives are recognized only by a synthesis tool and not by a simulation tool. In either of the cases, if the designer is not careful in specifying the directive, mismatches can occur.

Here is an example of a full_case synthesis directive.

```
reg [1:0] CurrentState, NextState;
. . .
case (CurrentState)              // synthesis full_case
  2'b01 : NextState = 2'b10;
  2'b10 : NextState = 2'b01;
endcase
```

The full_case directive tells the synthesis tool that all possible values that can possibly occur in CurrentState have been listed and the value of Next State is a don't care for all other cases, and therefore, the synthesis tool should not generate latches for NextState. However this may not be true in simulation. It could happen that CurrentState for some reason, gets a value of 2'b00. In such a case, the case statement simulates as if NextState value is saved, but in the synthesized netlist, the value of NextState may not be saved.

ASST. PROF, DEPT. ECE, BGSIT, ACU, 2020-21.

Here is an example of a parallel_case synthesis directive.

```
case (1'b1)                    // synthesis parallel_case
  Gate1 : Mask1 = 1;
  Gate2 : Mask2 = 1;
  Gate3 : Mask3 = 1;
endcase
```

Simulation semantics of the case statement (the parallel_case directive is ignored since it is a comment) specifies that if Gate1 is a 1, then assign 1 to Mask1, else if Gate2 is a 1, assign 1 to Mask2, else if Gate3 is a 1, assign 1 to Mask3. However, with the parallel_case directive, instead of a priority if-structure being synthesized, a parallel decoder is synthesized. This can cause functional mismatches to occur. What if both Gate3 and Gate1 were 1 at the same time? In the case statement, the first branch is taken, whereas in the synthesized netlist, both branches 1 and 3 are enabled. Here is the semantics for the case statement expressed using an if statement.

```
if (Gate1)
  Mask1 = 1;
else if (Gate2)
  Mask2 = 1;
else if (Gate3)
  Mask3 = 1;
```

This is the semantics of the synthesized netlist.

```
if (Gate1)
  Mask1 = 1;

if (Gate2)
  Mask2 = 1;

if (Gate3)
  Mask3 = 1;
```

Recommendation: Use caution when using the synthesis directives: full case and parallel_case. Use only if really necessary.

4.14 Variable Asynchronous Preset

When synthesizing an asynchronous preset clear flip-flop, the recommendation is to assign only constant values under the asynchronous conditions. If a variable is asynchronously read, there is a potential for a functional mismatch to occur. Here is an example.

```
module VarPreset (ClkZ, PreLoad, LoadData, PrintBus,
                  QuickBus);
  input ClkZ, PreLoad;
  input [1:0] LoadData, PrintBus;
  output [1:0] QuickBus;
  reg [1:0] QuickBus;

  always @(negedge PreLoad or posedge ClkZ)
    if (! PreLoad)
      QuickBus <= LoadData; // Asynchounous data assign.
    else
      QuickBus <= PrintBus;
endmodule
```

B.S.BALAJI, ASST. PROF, DEPT. ECE, BGSIT - ACU. 2020-21

// Synthesized netlist is shown in Figure 4-15.

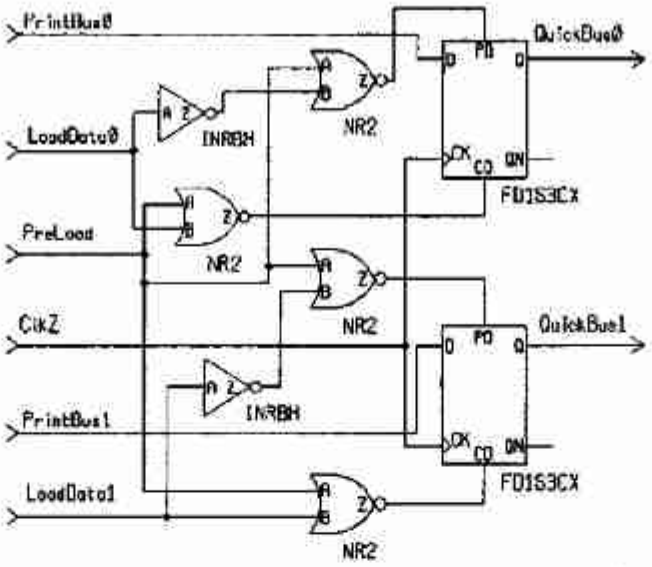


Figure 4-15 Variable asynchronous preset.

Two flip-flops with asynchronous preset and clear are synthesized for the variable QuickBus. The variable LoadData is connected to the preset clear inputs of the flip-flops through other logic. When PreLoad is active (is 0) and LoadData changes, the outputs of the flip-flops are immediately affected because of the asynchronous data change. However in the design model, any change on LoadData has no effect on the output QuickBus. Thus there is a mismatch.

Recommendation: Avoid asynchronously reading a variable and as signing it to a flip-flop; else ensure that there are no changes on asynchronous data when the asynchronous conditions are active.

4.15 Blocking and Non-blocking Assignments

In Chapter 2, we recommended that: blocking assignments be used for modeling combinational logic, and non-blocking assignments be used for modeling sequential logic; blocking assignments may be used for variables that are assigned and used, all within an always statement.

In this section, we explain why this recommendation is important to be followed; else there is a risk of getting functional mismatches.

4.15.1 Combinational Logic

Blocking assignments mirror the dataflow in a combinational circuit. Consider the following always statement.

```
reg TM, TN, TO, TZ;

always @ (A or B or C or D or E)
begin
  TM = A & B;
  TN = C & D;
  TO = TM | TN | E;
  TZ = ! TO;
end
```

BALAJI B S, ASST. PROF, DEPT. ECE, BGSIT - ACU. 2020-21.

All the assignments are blocking assignments. Statements within the sequential block imply to compute the value of TM first, then execute the second statement, assign to TN, then go to third statement, assign to TO, and so on. This mimics the dataflow through the combinational logic.

Let us now change all these to non-blocking assignments.

```
reg TM, TN, TO, TZ;

always @ (A or B or C or D or E)
begin
  TM <= A & B;
  TN <= C & D;
  TO <= TM | TN | E;
  TZ <= ! TO;
end
```

When the first assignment statement executes, TM does not get updated immediately but is scheduled to be assigned at the end of the current simulation cycle. Since all statement executions occur sequentially and in zero time, so when the third statement is executed, the old value of TM is used to compute the value of TO (TM has not been assigned its new value yet). Consequently, the output TZ does not reflect the and-or-invert behavior of the logic. The problem is that TM, TN, and TO all get updated at the end of the current simulation cycle and these updated values are not used again to reevaluate the logic.

A solution to this problem is to place variables TM, TN and TO also in the event list of the always statement, such as:

```
reg TM, TN, TO, TZ;

always @ (A or B or C or D or E)
begin
  TM <= A & B;
  TN <= C & D;
  TO <= TM | TN | E;
  TZ <= ! TO;
end
```

In this case, when TM, TN or TO changes, the always statement is re evaluated and eventually TZ does get the correct value. So there are two problems that have been identified:

- non-blocking assignments do not show the logical flow
- need to put all targets of assignments in the event list

These problems can simply be avoided by using blocking assignments when modeling combinational logic and are therefore recommended.

4.15.2 Sequential Logic

Let us first consider what happens if blocking assignments are exclusively used for modeling sequential logic. Consider the following two always statements.

```
always @(posedge ClkA)      // Label AwA
  . . . = DataOut;          // Read value of DataOut.

always @(posedge ClkA)      // Label AwB
  DataOut = . . .;          // Using blocking assignment.
```

The always statement, labeled AwB, assigns a value to DataOut in a blocking fashion and the always statement, labeled AwA, reads the value of DataOut. If these always statements were simulated in the sequence shown (a simulator orders the always statements to be executed in sequence based on event changes in the event list), and if CikA had a positive edge, the always statement AwA reads the current value of DataOut first and then the always statement AwB causes a new value to be assigned to DataOut. If the order of the always statements were reversed (or if a simulator chooses to reorder the execution of the always statements), execution of the always statement AwB occurs first causing DataOut to be assigned a new value in zero time. Subsequently, the read of DataOut in the always statement AwA uses the new updated value of DataOut. Thus it appears that depending on the order in which the always statements are executed, different results are obtained. The problem is really caused by the fact that when both always statements are ready for execution, the assignment to DataOut occurs in zero time and completes. So depending on which always statements gets executed first, the read of DataOut in AwA will either be an old value of DataOut or a new value of DataOut.

To avoid this simulation behavior dependence, it is best to force the assignment to occur at a later time, a time after which all reads are guaranteed to have been completed. This can be achieved by using the non blocking assignment. In such a case, the read of DataOut occurs at the current time, while a new value is assigned to DataOut at the end of the current simulation step (that is, after all reads are completed). This makes the behavior of the model insensitive to the order of the always state ments. Here are the always statements with non-blocking assignments used.

```
always @(posedge ClkA)      // Label AwA
  . . . = DataOut;          // Read value of DataOut.

always @(posedge ClkA)      // Label AwB
  DataOut <= . . .;       // Using non-blocking assignment.
```

So we see that if a variable is assigned a value in one always statement and its value read external to that always statement, the assignment should be a non-blocking procedural assignment.

What if the assignment and reading of a variable all occur in the same clocked always statement? In such a case, blocking assignments may be used.

```
reg [9:0] Total;

always @(negedge ClkB)
begin
  Total = LoadValue + 2;      // Blocking assignment.

  if (Total == 21)      // Value assigned in previous
                        // statement used.
    NumBus <= ControlBus;
  else
    NumBus <= DataBus;
end
```

Total is a variable assigned and then read within the same always statement. In this case, we would like the assignment to Total to be completed before the if condition is evaluated. Total is a temporary; a value is assigned to it and then read. Thus a blocking assignment is suitable for Total.

If a non-blocking assignment is used, such as:

```
reg [9:0] Total;

always @(negedge ClkB)
begin
  Total <= LoadValue + 2;     // Non-blocking assignment.

  if (Total == 21)            // Old value of Total read.
    NumBus <= ControlBus;
  else
    NumBus <= DataBus;
end
```

then the value of Total when the if condition is evaluated is the old value of Total, not the value that is scheduled to be assigned to it in the previous assignment.

Therefore, the recommendation is to use non-blocking assignments for variables that are read outside of the always statement in which they are assigned. Additionally, for variables that are assigned and used only within an always statement, use blocking assignments.

BGSIT - ACU 2020-21.

Module 5

MODELING EXAMPLES

In Module 3 and 4, we looked at the synthesis of Verilog HDL statements into gates. In this module we look at an orthogonal view, that is the task of modeling hardware elements for synthesis and how Verilog HDL can be used to achieve this. As before, we show both the Verilog HDL model and the schematic for the synthesized output.

This module also provides a number of more complicated Verilog HDL synthesis examples. These models illustrate the usage of Verilog HDL constructs collectively to model a design that can be synthesized.

Sequential logic and Combinational logic can be synthesized from a Verilog HDL description. There are two main styles for describing combinational logic:

1. Using continuous assignment statements: This is the most natural style, since it explicitly shows the parallelism in the hardware. It also implicitly shows the structure.
2. Using procedural assignment statements in a sequential block of an always statement: The statements describe the composition of intermediate values within a combinational logic block; this is because the language semantics specify that all statements in a sequential block execute sequentially.

Sequential logic elements, that is, flip-flops and latches, can be inferred by writing statements within an always statement using styles described in Module 3. It is best not to synthesize a memory as a two dimensional array of flip-flops because this is an inefficient way to implement a memory. The best way to create a memory is to instantiate a predefined memory block using a module instantiation statement.

5.1 Modeling Combinational Logic

One good approach for describing combinational logic is to use continuous assignment statements. An always statement can also be used to describe combinational logic; however, the synthesized logic may not be apparent from the description. If combinational logic is described using continuous assignment statements, then the synthesized logic is implicit in the description. Consider the following model of a built-in self-test cell.

B.S.BALAJI, ASST. PROF, DEPT. ECE, BGSIT – ACU. 2020-21

```
module BistCell (B0, B1, D0, D1, Z);
  input B0, B1, D0, D1;
  output Z;
  wire S1, S2, S3, S4;

  assign S1 = ~ (B0 & D1);
  assign S2 = ~ (D0 & B1);
  assign S3 = ~ (S2 | S1);
  assign S4 = S2 & S1;
  assign Z = ~ (S4 | S3);
endmodule
```

// Synthesized netlist is shown in Figure 5-1.

Notice the structure of the synthesized circuit is very similar to that of the continuous assignment statements.

Here is the same model, but this time the cell is described using an always statement.

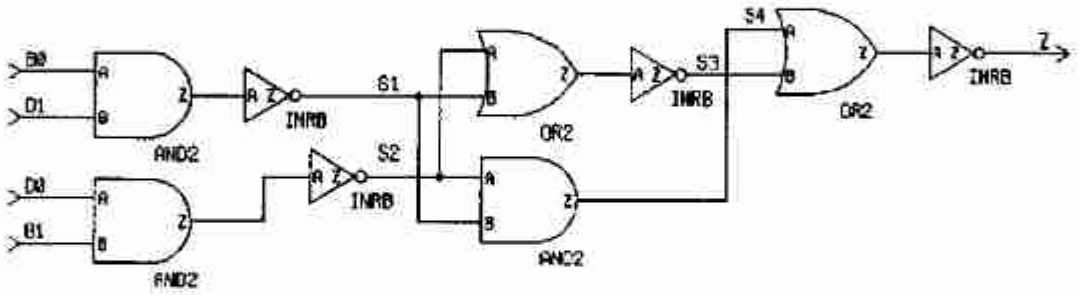


Figure 5-1 Combinational logic from continuous assignments.

```
module BistCellReg (B0, B1, D0, D1, Z);
  input B0, B1, D0, D1;
  output Z;
  reg Z;

  reg S1, S2, S3;

  always @(B0 or D0 or B1 or D1)
  begin
    S1 = ~ (B0 & D1);
    S2 = ~ (D0 & B1);
    S3 = ~ (S2 | S1);
    S1 = S2 & S1;
     Z = ~ (S1 | S3);
  end
endmodule
```

In module BistCell, each wire declared corresponded to a unique wire in the synthesized netlist. Not so with reg variable S1 in module BistCellReg. Notice that the variable SI is

used as a temporary in more than one place and does not represent one wire. The synthesized circuit still remains the same as that shown in Figure 5-1; however, the one-to-one mapping between the variables in the always statement and the nets in the synthesized netlist is not present.

Here is an example of a combinational logic model of a 2-to-1 multiplexer with an enable.

```
module Mux2To1 (A, B, Select, Enable, ZeeQ);
  input [1:0] A, B;
  input Select, Enable;
  output [1:0] ZeeQ;

  assign ZeeQ = (Enable) ? (Select ? A : B) : 'bz;
endmodule
```

//Synthesized netlist is shown in Figure 5-2.

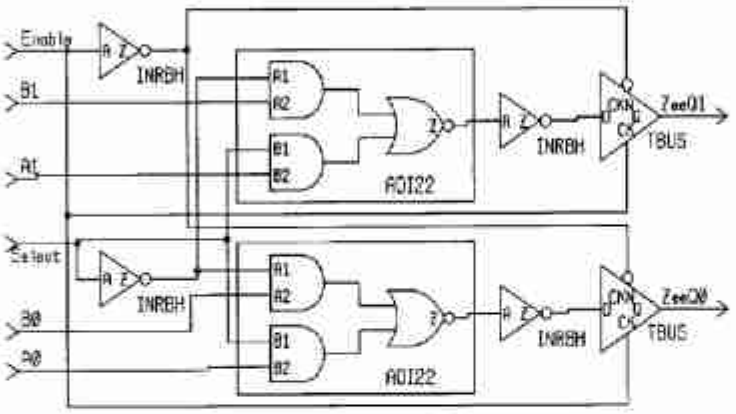


Figure 5-2 A 2-to-1 Mulitplexer.

5.2 Modeling Sequential Logic

5.2.1 Flip-flop

A flip-flop is inferred from a variable when it is assigned a value in a special form of always statement. This always statement is of the form:

```
always @ (<clock_event>)
  <statement>
```

where <*clock_event*> is one of:

```
posedge <clock_name>
negedge <clock_name>
```

The semantics of the always statement implies that all statements in <statement> are to be executed only when a rising edge or a falling edge of clock occurs. It is also known as clocked always statement.

When modeling sequential logic, it is recommended that a non-blocking procedural assignment be used for a variable that is assigned in a clocked always statement and its

value used outside of the always statement; this is to prevent any possibility of functional mismatch between the design model and its synthesized netlist. Such a target of a non-blocking assignment that appears in a clocked always statement accurately models the behavior of a sequential element.

Example

```
module PickOne (A, B, Clock, Control, Zee);
  input A, B, Clock, Control;
  output Zee;
  reg Zee;

  always @ (negedge Clock)
    if (Control)
      Zee <= A;
    else
      Zee <= B;
endmodule
```

// Synthesized netlist is shown in Figure 5.2.1 (A).

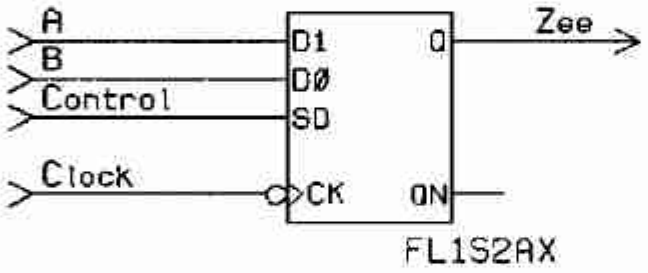


Figure 5.2.1 (A) Sequential logic synthesized using a special always statement.

The assignment to output Zee occurs only at the falling edge of the clock. Variable Zee is inferred to be a falling-edge-triggered flip-flop. The flip-flop shown in the figure has a data select, that is, the input Control selects A or B as the data for the flip-flop.

Example

```
module Incrementor (ClockA, Counter);
  parameter COUNTER_SIZE = 2;
  input ClockA;
  output [COUNTER_SIZE-1:0] Counter;
  reg [COUNTER_SIZE-1:0] Counter;

  always @ (posedge ClockA)
    Counter <= Counter + 1;
endmodule
```

// Synthesized netlist is shown in the Figure 5.2.1 (B).

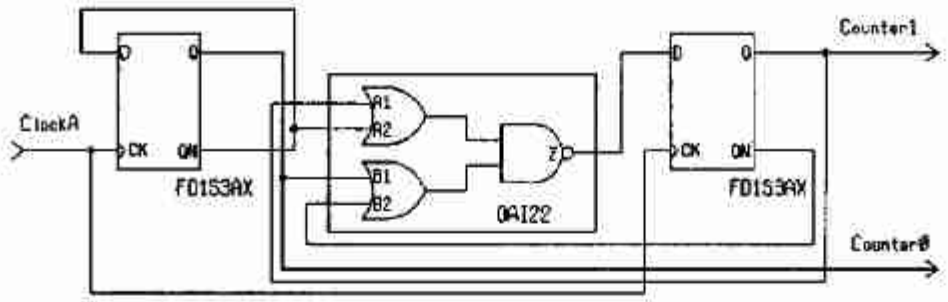


Figure 5.2.1 (B) Modeling of flip-flops

The logic in the always statement implies that every time there is a rising edge on ClockA, variable Counter is incremented. Since Counter is assigned under the control of a clock edge, rising-edge-triggered flip-flops are synthesized for Counter.

5.2.2 Flip-flop with asynchronous preset and or clear

To infer a flip-flop with asynchronous preset and clear? To generate such a flip-flop, a special form of if statement has to be used.

If a variable is assigned a value in any of the asynchronous and is also assigned in the synchronous part, that variable will get synthesized as a flip-flop with asynchronous preset and clear. Depending on the value being assigned, the flip-flop could either be aflip-flop with asynchronous preset (if a non-zero value is assigned), or a flip-flop with asynchronous clear (if a zero value is being assigned), or a flip-flop with both.

Here is an example of an up-down counter with asynchronous preset and clear.

```
module AsyncPreClrCounter (Clock, Preset, UpDown,
                           Clear, PresetData, Counter);
  parameter NUM_BITS = 2;
  input Clock, Preset, UpDown, Clear;
  input [NUM_BITS-1:0] PresetData;
  output [NUM_BITS-1:0] Counter;
  reg [NUM_BITS-1:0] Counter;

  always @ (posedge Preset or posedge Clear or
            posedge Clock)
    if (Preset)
      Counter <= PresetData;
    else if (Clear)
      Counter <= 0;
    else                    // Implicit posedge Clock.
    begin                   // Synchronous part.
      if (UpDown)
        Counter <= Counter + 1;
      else
        Counter <= Counter - 1;
    end
endmodule
```

// Synthesized netlist is shown in Figure 5.2.2

DEPT. ECE, BGSIT - ACU. 2020-21.

Having an asynchronous data input such as PresetData can cause a problem. Consider when Preset is 1 and then PresetData changes. The change of PresetData does not reflect in the Verilog HDL model while the changes propagate to Counter in the synthesized netlist. Avoid or be careful when using asynchronous data inputs.

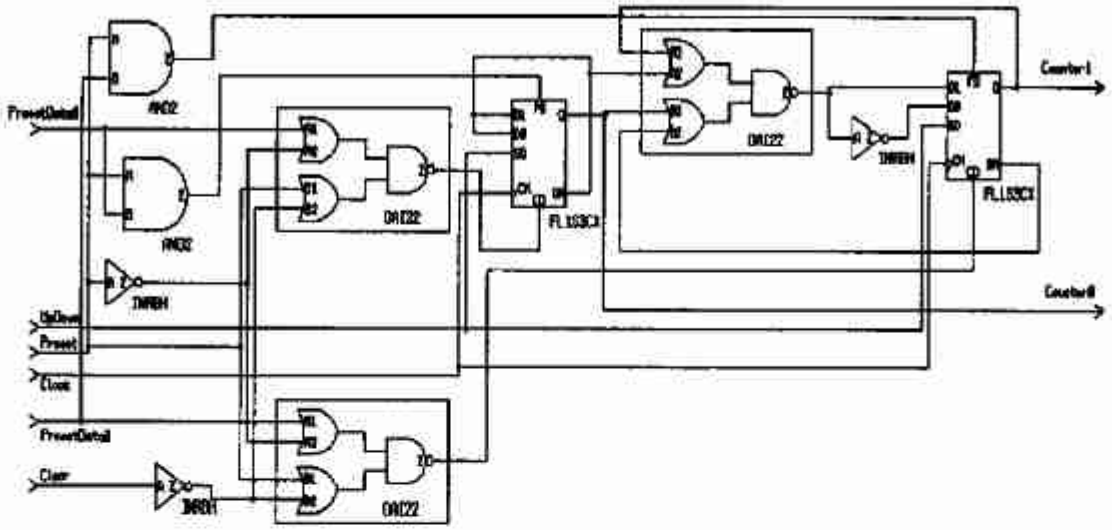


Figure 5.2.2. Flip-flops with asynchronous preset and clear.

5.2.3 Flip-flop with synchronous preset and or clear

To model a Flip-flop with synchronous preset and clear? In such a case, simply describe the synchronous preset and clear logic within a clocked always statement (an always statement with a clock event). Here is an example.

```
module SyncPresetCounter (Clock, Preset, UpDown,
                          PresetData, Counter);
  parameter NBITS = 2;
  input Clock, Preset, UpDown;
  input [0:NBITS-1] PresetData;
  output [0:NBITS-1] Counter;
  reg [0:NBITS-1] Counter;

  always @ (negedge Clock)
    if (Preset)

      Counter <= PresetData;
    else
      if (UpDown)
        Counter <= Counter + 1;
      else
        Counter <= Counter - 1;
endmodule
```

// Synthesized netlist is shown in Figure 5.2.3

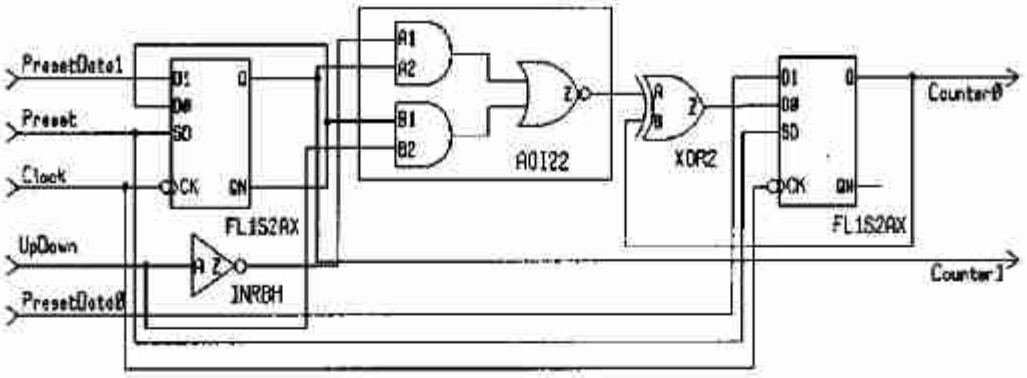


Figure 5.2.3. Flip-flops with synchronous preset and clear synthesized.

There are two approaches to synthesize this model. One approach is to direct the PresetData input into the synchronous preset input of the synthesized flip-flops; alternatively, the PresetData could be directed directly into the data input of the flip-flops. The synthesized netlist shown here shows the latter option; a synthesis system may optionally synthesize to the alternate approach.

5.2.4 Latch

A latch can be inferred by using an incompletely specified if statements or a case statement, that is, if a variable is not assigned a value in all branches of an if statement or a case statement, a latch is inferred for that variable. Here is an example.

```
module LatchExample (ClockA, CurrentState, NextState);
  input ClockA;
  input [3:0] CurrentState;
  output [3:0] NextState;
  reg [3:0] NextState;

  always @ (ClockA or CurrentState)
    if (ClockA)
      NextState = CurrentState;
endmodule
```

// Synthesized netlist is shown in Figure 5.2.4.

BALAJI B.S., ASST. PROF, DEPT. ECE, BGSIT - ACU, 2020-21.

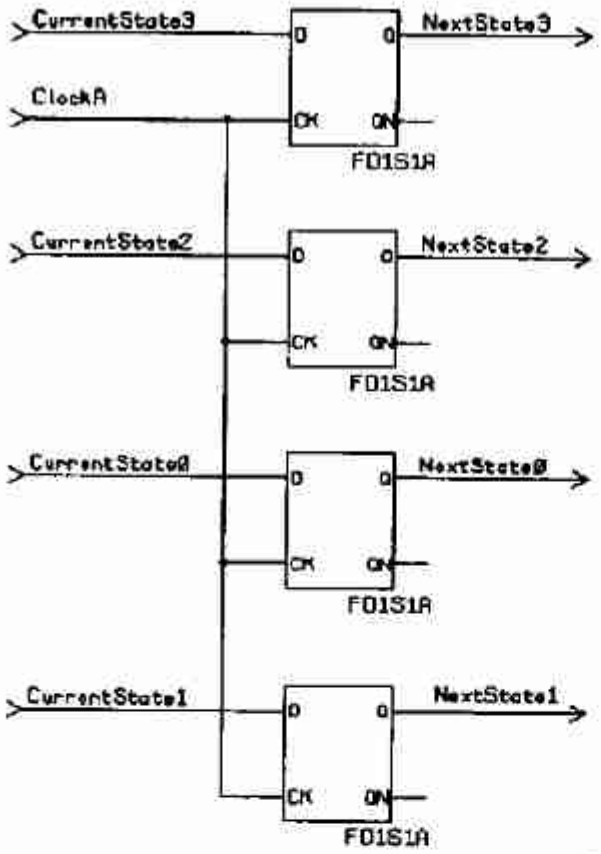


Figure 5.2.4 An incompletely specified condition infers a latch.

The variable NextState is assigned a value only when ClockA is 1. If ClockA is 0, NextState retains its previous value, thus inferring a latch.

5.2.5 Latch with asynchronous preset and or clear

If a variable that is inferred as a latch is assigned constant values in some branches of a conditional statement, bits that are 1 get assigned to the preset terminal of the latch, while those with 0 get assigned to the clear terminal. This is shown in the following example.

```
module AsyncLatch (ClockX, Reset, Set, CurrentState,
                   NextState);
  input ClockX, Reset, Set;
  input [3:0] CurrentState;
  output [3:0] NextState;
  reg [3:0] NextState;

  always @ (Reset or Set or ClockX or CurrentState)
    if (! Reset)
      NextState = 12;
    else if (! Set)
      NextState = 5;
    else if (! ClockX)
      NextState = CurrentState;
endmodule
```

// Synthesized netlist is shown in Figure 5.2.5

ASST. PROF, DEPT. ECE, BGSIT - ACU. 2020-21.

Four latches, with preset and clear terminals as appropriately required, are synthesized for NextState. A synthesis tool may alternately not generate a latch with asynchronous preset and clear, but direct the preset clear logic into the D-input of a simple latch.

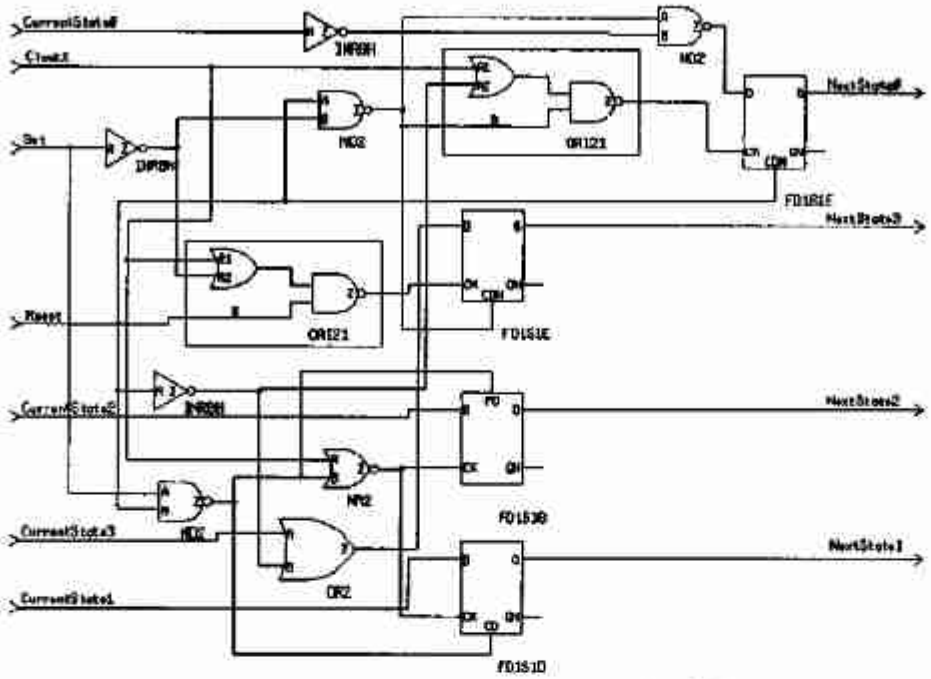


Figure-5.2.5 Latch with asynchronous preset and clear.

5.3 Modeling a Memory

A memory is best modeled as a component. Typically, synthesis tools are not efficient at designing a memory. More traditional techniques are generally used to build a memory. Once having built this module, it can then be instantiated in a synthesis model as a component using the module instantiation statement.

```
module ROM (Clock, OutEnable, Address, Q, Qbar);
  input Clock, OutEnable;
  input [M-1:0] Address;
  output [N-1:0] Q, Qbar;

  // Memory description here (might not be
  // synthesizable).
  . . .
endmodule

module MyModule ( . . . );
  wire Clk, Enable;
  wire [M-1:0] Abus;
  wire [N-1:0] Dbus;

  ROM R1 (.Clock(Clk), .OutEnable(Enable),
          .Address(Abus), .Q(Dbus), .Qbar());
  . . .
endmodule
```

A register file can be modeled as a two-dimensional reg variable (a two-dimensional reg variable is referred to as memory in Verilog HDL), which can then be synthesized. Here is an example of a register file.

```
module RegFileWithMemory (Clk, ReadWrite, Index,
                          DataIn, DataOut);
  parameter N = 2, M = 2;
  input Clk, ReadWrite;
  input [1:N] Index;      // Range need not be that large.
  input [0:M-1] DataIn;
  output [0:M-1] DataOut;
  reg [0:M-1] DataOut;

  reg [0:M-1] RegFile [0:N-1];

  always @ (negedge Clk)
    if (ReadWrite)
      DataOut <= RegFile[Index];
    else
      RegFile[Index] <= DataIn;
endmodule
```

// Synthesized netlist of a 2-by-2 register file is shown in Figure 5-3.

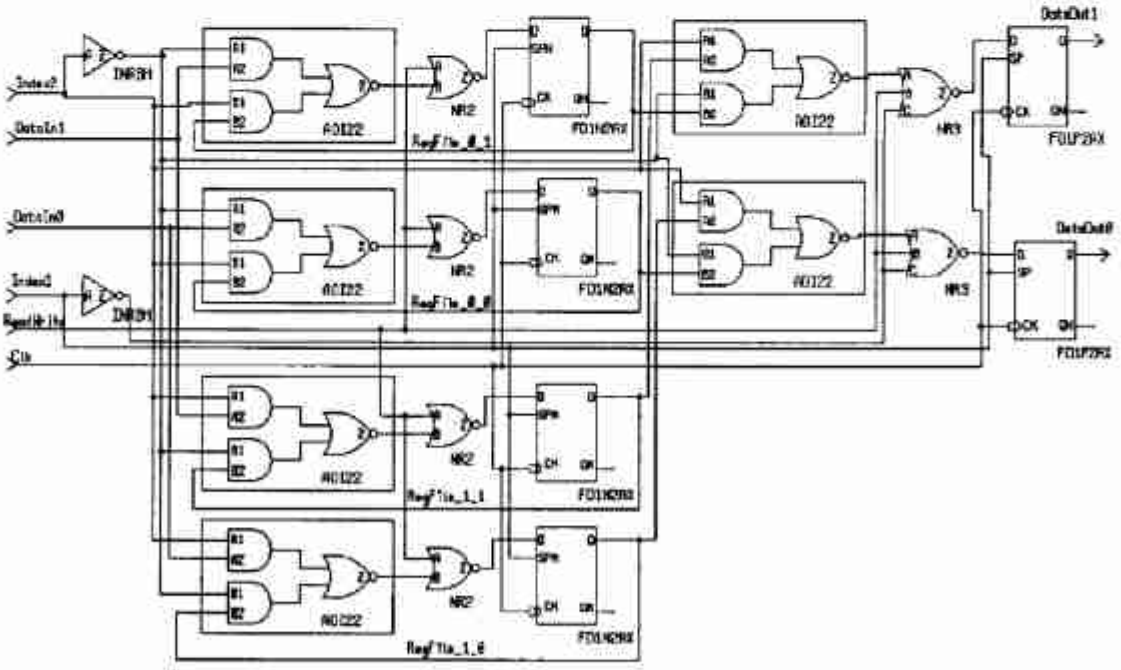


Figure 5-3 A 2-by-2 Register file.

Note that there are a total of six flip-flops synthesized, four for the register file Reg File and two for DataOut.

5.4 Writing Boolean Equations

Boolean equations represent combinational logic. Boolean equations are best represented using continuous assignment statements. Here is an example of a Gray code to binary code convertor using Boolean equations.

A B C	Binary Code
0 0 0	0 0 0
0 0 1	0 0 1
0 1 1	0 1 0
0 1 0	0 1 1
1 1 0	1 0 0
1 1 1	1 0 1
1 0 1	1 1 0
1 0 0	1 1 1

```
module GrayToBinary (A, B, C, Bc0, Bc1, Bc2);
  input A, B, C;
  output Bc0, Bc1, Bc2;
  wire NotA, NotB, NotC;

  assign NotC = ~ C;
  assign NotB = ~ B;
  assign NotA = ~ A;

  assign Bc0 = (A & B & NotC) | (A & B & C) |
               (A & NotB & C) | (A & NotB & NotC);
  assign Bc1 = (NotA & B & C) | (NotA & B & NotC) |
               (A & NotB & C) | (A & NotB & NotC);
  assign Bc2 = (NotA & NotB & C) | (NotA & B & NotC) |
               (A & B & C) | (A & NotB & NotC);
endmodule
```

//Synthesized netlist is shown in Figure 5-4.

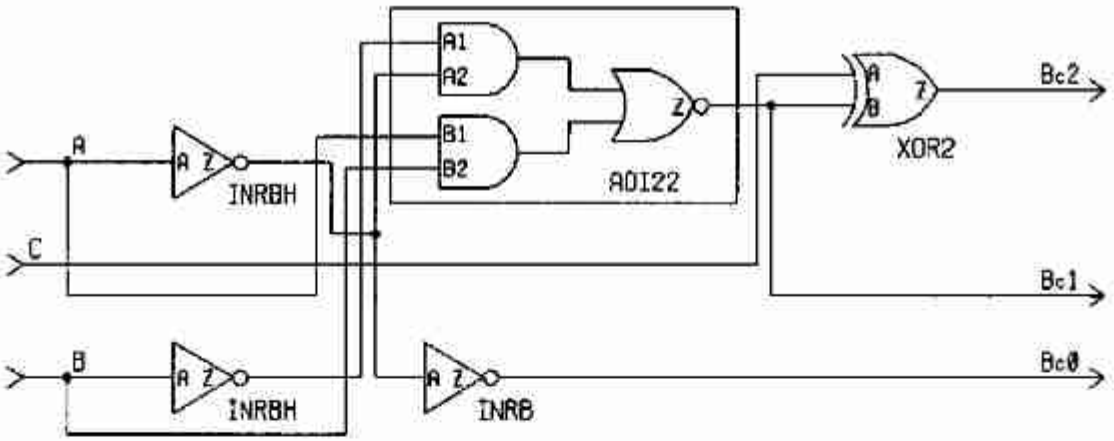


Figure 5-4 Gray to binary logic.

5.5 Modeling a Finite State Machine

5.5.1 Moore FSM

In a Moore finite state machine, the output of the circuit is dependent only on the state of the machine and not on its inputs. This is described pictorially in Figure 5-5. Since the outputs are dependent only on the state, a good way to describe a Moore machine is to use an always statement with a case statement. The case statement is used to switch between the various states and the output logic for each state is described in the appropriate branch. The always statement can have the clock event in its event list to indicate that it is a clocked always statement. This models the condition of a finite state machine going from state to state synchronously on every clock edge. The machine state itself is modeled using a reg variable (a variable of reg data type).

Here is an example of a Moore finite state machine. A reg variable MooreState is used to model the machine state which can have either of the four states. The event list indicates that the state transitions occur synchronously on every rising clock edge.

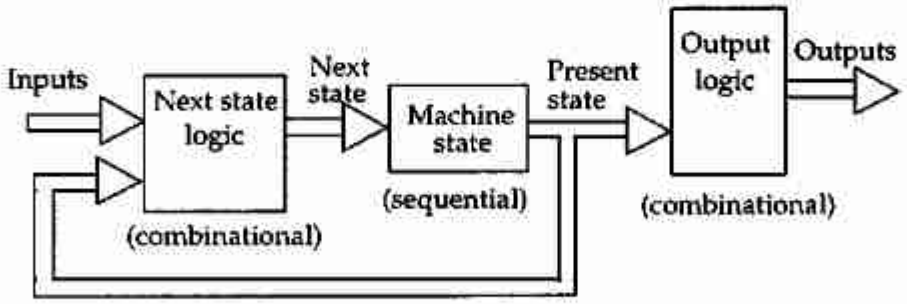


Figure 5-5 A Moore finite state machine.

```
module MooreFSM (A, ClkM, Z);
  input A, ClkM;
  output Z;
```

```
reg Z;

parameter S0 = 0, S1 = 1, S2 = 2, S3 = 3;
reg [0:1] MooreState;

always @ (posedge ClkM)
  case (MooreState)
    S0 :
      begin
        Z <= 1;
        MooreState <= (! A) ? S0 : S2;
      end
    S1 :
      begin
        Z <= 0;
        MooreState <= (! A) ? S0 : S2;
      end
    S2 :
      begin
        Z <= 0;
        MooreState <= (! A) ? S2 : S3;
      end
    S3:
      begin
        Z <= 1;
        MooreState <= (! A) ? S1 : S3;
      end
  endcase
endmodule
```

// Synthesized netlist for a Moore FSM is shown in Figure 5-6.

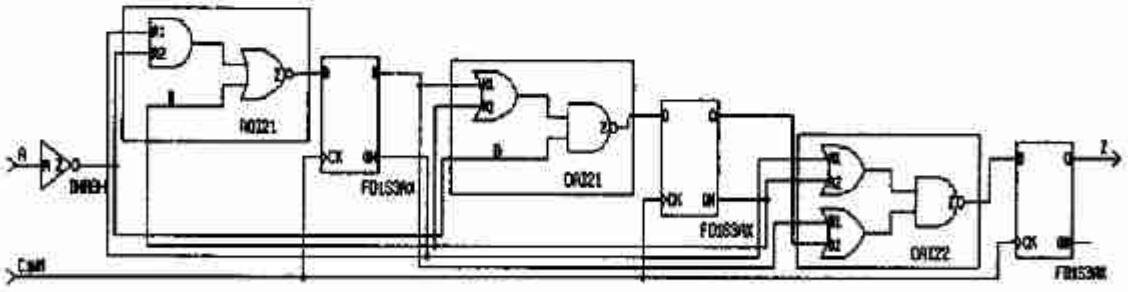


Figure 5-6 The synthesized netlist for the Moore FSM model.

When this model is synthesized, three flip-flops are inferred; two to hold the value of the machine state (MooreState) and one for the output Z. States are encoded using sequential state assignment.

In the previous example, the output is also saved in a flip-flop. What if a non-latched output is required? In this case, the assignments to Z can be separated out into a second always statement, as shown in the model next.

```
module MooreFSM2 (A, ClkM, Z);
  input A, ClkM;
  output Z;
  reg Z;

  parameter S0 = 0, S1 = 1, S2 = 2, S3 = 3;
  reg [0:1] MooreState;

  always @ (posedge ClkM)
    case (MooreState)
      S0 : MooreState <= (! A) ? S0 : S2;
      S1 : MooreState <= (! A) ? S0 : S2;
      S2 : MooreState <= (! A) ? S2 : S3;
      S3 : MooreState <= (! A) ? S1 : S3;
    endcase

  // Shows clearly that output is dependent on
  // only state.
  always @ (MooreState)
    case (MooreState)
      S0 : Z = 1;
      S1 : Z = 0;
      S2 : Z = 0;


      S3 : Z = 1;
    endcase
endmodule
```

// Synthesized netlist for a Moore FSM with no latched output is shown in Figure 5-7.

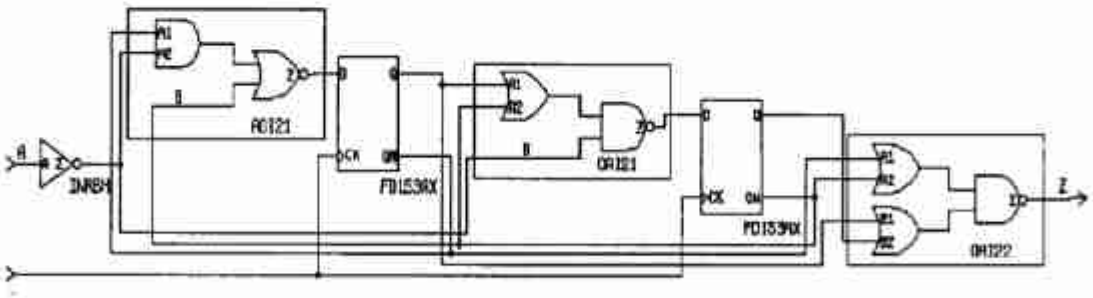


Figure 5-7 The synthesized netlist for the Moore FSM model with no latched output.

5.5.2 Mealy FSM

In a Mealy finite state machine, the output is dependent both on the machine state as well as on the inputs to the finite state machine. This is shown pictorially in Figure 5-8. Notice that in this case, outputs can change asynchronously with respect to clock.

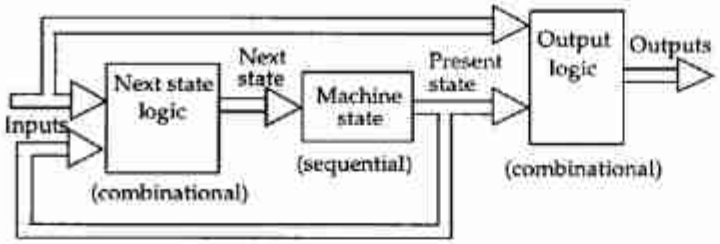


Figure 5-8 A Mealy finite sate machine.

One of the best ways of describing a Mealy finite state machine is by using two always statements, one for describing the sequential logic, and one for describing the combinational logic (this includes both next state logic and output logic). It is necessary to do this since any changes on inputs directly affect the outputs used to describe the combinational logic. The state of the machine is modeled using a reg variable.

Here is an example of a Mealy finite state machine. Variable MealyState holds the machine state, while NextState is used to pass information from the combinational logic always statement to the sequential logic always statement. Input Reset asynchronously resets the state ST0.

BALAJI B S, ASST. PROF, DEPT. ECE, BGSIT - ACU, 2020-21

```
module MealyFSM (A, ClkB, Reset, Z);
  input A, ClkB, Reset;
  output Z;
  reg Z;

  parameter ST0 = 4'b00, ST1 = 4'b01, ST2 = 4'b10;
  reg [0:1] MealyState, NextState;

  // Sequential logic:
  always @ (posedge Reset or posedge ClkB)
    if (Reset)
      MealyState <= ST0;
    else
      MealyState <= NextState;

  // Combinational logic:
  always @ (MealyState or A)
    case (MealyState)
      ST0 :
        begin
          Z = (A) ? 1 : 0;
          NextState = (A) ? ST2 : ST0;
        end
      ST1 :
        begin
          Z = (A) ? 1 : 0;
          NextState = (A) ? ST0 : ST1;
        end
      ST2 :
        begin
          Z = 0;
          NextState = (A) ? ST1 : ST2;
        end
      default : // default behavior; required, else

                // latches are inferred for Z and NextState.
        begin
          Z = 0; NextState = ST0;
        end
    endcase
endmodule
```

// Synthesized netlist of a Mealy FSM is shown in Figure 5-9.

- ACU. 2020-21.

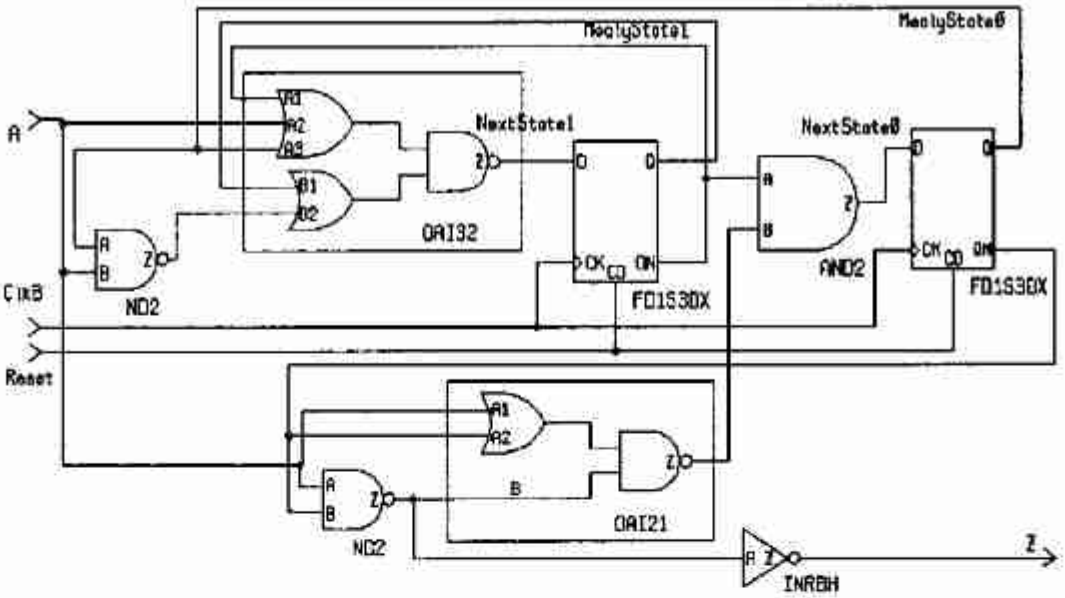


Figure 5-9 The synthesized netlist for the Mealy FSM example.

Two flip-flops are inferred to hold the value of the variable MealyState with the specified state assignment. The default branch in the case statement can be avoided by specifying the case statement as "full case", as shown next.

```
always @ (MealyState or A)
  case (MealyState)    // synthesis full_case
    ST0 :
      begin
        Z = (A) ? 1 : 0;
        NextState = (A) ? ST2 : ST0;
      end
    ST1 :
      begin

        Z = (A) ? 1 : 0;
        NextState = (A) ? ST0 : ST1;
      end
    ST2 :
      begin
        Z = 0;
        NextState = (A) ? ST1 : ST2;
      end
endcase
```

In this case, no latches are inferred for Z and NextState since the full_case synthesis directive states that no other case item values can occur. However, the preferred style for not inferring latches is to use the default branch.

Here is another example of a Mealy FSM, this one uses one-hot state encoding.

```
module MealyFSM2 (A, ClkC, Reset, Z);
  input A, ClkC, Reset;
  output Z;
  reg Z;

  parameter ST0 = 2'd0, ST1 = 2'd1, ST2 = 2'd2;
  reg [0:2] NextState, MealyState;

  // Sequential logic:
  always @ (posedge Reset or posedge ClkC)
    if (Reset)
    begin
      MealyState <= 0;
      MealyState[ST0] <= 1'b1;
    end
    else
      MealyState <= NextState;

  // Combinational logic:
  always @ (MealyState or A)
  begin
    NextState = 3'b0;        // Default assignment.
    Z = 1'b0;

    case (1'b1)
      MealyState[ST0]:

        if (A)
        begin
          Z = 1'b1;
          NextState[ST2] = 1'b1;
        end
        else
          NextState[ST0] = 1'b1;
      MealyState[ST1]:
        if (A)
        begin
          Z = 1'b1;
          NextState[ST0] = 1'b1;
        end
        else
          NextState[ST1] = 1'b1;
      MealyState[ST2]:
        if (A)
          NextState[ST1] = 1'b1;
        else
          NextState[ST2] = 1'b1;
    endcase
  end
endmodule
```

// Synthesized netlist of a Mealy FSM is shown in Figure 5-10.

BGSIT - ACU. 2020-21.

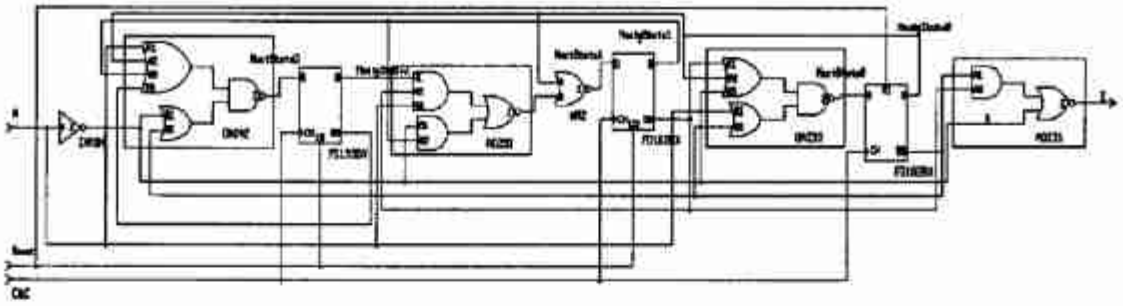


Figure 5-10 The synthesized netlist for the Mealy FSM example.

5.5.3 Encoding States

There are many ways to model the machine states of a finite state ma chine. Described here are some of the most common ones. The Moore FSM module described earlier is used as an example in describing these encodings.

Using Integers

The simplest way is to assign integer values to states.

```
integer MooreState;
. . .
case (MooreState)
   0 : . . .
    MooreState = 2;
    . . .
   1 :
    . . .
endcase
```

The problem with this approach is that since it is impractical to list all possible values an integer can take, to avoid latches either the default case branch must be specified or the full_case synthesis directive must be used. Another problem with this approach is not good readability.

Using Parameter Declarations

Another option is to declare parameters and use these in the case statement.

The advantage of this approach is that the state encodings are described explicitly in one place and can be changed easily. If either the parameter declarations or the integer values are used directly, a synthesis system uses the minimum number of bits needed to encode the integer value. In the above example, only two bits are needed for state encoding since the maximum integer value is 3.

What if a different encoding has to be specified? This can be done by describing each state as a vector of bits.

BALAJI B S, ASST. , ROF, DEPT. ECE, BGSIT - ACU. 2020-21.

```
parameter S0 = 0, S1 = 1, S2 = 2, S3 = 3;

reg [0:1] MooreState;
. . .
case (MooreState)
   S0 : . . .
    MooreState = S2;
    . . .
   S1 :
    . . .
endcase
```

In this case, the number of bits required for state encoding is dictated by the number of bits in the parameter which in this example is 3 bits. Of course, the machine state MooreState must be made wide enough to hold the new size of three bits.

5.6 Modeling an Universal Shift Register

Here is a synthesis model of a 3-bit universal shift register. The universal shift register performs the following functions:

i. hold value
ii. shift left
iii. shift right
iv. load value

This universal register can be used as a serial-in, serial-out shift register, parallel-in, serial-out shift register, serial-in, parallel-out shift register, and as a parallel-in, parallel-out shift register. Here is the state table for the 3-bit universal shift register.

Function	*Inputs* (*S0 S1*)		*Next state* (*Q*[2]	*Q*[1]	*Q*[0])
Hold	0	0	*Q*[2]	*Q*[1]	*Q*[0]
Shift left	0	1	*Q*[1]	*Q*[0]	*RightIn*
Shift right	1	0	*LeftIn*	*Q*[2]	*Q*[1]
Load	1	1	*ParIn*[2]	*ParIn*[1]	*ParIn*[0]

The synthesis model follows.

B.S.BALAJI, ASST. PROF, DEPT. ECE, BGSIT – ACU. 2020-21.

```
module UnivShiftRegister (Clock, Clear, LeftIn,
                          RightIn, S0, S1, ParIn, Q);
  input Clock, Clear, LeftIn, RightIn, S0, S1;
  input [2:0] ParIn;
  output [2:0] Q;
  reg [2:0] Q;

  always @(negedge Clear or posedge Clock)
    if (! Clear)
      Q <= 3'b000;
    else
      case ({S0, S1})
        2'b00 : ;
        2'b01 :
          Q <= {Q[1:0], RightIn};
        2'b10 :
          Q <= {LeftIn, Q[2:1]};
        2'b11 :
          Q <= ParIn;
      endcase
endmodule
```

//Synthesized netlist of Universal Shift Register is shown in Figure 5-11.

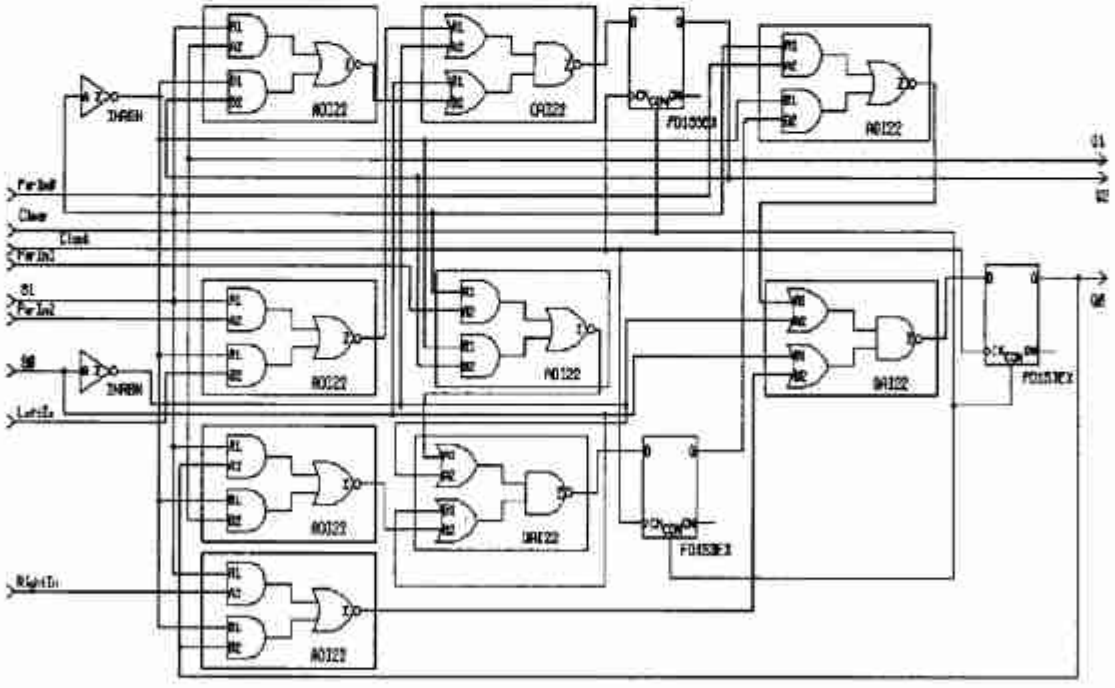


Figure 5-11 A 3-bit Universal Shift Register.

5.7 Modeling an ALU

5.7.1 A Parameterized ALU

Here is an example of a parameterized N-bit arithmetic-logic-unit that performs an exclusive-or, less than, and an increment-by-1 operation.

```
module ArithLogicUnit (A, B, Select, CompareOut,
                       DataOut);
  parameter N = 2;
  input [N-1:0] A, B;
  input [2:0] Select;
  output CompareOut;
  output [N-1:0] DataOut;
  reg CompareOut;
  reg [N-1:0] DataOut;

  parameter OP_XOR = 3'b001, OP_INCRA = 3'b010,
            OP_LT = 3'b100;

  always @ (A or B or Select)
    case (Select)
      OP_INCRA :
        begin
          DataOut = A + 1;


          CompareOut = 'bx;
        end
      OP_XOR :
        begin
          DataOut = A ^ B;
          CompareOut = 'bx;
        end
      OP_LT :
        begin
          CompareOut = A < B;
          DataOut = 'bx;
        end
      default :
        begin
          CompareOut = 'bx;
          DataOut = 'bx;
        end
    endcase
endmodule
```

// Synthesized netlist of a Parameterized ALU is shown in Figure 5-12.

A different size ALU can be synthesized by specifying a different value for the parameter when it is instantiated. This is shown in the following example for a 4-bit ALU.

ACU. 2020-21. EPT. ECE,

```
module FourBitALU (A, B, Sel, Cmp, Data);
  parameter ALU_SIZE = 4;
  input [ALU_SIZE-1:0] A, B;
  input [2:0] Sel;
  output Cmp;
  output [ALU_SIZE-1:0] Data;

  ArithLogicUnit #(ALU_SIZE) InstA (A, B, Sel,
                                    Cmp, Data);
endmodule
```

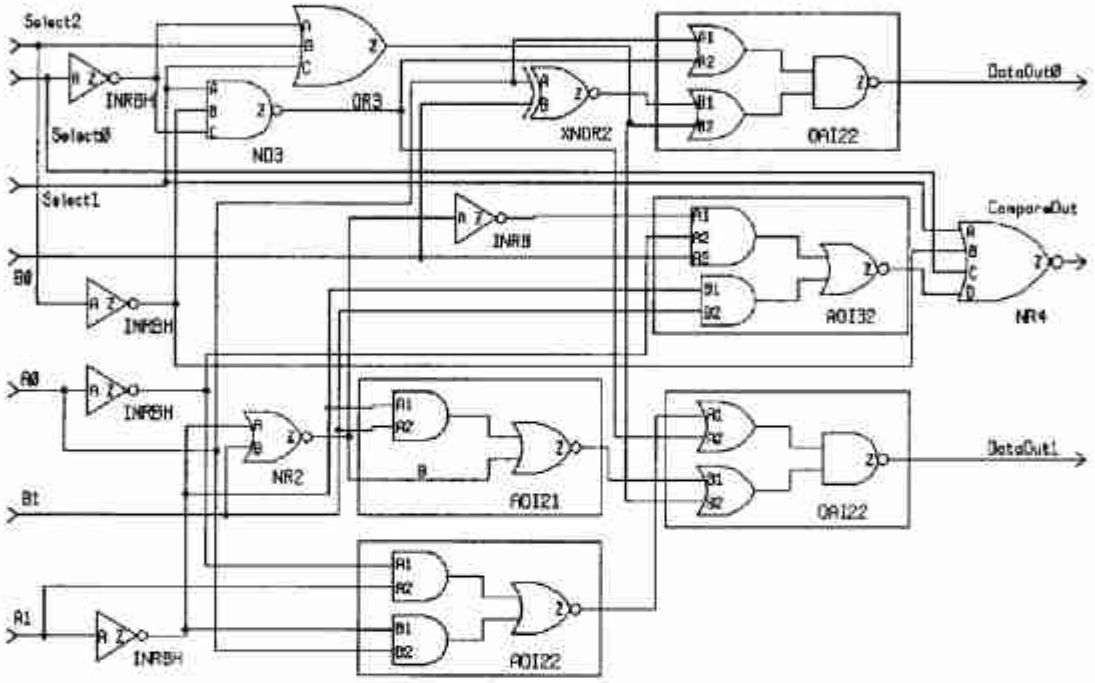


Figure 5-12 A 2-bitALU.

5.7.2 A Simple ALU

Here is a model of a different simple arithmetic-logic-unit. This logic unit performs four functions: add, nand, greater-than and exclusive-or. A continuous assignment statement with a conditional expression is used to model the arithmetic-logic-unit.

```
module CustomALU (A, B, OpCode, DataZ, CompZ);
  parameter NBITS = 2;
  input [NBITS-1:0] A, B;
  input [1:0] OpCode;
  output [NBITS-1:0] DataZ;
  output CompZ;
  parameter ADD_OP = 0, NAND_OP = 1, GT_OP = 2,
            XOR_OP = 3;

  assign DataZ =(OpCode == ADD_OP) ? A + B :
                (OpCode == NAND_OP) ? ~ (A & B) :
                (OpCode == XOR_OP) ? A ^ B :
                'bx;

  assign CompZ = (OpCode == GT_OP) ? A > B : 'bx;
endmodule
```

// Synthesized netlist of a simple ALU is shown in Figure 5-13.

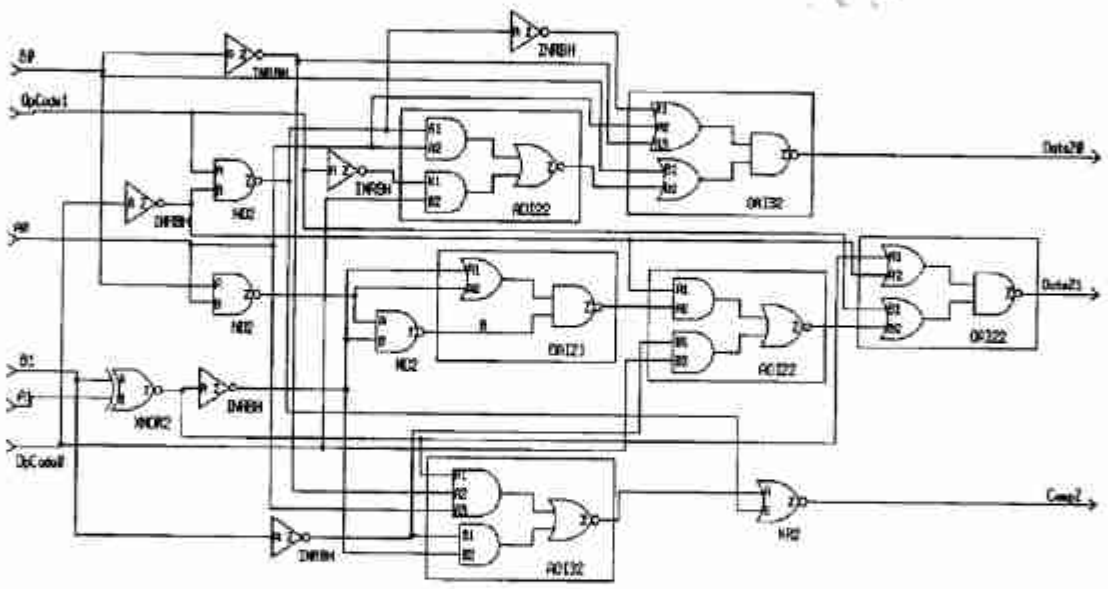


Figure 5-13 A 2-bit custom ALU.

5.8 Modeling a Counter

5.8.1 Binary Counter

Here is a model for a parameterized N-bit binary up-down counter with synchronous preset and preclear controls. The counting is synchronized to the rising edge of a clock.

```
module BinaryCounter (Ck, UpDown, PresetClear,
                      LoadData, DataIn, Q, QN);
  parameter NBITS = 2;
  input Ck, UpDown, PresetClear, LoadData;
  input [NBITS-1:0] DataIn;
  output [NBITS-1:0] Q;
  output [NBITS-1:0] QN;

  reg [NBITS-1:0] Counter;


   always @ (posedge Ck)
     if (PresetClear)
       Counter <= 0;
     else if (~ LoadData)
       Counter <= DataIn;
     else if (UpDown)
       Counter <= Counter + 1;
     else
       Counter <= Counter - 1;

   assign Q = Counter;
   assign QN = ~ Counter;
endmodule
```

// Synthesized netlist of a 2-bit binary counter is shown in Figure 5-14.

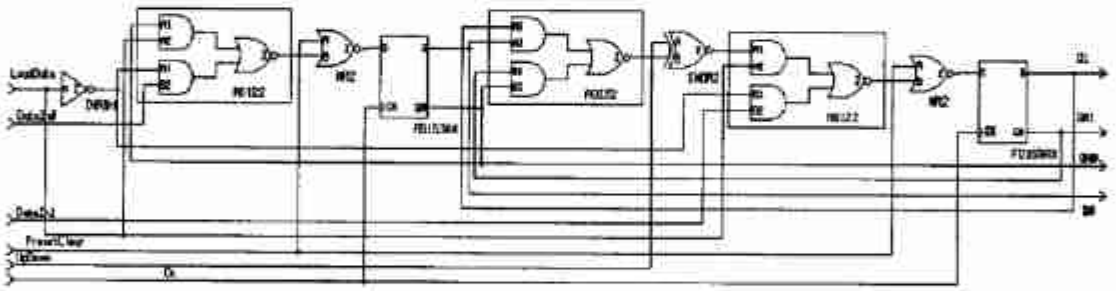


Figure 5-14 A 2-bit up-down, loadable, clearable binary counter.

5.8.2 Modulo-N Counter

Here is a model of a modulo-N binary up-counter. This counter has only a synchronous preclear control and all transitions occur on the rising clock edge.

```
// Number of bits in counter: NBITS
// Modulo: UPTO
module ModuloN_Cntr (Clock, Clear, Q, QBAR);
  parameter NBITS = 2, UPTO = 3;
  input Clock, Clear;
  output [NBITS-1:0] Q, QBAR;
  reg [NBITS-1:0] Counter;
```

B.S.BALAJI, ASST. PROF, DEPT. ECE, BGSIT - ACU. 2020-21.

```
    always @ (posedge Clock)
      if (Clear)
        Counter <= 0;
      else
        Counter <= (Counter + 1) % UPTO;

    assign Q = Counter;
    assign QBAR = ~ Counter;
  endmodule
```

// Synthesized netlist for a modulo-3 counter is shown in Figure 5-15.

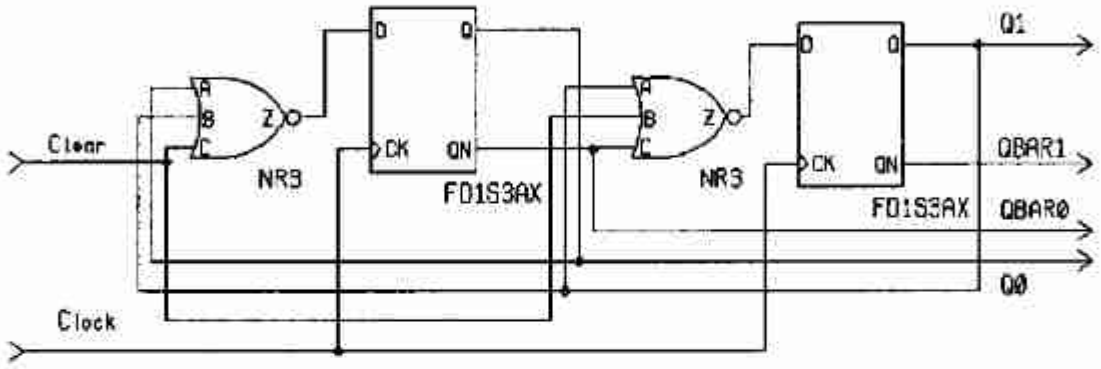


Figure 5-15 A modulo-3 binary counter.

5.8.3 Johnson Counter

A Johnson counter is a shift-type counter. Here is an example of a 3- A bit Johnson counter stream.

```
000
001
011
111
110
100
000
```

The keys to modeling a Johnson counter are:

i. If the most significant bit (the leftmost bit) of the counter is a 1, then a 0 has to be shifted in from the right.

ii. If the most significant bit is a 0, then a 1 has to be shifted in from the right.

Here is the model for a parameterized N-bit Johnson counter with an asynchronous preclear control.

```
module JohnsonCounter (ClockJ, PreClear, Q);
  parameter NBITS = 3;
  input ClockJ, PreClear;
  output [1:NBITS] Q;
  reg [1:NBITS] Q;

  always @ (negedge PreClear or negedge ClockJ)
    if (! PreClear)
      Q <= 0;
    else
    begin
      if (! Q[1])
        Q <= {Q[1:NBITS-1], 1'b1};
      else
        Q <= {Q[1:NBITS-1], 1'b0};
    end
endmodule
```

// Synthesized netlist for a 3-bit Johnson counter is shown in Figure 5-16.

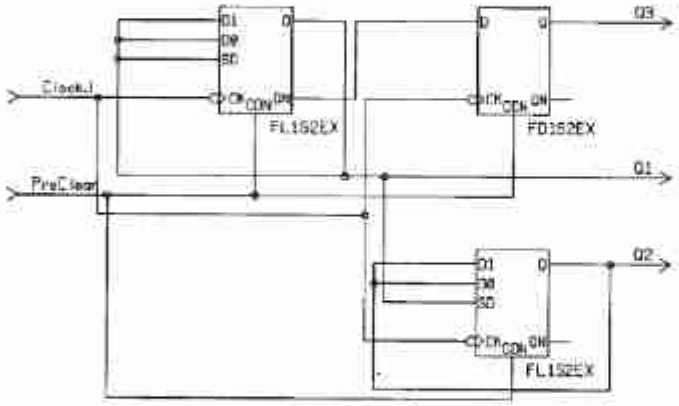


Figure 5-16 A 3-bit Johnson counter.

5.8.4 Gray Counter

A Gray counter is a binary counter with the following conversion logic:

i. The first Gray bit (the leftmost bit) is the same as the first bi nary bit.

ii. The second Gray bit is determined by xor'ing the second binary bit with the first binary bit, and so on; that is, each pair of adjacent bits are xor'ed to get the next Gray bit.

For example, a binary count of 4'b1100 corresponds to a Gray count of 4'b1010. Here is a Verilog HDL model for a parameterized N-bit Gray up counter with synchronous preclear.

BALAJI B S, ASST. . . BGSIT - ACU. 2020-21.

```
module GrayCounter (ClockG, Clear, Q, QN);
  parameter NBITS = 3;
  input ClockG, Clear;
  output [1:NBITS] Q, QN;

  reg [1:NBITS] Counter, GrayCount;
  integer K;

  always @ (posedge ClockG)

   if (Clear)
     Counter <= 0;
   else
     Counter <= Counter + 1;

 always @ (Counter)
 begin
   GrayCount[1] = Counter[1];

   for (K = 2; K <= NBITS; K = K+1)
     GrayCount[K] = Counter[K-1] ^ Counter[K];
 end

 assign Q = GrayCount;
 assign QN = ~ GrayCount;
endmodule
```

// Synthesized netlist for a 3-bit Gray counter is shown in Figure 5-17.

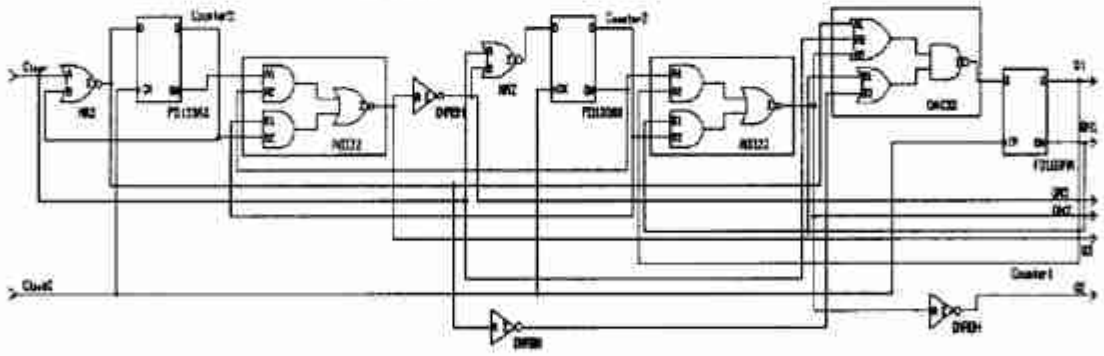


Figure 5-17 A 3-bit Gray counter.

5.9 Modeling a Parameterized Adder

Here is a model for a parameterized N-bit adder with carry input and carry output. The leftmost bit is the most significant bit. Figure 5-18 shows the synthesized netlist for a 3-bit adder with carry-in and carry-out.

```
module AddWithCarryInCarryOut (OpdA, OpdB, CarryIn,
                               CarryOut, Sum);
  parameter NUMBITS = 3;
  input [NUMBITS:1] OpdA, OpdB;
  input CarryIn;
  output CarryOut;
  output [NUMBITS:1] Sum;

  assign {CarryOut, Sum} = OpdA + OpdB + CarryIn;
endmodule
```

//Synthesized netlist for a 3-bit parameterized adder is shown in Figure 5-18.

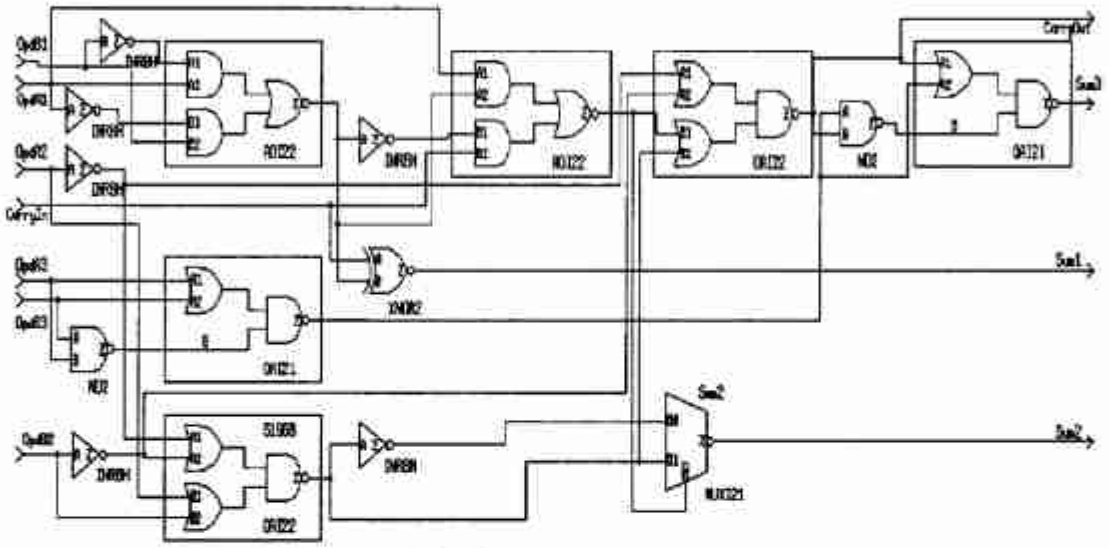


Figure 5-18 A 3-bit adder with carry-in and carry-out.

5.10 Modeling a Parameterized Comparator

Here is a model for a parameterized N-bit binary comparator. The input vectors are treated as unsigned quantities and a numerical comparison is made.

BALAJI B S, ASST. PROF.

```
module Comparator (A, B, EQ, GT, LT, NE, GE, LE);
  parameter NUMBITS = 2;
  input [NUMBITS:1] A, B;
  output EQ, GT, LT, NE, GE, LE;

  reg [5:0] ResultBus;
  // Bit 5 is EQ, bit 4 is GT, 3 is LT, 2 is NE,
  // 1 is GE and 0 is LE.

  always @ (A or B)
    if (A == B)
      ResultBus = 6'b100011;
    else if (A < B)
      ResultBus = 6'b001101;
    else // (A > B)
      ResultBus = 6'b010110;

  assign {EQ, GT, LT, NE, GE, LE} = ResultBus;
endmodule
```

// Synthesized netlist for a 2-bit comparator is shown in Figure 5-19

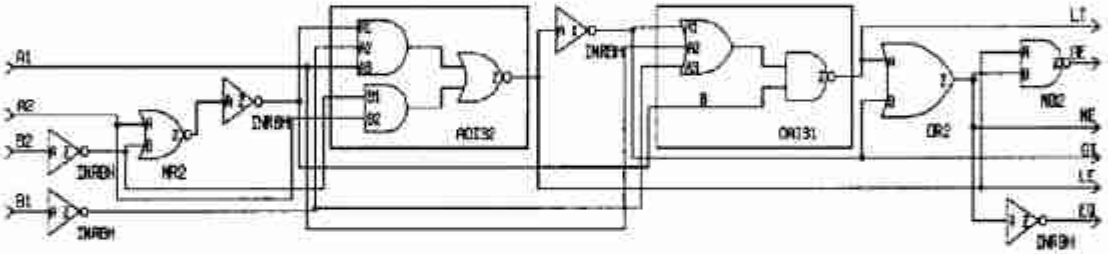


Figure 5-19 A 2-bit comparator

5.11 Modeling a Decoder

5.11.1 A Simple Decoder

Here is an example of a simple 2-by-4 decoder circuit. This is a combinational circuit modeled purely using continuous assignment statements. Delays specified with the assignment statements, if any, are typically ignored by a synthesis system.

```
module SimpleDecoder (A, B, Enable, DecodeOut);
  input A, B, Enable;
  output [0:3] DecodeOut;
  wire Abar, Bbar;

  assign Abar = ~ A;
  assign Bbar = ~ B;
  assign DecodeOut[0] = ~ (Enable & Abar & Bbar);
  assign DecodeOut[1] = ~ (Enable & Abar & B);
  assign DecodeOut[2] = ~ (Enable & A & Bbar);
  assign DecodeOut[3] = ~ (Enable & A & B);
endmodule
```

// Synthesized netlist is shown in Figure 5-20.

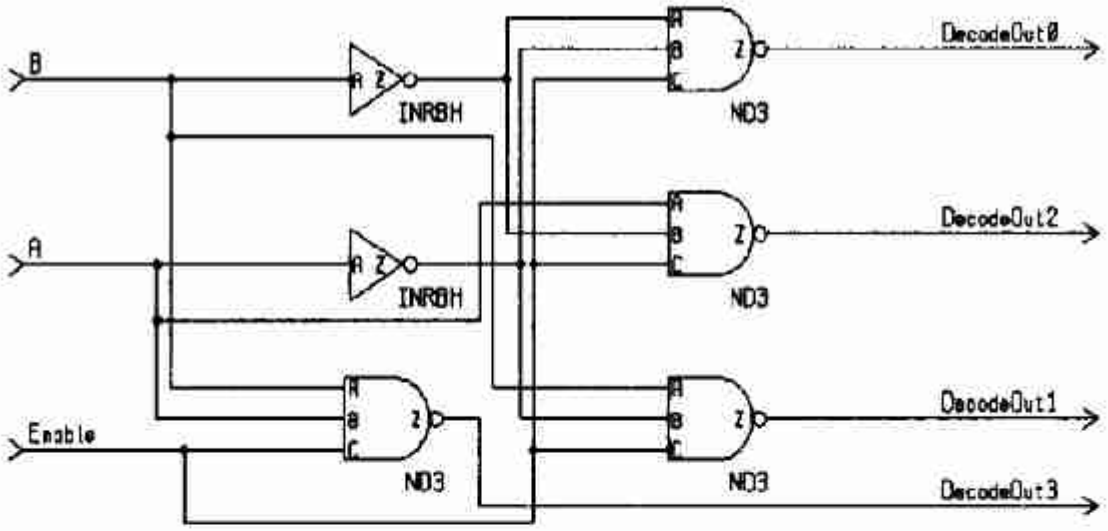


Figure 5-20 A Simple 2-by-4 Decoder.

5.11.2 Binary Decoder

Here is a model of a parameterized N-bit binary decoder.

```
module BinaryDecoder (SelectAddress, DecodeOut);
  parameter SBITS = 2;
  parameter OUT_BITS = 4; // Should be 2 to power of SBITS
  input [SBITS-1:0] SelectAddress;
  output [OUT_BITS-1:0] DecodeOut;
  reg [OUT_BITS-1:0] DecodeOut;

  integer k;

  always @ (SelectAddress)
    for (k = OUT_BITS - 1; k >= 0; k = k - 1)
      DecodeOut[k] = (k == SelectAddress) ? 'b1 : 'b0;
endmodule
```

// Synthesized netlist of a 2-bit binary decoder is shown in Figure 5-21.

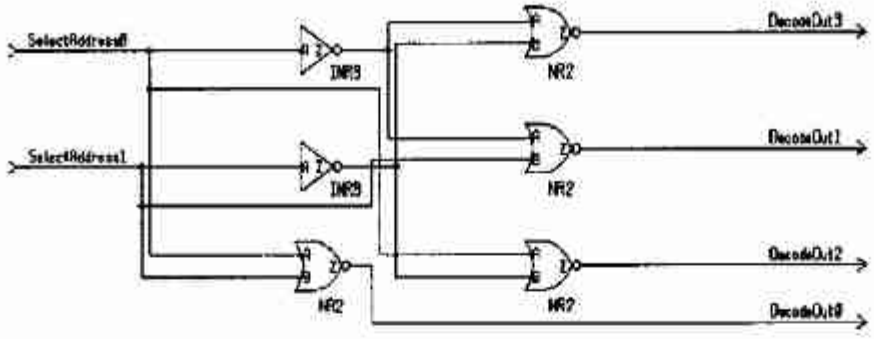


Figure 5-21 A 2-bit binary decoder.

5.11.3 Johnson decoder

Here is a model of a parameterized N-bit Johnson decoder with an enable control.

```
module JohnsonDecoder (S, En, Y);
  parameter N = 3;
  input [0:N-1] S;
  input En;
  output [0:2*N-1] Y;
  reg [0:2*N-1] Y;

  reg [0:2*N-1] Address;
  integer J;


   always @ (S or En)
    if (En == 'b1)
    begin
      Address = 0;

      for (J = 0; J < N; J = J + 1)
        if (S[J])
          Address = Address + 1;

      if (S[0])
        Address = 2*N - Address;

      Y = 'b0;
      Y[Address] = 'b1;
    end
    else if (En == 'b0)
      Y = 'b0;
    else
      Y = 'bx;
endmodule
```

// Synthesized netlist for a 3-bit Johnson decoder is shown in Figure 5-22.

BGSIT - ACU. 2020-21.

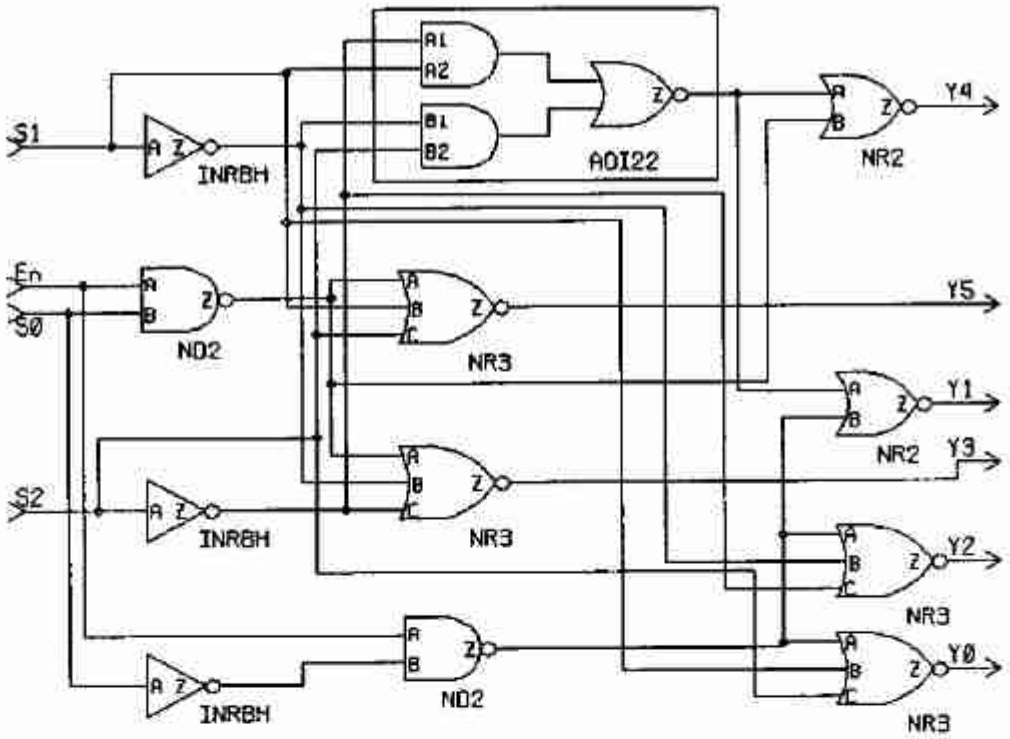


Figure 5-22 A 3-bit Johnson decoder.

5.12 Modeling a Multiplexer

5.12.1 A Simple Multiplexer

Here is a model of a 4-by-1 multiplexer circuit. In this case, a bit select in a continuous assignment statement has been used to model the combinational logic.

```
module SimpleMultiplexer (DataIn, SelectAddr, MuxOut);
  input [0:3] DataIn;
  input [0:1] SelectAddr;
  output MuxOut;
  assign MuxOut = DataIn[SelectAddr];
 endmodule
```

Synthesized netlist is shown in Figure 5-23.

BALAJI, ASST. PROF, DEPT. ECE, BGSIT

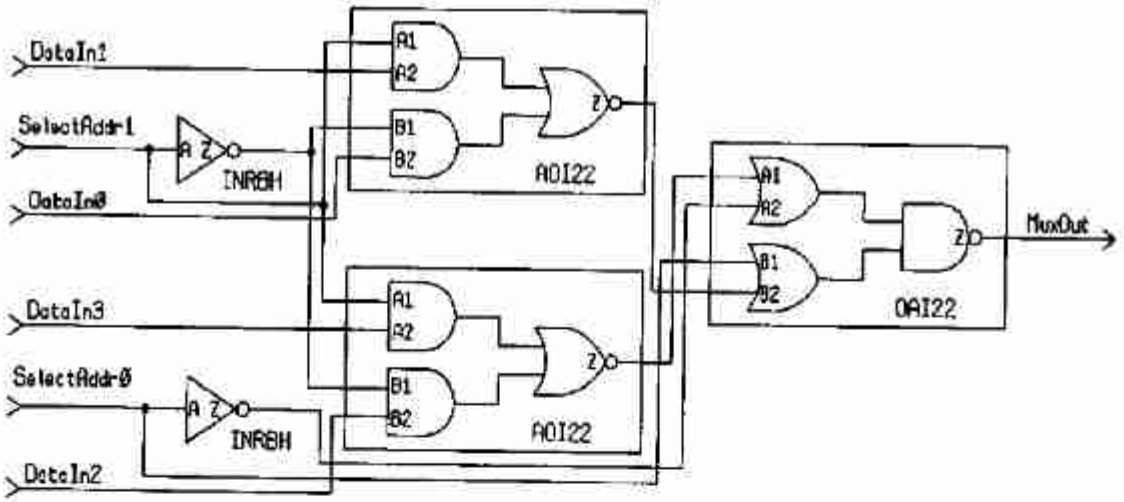


Figure 5-23 A 4-by-1 Multiplexer.

5.12.2 A Parameterized Multiplexer

Here is a model of a parameterized multiplexer. The number of bits per word and the number of words in the multiplexer are modeled as parameters. The input data lines are represented as a single array DataBus. The multiplexer has select lines which are non-encoded, and an enable signal. Inverted outputs are also provided. All outputs are three-state'ed based on the Enable condition.

```
module BinaryMultiplexer (DataBus, Select, Enable,
                          Y, Ybar);
  parameter NBITS = 2, WORDS = 2;
  input [NBITS * WORDS-1:0] DataBus;
    // DataBus is a linearized 2D array.
  input [WORDS-1:0] Select;     // Decoded select lines.
  input Enable;
  output [NBITS-1:0] Y, Ybar;
  reg [NBITS-1:0] Y, Ybar;

  integer K;

  function [WORDS-1:0] GetWordIndex;
```

```
    // Gets the first index that has value 1.
    input [WORDS-1:0] DecodedSelect;
    integer Inx;
    begin
      GetWordIndex = 0;

      for (Inx = WORDS - 1; Inx >= 0; Inx = Inx - 1)
        if (DecodedSelect[Inx] == 'b1)
          GetWordIndex = Inx;
    end
  endfunction

  always @ (DataBus or Select or Enable)
    if (Enable == 'b1)
    begin
      for (K = 0; K < NBITS; K = K + 1)
        Y[K] = DataBus[GetWordIndex(Select) * NBITS
                       + K];

      Ybar = ~ Y;
    end
    else if (Enable == 'b0)
    begin
      Y = 'bz;
      Ybar = 'bz;
    end
    else
    begin
      Y = 'bx;
      Ybar = 'bx;
    end
endmodule
```

// Synthesized netlist of a 2- by-2 multiplexer is shown in Figure 5-24.

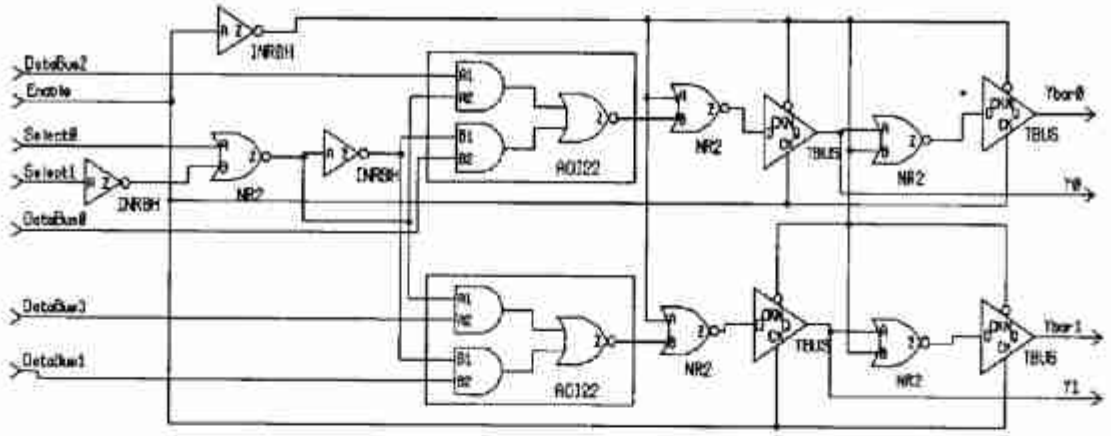


Figure 5-24 A 2-by-2 multiplexer.

5.13 Modeling a Parameterized Parity Generator

Here is a model of a parameterized N-bit parity generator circuit. The model provides both an odd parity and an even parity output.

```
module ParityGenerator (DataIn, OddPar, EvenPar);
  parameter NBITS = 4;
  input [NBITS-1:0] DataIn;
  output OddPar, EvenPar;

  assign EvenPar = ^ DataIn;
  assign OddPar = ~ EvenPar;
endmodule
```

// Synthesized netlist of a 4-bit parity generator is shown in Figure 5-25.

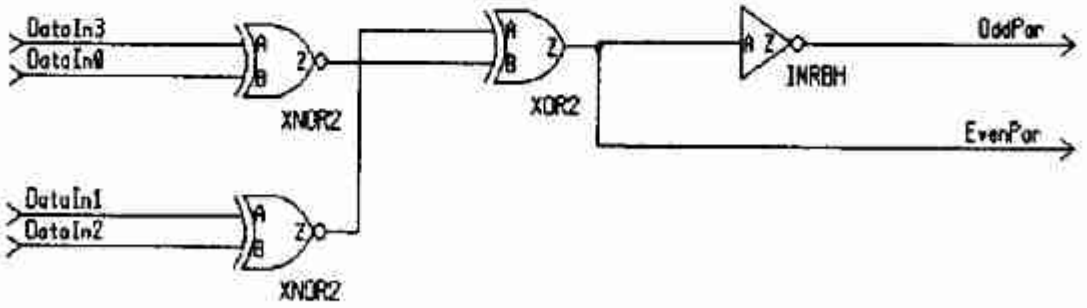


Figure 5-25 A 4-bit parity generator.

5.14 Modeling a Three-state Gate

A three-state gate is modeled by assigning the value z to a variable under the control of a condition. Here is an example.

```
module ThreeStateGates (ReadState, CpuBus, MainBus);
  input ReadState;
  input [0:3] CpuBus;
  output [0:3] MainBus;
  reg [0:3] MainBus;

  always @ (ReadState or CpuBus)
    if (ReadState)
      MainBus = 4'bz;
    else
      MainBus = CpuBus;
endmodule
```

//Synthesized netlist shown in Figure 5-26

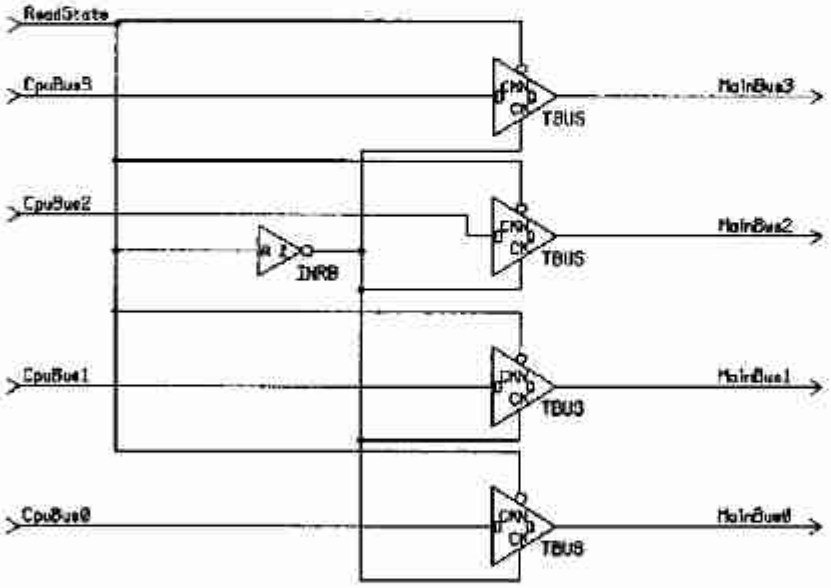


Figure 5-26 A bank of three-state gates.

The variable MainBus is three-state'ed as long as ReadState is true. If ReadState is false, the value of CpuBus is assigned to MainBus.

BGSIT - ACU. 2020-21.

5.15 A Count three 1's model

Here is a model that detects three 1's in a data stream appearing on input Data. The input is checked on every falling edge of clock. If three consecutive 1's are found on the input, the output is set to true, else it is set to false.

```
module Count3Ones (Data, Clock, Reset, SeqFound);
  input Data, Clock, Reset;
  output SeqFound;
  reg SeqFound;

  parameter PATTERN_SEARCHED_FOR = 3'b111;
  reg [2:0] Previous;

  always @ (negedge Clock)
    if (Reset)
    begin
      Previous <= 3'b000;
      SeqFound <= 1'b0;
    end
    else
    begin
      Previous <= {Previous[1:0], Data};
      SeqFound <= (Previous == PATTERN_SEARCHED_FOR);
    end
endmodule
```

//Synthesized netlist shown in Figure 5-27

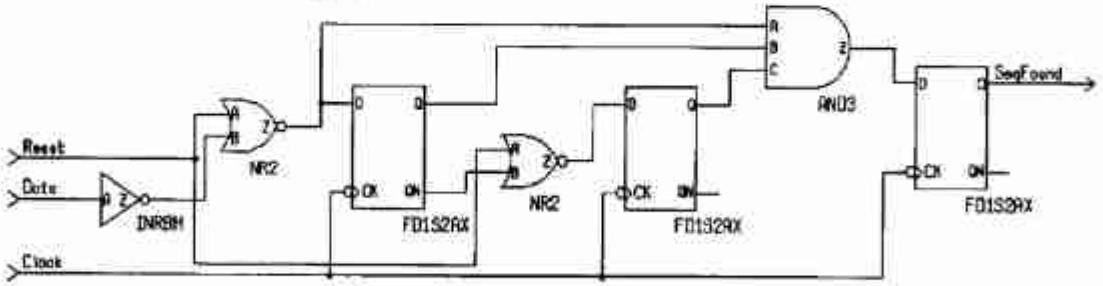


Figure 5-27 Circuit counts three consecutive ones.

Synthesis infers four flip-flops for this model, three for variable Previous and one for SeqFound. However, optimization reveals that one of the flip-flops for Previous is not necessary and hence it is removed. In this model, the output is latched since it is assigned a value under the control of a clock edge. If a latched output is not desired, then the assignment to SeqFound must be done outside the always statement. Such a module is shown next.

```
module NoLatchedOutput (Data, Clock, Reset, SeqFound);
  input Data, Clock, Reset;
  output SeqFound;

  parameter PATTERN_SEARCHED_FOR = 3'b111;
  reg [2:0] Previous;

  always @ (negedge Clock)
    if (Reset)
      Previous <= 3'b000;
    else
      Previous <= {Previous[1:0], Data};

  assign SeqFound = (Previous == PATTERN_SEARCHED_FOR);
endmodule
```

//Synthesized netlist shown in Figure 5-28

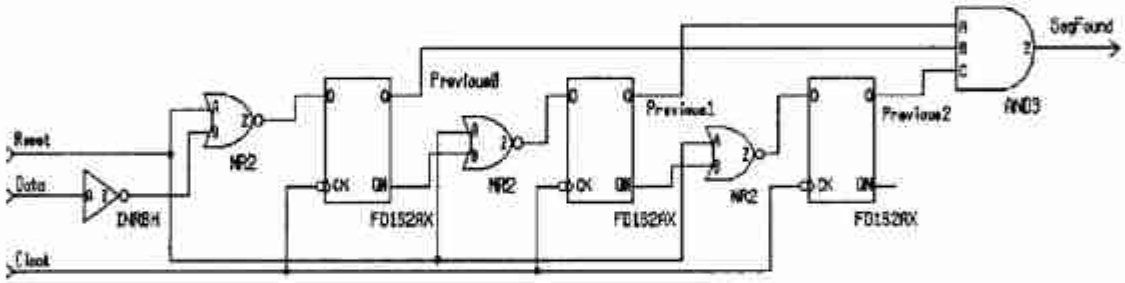


Figure 5-28 No latched output.

In this module, the output SeqFound is not latched. Synthesis infers three flip-flops for variable Previous; note that in this case, all the bits of Previous have to be latched.

BALAJI B S, ASST. PROF, BGSIT

5.16 A Factorial Model

Here is a model that generates the factorial of a number given in Data. The result is output in Result and Exponent as mantissa and exponent respectively. The exponent is base 2. The input Reset causes the model to reset.

```
module Factorial (Reset, Start, Clk, Data, Done,
                  Result, Exponent);
  input Reset, Start, Clk;
  input [4:0] Data;
  output Done;                    // Acknowledge signal.
  reg Done;
  output [7:0] Result, Exponent;
  reg [7:0] Result, Exponent;

  reg [4:0] InLatch;

  always @ (posedge Clk)
  begin: BLOCK_A
    integer NextResult, J;

    if ((Start && Done) || Reset)
    begin
      Result <= 'b1;
      Exponent <= 'b0;
      InLatch <= Data;
      Done <= 'b0;
    end
    else
    begin
      if ((InLatch > 1) && (! Done))
      begin
        NextResult = Result * InLatch;
        InLatch <= InLatch - 1;
      end
      else
        NextResult = Result;

      if (InLatch <= 1)
        Done <= 'b1;
```

```
        for (J = 1; J <= 5; J = J + 1)
        begin
          if (NextResult > 256)
          begin
            NextResult = NextResult >> 1;
            Exponent <= Exponent + 1;
          end
        end

        Result <= NextResult;
      end
    end
  endmodule
```

B.S.BALAJI, ASST. PROF, DEPT. ECE, BGSIT

When synthesized, flip-flops are inferred for Inlatch, Result, Exponent and Done.

5.17 An UART Model

Here is a model of a synthesizable UART circuit. This circuit converts RS-232 serial input data into parallel data out, and the parallel input data into RS-232 serial data out. The data byte is 8 bits in length. There are four major blocks in this UART model, as shown in Figure 5-29: RX, the receiver block, TX, the transmitter block, DIV, the clock divider and MP, the microprocessor block.

The first block DIV is a frequency divider. This block has 2 modes of operation, the normal mode and the test mode. In the test mode, the UART chip runs 16 times faster than in the normal mode. Also, the transmission data rate of the UART chip is times faster than the receiving rate. Each block is initialized by setting the reset line low by applying a 0 to port MR. The TX block accepts 8-bit parallel data from the microprocessor interface (MP) block and transmits it serially to the RS-232 port through port DOUT. Conversely, the RX block receives serial data input, and sends it in 8-bit parallel format to the MP block. Again, the transmitter runs at 16 times the speed of the receiver. The microprocessor interface (MP) block asynchronously controls the parallel data flow between the RX / TX blocks and the microprocessor data bus.

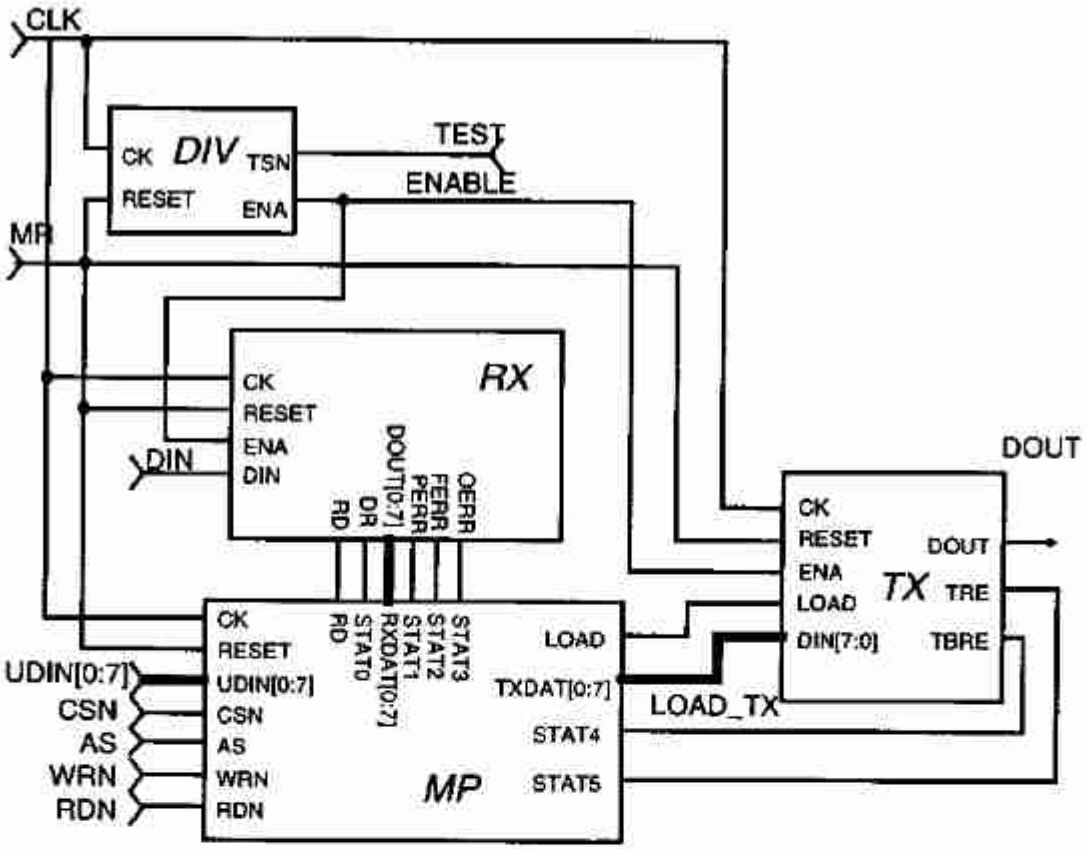


Figure 5-29 The UART circuit

The UART top-level model glues all these blocks together using module instantiations. The microprocessor entity, MP, is described in the structural style, that is, using module instances. The remaining three are described using the behavioral style. In this module, only the behavioral blocks are described.

Here is the behavioral model for the transmitter block TX. This model is a synthesizable model. Rising-edge-triggered flip-flops are inferred for variables TBR, TR, TRE, TBRE, DOUT, CBIT and PA; this is because these variables are assigned values under the control of clock CK.

```
module TX (CK, RESET, ENABLE, TLOAD, DIN, DOUT,
           TRE, TBRE);
  input CK, RESET, ENABLE, TLOAD;
  input [7:0] DIN;
  output DOUT, TRE, TBRE;
```

```
reg DOUT, TRE, TBRE;

reg [7:0] TBR, TR;
reg [3:0] CBIT;
reg PA;

always @ (posedge CK)
begin
  if (! RESET)
  begin
    TRE <= 'b1;
    TBRE <= 'b1;
    DOUT <= 'b1;
    CBIT <= 4'b0;
    PA <= 'b0;
  end
  else if (TLOAD)
  begin
    TBR <= DIN;
    TBRE <= 'b0;
  end
  else if (ENABLE)
  begin
    if (! TBRE && TRE)
    begin
      TR <= TBR;
      TRE <= 'b0;
      TBRE <= 'b1;
    end

    if (! TRE)
      case (CBIT)
        4'b0000:
          begin
            DOUT <= 'b0;
            CBIT <= CBIT + 1;
          end
        4'b0001, 4'b0010, 4'b0011, 4'b0100,
        4'b0101, 4'b0110, 4'b0111, 4'b1000:
          begin
            DOUT <= TR[0];
            PA <= PA ^ TR[0];
            TR <= {1'b1, TR[7:1]};
```

2020-21.

```
            CBIT <= CBIT + 1;
          end
        4'b1001:
          begin
            DOUT <= PA;
            PA <= 'b0;
            TR <= {1'b1, TR[7:1]};
            CBIT <= CBIT + 1;
          end
        4'b1010:
          begin
            DOUT <= TR[0];
            TR <= {1'b1, TR[7:1]};
            CBIT <= CBIT + 1;
          end
        4'b1011:
          begin
            DOUT <= TR[0];
            TRE <= 1'b1;
            TR <= {1'b1, TR[7:1]};
            CBIT <= 4'b0000;
          end
      endcase
    end                    // if (ENABLE)
  end                      // @(posedge CK)
endmodule
```

Here is the behavioral model for the receiver block RX. This model is also synthesizable. Flip-flops are inferred for variables START, CBIT, CSAM, DI, PI, SR, DR, DOUT, PERR, FERR and OERR.

```
module RX (CK, RESET, ENA, DIN, RD, DR, DOUT,
           PERR, FERR, OERR);
  input CK, RESET, ENA, DIN, RD;
  output DR;
  reg DR;
  output [7:0] DOUT;
  reg [7:0] DOUT;
  output PERR, FERR, OERR;
  reg PERR, FERR, OERR, START;

  reg [3:0] CBIT, CSAM;
```

```
reg DI, PI;
reg [7:0] SR;

always @ (posedge CK)
begin
  if (! RESET)
  begin
    CBIT <= 0;
    CSAM <= 0;
    START <= 0;
    PI <= 0;
    DR <= 0;
    PERR <= 0;
    FERR <= 0;
    OERR <= 0;
  end
  else                                  // if (RESET)
  begin
    if (RD)
      DR <= 0;

    if (ENA)
      if (! START)
      begin
        if (! DIN)
        begin
          CSAM <= CSAM + 1;
          START <= 1;
        end
      end
      else if (CSAM == 8)
      begin
        DI <= DIN;
        CSAM <= CSAM + 1;
      end
      else if (CSAM == 15)
        case (CBIT)
          0:
            begin
              if (DI == 1)
                START <= 0;
              else
                CBIT <= CBIT + 1;
```

```
            CSAM <= CSAM + 1;
          end
        1, 2, 3, 4, 5, 6, 7, 8:
          begin
            CBIT <= CBIT + 1;
            CSAM <= CSAM + 1;
            PI <= PI ^ DI;
            SR  <= {DI, SR[7:1]};
          end
        9:
          begin
            CBIT <= CBIT + 1;
            CSAM <= CSAM + 1;
            PI <= PI ^ DI;
          end
        10:
          begin
            PERR <= PI;
            PI <= 0;

            if ( ! DI)
              FERR <= 1;
            else
              FERR <= 0;

            if (DR)
              OERR <= 1;
            else
              OERR <= 0;

            DR <= 1;
            DOUT <= SR;
            CBIT <= 0;
            START <= 0;
          end
      endcase
      else  // ((0 <= CSAM < 8) || (8 < CSAM < 15))
        CSAM <= CSAM + 1;
    end                 // if (! RESET)
  end                   // @(posedge CK)
endmodule
```

Here is a synthesizable model for the divider block DIV. This circuit produces a pulse every sixteen clock cycles. If input TESTN is 0, ENA is set to a 1. Variable COUNT is inferred as flip-flops.

```
module DIV (CK, RESET, TESTN, ENA);
  input CK, RESET, TESTN;
  output ENA;

  reg [3:0] COUNT;

  always @(posedge CK)
    if (! RESET)
      COUNT <= 0;
    else if (! TESTN)
      COUNT <= 4'hF;
    else
      COUNT <= COUNT + 1;          // Increment counter.

  // Combinational part:
  assign ENA = (COUNT == 15);
endmodule
```

5.18 A Blackjack model

Here is a synthesizable model of a blackjack program. This program is played with a deck of cards. Cards 2 to 10 have values equal to their face value, and an ace has a value of either 1 or 11. The object of the game is to accept a number of random cards such that the total score (sum of values of all cards) is as close as possible to 21 without exceeding 21.

The input InsertCard indicates when the program is ready to accept a new card. A card is accepted at the rising edge of Clock if InsertCard is true. Input Card Value has the value of the card. If a sequence of cards is accepted such that the total falls between 17 and 21, then output Won is set to true, indicating that the game has been won. If total exceeds 21, then the program checks to see if an ace was accepted as a 1 or a 11; if it was accepted as a 11, the value of ace is changed to 1 and the program gets ready to accept a new card; if not, output Lost is set to true indicating that it has lost. If either Won or Lost is set, no more cards are accepted. The game can be reset by setting NewGame to true.

```
module Blackjack (CardValue, Clock, InsertCard,
                  NewGame, TotalPoints, Won, Lost);
  input [0:3] CardValue;
  input Clock, InsertCard, NewGame;
  output [0:5] TotalPoints;
  output Won, Lost;
  reg Won, Lost;

  reg AceAvailable, AceValueIs11;
  reg [0:5] TotPts;
  parameter TRUE = 1'b1, FALSE = 1'b0;

  always @ (posedge NewGame or posedge Clock)
    if (NewGame)
    begin
      Won <= FALSE;
      Lost <= FALSE;
      AceAvailable = FALSE;
      AceValueIs11 = FALSE;
      TotPts = 0;
    end
    else       // posedge Clock
      if (InsertCard && ! Won && ! Lost)
      begin
        if (CardValue == 4'd11)
        begin
          AceAvailable = TRUE;
          AceValueIs11 = TRUE;
        end

        TotPts = TotPts + CardValue;

        if ((TotPts >= 17) && (TotPts <= 21))
          Won <= TRUE;
        else if ((TotPts >= 22) && (TotPts <= 31))
        begin
          if (AceAvailable && AceValueIs11)
          begin
            AceValueIs11 = FALSE;
            TotPts = TotPts - 10;
          end
          else
            Lost <= TRUE;


        end
      end

  assign TotalPoints = TotPts;
endmodule
```

ACU. 2020-21.

Object oriented Programming Using C++

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

Module - 1

Beginning With C++ and its features:

What is C++

C++ is a general-purpose programming language It has imperative, object-oriented and generic programming features, while also providing facilities for Low-level memory manipulation.

It was developed by Bjarne Stroustrup in 1980's

OOP:- object-oriented programming refers to a type of computer programming (S/w design) in which programmers define not only the data type of a data structure, but also the types of operations(functions) that can be applied to the data structure.

Application of C++

* C++ is a Versatile language for handling very large programs.
* It is Suitable for Virtually any programing task Including development of Editors, Compilers, database, Communication Systems & any Complex real-life Application System
* C++ allows us to Create hierarchy-related objects. We can build Special object-oriented libraries which can be used later by many programmers.
* C++ is able to map the real-world problem properly

* C++ programs are easily maintainable & Expandable.

* A Simple C++ program

```
#include <iostream> // Include header file
    using name space std;
     int main()
    {
       Cout << "C++ is better than C.\n"; // C++ Statement
        return 0;
    } // End of Example
```

In above example contains only one function i.e main(). i.e Execution begins at main().

Comments:- Comments start with a double slash sym-bol & terminate at the end of the line.

double slash comment is basically a single line comment.

/*, */ used for multiline comment.

Output operator:- Cout & <<

Cout is a predefined object that represents the stan-dard o/p stream in C++.

IoStream file:-

C++ program typically contains pre-processor directive Statements at the beginning. Such Statement are precee-ded with a # Symbol to Indicate the presence of a pre-processor directive to the compiler

* All C++ programs begin with a #include directive that include the specified header file contents into main program.

Namespace:.

This defines a scope for the identifiers that are used in a program.

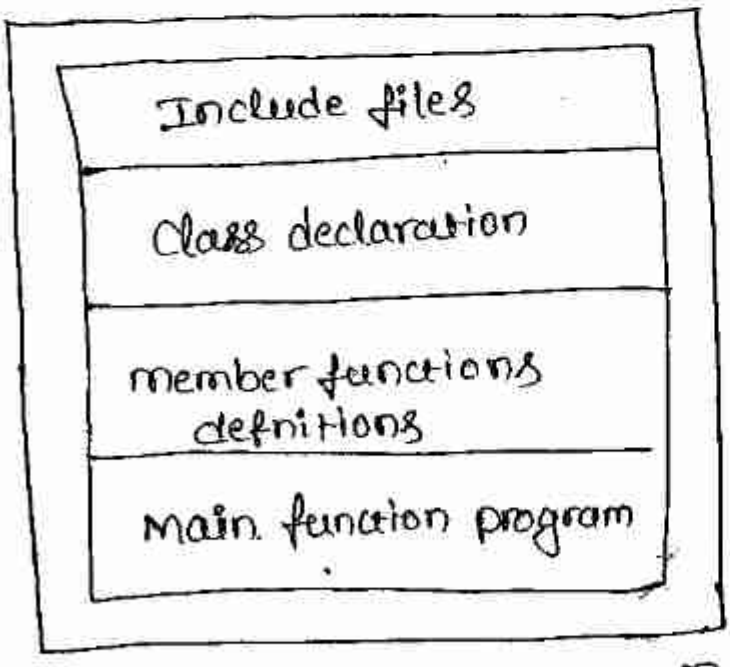
Using namespace std;

Std is the namespace where C++ Std class libraries are defined.

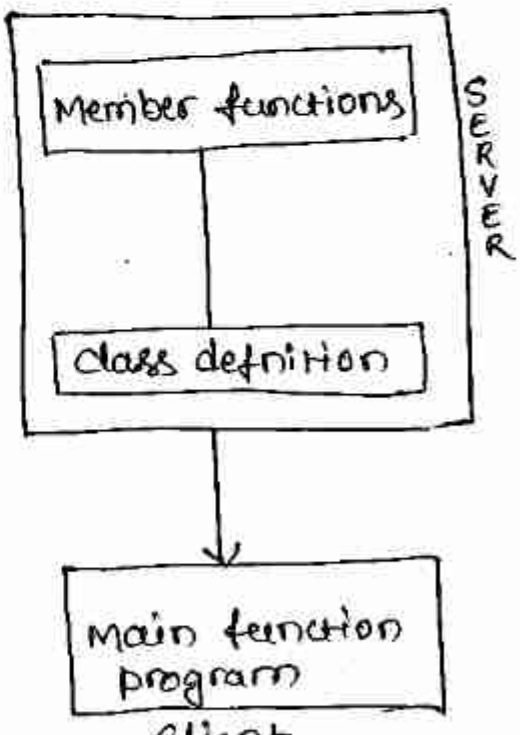
Return type of main()

In C++ main() returns an integer type value to the operating system. ∴ Every main() in C++ should end with a return (0) statement. Otherwise a warning (or) an error might occur.

Structure of C++ program



Structure of a C++ program

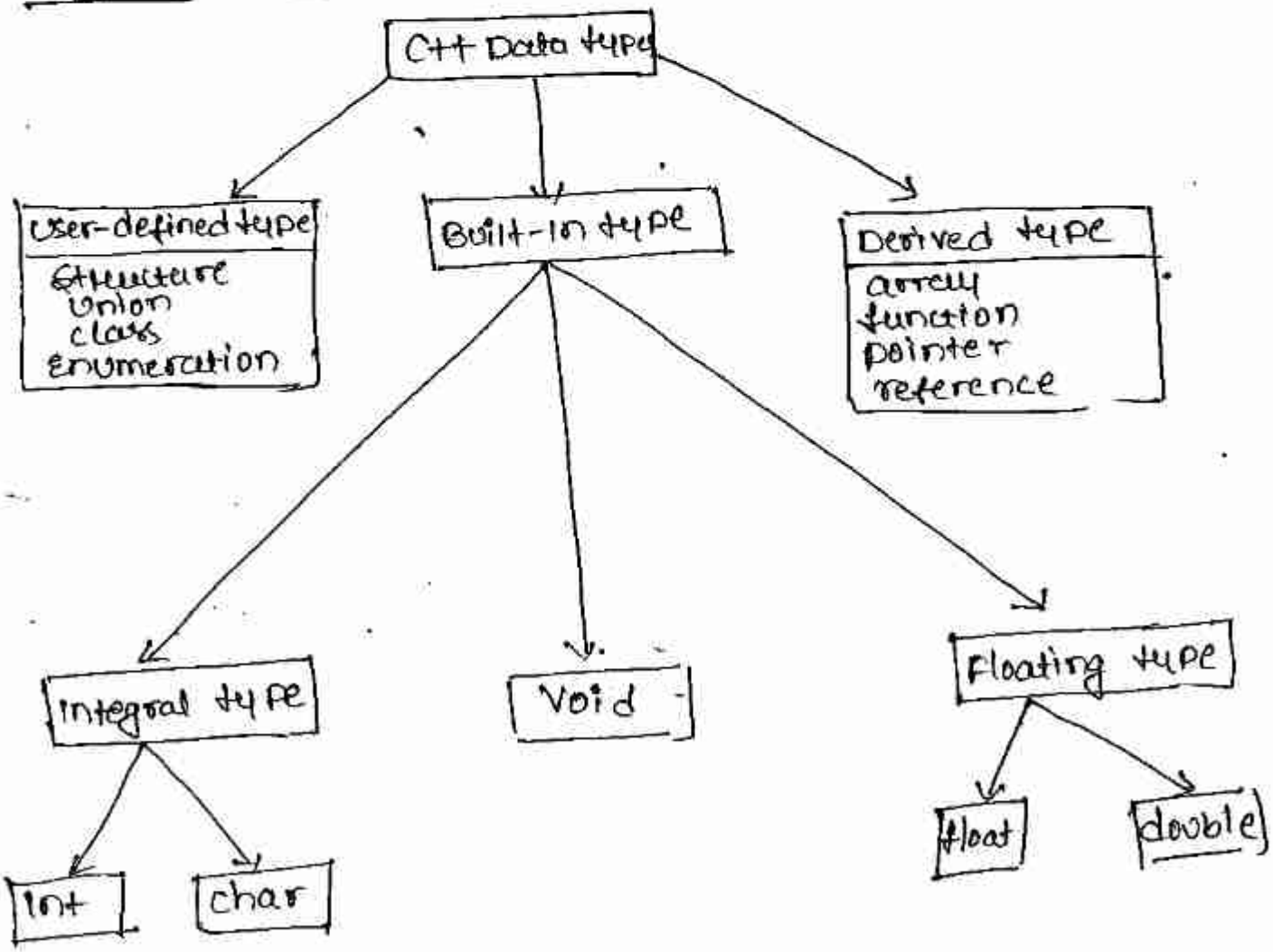


The client-server Model.

* A typical C++ program would contain 4 section as shown above

* The class declarations are placed in a header file & the definitions of member functions go into another file.
* This approach Enables the programmer to separate the abstract specification of the Interface from the implementation details (class definition from member functions definition)
* Finally, the main program that uses the class is placed in a 3rd file which includes the previous 2 files as well as any other files required.

Different data types



* Both C & C++ compilers support all the built-in (also known as basic (or) fundamental) data types.

Type	Bytes	Range
char	1	-128 to 127
unsigned char	1	0 to 255
signed char	1	-128 to 127
int	2	-32768 to 32767
unsigned int	2	0 to 65535
Signed int	2	-32768 to 32767
Short int	2	~~0 to 65585~~
unsigned short int	2	0 to 65535
Signed short int	2	-32768 to 32767
long int	4	-2147483648 to +ve
Signed long int	4	-2147483648 to +ve
unsigned long int	4	0 to 4294967295
float	4	3.4E-38 to 3.4E+38
double	8	1.7E-308 to 1.7E+308
long double	10	.3.4E-4932 to 1.1E+4932

Variables

* C++ allows the declaration of a variable anywhere in the scope.
* This means that a variable can be declared right at the place of its first use.

Examples

```
int main()
{
    float x;                    // declaration
    float Sum=0;
  for(int i=1; i<5; i++) // declaration
   {
     Cin >> x;
     Sum = Sum + x;
   }
     float average;   // declaration
     average = sum/(i-1);
     Cout << average;
     return 0;
   }
```

Dynamic Initialization of Variables

* In C++ permits initialization of the variables at run time. This is referred to as dynamic initialization
* A variable can be initialized at run time using expressions at the place of declaration.

Ex:-

```
- - - - -
- - - - -
int n = Strlen(String);
- - - - -
float area = 3.14159 * rad * rad ;
```

Here both the declaration & the initialization of a variable can be done simultaneously at the place where the variable is used for the 1st time.

If total = total+10;

This will change both total & Sum to 110

If Sum=0;

Will change both total & Sum is equal to '0'.

Example:

```
Int n[10];
Int &x = n[10]; // x is alternate for n[10]
Char &a = '\n'; // initialize reference to a
                   literal
```

Here x is an alternative (Alias) to the array n[10].
The variable a is initialized to the Newline constant.
A major application of reference variables is in passing arguments to functions

```
Let Void f(int & x) // uses reference
    { x = x+10; //x is incremented; So also m
    }
    int main()
    { int m = 10;
      f(m);    // function call
    - - - - -
    - - - - -
    }
```

If function f(m) is executed, the following initialization occurs

```
int &x = m
```

Thus, x becomes an alias (alternate) of m after executing the statement f(m)

let float average; // declare where it is necessary
average = sum/9;

Both can be combined into single statement as
float average = sum/i; // Initialize dynamically at run time.

Reference Variables

* C++ introduces a new kind of variable known as the reference variable.
* It provides an alternative name for a previously defined variable.
* for example, if we make the variable Sum a reference to the variable total.
* Then Sum & total can be used interchangeably to represent that variable.

A reference variable is created as follows.

datatype & reference-name = variable-name

Example

```
float total = 100;
float &Sum = total;
```

Here total is a float type variable that has already been declared.
Sum is the alternative name declared to represent the variable total.

```
∴ Cout << total;
  & Cout << Sum;
```

both print the value 100.

Operators in C++

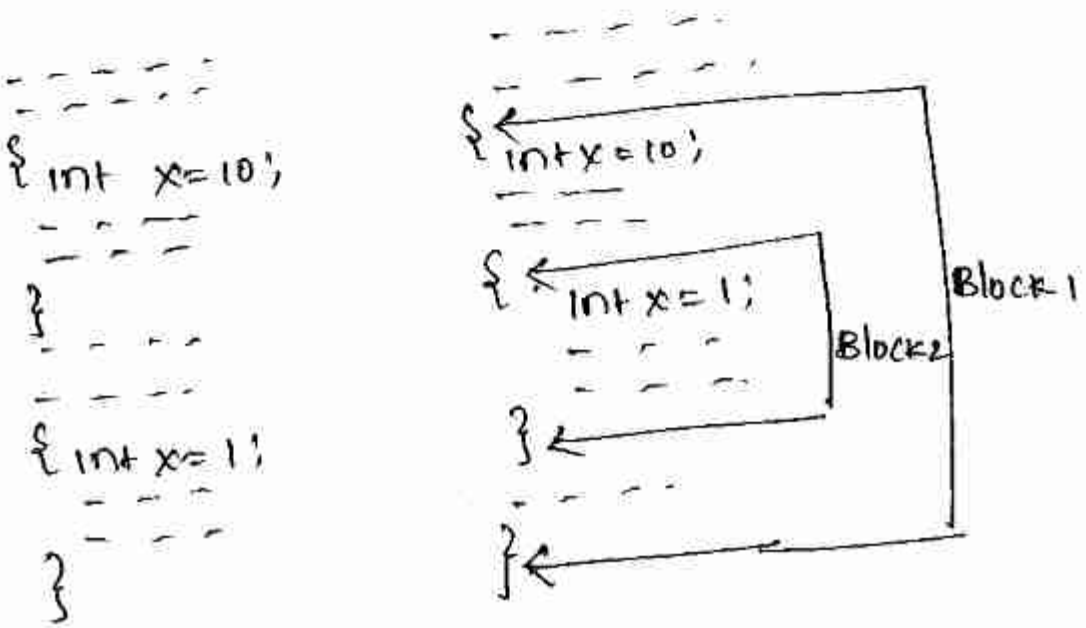
* C++ has a rich set of operators. All C operators are valid in C++ also.
* New operators in C++ are insertion operator <<, & the Extraction operators >>. Other operators are follows

::	Scope resolution operator
::*	pointer to member declaration
->*	pointer to member operator
.*	pointer to member operator
delete	memory release operator
endl	line feed operator
new	memory allocation operator
setw	field width operator

In addition, C++ also allows us to provide new definitions to some of the built-in operators. i.e we can give several meanings to an operator, depending upon the types of arguments used. This process is known as operator overloading

Scope Resolution operator

Exam:-



Here 2 declarations of x refer to two different memory locations containing different values.
i.e Statements in 2nd block cannot refer to variable x declared in the 1st block & vice versa

```
#include <iostream>
using name space std;
int m=10;
int main()          → // global m
{
    int m=20; // m redeclared, local to main
    {
        int k=m;
        int m=80; // m declared again
                  // local to inner block
        cout<<" We are in inner block \n";
        cout<<"k= "<<k<<"\n";
        cout<<"m= "<<m<<"\n";
        cout<<"::m= "<<::m<<"\n";
    }
    cout<<"\n We are in outer block\n";
    cout<<"m= "<<m<<"\n";
    cout<<"::m= "<<::m<<"\n";
    return 0;
}
```

o/p

We are in inner block
k=20
m=30
::m=10

We are in outer block
m=20
::m=10

Member Dereferencing operators

* C++ also permits us to access the class members through pointers.
* C++ provides a set of 3 pointer-to-member operators

Operator	function
.*	
::	To declare a pointer to a member of a class
*	To access a member using object name & a pointer to that member
->*	To access a member using a pointer to the object & a pointer to that member

Memory management operators

C++ uses 2 operator ① New ② delete that performs the task of allocating & freeing the memory in a better & Easier way.

* These operators manipulate memory on the free store they are also known as free store operators.
* An object can be created by using New & destroy -ed by using delete.

New operator takes the following general form

pointer-variable = New data-type

Ex `p= new int;` p is a pointer of type int

`q= new float;` q ← float

(or)
```
int *p = new int;
float*p= new float;
```

=>
```
*p = 25;
*q = 7.5;
```

i.e assign 25 to Newly created int object & 7.5 to the float object.

Also we can initialize the memory using New operator as follows.

pointer-Variable = new data-type (value);

Ex:. int *p = new int (25);
float *q = new float (7.5);

* New can be used to create memory space for any data type including user-defined types such as arrays, structures & classes.

i.e pointer-variable = new data-type[size];

Ex:. int *p = new int[10];

When a data object is no longer needed, It is destroyed to release the memory space for reuse.

delete pointer-variable.

Ex:. delete p;
delete q;

If we want to free a dynamically allocated array

delete [size] pointer-variable;

Ex:. delete [] p;

```
#include <iostream>
#include<conio.h>
using name space std;
Void main()
{ int *arr;
  int size;
```

```
Cout<<"Enter the Size of the integer array: ";
Cin>>Size;
Cout<<" Creating an array of size "<<size<<"..";
arr = new int [size];
Cout<<"\n Dynamic allocation of memory for array
        arr is Successful.";
 delete arr;
 getch();
}
```

o/p Enter the size of the integer array: 5
Creating an array of size 5
- - - Successful.

delete (or) free from dynamically allocated mem of array.

Type cast operator

C++ permits explicit type conversion of variables (or) expressions using the type cast operators.

i.e (type-name) expression // C notation
type-name (expression) // C++ notation

ex: average = sum / (float) i; // C notation
average = sum / float (i); // C++ notation

∴ typedef used as functional notation

i.e typedef int * int_pt;
p = int_pt (q);

Expressions

An Expression is a combination of operators, constants & Variables arranged as per the rules of the language.

Seven types

(1) Constant expressions
(2) Integral
(3) float
(4) pointer
(5) Relational
(6) Logical
(7) Bitwise

The Combinations of the above expressions are known as Compound Expression

(1) Constant Expressions

It Consist of only constant values

Ex:. 15
20+5/2.0
'x'

(2) Integral Expressions

Integral Expressions are those which produce integer results

Ex:. m
m*n-5
m*'x'
5+int(2.0)

m & n are integers. Variables

Float Expressions

After all conversions, Expression produce floating point results.

Ex:. x+y x & y are floating-point variables
x*y
6 + float(10)
10.75

pointer Expressions

This produce address values

ex &m
ptr m is variable
ptr+1 ptr is pointer
"x42"

Relational Expressions (or) Boolean Expressions

This yields result of type bool which takes a value true (or) false.

Ex:. x<=y
a+b==c+d
m+n>100

Logical Expressions

This combine two (or) more relational Expressions & produce bool type result.

Ex a>b && x==10
x==10 || y==5

Bitwise Expressions

It is used to manipulate data at bit level
It is used for testing or Shifting bits
Ex:- x<<3 // shift three bit position to left
y>>1 // shift one bit position to right.

Operator overloading

* overloading means assigning different meanings to an operation.
* INPUT O/p & ~~O~~ I/p operators are >> & << although << (built-in definition) operator ~~can~~ is for shifting of bits, It is also used for displaying the values of various data types.

Ex:- Cout << 75.86; // for displaying double type value

Cout << "well done"; // for displaying a char value.

* Almost all c++ operators can be overloaded with a few exceptions such as the member-access operators Conditional operator (?:), Scope resolution operator(::) & Size operator (sizeof).

~~Control~~ Structures

Control Structures

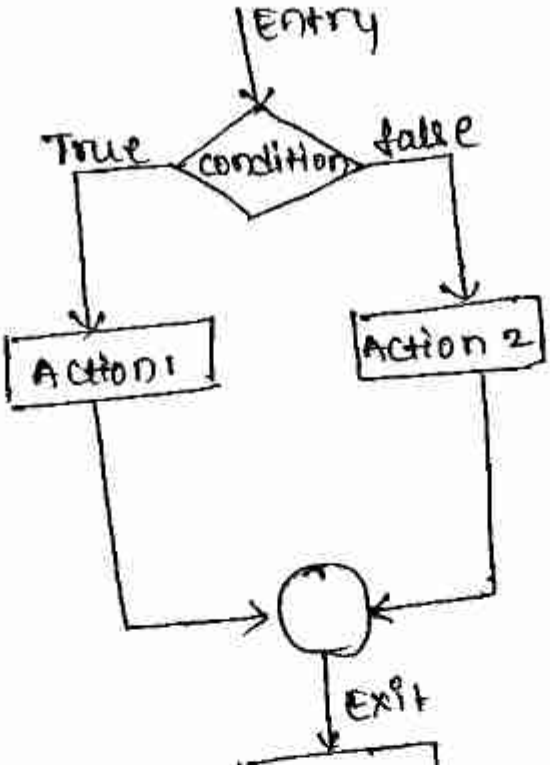
3 Control Structures in C++

(1) Sequence Structures (Straight line)
(2) Selection Structures (branching)
(3) Loop Structures (Iteration or Repetition)

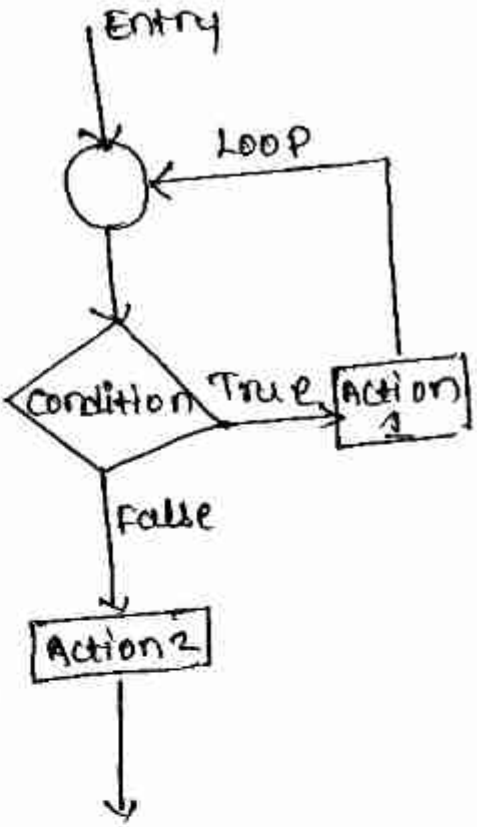
The approach of using one or more of these basic control constructs in programming is known as Structured programming.



(a) Sequence Structure



(2) Selection Structure

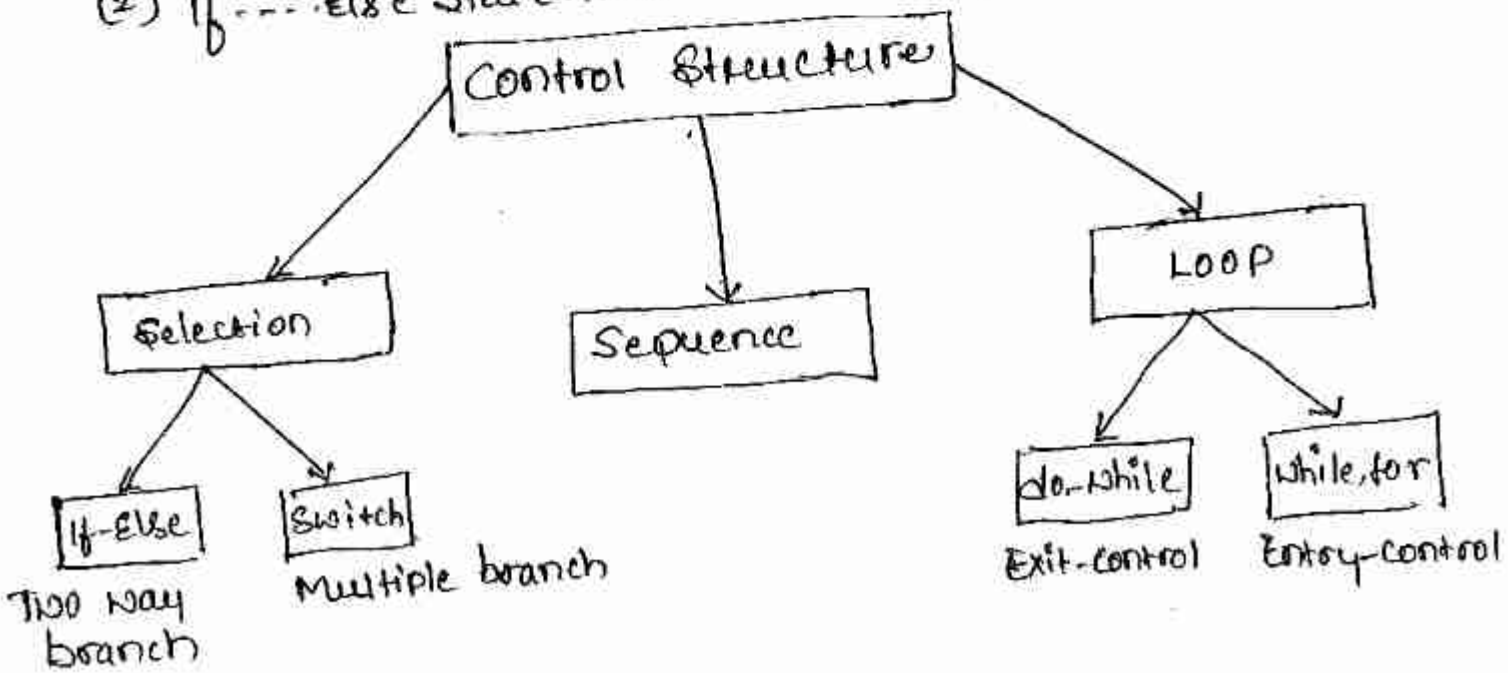


(3) Loop Structure.

The If Statement

The If Statement is implemented in 2 forms

(1) Simple If Statement
(2) If ... else Statement



* Simple If Statement

Ex:

```
If (Expression is true)
 {
   action1;
 }
action2;
action3;
```

* If....Else Statement

```
If (Expression is true)
 { action1;
 }
 Else
 { action2;
 }action3;
```

Switch Statement

This is a multiple-branching statement, based on condition.

The control is transferred to one of the many possible points.

```
Switch (Expression)
 { case1;
   {
   action1;
   }
   case2;
   {action2;
   }
   case3;
   {
   action3;
   }
   default;
   {action4;
   }
}action5;
```

Do-While Statement

do-while is an exit-controlled loop.
Based on a condition, the control is transferred back to a particular point in the program

```
do
 {
   action1;
 }
 while (condition is true);
 action2;
```

While Statement

This is also a loop structure.

```
while (condition is true)
  {
    action1;
  }
  action2;
```

For Statement

This is used when an action is to be repeated for a predetermined Number of times.

```
for (initial value; test; increment)
    { action1;
    }
   action2;
```

Module-2

FUNCTIONS

* Functions are building blocks of C++ programs
* It is used to perform different tasks depending on the arguments passed to it.

Main function

* In C++, the main(), returns a value of type int to the operating System.

i.e int main()

int main(int argc, char *argv[]);

It uses return statement for termination.

→
```
int main()
{
  - - -
  - - -
  return 0;
}
```

Here return value is also called as Exit value. is used to determine if there is any problem.

The Normal convention, if Exit value is zero means the program ran successfully. While a "non-zero" value means there was a problem.

* Function prototyping is one of the major features of C++
* It describes the function interface to the compiler by giving details such as the Number and type of arguments and the type of return values.
* The function prototype declaration in calling program is type function-name (argument-list);

The argument-list contains the types & names of arguments that must be passed to the function.

Examples

```
float Volume (int x, float y, float z);
```

In general, we can either include (or exclude the Varia -ble names in the argument list of prototypes.

Example

```
float Volume (int a, float b, float c)
{
    float v = a * b * c;
    - - -
    - - -
}
```

Inline functions

* When a function is likely to be called many times.
* Everytimes, it takes a lot of extra time in executing a series of instructions for tasks such as jumping to the function, saving registers pushing arguments into the stack, & returning to the calling function.
* C++ has a different solution to this problem.
* An inline function is a function that is expan -ded in line when it is invoked.
* i.e the compiler replaces the function call with the corresponding function code (Similar to macros expansions). Inline functions are defined as follows

```
inline function-header
{
    function body
}
```

Inline functions

When function is called, controller jumps to perticul -ar function. Saving registers, pushing arguments into the stack, & returning to the calling function. This will take more time.

* one solution to this problem is macros but it is not real function ∴ Error checking does not occur during compilation.
* C++ has a different solution to this problem. To Eliminate the cost of calls to the small functions, C++ proposes a new feature called inline functio
* An inline function is a function that is expan -ded in line when it is invoked. i.e the compilier replaces the function call with the corresponding function code.

```
inline function-header
{
    function body
}
```

* All inline functions must be defined before they are called.
* Inline keyword sends a request, not command to the compiler
* If inline function defrition is too long (or) too complicated as this case compiler may ignore it.

Example

```
#include<iostream>
using namespace std;
inline float mul(float x, float y)
  {
    return (x*y);
  }
  inline double div(double p, double q)
   {
     return (p/q);
   }
int main()
 {
   float a=12.345;
   float b=9.82;
   Cout<< mul(a,b)<<"\n";
   Cout<< div(a,b)<<"\n";
    return 0;
 }
```

In the above program Inline function is declared before Main program. With respect to inline function Mul is displayed as 121.228 i.e value of a & b is used by & x&y & done the Multiplication & it is displayed by using Cout and div = 1.25713.

* Function Overloading

The Same function name is used to create functions that performs a variety of different tasks. This is known as function polymorphism in OOP.

* using the Concept of function overloading. Consider the following example. A family of functions is designed with one function name but different argument list.

* The Correct function to be invoked is determined by checking the number and type of the arguments but not on the function type.

```
// Declarations
int add (int a, int b); // prototype 1
int add (int a, int b, int c); // prototype 2
int
double add (double x, double y); // prototype 3
double add (int p, double q); // prototype 4
double add (double p, int q); // prototype 5

// Function calls
Cout << add (5, 10); // uses prototype 1
Cout << add (15, 10.0); // uses prototype 4
cout << add (12.5, 7.5); //uses prototype 3
Cout << add (5, 10, 15); // uses ——— 2
Cout << add (0.75, 5); // uses ——— 5
```

→ A function call 1st matches the prototype having the same number and type of arguments & then calls the appropriate function for execution.

The function selection involves the following steps

1. The compiler 1st tries to find an exact match in which types of actual arguments are the same and use that function.
2. If an exact match is not found, the compiler uses the integral promotions to the actual arguments i.e Char to int
   float to double
3. When either of them fails, the compiler tries to use the built-in conversions (the implicit assignment conversions) to the actual arguments & then uses the functions whose match is unique.

for ex:- long Square (long n)
double Square (double x)

A function call is Square (10)

Here there is Multiple Conversion & Multiple matches
∴ Compiler generate an error message.

Here function call is integer argument either converted into long or double.

4. If all of the steps fail, then Compiler will try the user-defined conversions in combination with integral promotions & built-in conversions to find a unique match.

Example program

```
// Function Volume() is overloaded 3 times
#include <iostream>
using namespace std;
// Declaration (prototypes)
int Volume (int);
double Volume (double, int);
long Volume (long, int, int);
int main ()
{
  cout<<"calling the Volume() function for computin
  the volume of a cube — "<<Volume(10)<<"\n";
  cout<<"calling the Volume () function for computi
  the Volume of a cylinder — "<<Volume(2.5, 8)<<"\n
  cout<<"calling the Volume() function for computing
  the Volume of a rectangular box — "<<Volume(100L,7
                                              15);
  return 0;
}
// Function definitions
int Volume (int s) // cube
{
  return (s*s*s);
}
```

```
double volume (double r, int h) // Cylinder
{ return (3.14519 * r * r * h);
}
long volume (long l, int b, int h) // Rectangular box
{ return (l * b * h);
}
```

O/P

(1) Calling the Volume() function for computing the Volume of a cube — 1000

(2) Calling the Volume() function for computing the Volume of a Cylinder — 157.26

(3) Calling the Volume() function for computing the Volume of a Rectangular box — 112500.

~~Friend and~~

Specifying a class

* A class is a way to bind the data and its associated functions together.
* It allows the data & functions to be hidden, if necessary from external use.
* A class specification has 2 parts
  (1) Class declaration
  (2) Class function definitions.

(1) Class declaration:- describes the type and scope of its members

(2) Class function definitions:- Describes how the class functions are implemented.

General form of a Class declaration

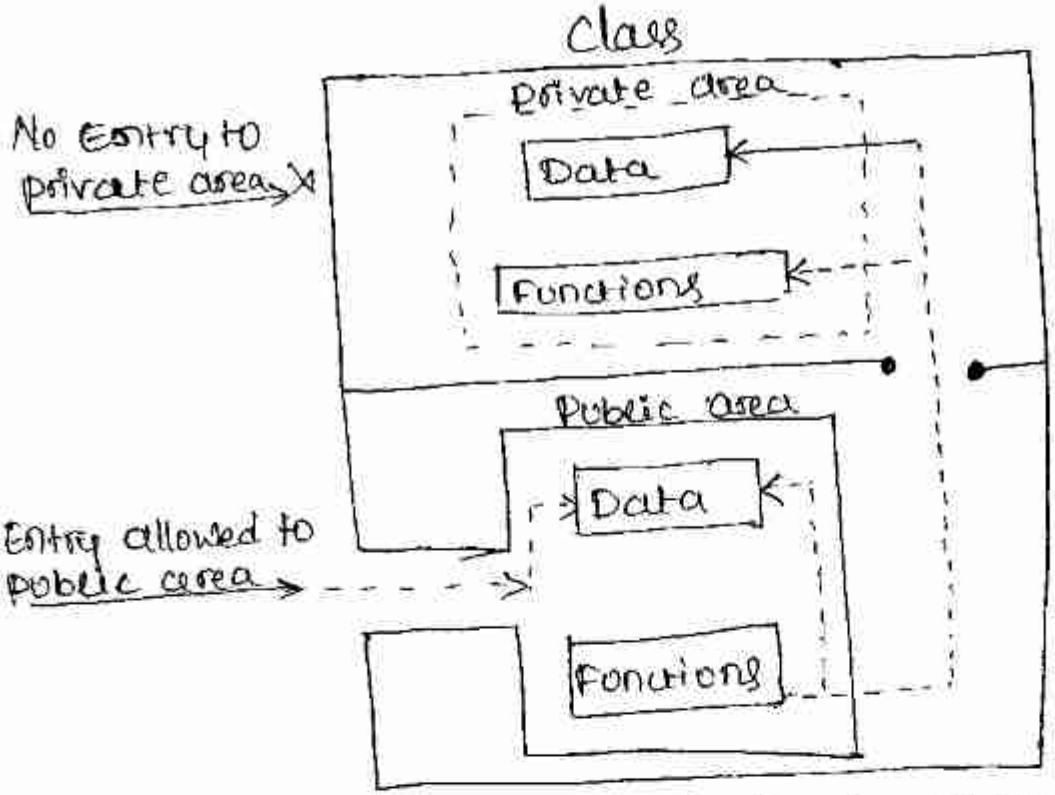
```
class class_name
{
  private:
    variable declaration;
    function declaration;
  public:
    variable declaration;
    function declaration;
};
```

The keyword class specifies data of type class_name.
Body of a class is enclosed within braces & terminated by a semicolon.
Body contains declaration of variables and functions
These are collectively called as class members.

* The class members are usually grouped under private & public
* In private:- private can be accessed only from within the class
* public:- members can be accessed from outside the class also.
* The data hiding:- using private declaration, is the key feature of OOP.

The variables declared inside the class are known as data members & the functions are known as member functions

* The member functions can have access to the private data members & private functions
* Public members can accessed from outside the class



Data hiding in classes

Example

```
class item
{
    int number;          // Variable declaration
    float cost;          // private by default
  public:
    void getdata(int a, float b); // Functions declaratic
    void putdata(void); // using prototype
                        // Ends with semicolon
};
```

* Here data members are private by default while both the functions are public.
* getdata() can be used to assign values to the member variables number & cost
* putdata() used for displaying variables.

C++ program with class

```
#include <iostream>
using namespace std;
class item          // class followed with class_name
{
  int number;       // variable names is declared as private
  float cost;       // private by default
  public:
  void getdata(int a, float b); // prototype declaration
                                to de defined.
  // function defined inside class
  void putdata(void)
   {
    cout << "number: " << number << "\n";
    cout << "cost: " << cost << "\n";
   }
};
//---.member function definition-----
  void item :: getdata(int a, float b) // use member
                                       ship label.
  { number = a;     // private variables
    cost = b;       // directly used
  }
//------.main program.------
  int main()
   { item x;
```

```
Cout << "\n object X" << "\n";
X.getdata (100, 299.95); // call member function
X.putdata(); // call member function
item y;
Cout << "\n object y" << "\n";
Y.getdata (200, 175.50);
Y.putdata();
return 0;
}
```

O/p is

```
object X
number : 100
cost   : 299.95

Object Y
number : 200
cost   : 175.5
```

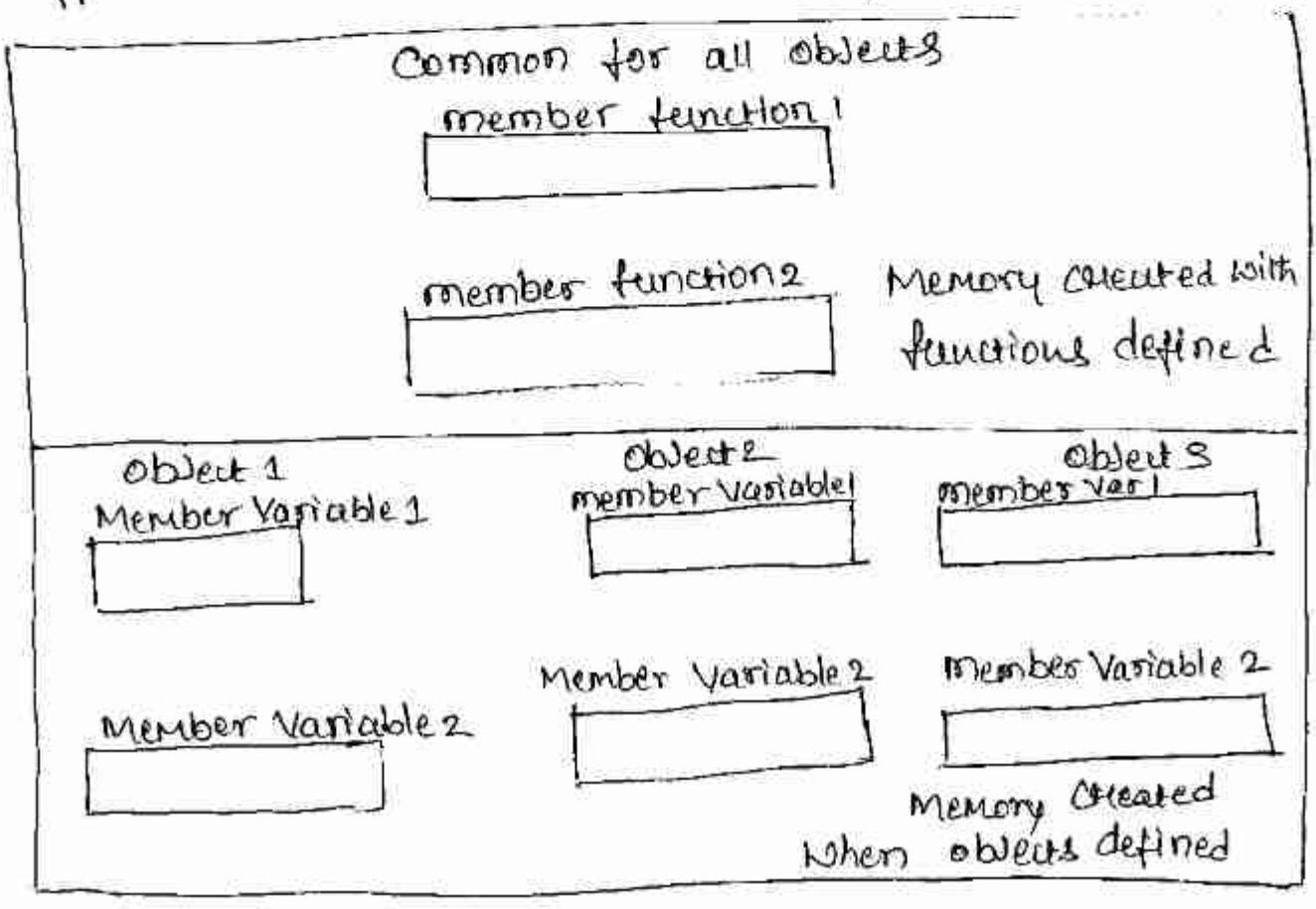
Arrays within a class

* The arrays can be used as member variables in a class.
* i.e `Const int Size = 10; // provides Value for array size`

```
class array
{
    int a[size]; // 'a' is int type array
    Public:
     Void setVal (void);
     Void display (void);
};
```

Memory Allocation for Objects

* N.KT. Memory Space for objects is allocated when they are declared & not when the class is specified.
* Member functions are created and placed in the Memory Space only once when they are defined as a part of a class specification
* Since all objects belonging to that class use the same member functions ∴ no separate memory space is required
* Only Space for member variables is allocated separately for each object.
* Separate memory locations for the objects are essential, because the member variables will hold different data values for different objects.



Arrays of objects

* Arrays of Variables that are of the type class. Such Variables are called arrays of objects.
* Consider the following defnition

```
Class Employee
{
    char name[30];
    float age;
  public:
    Void getdata(void);
    Void putdata(void);
};
```

The identifier Employee is a user-defined data type & can be used to create objects that related to different catagories of the Employees.

Example

```
Employee manager[3]; // array of manager
Employee foreman[15]; // array of foreman
Employee worker[75]; // array of worker.
```

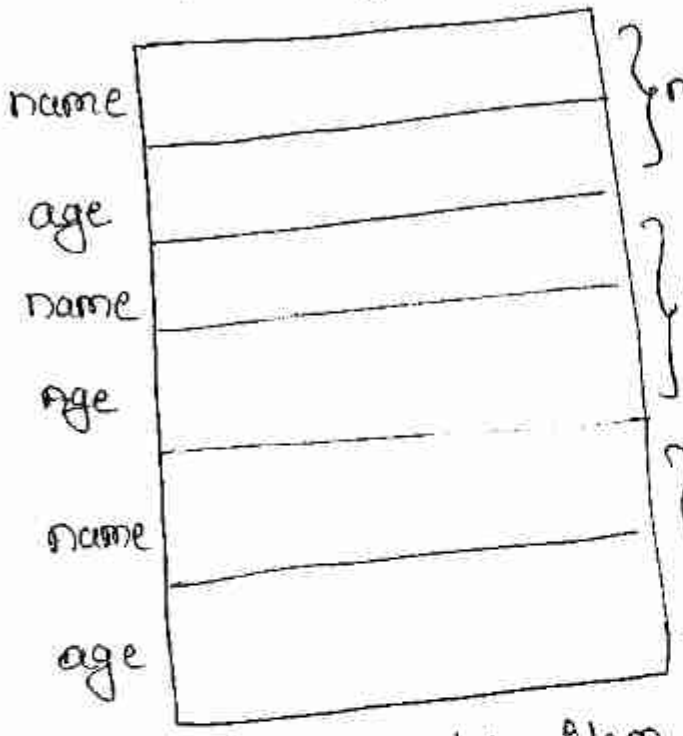
* i.e array of manager contains 3 objects (manager) namely, manager[0], manager[1] & manager[2], of type Employee class

My foreman array contains 15 objects (foreman)
worker array ——→ 75 objects (workers)

* 'dot' member operator is used to access the member functions.

Example: manager[i].putdata();

This will display data of the i^{th} element of the array manager



Storage of data item of an object array

* An array of objects is stored inside the memory
* Only space for data items of the objects is created
* member functions are stored separately & will be used by all the objects.

Program

```
#Include <iostream>
using namespace std;
class employee
{ char name[30]; //String as class member
  float age;
  public:
  void getdata(void);
  void putdata(void);
};
```

```
Void employee :: getdata(void)
 {
   Cout << "Enter name: ";
    Cin >> name;
    Cout <<" Enter age: ";
    Cin>> age;
  }
 Void employee :: putdata(void)
  { Cout << "Name: "<< name<<"\n";
     Cout<<" Age: "<< Age<<"\n";
  }
  Const int size=3;
   int main()
   { employee manager[size];
      for(int i=0; i<size; i++)
      { Cout<<"\n Details of manager"<< i+K<<"\n";
        Manager[i].getdata();
       }
       Cout <<"\n";
       for(i=0; i<size; i++)
       { Cout <<"\n manager" <<i+1 <<"\n";
          manager[i].putdata();
        }
        return 0;
   }.
```

I/P

Details of Manager 1
Enter name: XYZ
Enter age: 45

Details of Manager 2
Enter name: PQR
Enter age: 50

Details of Manager 3
Enter name: abc
Enter age: 55

O/P

Manager 1
Name: XYZ
Age: 45

Manager 2
Name: PQR
Age: 50

Manager 3
Name: abc
Age: 55

Member functions

Member functions can be defined in two places
(1) outside the class definition
(2) Inside the class definition

(1) outside the class definition

~~* Member functions are declared (or) defined separately outside the class.~~

* member functions that are declared inside a class have to be defined separately outside the class.
* An important difference b/w a member function & a normal function is that a member function incorporates a membership 'identity label' in the header.
* This label tells the compiler which class the function belongs to.

The general form of a member function definition is

```
Return-type  class_name :: Function-name (argument declaration)
    {
    Function body
    }
```

Consider the following Example

```
Void item :: getdata (int a, float b)
    {
     number = a;
      cost = b;
    }
    Void item :: putdata (void)
     {  Cout << "Number:" << number << "\n";
        Cout << "cost:" << cost << "\n";
     }
```

It has Some Special characteristics

a) Several different classes can use the same function name

(2) Member functions can access the private data of the class

(3) A member function can call another member function directly, without using the dot operator

Inside the class definition

* Another method of defining a member function is to replace the function declaration by the actual function def definition inside the class.

i'e

```
class item
{
    int number;
    float cost;
  public:
    void getdata(int a, float b); // declaration
                                  // inline function
    void putdata (void) // defining inside the class
      { cout << number << "\n";
        cout << cost << "\n";
      }
};
```

Pointers to Members

* It is possible to take the address of a member of a class and assign it to a pointer
* The address of a member can be obtained by applying the operator '&'.
* A class member pointer can be declared using the operator ::* with class name.

PTO

Example

```
Class A
{
  private:
    int m;
  public:
    void show();
};
```

∴ pointer to the member 'm' as follows

```
int A::* ip = &A::m;
```

Here ip pointer created thers acts like a class member in that it Must be invoked with a class object.

→ A::* means pointer to member of class A

→ &A::M means address of the m member of A class.

* The pointer ip can be used to access the member m inside member function (or friend functions)

* i.e Let a is an object of A declared in a member funcion. ∴ m is access using pointer ip as follows

i.e

```
In A ::*ip = &A::m;
:
Cout<< a.*ip // display
:
:
Cout << a.m;
```

ip holds the address of m & the value is displayed in a

Directly the value of m is displayed in a by using dot operator.

Friendly functions

* Consider 2 classes manager & Scientist.
* Let the function income tax() is operated on the Objects of both these classes
* In such situation C++ allows the common function to made friendly with both the classes
* Thereby allowing function to access to the private data of these classes.
* Such a function need not be a member of any of these classes.

friend function is declared below

```
class ABC
 {
   - - - - -
   - - - - -
   public:
     - - - - -
     - - - - -
     friend void xyz(void);//declaration
 };
```

A friend function possesses certain special characteristics

(1) It is not in the scope of the class to which it has been declared as friend

3) Since It cannot be called using the object of that class

4) It can be invoked like a normal function without the help of any object

5) unlike member function, it cannot access the member names directly. So it must use dot membership operator

6) it can be declared either in the public or private part of class

7) It has objects as arguments.

program

```
#include<iostream>
using namespace std;
class Sample
{ int a;
  int b;
 public:
  void setvalue()
   { a=25;
     b=40;
   }
  friend float mean(sample s);
};
    float mean(Sample s)
  { return float(s.a+s.b)/2.0;
  }
```

```
int main()
{ Sample x; // object x
  x.setvalue();
  cout<<"mean value = "<<mean(x)<<"\n";
  return 0;
}
```

o/p Mean Value = 32.5

Member functions of one class can be friend functions of another class. In such cases, they are defined using the Scope resolution operator

```
class x
{
  ....
  ....
  int fun1(); // member function of X
  .....
};
class y
{
  ....
  ....
  friend int x::fun1(); // fun1() of X
                        // is friend of Y
  ....
};
```

* All the member functions of one class as the friend functions of another class. The class is called a friend class

Virtual functions

polymorphism refers to the property by which objects belonging to different classes are able to respond to the same message, but in different forms. This can be achieved using virtual functions

* If we use the same function name in both the base and derived classes.
* The function in base class is declared as Virtual using keyword Virtual.
* ∴ C++ determines which function to use at run time based on the type of object pointed by the base pointer, rather than the type of the pointer.
* Thus by making the base pointer to point different objects.

Program

```
#include <iostream>
using namespace std;
class base
{
```

```
Void display()
 {cout <<" \n Display base";}
  }
  Virtual Void Show()
   {cout <<" \n Show base";
    }
  };
  Class Derived : public base
   {  public:
       Void display
        {cout <<" \n Display derived";
         }
       Void Show()
        {cout <<"\n Show derived ";
        }
    };  }
   int main()
    {  Base B;
       Derived D;
       Base *bptr;
       Cout <<"\n bptr points to base \n";
       bptr = &B;
       bptr->display(); // calls base version
       bptr-> Show(); // calls base version
```

```
cout << "\n\n bptr points to Derived \n";
    bptr = &D;
    bptr -> display(); //calls Base version
    bptr -> show(); // calls derived version
    return 0;
}
```

Module-3

Constructors & Destructors & Operator overloading

C++ provides a Special member function called the constructor which enables an object to initialize, itself when it is created. This is known as automatic initialization of objects.

It also provides another member function called the destructor that destroys the objects when they are no longer required.

Constructors

* A constructor is a Special member function whose task is to initialize the objects of its class.
* It is Special because its name is the same as the class name.
* The constructor is invoked whenever an object of its associated class is created.
* It is called constructor because it constructs the values of data members of the class.

A constructor is declared & defined as follows

```
class Integer
{ int m,n;
  public:
  Integer (void); // constructor declared
  ....
  ....
};
Integer :: Integer(void) // constructor defined
  { m=0; n=0;
  }
```

* constructor is used initialized an object automatically.

i.e integer int1;

It not only create object int1 of type integer but also initialize data members of integer class.

* A constructor that accepts no parameters is called the default constructor

Let class A

The default constructor of class A is A::A()

Special characteristics

(1) constructor should be declared in the public section

(2) They are invoked automatically when the object are created.

(3) They donot have return types.

(4) They cannot be inherited.

(5) They can have default arguments

(6) It cannot be virtual.

(7) An object with a constructor (or destructor) cannot be used as a member of a union

(8) They may make implicit calls to the New & delete when Memory allocation is required.

* The constructor integer (), is used to initializes the data members of all the objects to zero.

* However to initialize the various data elements of different objects with different values when they are created

* This is achieved by passing arguments to the constructor function when the objects are created. This is called parameterized constructors

The constructor integer() may be modified to take arguments as shown below

```
class integer
{
  int m, n;
  public:
   integer (int x, int y); //parameterized
    . . . . .                constructor
    . . . . .
};
integer :: integer (int x, int y)
 { m=x; n=y;
 }
```

When a constructor has been parameterized the object declaration statement i-e integer int1; may not work.

object is declared by 2 ways

- By calling the constructor explicity
- By calling the constructor implicity

```
integer int1 = integer (0, 100); // Explicit call
integer int1 (0, 100); // implicit call
```

Program

Class With Constructor

```
#Include <iostream>
 using namespace std;
 class integer
 { int m,n;
   public:
   integer(int, int);   // constructor declared
    void display (void)
    { cout <<"m= "<<m<<"\n";
      cout <<"n= "<<n<<"\n";
    }
 };
 integer :: integer (int x, int y) //constructor defined
 {
    m=x; n=y;
 }
 int  main()
 { integer  int1 (0,100); //constructor called implicitly
   integer  int2 = integer(25.75) // ← Explicitly
    cout <<"\n object1"<< "\n";
    int1. display();
    cout <<"\n object2"<< "\n";
    int2. display();
    return 0;
  }.
```

O/P

Object 1	object 2
m=0	m=25
n=100	n=75

Multiple Constructors in a class
(or)
constructor overloading

When more than one constructor function is defined in a class, it called constructor overloaded

Consider 2 kinds of Constructor

```
integer();              // No arguments
integer(int, int);      // Two arguments
```

In 1st case constructor itself supplies the data values & No values are passed by calling ~~prog~~ program

In 2nd case function call passes the appropriate values from main().

Let
```
class integer
{
  int m, n;
  public:
  integer() {m=0; n=0;}  // constructor 1
  integer(int a, int b)  // constructor 2
  {m=a; n=b;}
  integer(integer &i)    // constructor 3
  {m=i.m; n=i.n;}        // (copy constructor)
```

This declares 3 constructors for an integer object

1st Constructor receives no arguments

2nd ——— two ———

3rd ——— one integer object as an argument.

for Ex in main program if I1, I2 & I3 are 3 objects are constructed by using constructor it as follows

Ex1 integer I1 → This would automatically call constructor 1 & set m=0 & n=0

Ex2 integer I2(20, 40) → This would automatically call constructor 2 & initialize data members m & n of I2 i.e 20 & 40

Ex3 integer I3(I2); → This would automatically call constructor 3 & ie it copies the I2 values into I3. This is also called as copy constructor.

Program overloaded constructors

```
#include <iostream>
using namespace std;
class complex                    // class is declared
{ float x, y;
  public:
  complex() { }                  // Constructor no arguments
  complex(float a){x=y=a;}       // Constructor-one
  complex (float real, float imag)
   { x=real; y= imag;}           // Constructor -2 arguments

  friend complex Sum(complex , complex);
   friend void show (complex);
};
complex Sum (complex c1, complex c2)
{
    complex c3;
    c3.x = c1.x + c2.x;
    c3.y = c1.y + c2.y;
     return (c3);
}
void  show(complex c)
{ cout <<c.x<<"+j"<<c.y<<"\n";
 }
```

```
int main()
{
  Complex A (2.7, 3.5); // define & initialize
  Complex B (1.6);      // define & initialize
  Complex C;
  C = SUM (A,B);
  Cout << "A = "; Show (A);
  Cout << "B = "; Show (B);
  Cout << "C = "; Show (C);
```

O/P A = 2.7 + J 3.5
B = 1.6 + J 1.6
C = 4.3 + J 5.1

Copy constructor

* A Copy constructor is used to declare and initialize an object from another object

Let integer $I_2 (I_1)$; ①

This means object I_2 is initialized (member-by-member) by the object I_1 (or) copy the values of I_1 to I_2.

above statement we can also write as

'integer $I_2 = I_1$;

(or) $I_2 = I_1$; → provided I_1 & I_2 are objects

This statement is valid.

Program

```
#include <iostream>
using namespace std;
class code
{
  int id;
  public:
  code() { }                    // Constructor
  code(int a) {id=a;} // Constructor with one argu
  code(code &x)                 // Copy constructor
  {
    id = x.id;
  }
  void display(void)
  { cout << id;
  }
};
int main()
{ code A(100); // Object A is created & initialized
  code B(A);  // Copy constructor called
  code C=A;   // Copy
  code D;     // D is created not initialized
  D=A;        // copy constructor called
  cout << "In id of A: "; A.display();
  cout << "In id of B: "; B.display();
  cout << "In id of C: "; BC.display();
} return; cout << "In id of D: "; D.display();
```

o/p

id of A: 100
Id of B: 100
id of C: 100
id of D: 100

Dynamic constructors

* Allocation of Memory to objects at the time of their Construction is known as dynamic Construction of objects.
* Constructors are used to allocate Memory while creating objects.
* This would helps in saving of memory. When objects are not of the same size.
* The Memory is allocated with the help of New operator.

Program shows the use of New in constructor that are used to construct strings in objects

Program

```
#Include <iostream>
#Include <String>
  Using namespace std;
  Class Create        // Class declaration With class name
                         Create
   {
     char *name;
     int length;
     public:
       & Create()                    // Constructor-1 declared
        {length =0;                     With no Arguments
         name = new char[length+1];
        }
        Create (char *s)      // Constructor-2 declared
        { length = Strlen(s);          With one Argument
          name = new char[length+1];
          Strcpy (name, s);
         }
         Void display (void)
         { cout << name << "\n";
          }
          Void Join(String &a, string&b);
   };
  Void Create :: Join(Create &a , Create&b)
   { length = a.length + b. Length;
     delete name;
     name = new char[length+1]; //dynamic allocation
```

```
    strcpy (name , a.name) ;
    strcat (name, b.name);
};

int main( )
{
    char *first = "Joseph" ;
    String name1(first), name2("Louis"); name3("Lagrange"),
    S1, S2;
    S1. Join (name 1, name2) ;
    S2. Join (S1, name3);
    name1 . display ( );
    name2. display ( ) ;
    name3. display ( ) ;
    S1. display( ) ;
    S2. display ( ) ;
    return 0 ;
}
```

o/p

```
Joseph
Louis
Lagrange
Joseph Louis
Joseph louis lagrange.
```

Destructors

* It is used to destroy the objects that have been created by a constructor.
* Destructor is a member function whose name is the same as the class name but is preceeded by a tilde(~).
* It never takes any argument nor does it return any value.
* It will be invoked implicitly by the compiler upon exit from the program to clean up storage that is no longer accessible.
* delete is the operator to deallocate the Memory Space

```
Matrix ::~Matrix( )
 {
   for(int i=0; i<d1; i++)
    delete p[i]
    deletep;
  }
```

Program

Implementation of Destructors

```
#include<iostream>
using namespace std;
 int count=0;
 class test
 {public:
     test( )
```

```
{
    count++;
    Cout << "\n\n Constructor msg: Object number " << count <<
        "Created..";
}
~test()
{ cout << "\n\n Destructor msg: Object number " << count <<
        destroyed.." ;
    count--;
}
};
int main()
{ cout << "Inside the Main block..";
  cout << "\n Creating first Object T1..";
   test T1;
   { cout << "\n\n Inside block1..";
     cout << "\n\n Creating 2 more Objects T2 & T3..";
     test T2, T3;
     cout << "\n\n leaving block 1..";
   }
   cout << "\n\n Back Inside the Main block..";
   return 0;
}
```

O/P

```
Inside the Main block..
Creating first Object T1..
constructor msg: Object number 1 created..
Inside Block 1:
Creating two more Objects T2 & T3....
constructor msg: Object number 2 Created:..
```

```
Constructor msg: object number 3 created...
leaving block1:...
Destructor msg: object number 3 destroyed.
Destructor msg: object number 2 destroyed.
Back inside the main block..
Destructor msg: object number 1 destroyed.
```

Defining operator overloading

* To define an additional task to an operator, we must specify what it means in relation to the class to which the operator is applied.
* This is done with the help of special function called operator function.

```
return type class name :: operator op (arglist)
{
    function body // task defined
}
```

Here return type is the type of value returned by the specified operation

'op' is the operator being overloaded.

Operator is a keyword.

* Operator functions must be either member function (or) friend function.
* friend function will have only one argument for unary operators & 2 for binary operator
* Member function has no argument for unary & one for binary operators.

Operators functions are declared in the class using prototypes as follows

```
Vector operator + (vector);          // vector addition
Vector operator -();                 // unary minus
friend vector operator+(vector,vector); // vector addition
friend Vector operator-(vector);     // unary minus
Vector operator -(vector &a);        // subtraction
int operator == (vector);            // comparision
friend int operator ==(vector,vector)// comparision
```

Overloading unary operators

* Consider the unary minus operator
* It takes just one operand.
* This operator changes the sign of an operand when applied to a basic data item.

program

```
#include <iostream>
using namespace std;
class space
{
  int x;
  int y;
  int z;
 public:
  void get data (int a, int b, int c);
  void display (void);
  void operator -(); // overload unary minus
};
```

```
Void Space :: getdata (int a, int b, int c)
  { x=a;
    y=b;
    z=c;
  }
Void Space :: display(void)
  {
    Cout<<"x= "<<x<<" ";
    Cout<<"y = "<<y<<" ";
    Cout <<"z = "<<z<<"\n";
  }
Void Space:: operator-()
  {
    x = -x;
    y = -y;
    z = -z;
  }
int main
  {
    Space s;
    s.getdata (10, -20, 30);
    Cout<<"S : ";
    S. display ();
    -S;          // activates  operator -() function
    cout <<"-S : ";
    S. display();
    return 0;
  }
```

o/p S: x = 10 y = -20 z = 30

-S: x = -10 y = 20 z = -30

Overloading binary operators

* Binary operator ~~is~~ ie operated on 2 operand.

consider the following functional notation

C = Sum (A, B)

is replaced as C = A + B // Arithmetic notation

i.e functional notation can be replaced by a natural looking expression

Program

```
#include<iostream>
using namespace std;
class complex
{ float x;          // real part
  float y;          // imaginary part
 public:
  complex() { }     // constructor 1
  complex (float real, float imag) // constructor 2
  { x=real; y= imag; }
   Complex operator + (complex);
     void display (void);
};
Complex Complex:: operator+ (complex c)
{
    Complex temp;                    //temporary
    temp.x = x + c.x;
    temp.y = y + c.y;
    return (temp);
}
```

```
Void complex :: display (void)
  {
  cout << x << " +j " << y << "\n";
  }
  int main()
  {
  complex c1, c2, c3;  // invokes constructor 1
    c1 = complex (2.5, 3.5); // invokes constructor 2
    c2 = complex (1.6, 2.7);
    c3 = c1 + c2;
    cout << "c1 = "; c1.display();
    cout << "c2 = "; c2.display();
    cout << "c3 = "; c3.display();

    return 0;
  }
```

O/P

c1 = 2.5 +J 3.5
c2 = 1.6 +J 2.7
c3 = 4.1 +J 6.2

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

MODULE-2

Inheritance

* Defining Derived Classes

A derived class can be defined by specifying its relationship with base class in addition to its own details

general form

```
class derived-class-name : visibility-mode base-class-name
{
    ....//
    ....// members of derived class
    ....//
};
```

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

* Here colon indicates that the derived-class-name is derived from the base-class-name.
* visibility mode specifies whether the features of the base are privately derived (or) publicly derived.
* The default visibility-mode is private.

Examples

```
class derived : private base // private derivation
{ members of derived
};
class derived : public base // public derivation
{ members of derived
};
Class derived : base // private derivation by
{ members of derived by default.
};
```

* When a base class is privately inherited by a derived class, 'public members' of the base class becomes 'private members' of the derived class.
* When the base class is 'publicly inherited', 'public members of the base class becomes public members of the derived class & ∴ they are accessible to the objects of the derived class
* In both the cases, the private members are not inherited.

Single inheritance : public

Let B is base class — Contain 1 private data members
1 public
3 public member functio

D is derived class — contains 1 private data member
2 public member function

```
#include <iostream>
using namespace std;
class B
{
    int a;    // private; not inheritable
public:
    int b;    // public; ready for inheritance
    void set_ab();
    int get_a(void);
    void show_a(void);
};
```

```
class D : public B // public derivation
 {
     int c;
   public:
     Void Mul (Void);
     Void display (void);
 };
Void B :: Set_ab (Void)
 { a=5; b=10;
 }
int B :: get_a()
 {
   Return a;
 }
Void B :: Show_a()
 {
   a=5; b=10;
 }
int B :: get_a()
 {
   return a;
 }
Void B :: Show_a()
 { cout << "a = " <<a<< "in";
 }
Void D :: Mul()
 { c=b* get_a();
 }
Void D :: Display()
 { Cout << "a = " << get_a() << "in";
```

```
    cout<<"b= "<< b<< "\n";
    cout<<"c= "<<c<< "\n";
}
int main()
{
    D d;
    d.set_ab();
    d.MUL();
    d.show_a();
    d.display();
    d.d = 20;
    d.MUL();
    d.display();
    return 0;
}
```

O/P

d

```
a=5
a=5
b=10
c=50
a=5
b=20
c=100
```

Single Inheritance : private

```
#include <iostream>
using namespace std;
class B
{ int a; // private: not inheritable
  public: // public: ready for inheritance.
    int b;
    void get_ab();
     int get_a(void);
     void show_a(void);
};
 Class D : private B    // private derivation
  { int c;   // private
    public:
      void mul(void);
      void display(void);
  };
 void B :: get_ab (void)
  { cout << "Enter values of a & b: ";
    cin >> a >> b;
  }
  int B :: get_a()
  { return a;
  }
  void B :: show_a()
   { cout << "a = " << a << "\n";
   }
  void D :: mul()
   { get_a();
     c = b * get_a();
   }
```

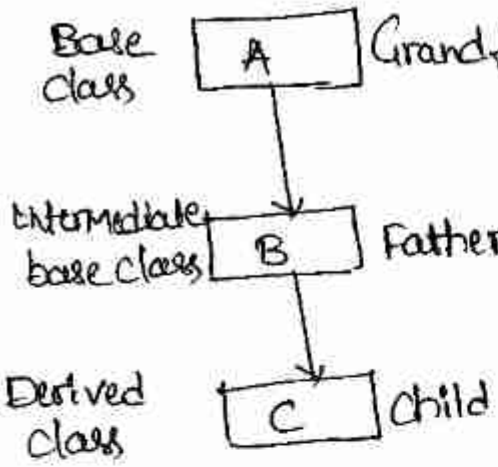
```
Void D :: Display()
{ Show_a();
  Cout << "b = " << b << "\n";
     << "c = " << c << "\n\n";
}
Int Main()
{ D d;
  d.get_ab() ; Won't work :. it is private for
                              Derived class
  d.MUL();
  d.Show_a() ; ——————
  d.display(); //public in Derived class
  d.MUL();
  d.display();
   Return 0;
}
```

O/P

```
Enter values of a & b : 5 10
a=5
b=10
c=50
Enter values of a & b : 12 20
a=12
b=20
c=240
```

Multilevel inheritance

Here a class is derived from another derived class.



Here class A serves as a base class for the derived class B, which in turn serves as a base class for the derived class C.

Program

```
#include <iostream>
using namespace std;
class student
{ protected:
    int roll_number;
  public:
    void get_number(int);
    void put_number(void);
};
void student::get_number(int a)
{ roll_number = a;
}
void student::put_number()
{ cout << "Roll Number: " << roll_number << "\n";
}
class test : public student   // test is derived class from base class student
{ protected:
    float sub1;
```

```
    float sub2;
    public:
        void get_marks (float, float);
        void put_marks (void);
};
void test :: get_marks (float x, float y)
{   sub1 = x;
    sub2 = y;
}
void test :: put_marks()
{ cout << " marks in SUB1 = " << sub1 << "\n";
  cout << "marks in SUB2 = " << sub2 << "\n\n";
}
class result :: display (void)
{ total = sub1 + sub2;
  put_number();
  put_marks();
  cout << "Total = " << total << "\n";
}
int main()
{
    result student1;
    student1.get_number (111);
    student.get_marks (75.0, 59.5);
    student.display();
    return 0;
}
```

O/P

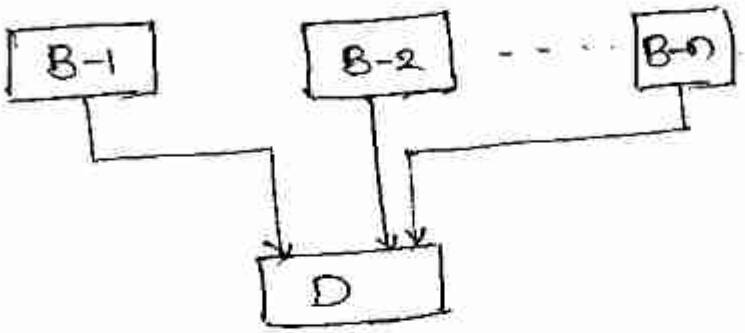
```
Roll Number : 111
Marks in SUB1 = 75
Marks in SUB2 = 59.5
Total = 134.5
```

Multiple Inheritance

* A class can inherit the attributes (a quality or) features of two (or) more classes this is known as Multiple inheritance.

(or)

* Multiple inheritance is one it allows to combine the features of several Existing classes for defining New classes.



Syntax

```
class D: visibility B-1, visibility B-2 ------
  {
    ----
    ----  (Body of D)
    ----
  };
```

* Here visibility may be either Private (or) public. The base classes are separated by commas.

```
#include <iostream>
using namespace std;
 class M
  {
    protected:
      int m;
    public:
      void get_m(int);
  };
```

```
Class N
{ protected:
    int n;
  public:
    Void get_n(int);
};
Class P : public M, public N
{
  public:
    Void display (void);
};
Void M:: get_M (int x)
  { m=x;
  }
Void N :: get_n(int y)
  { n=y;
  }
Void P:: display (void)
  { Cout << "m = " <<m << "\n";
    Cout<< "n = " <<n << "\n";
    Cout << "m*n = "<<m*n<<"\n";
  }
Int Main()
 { P p;
   P.get_m(10);
   P.get_n(20);
   P.display ();
 } return 0;
```

O/P

```
m=10
n= 20
m*n= 200
```

Pointers to Objects

* A pointer is used to point to an object created by a class.
* Consider an example

```
item x
```

item is class & x is object

We can define pointer ptr of type item as follows

```
item *ptr;
```

* Object pointers are useful in creating objects at run time. We can also use an object pointer to access the public members of an object
* pointes are also used to point the address of variable. i.e

```
Let item x; // class item x is variable
    item *ptr= &x; // ptr is pointed to address
                   of variable x.
```

* WKT Member functions of item is refered by 2 ways ① using dot operator
  ② arrow operator

i.e

```
x.getdata(100, 75.5);
x.show();
```

in by

```
ptr -> getdata(100, 75.5);
ptr -> show();
```

i.e *ptr is an alias of 'x'

* pointers are also used to create object by using New operator

i.e

```
Item *ptr = New Item;
```

* Pointers are also used to create arrays of object.

i.e item *ptr = new item[10];

Program

program of code & price of different item using pointers to object concept.

```
#include<iostream>
using namespace std;
class item          // class item is created
{ int code;         // Initialization of code & its price
  float price;      //  of item
public:
  void getdata(int a, float b) // declaration of function
  { code=a;                    // defining function so that
    price=b;                   // the value of code & price
  }                            // is assigned to a & b
  void show()                  // declaration of function
  {                            //  for display
    cout<<"code:"<<code<<"\n";
    cout<<"price:"<<price<<"\n";
  }
};
  const int size=2; // size is variable is assigned
                    // to 2 & it is constant.
int main()
{ item *p=new item[size]; // array of object. is
                          // created using new
                          // operator & pointer
```

```
item *d = P;   // New pointer d points to Existing
               pointer P
int x, i;
float y;
for (i=0; i<size; i++)   // To read code & price
                            of item1 & item2
{
  Cout << " Input code & price for item " << i+1; // ith item
  Cin >> x >> y; // Entered value is read to
                    x & y respectively
  P->getdata(x,y); // calling getdata function
                      using pointer with arrow
                      operator -> to class 'item
                      So that code = x & price = y
  P++; // points to New Next memory location
          for item2, to entering code & price
}
for (i=0; i<size; i++)   // For display the item1 &
                            item2 on screen.
{
  Cout << " item : " << i+1 << "\n";
  d->show();   // calling for display item1 &
                  item2 to class 'item
  d++;   // points to Next Mem loc for
            displaying item2
}
return 0;
}
```

O/P

Input code and price for item1 40 500
Input code and price for item2 50 600

Item:1
code: 40
Price: 500

Item: 2
Code: 50
price: 600

this pointer

* 'this' is pointer is to represent an object that invokes a member function.
* 'this' is a pointer that points to the object for which this function was called.
* For example, the function call A.Max() will set the pointer this to address of the object A.
* Consider the following Example

Max = A.greater(B)

Here the function return to B, if the age of B is greater than that of A.
Otherwise A is greater than B so that function returns to object A. Using this pointer.

Program

To find 'greatest Age of 3person by using this pointer. using this pointer we have to check the condition by using function so that this pointer return to the particular object.

```
#include<iostream>
# include<cstring>
 using namespace std;
 class person    // class name is person
 { char name[20]; // array name with size 20
   float age;    // age is variable type float
 Public:
    pre person(char *s, float a) // Function is
     { strcpy(name, s);           declared it has
       age = a;                   pointer s points
     }                            the character entered,
                                  passed by main
                                  program, that is
                                  copied to name array
                                  & a is the variable
                                  it takes Appropriate
                                  age of name pointed
                                  by s
person & person :: greater (person & x) // condition
 { if (x.age >= age)
     return x;
   else
     return *this;
 }
```

// If age of P3 is greater than P1 it return object P3 otherwise P1 by this pointer

```
void display (void)
  { cout << "Name: " << name << "\n";
    cout << "Age: " << age << "\n";
  }
};
int main()
{
  person P1 ("John", 37, 50),   // 's' pointer pointed to
         P2 ("Ahmed", 29.0),    P1 of Name John & its
         P3 ("Hebber", 40.25);  age copied to arrow
                                name wise for P2 & P3

  person P = P1.greater(P3); // i.e P3 is greater than
                               P1. It calls the
  cout << "Elder person is: \n";  1st function of class
  P1.display(); // call 2nd      person, so that
                function of     x = P3 if P3 > P1
                class person    it returns P3 other
                                -wise P1 is return to
                                x by this pointer
                                wise for P1 & P2

  P = P1.greater(P2); // checks P2 is greater than
                        P1
  cout << "Elderperson is: \n";
  P.display();
  return 0;
}
```

O/P

```
Elder person is :
Name: Hebber
Age : 40.25
Elder person is :
Name: John
Age: 37.5
```

Pointers to Derived classes

* NKT pointers are used to point Base objects.
* We can also used to point objects of derived classes.
* The single pointers variable can be Made to point to objects belongs to the different classes.

for Example B is the base class
D is Derived class

Then if pointer is declared in B can also be a pointer to D also.

```
Let B *cptr; // cptr is pointer is declared in
                Base class B
    B b;     // b is the object of class B
    D d;     // D is derived class & d is object of
                derived class
    Cptr = &b; // Cptr is pointer pointed to object
                  b
    ∴ cptr = &d; // cptr is also used to point d
                    also
```

Note:- Using pointer we can access only Members which are inherited from base class in derived class but not the members belonging to derived class. In order to access the Member of derive class we have to declare New pointer in derived class

Program

How base pointer points to the derived class.

```
#include<iostream>
using namespace std;
class BC          // Base class name BC
 { public:
     int b;       // b is a public variable
     void show()
     {cout<<"b= :"<<b<<"\n";
      }
};
class DC : public BC  // class 'DC' is derived class
                      from base class BC
 {
    Public:
      int d;          // d is the own member of
                         derived class
      void show()  // function derived from base
                      class BC
       { cout<<"b= :"<<b<<"\n";
         cout<<"d= : "<<d<<"\n";
};
```

```
Int Main()
{
  BC *bptr;   // initializing bptr pointer under
  BC base;// Variable base class BC & it is also used
              in derived class also.
  bptr = &base; // bptr points to address of Variable
                   base in base class BC
  bptr->b=100 ; //Access Base Class BC via pointer
  Cout<<"bptr points to base object \n";
   bptr -> Show(); // pointer access base class BC
                      Member function
   Dc derived ; // Derived is variable of Derived
                   class DC
   bptr = &derived ;//bptr pointer of base class
                      used to address derived
                      Variable in derived class DC
   bptr->b=200; // Access of derived class via
                   pointer

   cout <<"bptr now points to derived object \n";
    bptr->Show(); // pointer access function
                     from derived class

    DC *dptr; // to access d from DC we have
                 de Create New pointer 'dptr'
    dptr = &derived;
    dptr=
     dptr->d =800;//New pointer dptr is used
                   to access d from Dc
```

```
    cout <<"dptr is derived type pointer \n";
     dptr->show();
    cout <<"using (CDC*)bptr) \n";
     ((CDC*)bptr)->d=400;
     ((DC*)bptr)->show();
     return 0;
}
```

o/p

bptr points base object
b=100
bptr now points to derived object
b=200
dptr is derived type pointer
b=200
d=800
using (CDC*)bptr)
b=200
d=400

Virtual function

The property by which objects belonging to different classes are able to respond the same message ~~is called~~ but in different forms is called polymorphism.
- This is achieved by virtual function.
Consider when we use same function name in both base & derived class The function

In base class is declared as Virtual.

* When a function is made Virtual C++ determines which function to use at run time base on type of Object pointed by base pointer, rather than type of the pointer

Program

```
#include <iostream>
using namespace std;
class Base   // class name base is created.
 {
    public:
       Void display()   // declare & defining of Function
        { cout<< "\n Display base" ;}
       Virtual Void Show()   // Void Show() function is used both in base & Derived class Hence it is preceeded by Virtual
        { cout<< "\n Show base"; }
 };
class Derived : public base   // Derived class is declared with the help of base class
 { public:
       Void display()   // Here Void display() & Void Show are same function used name in both base & derived class but Void Show is Virtual function
        { cout<<" \n Display derived"; }
       Void Show()
        { cout <<"\n Show Derived ";
        }
};
```

```
Int Main()
{ Base B;    // Base class of B object is
                created
  Derived D; // Derived class of D object is
                created
  Base *bptr; // bptr is a pointer is declared
                in base class
  Cout << "In bptr points to base \n";
  bptr = &B; // bptr points address of object B
                of base class
  bptr -> display(); // display function is accessed
                from base class via bptr
  bptr -> show(); // show() function is accessed
                from base class of object B"
                via bptr
  Cout << "\n \n bptr points to Derived \n";
  bptr = &D;  // base pointer bptr Now points
                to derived class of object D
  bptr -> display(); // even bptr address of Derived
                class but compiler ignores its
                content of bptr, so that function
                is executed in base class
  bptr -> show(); // since show is virtual function
                :. bptr call show function
                in derived class

  return 0;
}
```

O/P

bptr points to Base

Display base
Show base

bptr points to Derived
Display base (calls from base class)
Show Derived.

Pure virtual function

* A pure virtual function is a function declared in a base class that has no definition relative to the base class. is called pure virtual function

Ex. Let Void display() is declared in base class as Virtual Void display() = 0 ; // no definition in base class such function is ~~cla~~ called as do-nothing functions.

In these cases, the compiler requires each derived class to either define the function (or) redeclare it as a pure virtual function.

* Virtual functions cannot be used to declare any object of base. Such class is called as Abstract base classes.

Program of pure virtual function

```
#include<iostream>
using namespace std;
class Balaguruswamy          // base class
{
  public:
  virtual void example()=0; // pure virtual function
                            having no definition
                            (or)
                            do-nothing function
};

class c:public Balaguruswamy // declaration of
                               derived class C,
 {
  public:
        void example()
         {cout<<"C text book written by Balaguruswamy";
         }
};
 class oops:public balaguruswamy // declaration
                                    of derived class
                                    OOPS
  {
   public:
     void example()
     {cout<<" oops text book written by balaguruswamy
     }.
};
```

```
Void main()
{ Exforsys * array[2]; // pointer array is not declared
                       in base class, so that we
                       access the derived class via
                       array for virtual function
  C e1;                Void example (redefined
  OPPS e2;              in be derived class)
  array[0] = &e1;
  array[1] = &e2;

  array[0] -> example();
  arra [1] -> example();
}
```

o/p

C text book written by balaguruswamy
oops text book written by balaguruswamy

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

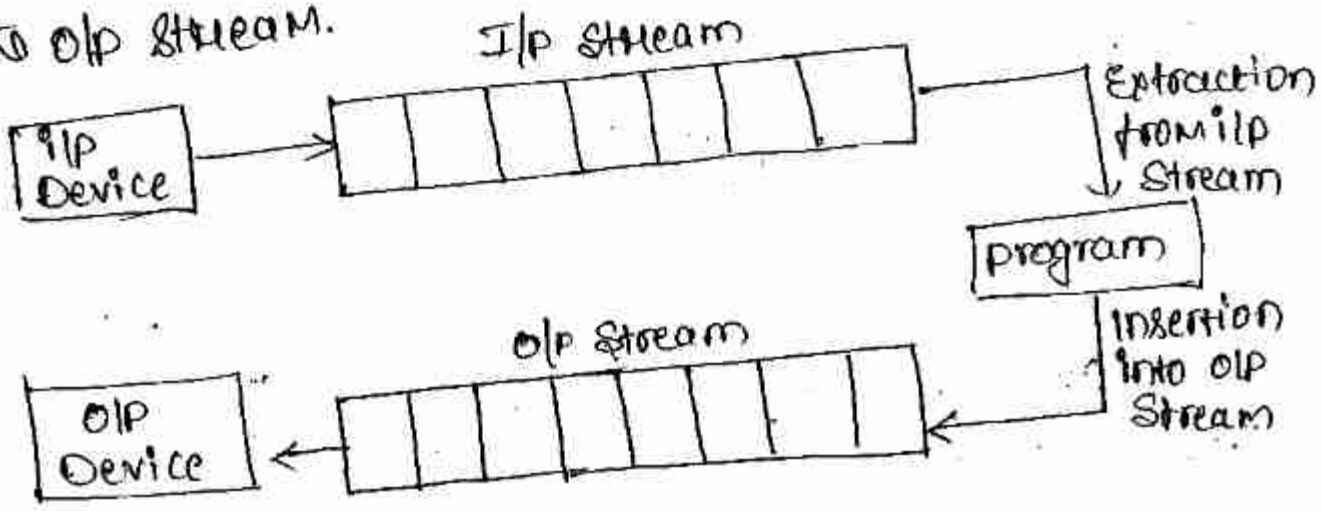
MODULE-2

Streams and Working With files

* C++ uses the Concept of Stream and Stream classes to implement its I/o operations with the console and disk files.

C++ Streams

* The I/o System in C++ is designed to work with a wide Variety of devices including terminals, disks and tape drives.
* I/o System is an interface to the programmer, i.e inde-pendent of the actual device being accessed. This inter-face is known as Stream.
* A Stream is a sequence of bytes. It acts either as a source from i/p data,
* It acts destination to which the o/p data can be sent.
* The source stream that provides data to the program is called the I/p stream and the destination stream that receive o/p from the program is called the o/p stream.



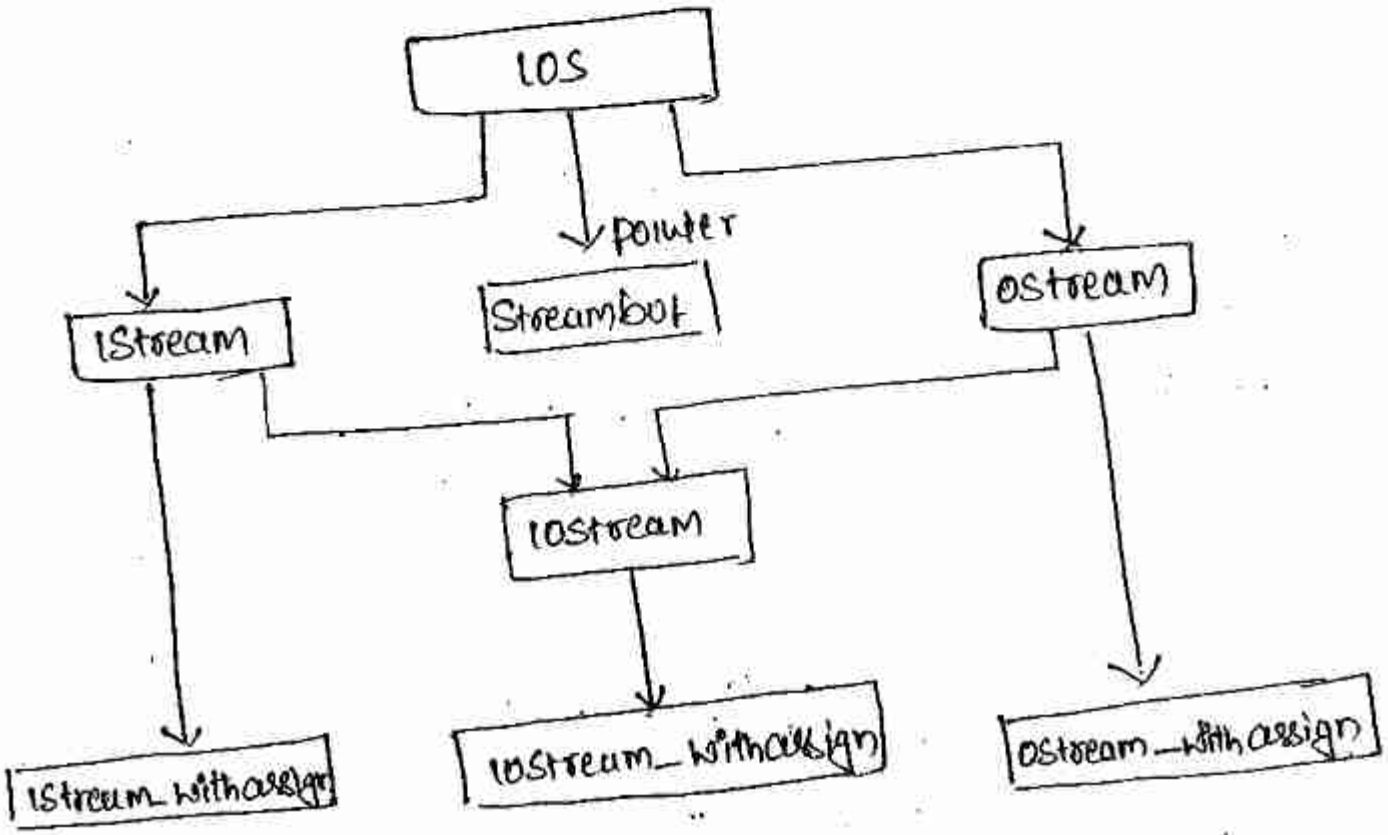
* Here data in the i/p stream can come from the keyboard (or) any other storage device. for example Cin → is the i/p stream connected to the standard i/p device i.e usually keyboard.

* The data in the o/p stream can go to the screen(or) any other storage device. for example Cout → is the o/p stream connected to standard o/p device i.e screen.

* These streams are redirect to other devices (or) files, if necessary.

C++ stream classes

The C++ I/O System contains a hierarchy of classes that are used to define various streams to deal with both the console and disk files. These classes are ~~et~~ called stream classes.



Class name	Contents
IOS (General i/p Stream class)	* Contains basic facilities that are used by all other i/p and o/p classes * Also contains a pointer to a buffer object * Declares constants and functions that are necessary for handing formatted i/p & o/p operations
IStream (i/p Stream)	* Inherits the properties of IOS * Declares i/p function such as get(), getline(), & read() * Contains overloaded extraction operator >>
OStream (o/p Stream)	* Inherits the properties of IOS * Declares o/p function put() & write() * Contains overloaded insertion Operator <<
IOStream (i/p & o/p Stream)	* Inherits the properties of IOS IStream & OStream through Multiple inheritance
Streambuf	* provides an interface to physical device through buffers * Acts as a base for filebuf class used IOS files

Unformatted I/O operations

UNFormatted data → Data which is received by the program without any modifications and sent to the O/p device without any modifications is known as Unformatted data.

Formatted data → Data which is received by the program with modifications & sent to O/p device with modifications is known as formatted data

Ex: Decimal value is displayed on screen in Hexadecimal No i.e 15 displayed as 'F' on screen.

* Here we are using Cin object & Cout object for Istream & ostream for read & display
* Cin statement uses >> (extraction operator) to read & Cout statement uses << (insertion operator) to display

Ex:
```
int a,
float b,
Char ch,
Cin>>a>>b>>ch;
Cout<<a<<b<<ch;
```

Here Cin & Cout are used in cascading fashion.

* TO read a single character, we can use get() of Istream
* TO Print a single character, we can use put() of ostream

* In get() we use both function get(char*) get(void)

These are used to read character including the blank space, tab & the Newline character

get(char) → used for assign i/p char to its argument

get(void) → returns the i/p char

put() → is a Member of ostream used to o/p a line of text, character by character.

Ex:- Cout.put('x'); displays the character x

Cout.put(ch); display the value of Variable ch.

Character I/O With get() & put()

```
#include <iostream>
using namespace std;
int main()
{ int cout = 0;//assign cout Variable as '0'.
  char c ;//C is variable of type char
  cout << " INPUT TEXT \n"; // message is displayed for input characters.
  Cin.get(c); // The char enters from keyboard is i/p to C, variable
  while (c!='\n') // If Entered char is Not Equal to \n until it reads the value & stores to 'C' Variable
  {
    cout.put(c); // → display line of text ie char by char
    count++; // counts the # of char
    Cin.get(c);
```

```
}
cout <<"\n Number of characters = "<<count<<"\n";
return 0;
}
```

O/P

INPUT TEXT

object Oriented programming

O/P

object oriented programming

Number of characters = 27

Ans: To Read a String (or) a line of text, we can use getline() function & to print a string we can use write() function.

General formate

getline (string, size) → Reads a whole line of text & Ends with a New line chars

write (string, size) → print a whole line of text

Ex:

```
#include <iostream>
using namespace std;
int main()
{ char str[30];
  cin.getline (str, 10);
  cout.write (str, 10);
  return 0;
}
```

O/P

I/P object oriented program

O/P

object oriented progra

Formatted Console I/O Operations

C++ Supports a number of features that could be used for formatting the output.

(1) ios class functions and flags.

(2) Manipulators.

(3) User-defined o/p functions.

The ios class contains a large No of member functions that would help us to format the o/p in a No of ways

ios format functions

function	Task
(1) width()	To Specify the required field Size for displaying an o/p value
(2) precision()	To Specify the No of digits to be displayed after the decimal point of a float value
(3) fill()	To Specify a character that is used to fill the unused portion of field
(4) setf()	To Specify format flags that can control the form of o/p display (Such as left-Justification & right-Justification)
(5) unsetf()	To Clear the flags Specified

*Manipulators are special functions that can be included in I/O statement to alter the format parameter of a stream.

* To access these Manipulators, the file iomanip should be included in the program

Manipulators	Equivalent ios function
setw()	width()
setprecision()	precision()
setfill()	fill()
setiosflags()	setf()
resetioflags()	unsetf()

width()

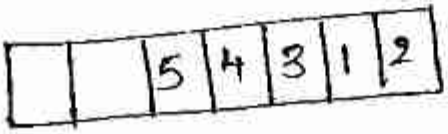
* It is used to define the width of a field necessary for o/p of an item.
* We have to use an object to invoke it.

i.e cout.width(w);

*Here w is field width (No of columns)
* The o/p will be printed at the right end of the field
* width() is used for only one item

Ex: cout.width(5);
cout << 543 << 12 << "\n";

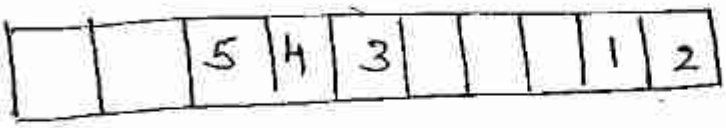
o/p



For example

```
Cout.width(5);
Cout<<5<<3;
Cout.width(5);
Cout<<12<<"\n";
```

Here field width should be specified for each item separately



* Note

If the specified field width is smaller than the size of the value to be printed, C++ automatically expands the field to fit the value.

Program

```
#include<iostream>
using namespace std;
int main()
{ int items[4]={10,8,12,15}; // item is array of 4 with costs
  int cost[4]={75,100,60,99}; are declared

  cout.width(5);
  cout<<"ITEMS"; // The 4 items code are filled with field width of 5
  cout.width(8);
  cout<<"COST"; // my cost is filled with field width of 8 (from Right to Left
```

```
Cout.width(15);
Cout<<"TOTAL VALUE"<<"\n";
```

|| total cost is filled with field width of 15

```
int sum=0;
for(int i=0; i<4; i++)
 {
   Cout.width(5);
   Cout<<items[i];
   Cout.width(8);
   Cout<<cost[i];
   int value = items[i]*cost[i];
   Cout.width(15);
   Cout<<value<<"\n";
   Sum = Sum+value;
 }
Cout<<"\n Grand Total=";
Cout.width(2);
Cout<<sum<<"\n";
return 0;
}
```

O/P

ITEMS	COST	TOTAL VALUE
10	75	750
8	100	800
12	60	720
15	99	1485

Grand TOTAL = 3755

ITEMS	1	5	1	2	8	1	0
COST	9	9	6	5	1	0	0	7	5
TOTAL VALUE	1	4	8	5	7	2	0	8	0	0	7	5	0
Grand Total=	3	7	5	5									

Precision ( )

* It is used to Specify the Number of digits to be displayed after the decimal point while printing the floating-point numbers.

```
Cout.precision(d);
```

Where d is the No of digits to the right of decimal point.

* precision() is declared. So that it not only print the No of ~~decimal~~ digit after the decimal point but also it rounded of the nearest value
* Trailing zeros are truncated
* precision setting stays in effect until it is reset
* Default precision is 6 digits.

Program

```
#include<iostream>
#include<cmath>
using namespace std;
int main()
{
  cout<<"\n precision set to 5 digits \n\n";
  cout.precision(5); // precision parameter set to 5
  cout<<" sqrt(10)="<<sqrt(10)<<"\n";
  cout.precision(6); // Default precision is set for
                     6 digits after decimal point
  cout.<<" sqrt(10)= "<<sqrt(10)<<default setting \n"
```

O/P

precision set to 5 digits

Sqrt(10) = 3.1623

Sqrt(10) = 3.162278 (default setting)

fill() functions

* If the values printing much larger field width than required by the values.
* The unsu. unused fields are filled by white spaces by default.

Example: Cout.fill(ch);

ch represents the character which is used for filling the unused positions.

Ex:- Cout.fill('*');
Cout.width(10);
Cout << 5250 << "\n";

→ O/P

*	*	*	*	*	*	5	2	5	0

* We can use other character also is used to fill unused space.

Program of padding with fill() & width()

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
Cout << Using Fill() & width() "\n";
Cout.fill('*')
cout.width(10);
Cout.precision(5);
```

Managing O/p with manipulators

* This is one is used to manipulates O/p formats
* More 2 (or) more manipulators can be used as a chain in one statement.
* We must include the file 'iomanip' in the program It provides a set of manipulators.

Manipulator	meaning	Equivalent
setw(int w)	Set the field to w	width()
setprecision(int d)	Set the floating point precision to d	
setfill(int c)	Set the fill character to c.	fill()
setiosflags(long f)	Set the format flag f	setf()
resetiosflags(long f)	clear the flag specified	unsetf()
endl	insert new line & flush stream	"\n"

Ex: `cout << setw(10) << 12345;`

This works similar to width()

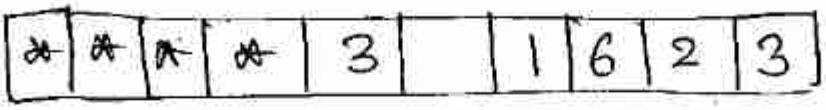
i.e `cout.width(10);`
`cout << 12345;`

O/p



```
    cout << "sqrt(10) = " << sqrt(10) << "\n\n";
    return 0;
}
```

O/P



Setf() functions

*It is a member function of the ios class. Can provide to print the text. left-justified

cout.setf(arg1, arg2) → bitfield specifies the group to which the formatting flag belongs.

↓ formatting flags defined in the class ios (Specifies action)

Examples

Format required	flag (arg1)	Bit-field (arg2)
Left-Justified o/p	ios::left	ios::adjustfield
Right-Justified o/p	ios::Right	ios::adjustfield
Fixed point notation	ios::fixed	ios::floatfield
Octal base	ios::oct	ios::basefield

Eg:.
```
cout.fill('*')
cout.setf(ios::left, ios::adjustfield);
cout.width(10);
cout << "TABLE 1" << "\n";
```

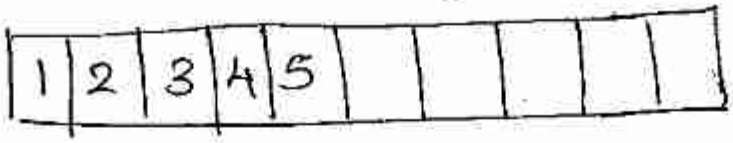
O/P



* For Left Justification

```
cout << setw(10) << setiosflags(ios::left) << 12345;
```

o/p

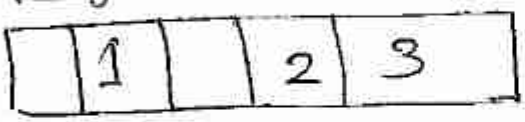


One Statement can be used to format o/p for 2 (or) more values.

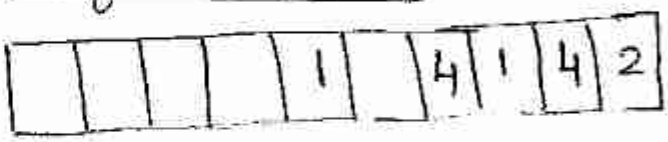
Ex: (1) `cout << setw(5) << setprecision(2) << 1.2345;`

(2) `cout << setw(10) << setprecision(4) << sqrt(2);`

o/p of (1) is



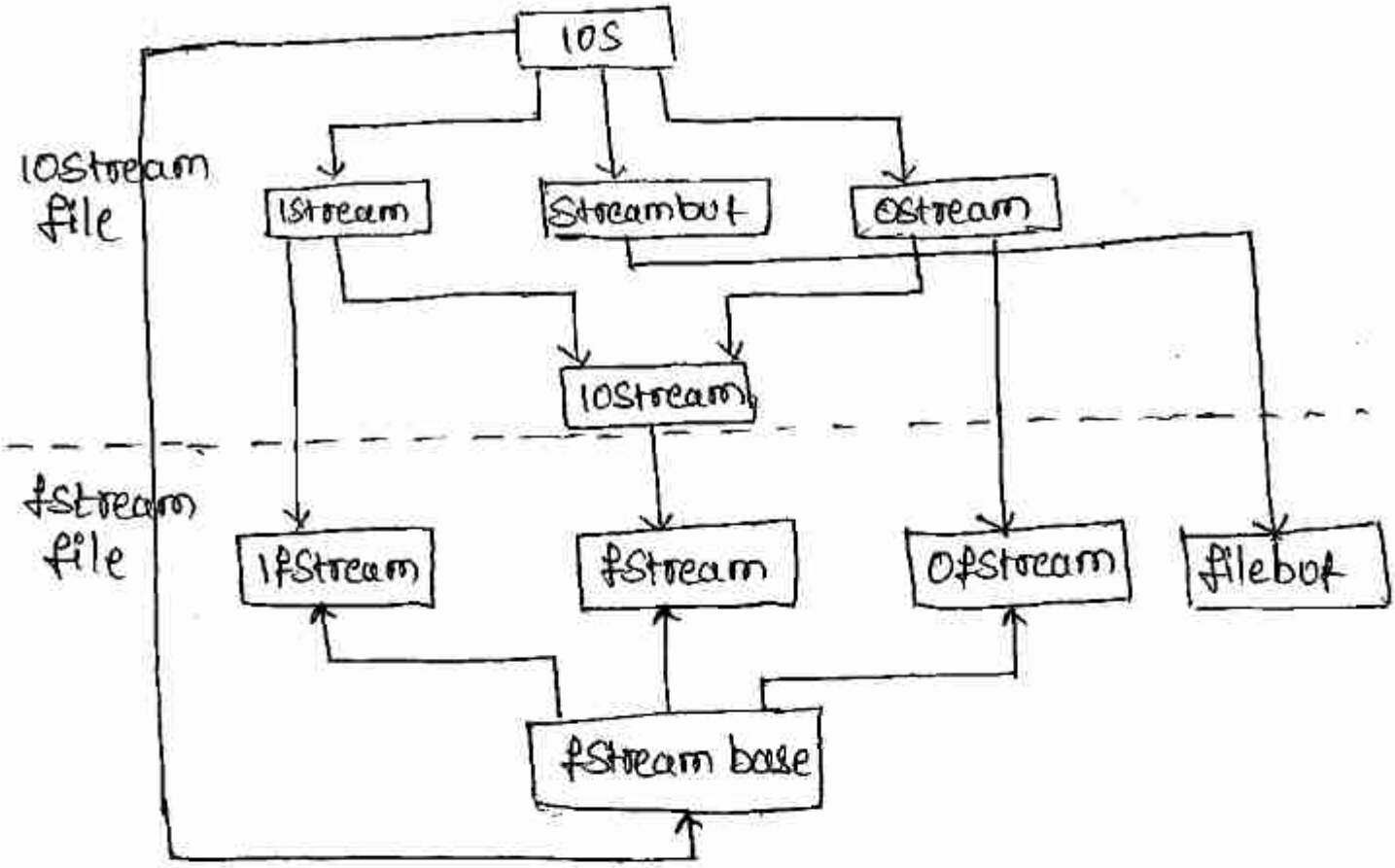
o/p of 2nd example



Classes for file Stream operations

file → A file is a collection of related data stored in a particular area on the disk. So that programs can be designed to perform the read and write operations on these files.

* The Ilo System of C++ contains a set of classes that defines the file handling methods. These include ifstream, ofstream & fstream. These classes are derived from fstreambase & from corresponding iostream



Class	Contents
filebuf	Its purpose is to set the file buffer to read & write. Contains openprot Constant used in open() of file Stream Classes. Also contain close() & open() as members
fstreambase	It serves as a base for fstream ifstream & ofstream class. Contains open() & close()
ifstream	provides i/p operations. Contains open(). Inherits the function get() getline() etc from istream
ofstream	provides o/p operations contains open(). Inherits put(), tellp() & write() from ostream
fstream	provides support for i/p & o/p operations It contain open() · Inherits all function from istream & ostream through

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

Module-3

1) Functions and Parameters

Value Parameters:

Consider example below

```
int fun(int a, int b, int c)
{
    return a+b*c;
}
```

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

~~In this program function fun, the expression~~

```
int main()
{
    int x, y, z;
    z= fun(2, a, 4);
return 0;
}
```

→ In the above program, Then function fun compiles expression a+b*c, where 'a, b' & 'c' are integers and result is also integer.

→ where a, b & c are formal parameters and 2, x & y are actual parameters that correspond to a, b and c respectively.

→ a, b & C are are actually Value parameters.

→ At run time the value of the actual parameter that corresponds to a value formal parameter is copied into the formal parameter before the function is executed.

→ Copying is done using the copy constructor for the data type of the formal parameters.

→ If the actual and formal parameters are of different data types, a type conversion is performed from the type of the actual parameter to that of the value formal value parameter.

→ When the invocation fun(2, x, y) is executed, 'a' is assigned the value 2, b ← x and c ← y

→ In case 'x' and 'y' are not of type int, then type conversion is done to x & y before assigning values to b and c.

→ When function terminates, destructors for the data types of the formal parameters destroy the formal value parameters.

→ When function terminates the formal value will not copy back to actual parameters. hence function invocation does not changes the value of actual parameters.

Template Functions

→ Writing same function for different data types using Templates called Template function. (Generic function)

Example:

```
int fun(int a, int b, int c)
{
    return a*b+c;
}
float fun(float a, float b, float c)
{
    return a*b+c;
}
```

→ Example above shows written another function to compute the same expression as first function.

→ In the second function a, b & c are of type float and the result is also of type float.

→ Rather than write a new version of the code for every data type of the formal arguments, we can write a generic code. as shown below (using template)

```
template <class T>        (note: class (or) typename)
T fun(T a, T b, T c)
{
    return a+b*c;
}
```

where T → can be of any data types.

Reference Parameters

→ It is defined as alternative name given to the other parameters.

Ex:

```
int x;
int &y = x;
```

→ Consider a function below, (generic function)

```
template <class T>
T fun(T a, T b, T c)
{ return a+b*c;
}
```

→ consider the operations involved when a function is invoked and when it terminates.

→ The use of value parameters increases the run time cost

→ when a, b and c are value parameters, the copy constructor for type 'T' copies the values of the corresponding actual parameters into the formal parameters a, b & c upon entry into the function.

→ At the time of exiting the function the destructor for type 'T' is invoked, and formal parameters a, b & c are destroyed.

→ Hence runtime cost increases & to decrease this runtime cost reference parameters are used in generic functions

Consider example below

```
template <class T>
T fun(T& a, T& b, T& c)
{
    return a+b*c;
}
```

```
int main()
{ int x, y, z;
  fun(x, y, z);
return 0;
}
```

→ where a, b & c are reference parameters

→ if function is invoked by the statement fun(x, y, z) where x, y & z are of some data types are used in the place of a, b & c.

→ Hence this program does not copy actual parameters values at the time of invocations and does not invoke the type 'T' destructor upon exit.

Constant Reference parameters

→ Reference parameters that are not changed by the function.

Example:

```
template <class T>
T fun(const T& a, const T& b, const T& c)
{ return a+b*c;
}
```

→ where values of a, b & c do not change

→ using 'const' qualifier to designate reference parameters that the function doesn't change the value.

Consider other example

```
template < class Ta, class Tb, class Tc>
Ta fun ( const Ta & a, const Tb & b, const Tc & c)
{   return a+b*c;
}
```

→ In the above program each parameters may be of a different type and the result is of the same type as the first parameter.

Return Values

① Return by Value

→ Function may make a value return (or) a const reference return and (or) reference return.

→ In return, the object that is being returned is copied into the invoking environment.

Example:

```
template < class Ta, class Tb, class Tc>
Ta function (const Ta &a, const& Tb, const Tc & c)
{   return a+b*c;
}
```

Note:

→ Copying is necessary in all versions of the functions 'function', as the result of the expression computed by this function is saved in a local temporary variable.

→ when the function terminates, the space allocated to this temporary variable is destroyed automatically.

→ To avoid losing this value, we copy it from the temporary variable into the return environment before releasing the space allocated to temporary variables.

② Return by reference:

→ Specify a reference return by adding the symbol '&' as a suffix to the return type. The function header.

Eg: T& Mystery (int i, T& z) { return z; }

→ Defines a function 'mystery' that makes a reference return of type T, as shown in example above and it return 'z';

→ This return would not involve copying the value of 'z' into the return environment.

→ When function mystery terminates, the space allocated to the value formal parameter 'i' and all local variables is released.

→ Because 'z' is simply a reference to an actual parameters, it is not effected.

③ Const reference return:

→ It is specified by adding the keyword const to the function header.

Eg: const T& mystery (int i, T& z)

→ It is similar to reference return except that the item returned is designated a constant object.

OVERLOADED Functions

→ Function is defined by the data types of the method's formal parameters and the number of formal parameter

→ C++ allows you to define two (or) more functions with the same name provided, so with different number of parameters with different datatypes

→ The Ability to define several function with the same name is called function overloading.

Ex:

```
int function_name(int a, int b)
 {  return a*b;
 }
void function_name(int a, int b, int c)
 {  cout<< "Sum=" << a+b+c <<endl;
 }
int main()
{  int d;
   function_name(3,4,5);
   d= function_name(6,7);
   cout<< d <<endl;
return 0;
}
```

DYNAMIC MEMORY ALLOCATION

i) The Operator new:

→ Dynamic memory allocation may be done using the C++ operator **new**.

→ This operator returns a pointer to the allocated memory.

Ex: Dynamically allocated memory for an integer, we must declare a variable to a pointer to an integer using following statement

```
int *y = new int
```

→ when the program needs to actually use the integer memory may be allocated. to it using this syntax It returns a integer pointer (y)

→ new operator allocates enough memory to hold a integer, and a pointer to this memory is returned and saved in 'y'.

Example:

```
int *y = new int;
   *y=10;
      (or)
int *y = new int (10);
      (or)
int *y;
 y= new int (10);
```

ii) One Dimensional Arrays

→ The size of arrays may not be known at compile time and may, in fact, change from one dimension invocation of the function to the next.

→ Memory for these arrays needs to be allocated dynamically.

Ex: float *x = new float [n];

→ To create a one-dimensional floating-pointer array 'x' at run time, we must declare 'x' as a pointer to a 'float' and then allocate enough memory for the array.

→ n → is size of the array represented in Square braces [ ]

→ The operator 'new' allocates memory for 'n' floating-point numbers and returns a pointer to the first of these.

iii) Exception Handling

Ex: float *x = new float [n];

→ when we executed above statement and the computer doesn't have enough memory for 'n' floating-point members.

→ In this case 'new' cannot possibly allocate the desired amount of memory and an exception of type bad_alloc is thrown.

→ we may detect the failure of 'new' by catching the exception with the try-catch construct:

```
float *x
try{ x = new float[n]; }
 catch (bad_alloc e)
{ // Enter only when 'new' fails
    cout << "Out of Memory" << endl;
    exit(1);
 }
```

iv) The operator delete

→ Dynamically allocated memory should be free when it is no longer needed.

→ The freed memory can then be reused to create new dynamically allocated structures.

→ We can use the C++ operator 'delete' to free space allocated using the operator 'new.'

Ex:

```
int *x = new int;
float y = new float[10];
delete [] y;
delete x;
```

→ where memory is allocated for x and array y of size 10 using new operator and deallocated using delete operator.

V) Two dimensional Array

→ In C++ the dimensions of the array should be known at the compile time.

→ When these mechanisms are used, it is difficult to write functions that allow a formal parameter which is a two-dimensional array whose second dimension is unknown.

→ The formal parameter is a two-dimensional array, we must specify the value of the second dimension also.
eg. a[ ][10] is a valid formal parameter for a function
a[7][ ] is not valid.

→ To overcome these limitations is to use dynamic memory allocation for 2-dimensional arrays.

→ When both dimensions of the array are known at compile time, the arrays may be created ~~at same time~~ using a syntax similar to that used for one-dimensional

Ex:

char c[7][5];

→ when at least one of the dimensions is known at compile time, the array must be created at run time using the new operator.

Ex: A two dimensional array for which the number of column's (5) is known at compile time, may be allocated using following syntax

```
char (*c) [5];
try { c = new char [n][5]; }
catch (bad_alloc)
{ // enter only when new fails
   cout << "out of memory" << endl;
   exit(1);
}
```

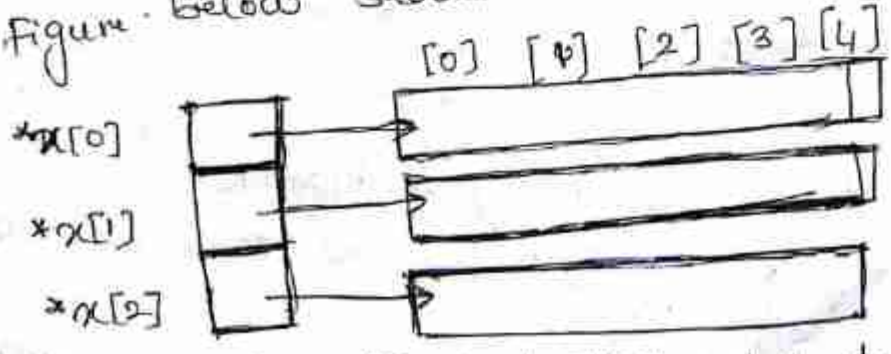
→ The number of rows 'n' may be determined at run time either via computation (or) user input.

→ when number of columns is not known at compile time the array cannot be allocated by a simple invocation of new (even if no of rows is known)

→ The number of rows 'n' may be determined at run time either via computation (or) user input.

→ when the number of columns is not known at compile time, the array cannot be created using 'new' operator.

→ Figure below shows the structure of '3 by 5' array 'x'



→ where x[0], x[1] and x[2] point to the first element of rows 0, 1, and 2 respectively.

→ If 'x' is to be a character array, then x[0:2] are pointers to characters and 'x' is itself a pointer to pointer to a character. The 'x' declared as follows.

```
char **x;
```

→ To create the memory structure for 2-dimensional array code as follows:

```
template <class T>
bool make2dArray(T **&x, int rows, int col)
{ // create a two dimensional array

    try { // create pointer for the rows
        x = new T* [rows];
        // get memory for each row
        for (int i=0; i<rows; i++)
            x[i] = new T[col];
        return true;
    }
    catch (bad_alloc) {
        return false;
    }
```

→ where a two-dimensional array of type T.

→ The array has number of rows as 'rows' and no of columns as 'col'.

→ The code first gets memory for the pointers x[0] through x[rows-1].

→ Next it gets memory for each rows of the array.

→ If one of these invocations of new throws an exception program control is transferred to the **catch** block and the value **false** is returned.

→ If none of the invocations of new throws an exception, the array connection is successful and 'make2dArray' returns the value true.

→ The failure of 'make2dArray' may also be reported to the invoking function by simply doing nothing.

→ Let us consider following code;

```
template <class T>
void make2dArray(T **x, int rows, int col)
{ // create a two-dimensional array
  // create pointer for the rows
  x = new T*[rows];
  // get memory for each row
  for(int i=0; i<rows; i++)
    x[i] = new T[col];
}
```

(prog. 1)

→ In above code, the invoking function can catch any exception thrown by new.

→ when 'new' throws any exception, then the following code can be used

```
try { makeArray(x, r, c); }
catch (bad_alloc)
{
    cout << " could not create x" << endl;
    exit(1);
}
// to determine a shortage of memory.
```

→ we can free the memory allocated to a two-dimensional array in (prog.1) by first freeing the memory allocated in the for loop to each row and then freeing the memory allocated for the row pointers. as shown in program below

```
template < class T >
void delete2dArray (T **x, int rows)
{ // Delete the two-dimensional array x
  // delete the memory for each row
  for (int i=0; i<rows; i++)
    delete [] x[i];
  // delete the row pointers
    delete [] x;
    x = NULL;
}
```

RECURSION

-> A recursive function (or) method invokes itself.

-> Recursion is divided into:

* Direct Recursion: The code for function 'f' contains a statement that invokes 'f'.

Ex:

```
func()
{
    func(); (or)
}
```

```
f()
{
    f();
}
```

* Indirect recursion: The function 'f' invokes some other function 'g', which invokes yet another function 'h' and so on until function 'f' again invoked

Ex:

```
f()
{
    g();
}
g()
{
    h();
}
h()
{
    f();
}
```

i) Recursive Mathematical Function:

→ In mathematics we define a function in terms of itself.

Ex: The factorial function $f(n) = n!$, for n an integer is defined as follows:

$$f(n) = \begin{cases} 1 & n \leq 1 \\ n f(n-1) & n > 1 \end{cases} \longrightarrow (1)$$

where: when $n \leq 1$ ($n = \ldots -2, -1, 0, 1$) $f(n) = 1$

ex: $n = -2, -1, 0, 1$

$f(-2) = f(-1) = f(0) = f(1) = 1$

Ex: when $n = 3$

$f(3) = 3 f(3-1) = 3 f(2) = 3 * 2 f(2-1) = 3 * 2 * 1$

$\boxed{f(3) = 6}$

→ Hence $f(n)$ is defined recursively, as the definition of f now contains an occurrence of $f(n)$ on the right side.

→ for recursive definition of $f(n)$ to be a complete specification of 'f', it must meet the following requirements:

1) * The definition must include a base call occurrence in which $f(n)$ is defined directly for one (or) more values of n.

* We assume that the domain of f is the non-negative integers and that the base covers the case $0 \leq n \leq k$ for some constant k.

⇒ In the **recursive** component all occurrences of f on the right side should have a parameter smaller than 'n' so that repeated application of the recursive component transforms all occurrences of f on the right side to occurrences of 'f' in the base.

→ In equation (1) the base is $f(n) = 1$ for $n \leq 1$; in the recursive component $f(n) = n\,f(n-1)$, the parameter of 'f' on the right side is $n-1$,

→ Repeated application of the recursive component transforms $f(n-1)$ to $f(n-2), f(n-3) \ldots$ and finally $f(1)$ which is included in the base.

Ex: $f(5) = 5f(4) = 20f(3) = 60f(2) = 120f(1) = \underline{120}$.

ii) Induction

→ In proof by induction, we establish the validity of a claim such as

$$\sum_{i=0}^{n} i = n(n+1)/2, \quad n \geq 0; \quad \rightarrow (1)$$

→ The claim is true for one or more base values of 'n'

→ We Assume that claim is true for values of n from 0 through m where m is an arbitrary integer greater than or equal to the largest 'n' covered in the base

~~for~~

→ This methodology leads to a proof that has three components

* Induction base
* Induction hypothesis
* Induction step.

Induction base:

→ From equation ① the induction base we establish correctness for $n=0$. At this time the left side is $\sum_{i=0}^{0} i = 0$, and the right side is also 0.

→ equation ① is valid when $n=0$.

Induction Hypothesis:

We assume that equation is valid for $n \leq m$ where 'm' is an arbitrary integer ≥ 0.

Induction step:

→ The Equation is valid for $n = m+1$

→ For this value of 'n', the left side is

$$\sum_{i=0}^{m+1} i = m+1 + \sum_{i=0}^{m} i \rightarrow ②$$

where $\sum_{i=0}^{m} i = \frac{m(m+1)}{2}$

eqn ② becomes

$$\sum_{i=0}^{m+1} i = m+1 + \frac{m(m+1)}{2} = \frac{(m+1)(m+2)}{2} = \text{RHS}$$

⇒ equals the right side.

→ Hence a proof by induction is not a circular proof in the same reasons that a recursive definition is not

iii) Recursive C++ functions

→ C++ allows us to write recursive functions.

→ A proper recursive function must include a base component.

→ The recursive component of the function should use smaller values of the function parameters. so that repeated invocation of the function results in parameters equal to those included in the base component.

Example①: Factorial of a number:

```
int factorial(int n)
{ // to compute factorial of 'n' number
  if(n<=1)
    return 1;
  else
    return (n * factorial(n-1));
}
```

→ From the above program the base component covers the cases when $n \leq 1$.

* consider factorial(2)

→ To compute 2 * factorial(1) in the else statement, the computation of 'factorial(2)' is suspended and 'factorial' invoked with n=1.

→ when factorial(2) is suspended, the program state is saved in a recursive stack.

→ The state is restored when factorial(1) completes.

Example ② Sum of array of 'n' elements :

* Template function 'sum' computes the sum of elements a[0] through a[n-1]. When n=0, the method returns the value 0.

```
template <class T>
T sum(T a[], int n)
{
    T sum_n = 0;
    for (int i=0; i<n; i++)
        sum_n += a[i];
    return sum_n;
}
```

→ We can write above program using recursive function as follows.

```
template <class T>
T sum(T a[], int n)
{
    if (n>0)
        return sum(a, n-1) + a[n-1];
    else
        return 0;
}
```

→ From above recursion, when n=0, the sum is 0; when n>0, the sum of 'n' elements is the sum of the first (n-1) elements plus the last element.

Example 3: [Permutations]

* Often we wish to examin all permutations of 'n' distinct elements to determine the best one.

ex: the permutations of the elements a, b and c are abc, acb, bac, bca, cab and cba.

* The number of permutation of 'n' elements is n!

→ The permutations of 'n' elements is quite difficult, we can develop a recursive one with modest effort.

* Let $E = \{e_1, \ldots, e_n\}$ denote the set of n elements whose permutations are to be generated;
* Let E_i be the set obtained by removing element 'i' from E
* Let perm(X) denote the permutations of the elements int set X;
* Let e_i • perm(X) denote the permutation list obtained by prefixing each permutation in perm(X) with element e_i

ex: if $E = \{a, b, c\}$ then $E_1 = \{b, c\}$

perm(E_1) = (bc, cb)

e_1 • perm(E_1) = (abc, acb)

→ For the recursion base, we use n=1.

Since only one permutation is possible when we have only one element, perm(E) = (e) where 'e' is the lone element in E.

→ when $n > 1$, and perm(E) is the list

→ $e_1 \cdot perm(E_1)$ followed by $e_2 \cdot perm(E_2)$ ..... $e_n \cdot perm(E_n)$

$perm(E) = e_1 \cdot perm(E_1), e_2 \cdot perm(E_2), e_3 \cdot perm(E_3), \ldots\ldots, e_n \cdot perm(E_n)$

→ This recursive definition of perm(E) defines perm(E) in terms of n perm(x)s, each of which involves an X with $n-1$ elements

For when $n=3$ and $E = (a, b, c)$

then $perm(E) = a.perm(\{b, c\}), b.perm(\{a, c\}), c.perm\{(b, a)\}$ → ①

$perm(\{b, c\}) = b.per(c), c.perm(b) =$ bac, acb.

$perm(\{a, c\}) = a.perm(c), c.perm(a) =$ ac, ca

$perm(\{b, a\}) = b.perm(a), a.perm(b) =$ ba, ab

Hence equation ① becomes

$perm(E) = a.[b.perm(c), c.perm(b)], b.[a.perm(c), c.perm(a)], c.[b.perm(a), a.perm(b)]$

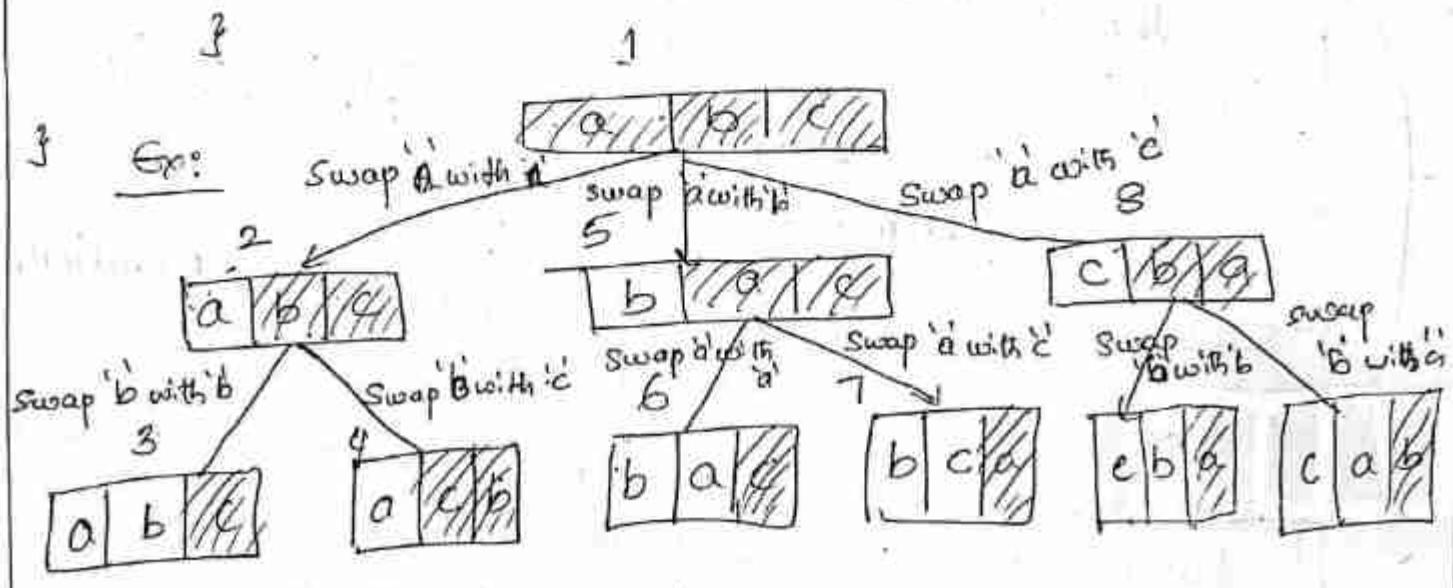
$= (ab.perm(c), ac.perm(b), ba.perm(c), bc.perm(a), cb.perm(a), ca.perm(b))$

$perm(E) = (abc, acb, bac, bca, cba, cab)$

program (permutation) [Recursive Method]

```
template <template T>
void perm (T list[], int k, int m)
          // generate all permutations of list[k:m]
{ int i;
  if (k==m)
  { // list[k:m] has one permutation, o/p is
    copy (list, list+m+1, ostream_iterator<T>(cout,""));
    cout<<endl;                    (or) cout<<list<<endl;
  }
else { // list[k:m] have more the one permutation
  // generate these recursively
  for (i=k; i<=m; i++)
    { swap (list[k], list[i])
      perm (list, k+1, m);
      swap (list[k], list[i]);
    }
}
```

Ex:



DATA OBJECTS AND STRUCTURES

→ A data object is a set of values (instances)

Example:

* boolean = {false, true}
* digit = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
* letter = {A, B, C, ..., Z, a, b, c, ..., z}
* natural Numbers = {0, 1, 2, ...}
* Integers = {±0, ±1, ±2, ±3, ...}
* string = {a, b, ..., aa, ab, ac, ...}

→ where boolean, digit, letter, natural Numbers, Integers and strings are called data objects.

→ The individual instances of a data objects as being primitive (or) automic (or) composed of values (or) instances of another data objects.

→ In the boolean data objects true & false are the instances.

→ In the case of letter data object the instances are refered as element.

For Example:

* Each instance of the data object Natural Numbers can be regarded as automic.
* In this case we are not concerned with a further decomposition of the instances of this data object
* Each instance of a Natural Number as being composed of several instances of the data object digit.

Eg: In the natural number 675 comprises the digits 6,7,5

→ The data object string is the set of all possible string instances. Each element in the string data object instance as being composed of several instances of the data object Letter.

Example: 'a trip to Hawaii' is a string data object where 'a', 't', 'r', 'i', 'p', 't', 'o', 'H', 'a', 'w', 'a', 'i' & 'i' are the instances of data object letter.

→ In additions to interrelationships, a set of operations is generally associated with any data object.

→ These operations may transform one instance of an object into another instance of that object, (or) into an instance of another data object (or) do both these transformations.

→ These operation could simply create a new instance without transforming the instances from which the new one is created.

Example:

* the operation 'add' defined on the natural numbers creates a new natural number that is the sum of the two numbers to be added. In that two numbers that get added are unaltered.

→ A data structure is a data object together with relationship that exit among the instances and among the individual element that compose an instance.

→ When we study data structure, we are concerned with the representation of data objects (with instances) as well as the implementation of the operations of interest for the data objects.

THE LINEAR LIST DATA STRUCTURE

→ Each instance of the data structure linear list (or) ordered list is an ordered collection of elements.

→ Each instance is of the form $(e_0, e_1, e_2, \ldots e_{n-1})$ where n is finite natural numbers & length is fixed.

e_i → elements of the list.

i → index of e_i

→ Elements may be viewed as atomic, as their individual structure is not relevant to the structure of the list.

→ when $n=0$ the list is empty.

→ when $n>0$, e_0 is the zeroth element and e_{n-1} is the last element.

→ we say that 'e_0' comes before 'e_1', 'e_1' comes before 'e_2' and so on.

→ other than precedence relation, no other structure exits in a linear list.

→ Examples of linear list:

1. an alphabetized (ordered by names) list of students in class,
2. a list of exam scores in non decreasing order.
3. An alphabetized list of members of BJP.
4. A list of gold-medal winners in the Olympics Men's basketball event ordered by year:

→ With these examples in mind we see the need to perform the following operations on a linear list.

* Create a linear list
* Destroy a linear list
* Determine whether the list is empty
* Determine the size of the list
* Find the element with a given index
* Find the index of a given element
* Delete, erase (or) remove an element given its index
* Insert a new element so that it has a given index.
* output the list elements in order, left to right

i) The Abstract Data Type Linear List

→ A linear list may be specified as an abstract data type (ADT) in which we provide a specification of the instances as well as of the operations that are to be performed.

→ The ADT specification is independent of any representation and programming language we have in mind.

→ All representations of the ADT must satisfy the specification, and the specification becames a way to validate the representation.

→ In addition, all representation that satisfy the specification may be used interchangeably in applications of the data type.

→ In ADT, table below we have omitted specifying operations to create and destroy instances of the data type.

→ All ADT specifications implicitly include an operation to create an empty instance and optionally, an operation to destroy an instance.

Table 1: Abstract Type linear list

{ instances:
ordered finite collections of zero (or) more elements

operations:
empty(): return true if the list is empty, false otherwise.
size(): returns the list size (no of elements in the list)
get(index): returns the (index)th element of the list
indexOf(x): returns the index of the first occurence of 'x' in the list, returns (-1) if 'x' is not in the list.

erase(index) : remove/delete the (index)th element, elements with higher index have their index reduced by 1.

insert(index, x) : insert 'x' as the (index)th element, elements with index ≥ index have their index increased by 1.

output() : output the list elements from left to Right.

---

ii) The Abstract class linear list

→ C++ supports two types of classes — abstract and concrete

→ An abstract class is a class that contains a member function for which no implementation has been specified. Such a function, called a pure virtual function, is specified using the zero initializer as in

```
virtual int myPureVirtualFunction(int x) = 0;
```

→ A concrete class contains no pure virtual function.

* only concrete classes may be instantiated.
* ~~they~~ create an instance (or) object only of a concrete class.
* we can create pointer to objects of an abstract class.

~~programs~~

program: ~~Abstract class specification of a linear list~~

P.T.O

Program: Abstract class specification of a linear list

```
template <class T>
class linearList
{
  public:
    virtual ~linearList() { };
    virtual bool empty() const=0; // return true if
                                  list is empty
    virtual int size() const=0; // return number of
                                elements in list
    virtual T& get(int index) const=0; // returns the
                                element index.
    virtual int indexOf(const T& theElement) const=0
      // return index of first occurance of the
                                element.
    virtual void erase(int index)=0;
      // remove the element whose index is
                                index.
    virtual void insert(int i, const T& e)=0;
      // insert element e so that its index is i
    virtual void output(ostream& out) const=0;
      // insert list into stream out
};
```

ARRAY REPRESENTATION

i) The array Representation:

→ we use an array to store the list elements.

→ we can pack several list instances into a single array. it is easier to use a different array for each instance

→ Individual elements of an instance are located in the array using mathematical formula.

→ One dimensional element to store the elements of a linear list.

The array element has positions.

e[n] = {e[0], e[1] e[2] . . . . e[n-1]}; . . .

where e → array element.
n → array length

→ each array position can be used to store single list element.

→ we need to map the elements of the list to position in the array and the formula is

$$\text{Location}(i) = i \quad \rightarrow (1)$$

eqn (1) says ith element of the list is stored in position i of the array.

→ The figure (1) below shows how the list [5, 2, 4, 8, 1] is stored in the array element using eqn (1)

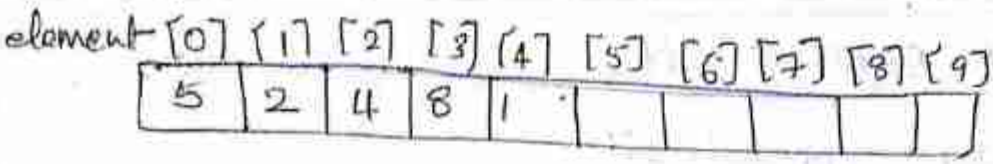


fig ①

→ The length of the array in fig ① is 10, and the size of the list is 5.

→ equation ① is the natural choice to map list elements into array position

→ Other formula also available to map list element into array position is;

location (i) = n − i − 1 [n – length of array] → ②

→ with eqn ② the list elements stores beginning at the right end of the array element, as show in fig ②

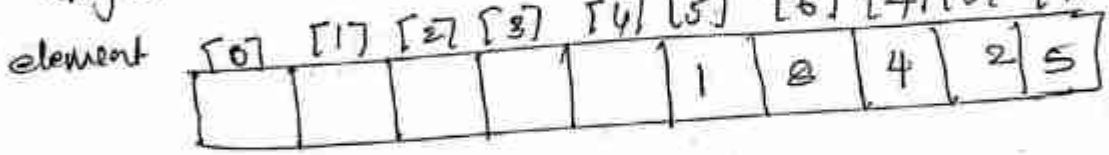
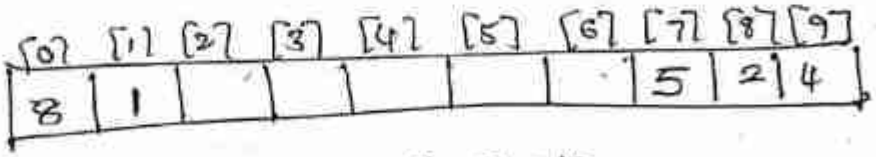


fig ②

→ we have other formula to store element of list in array.

location (i) = (location(0) + i) % n → ③

→ the list elements are stored in array position as shown in fig ③



Example: location(0) = 7 & n = 10

location (0) = (7+0) % 10 = 7
location (1) = (7+1) % 10 = 8
location (3) = (7+3) % 10 = 0

→ we may remove element 'ei' from the list by moving elements to its right down by 1 (to left)

Ex: list = [5, 2, 4, 8, 1]

when i = 1

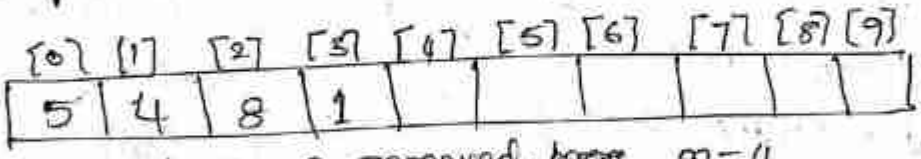
$e_1 = 2$ [second element in the list.]

* we have to move the elements $e_2 = 4$, $e_3 = 8$ & $e_4 = 1$, which are to the right of e_1.

→ we may remove element 'ei' from the list by moving elements to its left by one position.

Ex: list = [5, 2, 4, 8, 1]

* i = 1 => Second position $e_1 = 2$, and shift $e_2 = 4$, $e_3 = 8$ & $e_4 = 1$ to left by 1 position as shown in figure below



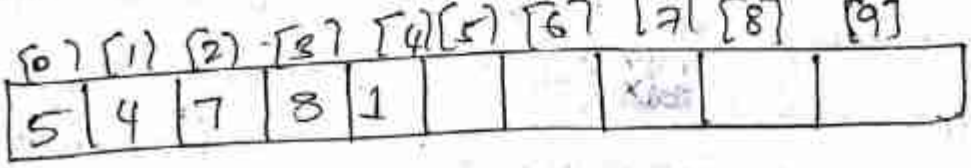
→ fig (4)

element $e_1 = 2$ removed. from $n = 4$

→ To insert an element so that it becomes element 'i' of a list, we must move the existing elements 'ei' to Right after ith position of insertion.

as shown in fig (5).

Ex: let us consider list shown in fig (4) and insert '7' in position i = 2. shift 8 & 1 to Right by one position



(i) Changing the length of a One-Dimensional Array

→ To increase (or) decrease the length of a one-dimensional array 'A' that contains elements in positions A[0:n-1].

* we first define an array of the new length
* Then copy the 'n' elements from 'A' to the new array
* Finally change the value of 'A' so that it references the new array.

→ To create an array of length 'n', it take time O(1)

→ the invocation of 'new' to create the new array may cause a 'bad_alloc' exception to be thrown.

→ If 'new' is successful in creating the new array the program below spends time of O(n) to copy elements from the source array into the destination array.

∴ The time complexity of program is O(n).

program:

[Note: #include<exception>]

```
template <class T>
void changelength (T *&a, int old_n, int new_n)
{
   if (new_n < 0)
      throw illegalParameterValue ("new length must be >=0");

   T* temp = new T[new_n]; // new array
   int num = min (old_n, new_n); // copy the numbers
   copy (a, a+num, temp)          ['algorith' header file]
   delete [] a;                   // deallocation.
   a = temp;
}
```

⇒ When an array is used to represent a data structure whose size increases by normally, the array length doubled whenever the array became full. This process is called as "array doubling."

⇒ When array doubled is used, the total time spent in changing the array length is not more than the total time spent in inserting element into the data structure.

(iii) The class Arraylist :

program for array list :

```
template <class T>
class arraylist : public linearlist<T>
{
  public:
    arraylist (int initialCapacity = 10);  // constructor
    arraylist (const arraylist<T>&);  // copy constructor
   ~arraylist () { delete [] element; }  // destructor

  // ADT methods
  const bool empty() { return listSize == 0; }
  const int size() { return listSize; }
  const T& get (int theIndex);
  const int indexOf (const T& theElement);
  void erase (int theIndex);
  void insert (int theIndex, const T& theElement);
  const void output (ostream& out);
  // addition method
  const int capacity()
  { return arrayLength; }
protected:
  const void checkIndex (int theIndex);  // throws illegalIndex if theIndex invalid
  T* element;  // 1D array to hold list elements
  int arrayLength;  // capacity of 1-D array
  int listSize;  // number of elements in list
};
```

program ①

Class Definition for 'arraylist':

→ In C++ the class 'arraylist' that implements the ADT 'linearlist' using equation;

$$location(i) = i$$

→ The program ① gives the class header, data members and function prototypes.

→ 'arraylist' is to be a concrete class that extends the abstract class linearlist, it must provide an implementation of all the methods of the abstract class 'linearlist'

→ class 'arraylist' not declared in 'linearlist'.

Constructor and Copy constructor for 'arraylist'

→ program ② below shows the class constructor and destructor,

→ The constructor creates an array whose length is 'initialCapacity', the default value for this length is 10. Also it sets 'arraylength' to 'initialcapacity' and 'listsize' to '0'.

→ The copy constructor makes a copy (or clone of an object.

~~→ The code for the empty~~

→ we assume that the time complexity of the 'new' operator is O(1), we see that the time complexity of the constructor is O(1) when 'T' is a primitive data type of C++.

→ when 'T' is user defined data type, the complexity of the constructor is O(initialCapacity).

→ The time complexity of 'empty', 'size' and 'initialCapacity' is O(1) and the copy constructor is O(n), where 'n' size of list to be copied.

Program ⑤

```
template <class T>
arraylist <T> :: arraylist (int initialCapacity)
{
  if (initialCapacity<1)
    {
      cout<< " length must be >0" <<endl;
    }
  else
      arraylength = initialCapacity;
      element = new T [arraylength];
      listsize=0;
}
template <class T>
arraylist<T>:: arraylist (const arraylist<T> & thelist)
{
  arraylength = thelist. arraylength;
  listsize    = thelist. listsize;
  element     = new T[arraylength];
  copy (thelist. element, thelist. element+ listsize , element);
}
```

Instantiating arraylist:

→ Linear lists that are represented as arrays may be creating/instantiated using statements given below:

* to create linear list with initial capacity 100.

```
linearlist *x = (linearlist) new arraylist<int> (100);
arraylist<double> y(100);
```

* Create linear list with the default capacity.

```
arraylist <char> z;
```

// create a linear list that is a copy of the list 'y'

~~arraylist<double> y(100);~~

arraylist<double> w(y);

Elementary Methods of 'arraylist';

→ program below shows the implementation of the 'checkIndex', 'get' and 'indexOf' methods in program below [program ③]

→ The complexity of 'checkIndex' and 'get' is O(1) and that of 'indexOf' is O(max{list-size, 1}).

program ③:

```
template<class T>
const void arraylist<T>::checkIndex(int theIndex)
{
    if (theIndex<0 || theIndex >= listSize) //verify index b/w 0 & listSize-1
    {
        cout<<" Range is exceeded"<<endl;
    }
}
template<class T>                    (int theIndex)
const T& arraylist<T>::get (int theIndex)
{
    checkIndex (theIndex);
    return element[theIndex]; // if no exception.
}
```

```
template <class T>
const int arrayList<T>::indexOf (const T& theElement)
{ // return (-1) if theElement not in list

  // search for the element
  int theIndex = (int) find (element, element + listSize, theElement);

  // check if the element was found.
  if (theIndex == listSize)
     return -1; // not found
  else return theIndex;
}
```

Removing an element

⇒ to remove (or) delete the (theIndex)th element from a list, we need to first find that the list contains an element with this index and then delete this element.

→ If the list does not have (theIndex)th element, an exception occurs because the ADT linearList, doesn't tell us what to do at this time.

→ when there is a (theIndex)th element, we can perform the deletion by using the copy algorithm to move elements to left by 1 position.

→ Function erase implements the delete/Remove operation.

→ When there is no (theindex)th element, an exception is thrown and taken by 'erase' is O(1).

→ When the list has (theindex)th element, 'listsize - theindex' elements are moved, taking Θ(listsize - theIndex) time.

program ④

```
template <class T>
void arrayList<T>::erase(int theIndex)
{ // delete the element whose index is 'theIndex'
  // throw illegalIndex exception if no such element
    checkIndex(theIndex);
  // valid index, shift elements with higher index
  copy(element + theIndex + 1, element + listSize, element + theIndex);
  element[--listSize].~T(); // invoke destructor
}
```

Inserting an element :

~~→ To insert a new element as the 'theindex' element in the list, we need to first move element 'theindex' through 'listsize-1' one position right~~

→ To insert a new element as the 'theindex' element in the list [steps as follows]

* we need to first move element 'theindex' through 'listsize-1' one position right.
* then insert the new element in position 'theIndex'.
* Then increment 'listsize' by 1.

→ The 'copy_backward' function moves elements begining with the rightmost one that is to be moved. (last element [copying from last element to first element from the list]

Note: the method doubles the length of the array elements in case the array has no space to accommodate the new element that is to be inserted

→ To determine exception to be throwon in terms of time complexity is O(1).

→ the time complexity for doubling array length is O(arraylength)

→ the time complexity for shift elements is O(listsize - theindex)

→ program ⑤ shows the 'Insert element at certain position.

program ⑤:

```
template <class T>
void arraylist<T>::insert(int theIndex, const T& theElement)
{ if(theIndex<0 || theIndex > listSize) //If index is invalid.
    { cout<<"Out of range"<<endl;
      return;
    }
    // if valid  check for space.
    if( listSize == arrayLength) // no space.
{ int newArrayLength = arrayLength*2;
    T* element1 = new T[newArrayLength];
    copy(element, element+listSize, element1);
}.
```

```
// shift elements right one position.
copy_backward (element+theIndex, element+listSize, element
                                                +listSize+1)

element[theIndex] = theElement;
listSize++;
}
```

The function output and overloading <<;

→ program ⑥ below gives the code for output.
→ The complexity of this code is O(listSize) under the assumption that the time to needed to insert a single element into the output is O(1)
→ Also gives to code to overload the stream insertion, operator <<.

program ⑥:

```
template<class T>
void arrayList<T>::output(ostream& out) const
{ copy(element, element+listSize, ostream_iterator<T>(cout, " ")); }

// overloading <<
template<class T>
ostream& operator<<(ostream& out, const arrayList<T>& x)
{
   x.output(out);
   return out;
}
```

iv) Iterators in C++

→ * Iterators behaves like pointers and are used to access container elements.

* They are often used to traverse from one element to another, a process known as Iterating through the container

→ The program below shows how to use a pointer 'y' to an array element to iterate through the array elements

Program: using an array Iterator;

```
int main()
{
    int x[3] = {0, 1, 2};
    // use a pointer 'y' to iterate through the array 'x'
    for (int *y = x; y != x+3; y++)
        cout << *y << endl;
    return 0;
}
```

→ In the above program the data type of the pointer *y is int*, which indicates that 'y' point to elements of type int.

→ The expression 'y++' increments the pointer so that it advances to the next element of the array.

→ Similarly, x+3 is a pointer '3' position from x; i.e points one position past the last element x[2] of the array.

→ The following code is equivalent to the for loop of program above.

```
for (int i=0 ; i!=3 ; i++)
    cout << a[i];
```

→ program above is generalized easily to output the elements of any object for which an iterator is defined.

```
for ( iterator i=start ; i!=end ; i++)
    cout << *i<<endl;
```

→ The above for loop output all elements in the range [start, end].

→ In the above loop 'iterator' is the datatype of the iterator, 'start' is the iterator value for the first element in the range and 'end' is the value the 'iterator' has when incremented one past the last element to be output.

→ The concept of an iterator is fundamental to writing generic code in c++. The program below shows copy function copy elements of any object that has an iterator for which the operators !=, *, and ++.

```
template<class Iterator>
void copy(iterator start, iterator end, iterator to)
{   while (start!=end)
    {   *to = *start;
        start++;
        to++;
    }
}
```

⇒ In C++ defines five categories of iterators

- input
- output
- forward
- bidirectional
- Random access.

→ input iterators additionally provide read access to the elements pointed at and support the pre- and post-increment operator (++).

→ output iterators provides write access to the elements and also permit iterator '++' operator.

→ Forward iterators may be advanced using the increment operator ++ (++)

→ bidirectional iterators may be incremented as well as decrement (--)

→ Random access iterators are the most general. They permit pointer jumps by arbitrary amount as well as arithmetic pointer.

V) An Iterator for arraylist:

→ C++ class iterator what will serve as a bidirectional iterator for arraylist.

→ Here In addition to the privious arraylist, two public methods begin() and end() to arraylist.

→ These methods, respectively, returns iterators whose value is a pointer to the first element of the list (element) and a pointer to the one position past the last element

→ The code for these two methods of 'arraylist' is

```
class iterator;
iterator begin()
{ return iterator(element);
}
iterator end()
{ return iterator(element + listsize);
}
```

→ Program below shows the code for the class iterator. Five 'typedef' statements are required by C++ to recognize our iterator class as a bidirectional iterator and generate proper code to employ bidirectional iterators.

```
class iterator
{ public:
        typedef bidirectional_iterator iterator_category
        typedef T value_type;
        typedef ptrdiff_t different_type;
        typedef T* pointer;
        typedef T& reference;
  // constructor
  iterator (T* theposition = 0)
  { position = theposition;
  }
  // dereferencing operators
  T& operator*() const
  { return *position;
  }
```

```
T& operator->() const
{ return &* position;
}
// increment
iterator& operator++() // pre increment
{ ++position;
return *this;
}
iterator operator++(int) // post increment
{ iterator old = *this;
++position;
return old;
}
// decrement
iterator& operator--()
{ --position;
return *this;
}
iterator operator--(int)
{ iterator old = *this;
--position;
return old;
}
// equality testing
bool operator!=(const iterator right) const
{ return position != right.position;
}
bool operator==(const iterator right) const
{ return position == right.position;
}
```

```
protected:
        T* position; // pointer to a list element
};
```

→ An instance of our list iterator may be created and initialized using a statement such as

```
arraylist<T>::iterator x = y.begin();
```

where 'y' is of type 'arraylist'.
'x' is of type 'iterator'

→ we can reverse the elements in a list 'y' using function 'reverse()' and sum the list elements using function 'accumulate()'.

```
reverse(y.begin(), y.end());
int sum = accumulate(y.begin(), y.end(), 0);
```

VECTOR REPRESENTATION

→ The vector is the most widely used container. It stores elements in contiguous memory location and enables direct access to any element.

→ A vector can change its size dynamically and therefore allocate memory as needed at run time.

→ The vector container supports random access iterators, and a wide range of iterator operations may be applied to a 'vector' iterator.

Class vector supports a number of constructors for creating vector objects

```
vector<int> v1;          // zero length int vector
vector<double> v2(10);   // 10-element double vector
vector<int> v3(v4);      // create v3 from v4
vector<int> v(5,2);      // 5-elements vector of 2s
```

→ The class 'vector' doesn't have a constructor equivalent to that of 'arraylist'; nor does it have methods with names get, indexOf, and output.

→ 'vector' has a methods 'erase' and 'insert' that respectively delete and add an element,

→ The vector methods need to know the memory address where operations is to be performed.

→ The difference between 'vector' and 'arraylist' is that the two classes throw different types of exceptions when something goes wrong.

→ In the program below define a class 'vectorlist' that uses a 'vector' to represent a linear list and whose methods have the same signature and behavior as those of 'linearlist'.

P.T.O

Program : An array linear list implemented using a vector

```
template< class T>
class vectorlist : public linearlist<T>
{
  public:
      vectorlist (int initialCapacity=10); // constructor
      vectorlist (const vectorlist<T> &); // copy constructor
     ~vectorlist () {   // destructor
        delete element; }

  // ADT methods
   bool empty() const;
   {
   ;
   int size() const;
   T& get (int theIndex) const;
   int indexOf( const T& theElement) const;
   void erase (int theIndex);
   void insert( int theIndex, const T& theElement);
   void output (ostream & out) const;

  // Additional method
   int capacity() const;

  // iterators to start and end of list
                vector<T>::iterator iterator;

   iterator begin()
   { return element->begin(); }
   iterator end() {
    return element->end();
   }
```

```
protected:
    // Additional members of vectorlist
    void checkIndex(int theIndex) const;
    vector<T>* element;  // vector to hold list elements
};
```

program: Constructor for vectorlist

```
template<class T>
vectorList<T>::vectorList(int initialCapacity)
{  if (initialCapacity<1)
   {  cout<<"Must be >0"<<endl;
      return;
   }
   element = new vector<T>;  // create an empty vector with space 0
   element->reserve(initialCapacity);
                  // increase vector capacity from '0' to initialCapacity
}

template<class T>
vectorList<T>::vectorList(const vectorList<T>& theList)
{  // copy constructor
   element = new vector<T>(*theList.element);
}
```

SINGLY LINKED LISTS AND CHAINS

i) The Representation:

→ In a linked representation each element of an instance of a data object is represented in a cell (or) node.

→ Nodes not be components of an array, and no formula is used to locate individual elements.

→ Each node consists of 2-components;

* data field → consists of elements of instance
* link field → holds address of next node

Let $L = (e_0, e_1, e_2, \ldots, e_{n-1})$ be a linear list.

→ Each element e_i is represented in Separate Node.

→ each node has exactly one link field that is used locate the next element in the linear list.

i.e node for e_i links to that for e_{i+1}, $0 \le i < n-1$.

→ The last node of the linked list has the address field (or) link field is "NULL".

→ figure below shows the linked list representation of the list $L = (e_0, e_1, e_2, \ldots e_{n-1})$

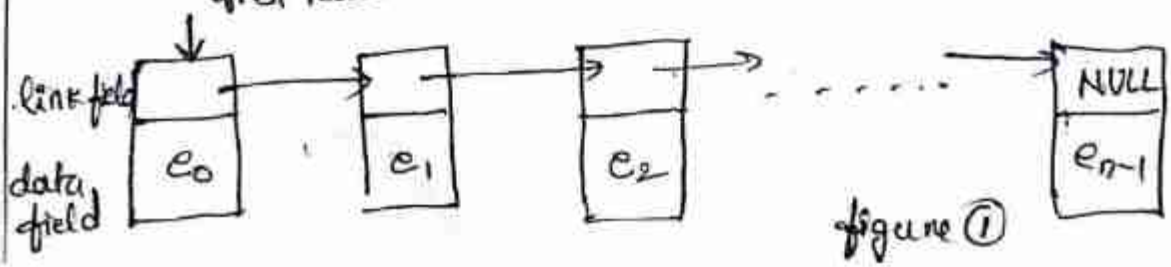


figure ①

→ To locate the element e_2, for example, we must start at 'firstnode': follow the pointer in 'firstnode' to the next node; follow one more pointer to get to the node with e_2.

→ Nodes are ordered from left to right with each node linking to the next, and the last node has a NULL link, the structure is also called a chain.

→ The figure ① above is called as • singly linked list.

* To remove element from ~~list~~ linked list.

Example: the element e_2 whose index is 2 from the chain of figure ②, we do the following

- locate the second node (i.e the node with e_1) in the chain
- link the second node to the fourth node.

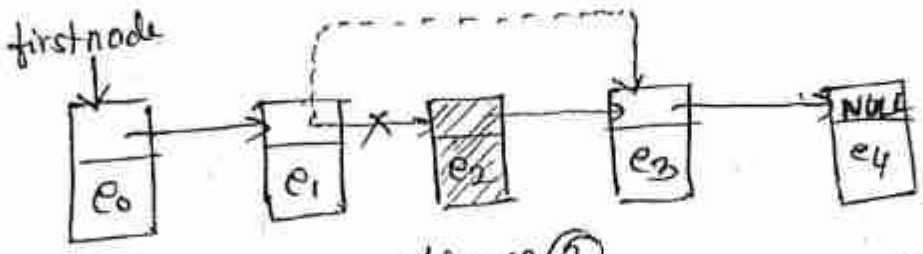


figure ②

→ the removal of the third node from figure ② automatically decrements the index of the succeeding node by 1

* To Insert new element as the (index)th element in a chain
→ we need to first locate the (index−1)th element and then insert a new node just after it.
→ figure ③ shows the link changes needed for the two cases index = 0 and 0 < index ≤ listsize

→ Solid pointers exist prior to the insert, and those shown as broken lines exist following the insert

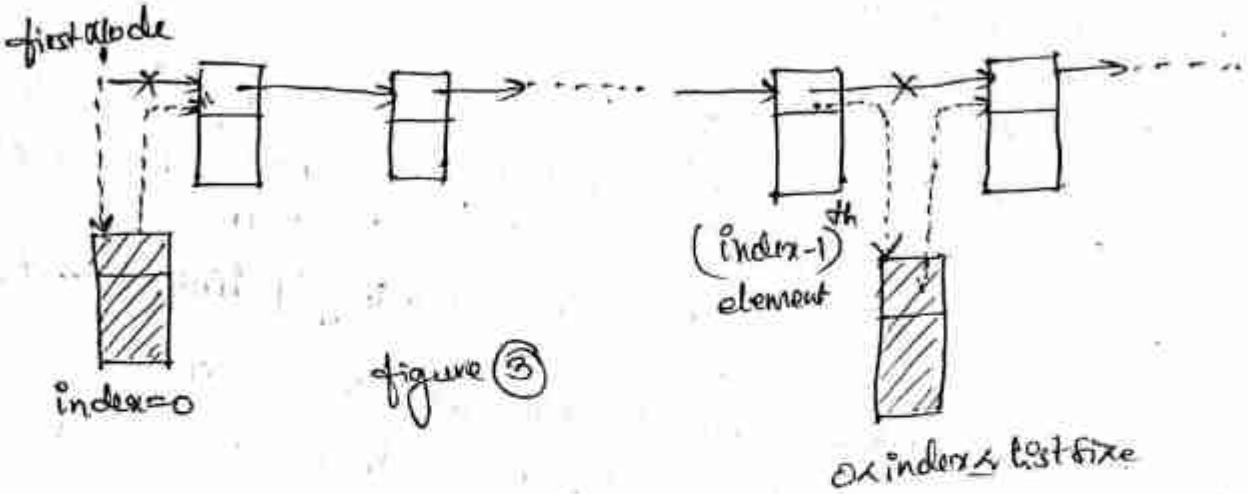


figure (3)

ii) The struct chainNode:

→ we define a struct 'chainNode' & a class chain.

→ program ① below gives the struct 'chainNode', which defines the data type of the nodes used.

→ The data member 'element' holds a list element and the data member 'next' holds a pointer to the next node in the chain and is the node's link field.

→ The two constructors of the 'chainNode' use the syntax 'this→element' and 'this→next' to access the data members of the constructed instance.

program ①: struct definition for a chain Node.

```
template<class T>
struct chainNode
{   // data members.
    T element;
    chainNode<T> *next;

    // methods.
    chainNode() { }
    chainNode(const T& element)
    { this->element = element;
    }
    chainNode(const T& element, chainNode<T>* next)
    { this->element = element;
      this->next = next;
    }
};
```

iii) The class 'chain'

Header, empty and size

→ The class chain implements a linear list as a singly linked list of nodes in which the last node has the pointer NULL.

→ program ② shows the class header, data members and code for the 'empty' and 'size' methods.

→ The data members are 'firstNode' and 'listsize'

→ 'firstNode' is a pointer to the first node.

→ When the chain has no first node, that is, when the chain is empty, 'firstNode' is NULL.

→ 'listSize' gives the number of elements in the list, which equals the number of nodes in the chain.

program ② : Header for the class chain

```
template<class T>
class chain : public linearList<T>
{
   public:
      chain(int initialCapacity=10); // constructor
      chain(const chain<T>&); // copy constructor
      ~chain();

      // ADT methods.
   bool empty() const { return listSize == 0; }
   int size() const { return listSize; }
   T& get(int theIndex) const;
   int indexOf(const T& theElement) const;
   void erase(int theIndex);
   void insert(int theIndex, const T& theElement);
   void output(ostream& out) const;

 protected:
      void checkIndex(int theIndex) const;
            // throw error (or exception if 'theIndex' is invalid
   chainNode<T>* firstNode; // pointer to first node in chain
   int listSize;     // number of elements in list
};
```

constructors and copy constructor;

→ program ③ gives the constructor and copy constructor for chain.

→ To create an empty chain, we need to set the first node pointer 'firstNode' to NULL.

→ when a linear list is represented by an array, we do not allocated space for the expected maximum number of elements at the time the chain is created.

→ we have a constructor which has 'initialCapacity' as a formal parameter in order to be compatible with the class arraylist.

program ③: constructor and copy constructor.

```
template<class T>
chain<T>::chain(int initialCapacity).
{   if (initialCapacity<1)
      { cout<<" Must be >0";
      }
   firstNode = NULL;
   listSize = 0;
}
template<T>  // copy constructor
chain<T>::chain(const chain<T> &theList)
{   listSize = theList.listSize;
    if (listSize == 0)  // list is empty
      {  firstNode = NULL;
         return;
      }
```

```
// non empty list
chainNode<T> * sourceNode = theList.firstNode. // node in the list to copy from

firstNode = new chainNode<T>(sourceNode->element);
                    // copy first element of 'theList'

sourceNode = sourceNode->next;
chainNode<T> *targetNode = firstNode;
while (sourceNode != NULL)
{  // copy remaining elements
   tempNode = new chainNode<T>(source->element);
   targetNode->next = tempNode;
   targetNode = targetNode->next;
   sourceNode = sourceNode->next;
}
target->next = NULL; // end of the chain
}
```

The destructor:

→ The destructor deletes the nodes of the chain one by one.

→ The strategy used in our destructor code is to repeatedly delete the first node in the chain until the remaining has no first node.

→ we must save the pointer to the second node.

```
template<class T>
chain<T>::~chain()
{ while (firstNode != NULL)
  { chainNode<T> *nextNode = firstNode->next;
    delete firstNode;
    firstNode = nextNode;
  }
}
```

the method 'get'.

→ To find the (theIndex)th element of a chain, however, we must start at the first node and follow the 'next' links until we reach the desired node;

→ we must follow 'theIndex' number of links.

→ program below gives the method to return the (theIndex)th element.

→ The method 'checkIndex' is the same as that defined for 'arrayList'. The complexity of 'chain<T>::get' is O(theIndex).

```
template <class T>
T& chain<T>::get(int theIndex) const
{ // Return the element whose index is 'theIndex'.
  // check index if it is not valid throw error
  checkIndex(theIndex);

  // move to desired node
  chainNode<T> *currentNode = firstNode;
  for(int i=0; i<theIndex; i++)
     currentNode = currentNode->next;
  return currentNode->element;
}
```

The method 'indexOf':

→ Program below shows the code for 'indexOf'

→ To traverse from one element to other elements when chain is used, the only way to go from one node to the next is to follow the link or pointer in the current Node.

→ The time complexity of 'chain<T> :: indexOf' is O(listSize)

```
template <class T>
int chain<T>::indexOf(const T& theElement) const
{ // Return index of first occurance of theElement.
  // Return -1 if theElement not in list
  // search the chain for theElement
  chainNode<T> *currentNode = firstNode;
  int index=0;  // index of currentNode.
  while ((currentNode != NULL) && (currentNode->element != theElement))
  {
     currentNode = currentNode->next; // move to next Node;
     index++;
  }
  if (currentNode == NULL)   // make sure we found matching element.
     return -1;
  else
     return index;
}
```

The method erase

→ There are three cases to consider;

* theIndex < 0 (or) theIndex ≥ listSize. In this case the erase operation fails because there is no (theIndex)th element to erase.
* The (zero)th element is to be erased from a nonempty chain.
* An element other that the zeroth element is to be erased.

→ when list is empty, no elements to erase.

→ If theIndex < 0, no elements to erase (Invalid index)

→ theIndex = 0, [erase (zero)th element, when list is nonempty]

→ theIndex = listSize - 1 [erase the element of given index]

```
template <class T>
void chain<T> :: erase (int theIndex)
{  checkIndex(theIndex); // check the index if valid (or) not:
   // Delete the element whose index is theIndex

   // valid Index
   ChainNode<T> * deleteNode;
   if (theIndex == 0)
   {  // remove first node from chain
      deleteNode = firstNode;
      firstNode = firstNode->next;
   }
   else
```

```
else
{
    chainNode<T> *p = firstNode;
    for (int i=0; i<theIndex-1; i++)
        p= p->next;
    deleteNode = p->next;
    p->next = p->next->next; // remove deleteNode from chain
}
listSize--;
delete deleteNode;
}
```

The method 'insert':

→ Insertion and erasing work in a similar way.

→ To insert a new element as the (theIndex)th one in a chain, we need to first locate the [(theIndex)-1]th element and then insert a new node just after it.

→ Time complexity of Insertion is O(listSize).

→ program below gives the code for insertion.

```
template<class T>
void chain<T>::insert(int theIndex, const T& theElement)
{ if (theIndex < 0 || theIndex > listSize)
    { cout << "Out of Range." << endl;
      return;
    }
  // if valid index
```

cont:

cont....

```
if (theIndex == 0)  // Insert at front
{
chainNode<T>* temp = new chainNode<T>;
   temp->element = theElement;
   temp->next = firstNode;
   firstNode = temp;
}
else { // use 'p' to get to insert
   chainNode<T> *p = firstNode;
   for (int i=0; i<(theIndex-1); i++)
      p = p->next;
   // Insert after 'p'
   temp = new chainNode<T>
   temp->element = theElement;
   p->next = temp;
   temp->next = p->next->next;
}
listSize++;
}
```

Outputting a chain:

→ program below gives the code for the 'output' and also overloading '<<' operator.

```
template <class T>
void chain<T> ::output (ostream &out) const
{ for (chainNode<T>* currentNode = firstNode; currentNode != NULL;
                    currentNode = currentNode->next)
   out << currentNode->element << endl;
}
```

```
//overload
template <class T>
ostream& operator<< (ostream &out, const chain<T> &x)
{  x.output(out);
   return out;
}
```

The member class 'iterator'

→ By using next pointers we can efficiently move from a Node to its Next node in the chain.

→ There is no efficient way to move from a node to its predecessor ((or) privious Node).

→ For 'ArrayList' we defined a bidirectional iterator that enabled us to move from any list element to both its next element and its privious element. in O(1) time.

```
class iterator
{ protected: chainNode<T> *node;

  public:
        // constructor
        Iterator (chainNode<T> *theNode = NULL)
        { node = theNode;
        }
```

```
// dereferencing operators
T& operator*() const { return node->element; }
T* operator->() const { return &node->element; }

// increment
iterator& operator++() // pre increment
{ node = node->next;
  return *this;
}
iterator operator(int) // post increment
{ iterator old = *this;
  node = node->next;
  return old;
}
// equality testing
bool operator!=(const iterator right) const
{ return (node != right.node); }
bool operator==(const iterator right) const
{ return (node == right.node); }
};
```

iv) Extensions to the ADT 'linearList':

→ ADT includes additional functions such as

* clear() → remove all elements from the list
* push_back(Element) → Insert 'theElement' at the end of the list.

→ The program below shows the extension of ADT 'linearList'

Program: - Abstract class for Extended linear list:

```
template <class T>
class extendedLinearList : linearList <T>
{ public:
      virtual ~extendedLinearList () { };
      virtual void clear() = 0; // empty the list
      virtual void push_back(const T& theElement) = 0;
                  // Insert theElement at end of list
};
```

V) The Class 'extendedChain'

→ class 'extendedChain' that provides a linked implementation of the abstract class 'extendedLinearList'.

→ The easiest way to arrive at the class 'extendedChain' is to derive it from 'chain'.

→ To insert an element at the end of a chain, we add a new data member 'lastNode' that points to the last node in the chain.

→ In addition of this new data member requires us to make changes in the implementation of the methods 'erase' and 'insert', because these methods may change the last node in the chain.

→ when these method changes last node, they must also update the new data member 'lastNode'.

→ the class 'extendedChain' will declare the data member 'lastNode', provides modified implementations of the methods 'erase' and 'insert'.

→ program below gives the methods clear and push_back

program: The methods 'clear' and 'push_back' of "extendedChain<T>"

```
template <class T>
void extendedChain<T>::clear()
{  while (firstNode != NULL)
   {  ChainNode<T> *nextNode;
      nextNode = firstNode->next;
      delete firstNode;          // delete first node
      firstNode = nextNode;
   }
   listSize = 0;
}
template <class T>
void extendedChain<T>::push_back(const T& theElement)
{ // Insert element at the end.
   ChainNode<T> *newNode = new ChainNode<T>;
   newNode->element = theElement
   newNode->next = NULL;
   if (firstNode == NULL)      // chain is empty
      firstNode = lastNode = newNode;

   else
   {  chainNode<T> *temp;
      temp = firstNode;
      while ( temp->Next != NULL)
      {     temp = temp->Next
      }  lastNode = temp;
```

P.T.O

```
lastNode->next = newNode;
lastNode = newNode;
}
listSize++;
}
```

VI) Performance Measurement

Memory comparison:

Arraylist

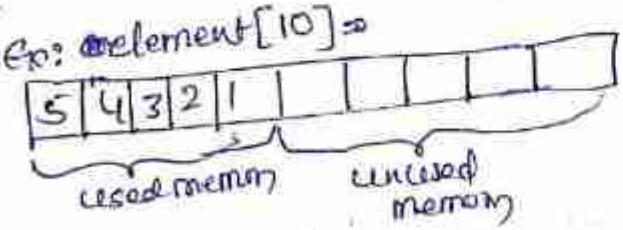
→ Array is the contiguous block memory, allocation.

→ Elements are stored in consecutive manner in memory

→ Size of array is fixed

→ There is unused memory in array

Ex: element[10] =



Total memory allocation is =

10 x Size of integer = 10 x 2 = 20 bytes

→ Memory may not available as one large block.

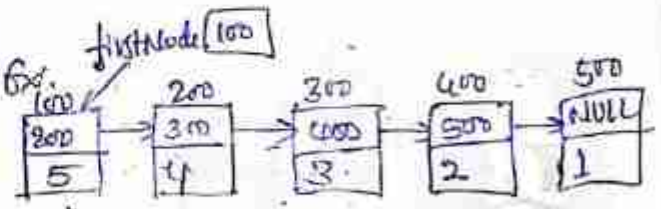
Linked list

→ It is a random memory allocation to each and every node

→ Elements are stored at any available place or address of node is stored in privious node

→ Size of linked list is not fixed it depends on number of element to be stored.

→ No unused memory in linked list



Note

→ Extra memory used for 'Link' field. (or) pointer variable

Total memory required is =

(5 x 2) + (link field x 2)

= (5 x 2) + (5 x 2) = 10 + 10 > 20 bytes

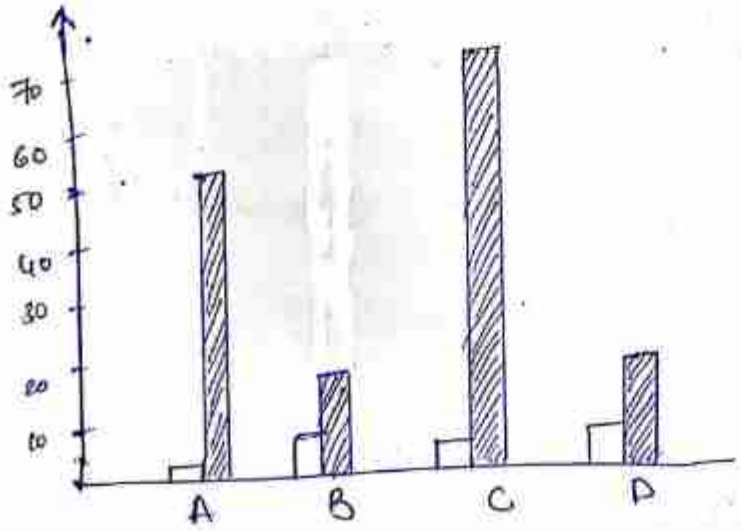
→ memory may be available as multiple small blocks.

Run-Time Comparision

* chain<T>::get() to be much slower than arraylist<T>::get()
* The complexity of chain<T>::get() is O(listsize)
  The complexity of arraylist<T>::get() is O(1)
* 'chain<T>::get()' took 13.2 second to perform operation while 'arraylist<T>::get()' took 1.0msec
* figure (a) & (b) shows the time taken by arraylist and chain for operation sequences

operations	arraylist	chain
get()	1.0 Sec	13.2 Sec
indexOf()	3.2 Sec	13 Sec
best-case insert	2.1 ms	45.1 ms
average insert	1.5 ms	49.3 Sec
worst case insert	2.5 Sec	12.9 Sec
best case erase	2 msec	2.1 ms
average erase	1.5 Sec	68.8 Sec
worst case erase	2.5 Sec	12.9 Se

figure (a)
Time for 50000 operations



A → average insert
B → Worst-case insert
C → Average erase
D → Worst-case erase
→ array list
→ Linked list

figure (b)

→ Even though the insert and erase sequences used in our worst-case test for chains cause our methods to do the maximum work, the sequences do not ensure maximum run time because of the cache effect.

(*) → The average case insert time were obtained by inserting into random positions of the linear list.

→ Nodes that are adjacent in the chain are randomly located in memory.

→ Hence when you move down the chain from left to right, you need to access random memory locations. This result in many cache misses.

→ The same is true for the average case erase experiment, which uses a randomly constructed chain.

(*) → For the worst case experiment, successive inserts are made at the right end of the chain.

→ Since for our experimental setup, successive calls to new return nodes that are adjacent in memory, nodes that are adjacent in the chain also are adjacent in memory.

→ So, when you move down the chain from left to right you access adjacent memory; a memory access pattern that is favored by the cache management scheme. So, the number of cache misses is reduced.

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

Definition: A stack is a linear List in which insertions (Push) and removals (pop) take place at the same end. This end is called the **top**; the other end of the list is called the **bottom**. (LIFO operation).

→ Figure (a) below shows a stack with four elements.

Suppose we wish to add element E to the stack of figure (a). This element will have to be placed on top of element D, giving us the configuration of fig (b)

→ if we are removing elements then first remove E then remove C, then, top is element 'B' in stack shown in fig (c) and B from fig (a)

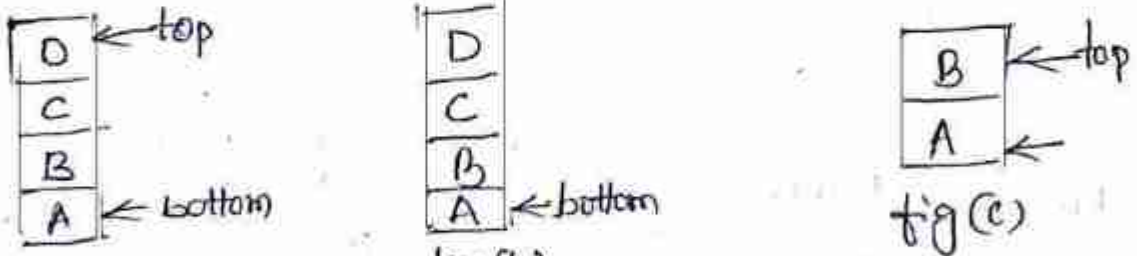


fig (a) fig (b) fig (c)

fig: stack configurations

The Abstract Data type

ABSTRACT DATA TYPE (ADT of stack)

→ The ADT stack is specified in figure below.

```
Abstract DataType stack
{
   instances:
      linear list of elements; one end is called the bottom; the
      other is the top;

   operations:
      empty() : Returns true if the stack is empty, returns
                false otherwise.
      size() : Return the number of elements in the stack;
      top() : Returns the top element of the stacks;
      pop() : Remove the top element from the stacks;
      push(x) : Add element x at the top of the stacks;
}
```

C++ Abstract class corresponds to the stacks ADT.

```
template<class T>
class stacks
{ public:
   virtual ~stacks() { }
   virtual bool empty() const = 0; // returns true iff stacks is empty
   virtual int size() const = 0; // returns number of elements in stack
   virtual T& top() = 0; // returns reference to the top element
   virtual void pop() = 0; // remove the top element
   virtual void push(const T& theElement) = 0; // Insert theElement
                                               at the top of the stacks
};
```

ARRAY REPRESENTATION

→ A stack is a linear list with the restriction that additions and deletions take place at the same end.

i) Implementation as a Derived class

→ program below gives the class 'derivedArraystack' whose base classes are 'arrayList' and 'stacks'.

→ The constructor for 'derived Arraystacks' simply invokes the constructor for the base 'arrayList' which allocates a one-dimensional array whose capacity is 'initialCapacity'.

→ Default value for initialCapacity = 10

```
Templati<class T>
class derivedArrayStacks : public arrayList<T>, public stacks<T>
{
  public:
      derivedArrayStacks (int initialCapacity=10)
                  : arrayList<T> (initialCapacity){ }

      bool empty() const
      { return arrayList<T>::empty();
      }
      int size() const
      { return arrayList<T> :: size();
      }
};
```

..... contd.....

P. T. O

```
T& top()
{
  if (arrayList<T>::empty())
  { cout<<"stack is empty"<<endl;
    return;
  }
  else
    return get(arrayList<T>::size()-1);
}
void pop()
{
  if(arrayList<T>::empty())
  { cout<<"stack is empty"<<endl;
    return;
  }
  else
    erase(arrayList<T>::size()-1);
}
void push(const T& theElement)
{
  insert(arrayList<T>::size(), theElement);
}
};
```

→ The complexity of the constructor is O(1) when 'T' is a primitive data type and is O(initialCapacity) when 'T' is user defined data type.

→ The complexity of push, pop and top is O(1)

→ top() and pop() check whether the stack is empty before they invoke the base class methods get() and erase() respectively.

→ the invoked base-class methods will throw an exception when invoked with an empty stack.

→ LIFO stack discipline is enforced on instaces of type 'derivedArrayStacks'.

ii) The class 'arraystacks':

→ Faster implementation of an array stacks is to develop a class that employs an array stack to hold the stack elements.

→ program below gives class 'arraystacks'.

→ The bottom element of the stack is stored in element[0] and top element is element[top].

```
template <class T>
class arrayStack : public stack<T>

  private:
     int stackTop; // current top of stack

  public:
     arrayStack (int initialCapacity=10);
     ~arrayStack() { delete [] element; }
     bool empty() const { return stackTop==-1; }
     int size() const
     { return stackTop+1;
     }
     T& top()
     { if (stackTop == -1)
        { cout << "stack is empty" << endl;
          return;
```

```
{    if (stackTop == -1)
       { cout << "stack is empty" << endl;
         return;
       }
     else element[stackTop--].~T(); //destructor to T
```

```
// constructor
template <class T>
arraystack<T>::arraystack(int initialCapacity)
{ if (initialCapacity < 1)
  { cout << "invalid stack capacity" << endl;
    return;
  }
  arrayLength = initialCapacity;
  element = new T[arrayLength];
}
// push operation
template <class T>
void arraystack<T>::push(const T& theElement)
{ if (stackTop == arrayLength - 1);
                                        * arrayLength);
      arrayLength = arra
    }
  element[++stackTop] =                  // add at top of stack
}
```

iii) Performance Measurement

→ The array stack classes 'arrayStack' (AS), 'derivedArrayStack' (DAS) and the C++ class 'stack' implement all methods of the stack ADT so as to have the same complexity, the observed performance of the methods is expected to be different for each class.

→ Define an n-sequence to be a sequence of n push operations followed by 'n' alternating sequence of n top and n pop operations.

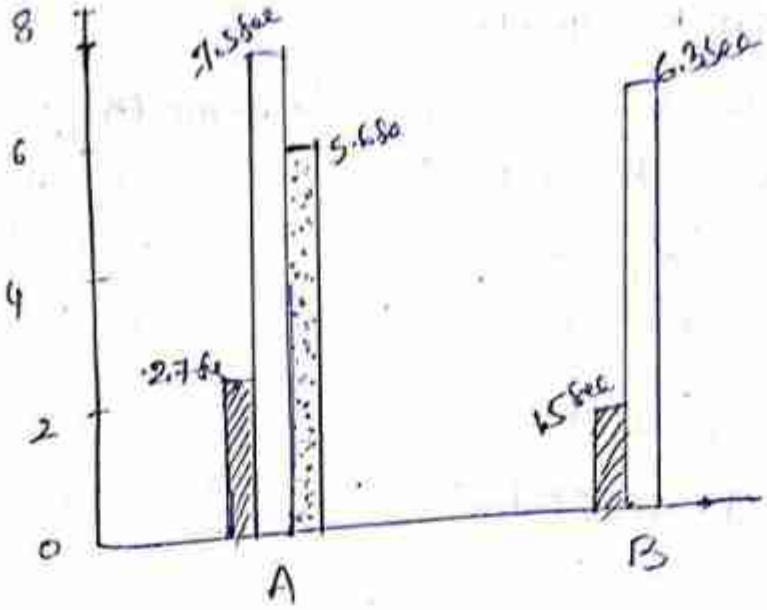
→ fig (a) shows the measured times to perform a 50,000,000 - sequence, and fig (b) shows these time as a bar chart.

→ For all stack classes except the 'stack' class, we obtain the run times for the cases.

* Start with a stack having a default capacity
* Start with a stacks whose initial capacity = 50,000,000

class	Initial Capacity	
	default	50,000,000
arrayStack	2.7 Sec	1.5 sec
derivedArrayStack	7.5 Sec	6.3 Sec
Stacks	5.6 Sec	—

fig (a) - time taken by different array stacks - implementation



A → default initial capacity
B → initial capacity is 50,000,000.

→ when using a chain to repr ...ck, we must decide which end of the chain corresponds to the stack top. ... associate the Right end of the chain w.r. the stack, then the stacks operations top, push and Pop are implemented using ... et(size()-1), ~~insert~~ ... erase(size()-1) respectively. at(size(), theElement) and ... O(size()) time.

⇒ If we associate the ... top, then the chain operations to use are get(0), insert(0, theElement) and erase(0). ... O(1) time (

class linkedStacks;

→ In the case of class 'arraystack', we can improve the run-time performance by customizing the code and not deriving our linked-stack class, from the class **Chain**.

→ program below shows the class 'linked stack'.

```
template<class T>
class linkedstacks : public stack<T>, public chainNode<T>
{ private:
        chainNode<T> *stackTop; // pointer to stack top
        int stackSize;          // no of elements in stack

  public:
  linkedstack (int initialCapacity = 10)
  { stackTop = NULL;
    stackSize = 0;
  }
~linkedStack();
bool empty() const
{  return stackSize == 0;
}
int size() const { return stackSize; }

T& top()
{  if (stackSize == 0)
   { cout << "stack is empty" << endl;
     return;
   }
   else
     return (stackTop -> element);
}
```

void pop();
void push(const T& theElement)

chainNode<T> *temp = new

temp->next = stackTop;
stackTop = temp;

template <...
linkedStack<T>::~linkedStack()
{ while (stackTop != NULL)

stackTop->next;

}
}

pop operation

template <class T>
void linkedStack<T>::pop()
{ if (stackSize == 0)
{ cout << "Stack is empty" << endl;
return;
}
chainNode<T> *nextNode = stackTop->next;

delete stackTop;

i) Parenthesis Matching

Problem Description:

* In this problem we are to match the left and right parentheses in a character string.

Example: the string (a * (b+c)+d)

0 1 2 3 4 5 6 7 8 9 10

* Left from above example left parentheses at positions 0 and 3 and right parentheses at positions 7 and 10.

Left parentheses at position '0' matches right at position '10', while the left parenthesis at position '3' ... right parentheses at

objective is to write a c++ program that inputs a string and outputs the pairs of matched p as well as those parentheses for which there is no match

scan the input expression from left to right, then right parentheses is matched to the m ntly unmatched left paran

* This observation to save the position of left parenthes a stack as they are encountered in a left to right scan.

→ when Right parentheses is encountered, it is matched to the left parenthesis at the top of the stack

→ The left matched parenthesis is deleted from the stack

defined

pression

0; i<

```
else
  if (exp.at(i)==')')
try{ // remove location of matching '(' from stack
   cout << s.top() << ' ' << i << endl;
   s.pop; //unstack match.

{ // stack was ...
   ...t << " No match

// remaining '(' in stack a :s!" , no match exits
 while (!s.empty())                 I or right parenthes
{          "No matching for left parenthesis"   s.top()<<
                                                endl;
   s.pop();
```

Complexity:

→ from the program above the time complexity of program is O(n) where 'n' is the length of the input expression.

→ push and pop takes O(n) where expression.

→ one Pop takes O(1).

py://t/rcx/?0Y/

ii) Towers of Hanoi.

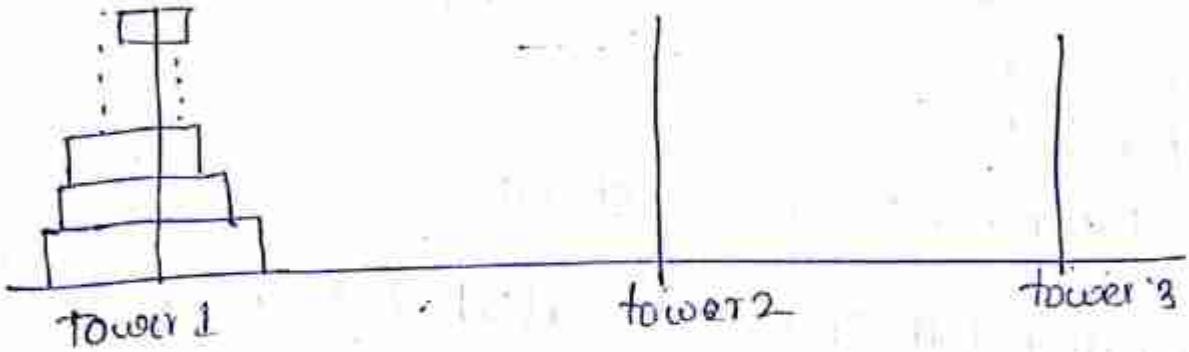
problem description:

* There was a diamond tower (tower 1) with 64 golden disks; as shown in figure below.
* The disks were of decreasing size and were stacked on the tower in decreasing order of size from bottom to top.
* Next to this (tower 1) are two other diamond towers (tower 2 and 3). At the time of creation, have been attempted to move disks from tower 1 to tower 2 using tower 3 for intermediate stage.
* only one disk can be moved at a time.

ii) Tower of Hanoi

Problem Description:

* we are given 'n' disks and three towers.
* The disks are initially stacked on tower 1 in decreasing order of size from bottom to top.
* we are to move the disks to tower 2, one disk at a time, such that no disks is ever on top of a smaller one.



Solution:

* To get the largest disk to the bottom of tower 2, we move the remaining 'n-1' disks to tower 3 and then move the largest to tower 2.
* Now we are left with the task of moving the 'n-1' disks from tower 3 to tower 2.

P.T.O

Implementation:

Program 1: gives recursive C++ code for this solution.

```
Void towerOfHanoi (int n, int x, int y, int z)
{
  if (n>0)
  {
    towerOfHanoi (n-1, x, z, y);
    cout<< "Move top disk from tower " << x << " to top of tower"
                                    << y << endl;
    towerOfHanoi (n-1, z, y, x);
  }
}
```

Program 2: using stack

```
array stack<int> tower[3]; // 3-towers
Void moveAndShow (int, int, int, int);
Void towerOfHanoi (int n)   // n → number of disks
{
  for (int d=n; d>0; d--) // initialize
      tower[1].push(d); // add disk d to tower 1
  // move n disks from tower 1 to tower 3
  moveAndShow (n, 1, 2, 3);
}
```

Contd....

int d = tower[x].top(); // move a disk from top of stack x to top.

moveAndShow(n-1, z, y, x);

Complexity:

→ From program 1 is proportional to the number of lines of output generated, and the number of lines output is equal to the number of moves performed.

→ obtain the following recurrence for the number of moves;

$$moves(n) = \begin{cases} 0, & n=0 \\ 2 * moves(n-1) + 1, & n>0 \end{cases}$$

n → number of disks.

$O(2^n)$

→ complexity tower Hanoi is

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

Module-4 Queues

Definition:

* A queue is a linear list in which insertions (additions and puts) and deletions (removals) take place at different ends.
* The end at which new elements are added is called the back (or) rear and that from which old elements are deleted is called the front.

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

⇒ A queue with three elements shown in figure below.

* The first element we delete from the queue in figure (a) is A.
  the deletion, the configuration of figure (b) results.
* To add element D to the queue of figure (b), we must place it after element C is shown in figure (c)

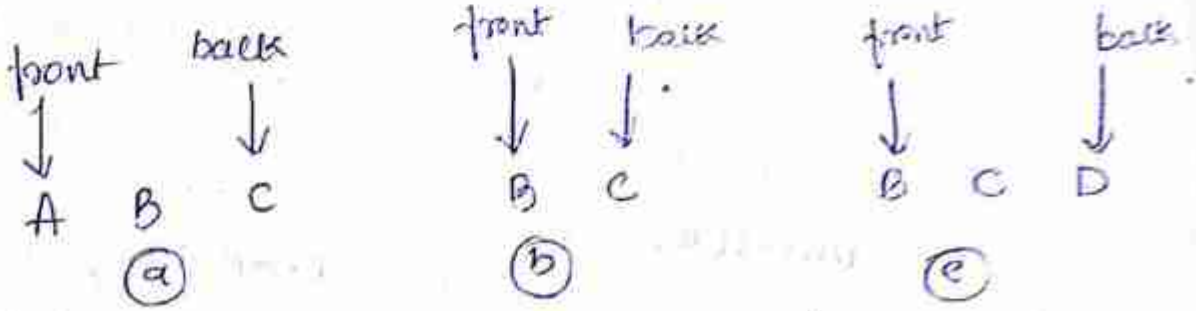


figure: queue Examples

⇒ So, queue is a FIFO list, where as a stack is LIFO

ABSTRACT DATA TYPE

→ ADT queue is specified as below.

```
AbstractDataType queue
{
    instances:
        ordered list of elements; one end is called
        the front; the other is the back.

    Operations:
        empty() : Return true if the queue is empty, return
                  false otherwise
        size(): Return the number of elements in the queue.
        front(): Return the front element of the queue.
        back(): Return the back element of the queue.
        pop(): Remove an element from the front
               of the queue.
        push(x): Add element x at the back of the
                 queue.
};
```

→ program below gives the C++ abstract class that corresponds to ADT queue.

```
template <class T>
class queue
{
  public:
    virtual ~queue() { }
    virtual bool empty() const = 0; // return true if
                                    queue is empty
    virtual int size() const = 0; // return number of
                                  elements in queue.
    virtual T& front() const = 0; // return the front
                                  element of queue.
```

```
virtual T& back() = 0; // return element at back end of queue
virtual void pop() = 0; // remove the front element
virtual void push(const T& theElement) = 0;
                // add theElement at the back of the queue
};
```

ARRAY REPRESENTATION

i) The Representation:

→ the queue elements are mapped into an array queue using the equation:

$$location(i) = i \quad \longrightarrow (1)$$

element 'i' the queue is stored in queue[i], $i \geq 0$

Let arrayLength be the length (or) capacity of the array queue and let queueFront and queueBack, respectively, be the locations of the front and back elements of the queue.

→ when equation (1) used, queueFront = 0 and the queue size is queueBack + 1.

→ An empty queue has queueBack = -1

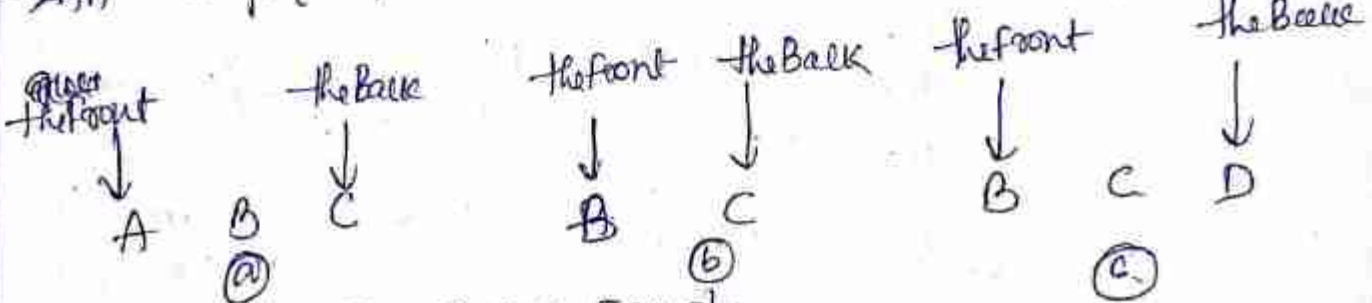
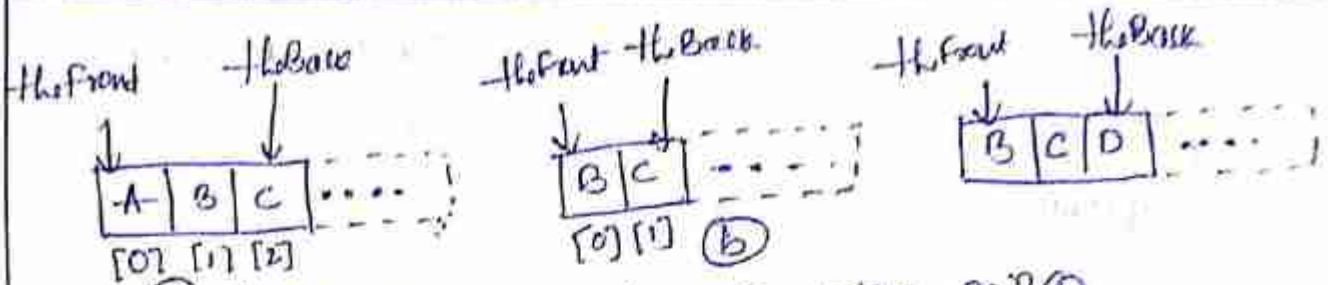


fig (1) queue samples



(a) (b) (c)

fig (2) Queues a figure (1) using eq^n (1)

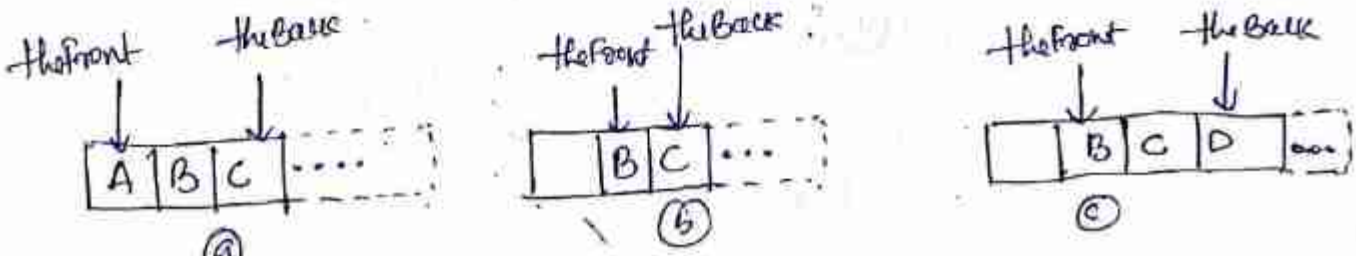
→ To push an element into a queue, we need to increase queueBack by 1 and place new element at queue[queueBack]. push operation requires O(1) time.

→ To pop an element, we must slide the elements in position 1 through queueBack one position down the array. Sliding the elements takes O(n) time where 'n' is the number of elements in the queue.

→ we can pop an element in O(1) time if we use the following equation:

location(i) = location(front element) + i ——→ (2)

→ equation (2) does not required us to shift the queue one position left each time an element is popped from the queue, we simply increase location(front element) by 1.



(a) (b) (c)

fig (3) Queue of fig (1) using equation (2)

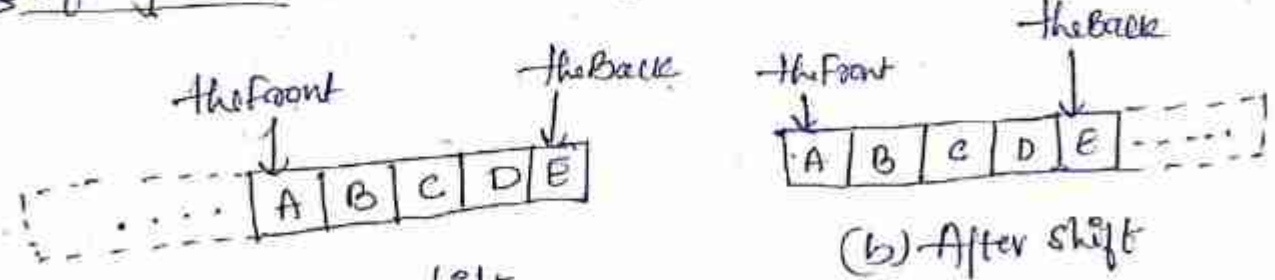
from figure (3) Notice that queueFront = location (front element), queueBack = location (last element) and an empty queue has queueBack < queueFront.

⇒ * fig (3) shows, each pop operation causes 'queueFront' to move right by 1.

* Hence there will be times when queueBack = arrayLength - 1 and queueFront > 0.

* At this time number of elements in the queue is less than 'arrayLength' and there is space for additional elements at the left end of the array.

* To continue inserting elements into the queue, we can shift all elements to the left end of the queue and create space at the right end. [worst case time for shifting O(1)] show in fig (4)



(a) Before shift (b) After shift

fig (4): shifting queue

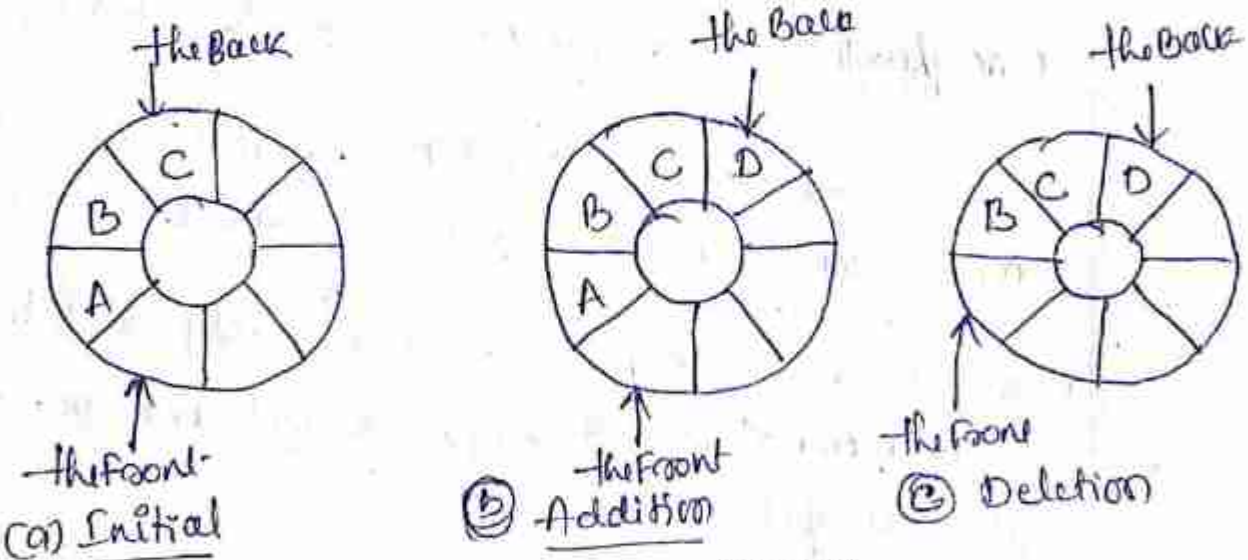
* worst case push and pop takes O(1) when we permit the queue to wrap around the end of the array.

* When the array is viewed as a circle, each array position has a next and a previous position

* The position next to position (arrayLength - 1) is 0, and the position precedes 0 is (arrayLength - 1).

* Circular array representation of a queue uses the following mapping function:

location(i) = (location(front element) + i) % arrayLength



fig(5) Circular queue.

* fig(5) the queueFront points one position counter clockwise from the location of the front element in the queue.
* queueBack is unchanged.

→ fig(5.a)

* pushing an element into the queue of fig 5.(a) results in the queue of fig 5.(b). Popping an element from the queue of fig 5.(b) results in the queue of fig 5.(c).

* when queueBack = queueFront = 0 defines an empty queue.

* we can push elements into the queue until the number of elements in the array queue equals arrayLength (i.e queue Becomes full). shown in fig ⑥

* when queueFront = queueBack, which is the same condition as when the queue is empty and full

* we cannot distiguish b/w an empty and full queue. to overcome this difficulty, we will not permit a queue to get full.

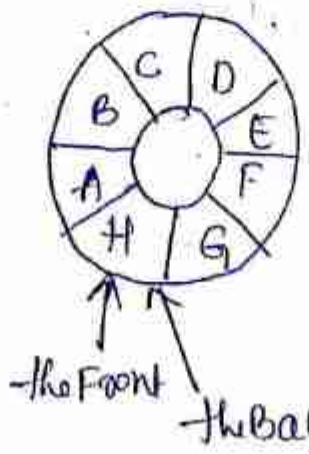


fig ⑥: Circular queue with arrayLength elements

ii) The class arrayQueue

→ The class "arrayQueue" uses the equation below to map a queue into a one-dimensional array 'queue'.

→ location(i) = [location(front element) + i] % arrayLength

→ The data members of "arrayQueue" are queueFront, queueBack, and element; the codes for all methods other than push and pop are similar to those for the corresponding methods of 'arrayStack', shown in program below;

```
template <class T>
class arrayQueue : public queue
{ protected:
        T * element;
        int queueFront;
        int queueBack;
        int arrayLength;
  public:
        arrayQueue (int initialCapacity);
        ~arrayQueue ( ){ delete [] element }
        bool empty() const;
        int size() const;
        T& front() const;
        T& back() const;
        void pop() const;
        void push (const T& theElement);
}
```

Push operation:

```
template <class T>
void arrayQueue<T>::push(const T& theElement)
{ // Add element to queue.
  if((queueBack+1)% arrayLength == queueFront)
  {
    // program ① code comes here //
  }
  queueBack = (queueBack+1) % arrayLength;
  element[queueBack] = theElement;
}
```

→ when circular queue is used, it is better to flatten out the array as fig (a). This figure shows a queue with seven elements in an array whose length is 8.

→ fig(b) shows the the flattened view of the same queue

→ fig(c) shows the array after array doubling by 'changeLength1D' function.

→ fig(d) shows that, the elements in the right segment to the right end of the array.

→ fig(e) shows an alternative configuration for the array after doubling. This configuration may be obtained as follows:

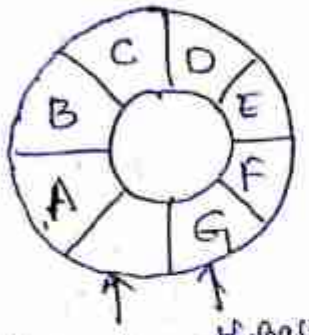
- Create a new array newQueue of twice the length
- Copy the second segment to position in newQueue

to queue[arrayLength]

beginning at 0. [i.e the elements queue[queueFront+1] to queue[rear]]
- Copy the first segment to position in newQueue beginning at arrayLength − queueFront + 1.

→ Program ① gives the configuration of fig(e).

{ use formula: location(i) = (location(frontElement)+i) % arrayLength }



fig(a) :- A full circular queue

queue [0] [1] [2] [3] [4] [5] [6] [7]

C	D	E	F	G		A	B

theFront = 5, theBack = 4

fig(b) flattened view of Circular full queue.

[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]

C	D	E	F	G		A	B								

theFront = 5, theBack = 4

fig(c) after array doubling

[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]

C	D	E	F	G										A	B

theFront = 13, theBack = 4

fig(d) : after shifting right segment by arrayLength

[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]

A	B	C	D	E	F	G									

theFront = 15, theBack = 6

fig(e) : alternative configuration.

program ① : Doubling the length of the array Queue

```
// allocate a new array.
T* newQueue = new T[2* arrayLength];
// copy elements into new array
int start = (theFront + 1)% arrayLength;
if (start < 2)  // no wrap around
    copy (element + start, (element+start+arrayLength-1), newQueue);
else
{   copy (element + start, element + arrayLength, newQueue);
    copy (element, element + theBack+1, newQueue + arrayLength - start);
}
// switch to newQueue and set theFront and theBack
theFront = 2* arrayLength -1;
theBack = arrayLength -2; // queue size arrayLength -1
arrayLength *=2;
queue = newQueue;
}
```

program (2) : popping an element from queue.

```
void pop()
{  if(queueFront == queueBack)
   {  cout << "queue is empty" << endl;
      return;
   }
   queueFront = (queueFront+1)%arrayLength;
   queue[queueFront].~T(); // destructor for T
}
```

LINKED REPRESENTATION

→ Queue like a stack can be represented as a chain we need two variables · queueFront and queueBack to keep track of the two ends of the queue.

→ There are two possibilities for binding these two variables to the two ends of a chain.

→ The nodes can be linked from front to back [fig 1(a)] (or) from back to front [fig 1 (b)].

→ The relative difficulty of performing pushes and pops determines the direction linkage.

→ fig ② and fig ③ respectively shows the mechanics of push and a pop.

→ We can see that both linkages directions are well suited for puses, but the front to back linkage, is more efficient for pops.

→ Hence we will link the nodes in a queue in a queue from front to back.

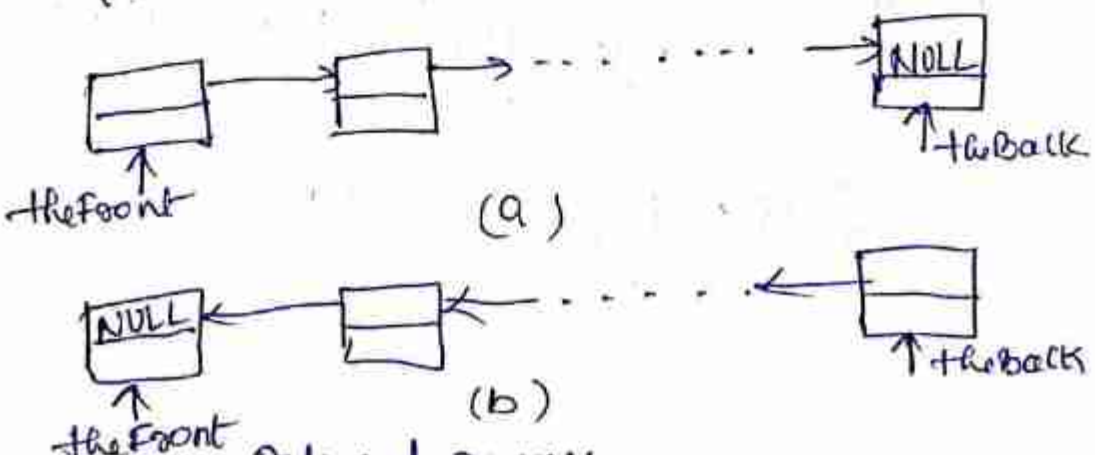
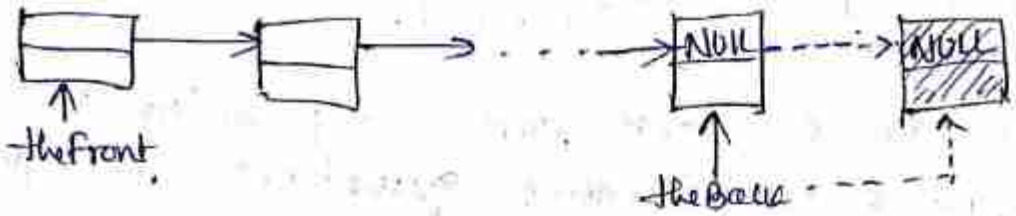
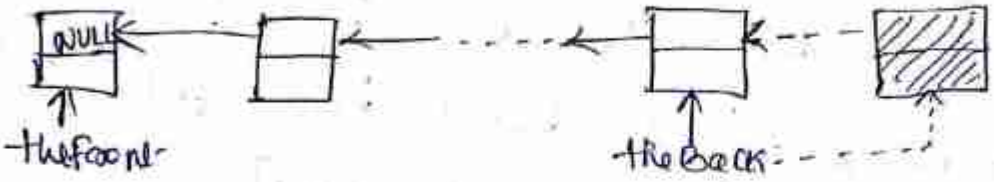


fig ①. Linked queues

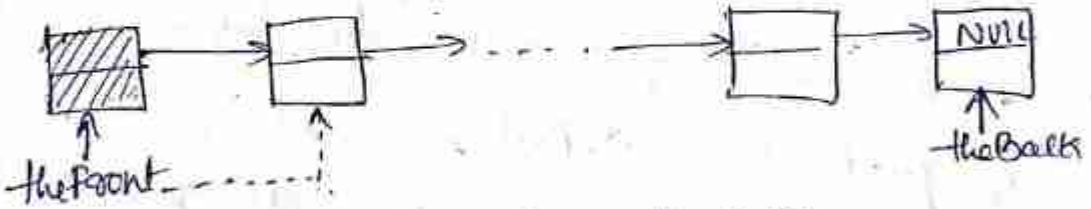


(a) Pushing into Fig ① ⑨



(b) pushing into fig 1.⑥

fig ② Pushing an element into a linked queue.



(a) popping from fig 1.(a)

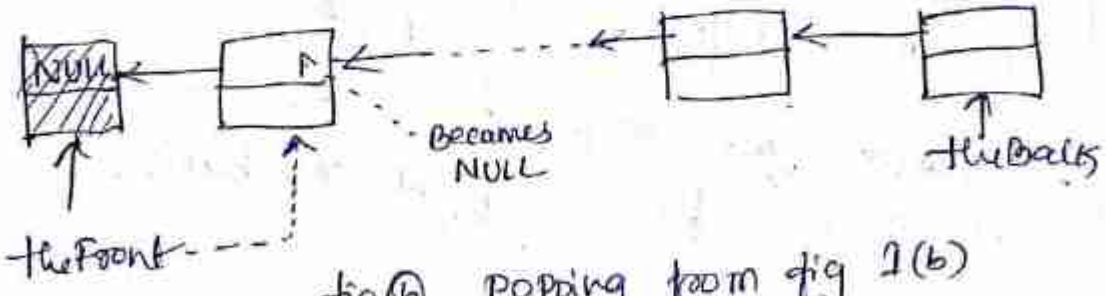


fig ⑥ popping from fig 1(b)

fig ③: popping an element from a linked queue

→ we can use the initial values queueFront = queueBack = =NULL and the boundary value queueFront = NULL iff the queue is empty.

→ The class linkedQueue may be defined as a derived class of extendedchain.

→ program below gives the push and pop operation of linkedQueue.

Program: Push & pop of linkedQueue

```
template <class T>
class linkedQueue : public extendedChain<T>, public Queue
                                    public ChainNode<T>
{ private:
    ChainNode<T> *queueFront;
    ChainNode<T> *queueBack;
    int queueSize;

  public:
    linkedQueue (int initialCapacity);
    linkedQueue();
    bool empty() const;
    int size() const;
    T& front() const;
    T& back() const;
    void pop() const;
    void push(const T& theElement);

};
```

⇒ Push operation : Add element at the queueBack

```
template <class T>
void linkedQueue<T>::push(const T& theElement)
{
  // create node for new element
chainNode<T> *newNode = new chainNode<T>;
  newNode->element = theElement;
  newNode->next = NULL;

if(queueSize == 0)
   queueFront = newNode;   // queue is empty
  else
   queueBack->next = newNode; // queue is not empty
    queueBack = newNode;
    queueSize++;
}
// Pop operation:
template <class T>
{  if(queueFront == NULL)
     { cout<<" Pop operation not possible"<<endl;
       return;
     }
 chainNode<T> *nextNode = queueFront->next
    delete queueFront;
    queueFront = nextNode;
   queueSize--;
}
```

APPLICATIONS

i) Railroad Car Rearrangement:

Problem description and Solution Strategy

→ the railroad car rearrangement problem holding tracks like blw the input and output track as shown in figure below.

→ These tracks operates in a FIFO manner and so may be regarded as queues.

→ moving a car from a holding track to the input track (or) from the output track to a holding track is forbidden.

→ All cars motion, is in the direction indicated by the arrowheads show in figure.

→ we reserve track H_k for moving cars directly from the input track to the output track. So, only $(k-1)$ tracks are available to hold cars that are not ready to be output.

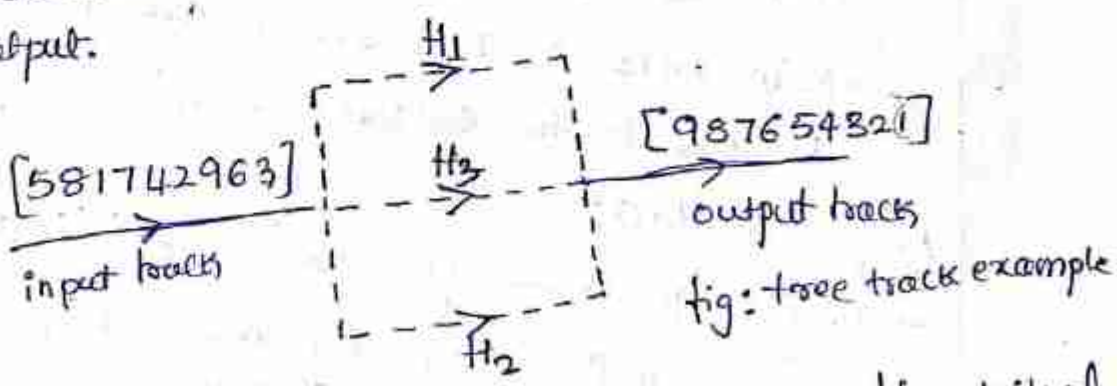


fig: three track example

→ consider rearranging nine cars that have the initial ordering 5, 8, 1, 7, 4, 2, 9, 6, 3.

→ Assume $k=3$, car 3 cannot moved directly to the output-tracks, as car 1 and car 2 must come before it.

→ So car 3 is moved to H1. car 6 can be placed behind car 3 in H1, as car 6 is to be output after car 3. Car 9 can now be placed after car 6 in H_1.

→ Car 2 cannot be placed after car 9 as car 2 is to be output before car 9. So, it is placed at the front of H2

→ car 4 can now be placed after car 2 in H_2 and car 7 can be placed after it. car 1 can be moved to the output using H_3. Next car 2 is moved from H2 to the output.

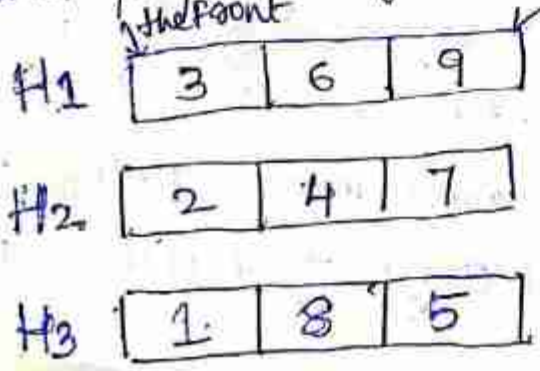
→ Then car 3 is moved from H1 to the output, and car 4 is moved from H_2 to the output. Car 5 is to be output next. It is

→ car 8 is moved from the input track to H2. Then car 5 is moved to the out put tracks.

→ Now, cars 6, 7, 8 and 9 are moved from holding tracks to the output tracks.

Implementation:

→ program ① below gives the code for 'outputFromHoldingTracks' and program ② gives the new code for 'putInHoldingTrack'.



Program:

```
arrayStack<int> *track; // array of holding tracks
int numberOfCars;
int numberOfTracks;
int smallestCar; // smallest car in any holding tracks
int itsTrack; // holding track with car smallestCar

bool railroad (int inputOrder[], int theNumberOfCars, int
                                        theNumberOfTracks)
{ // rearrange railroad cars & Return true if successful,
  numberOfCars = theNumberOfCars;
  numberOfTracks = theNumberOfTracks;
  // create queues for use as holding tracks
  tracks = new arrayStack<int> [numberOfTracks];
  int nextCarToOutput = 1;
  smallestCar = numberOfCars+1; // no car in holding tracks;
  // rearrange cars
  for( int i=1 ; i<numberOfCars ; i++)
  { if (inputOrder[i] == nextCarToOutput)
    { cout << "move car" << inputOrder[i] << "from input -
          - tracks to output track" << endl;
      nextCarToOutput++;
    // output from holding tracks
    while (smallestCar == nextCarToOutput)
    { outputFromHoldingTrack();
      nextCarToOutput++;
    }
  }
}
```

```
else
    // put car inputOrder[i] in a holding track
    if( !putInHoldingTrack(inputOrder[i])
        return false;
}
  return true;
} /* Function to output a railroad car */
void OutputFromHoldingTrack()
{ // output the smallest car from the holding track
   track[itsTrack].pop();
   cout << "Move car" << smallestCar << "from holding track" << itsTrack
                << "to output track" << endl;
   // find new smallestCar and itsTrack by checking all queue fronts
    smallestCar = numberOfCars + 2;
   for (int i = 1; i <= numberOfTracks; i++)
      if(!track[i].empty() && track[i].front() < smallestCar)
      {  smallestCar = track[i].front();
         itsTrack = i;
      }
}
/* Function to put a railroad car into a holding track */
bool putInHoldingTrack(int c) // put car c into holding track
{  int bestTrack = 0; // best track so far
   int bestLast = 0; // last car in bestTrack
  .Message
```

```
// scan tracks
for (int i=1; i<=numberOfTracks; i++)
{ if (!track[i].empty())
  { int lastCar = track[i].back();
    if (c > lastCar && lastCar > bestLast)
      { // track i has bigger car at its rear
        bestLast = lastCar;
        bestTrack = i;
      }
  }
  else // track i empty
    if (bestTrack == 0)
      bestTrack = i;
  }
  if (bestTracks == 0)
    return false; // no feasible tracks
  // add c to best track
  track[bestTracks].push(c);
  cout << "Move car" << c << "from i/p track"
       << "to holding track" << bestTrack << endl;
  // update smallest car and its track.
  if (c < smallestCar)
  { smallestCar = c;
    itsTrack = bestTrack;
  }
  return true;
}
```

DICTONARIES

It is a collection of pairs of the form (k, v), where 'k' is a key and 'v' is the value associated with the key 'k' [v is the value whose key is k]

Note: No two pairs in a dictionary have the same key.

The following operations are performed on a dictionary:

* Determine whether (or) not the dictionary is empty.
* Determine the dictionary size [i.e no of pairs].
* Find the pair with a specified key.
* Insert a pair into the dictionary.
* Delete (or) erase the pair with a specified key.

Example ①

The class list for the data structure course is a dictionary with as many pairs as students registered for the course. when a new student registers, a pair/record corresponding to this student is inserted into the dictionary; when student drops the course, his/her record may be deleted. During the course the instructor may query the dictionary to determine the record corresponding to a particular student and make changes to the record (eg: Add/change test (or) assignment scores)

* The student name be used as the **key**.
* The remaining information in a record is the value associated with the key.

The Abstract Data Type of Dictionary

→ In this specification, P.first and P.second, respectively, refer to the first and second components of the pair 'P'.

→ The first component of a pair is its key, and the second is the value.

→ When the dictionary contains no pairs with key P.first, insert(P) inserts the pair 'P' into the dictionary.

→ When the dictionary already contains a pair with key P.first, the old value associated with P.first is replaced by the new value P.second.

→ Abstract Data type (ADT) of Dictionary is shown below;

AbstractDataType Dictionary

{

instances:
collection of pairs with distinct keys.

operations:
empty(): return true iff the dictionary is empty
size(): return the number of pairs in the dictionary
find(K): return the pair with key 'K'.
insert(P): insert the pair 'P' into the dictionary
erase(K): delete the pair with key 'K'

}

Abstract class Dictionary

```
template <class K, class E>
class dictionary
{
  public:
    virtual ~dictionary() { }
    virtual bool empty() const = 0; // return true iff is empty.
    virtual int size() const = 0; // return the number of pairs in dictionary
    virtual pair<const K, E>* find (const K&) const = 0;
      // return pointer to matching pair
    virtual void erase (const K&) = 0;
      // remove matching pair
    virtual void insert (const pair<const K, E>&) = 0;
      // insert a (key, value) pair into the dictionary
};
```

LINEAR LIST REPRESENTATION

Dictionary may be maintained as an ordered linear list $(P_0, P_1, \ldots)$ where the P_i's the dictionary pairs in ascending order of key.

→ program ① gives the code for find an value of key.

→ program ② gives the code for insert the pair into dictionary

→ program ③ gives the code for delete the pair for the key in the dictionary.

program ①: The method SortedChain<K,E>::find

```
template<const K,E>
pair<const K,E>* sortedChain<K,E>::find(const K& theKey)
{ // returns pointer to matched pair
   pairNode<K,E>* currentNode = firstNode;
   // search for match with key
   while ((currentNode != NULL) &&
          (currentNode->element.first != theKey))
      currentNode = currentNode->next;
   //verify match
   if ((currentNode != NULL && current) &&
       (currentNode->element.first == theKey))
      return &currentNode->element;
      //no match
   return NULL;
}
```

program ②: The method SortedChain<K,E>::insert

```
template<class K, class E>
void sortedChain<K,E>::insert(const pair<const K,E>& thePair)
{  pairNode<K,E> *p = firstNode, *tp = NULL; // tp trails p
   while (p != NULL && p->element.first < thePair.first)
   {  tp = p;
      p = p->next;
   }
```

```
// check if there is a matching pair
if (p != NULL && p->element.first == thePair.first)
{ // replace old value
  p->element.second = thePair.second;
  return;
}
// no match, set up node for thePair
pairNode<K,E> *newNode = new pairNode<K,E>(thePair, p);
if (tp == NULL)
    firstNode = newNode;
else
    tp->next = newNode;

  dSize++;
  return;
}
```

Program ③: The method SortedChain<K,E>::erase

```
template <class K, class E>
void sortedChain<K,E>::erase(const K& theKey)
{ pairNode<K,E> *p = firstNode, *tp = NULL; //tp trails p
  // search for match with theKey
  while (p != NULL && p->element.first < theKey)
  { tp = p;
    p = p->next;
  }
```

```
//Normal branch
if (p != NULL && p->element.first == theKey)
  { //found a match & remove 'p' from chain
    if (tp == NULL)
      firstNode = p->next;
    else
      tp->next = p->next;

    delete p;
    dsize--;
  }
}
```

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

TREES

→ A Tree is a widely used abstract data type (ADT) (or) data structure implementing this ADT, that simulate a hierarchical tree structure, with a root value and subtrees of children with a parent node, represented as a set of linked nodes.

Example: Governmental Subdivision

→ Figure below shows a hierarchical drawing of the branches of the federal government. At the top of the hierarchy, we have the entire federal government.

→ At the next level of the hierarchy, we have drawn its major subdivisions.

→ Each department may be further subdivided.

→ These subdivisions are drawn at the next level of the hierarchy.

Ex: Department of Defense has been subdivided into the Army, Navy, Airforce and Marines.

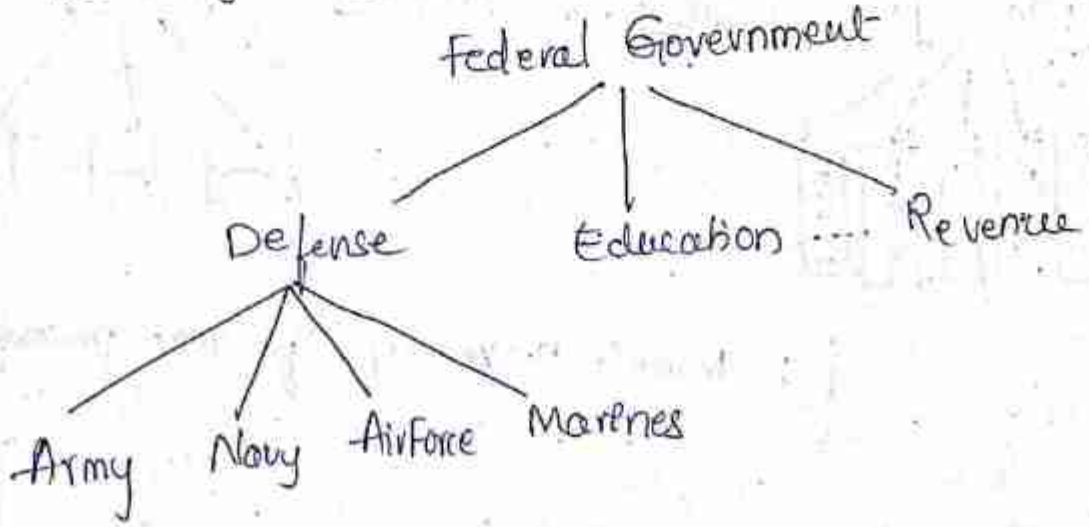


figure: Modules of the federal government

Defination: A tree is a finite nonempty set of elements. One of these elements is called the root, and the remaining elements are partitioned into trees which are called the subtrees of tree

Example 2 : Software Engineering

→ In this we decompose a large and complex task into a collection of smaller, less complex tasks

→ The objective is to divide the software system into many functionally independent parts (or) modules so that each can be developed independently.

→ The decision reduces the overall software development time, as it is much easier to solve several small problems than one large one.

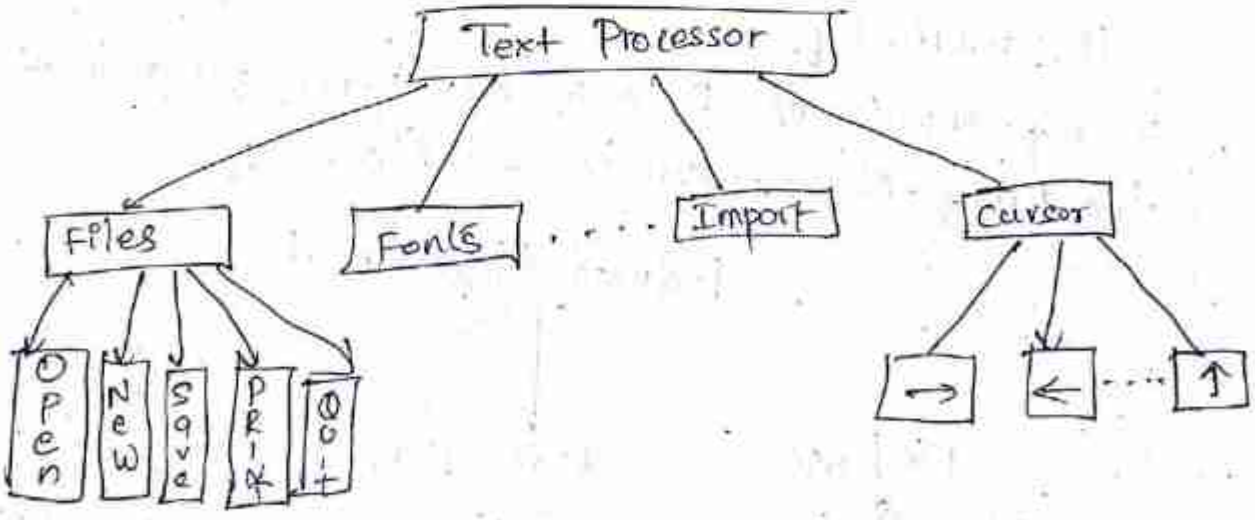


fig: Module hierarchy for text processor.

BINARY TREES

Defination: A binary tree 't' is a finite collection of elements, when the binary tree is not empty, it has a root element and the remaining elements are partitioned into two binary trees, which are called the left and right subtrees of 't'.

Differences blw Trees and Binary Trees

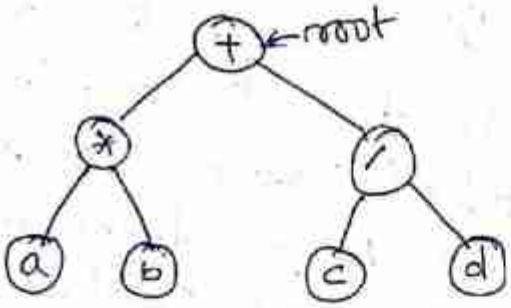
Trees	Binary Tree
* Each element in a tree can have any number of subtrees	* Each element in a binary tree has exactly two subtrees
* The subtrees in a tree are unordered	* The subtrees of each element in a binary tree are ordered.
* cannot be empty	* A binary tree can be empty

→ Binary Tree is drawn with its root at the top. The elements in the left subtree of the root are drawn below and to to left of the root.

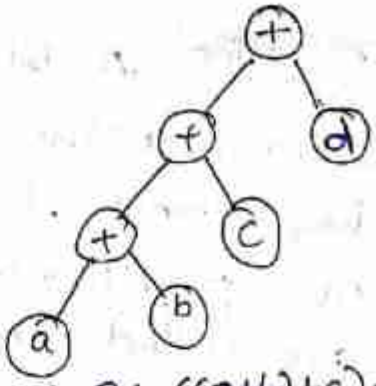
→ Between each element and its children is a line (or) edge.

→ Figure below shows some binary trees that represent arithmetic expressions. Each operator (+, -, *, /) may have one (or) two operands. The left operand is the left subtree of the operator. The right operand is its right subtree. The leaf elements in an expression tree are either constant (or) variable. [expression tree contains no parentheses]

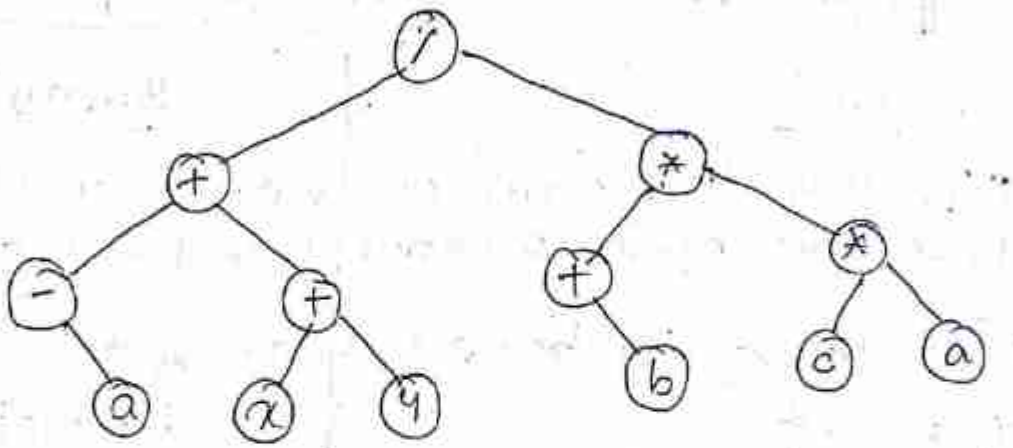
Examples



fig(a): (a*b)+(c/d)



fig(b): ((a+b)+c)+d



fig(c): ((-a)+(x+y))/((+b)*(c*a))

figure: Expression tree.

properties of Binary Trees

property(1): The degree of every binary tree with 'n' elements, n>0, has exactly n-1 edges.

proof: Every element in a binary tree has exactly one parent. There is exactly one edge b/w each child and its parent. So the number of edges is n-1

property(2): A binary tree of height h, h≥0, has at least 'h' and at most 2^h-1 elements in it

Proof:

Since each level has at least one element, the number of elements is at least h. Each element can have at most two children, the number of elements at level 'i' at most 2^{i-1}, $i>0$,

* For $h=0$ (or) $i=0$, total number of elements is 0, which equals $2^0-1=0$
* For $h>0$, the number of elements cannot exceed

$$\sum_{i=1}^{h} 2^{i-1} = 2^h - 1$$

property ③: The height of a binary tree that contains n, $n \geq 0$, elements is at most n at least $\lceil \log_2(n+1) \rceil$

Proof: there must be at least one element at each level, the height cannot exceed n. A binary tree of height 'h' can have no more than $2^h - 1$ elements

Hence $h \geq \log_2(n+1)$. Since 'h' is an integer we get $h \geq \log_2(n+1)$.

→ A binary tree of height 'h' that contains exactly $2^h - 1$ elements is called a full binary Tree (or) perfect binary Tree

Ex:

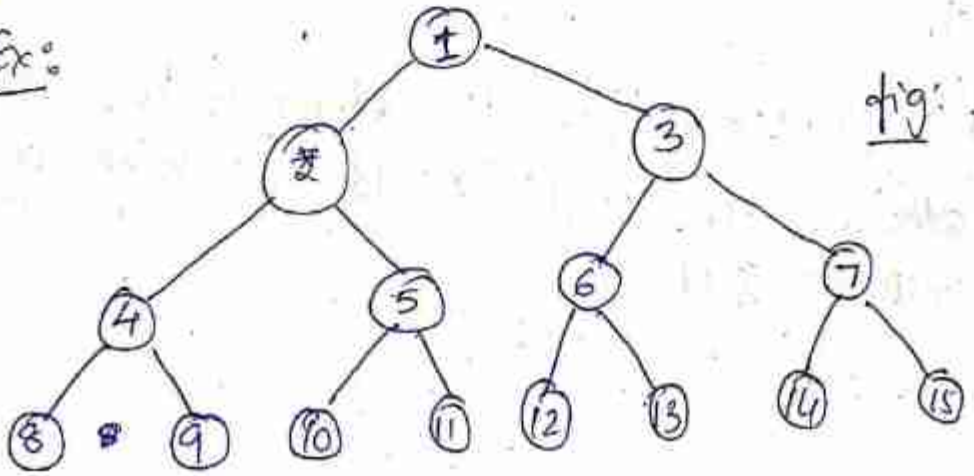


fig: Full binary Tree

→ If we delete the K elements numbered $2^h - i$, $1 \le i \le K < 2^h$. The resulting binary tree is called as Complete binary Tree.

→ A complete binary Tree: All levels except possibly the last are completely filled and all nodes are as left as possible

ex:

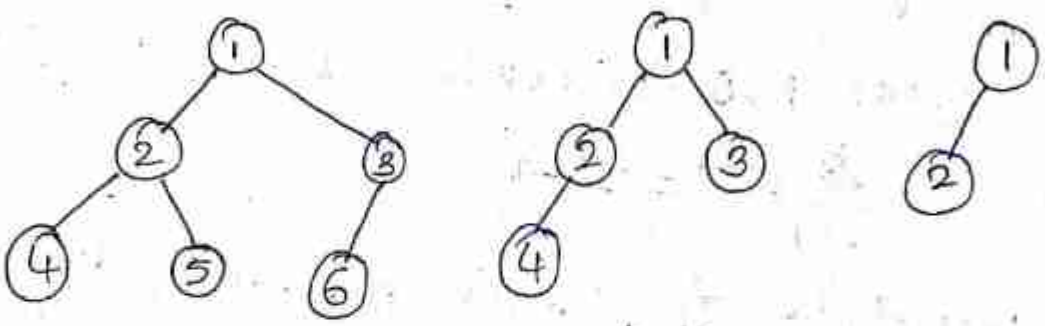


fig: Complete binary tree.

property (4): Let i, $1 \le i \le n$, be the number assigned to an element of a complete binary Tree. The following are true:

i) if $i = 1$, then this element is the root of the binary tree. if $i > 1$, then the parent of this element has been assigned the number $(i/2)$.

ii) if $2i > n$, then this element has no left child. otherwise, its left child has been assigned the number $2i$.

iii) if $2i+1 > n$, then this element has no right child. otherwise, its right child has been assigned the number $2i+1$,

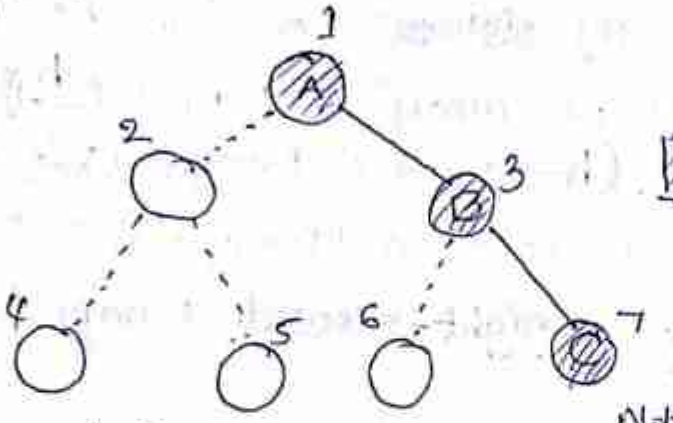
REPRESENTATION OF BINARY TREES

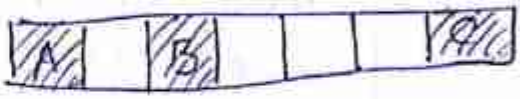
1) Array-Based Representation

→ As per the property number (4), the binary tree to be represented is regarded as a complete binary tree with some missing elements

→ Figure① below shows binary tree with three elements

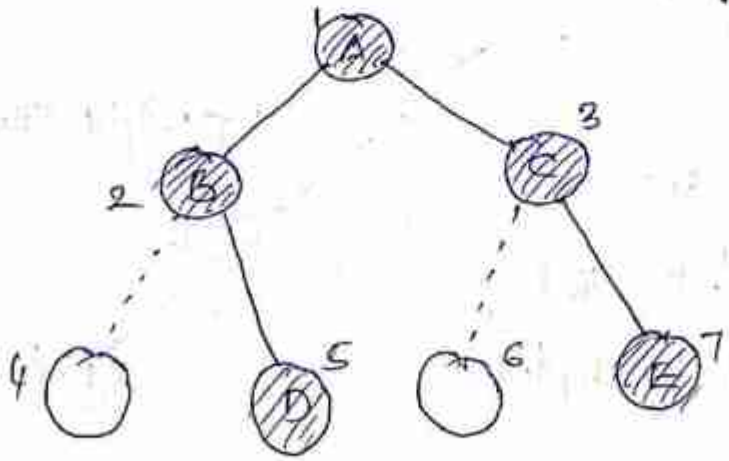
→ Figure② below shows binary tree with five elements

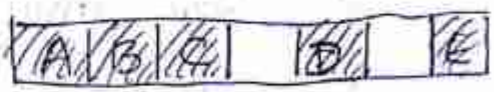




fig①

Note: unshaded circles represents missing elements





fig②

fig: Incomplete binary Trees

→ In an array representation, the binary tree is represented in an array by storing each element at the array position corresponding to the number assigned to it.

→ In figure ① & ② the missing elements are represented by white boxes.

→ As can be seen, this representation scheme is quite wasteful of space when many elements are missing.

→ In fact, a binary ~~element~~ tree that has 'n' elements may requires an array size up to $(2^h - 1)$ for its representation. (h → height of binary tree from root Node)

→ figure ③ below shows a binary tree with four elements called right-skewed binary tree.

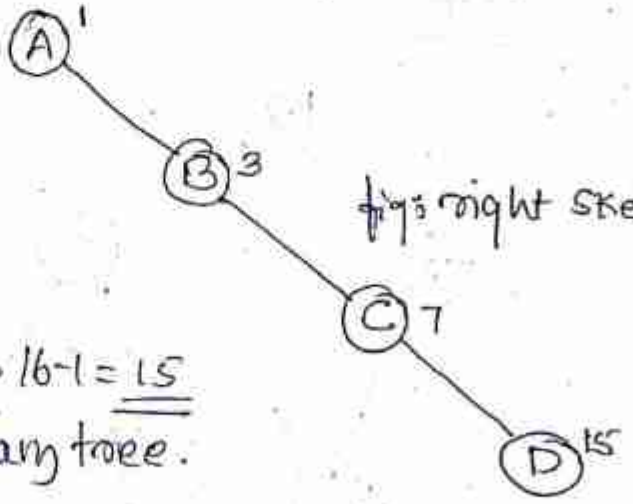


fig: right skewed binary tree

Size of array is

$= 2^h - 1 = 2^4 - 1 \Rightarrow 16 - 1 = 15$

h → height of binary tree.

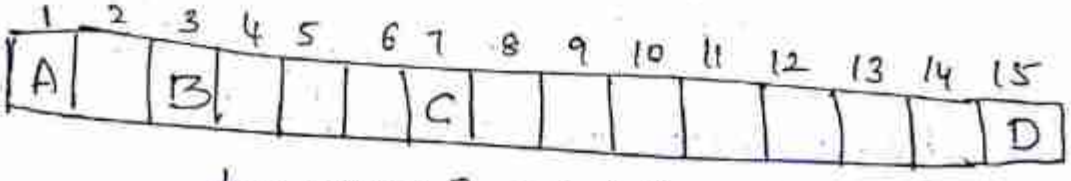


fig: array Representation

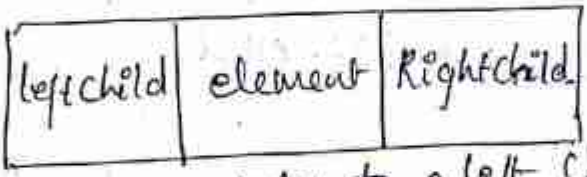
fig ③ : Right skewed binary Tree

② Linked representation:

→ The most popular way to represent a binary tree is by using doubly linked list.

→ Each node of doubly linked list consists of 3-fields
* 'left link' is a pointer point to left child Node.
* Right link is a pointer point to Right child Node
* Data field hold the value of node.

→ Below diagram shows Node representation using doubly linked list



Node

leftchild → is a pointer to a left child of parent Node.
Rightchild → is a pointer to a Right child Node.
element → is a datafield holds the data.

→ If no childrens to a node, is called **Leaf Node**, the leaf Node's leftchild = Rightchild = NULL, means no child nodes.

→ n-element binary tree has exactly n-1 edges, we are left with 2n-(n-1) = n+1 pointers fields that have no value. These pointer are set to NULL

→ we can access all nodes in a binary tree by starting at the root and following leftchild and rightchild pointers.

program: Node Struct for linked binary trees

```
template <class T>
struct binaryTreeNode
{  T element;
   binaryTreeNode<T> *leftChild; // left subtree
   binaryTreeNode<T> *RightChild; // right subtree

   binaryTreeNode()
   {  leftChild = RightChild = NULL;
   }

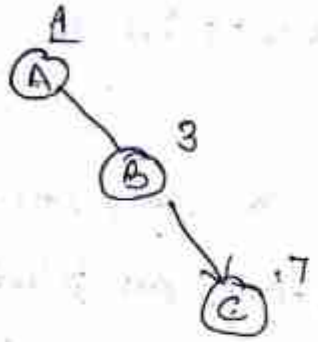
   binaryTreeNode (const T& theElement)
   {  element = theElement;
      leftChild = RightChild = NULL;
   }
   binaryTreeNode(const T& theElement, binaryTreeNode *L, binaryTreeNode
                                                          *R)
   {  element = theElement;
      leftChild = L;
      RightChild = R;
   }
};
```

→ Each pointer from a parent node to a child node represents an edge in the drawing of a binary tree

→ Since an n-element binary tree has exactly $n-1$ edges, we left with $n+1$ pointer fields that have no value.

→ figure below shows the linked representation of the binary tree.

Ex: 1)



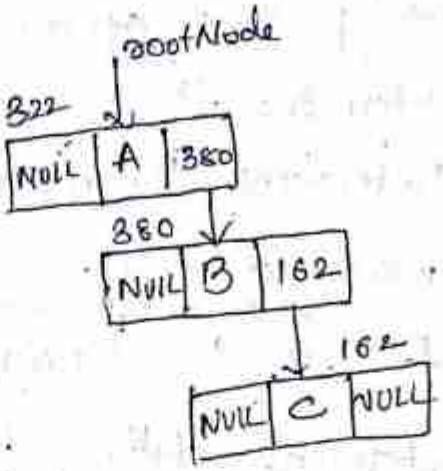
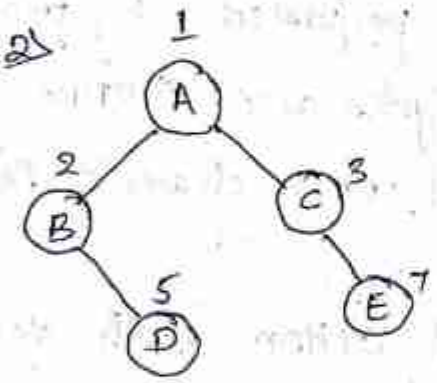


fig (1):

Ex: 2)



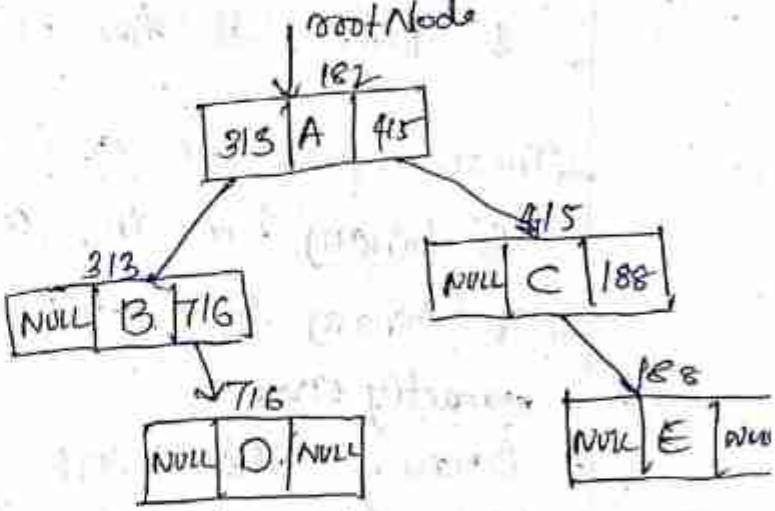


fig (2)

→ fig (1) & fig (2) shows linked representation of binary Trees.

P.T.O

COMMON BINARY TREE OPERATION

→ Commonly used operations on binary Trees are:

* Determine its height.
* Determine the number of elements in it.
* Make a copy.
* Display the binary tree on a screen (or) on paper
* Determine whether two binary trees are identical
* Make the tree empty.

→ These operations can be performed by traversing the binary tree in a systematic manner.

→ In binary tree traversal, each element is visited exactly once.

→ During this visit all action with respect to this element is taken. Such as writing elements on a screen, evaluating the expressions and adding one to a running count of the number of elements in the binary tree.

BINARY TREE TRAVERSAL

There are four ways to traverse a binary tree:

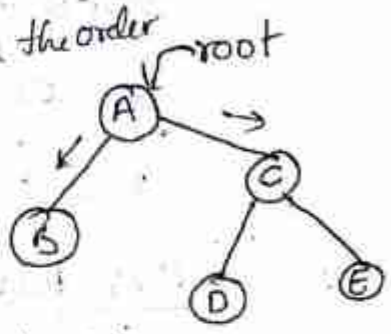
1) preorder [<root> <left> <right>]
2) Inorder [<left> <root> <right>]
3) Post Order [<left> <right> <root>]
4) Level Order.

1> Preorder Traversal

→ In preOrder Traversal, first we visit root node and display the ~~data~~ data of root node, then visit left subtree and then visit right subtree.

Note: Preorder Traversal flow as follows in the order

* visit root Node / display data
* visit left sub-tree
* visit Right sub-tree



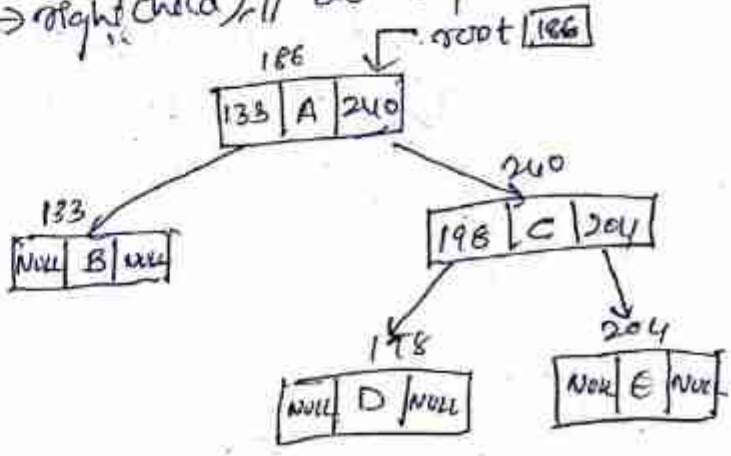
Eg. fig ①

Eg: from figure ① using preorder Traversal

⇒ A, B, C, D, E

Program:

```
template <class T>
void preOrder (binaryTreeNode<T> *root)
{  if (root != NULL)
   {  cout << root->element << endl; // visit tree root
      preOrder (root->leftChild); // do left subtree
      preOrder (root->rightChild); // do right subtree
   }
}
```

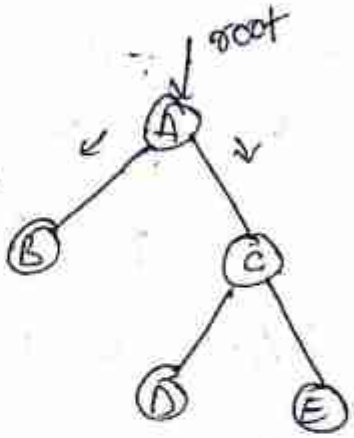


2> Inorder Traversal

->In Inorder Traversal flow as follows in the following order:

* Visit left subtree
* Visit root (or) display data
* Visit right subtree

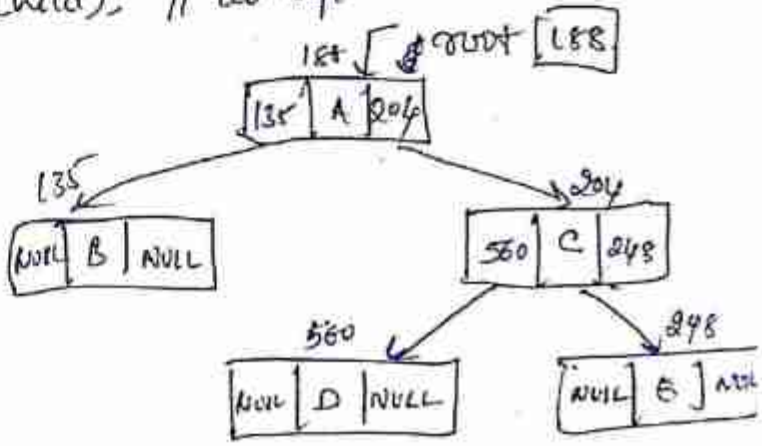
Ex: from figure (a) Tree using Inorder Traversal result is



⇒ B, A, D, C, E

program:

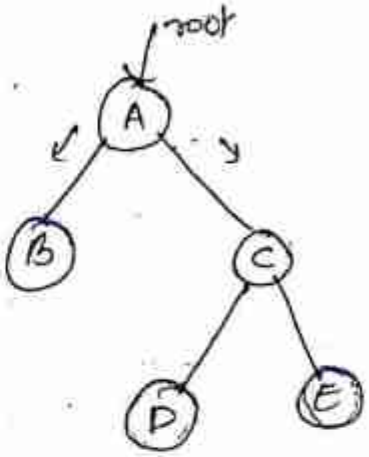
```
template<class T>
Void Inorder(binaryTreeNode<T> *root)
{
  if(root!=NULL)
  {
  Inorder(root->leftchild);   // do left subtree
   cout<<"root->element<<endl; // visit root
   Inorder(root->RightChild); // do left subtree
  }
}
```



3> Post Order Traversal

→ In Postorder Traversal flow as follows in the following order

* Visit left subtree
* Visit Right subtree
* Visit root / display data.

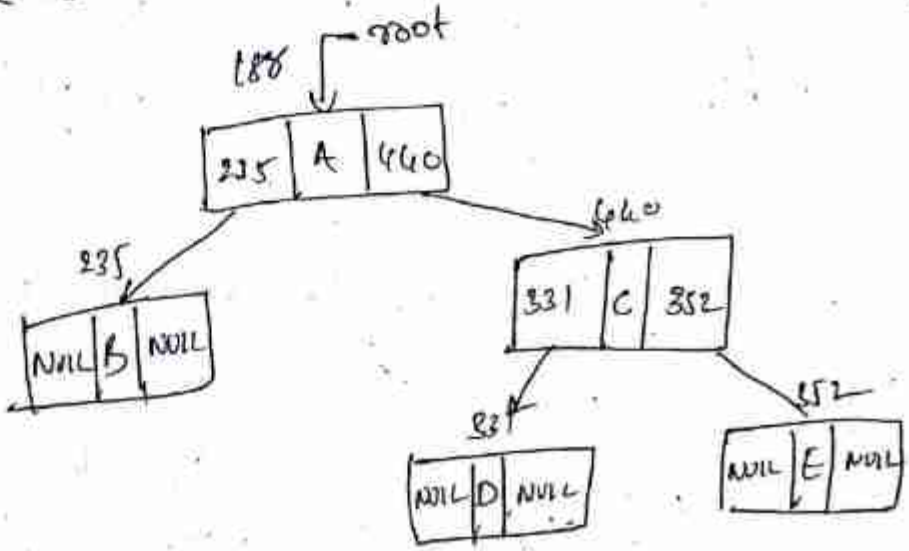


Ex: from figure (A) Tree using postorder Traversal the result is

⇒ B, D, E, C, A

Program:

```
templat <class T>
Void postorder(binaryTreeNode<T> *root)
{ if(root!=NULL)
  {
    postorder(root->leftchild); // visit left subtree
    postorder(root->rightchild); // visit Right subtree
    Cout << root->element << endl; // Visit root
  }
}
```



Level-order Traversal

→ The elements are visited by level from top to bottom.
→ within levels, elements are visited from left to right.
→ It is difficult to write a recursive function for level-order traversal, as the correct data structure to use here is a queue.

→ program below shows the level-order Traversal:
* In this, if tree is not empty, root is visited and its children, if any, are added its address to the queue.
* we attempt to access the front element of the queue.
* when queue is empty it will throw exception.
* if queue is not empty, the front address of child will be next that is to be visited.

program: level-order.

```
template <class T>
void levelOrder (binaryTree<T> *root)
{
    arrayQueue<binaryTree<T>*> q;
    while (root != NULL)
    {
        cout << root->element << endl;
        // put root's children on queue
        if (root->leftChild != NULL)
            q.push (root->leftChild);
```

```
    if (root -> rightChild != NULL)
      q.push (root -> rightChild)

  // get next node to visit

    root = q.front();
    q.pop();
   }
}
```

THE ADT BinaryTree

ADT of binaryTree is shown below, the number of operations we may wish to perform on a binary tree is quit large, we list only some of the commonly performed ones.

```
AbstractDataType binaryTree
{ instances: collection of elements, partitioned into a root,
             left subtree and right subtree; each subtree is
             also a binaryTree.

  operations:
    empty(): return 'true' if empty, return false otherwise
    size(): return number of elements in the tree.
  preOrder(visit): preOrder traversal of binary tree, visit is
                   the visit function to use.
  inOrder(visit): inorder traversal of binary tree;
  postOrder(visit): post order traversal of binary tree;
  levelOrder(visit): level-order traversal of binary tree;
};
```

THE CLASS linkedBinaryTree

→ class linkedBinaryTree derives from the abstract class binaryTree and uses Nodes of the type binaryTreeNode.

→ Abstract class binaryTree

```
template <class T>
class binaryTree : public
{
  public:
    virtual ~binaryTree() {};
    virtual bool empty() const=0;
    virtual int size() const =0
    virtual void preorder (binaryTreeNode<T> *root)=0;
    virtual void inorder (binaryTreeNode<T> *root)=0;
    virtual void postorder (binaryTreeNode<T> *root)=0;
    virtual void levelOrder (binaryTreeNode<T> *root)=0;
};
```

program: Data members and visit Method.

```
template<class T>
class linkedBinaryTree : public binaryTree<T>
{ private:
      binaryTreeNode<T> *root; // pointer to root
      int treeSize;
      ~~bool~~ *
  public:
      linkedBinaryTree() // constructor
      { root = NULL;
        treeSize = 0;
      }
```

```
~linkedBinaryTree() {  // destructor.
  { delete root;
  }
bool empty() const { return treeSize == 0; }
int size() const { return treeSize; }
void preOrder (binaryTreeNode<T> *root);
void Inorder ( binaryTreeNode<T> *root);
void postOrder ( binaryTreeNode<T> *root);
void levelOrder ( binaryTreeNode<T> *root);
int height (binaryTreeNode<T> *root) const;
};
```

preOrder Method:

```
template <class T>
void linkedBinaryTree<T>::preOrder (binaryTreeNode<T>
                                    *root)
{ if (root != NULL)
  {
    cout << root->element << endl;
    preOrder (root->leftChild);
    preOrder (root->RightChild);
  }
}
```

P.T.O

Program: Determining the height of a binary Tree

```
template <class T>
int linkedBinaryTree<T> :: height (binaryTreeNode<T> *root)
{
   if (root == NULL)
     { cout << "Tree is empty" << endl;
       return 0;
     }
   int hl = height(root -> leftChild); // height of left subtree
   int hr = height (root -> rightChild); //height of right subtree

   if (hl > hr)
     return ++hl;
   else
     return ++hr;
}
```

ManojKumar S B
Assistant Professor
Dept. of ECE
BGS Institute of Technology
B G Nagara-571448

Module-5 Priority Queues

PRIORITY QUEUES → is a Abstract data Type.

→ priority Queue is a collection of elements, Each Element has a priority (or) value.

→ The operations performed on a priority queue are
- find an element → top operation.
- insert a new element → push operation.
- remove an element. → pop operation.

→ Element of higher priority should be processed before an element of lower priority.

~~→ Same pro~~

→ If two elements are same priority it will be process on FIFS. [First in first Serve]

→ The insert operation; insert an item with given priority

→ The find operation; returns the highest priority item.

→ The remove operation; remove the highest priority item.

Implementation using array:

→ Here first we allocate memory of some size. dynamically. Then we follow following operations:

- Insert : operation: can be implemented by adding an item at end of array in O(1) time.

There are two types of priority Queues:

i) Min priority Queue [Ascending order Priority Queue]
the find operation finds the element with minimum priority (Low priority value having highest priority), and the remove operation removes this element.

ii) Max priority Queue [Descending order Priority Queue]
the find operation finds the element with maximum priority (highest priority value having first priority), and the remove operation removes this element.

Implementation using LINEAR LIST.

→ Simple way to represent a max priority Queue is as an unordered linear list with n elements.

→ new elements are easily inserted at the right end of this list.

→ Hence the insert (or) push operation takes O(1) time.

→ A pop operation requires a search for the element with largest priority followed by the removal of this element takes O(n) time.

if linked list (or) chain is used, push's can be performed at the front of the chain O(1) time and pop takes O(n) time.

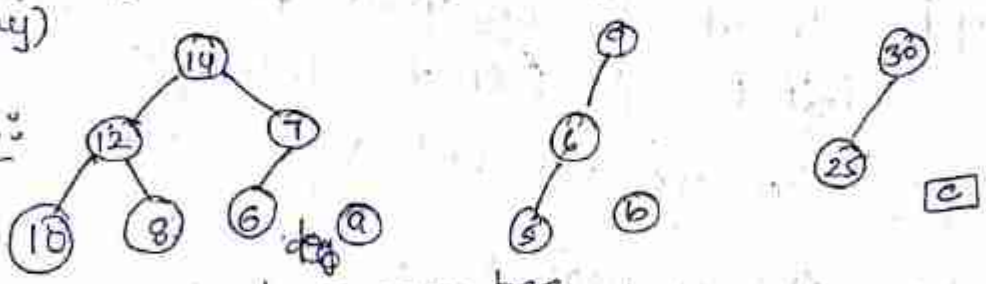
⇒ An alternative is to use an ordered linear list.
* the elements are in nondecreasing order and in nonincreasing order when we use an ordered chain.
* The pop time for each representation is O(1) and the push time is O(n).

HEAPS

Definitions: A heap is a specific tree based data structure in which all the nodes of tree are in a specific order.

max tree: is a tree in which the value in each node is greater than (or) equal to those in its children (if any)

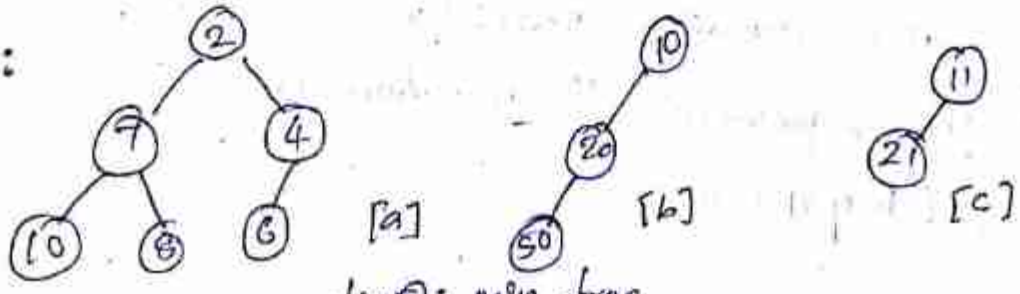
Ex:



fig①: max tree

Min tree: is a tree in which the value in each node less than (or) equal to those in its children (if any)

Ex:



fig②: min tree

Max Heap: is a max tree that is also a complete binary tree.

Min Heap: is a min tree that is also a complete binary tree.

→ Example for max heap is fig ①.a and fig ① ⓒ but fig ①.b is not a complete binary tree.

→ Example for min heap is fig ②.a and fig ②.ⓒ but fig ②.b is not a complete binary tree.

Insertion into a Max Heap

Insertion operation in max heap is performed as follows

Step 1: Insert the new node as last leaf from left to right (in the incomplete level)

Step 2: Compare new node value with its parent node

Step 3: If new node value is greater than its parent, the swap both of them.

Step 4: Repeat Step-2 & step-3 until new node value is less than its parent node (or) new node reached to root.

* To implement the insertion into a Max heap takes $O(\log n)$ time.

Example:

S1: Insert new node value 5 into the following max heap from left to right of incomplete level.

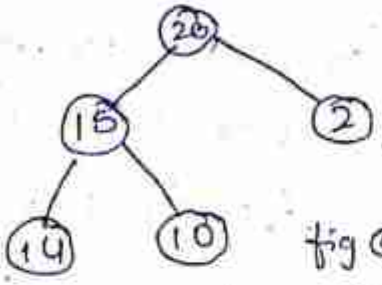


fig (a)

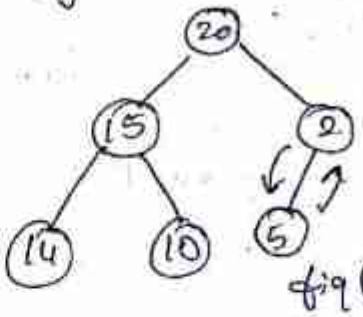


fig (b)

* new node value (5) is taken as left child to node value (2) shown in fig (b)

S2: Compare new node value (5) with its parent value (2).

S3: it is greater than its parent then swap it shown in fig (b) and fig (c)

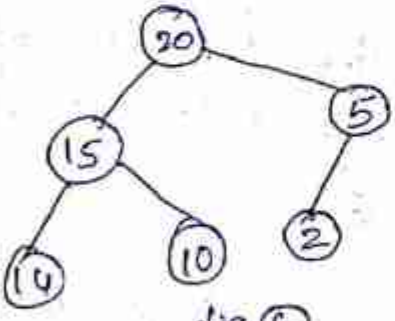


fig (c)

* again Compare (15) node value (5) with its parent (20) it is less than its parent, no swap takes place hence it holds the max heap property.

Deletion from a max heap

→ ~~In a max heap deleting last node is very simple~~

→ When an element is to be removed from a max heap, it is taken from the root node of the heap.

→ In a max heap, deleting last node is very simple as it not disturbing max heap properties.

→ We use following steps to delete root node from a max heap.

Step1: Swap the root node with last node in max heap.

Step2: delete last node.

Step3: Now, compare root value with its left child value.

Step4: if root value is smaller than its left child, then compare left child with its right sibling, else goto step 6

Step5: If left child value is larger than its right sibiling, then swap root with left child. otherwise swap root with is right child.

Step 6: If root value is larger than its left child then compare root value with its right child value.

Step 7: If root value is smaller than its right child, then swap root with right child. otherwise stop the process.

Step 8: Repeat the same until root node is fixed at its exact position.

Example:

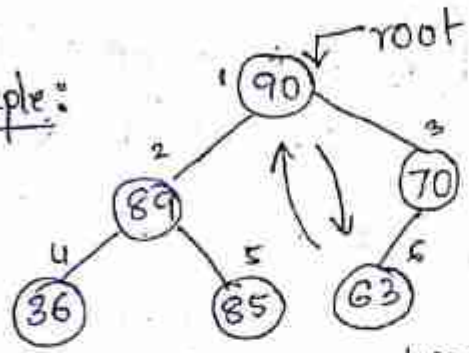


fig (a) max heap

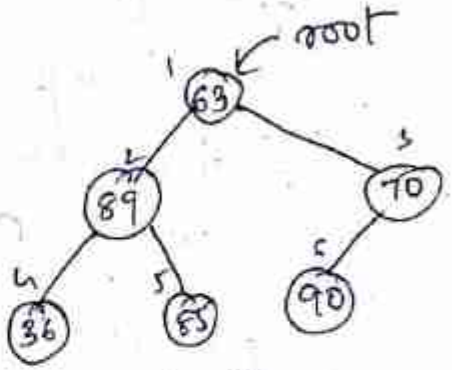


fig (b)

Step 1: Swap the root node (90) with last node (63) shown in fig (b)

Step 2: Delete last node (90). after deleting last node (90) max heap shown in fig (c)

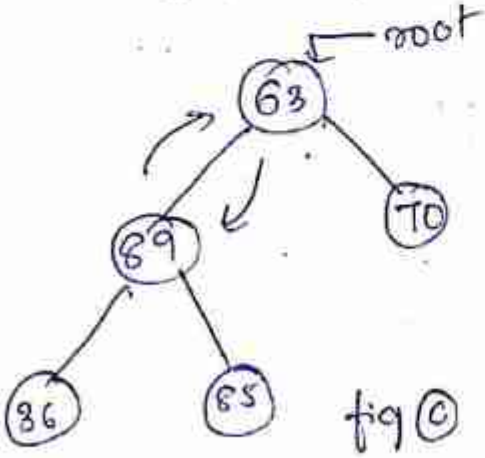


fig (c)

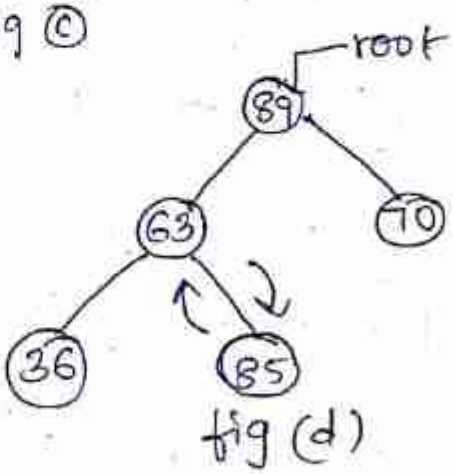


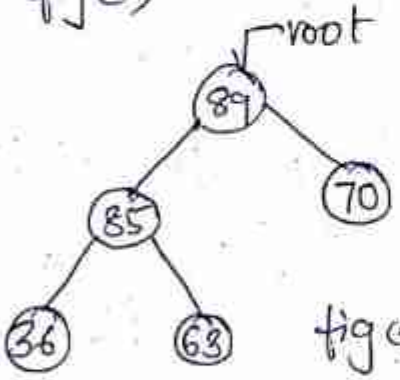
fig (d)

Step 3: Compare root node (63) with its left child (89) Here root value (63) is smaller than its left child (89), So, compare left child (89) with its right sibling (70)

Step 4: Here, left child value(89) is larger than its right sibling(70). So, swap root (63) with left child (89) shown in fig(d).

Step 5: Now compare with value (63) with its left child(36). Here with value 63 is larger than its left child, So, we compare node with value 63 with its right child 85.

Step 6: Here node with 63 is smaller than its right child (85). So, swap both of them shown in fig(e)



fig(e)

Step 7: Stop the process node value 63 reaches the position leaf node (no child nodes for this).

* To perform deleting from a max heap takes O(log n) time.

Max heap Initialization

→ We begin with a heap that contains $n > 0$ elements. We can construct this initial nonempty heap by performing 'n' inserts into an initially heap.

→ The total time taken by these 'n' insert is $O(n \log n)$.

* Assume that $n = 10$ and the priority of the elements in $a[1:10]$ is $[20, 12, 35, 15, 10, 80, 30, 17, 2, 1]$
* this array is represented as a complete binary tree as shown in fig (a) below. but it is not a max tree

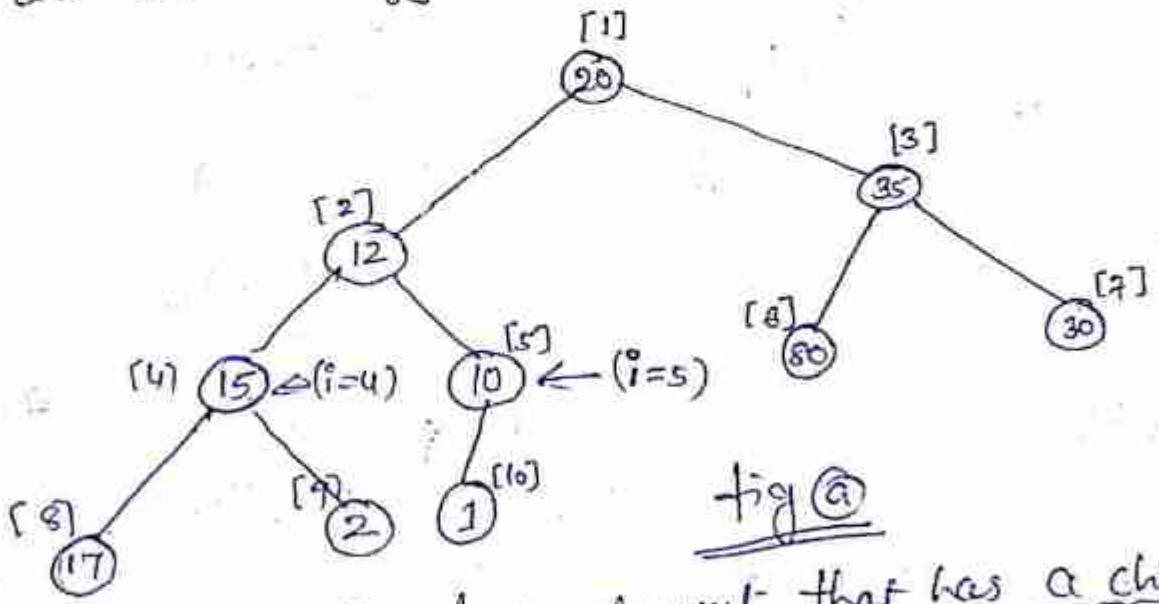


fig (a)

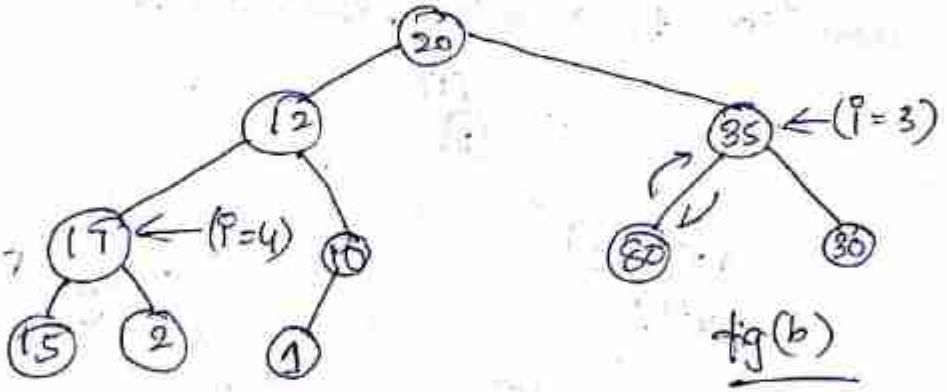
* Begin with the last element that has a child. This element is at position $i = (n/2)$ of the array.
* If the subtree that is rooted at this position is a max heap, then no work is done here.
* If this subtree is not a max heap, then we adjust the subtree so that it is a heap.

* Following this adjustment, we examine the subtree whose root is at i-1, then i-2, and so on until we have examined the root of the entire binary tree. which is at position 1.

Example: fig(a) shows the ~~binary~~ complete binary tree for array of 10 elements with priority,

i.e a[1:10] = [20, 12, 35, 15, 10, 80, 30, 17, 2, 1]

* Initially i=5, The subtree with root at 'i' is a max heap, is 10>1. then no swap [Sub tree is max heap]



fig(b)

* Then i=4, the subtree rooted at position 4. This subtree is not max heap, as 15<17. To convert this subtree into max heap, the 15 and 17 are interchanged [shown in fig (b)]
* next subtree examined has its root at position 3 (i=3) to make this subtree into a max heap, we interchange we interchange the 80 and 35. [shown in fig (c)]

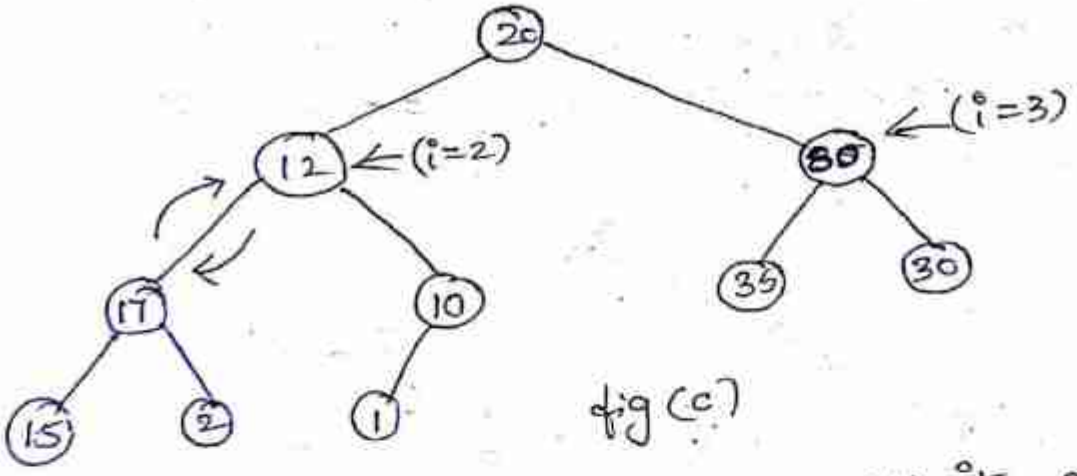
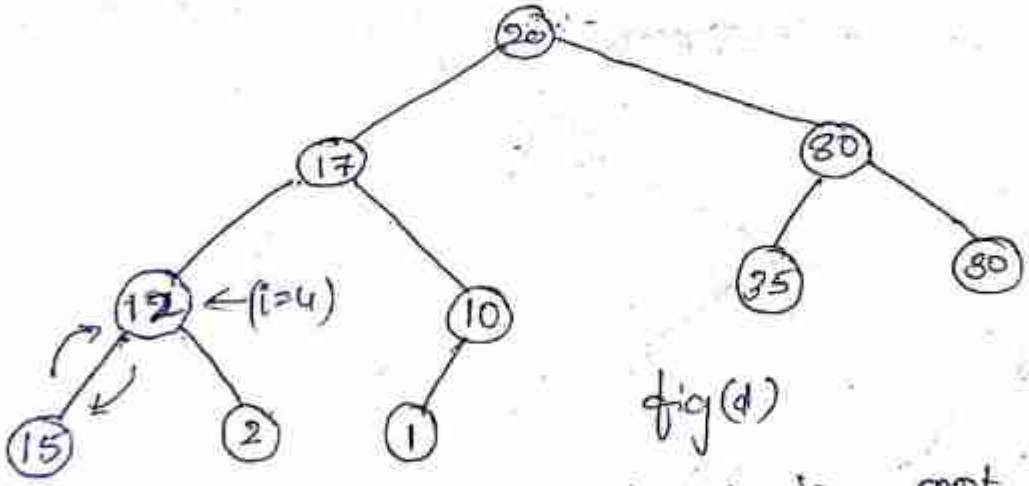


fig (c)

* Next we examine the subtree with its root at position 2. (i=2). where 12 < 17, exchange 15 and 17. shown in fig(d). ~~now the subtree~~ now heap.



fig(d)

* The left subtree of node 17 is root max heap. Compare 12 & 15 (12 < 15) and exchange 12 & 15 shown in fig (e). Now 15 is moved to position 4

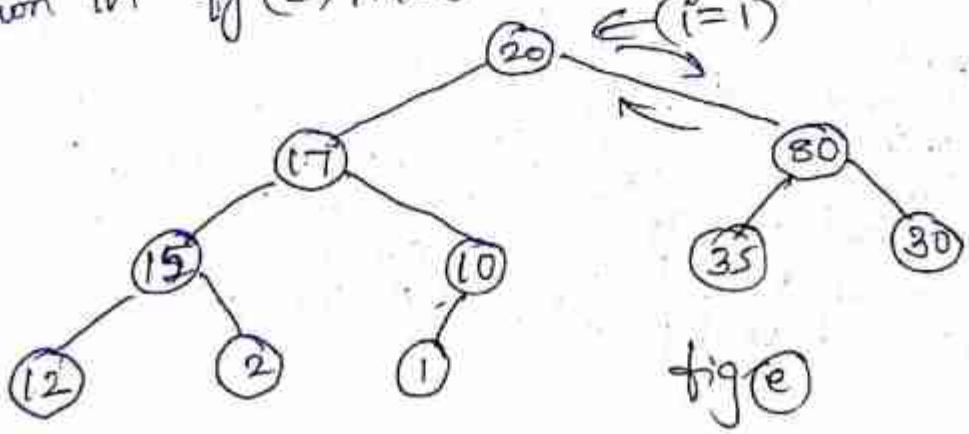
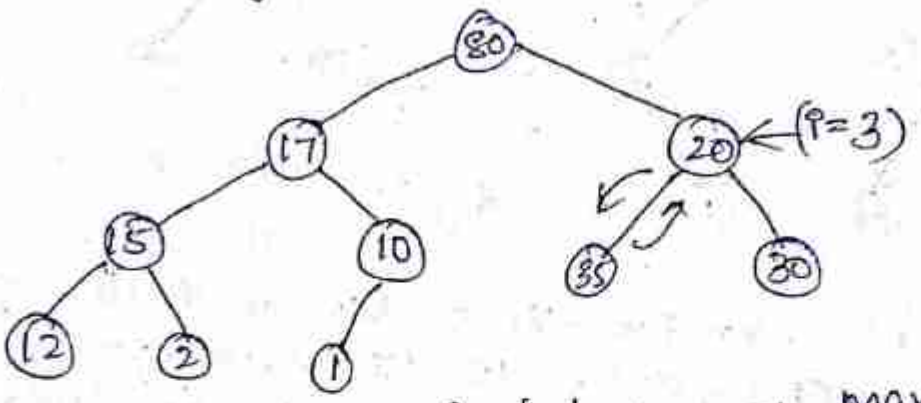
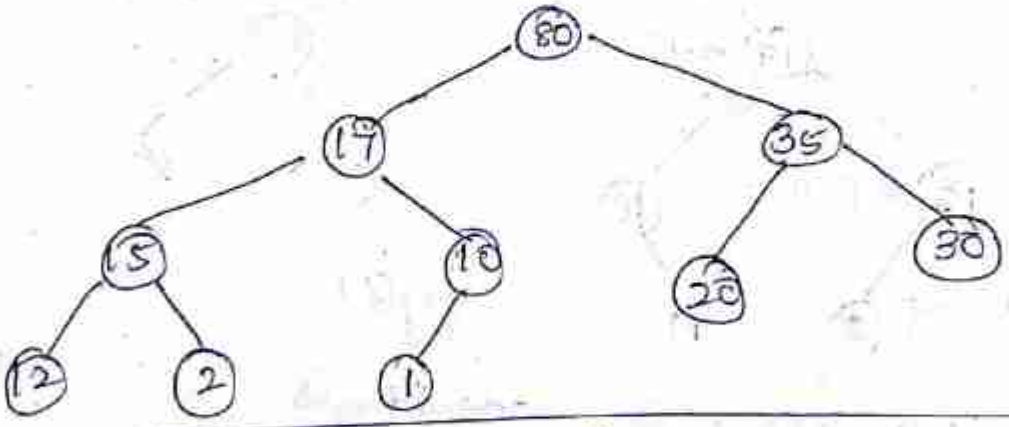


fig (e)

*Finally we examine position 1. The subtree with roots at position 2 and 3 are max heaps at this time when $20 < \max\{17, 80\}$. 80, 80 and 20 are exchanged shown in fig(f).



* But left subtree of 80 is not max heap. Where $20 < \max\{35, 30\}$. Exchange 20 & 35 shown below. Now complete tree is max heap.



Abstract class of max priority Queue

```
template <class T>
class maxPriorityQueue
{ public:
    virtual ~maxPriorityQueue();
    virtual bool empty() const = 0 // returns true iff queue is empty
    virtual int size() const = 0 // returns no of elements in queue
    virtual const T& top() = 0; //returns reference to the max element
    virtual void pop() = 0; // remove the top element
    virtual void push(const T& theElement) = 0; // add element to the Queue.
};
```

program:

```
template <class T>
class maxHeap : public maxPriorityQueue
{ public:
  T *heap;
  int arrayLength;
  int HeapSize;
  public:
    // all operations such as empty(), size(), top(), pop() &
};      push
```

program: Inserting an element into max heap

```
template <class T>
void maxHeap<T>::push (const T& theElement)
{ if (heapSize == arrayLength-1) //to increase array size.
    { //double array size
      changeLength1D (heap, arrayLength, 2*arrayLength);
      arrayLength *= 2;
    }
  int currentNode = ++heapSize;
while (currentNode != 1 && heap[currentNode/2] < theElement)
{ // cannot put theElement in heap[currentNode]
  heap[currentNode] = heap[currentNode/2]; //move element to down
    currentNode /= 2;
  }
  heap[currentNode] = theElement;
}
```

Program: Removing the max element from a max heap

```
template <class T>
void maxHeap<T>::pop()
{ if (heapSize==0)
    { cout<<" max heap is empty"<<endl;
      return;
    }
// Delete max element
heap[1].~T();
T lastElement = heap[heapSize--]; // remove last element & reheapify

// find place for lastElement starting at root
int currentNode=1, child=2; // child of currentNode
while (child <= heapSize)
{ if (child < heapSize && heap[child] < heap[child+1])
    child++;          // right child value Choose with its sibling.

  // we can put the lastElement in heap[currentNode]
  if (lastElement >= heap[child])
    break;  // yes

// if no
  heap[currentNode] = heap[child]; // move child up
  currentNode = child // move down a level
  child *=2;
}
heap[currentNode] = lastElement;
}
```

program: Initialize an nonempty heap

```
template <class T>
void maxHeap<T>::Initialize (T *theHeap, int theSize)
{ // initialize max heap theHeap[1 : theSize]
   delete [] heap;
   heap = theHeap;
   heapSize = theSize;
   // heapify
   for (int root = heapSize/2 ; root >= 1 ; root--)
   {  T rootElement = heap[root];
      // find the place to put rootElement
      int child = 2*root; // parent of child is target
                          location for rootElement
      while (child <= heapSize)
      {  // heap[child] should be larger than its sibling
         if (child < heapSize && heap[child] < heap[child+1])
            child++
         // can we put rootElement in heap[child/2]
         if (rootElement >= heap[child])
            break;
         heap[child/2] = heap[child] ; // move child up
         child *= 2;   // move down a level
      }
      heap[child/2] = rootElement;
}
```

Applications

Heap Sort :

→ To sort 'n' element in max heap requires $O(n \log n)$

→ Heap sorting can be performed as follows.

Step 1: Begin by initializing a max heap with the n elements to be sorted.

Step 2: extract (or) delete (or) remove elements from the max heap one at a time.

Step 3: after removing element reheapify the max heap

step 4: Repeat step ② & step ③ until the heap became empty.

→ The initialization of max heap takes $O(n)$ time

→ deleting (or) removing element takes $O(\log n)$ time.

→ total time for Heap sorting takes $O(n \log n)$ time.

→ program below gives the code for heap sorting:

```
template <class T>
void heapSort (T a[], int n)
{ // creat max heap of the elements
   maxHeap <T> H;
   H.Initialize (a, n);
   // extract (or) remove (pop) one by one from max heap
```

// remove (or) pop one by one from max heap.

```
for( int i=n-1; i>=1; i++)
{
    T x = H.top();
    H.pop();
    a[i+1] = x;
}
}
```

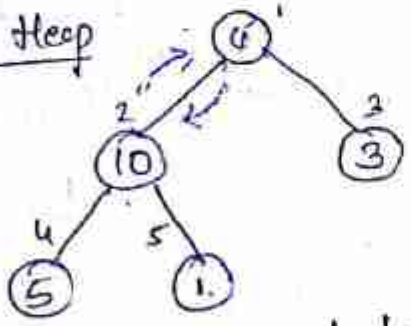
Example:

a[5] = {4, 10, 3, 5, 1} heapsize = 5

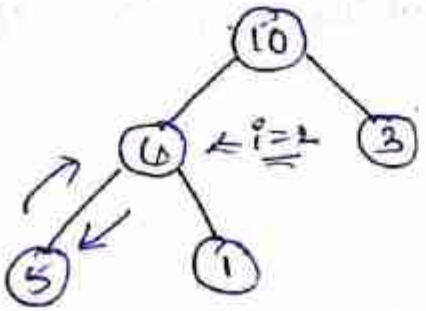
1	2	3	4	5
4	10	3	5	1

Step 1: Initialize max heap.

* Build Heap



* we need to transform Heap into max heap called heapify. [parent Node always greater than (or) equal to child nodes]



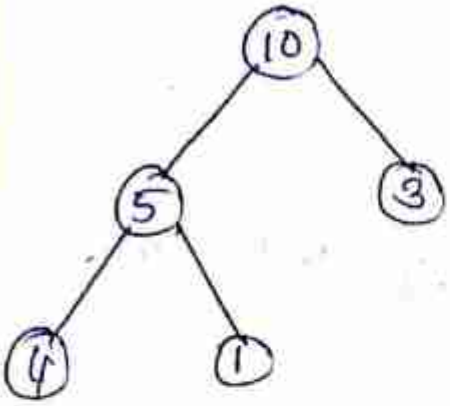
a[5]

1	2	3	4	5
10	4	3	5	1

⇒ 10 > 4

⇒ exchange 10 & 4

* left subtree of node with Element 10 is not max heap parent node (4) < its child (5).
So, exchange (4) & (5)



	1	2	3	4	5
a[5] =	10	5	3	4	1

⇒ swap (5 & 4)

Step 2; Remove the elements from max heap
Starting from root node [pop operation] max heap
* Exchange root node (10) & last element (1)

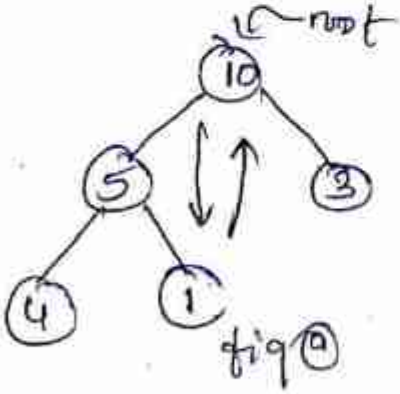


fig (a)

	1	2	3	4	5
	1	5	3	4	10

Exchange 10 & 1

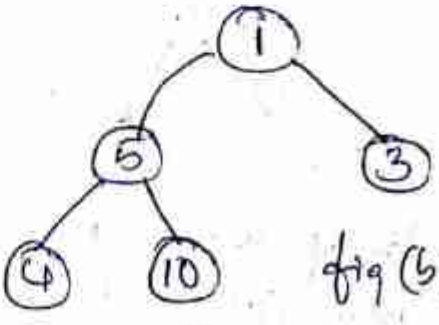


fig (b)

fig (b) shows root node & last node of elements are exchanged.
* delete node (10)

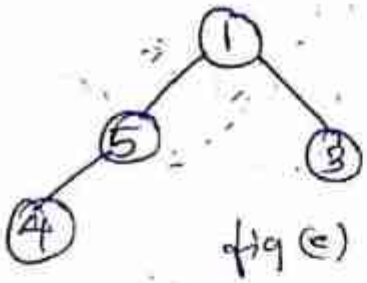


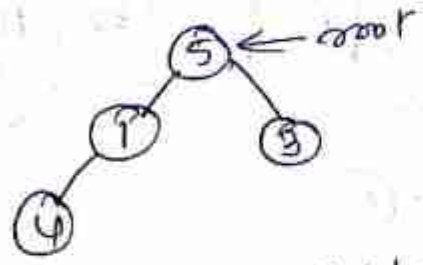
fig (c)

fig (c) → shows the node (10) removed.

1	5	3	4	10

* → heapify

→ swap (5) & (1) because parent (1) < its child (5)

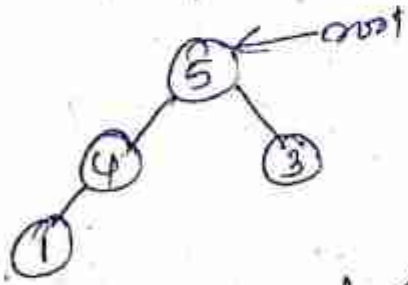


5	1	3	4	10

→ left Subtree of (5) is not max heap.

parent (1) < its child (4),

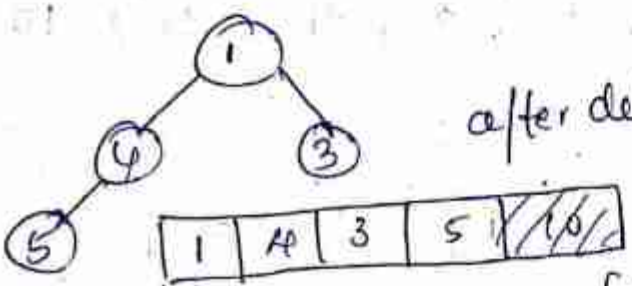
swap (1) & (4)



5	4	3	1	10

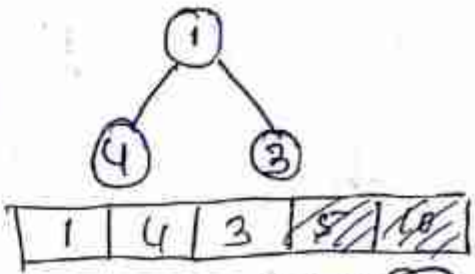
→ delete node (5)

* Swap (root (5) & last element (1))
* delete last element (5)



1	4	3	5	10

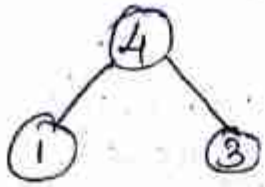
after deleting (5)

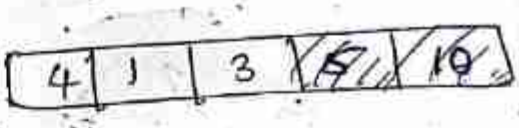


1	4	3	5	10

* heapify 1 < max {4, 3} then swap (1) & (4)

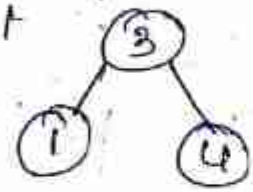
shown in fig below.

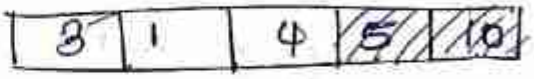




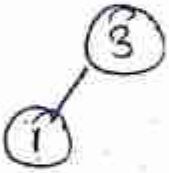
delete node (4)

* Swap (parent (4) & its child or last element (3))





* delete node with value (4)





delete node with value (3)

* Swap (parent (3) & last element (1))





* delete node with value (3)





delete node with value (1)

→ end of process.



BINARY SEARCH TREE OPERATIONS AND IMPLEMENTATION

Binaray Search tree Defination:

A binary Search tree is a binary tree that may be empty. A nonempty binary Search tree satisfies the following properties:

i) Every element has a values and no two elements have the same value.

ii) The value in the left subtree of the root smaller than the value in the root.

iii) the value in the right subtree of the root are larger than the ~~key~~ value in the root.

iv) the left and right subtrees of the root are also binary search tree.

Ex:

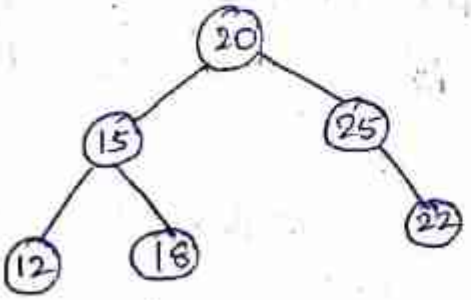


fig (a)

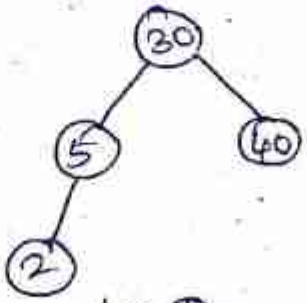


fig (b)

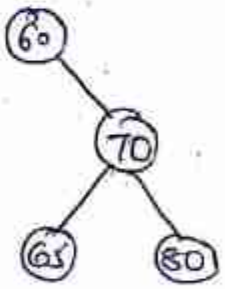


fig (c)

fig (a) → is not a binary search tree

fig (b) & fig (c) have the binary Search trees.

Operation on BST [Binary Search Tree]

* Searching in BST.
* Inserting an element into BST.
* Deleting an element from BST.
* Finding height of a Binary Search Tree (BST).

Search operation in BST

→ In a binary search tree, the search operation performed with $O(\log n)$ time complexity.

→ The search operation is performed as follows:

Step 1: Read the search element from the user

Step 2: Compare, the search element with the value of root node in the tree.

Step 3: If both are matching, then display and terminate the function

Step 4: If both are not matching, then check whether search element is smaller (or) larger than the node value.

Step 5: If search element is smaller, then continue the search process in left subtree.

Step 6: If search element is larger, then continue the search process in right subtree.

Step 7: Repeat the same until we found exact element (or) we reach the leaf node.

Step 8: If we reach to the node with search value, then display "Element is found" and terminate the function

Step 9: If we reach to a leaf node and it is also not matching, then display "Element not found" and terminate the function.

Ex:

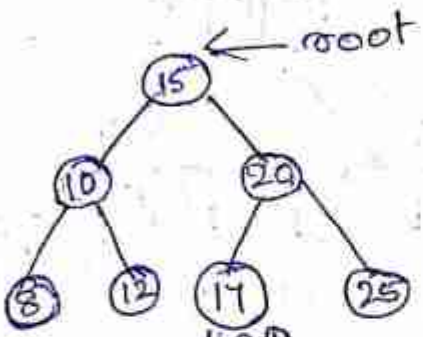


fig ①

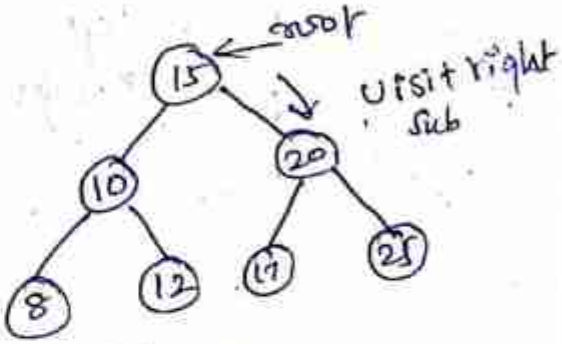


fig ②

BST

→ From above binary search tree search element (17)

* Visit root node [15] and compare element '17' with root node value 15
* '17' is greater than '15' the visit the right subtree of root node '15' shown in fig ②
* Visit node 20 in right subtree & compare with the search element 17. & it is less than node with value 20
* then visit the left subtree of node (20) show in fig ③

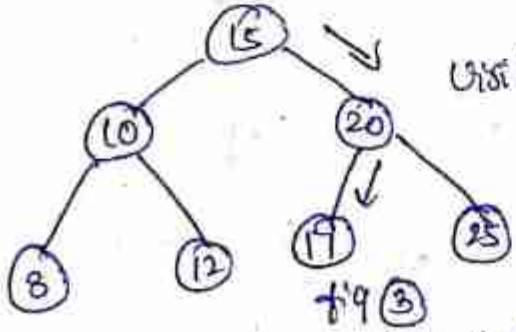


fig ③

* then compare, left subtree of node 20 & compare left child of node 20 is 17 with search element. If it

* both are equal then display "Element is found".

program for search element in BST [using doubly Linked list]

```
Struct Node
{
  int data;
  Node* left;
  Node *right;
};
```

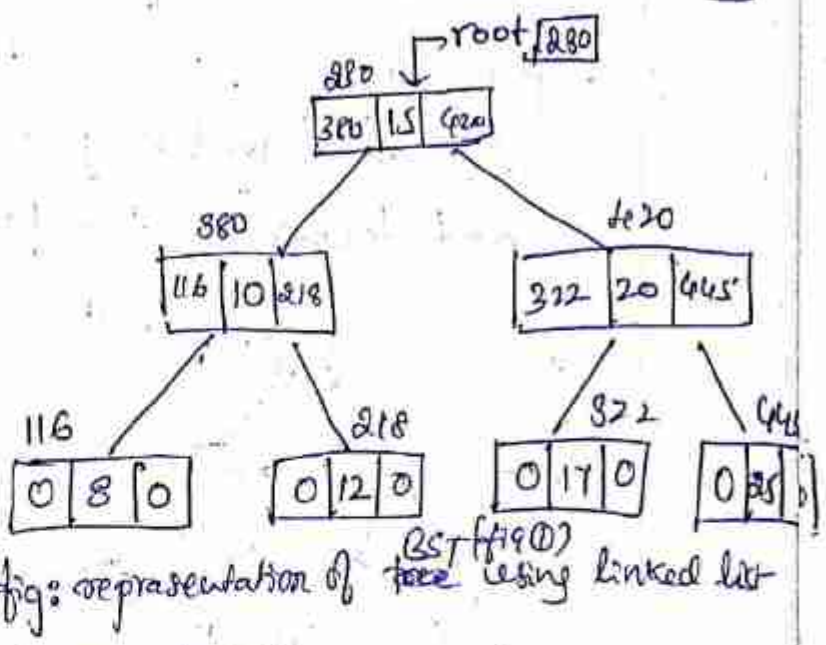


fig: representation of BST using linked list

```
bool search( Node *root, int data)
{ if (root==NULL)
    return false;
  else if (root->data == data)
    return true;
  else if (data <= root->data)
    return search (root->left, data);
  else
    return search (root->right, data);
}
int main()
{ Node *root=NULL; Creating Empty tree
  int num;
  cout<<"Enter number to be search"<<endl;
  cin>>num;
```

```
if (search(root, num) == true)
    cout<<"Element found"<<endl;
else
    cout<<"Element Not found"<<endl;
}
```

Insertion operation in BST [inserting node]

→ The Insertion operation is performed with O(log n) time.
→ In binary search tree, new node is always inserted as a leaf node.
→ The insertion operation is performed as follows:

Step1: Create a newNode with given value and set its left and right to NULL.
Step2: Check whether tree is Empty.
Step3: If the tree is empty, then set root to newnode.
Step4: If the tree is not empty, then check whether value of new Node is smaller (or) larger than the current Node.
Step5: If newNode is smaller than (or) equal to current node, then move to its left child. If new node is larger than the current Node, then move to its right child.
Step6: Repeat the above steps until we reach to a leaf node (i.e to NULL)
Step7: After reaching a leaf node, then insert the new node as left child. if new Node is smaller (or) equal to it. Otherwise insert it as right child.

Example:

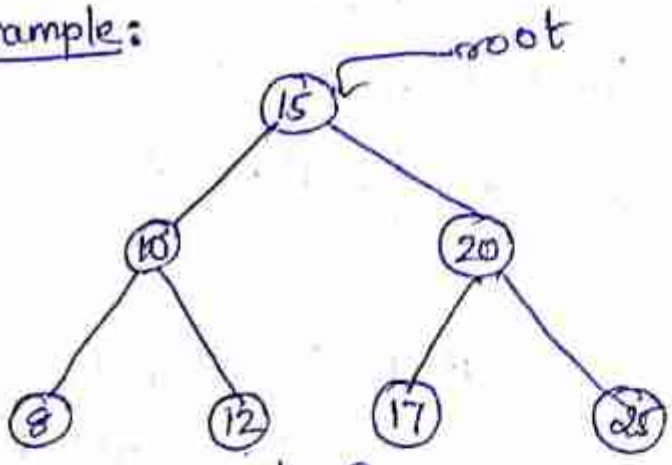


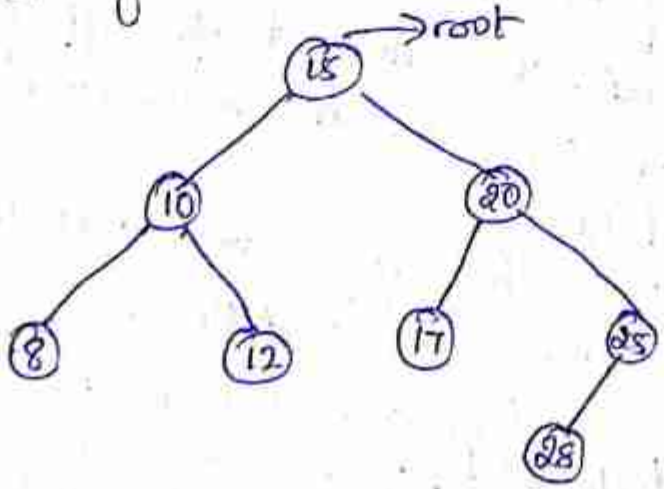
fig ⑨

→ fig ⑨ shows the BST. insert element with value '28' into the Binary Search tree.

* compare '28' with root of BST [15]. 28 > 15, hence move to its right child. (i.e 20)

* the compare node with '20' with the insert element '28'. 28 > 20. then move to the right child (25) of node '20'.

* then compare node (25) with the insert element 28 '28' greater than 25. move to right child of node '25'. But no child nodes for node (25). then we can insert new node with value '28' as the right child of node '25' as shown in figure below.



program: for insert element (or) node into BST [using doubly LL]

```
struct Node
{ int data;
  Node *left;
  Node *right;
};
Node* getNewNode (int thedata)
{
    Node* newNode = new Node;
    newNode->data = thedata;
    newNode->left = newNode->right = NULL;
    return newNode;
}
Node* insert (Node* root, int thedata )
{ if (root = NULL) //empty tree
  { root = getNewNode(thedata);
    return root;
  }
  else if (thedata <= root->data)
  { root->left = insert (root->left, thedata);
  }
  else { root->right = insert(root->right, thedata);
  }
  return root;
}
```

```
int main()
{ Node *root = NULL; // creating an empty tree
    root = Insert(root, 15);
    root = Insert(root, 10);
    root = Insert(root, 80);

  return 0;
}
```

Delete operation in BST [deleting a node]

→ The delete operation is performed with O(log n) time

→ Deleting a node from Binary Search Tree has following cases:

Case 1: Deleting a leaf node (node with no children)
Case 2: Deleting a node with one child.
Case 3: Deleting a node with two children.

Case 1: Deleting a leaf Node

We use the following steps to delete a leaf node from BST;

step 1: find the node to be deleted using search operation.

Step 2: Delete the node using delete function (If it is a leaf) and terminate the function.

Case 2: Deleting a node with one child

Step 1: Find a node to be deleted using search operation

Step 2: If it has only one child, then create a link blw its parent and child nodes.

Step 3: Delete the node using delete function and terminate the function

Case 3: Deleting a node with two children

Step 1: Find the node to be deleted using search operation

Step 2: If it has two children, then find the largest node in its left subtree (or) the smallest node in its right subtree.

Step 3: Swap both deleting node and node which found in above step.

Step 4: then check whether deleting node come to case 1 (or) case 2 else goto step 2

Step 5: If it comes to case 1, then delete using case 1 logic.

step 6: If it comes to case 2, then delete using case 2 logic;

Step 7: Repeat the same process until node is deleted from the tree.

Program: Deleting a node from Binary Tree

```
#Include<iostream>
using namespace std;
Struct Node
{ int data;
   Node * left;
   Node * right;
};
Node*  Delete(Node *root, int thedata)
{
     if(root = NULL)
        return root;
    else if(thedata < root->data)
        root->left = Delete(root->left, thedata);
    elseif(thedata > root->data)
        root->right = Delete(root->right, thedata);
    else // if data found, get ready to delete
    {
        // Case (1): No child
      if(root->left == NULL && root->right==NULL)
        { delete root;
          root = NULL;
          return root;
        }
       // Case (2): one child
   else if(root->left == NULL)
     { Node *temp=root;
```

```
        root = root->right;
        delete temp;
        return root;
    }
    else if (root->right == NULL)
    {   Node *temp = root;
        root = root->left;
        delete temp;
        return root;
    }
    // case③: children
    else
    {   Node *temp = findmin(root->right);
        root->data = temp->data
        root->right = Delete(root->right, temp->data);
    }
}
return root;
}
```

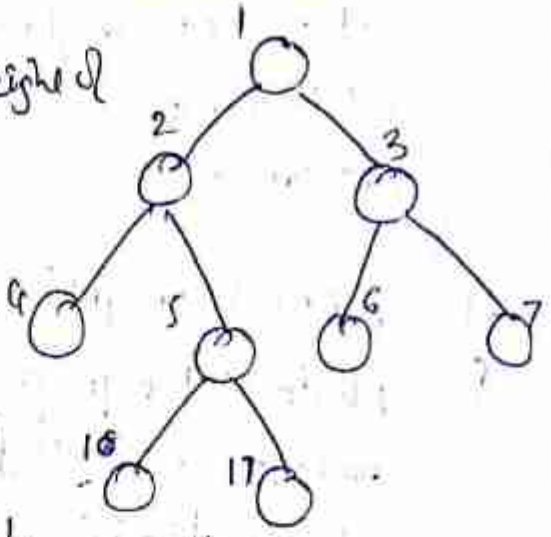
Find Height of a binary tree

→ Height of tree is the number of edges in longest path from root to a leaf node.

→ Height of tree = Height of root.

→ Height of tree = number of edges in longest path + 1

From the figures the height of tree is.....



longest path is

1 → 2 → 5 → 10 (or)

1 → 2 → 5 → 11

height = number of edges in longest path + 1

= 3 + 1

height = 4

program:

```
template <class T>
int findheight (node *root)
{  if(root == NULL)
     {. cout<<"tree is empty"<<endl;
        return;
     }
int leftHeight = find_height(root->left);
int rightHeight = find_height(root->right);

return (max(leftHeight, rightHeight)+1);
}
```

```
struct node
{  T data;
   node *left;
   node *right;
};
```