ADICHUNCHANAGIRI UNIVERSITY

B G S INSTITUTE OF TECHNOLOGY

B G Nagar – 571448.

DEPARTMENT
OF
COMPUTER SCIENCE AND ENGINEERING

<u>LAB COURSE FILE & MANUAL</u>

Academic Year	:	2019 – 2020 (ODD SEMESTER)
Programme (UG/PG)	:	UG
Year / Semester	:	2nd Year / 3th Semester
Course Code	:	18CSL36
Course Title	:	Nethravathi H M

Prepared By:

Mrs Nethravathi H M
Assistant Professor,
Department of CS&E
B G S Institute of Technology

B G S INSTITUTE OF TECHNOLOGY

VISION

BGSIT is commited to the cause of creating tomorrow's engineers by providing quality education inculcating ethical values.

MISSION

M1: Imparting quality technical education by nurturing a conducive learning environment.

M2: Offering professional training to meet industry requirements.

M3: Providing education with a moral-cultural base and spiritual touch.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION

To produce engineers by possessing good technical knowledge and ethics through quality education and research.

MISSION

M1: Achieve excellence by providing good infrastructure and competent faculty.

M2: Strengthening the technical, soft skills, leadership qualities and ethical values to meet the industry requirements.

M3: Facilitate experimental learning through research projects

INSTRUCTIONS TO STUDENTS

Computer Lab Safety Rules for Protecting Equipment

- Turn off the machine once you are done using it.
- Do not plug in external devices without scanning them for computer viruses.
- Try not to touch any of the circuit boards and power sockets when a device is connected to them and switched on.
- Always maintain an extra copy of all your important data files.

General Safety Guidelines to be followed at all times

- All users of the laboratory are to follow the directions of Academic/Laboratory Technician staff member.
- Students should not attempt to repair, open, tamper or interfere with any of the computer, printing, cabling, air conditioning or other equipment in the laboratory.
- Students should be aware of office ergonomic guidelines for correct posture when using computer equipment.
- Please treat fellow users of the laboratory, and all equipment within the laboratory, with the appropriate level of care and respect.

DO's AND DON'TS

Do's

- Enter the log register
- Follow the dress code and ware ID card
- Always keep quiet. Be considerate to other lab users.
- Report any problems with the computer to the person in charge.
- Shut down the computer properly and keep the chairs aligned before leaving the lab.
- Know the location of the fire extinguisher and the first aid box and how to use them in case of an emergency.
- Report any broken plugs or exposed electrical wires to your lecturer/laboratory technician immediately.

Don'ts

- Do not eat or drink in the laboratory.
- Do not use mobile phone.
- Don't damage, remove, or disconnect any labels, parts, cables or equipment.
- Avoid stepping on electrical wires or any other computer cables.
- Do not install or download any software or modify or delete any system files on any lab computers.
- If you leave the lab, do not leave your personal belongings unattended.
- Do not open the system unit casing or monitor casing particularly when the power is turned on.
- Do not insert metal objects such as clips, pins and needles into the computer casings. They may cause fire.

LABORATORY RUBRICS

1. FOR 25 MARKS (2010 SCHEME)

Sl. No.	Description	Marks
1	**Continuous Evaluation** **a. Observation write up and punctuality** **b. Conduction of experiment and output** **c. Viva voice** **d. Record write up**	**15** **2.5** **5.0** **2.5** **5.0**
2	**Internal Test**	**10**

2. FOR 20 MARKS (2015 CBCS SCHEME)

Sl. No.	Description	Marks
1	**Continuous Evaluation** **a. Observation write up and punctuality** **b. Conduction of experiment and output** **c. Viva voice** **d. Record write up**	**12** **2.0** **4.0** **2.0** **4.0**
2	**Internal Test**	**08**

3. FOR 40 MARKS (2017 REVISED CBCS SCHEME)

Sl. No.	Description	Marks
1	**Continuous Evaluation** **a. Observation write up and punctuality** **b. Conduction of experiment and output** **c. Viva voice** **d. Record write up**	**30** **5.0** **10.0** **5.0** **10.0**
2	**Internal Test**	**10**

4. FOR 40 MARKS (2018 CBCS SCHEME)

Sl. No.	Description	Marks
1	**Continuous Evaluation** **e. Observation write up and punctuality** **f. Conduction of experiment and output** **g. Viva voice** **h. Record write up**	**30** **5.0** **10.0** **5.0** **10.0**
2	**Internal Test**	**10**

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Ability to apply Mathematical Methodologies, Management Principles and Ethics, Electronics and Embedded Systems and Programming Technologies to solve real time problems.

PSO 2: Ability to apply software design and development practices to develop software in emerging areas such as Internet of Things, Data Management, Social Networking and Security, Cloud and High Performance Computing.

COURSE OUTCOMES

Upon successful completion of this course, students should be able to:

Subject code:	**18CSL36**
CO1	Make Use of various Electronic devices like cathode ray oscilloscope, signal generators, digital trainer kit, millimeter and components like resistor, capacitor, opamp and integrated circuit.
CO2	Design and demonstrate various combinational logic circuits
CO3	Design and demonstrate various types of counters and Registers using Flip-flops
CO4	Make Use of simulation package to design circuits.

CO-PO-PSO MAPPING

COs	**POs**												**PSOs**	
	PO1	PO2	PO3	PO4	PO5	PO6	P07	PO8	PO9	P010	PO11	PO12	PSO1	PSO2
CO1	2	2	1	2	-	-	3	-	-	-	-	-	2	-
CO2	2	2	2	1	-	-	3	-	-	-	-	-	2	-
CO3	1	2	2	1	-	-	3	-	-	-	-	-	3	-
CO4	1	2	2	1	-	-	2	-	-	-	-	-	1	-
AVG	**1.5**	**2**	**1.7**	**1.2**			**2.7**						**2**	

ANALOG AND DIGITAL ELECTRONICS LABORATORY
Adichunchanagiri University
(Effective from the Academic Year 2019 -

20)
SEMESTER – III

Subject Code	18CSL36	**CIE Marks**	40
Number of Contact Hours/Week	0:2:2	**SEE Marks**	60
Total Number of Lab Contact Hours	36	**Exam Hours**	3 Hrs

CREDITS – 2

Course Learning Objectives: This course (18CSL36) will enable students to:

This laboratory course enable students to get practical experience in design, assembly and evaluation/testing of

- Analog components and circuits including Operational Amplifier, Timer,etc.
- Combinational logic circuits.
- Flip - Flops and their operations
- Counters and registers using flip-flops.
- Synchronous and Asynchronous sequential circuits.
- A/D and D/Aconverters

Descriptions (if any):

- Simulation packages preferred: Multisim, Modelsim, PSpice or any otherrelevant.
- For Part A (Analog Electronic Circuits) students must trace the wave form on Tracing sheet / Graph sheet and labeltrace.
- Continuous evaluation by the faculty must be carried by including performance of a student in both hardware implementation and simulation (if any) for the givencircuit.
- A batch not exceeding 4 must be formed for conducting the experiment. For simulation individual student must execute theprogram.

Laboratory Programs:

PART A (Analog Electronic Circuits)

1.	Design an astable multivibrator circuit for given duty cycle using NE 555 timer IC.
2.	Using ua 741 Opamp, design a Inverting Amplifier circuit and non inverting amplifier circuit
3.	Using ua 741 opamap, design a Schmitt trigger for any given UTP and LTP. And simulate the same.
4.	a. Design and construct a rectangular waveform generator (Op-Amp relaxation oscillator) for given frequency. b. Design and implement a rectangular waveform generator (Op-Amp relaxation oscillator) using a simulation package and observe the change in frequency when all resistor values are doubled.

PART B (Digital Electronic Circuits)

5.	Design and implement Half adder, Full Adder, Half Subtractor, Full Subtractor using basic gates. And implement the same in HDL.
6.	Given a 4-variable logic expression, simplify it using appropriate technique and realize the simplified logic expression using 8:1 multiplexer IC. And implement the same in HDL.
7.	Realize a J-K Master / Slave Flip-Flop using NAND gates and verify its truth table. And implement the same in HDL.
8.	Design and implement a mod-n (n<8) synchronous up counter using J-K Flip-Flop ICs and demonstrate its working.
9.	Design and implement an asynchronous counter using decade counter IC to count up from 0 to n (n<=9) and demonstrate on 7-segment display (using IC-7447)
10.	Design and implement code converter i)Binary to Gray code converter ii) Gray to Binary code converter

Laboratory Outcomes: The student should be able to:

- Use appropriate design equations / methods to design the given circuit.
- Examine and verify the design of both analog and digital circuits using simulators.
- Make us of electronic components, ICs, instruments and tools for design and testing of circuits for the given the appropriate inputs.
- Compile a laboratory journal which includes; aim, tool/instruments/software/components used, design equations used and designs, schematics, program listing, procedure followed, relevant theory, results as graphs and tables, interpreting and concluding the findings.

Conduct of Practical Examination:

- All laboratory experiments, excluding the first, are to be included for practical examination.
- Experiment distribution
  - For questions having only one part: Students are allowed to pick one experiment from the lot and are given equal opportunity.
  - For questions having part A and B: Students are allowed to pick one experiment from part A and one experiment from part B and are given equal opportunity.
- Change of experiment is allowed only once and marks allotted for procedure part to be made zero.
- Marks Distribution *(Subjected to change in accordance with university regulations)*
  a) For questions having only one part – Procedure + Execution + Viva-Voce: 15+70+15= 100 Marks
  b) For questions having part A andB
     i. PartA–Procedure+Execution+Viva=4+21+5=30Marks
     ii. PartB–Procedure+Execution+Viva=10+49+11=70Marks

ANALOG ELECTRONICS

Most of the fundamental electronic components – resistors, capacitors, inductors, diodes, transistors, and operational amplifiers – are all inherently analog. Circuits built with a combination of solely these components are usually analog.

Operational Amplifiers

Operational Amplifiers are the heart and soul of all modern electronic instruments. Their flexibility, stability and ability to execute many functions make op-amps the ideal choice for analog circuits. Historically, op-amps evolved from the field of analog computation where circuits were designed to add, subtract, multiply, integrate, and differentiate etc. in order to solve differential equations found in many engineering applications. Today analog computers op-amps are found in countless electronic circuits and instruments.

Operational Amplifiers (OAs) are highly stable, high gain dc difference amplifiers. Since there is no capacitive coupling between their various amplifying stages, they can handle signals from zero frequency (dc signals) up to a few hundred kHz. Their name is derived by the fact that they are used for performing mathematical operations on their input signal(s).

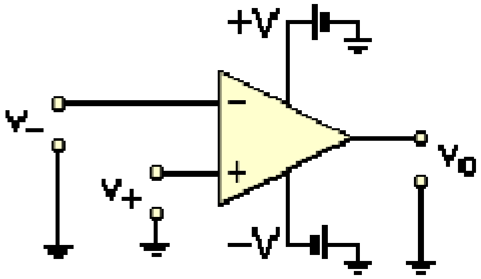


Figure 1. Symbol of the operational amplifier. Connections to power supplies are also shown. The output signal (voltage), vo, is given by: vo = A (v+ - v-).Figure 1 shows the symbol for an OA. There are two inputs, the inverting input (-) and the noninverting input (+). These symbols have nothing to do with the polarity of the applied input signals.

Analog circuits are usually complex combinations of op amps, resistors, caps, and other foundational electronic components.

The Major Equipment that will be used in Analog Electronics lab is

1. CRO : Cathode Ray Oscilloscope
2. ASG: Amplitude Signal Generator

An oscilloscope, previously called an oscillograph, and informally known as a scope, CRO (for cathode ray oscilloscope), or DSO (for the more modern digital storage oscilloscope), is a type of electronic test instrument that allows observation of constantly varying signal voltages. Actually cathode ray oscilloscope is very fast X-Y plotters that can display an input signal versus time or other signal as shown below. Oscilloscopes are used in the sciences, medicine, engineering, and the telecommunications industry. General-purpose instruments are used for maintenance of electronic equipment and laboratory work. Special-purpose oscilloscopes may be used for such purposes as analyzing an automotive ignition system or to display the waveform of the heartbeat as an electrocardiogram.

Description:

The basic oscilloscope, as shown in the illustration, is typically divided into four sections: the display, vertical controls, horizontal controls and trigger controls. The display is usually a CRT or LCD panel which is laid out with both horizontal and vertical reference lines referred to as the graticule. In addition to the screen, most display sections are equipped with three basic controls: a focus knob, an intensity knob and a beam finder button.

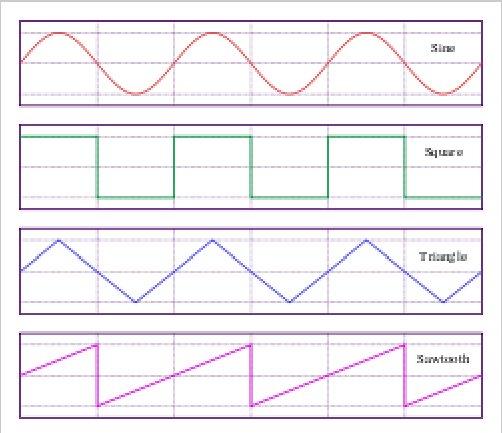
The vertical section controls the amplitude of the displayed signal. This section carries a Volts-per- Division (Volts/Div) selector knob, an AC/DC/Ground selector switch and the vertical (primary) input for the instrument. Additionally, this section is typically equipped with the vertical beam position knob.

The horizontal section controls the time base or "sweep" of the instrument. The primary control is the Seconds-per-Division (Sec/Div) selector switch. Also included is a horizontal input for plotting dual X-Y axis signals. The horizontal beam position knob is generally located in this section.

2. ASG: Amplitude Signal Generator

A function generator is usually a piece of electronic test equipment or software used to generate different types of electrical waveforms over a wide range of frequencies. Some of the most common waveforms produced by the function generator are the sine, square, triangular and saw tooth shapes. These waveforms can be either repetitive or single-shot.

Function generators are used in the development, test and repair of electronic equipment. For example, they may be used as a signal source to test amplifiers or to introduce an error signal into a control loop.

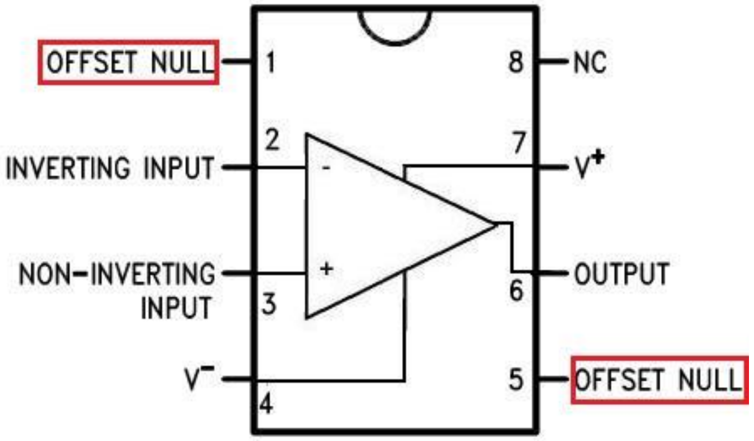


Sine, square, triangle, and sawtooth waveforms

Function generator basics:

Function generators, whether the old analog type or the newer digital type, have a few common features: • A way to select a waveform type: sine, square, and triangle are most common, but some will give ramps, pulses, –noise‖, or allow you to program a particular arbitrary shape. • A way to select the waveform frequency. Typical frequency ranges are from 0.01 Hz to 10 MHz. • A way to select the waveform amplitude. • At least two outputs. The –main‖ output, which is where you find the desired waveform, typically has a maximum voltage of 20 volts peak-to-peak, or ±10 volts range. The most common output impedance of the main output is 50 ohms, although lower output impedances can sometimes be found. A second output, sometimes called –sync‖, –aux‖ or –TTL‖ produces a square wave with standard 0 and 5 volt digital signal levels. It is used for synchronizing another device (such as an oscilloscope) to the possibly variable main output signal. A wide variety of other features are available on most modern function generators, such as –frequency sweep‖—the ability to automatically vary the frequency between a minimum and maximum value, –DC offset‖—a knob that adds a specified amount of DC voltage to the time-varying 1 waveform, and extra inputs or outputs that can be used to control these extra features by other instruments.

LM741 Pinout Diagram



DIGITAL ELECTRONICS

Digital circuits operate using digital, discrete signals. These circuits are usually made of a combination of transistors and logic gates and, at higher levels, microcontrollers or other computing chips. Most processors, whether they‘re big beefy processors in your computer, or tiny little microcontrollers, operate in the digital realm.

Logic Design: Circuit that takes the logical decision and the process are called logic gates. Each gate has one or more input and only one output. OR, AND and NOT are basic gates. NAND, NOR are known as universal gates. Basic gates form these gates.

AND GATE: The AND gate performs a logical multiplication commonly known as AND function. The output is high when both the inputs are high. The output is low level when any one of the inputs is low.

OR GATE: The OR gate performs a logical addition commonly known as OR function. The output is high when any one of the inputs is high. The output is low level when both the inputs are low.

NOT GATE: The NOT gate is called an inverter. The output is high when the input is low. The output is low when the input is high.

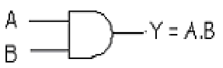
NAND GATE: The NAND gate is a contraction of AND-NOT. The output is high when both inputs are low and any one of the input is low .The output is low level when both inputs are high.

NOR GATE: The NOR gate is a contraction of OR-NOT. The output is high when both inputs are low. The output is low when one or both inputs are high.

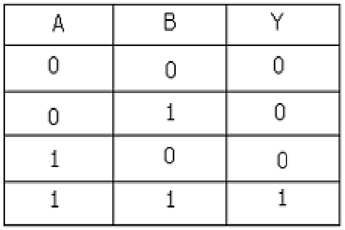
X-OR GATE: The output is high when any one of the inputs is high. The output is low when both the inputs are low and both the inputs are high.

AND GATE:

SYMBOL

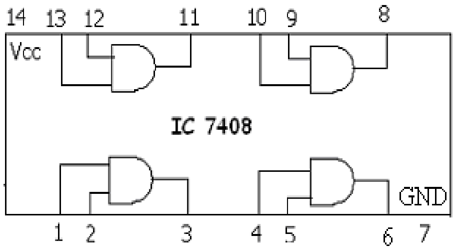


TRUTHTABLE



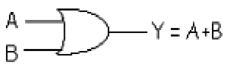
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

IC7408



OR GATE:

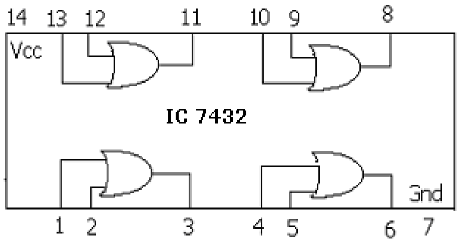
SYMBOL



TRUTH TABLE

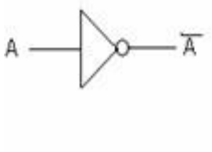
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

IC 7432



NOT GATE:

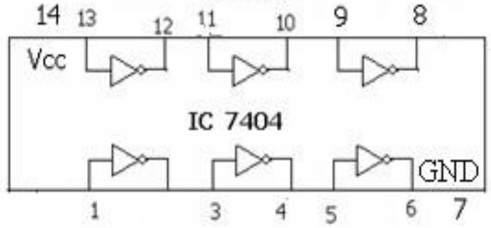
SYMBOL



TRUTH TABLE

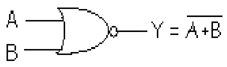
I/P (A)	O/P $(\overline{A})$
0	1
1	0

IC 7404



NOR GATE:

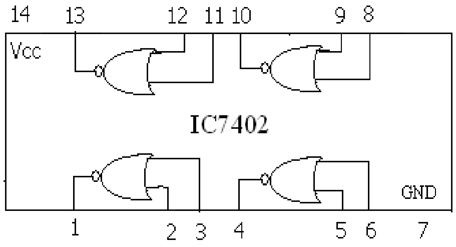
SYMBOL



TRUTH TABLE

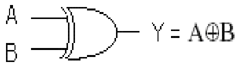
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

IC7402



X-OR GATE:

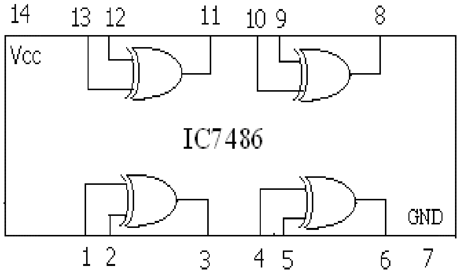
SYMBOL



TRUTH TABLE

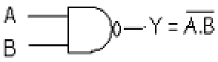
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

IC 7486

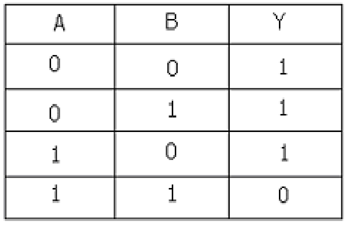


2-INPUT NAND GATE:

SYMBOL

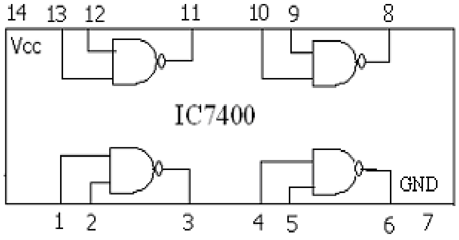


TRUTHTABLE



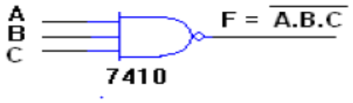
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

IC7408

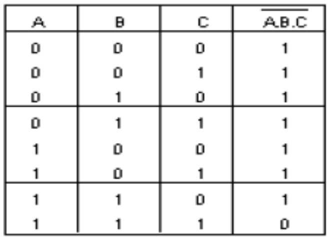


3-INPUT NAND GATE:

SYMBOL:

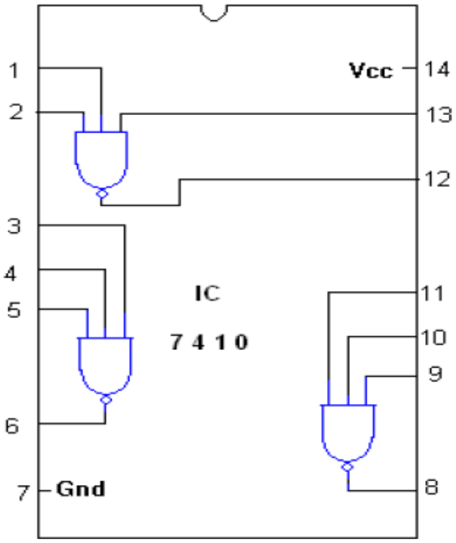


TRUTH TABLE



A	B	C	$\overline{A.B.C}$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

PIN DIAGRAM:



Multiplexer Multiplexer (or mux) is a device that selects one of several analog or digital input signals and forwards the selected input into a single line. A multiplexer of 2^ninputs has *n* select lines, which are used to select which input line to send to the output. Multiplexers are mainly used to increase the amount of data that can be sent over the network within a certain amount of time and bandwidth. A multiplexer is also called a **data selector**. Multiplexers can also be used to implement Boolean functions of multiple variables.

Shift register In digital circuits, a shift register is a cascade of flip flops, sharing the same clock, in which the output of each flip-flop is connected to the data' input of the next flip-flop in the chain, resulting in a circuit that shifts by one position the 'bit array' stored in it, ‒shifting in‖ the data present at its input and ‒shifting out‖ the last bit in the array, at each **transition** of the clock input.

Counter A Counter is a device which stores (and sometimes displays) the number of times a particular event or process has occurred, often in relationship to a clock signal.

EXPERIMENT -1

1.Design an astable multivibrator ciruit for three cases of duty cycle using NE 555 timer IC.

AIM : To design and implement an Astable multivibrator using 555 Timer for a given frequency and duty cycle.

COMPONENTS REQUIRED: 555 Timer IC, Resistors of 3.3KΩ, 6.8KΩ, Capacitors of 0.1 μF, 0.01 μF, Regulated power supply and CRO.

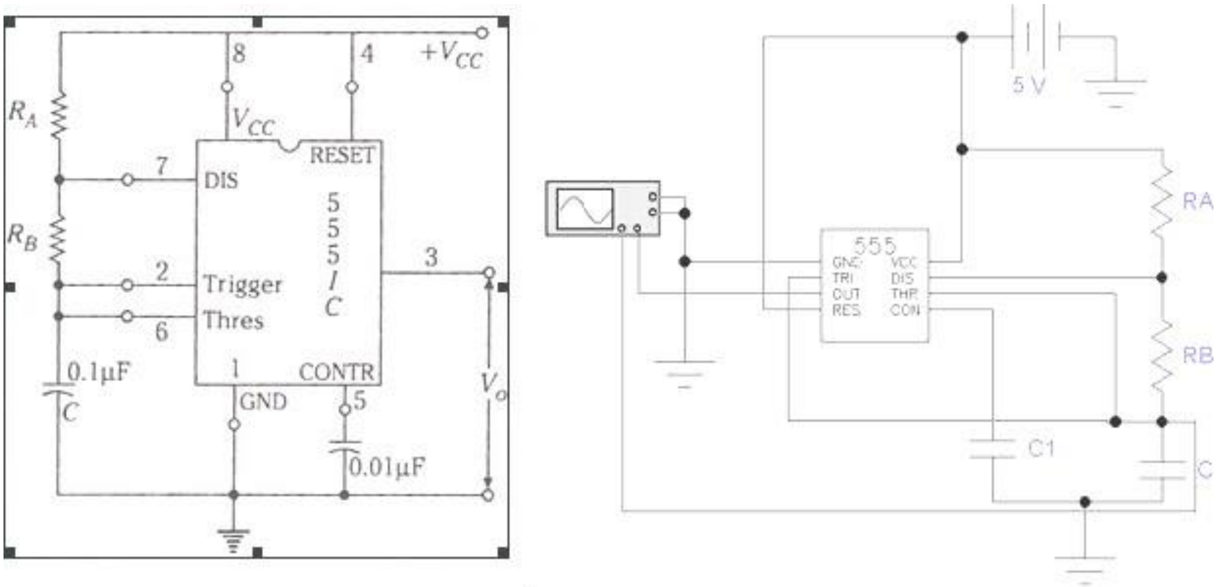


Fig: Circuit Diagram and actual connections

1. Design an Astable Multivibrator for the given frequency of 1KHZ and duty cycle of 60%.

Solution: Since the given frequency if 1KHZ, Hence T

= 1/F = 1/1KHZ = 1ms.

Duty cycle D = 60% = 60/100 = 0.6

Capacitor charging time = ton = 0.69R1C ----------------- (1)

Capacitor discharging time = toff = 0.69 R2 C----------- (2)

The Total Time is T = Ton +Toff ---------------------------(3)

The expression for Duty cycle is given by: D = Ton/T using this expression calculate for Ton

0.6 = Ton/1ms : Ton = 0.6 * 1ms = 0.6ms Now

calculate Toff from expression (3) T off = T-Ton

= 1ms – 0.6ms = 0.4ms Hence Ton = 0.6 ms and

Toff = 0.4ms

Now find the values of R1, R2 and C1 from expressions (1) and (2).

From (1) ton = 0.69R1C

w.k.t Ton = 0.6ms Assume C = 0.1µf , R1 =

Ton/0.69*0.1 µf. = 8.69KΩ

Similarly calculate for R2 from expression (2). The value of R2 = 5.9 kΩ.

2. Design an Astable Multivibrator for the given frequency of 1KHZ and duty cycle of 50%.

Solution: Since the given frequency if 1KHZ, Hence T

= 1/F = 1/1KHZ = 1ms.

Duty cycle D = 50% = 50/100 = 0.5

Capacitor charging time = ton = 0.69R1C ----------------- (1)

Capacitor discharging time = toff = 0.69 R2 C----------- (2)

The Total Time is T = Ton +Toff ---------------------------(3)

The expression for Duty cycle is given by: D = Ton/T using this expression calculate for Ton 0.5 =

Ton/1ms : Ton = 0.5 * 1ms = 0.5ms

Now calculate Toff from expression (3) T off =

T-Ton = 1ms – 0.5ms = 0.5ms Hence Ton =

0.6 ms and Toff = 0.5ms

Now find the values of R1, R2 and C1 from expressions (1) and (2).

From (1) ton = 0.69R1C

w.k.t Ton = 0.5ms Assume C = 0.1µf , R1 =

Ton/0.69*0.1 µf. = 7.2 KΩ

Similarly calculate for R2 from expression (2). The value of R2 = 7.2kΩ.

The Vcc determines the upper and lower threshold voltages (observed from the capacitor voltage waveform) as $V_{UT} = \frac{2}{3}V_{CC}$ & $V_{LT} = \frac{1}{3}V_{CC}$.

Note: The duty cycle determined by R_1 & R_2 can vary only between 50 & 100%. If R_A is much smaller than R_B, the duty cycle approaches 50%.

Example 2: frequency = 1kHz and duty cycle =75%, R_A = 7.2kΩ & R_B =3.6kΩ, choose R_A = 6.8kΩ and R_B = 3.3kΩ.

3. Design an Astable Multivibrator for the given frequency of 1KHZ and duty cycle of 75%.

Frequency = 1 kHz and duty cycle =75%, RA = 7.2kΩ & RB =3.6kΩ,

Duty cycle = tH / T = 0.75. Hence tH = 0.75T = 0.75ms and tL = T – tH = 0.25ms. Let

C=0.1µF and substituting in the above equations,

So RB= tL/0.693 x C =0.25 x 10-3/0.693 x .1x10-6=3.6kΩ RA =

(tH - 0.693 x RB x C) /0.693 x C

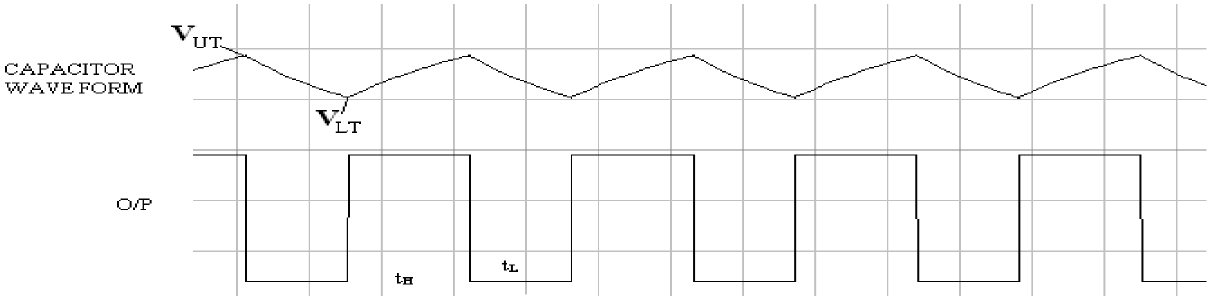
= 0.75 x 10-3 x 0.693 x 3.6 x 103 x 0.1x 10-6 / 0.693 x .1x10-6=7.2kΩ

Choose RA = 6.8kΩ and RB = 3.3kΩ.

PROCEDURE:

1. Before making the connections, check the components using multimeter.
2. Make the connections as shown in figure and switch on the power supply.
3. Observe the capacitor voltage waveform at 6th pin of 555 timer on CRO.
4. Observe the output waveform at 3rd pin of 555 timer on CRO (shown below).
5. Note down the amplitude levels, time period and hence calculate duty cycle.

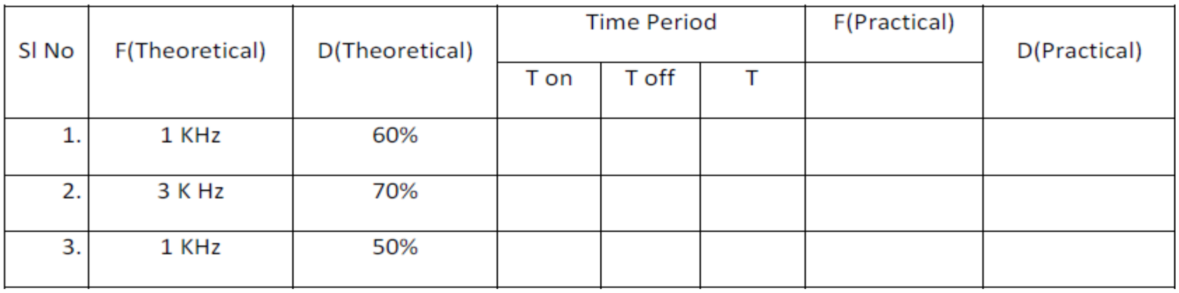
WAVEFORMS



THEORY:

Multivibrator is a form of oscillator, which has a non-sinusoidal output. The output waveform is rectangular. The multivibrators are classified as: **Astable or free running multivibrator**: It alternates automatically between two states (low and high for a rectangular output) and remains in each state for a time dependent upon the circuit constants. It is just an oscillator as it requires no external pulse for its operation. **Monostable or one shot multivibrator**: It has one stable state and one quasi stable. The application of an input pulse triggers the circuit time constants. After a period of time determined by the time constant, the circuit returns to its initial stable state. The process is repeated upon the application of each trigger pulse. **Bistable Multivibrators**: It has both stable states. It requires the application of an external triggering pulse to change the output from one state to other. After the output has changed its state, it remains in that state until the application of next trigger pulse. Flip flop is an example.

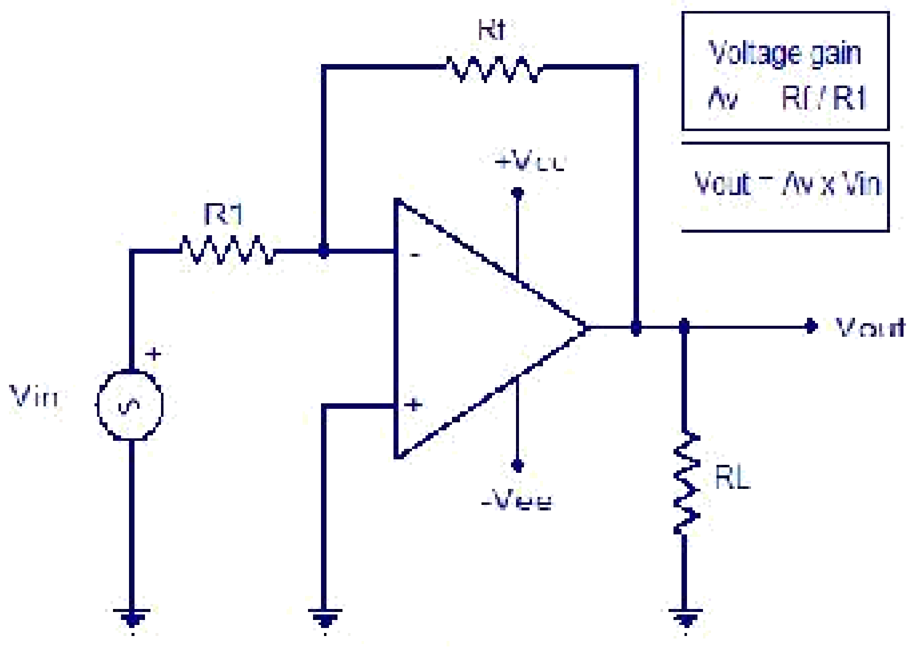
RESULTS:



Sl No	F(Theoretical)	D(Theoretical)	Time Period			F(Practical)	D(Practical)
			T on	T off	T		
1.	1 KHz	60%					
2.	3 K Hz	70%					
3.	1 KHz	50%					

EXPERIMENT -2
INVERTING AND NON INVERTING AMPLIFIER CIRCUIT

AIM: Design and realize Inverting and Non-inverting amplifier using 741 Op-amp.
Apparatus Required: CRO, Function Generator, Bread Board, 741 IC, ±12V supply, Resistors 1KΩ, 10KΩ, and connecting leads.
Theory:
An inverting amplifier using opamp is a type of amplifier using opamp where the output waveform will be phase opposite to the input waveform. The input waveform will be amplifier by the factor Av (voltage gain of the amplifier) in magnitude and its phase will be inverted. In the inverting amplifier circuit the signal to be amplified is applied to the inverting input of the opamp through the input resistance R1. Rf is the feedback resistor. Rf and Rin together determine the gain of the amplifier. Inverting operational amplifier gain can be expressed using the equation Av = – Rf/R1. Negative sign implies that the output signal is negated. The circuit diagram of a basic inverting amplifier using opamp is shown below.



opamp inverting amplifier

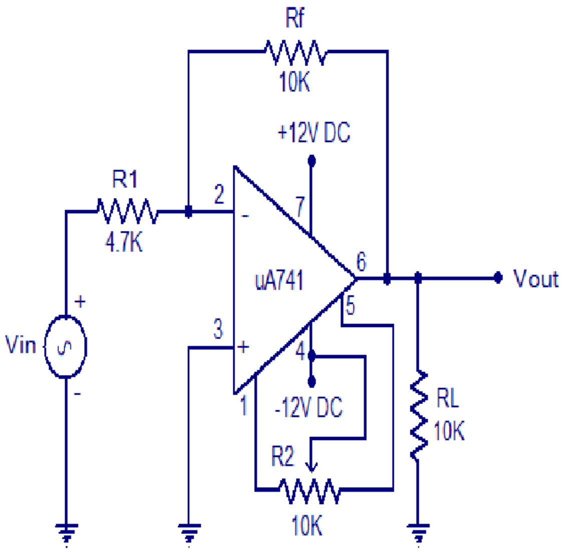
The input and output waveforms of an inverting amplifier using opamp is shown below. The graph is drawn assuming that the gain (Av) of the amplifier is 2 and the input signal is a sine wave. It is clear from the graph that the output is twice in magnitude when compared to the input (Vout = Av x Vin) and phase opposite to the input.

Practical inverting amplifier using 741.

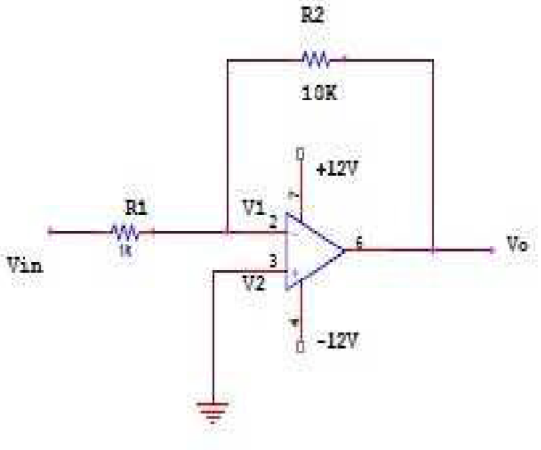
A simple practical inverting amplifier using 741 IC is shown below. uA 741 is a high performance and of course the most popular operational amplifier. It can be used in a verity of applications like integrator,

differentiator, voltage follower, amplifier etc. uA 741 has a wide supply voltage range (+/-22V DC) and has a high open loop gain. The IC has an integrated compensation network for improving stability and has short circuit protection. Signal to be amplified is applied to the inverting pi (pin2) of the IC. Non inverting pin (pin3) is connected to ground. R1 is the input resistor and Rf is the feedback resistor. Rf and R1 together sets the gain of the amplifier. With the used values of R1 and Rf the gain will be

(Av = -Rf/R1 = 10K/1K = 10). RL is the load resistor and the amplified signal will be available across it. POT R2 can be used for nullifying the output offset voltage. If you are planning to assemble the circuit, the power supply must be well regulated and filtered. Noise from the power supply can adversely affect the performance of the circuit. When assembling on PCB it is recommended to mount the IC on the board using an IC base.



opamp inverting amplifier



INVERTING AMPLIFIER

In the inverting amplifier only one input is applied and that is to the inverting input (V2) terminal. The Non inverting input terminal (V1) is grounded. Since, V1=0

V& V2=Vin

Vo= -A Vin

The negative sign indicates the output voltage is 1800 out of phase with respect to the input and amplified by gain A.

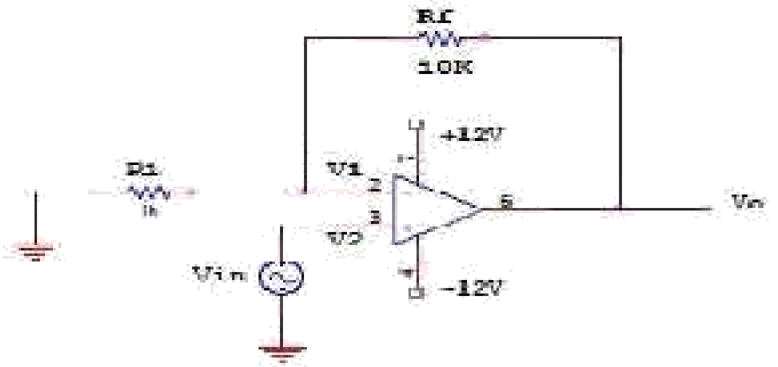
Practical Non-inverting amplifier using 741:

The input is applied to the non-inverting input terminal and the Inverting terminal is connected to the ground.

V1= Vin & V2=0 Volts

Vo= A Vin

The output voltage is larger than the input voltage by gain A & is in phase with the input signal.



NON-INVERTING AMPLIFIER

Procedure:

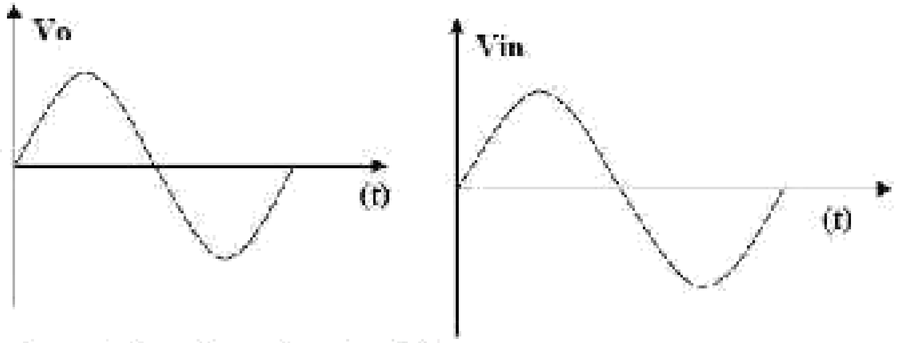
1) Connect the circuit for inverting, non inverting amplifier on a breadboard.

2) Connect the input terminal of the op-amp to function generator and output terminal to CRO.

3) Feed input from function generator and observe the output on CRO.

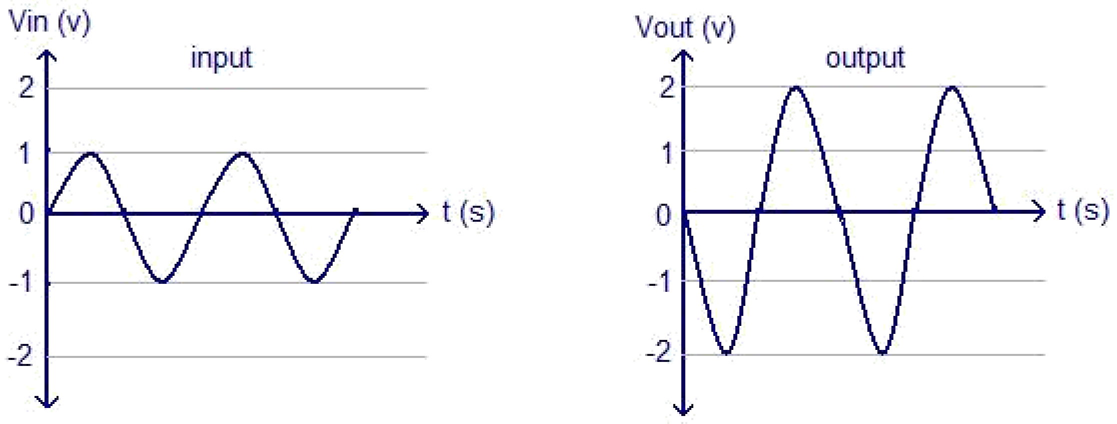
4) Draw the input and output waveforms on graph paper.

Output Waveform:



Output: Non- Inverting Amplifier

Input and output waveforms of an opamp inverting amplifier (gain assumed to be 2)



RESULT:

Hence verified and drawn the operation and respective waveforms of inverting and non-inverting amplifier**.**

EXPERIMENT -2

OP-AMP AS A RELAXATION OSCILLATOR

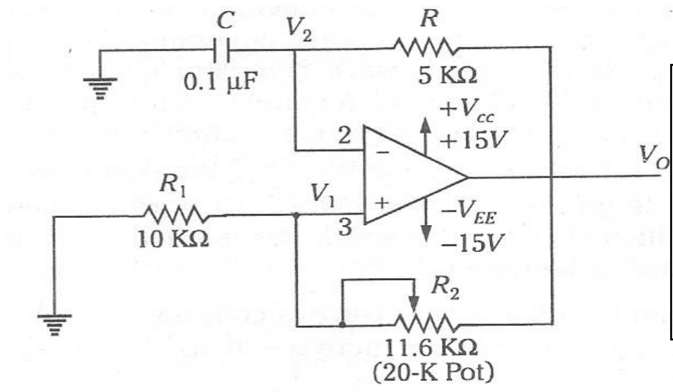
Using ua 741 Opamp, design a 1 kHz Relaxation Oscillator with 50% duty cycle. and simulate the

same.

AIM : To design and implement a rectangular waveform generator (op-amp relaxation oscillator) for a given frequency.

COMPONENTS REQUIRED:

Op-amp µA 741, Resistor of 1KΩ, 10KΩ, 20 kΩ Potentiometer, Capacitor of 0.1 µF, Regulated DC power supply, CRO



Values

C=0.1µF

R_1 = 10kΩ, R_2 = 11.6 kΩ,R = 4.7k/5.1kΩ

Fig: Circuit Diagram & actual connections

DESIGN:

T =2RC ln (1+β/1- β)--------- 1
Where,
β =R1/R1+ R2 is the feedback fraction

If R1 = R2, then from equation (1) we have T = 2RC ln(3)

Design a Relaxation Oscillator for a given frequency of 1KHZ. Relation between R1 and R2 is given R2 = 1.16 R1
Let R1= 10 kΩ. Then R2 = 11.6 kΩ.
To calculate R:
Formula to be used:

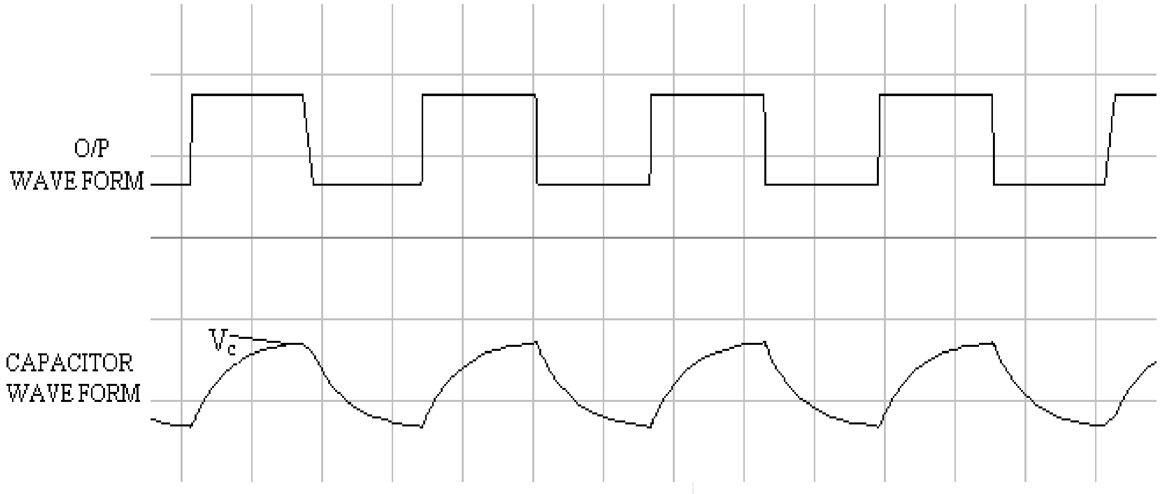
f0=1/2RC
C = 0.1µf then R= 5 kΩ

PROCEDURE :

1. Before making the connections check all the components using multimeter.

2. Make the connections as shown in figure and switch on the power supply.
3. Observe the voltage waveform across the capacitor on CRO.
4. Also observe the output waveform on CRO. Measure its amplitude and frequency.

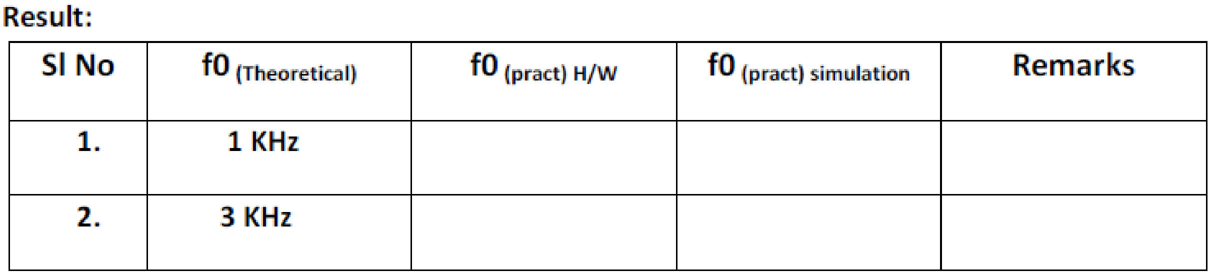
WAVEFORMS



THEORY:

Op-Amp Relaxation Oscillator is a simple Square wave generator which is also called as a Free running oscillator or Astable multivibrator or Relaxation oscillator. In this figure the op- amp operates in the saturation region. Here, a fraction (R2/(R1+R2)) of

output is fed back to the noninverting input terminal. Thus reference voltage is (R2/(R1+R2)) Vo. And may take values as +(R2/(R1+R2)) Vsat or - (R2/(R1+R2)) Vsat. The output is also fed back to the inverting input terminal after integrating by means of a low-pass RC combination. Thus whenever the voltage at inverting input terminal just exceeds reference voltage, switching takes place resulting in a square



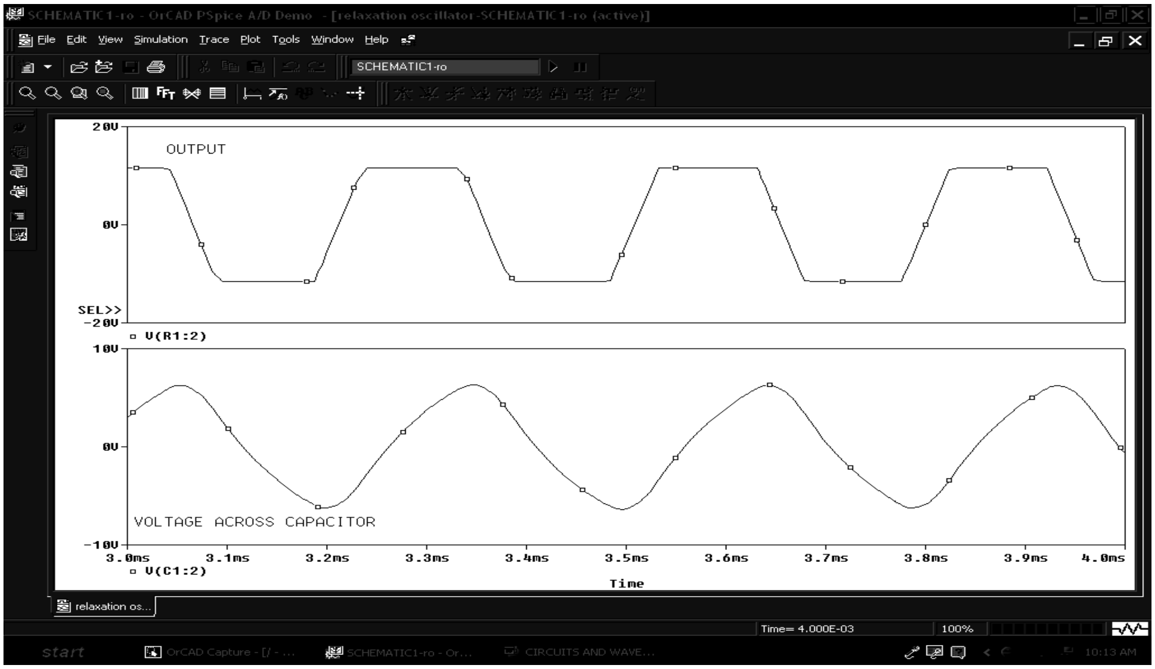
Result:

Sl No	f0 (Theoretical)	f0 (pract) H/W	f0 (pract) simulation	Remarks
1.	1 KHz			
2.	3 KHz			

Simulation part

Implement a rectangular waveform generator (Op-Amp relaxation oscillator) using a simulation

package and observe the change in frequency when all resistor values are doubled.



TYPE OF ANALYSIS : TIME DOMAIN RUN

TO TIME : 4ms MAXIMUM STEP

SIZE : 0.01ms

EXPERIMENT -3

SCHMITT TRIGGER

4.Using ua 741 opamap, design a window comparate for any given UTP and LTP. And simulate the same.

AIM : To design and implement a Schmitt trigger circuit using op-amp for the given UTP and LTP values.

COMPONENTS REQUIRED : IC µA 741, Resistor of 10KΩ, 100KΩ, DC regulated power supply, Signal generator and CRO.

CIRCUIT DIAGRAM:

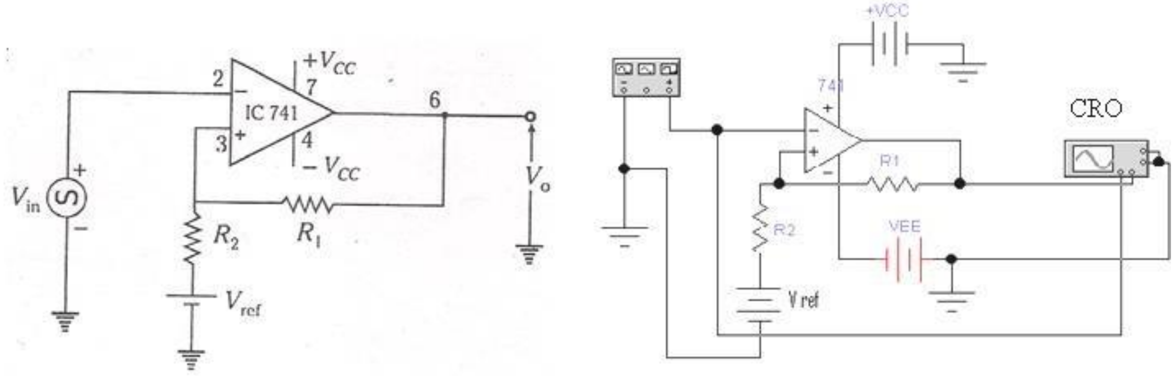


Fig 4: Circuit Diagram and actual connections of Schmitt Trigger Circuit

DESIGN:

From theory of Schmitt trigger circuit using op-amp, we have the trip points,

$$UTP = \frac{R_1 V_{ref}}{R_1 + R_2} + \frac{R_2 V_{sat}}{R_1 + R_2}$$ where V_{sat} is the positive saturation of the opamp = 100% of V_{cc}

$$\& \; LTP = \frac{R_1 V_{ref}}{R_1 + R_2} - \frac{R_2 V_{sat}}{R_1 + R_2}$$

Hence given the LTP & UTP values to find the $R_1, R_2 \,\&\, V_{ref}$ values, the following design is used.

$$UTP + LTP = \frac{2R_1 V_{ref}}{R_1 + R_2} \text{ - - - - - -(1)}$$

$$UTP - LTP = \frac{2R_1 V_{sat}}{R_1 + R_2} \text{ - - - - - -(2)}$$

Let $V_{sat} = 10\text{V}$, UTP = 4V & LTP = 2V, then equation (2) yields $R_1 = 10R_2$

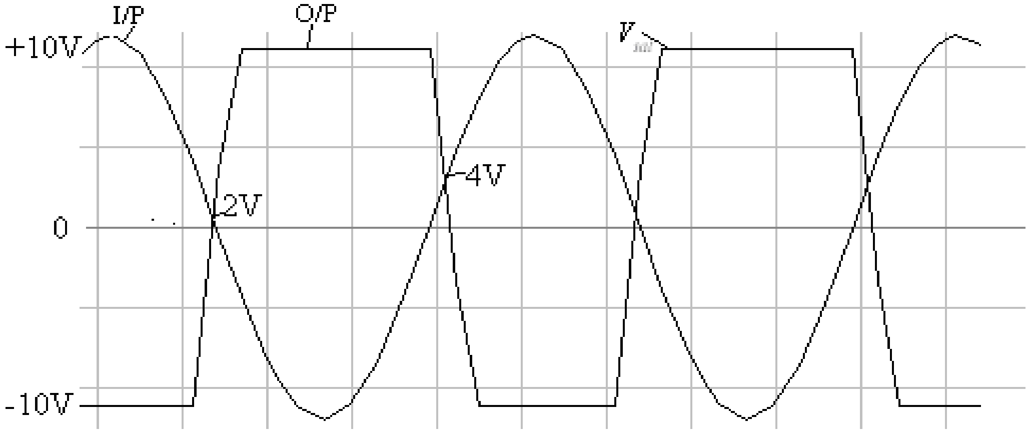
Let $R_2 = 10K\Omega$, then $R_1 = 100K\Omega$

From equation (1) we have V

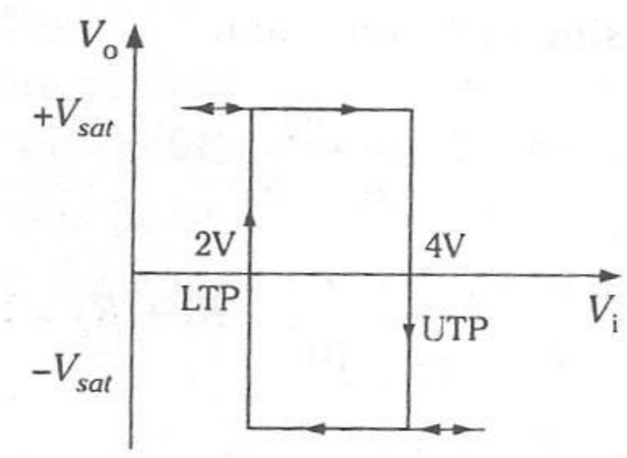
PROCEDURE :

1. Before doing the connections, check all the components using multimeter.
2. Make the connection as shown in circuit diagram.
3. Using a signal generator apply the sinusoidal input waveform of peak-to-peak amplitude of 15V, frequency 1kHz.
4. Keep the CRO in dual mode; apply input (V_{in}) signal to the channel 1 and observe the output (V_o) on channel 2 which is as shown in the waveform below. Note the amplitude levels from the waveforms.
5. Now keep CRO in X-Y mode and observe the hysteresis curve.

Waveforms:



CRO in DUAL mode



CRO in X-Y mode showing the Hysteresis curve

$$= \frac{(UTP + LTP)(R_1 + R_2)}{2R_1} = 3.33V$$

THEORY:

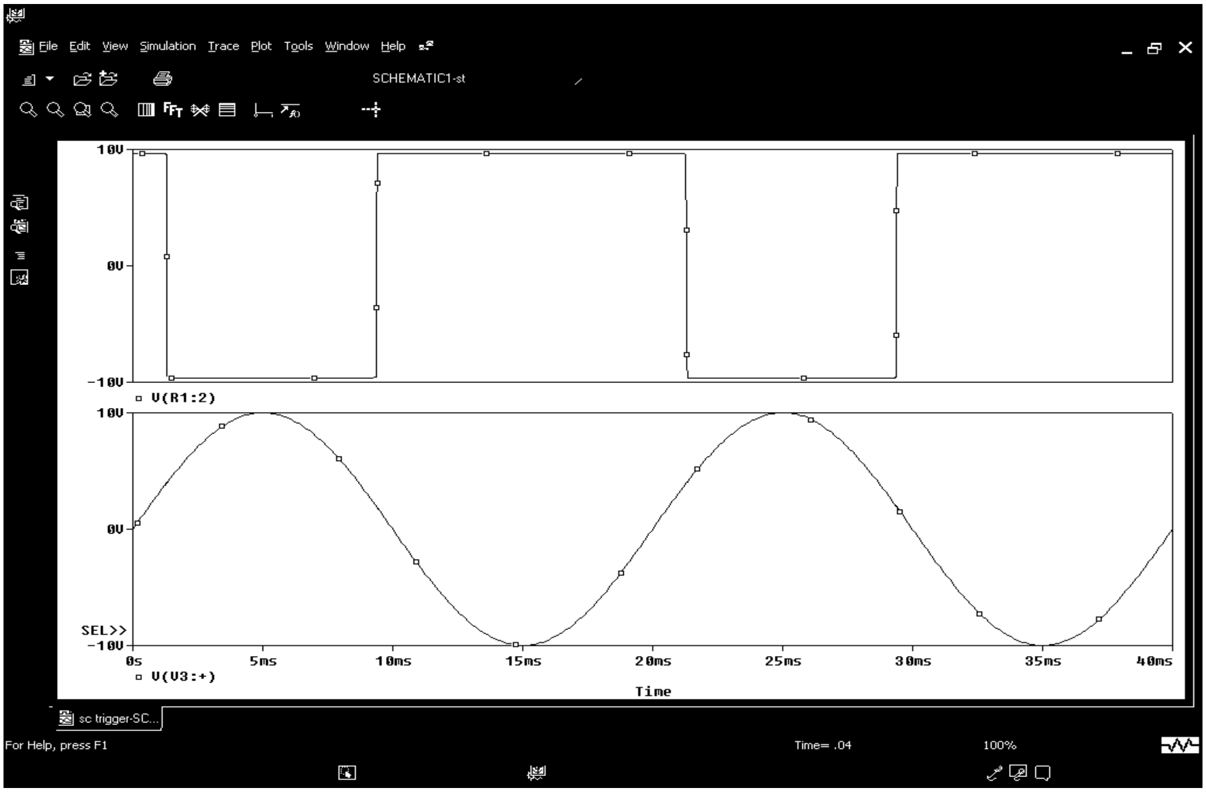
Schmitt Trigger converts an irregular shaped waveform to a square wave or pulse. Here, the input voltage triggers the output voltage every time it exceeds certain voltage levels called the upper threshold voltage V_{UTP} and lower threshold voltage V_{LTP}. The input voltage is applied to the inverting input. Because the feedback voltage is aiding the input voltage, the feedback is positive. A comparator using positive feedback is usually called a Schmitt Trigger. Schmitt Trigger is used as a squaring circuit, in digital circuitry, amplitude comparator, etc.

Result:

Sl. No	UTP(TH)	LTP(TH)	Hardware		Simulation	
			UTP(Prac)	LTP(Prac)	UTP(Prac)	LTP(Prac)
1.	+4	-4				
2.	+5	-5				
3.	+6	-2				

Simulation part

Design and Implement a Schmitt trigger using Op-Amp using a simulation package for two sets of UTP and LTP values.



Type of analysis: TIME DOMAIN (TRANSIENT) Run to

time: 40msec step size:0.1msec

PART-B EXPERIMENT

NO. 4

Design and implement half adder, full adder, half subtractor, full subtractor using basic gates.

AIM: Realise half adder, full adder, half subtractor, full subtractor using basic gates. **COMPONENTS REQUIRED: IC** 7408, 7432, 7404, patch chords, power chords and trainer kit.

THEORY:

An **adder**, also called summer, is a digital circuit that performs addition of numbers.

1. Half Adder:

It is a combinational circuit that performs the addition of two bits; this circuit needs two binary inputs and two binary outputs, with one producing sum output and other produce carry output. The half-adder is useful to add one binary digit quantities.

2. Full adder:

This type of adder is a little more difficult to implement than a half-adder. The main difference between a half-adder and a full-adder is that the full-adder has three inputs and two outputs.
The first two inputs are A and B and the third input is an input carry designated as CIN. The output carry is designated as COUT and the normal output Sum is designated as S.

3. Half Subtractor:

A half Subtractor is a multiple output combinational logic network that does the subtraction of two bits of binary data. It has input variables and two output variables. Two inputs are corresponding to two input bits and two output variables correspond to the difference bit and borrow bit.

4.Full subtractor:

The full subtractor is a combinational circuit which is used to perform subtraction of three bits the minuend A, subtrahend B with a carry-in C. It produces two outputs the Sum and the Carry our to the next higher stage. Here we elect to translate the truth table to two Sum of Products expressions and implement the Sum and Carry using AND-OR logic.

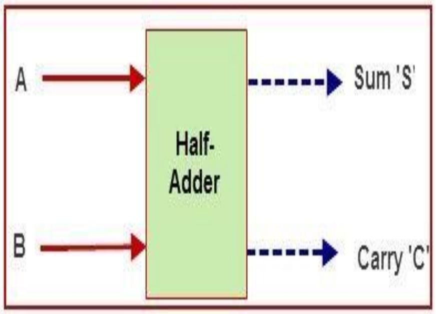
PROCEDURE:

1. Verify all the components and the patch chords whether they are in good condition or not.

2. Make connections as shown in the circuit diagram.
3. Give power supply to the trainer kit.
4. Provide input data to the circuit via switches.
5. Record and verify the output sequence for each combination of the select lines.

HALF ADDER:

Block Diagram



Truth Table

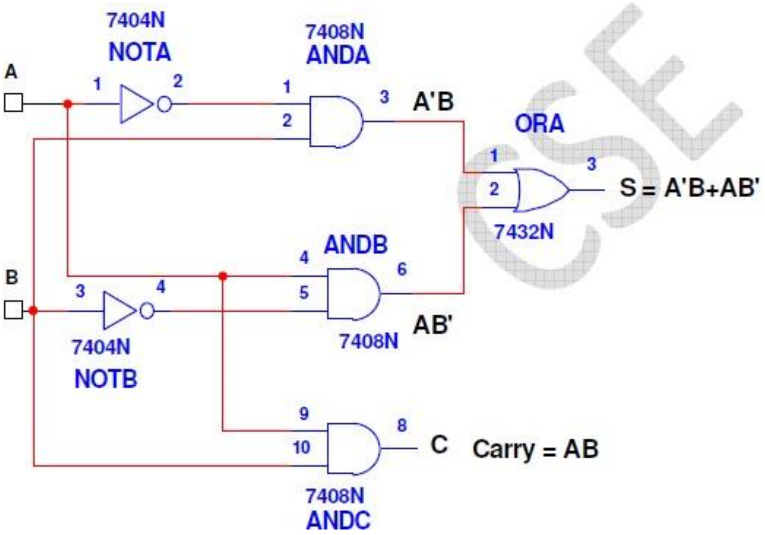
INPUTS		OUTPUTS	
A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

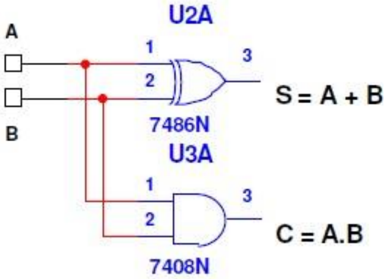
From the above truth table of Half adder:

The simplified Boolean function for Sum is: S = A‘.B + A.B‘ and Carry is C = A.B. The circuit diagram is as shown in Fig (A) Using basic gates.

By simplifying the above expression of sum and carry using De Morgan‘s Law we get: Sum = A ⊕ B and Carry = A.B. It is shown in Fig (B)

CIRCUIT DIAGRAM

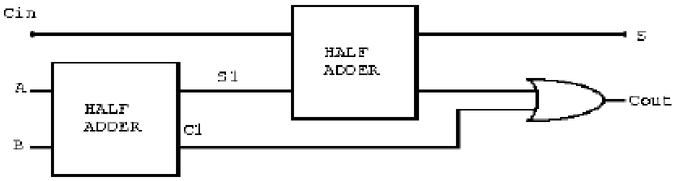


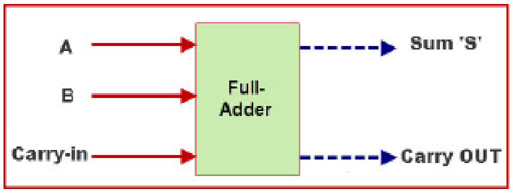


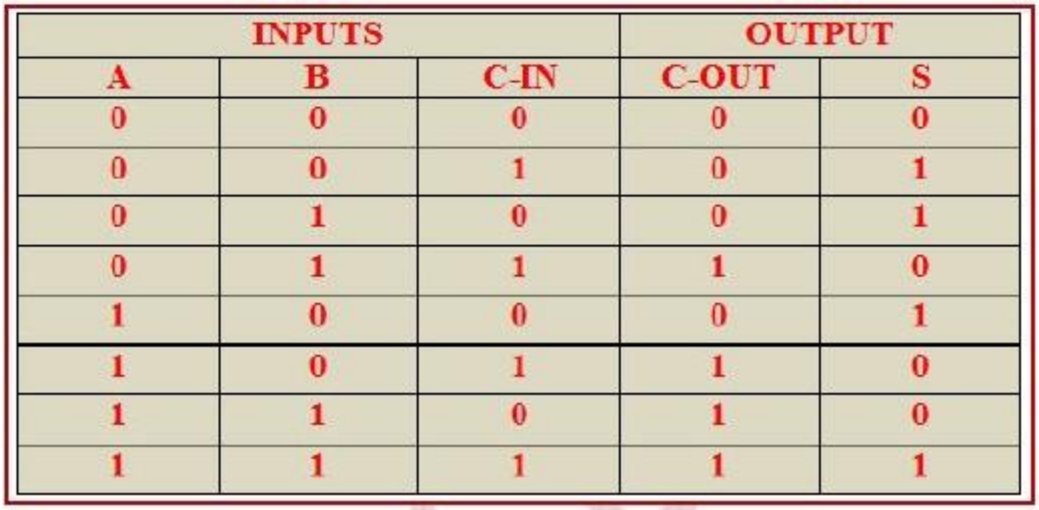
Fig(A) **Fig(B)**

FULL ADDER:

Full Adder Using Two Half Adders: **Full Adder Block Diagram**





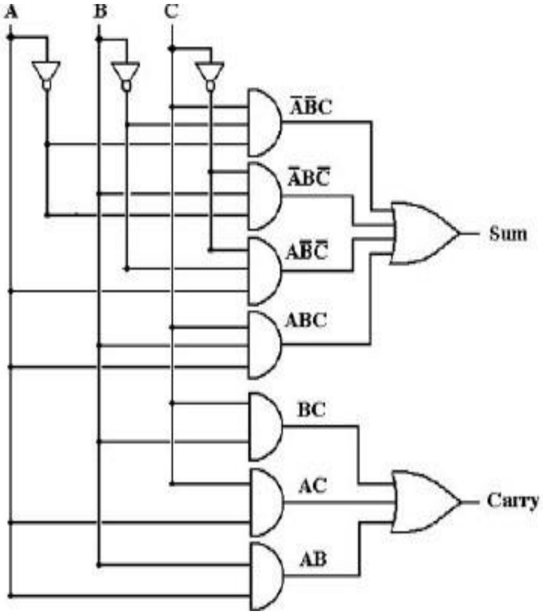


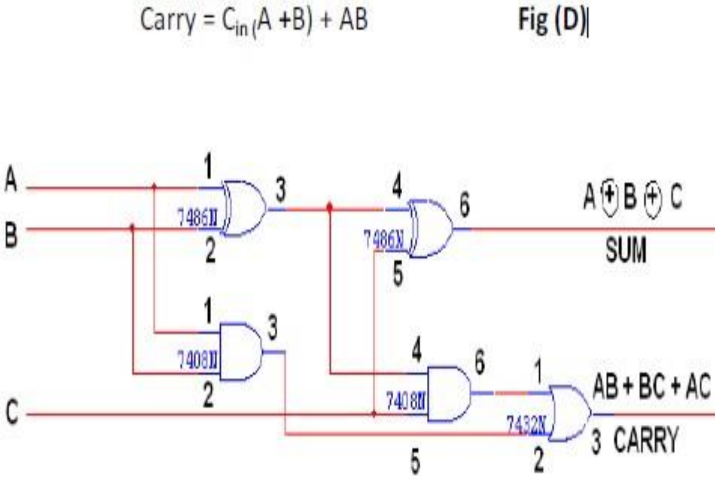
INPUTS			OUTPUT	
A	B	C-IN	C-OUT	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Fig: Truth table

The simplified Boolean function from the truth table using SoP method is: S =
A‘B‘C + A‘BC‘ + ABC‘ + ABC **Fig (C)**
The simplified Boolean function from the truth table Using SoP is C out
=AB + B Cin+ Cin A **Fig (C)**
By simplifying the above expression using DeMorgan‘s Law we get: Sum = A
$\oplus$ B $\oplus$ C and
Carry = A.B + Cin (A +B) **Fig (D)**

Here is one design of a Full-Adder using the basic Boolean gates and truth table.

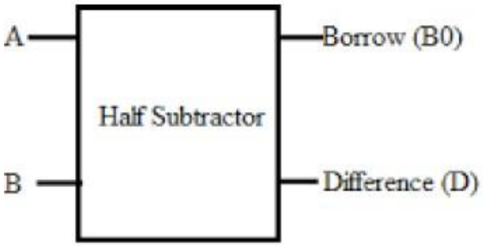




Fig(C) **Fig(D**

HALF SUBTRACTOR:

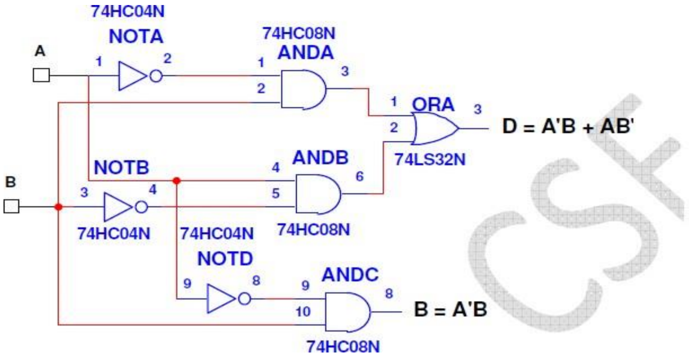
Half Subtractor:

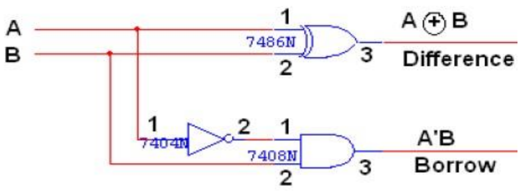


Truth Table

A	B	D	Bo
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

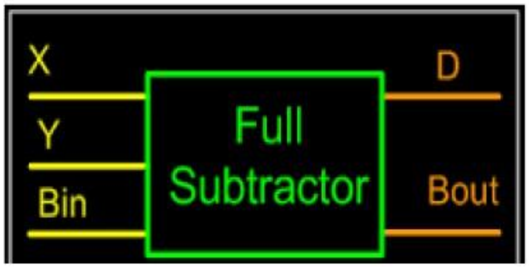
From the above truth table:
The expression for Difference using SOP is: A‘B + AB‘ and Borrow is: A‘B and the circuit Using Basic Gates is as shown in Fig.



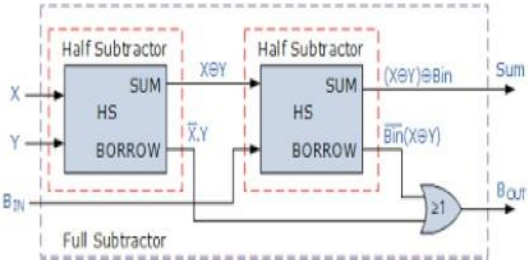


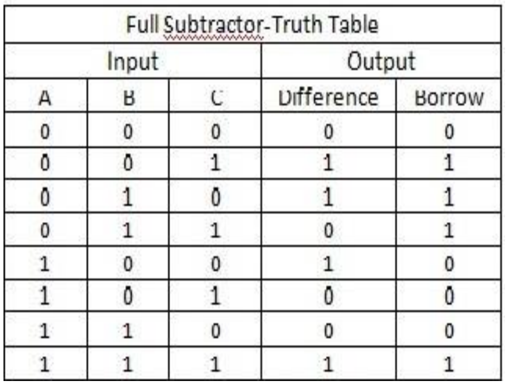
FULL SUBTRACTOR:

Full Subtractor:



Full Subtractor using two Half Subtractor



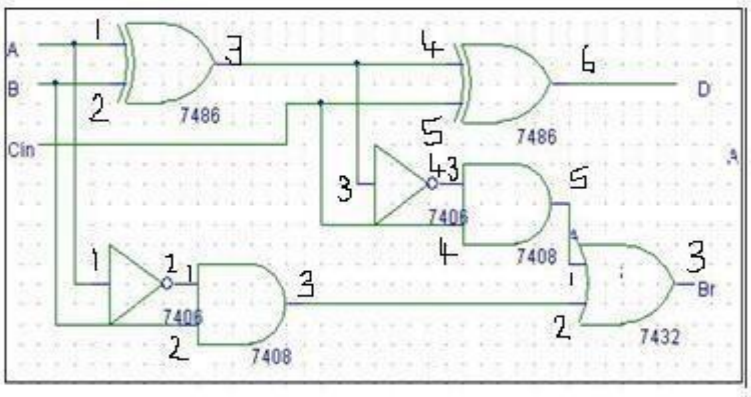


Full Subtractor-Truth Table				
Input			Output	
A	B	C	Difference	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

D= A ⊕ B ⊕ C

Br= $\overline{A}$ **B + B Cin +** $\overline{A}$ **Cin**

Circuit diagram:



Results:

1. Truth table verified for :

2. Truth table verified for :

3. Truth table verified for :

4. Truth table verified for :

EXPERIMENT NO. 5

5 Given a 4-variable logic expression, simplify it using appropriate technique and realize the simplified logic expression using 8:1 multiplexer IC. And implement the same in HDL.

AIM: To simplify the Boolean expression using Entered Variable map method and realize the simplified logic using 8:1 MUX.

COMPONENTS REQUIRED:IC 74151/IC74153,IC7404.IC7432, patch chords, power chords and trainer kit.

THEORY

Multiplexer sometimes is called universal logic circuit because a 2n to 1 multiplexer can be used as a design solution for any n variable truth table. Let‘s consider A B and C variables to be fed as select inputs, the fourth D as data input. We write all the combinations of 3 select inputs in first row along different columns. Now corresponding to each value of 4th variable D truth table output Y is written.The 4th column Y as a function of D.

PROCEDURE:

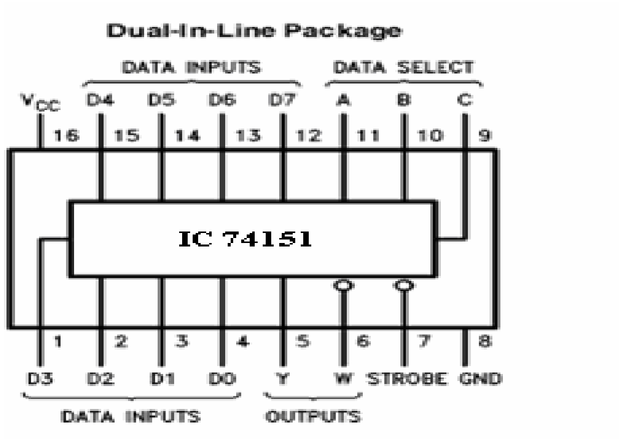
1 Verify all the components and the patch chords whether they are in good condition or not. 2 Make connections as shown in the circuit diagram.

3 Give power supply to the trainer kit.

4 Provide input data to the circuit via switches.

5 Record and verify the output sequence for each combination of the select lines.

PIN DIAGRAMS:



x: *E*

Simplify the function using MEV Technique f(A,B,C,D)=∑M(2,3,4,5,12,14) Then

$Y=\overline{AB}C\overline{D}+\overline{AB}CD+ \overline{A}B\overline{CD}+\overline{A}B\overline{C}D+ AB\overline{CD}+ABC\ \overline{D}$

A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

By Entered Variable MAP:

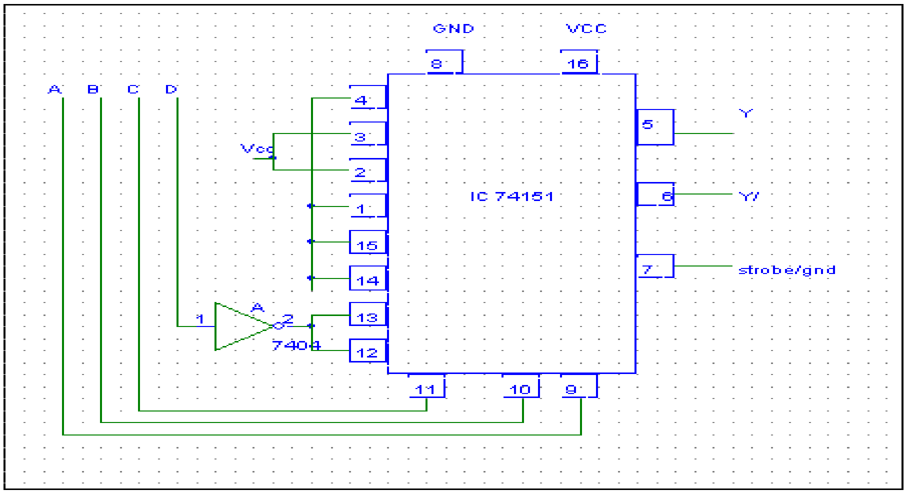
A	B	C	Y	Data Inputs
0	0	0	0	D0
0	0	1	1	D1
0	1	0	1	D2
0	1	1	0	D3
1	0	0	0	D4
1	0	1	0	D5
1	1	0	D	D6
1	1	1	D	D7

Simplification by K-Map: Y = ABC + ABC + ABD

	CD	CD	CD	CD
AB	0	0	1	1
AB	1	1	0	0
AB	1	0	0	1
AB	0	0	0	0

CIRCUIT DIAGRAM:

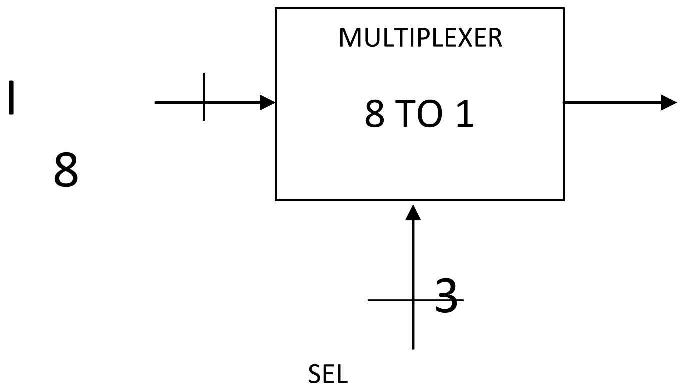
Simplify the given 4 variable Boolean expression and simplify it using 8:1 multiplexer.



RESULT: All the entries in the truth table are verified.

Simulation using VHDL

Write the Verilog/VHDL code for an 8:1 multiplexer. Simulate and verify its working.



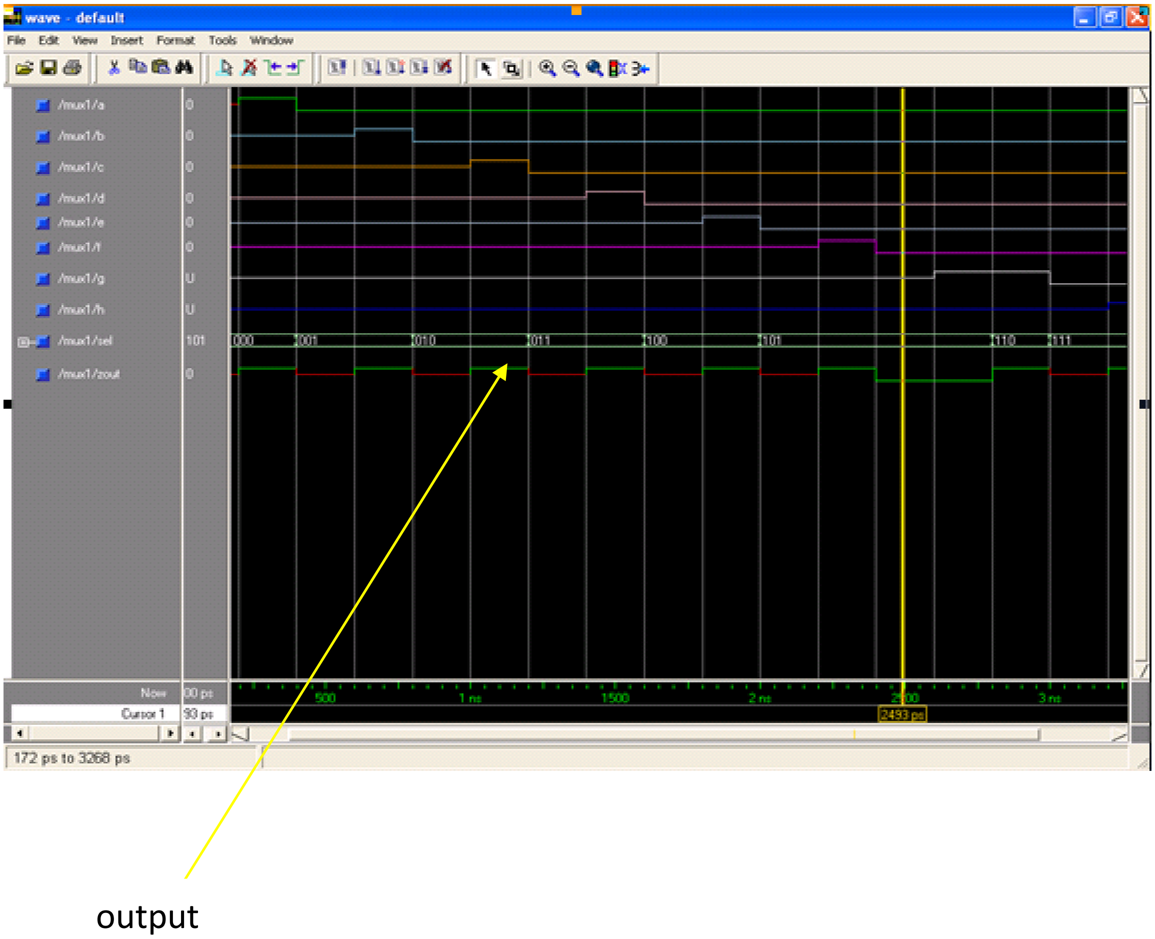
TruthTable

INPUTS			OUTPUTS
SEL (2)	SEL (1)	SEL (0)	Zout
0	0	0	I(0)
0	0	1	I(1)
0	1	0	I(2)
0	1	1	I(3)
1	0	0	I(4)
1	0	1	I(5)
0	1	1	I(6)
1	1	1	I(7)

```
-- VHDL code for 8 to 1 mux (Behavioral modeling). library

IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity mux1 is
   Port ( I : in std_logic_vector(7 down to 0); sel : in
        std_logic_vector(2 downto 0); zout : out
        std_logic);
end mux1;
architecture Behavioral of mux1 is

begin
  zout<=    I(0) when sel="000" else
```

```
            I(1)  when  sel="001"  else  I(2)
            when  sel="010"  else  I(3)  when
            sel="011"    else    I(4)    when
            sel="100"    else    I(5)    when
            sel="101"    else    I(6)    when
            sel="110" else I(7);

end Behavioral;
```

8:1 Mux Simulation Results



EXPERIMENT-6

6.Realize a J-K Master / Slave Flip-Flop using NAND gates and verify its truth table. And implement the same in HDL.

AIM: To realize a J-K Master/Slave FF using NAND gates and verifies its truth table

COMPONENTS REQUIRED: IC7410 three input NAND gate, IC7400 two input NAND gate, IC7404 NOT gate, patch chords, power chords and trainer kit

THEORY: The circuit below shows the solution. To the RS flip-flop we have added two new connections from the Q and Q' outputs back to the original input gates. Remember that a NAND gate may have any number of inputs, so this causes no trouble. To show that we have done this, we change the designations of the logic inputs and of the flip-flop itself. The inputs are now designated J (instead of S) and K (instead of R). The entire circuit is known as a *JK flip-flop*. In most ways, the JK flip-flop behaves just like the RS flip-flop. The Q and Q' outputs will only change state on the falling edge of the CLK signal, and the J and K inputs will control the future output state pretty much as before. However, there are some important differences.

Since one of the two logic inputs is always disabled according to the output state of the overall flip-flop, the master latch cannot change state back and forth while the CLK input is at logic 1. Instead, the enabled input can change the state of the master latch *once*, after which this latch will not change again. This was not true of the RS flip-flop.

If both the J and K inputs are held at logic 1 and the CLK signal continues to change, the Q and Q' outputs will simply change state with each falling edge of the CLK signal. (The master latch circuit will change state with each *rising* edge of CLK.) We can use this characteristic to advantage in a number of ways. A flip-flop built specifically to operate this way is typically designated as a *T* (for *Toggle*) flip-flop. The lone T input is in fact the CLK input for other types of flip-flops.

The JK flip-flop *must* be edge triggered in this manner. Any level-triggered JK latch circuit will oscillate rapidly if all three inputs are held at logic 1. This is not very useful. For the same reason, the T flip-flop must also be edge triggered. For both types, this is the only way to ensure that the flip-flop will change state only once on any given clock pulse.

Because the behavior of the JK flip-flop is completely predictable under all conditions, this is the preferred type of flip-flop for most logic circuit designs. The RS flip-flop is only used in applications where it can be guaranteed that both R and S cannot be logic 1 at the same time.

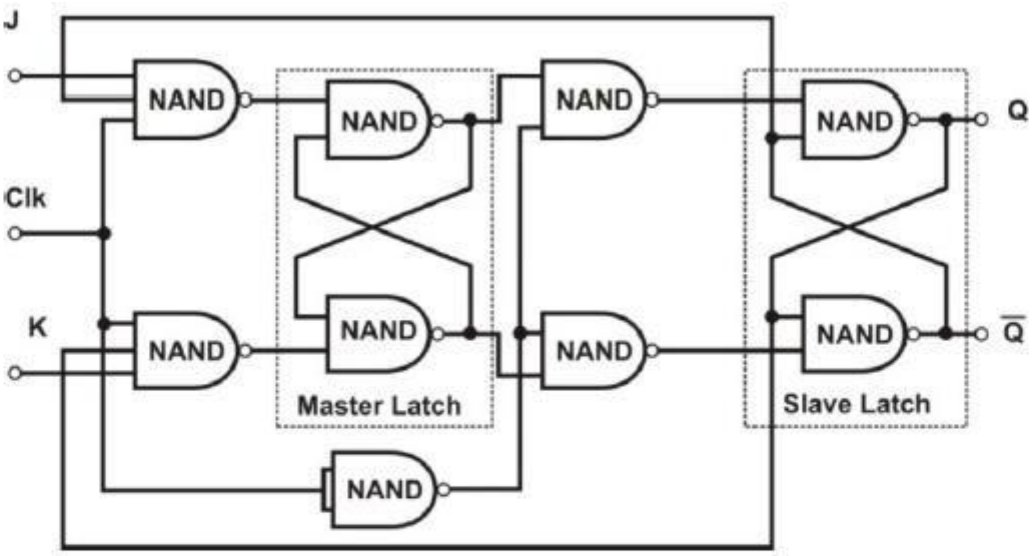
At the same time, there are some additional useful configurations of both latches and flip- flops. In the next pages, we will look first at the major configurations and note their properties. Then we will see how multiple flip-flops or latches can be combined to perform useful functions and operations.

Master Slave Flip Flop:

The control inputs to a clocked flip flop will be making a transition at approximately the same times as triggering edge of the clock input occurs. This can lead to unpredictable triggering.

A JK master flip flop is positive edge triggered, where as slave is negative edge triggered. Therefore master first responds to J and K inputs and then slave. If J=0 and K=1, master resets on arrival of positive clock edge. High output of the master drives the K input of the slave. For the trailing edge of the clock pulse the slave is forced to reset. If both the inputs are high, it changes the state or toggles on the arrival of the positive clock edge and the slave toggles on the negative clock edge. The slave does exactly what the master

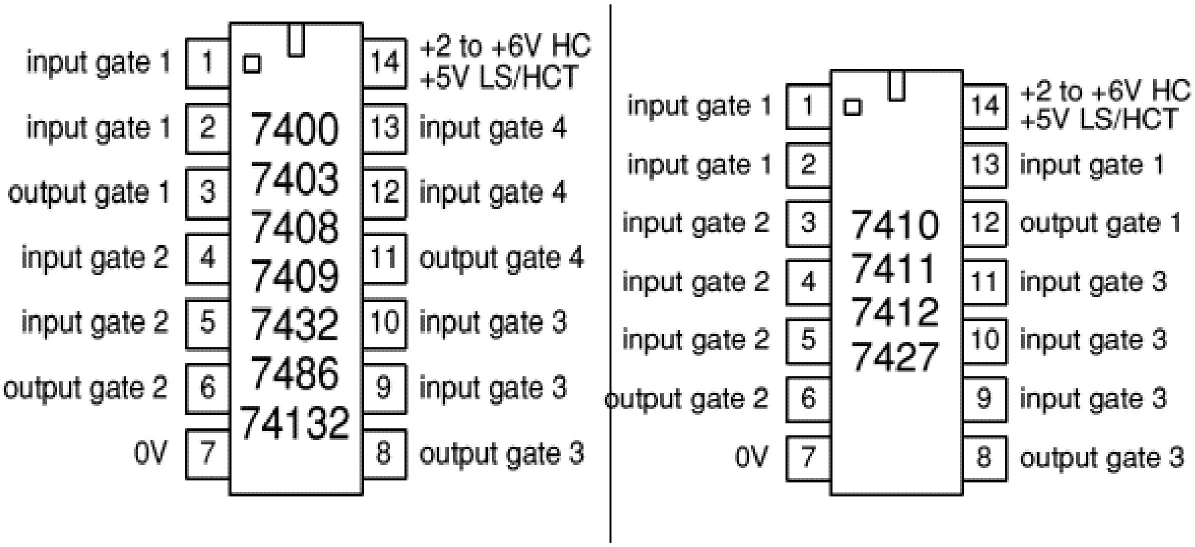
[redacted]



PROCEDURE:

1. Verify all the components and the patch chords whether they are in good condition or not.
2. Make connections as shown in the circuit diagram.
3. Give power supply to the trainer kit.
4. Provide input data to the circuit via switches.
5. Record and verify the output sequence for each combination of the select lines.

PIN DIAGRAMS:

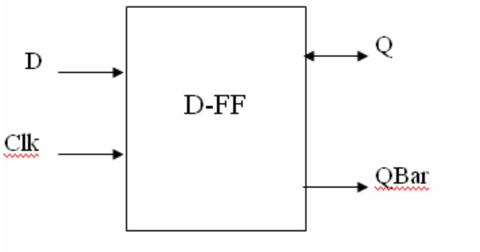


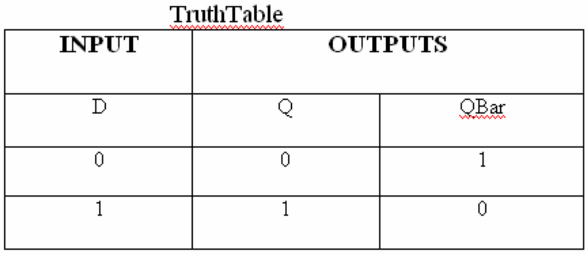
Clk	J	K	Q	$\overline{Q}$	comment
⎍	0	0	Q_0	$\overline{Q_0}$	No change
⎍	0	1	0	1	Reset
⎍	1	0	1	0	Set
⎍	1	1	Q_0	Q_0	toggle

RESULT: All the entries in the truth table are verified.

Simulation using VHDL

Write the Verilog/VHDL code for D Flip-Flop with positive-edge triggering. Simulate and verify its working



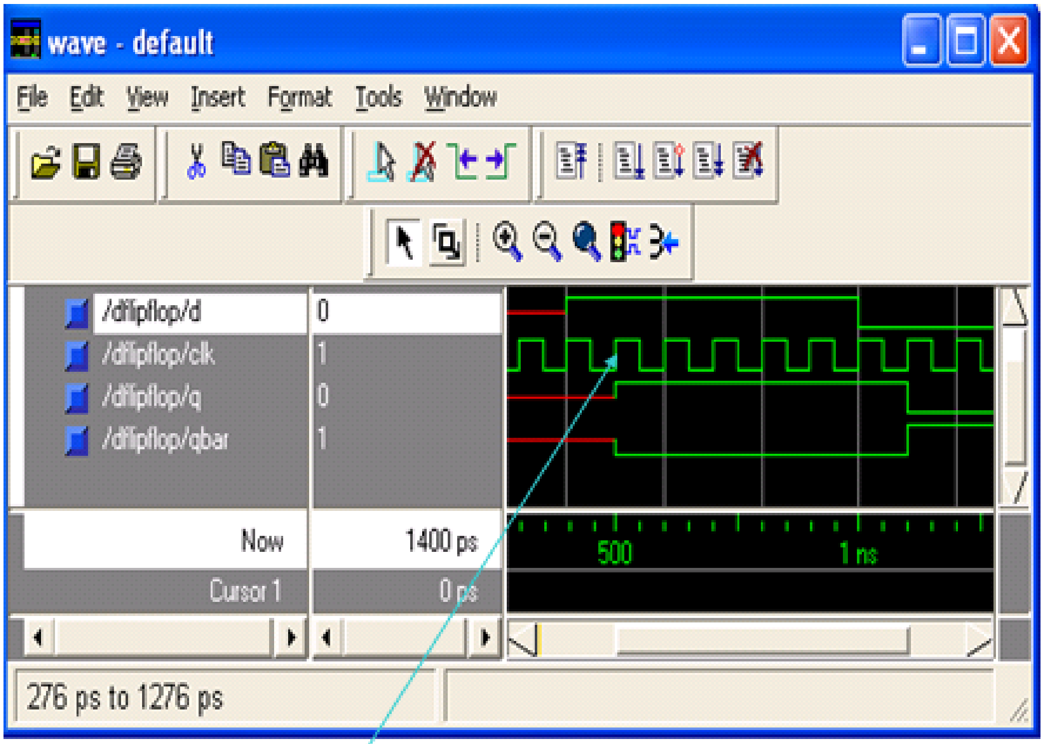


TruthTable

INPUT	OUTPUTS	
D	Q	QBar
0	0	1
1	1	0

```
VHDL code for D Flip Flop Counter. library
IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity dflipflop is
    Port ( D,Clk : in std_logic; Q :
           inout std_logic; Qbar : out
           std_logic);
end dflipflop;

architecture Behavioral of dflipflop is begin
 process(clk)
 begin
 if rising_edge(clk) then Q<=
 D;
 end if;
 end process;
 Qbar<= not Q;

end Behavioral;
```

D-FF Simulation Results



/dflipflop/d 0
/dflipflop/clk 1
/dflipflop/q 0
/dflipflop/qbar 1

Now 1400 ps
Cursor 1 0 ps

276 ps to 1276 ps

Positive edge triggered
(Output changes when clock signal goes low to high)

EXPERIMENT-7

7.Design and implement code converter I)Binary to Gray (II) Gray to Binary Code using basic gates.

AIM:

COMPONENTS REQUIRED:

THEORY: The logical circuit which converts binary code to equivalent gray code is known as binary to gray code converter. The gray code is a non weighted code. The successive gray code differs in one bit position only that means it is a unit distance code. It is also referred as cyclic code. It is not suitable for arithmetic operations. It is the most popular of the unit distance codes. It is also a reflective code. An n-bit Gray code can be obtained by reflecting an n-1 bit code about an axis after 2^{n-1} rows, and putting the MSB of 0 above the axis and the MSB of 1 below the axis.

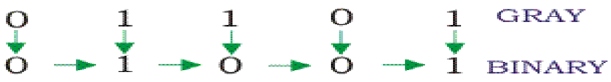
Steps To Convert Grey to Binary Code:

Following steps can make your idea clear on this type of conversions.

(1) The M.S.B of the binary number will be equal to the M.S.B of the given gray code.

(2) Now if the second gray bit is 0 the second binary bit will be same as the previous or the first bit. If the gray bit is 1 the second binary bit will alter. If it was 1 it will be 0 and if it was 0 it will be 1.

(3) This step is continued for all the bits to do Gray code to binary conversion.



In the above example the M.S.B of the binary will be 0 as the M.S.B of gray is 0. Now move to the next gray bit. As it is 1 the previous binary bit will alter i.e. it will be 1, thus the second binary bit will be 1.Next look at the third bit of the gray code. It is again 1 thus the previous bit i.e the second binary bit will again alter and the third bit of the binary number will be 0. Now, 4th bit of the given gray is 0 so the previous binary bit will be unchanged, i.e. 4th binary bit will be 0. Now again the 5th grey bit is 1 thus the previous binary bit will alter, it will be 1 from 0. Therefore the equivalent Binary number in case of gray code to binary conversion will be (01001)

Steps To Convert Binary to Gray Code:

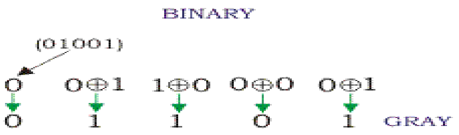
Steps given below elaborate on the idea on this type of conversion.

(1) The M.S.B. of the gray code will be exactly equal to the first bit of the given binary number.

(2) Now the second bit of the code will be exclusive-or of the first and second bit of the given binary Number, i.e if both the bits are same the result will be 0 and if they are different the result will be 1.

(3)The third bit of gray code will be equal to the exclusive-or of the second and third bit of the given Binary number.

One example given below can make your idea clear on this type of conversion. Now concentrate on the example where the M.S.B. of the binary is 0 so for it will be 0 for the most significant gray bit. Next, the XOR of the first and the second bit is done. The bits are different so the resultant gray bit will be 1. Again move to the next step, XOR of second and third bit is again 1 as they are different. Next, XOR of third and fourth bit is 0 as both the bits are same. Lastly the XOR of fourth and fifth bit is 1 as they are different. That is how the result of binary to gray code conversion of 01001 is done whose equivalent gray code is 01101.



PROCEDURE:

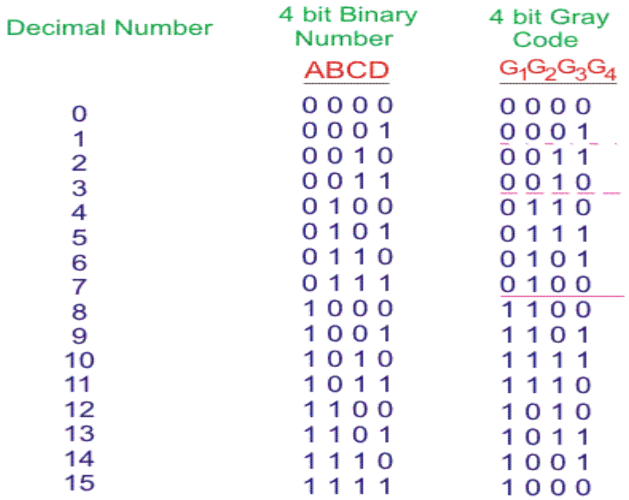
1 Verify all the components and the patch chords whether they are in good condition or not. 2
Make connections as shown in the circuit diagram.
3 Give power supply to the trainer kit.

4 Provide input data to the circuit via switches.

5 Record and verify the output sequence for each combination of the select lines.

Binary to Gray Code Converter

The 4 bits binary to gray code conversion table is given below,



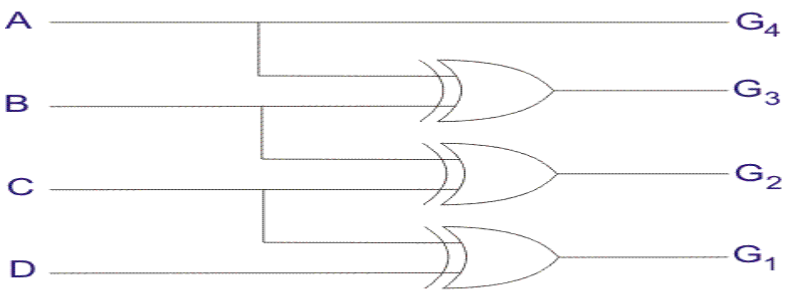
Decimal Number	4 bit Binary Number	4 bit Gray Code
	ABCD	$G_1G_2G_3G_4$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

That means, in 4 bit gray code, (4-1) or 3 bit code is reflected against the axis drawn after $(2^{4-1})^{th}$ or 8^{th} row. The bits of 4 bit gray code are considered as $G_4G_3G_2G_1$. Now from conversion table,

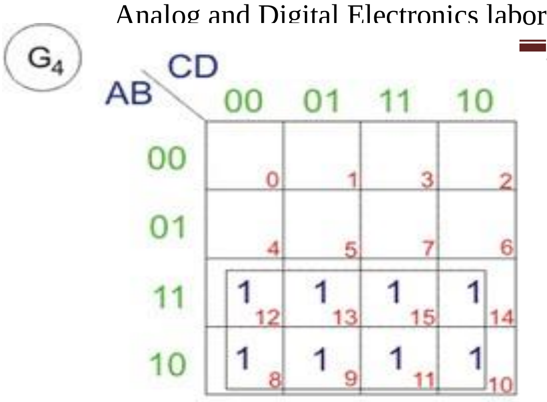
$$G4 = \sum m\ (8, 9, 10, 11, 12, 13, 14, 15)$$
$$G3 = \sum m\ (4, 5, 6, 7, 8, 9, 10, 11)$$
$$G2 = \sum m\ (2, 3, 4, 5, 10, 11, 12, 13)$$
$$G1 = \sum m\ (1, 2, 5, 6, 9, 10, 13, 14)$$

B3 = G3
B2 = G3 ⊕ G2 B1 =
B2 ⊕ G1
= (G3 ⊕ G2) ⊕ G1 B0 =
B1 ⊕G0
= (G3 ⊕G2) ⊕ (G1 ⊕ G0)

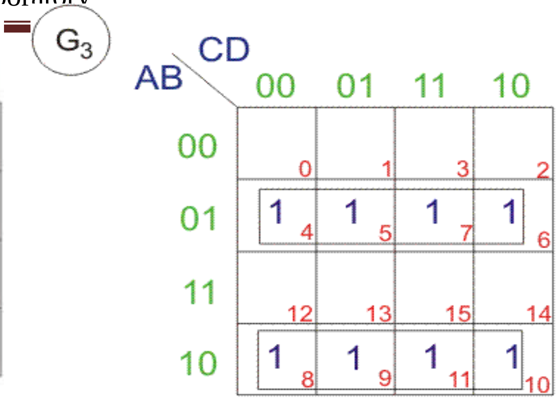
CIRCUIT DIAGRAM:



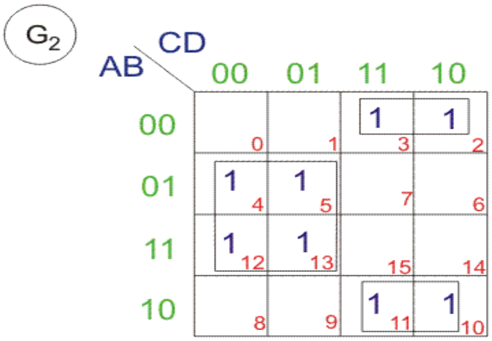
Logic Circuit for Binary to Gray Code Converter



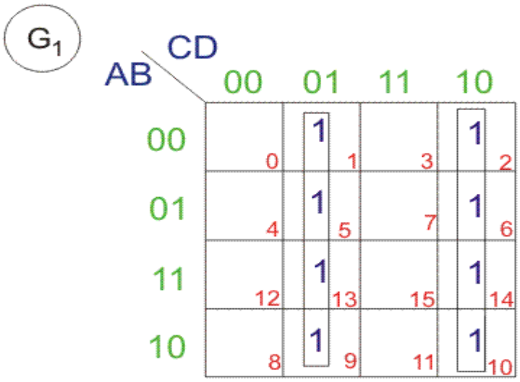
$G_4 = A$



$G_3 = \bar{A}B + A\bar{B} = A \oplus B$



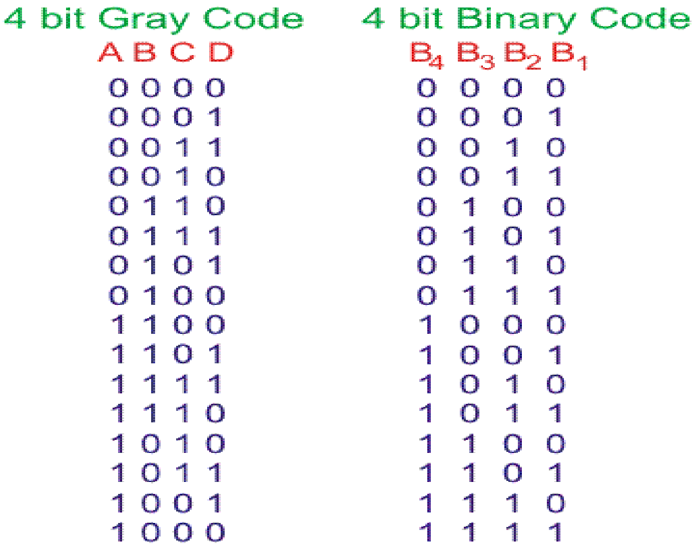
$G_2 = B\bar{C} + \bar{B}C = B \oplus C$



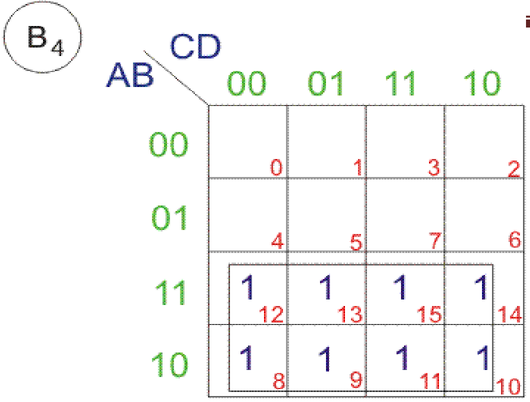
$G_1 = \bar{C}D + C\bar{D} = C \oplus D$

Gray to Binary Code Converter

Let us consider a 4 bit gray to binary code converter. To design a 4 bit gray to binary code converter, we first have to draw a conversion table.

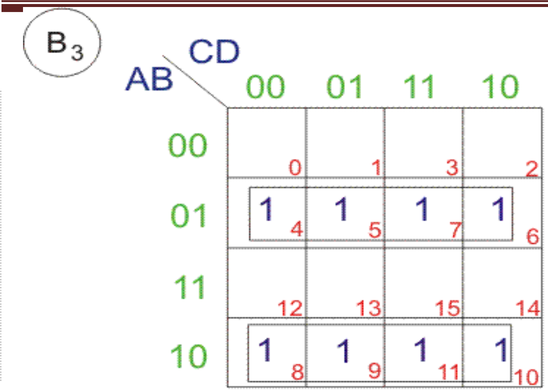


4 bit Gray Code A B C D	4 bit Binary Code B_4 B_3 B_2 B_1
0 0 0 0	0 0 0 0
0 0 0 1	0 0 0 1
0 0 1 1	0 0 1 0
0 0 1 0	0 0 1 1
0 1 1 0	0 1 0 0
0 1 1 1	0 1 0 1
0 1 0 1	0 1 1 0
0 1 0 0	0 1 1 1
1 1 0 0	1 0 0 0
1 1 0 1	1 0 0 1
1 1 1 1	1 0 1 0
1 1 1 0	1 0 1 1
1 0 1 0	1 1 0 0
1 0 1 1	1 1 0 1
1 0 0 1	1 1 1 0
1 0 0 0	1 1 1 1



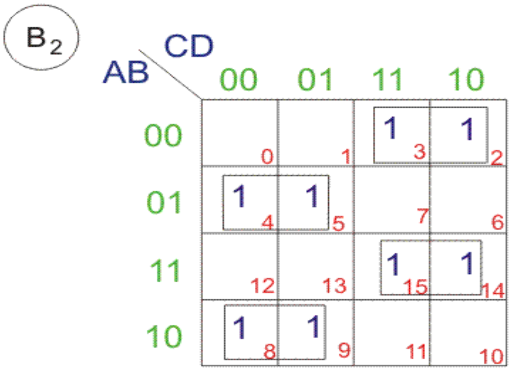
AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	1 12	1 13	1 15	1 14
10	1 8	1 9	1 11	1 10

$B_4 = A$

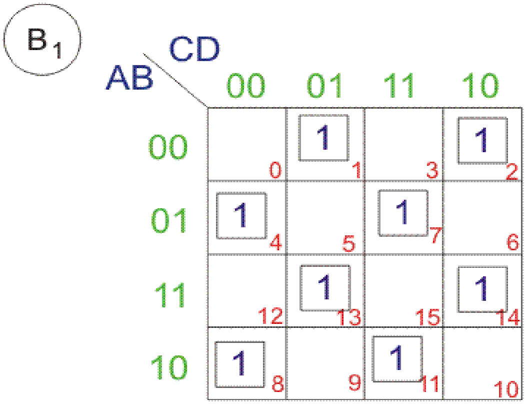


AB \ CD	00	01	11	10
00	0	1	3	2
01	1 4	1 5	1 7	1 6
11	12	13	15	14
10	1 8	1 9	1 11	1 10

$B_3 = \overline{A}B + A\overline{B} = A \oplus B$



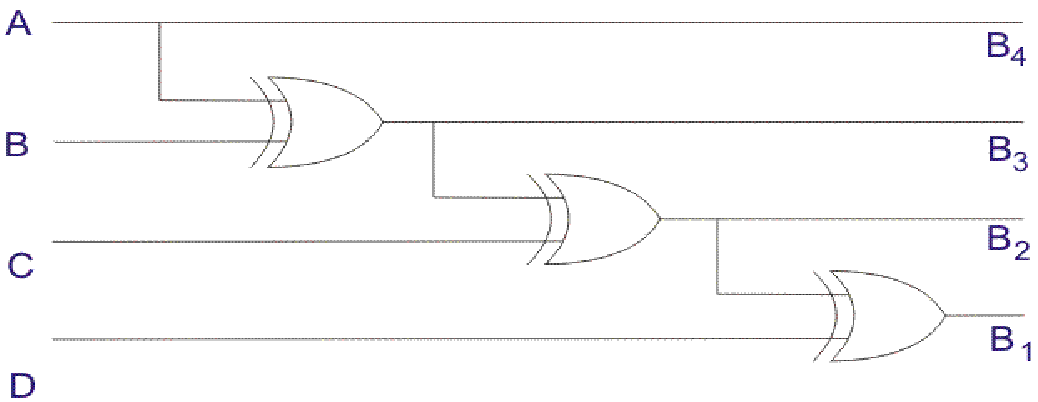
AB \ CD	00	01	11	10
00	0	1	1 3	1 2
01	1 4	1 5	7	6
11	12	13	1 15	1 14
10	1 8	1 9	11	10



AB \ CD	00	01	11	10
00	0	1 1	3	1 2
01	1 4	5	1 7	6
11	12	1 13	15	1 14
10	1 8	9	1 11	10

$$\begin{aligned} B_2 &= \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + ABC \\ &= A(\overline{B}\,\overline{C} + BC) + \overline{A}(B\overline{C} + \overline{B}C) \\ &= A(\overline{B\overline{C} + \overline{B}C}) + \overline{A}(B\overline{C} + \overline{B}C) \\ &= A(\overline{B \oplus C}) + \overline{A}(B \oplus C) = A \oplus B \oplus C \end{aligned}$$

$$B_1 = \overline{A}\,\overline{B}\,\overline{C}\,D + \overline{A}\,\overline{B}\,C\,\overline{D} + \overline{A}\,B\,\overline{C}\,\overline{D} + \overline{A}\,BCD + AB\overline{C}\,D + ABC\overline{D} + A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}CD \;=\; A \oplus B \oplus C \oplus D$$



Logic Circuit for Gray to Binary Code Converter

Results:

1. Truth Table verified for:

2. Truth Table verified for:

EXPERIMENT-8

Design and implement a mod-n (n<8) synchronous up counter using J-K Flip-Flop ICs and demonstrate its working.

AIM: To design and implement a mod n (a<8) synchronous up counter using JK FF

COMPONENTS REQUIRED: IC 7476, IC7408, patch chords, power chords and trainer kit

Theory:

In digital logic and computing, a counter is a device which stores (and sometimes displays) the number of times a particular event or process has occurred, often in relationship to a clock signal.

A synchronous counter is one whose output bits change sate simultaneously. Such a counter circuit can be built from JK flip-flop by connecting all the clock inputs together, so that each and every flipflop receives the exact same clock pulse at the exact same time.

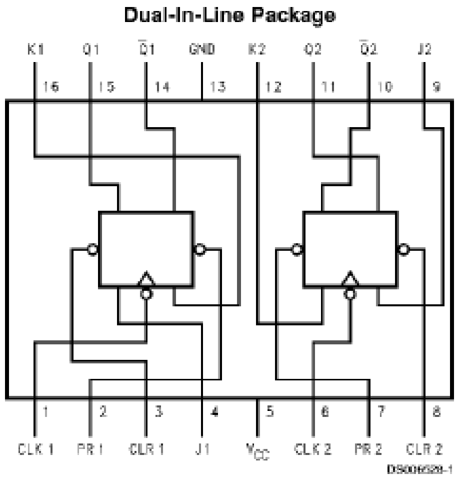
By examining the four-bit binary count sequence, it noticed that just before a bit toggles, all preceding bits are "high". That is a synchronous up-counter can be implemented by toggling the bit when all of the less significant bits are at a logic high state. For example, bit 1 toggles when bit 0 is logic high; bit 2 toggles when both bit 1 and bit 0 are logic high; bit 3 toggles when bit 2, bit 1 and bit 0 are all high; and so on.

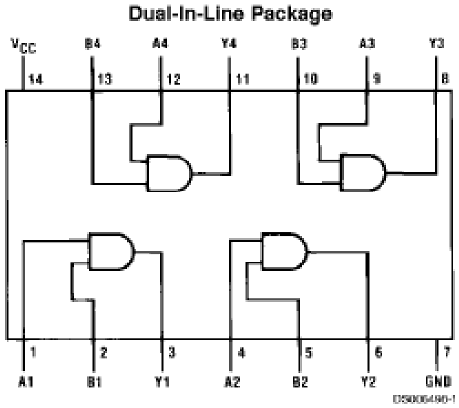
The ripple counter requires a finite amount of time for each flip flop to change state. This problem can be solved by using a synchronous parallel counter where every flip flop is triggered in synchronism with the clock, and all the output which are scheduled to change do so simultaneously.

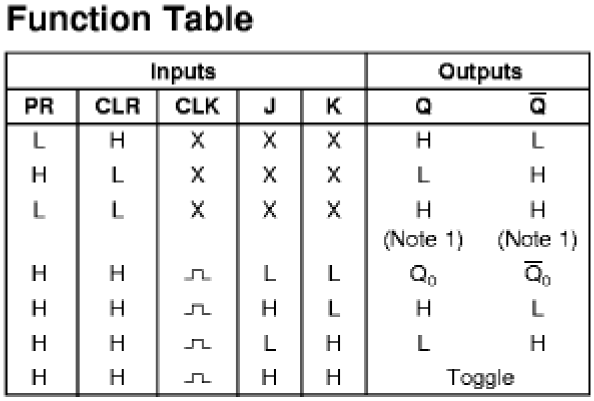
The counter progresses counting upwards in a natural binary sequence from count 000 to count 100 advancing count with every negative clock transition and get back to 000 after this cycle.

Circuit Diagram:

7476 **7408**



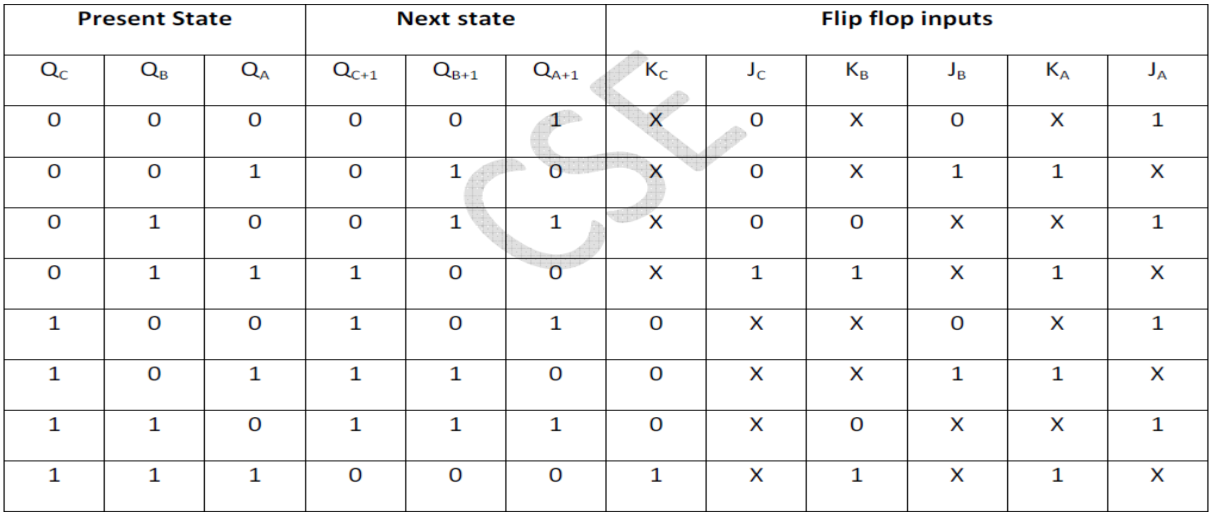




Function Table

Inputs					Outputs	
PR	CLR	CLK	J	K	Q	$\overline{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H (Note 1)	H (Note 1)
H	H	⎍	L	L	Q_0	$\overline{Q}_0$
H	H	⎍	H	L	H	L
H	H	⎍	L	H	L	H
H	H	⎍	H	H	Toggle	

DESIGN FOR MOD 8 UP COUNTER:



Present State			Next state			Flip flop inputs					
Q_C	Q_B	Q_A	Q_{C+1}	Q_{B+1}	Q_{A+1}	K_C	J_C	K_B	J_B	K_A	J_A
0	0	0	0	0	1	X	0	X	0	X	1
0	0	1	0	1	0	X	0	X	1	1	X
0	1	0	0	1	1	X	0	0	X	X	1
0	1	1	1	0	0	X	1	1	X	1	X
1	0	0	1	0	1	0	X	X	0	X	1
1	0	1	1	1	0	0	X	X	1	1	X
1	1	0	1	1	1	0	X	0	X	X	1
1	1	1	0	0	0	1	X	1	X	1	X

JA

QC \ QB QA	00	01	11	10
0	1	X	X	1
1	1	X	X	1

JA = 1

KA

QC \ QB QA	00	01	11	10
0	X	1	1	X
1	X	1	1	X

KA = 1

JB

QC \ QB QA	00	01	11	10
0	0	1	X	X
1	0	1	X	X

JB = QA

KB

QC \ QB QA	00	01	11	10
0	X	X	1	0
1	X	X	1	0

KB = QA

JC

QC \ QB QA	00	01	11	10
0	0	X	1	0
1	X	X	X	X

JC = QA QB

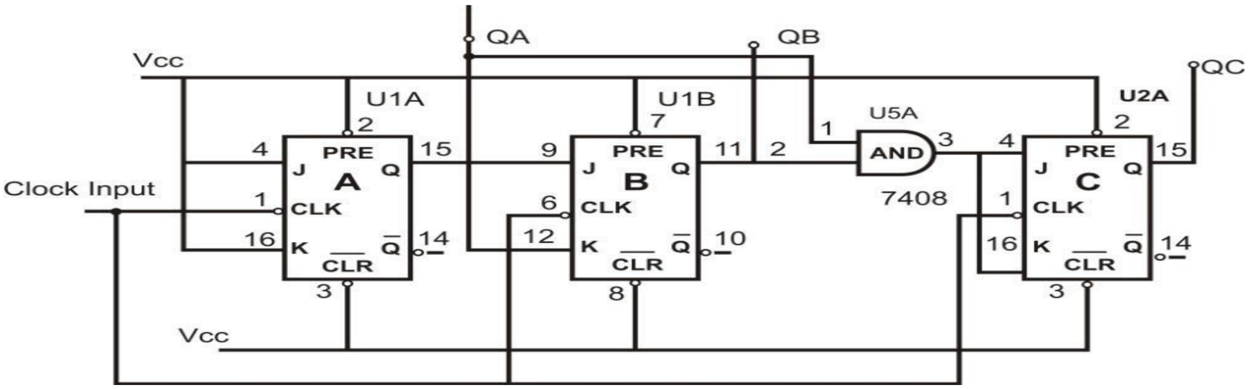
KC

QC \ QB QA	00	01	11	10
0	X	X	X	X
1	0	0	1	0

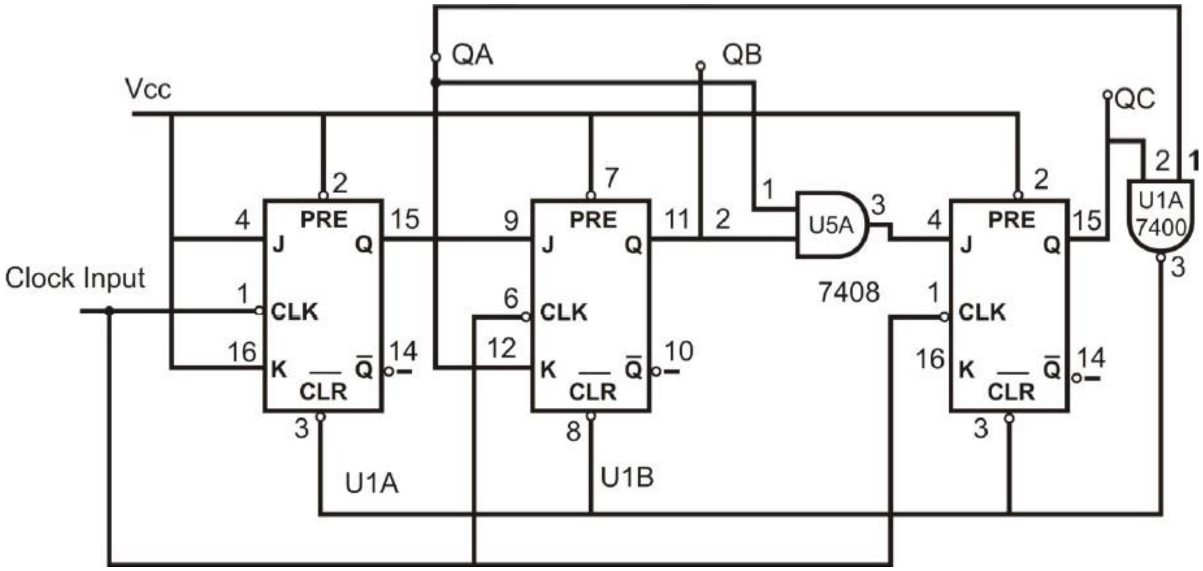
KC = QA QB

Qn	Qn+1	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Mod-8 Circuit Diagram:



Mod-5 Circuit Diagram:

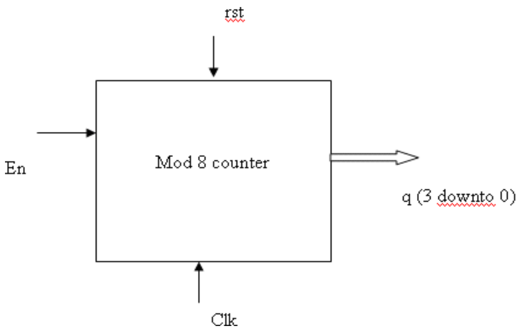


RESULT: All the entries in the truth table are verified.

Simulation using VHDL

Design and Develop the Verilog /VHDL code for Mod-8 up counter . Simulate and verify its working.

Truth Table

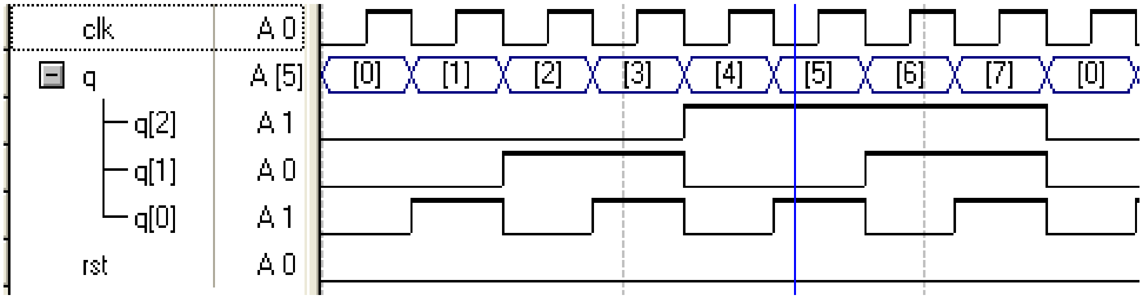


rst	Clk	En	Q
1	X	0	0000
0	1	1	0001
0	1	1	0010
0	1	1	0011
0	1	1	0100
0	1	1	0101
0	1	1	0110
0	1	1	0111

VHDL code for Mod-8 Counter.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity mod_8 is
          Port ( rst,clk,en: in std_logic;
          q : inout std_logic_vector(3 downto 0)); end

mod_8;

architecture Behavioral of mod_8 is begin
    process(clk,rst) is
                         begin
                    if rst='1' then q<="0000";
                              elsif rising_edge(clk) then



          if en='1' then
                                                  Q<=Q+1;
             end if;
             if Q="0111" then Q<=
             "0000";
               end if; end if;
             end process; end

Behavioral
```

Mod 8 Counter Simulation Results



EXPERIMENT-09

9.Design and implement asynchronous counter using decade counter IC to count up from 0 to n (n≤9) and demonstrate on 7- segment display using IC 7447.

AIM: To design and implement an asynchronous counter using decade counter IC to count up from 0to (n<9)

COMPONENTS REQUIRED: IC7490,7447, patch chords, power chords and trainer kit

THEORY: Asynchronous counter is a counter in which the clock signal is connected to the clock input of only first stage flip flop. The clock input of the second stage flip flop is triggered by the output of the first stage flip flop and so on. This introduces an inherent propagation delay time through a flip flop.

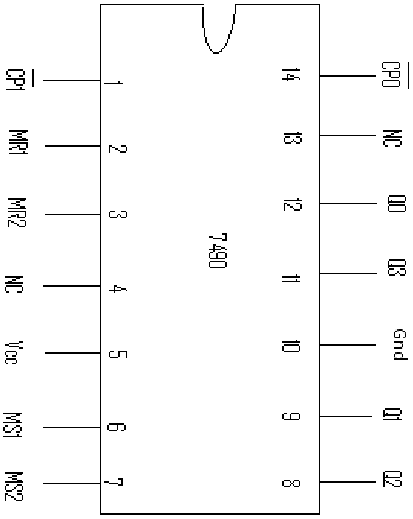
The 7490 is an asynchronous decade counter, able to count from 0 to 9 cyclically, and that is its natural mode. To make 7490 to work in normal mode the pin numbers 2, 3, 6, and 7 should hold at Low state. QA, QB, QC, QD are 4 output pins which gives the binary value of the decimal count. Pin 14 is Clock input.

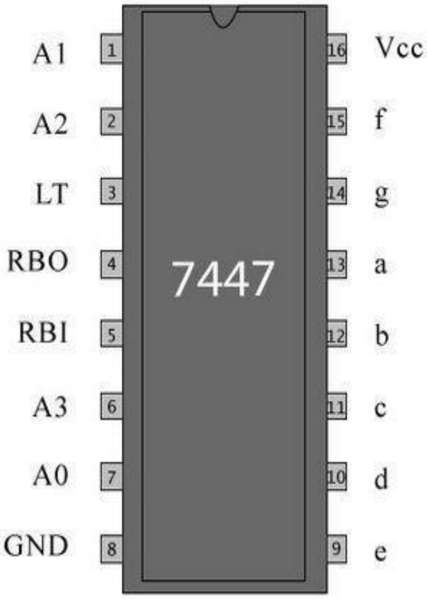
Pin 2 and 3: Set inputs. They held to Low to activate 7490 IC as decade counter. At any instant of time, if they provide High signal then the output will hold at Low state until Pin 2 and 3 brought to Low voltage.

Pin 6 and 7: Clear inputs. At any instant of time, if they provide High signal then the
output
will hold at High state until Pin 6 and 7 brought to Low voltage.

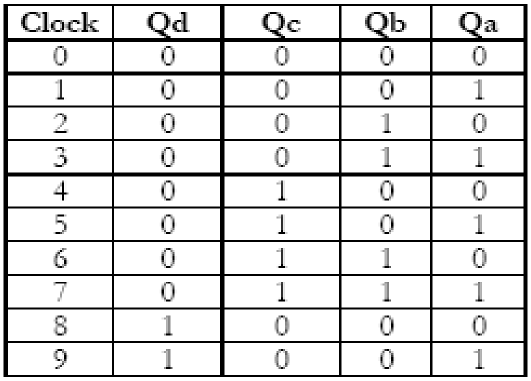
PROCEDURE:

1. Verify all the components and the patch chords whether they are in good condition or not.
2. Make connections as shown in the circuit diagram.
3. Give power supply to the trainer kit.
4. Provide input data to the circuit via switches.
5. Record and verify the output sequence for each combination of the select lines.

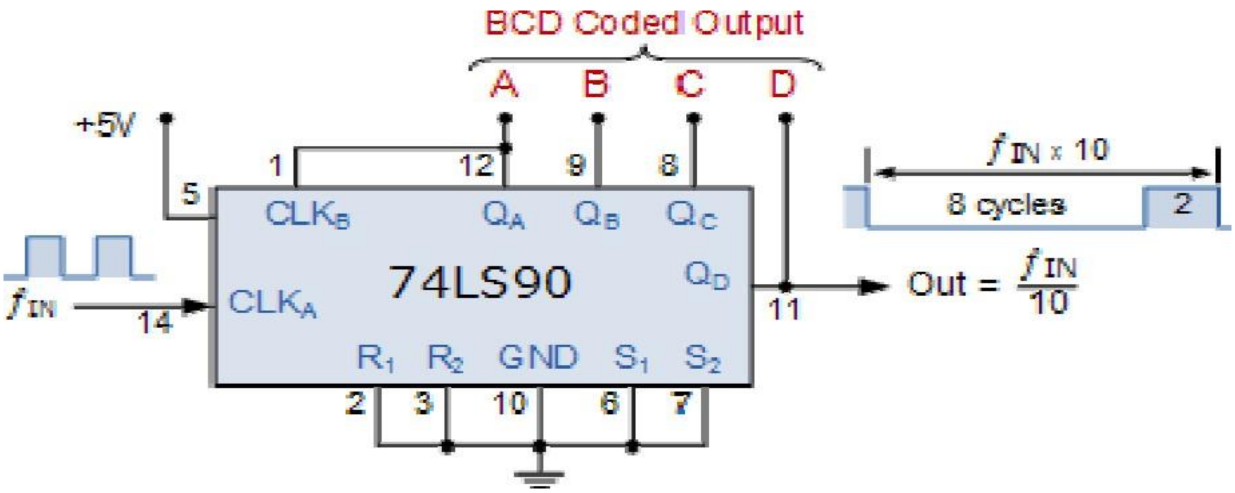




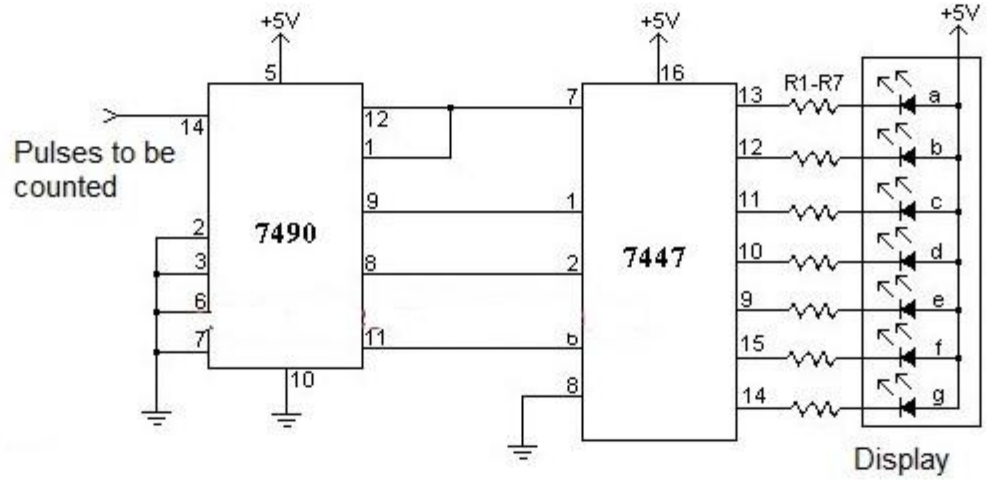
FUNCTION TABLE:



Clock	Qd	Qc	Qb	Qa
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1



Final 4-bit BCD Counter Circuit:



Clock	Qa	Qb	Qc
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
8	1	1	1

For mod 9
connect Q0 and Q3 to reset(clear) through an AND gate. Reset should not be connected to the switch
For mod8
Connect Q3 to reset
For mod7
Connect Q2, Q1,Q0 to reset through an And Gate
For Mod 6
Connect Q2 and Q1 to reset through an AND gate
For mod 5
Connect Q0 and Q2 to reset through an AND gate
For Mod 4 Connect Q2
to reset **For mod 3**
Connect Q1 and Q0 to reset through an AND gate
For mod 2 Connect
Q1 to reset **Function**

Then to summarize some of the **advantages of Asynchronous Counters:**

• Asynchronous Counters can easily be made from Toggle or D-type flip-flops.

• They are called –Asynchronous Counters‖ because the clock input of the flip-flops are not all driven by the same clock signal.

• Each output in the chain depends on a change in state from the previous flip-flops output.

• Asynchronous counters are sometimes called ripple counters because the data appears to –ripple‖ from the output of one flip-flop to the input of the next.

• They can be implemented using –divide-by-n‖ counter circuits.

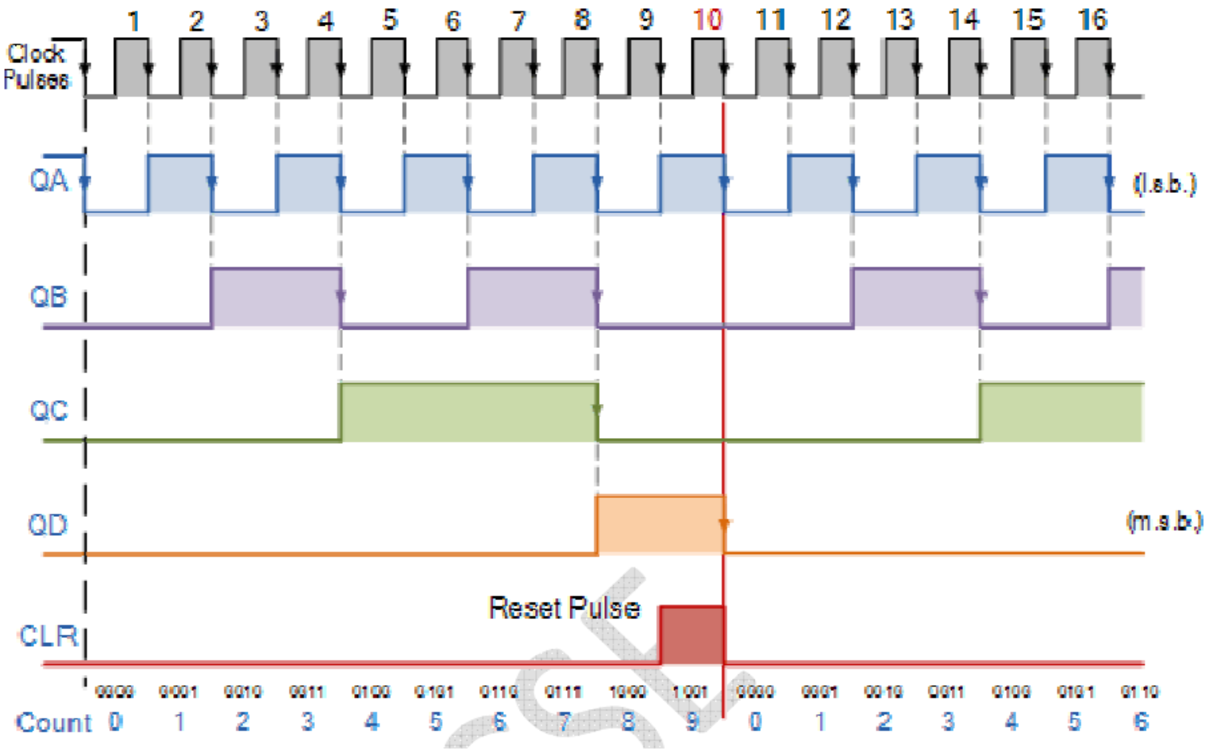
• Truncated counters can produce any modulus number count.

Disadvantages of Asynchronous Counters:

• An extra –re-synchronizing‖ output flip-flop may be required.

• To count a truncated sequence not equal to 2n, extra feedback logic is required.

• Counting a large number of bits, propagation delay by successive stages may become undesirably large.



RESULT: All the entries in the truth table are verified.

Simulation using VHDL

It is one of most popular software tool used to synthesize VHDL code. This tool Includes many steps. To make user feel comfortable with the tool the steps are given below:-

- Double click on Project navigator. (Assumed icon is present on desktop).
- Select **NEW PROJECT** in **FILE MENU.** Enter
  following details as per your convenience Project name
  : sample
  Project location : C:\example Top
  level module : HDL
- In **NEW PROJECT** dropdown Dialog box, Choose your appropriate device specification. Example is given below:
  Device family : Spartan2
  Device : xc2s200
  Package : PQ208 TOP
  Level Module : HDL Synthesis
  Tool : XST
  Simulation : Modelsim / others
  Generate sim lang : VHDL
- In source window right click on specification, select new source Enter
  the following details
  Entity: sample
  Architecture : **Behavioral**
  Enter the input and output port and modes.
  This will create **sample.VHD** source file. Click Next and finish the initial Project
  preparation.
- Double click on synthesis. If error occurs edit and correct VHDL code.
- Double click on Lunch modelsim (or any equivalent simulator if you are using) for functional simulation of your design.
- Right click on **sample.VHD** in source window, select new source Select
  source : Implementation constraints file.
  File name : sample

This will create sample. UCF constraints file.

- Double click on Edit constraint (Text) in process window. Edit and enter pin constraints with syntax:

  NET –NETNAME‖ LOC = –PIN NAME‖
- Double click on Implement, which will carry out translate, mapping, place and route of your design. Also generate program file by double clicking on it, intern which will create .bit file.
- Connect JTAG cable between your kit and parallel pot of your computer.
- Double click on configure device and select mode in which you want to configure your device. For ex: select slave serial mode in configuration window and finish your configuration.
- Right click on device and select _program‘. Verify your design giving appropriate inputs and check for the output.
- Also verify the actual working of the circuit using **pattern generator** & **logic analyzer**.

Sample Programs:

1. Write the Verilog /VHDL code for a 2 Input AND gate. Simulate and verify its working.

```
Entity And1 is
Port (A, B: IN STD_LOGIC; C: OUT
STD_LOGIC);
End And1;
Architecture Behavioral of And1 is Begin
C <= A AND B;
End Behavioral;
```

2. Write the Verilog /VHDL code for a half adder. Simulate and verify its working.

```
Entity Half_adder is
Port (A, B: IN STD_LOGIC;
SUM, CARRY: OUT STD_LOGIC);
End Half_adder;
Architecture Behavioral of Half_adder is Begin
SUM <= A XOR B; CARRY
<= A AND B;
End Behavioral;
```

VIVA QUESTIONS FOR ELECTRONIC CIRCUITS LAB

1. Define Schmitt trigger?

2. What are UTP and LTP values?
3. What is an Op-Amp relaxation oscillator?
4. What is a multivibrator?
5. How many types of multivibrators are there?
6. What is an Astable multivibrator?
7. What is a Monostable multivibrator?
8. Explain the pin diagram of 555 timer?
9. What is meant by duty cycle?
10. What is the function of a full wave rectifier? 11.

VIVA QUESTIONS FOR LOGIC DESIGN LAB

1. Distinguish between the sequential circuits and the Combinational circuits.
2. What is the expression for sum and carry of Half adder and full adder?
3. What is the disadvantage of ripple carry adder?
4. Mention the examples for sequential and combinational circuits.
5. What are the limitations of using K-map for simplifying the expression?
6. What are the different methods of simplifying the Boolean expression?
7. State DeMorgan‘s theorem.
8. What are universal gates?
9. Explain Multiplexer and Demultiplexer.
10. What is the significance of the select lines in MUX and DEMUX?
11. Indicate the number of inputs and outputs in a BCD Encoder.
12. What is latch?
13. What is Flip Flop?
14. Mention the different types of Flip Flops.
15. What is the difference between T Flip Flops and D Flip Flops?
16. What is Race around condition?
17. Which FF is used to eliminate the Race around condition?
18. Distinguish between the SR and JK FF.
19. How T FF is constructed from the JK FF?
20. How D FF is constructed from the JK FF?
21. What are Shift registers?
22. Mention the different types of shift registers.
23. Mention the applications of shift registers.
24. What is counter?
25. What is the difference between the synchronous and Asynchronous counters?
26. What is the difference between the up counter and down counter?
27. How many FFs are required to construct mod 16 counter?
28. What is truncated counter?
29. What is decade counter?
30. What is the binary sequence generated by mod-7 counter?
31. Distinguish between Johnson and Ring counter.