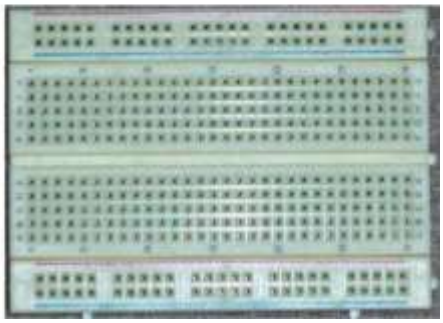
Seethalakshmi Ramaswami College (Autonomous)
Affiliated to Bharathidasan University
Accredited with "A" Grade (3rd cycle) by NAAC
Tiruchirappalli – 620 002

B.Sc. PROGRAMME

Department of Electronics



Electronics Lab

V - I Characteristics of P-N Junction Diodes

Aim:

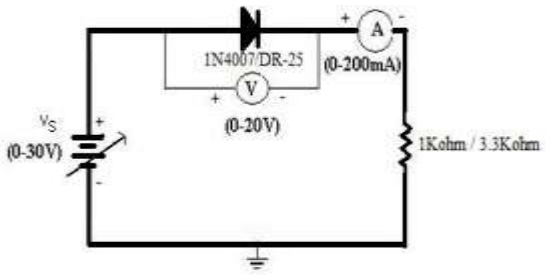
1. To plot V-I Characteristics of P-N Junction Diodes.
2. To find cut-in voltage for P-N Junction diodes.
3. To find static and dynamic resistances in both forward and reverse biased conditions.

Components required:

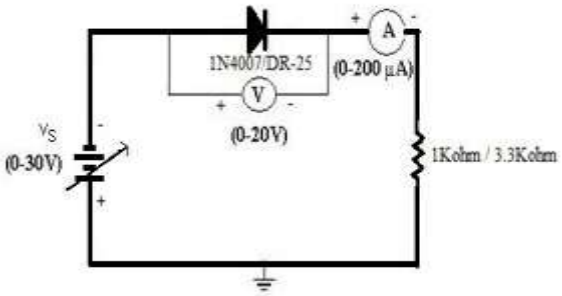
Name	Range	Quantity
Diodes 1N4007		1
Resistor 1K		1
Bread board		1
Regulated power supply	0-30V	1
Digital Ammeter	0-200µA/200mA	1
Digital Voltmeter	0-20V	1
Connecting Wires		

Circuit Diagrams

Forward Bias



Reverse Bias



Procedure:

Forward Bias Condition:

1. Connect the components as shown in the Fig.1.
2. Vary the supply voltage such that the voltage across the Silicon diode varies from 0 to 0.6 V in steps of 0.1 V and in steps of 0.02 V from 0.6 to 0.76 V. In each step record the current flowing through the diode as I.

Reverse Bias Condition:

1. Connect the diode in the reverse bias as shown in the Fig.2.
2. Vary the supply voltage such that the voltage across the diode varies from 0 to 10V in steps of 1 V. Record the current flowing through the diode in each step.
3. Now plot a graph between the voltage across the diode and the current flowing through the diode in forward and reverse bias, for Silicon and Germanium diodes on separate graph sheets. This graph is called the V-I characteristics of the diodes.
4. Calculate the static and dynamic resistance of each diode in forward and reverse bias using the following formulae

Static resistance, $R = V/I$

Dynamic resistance, $r = \Delta V/\Delta I$

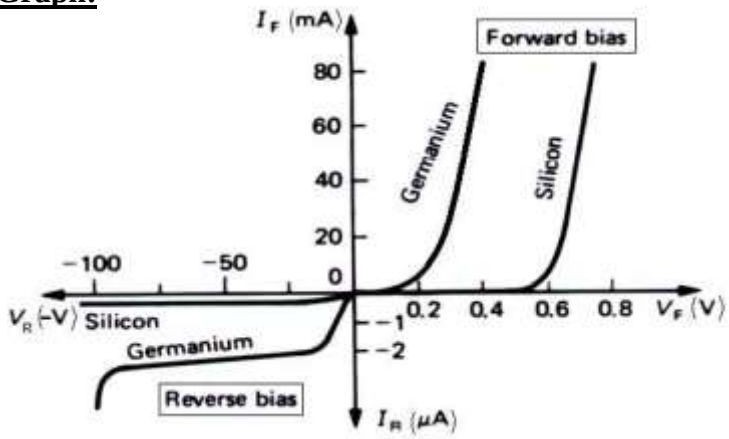
Forward Bias Condition

S. No.	Forward Voltage across the diode V_d (Volt)	Forward Current through the diode I_d (mA)

Reverse Bias condition

S. No.	Reverse Voltage across the diode V_R (Volt)	Reverse Current through the diode I_R (µA)

Calculations from Graph:



Static forward Resistance

$$R_{dc} = V_f / I_f \Omega$$

Static Reverse Resistance

$$R_{dc} = V_r / I_r \Omega$$

Dynamic Forward Resistance

$$r_{ac} = \Delta V_f / \Delta I_f \Omega$$

Dynamic Reverse Resistance

$$r_{ac} = \Delta V_r / \Delta I_r \Omega$$

Zener Diode Characteristics

Aim: To plot V-I Characteristics of Zener Diode.

Components:

Name	Quantity
Zener Diodes 1N4735A/ FZ 5.1	1
Resistor 1K	1

Equipments:

Name	Range	Quantity
Bread board		1
Regulated power supply	0-30V	1
Digital Ammeter	200mA	1
Digital Voltmeter	0-20V	1
Connecting Wires		

Specifications:

Breakdown Voltage = 5.1V

Power dissipation = 0.75W

Max Forward Current = 1A

.

Circuit Diagram:

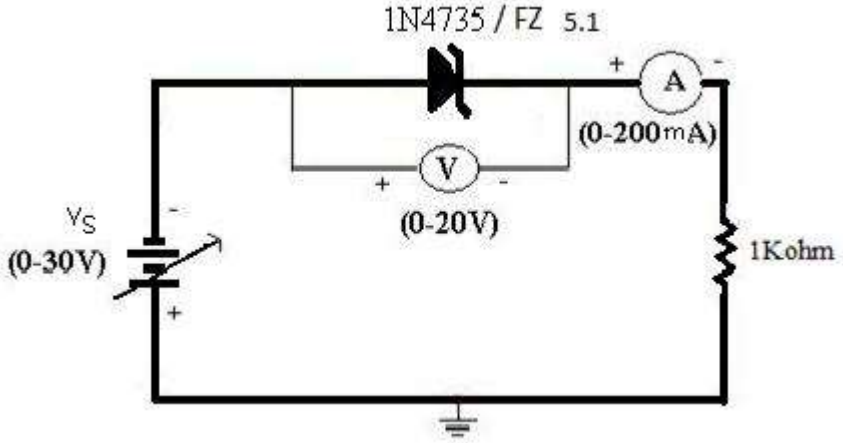


Fig. 1: Forward Bias Condition

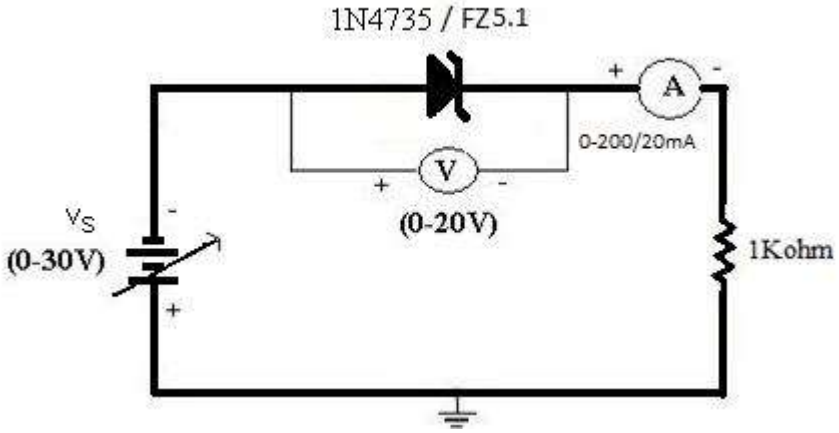


Fig. 2: Reverse Bias Condition

Procedure:

Forward Bias Condition:

1. Connect the circuit as shown in fig.1.
2. Vary V_F gradually from 0 to 0.6 V in steps of 0.1 V and in steps of 0.02 V from 0.6 to 0.76 V. In each step record the current flowing through the diode as I_F.
3. Tabulate different forward currents obtained for different forward voltages.

Reverse Bias Condition:

1. Connect the Zener diode in reverse bias as shown in the fig.2. Vary the voltage across the diode in steps of 1V from 0 V to 6 V and in steps 0.1 V till its breakdown voltage is reached. In each step note the current flowing through the diode
2. Plot a graph between V and I. This graph will be called the V-I characteristics of Zener diode. From the graph find out the breakdown voltage for the diode.

Forward Bias Condition:

S. No.	Forward Voltage across the diode V_F (volts)	Forward Current through the diode I_F (mA)

Observations:

Reverse Bias Condition:

S. No.	Reverse Voltage across the diode V_R (volts)	Reverse Current through the diode I_R (mA)

Graph:

1. Take a graph sheet and divide it into 4 equal parts. Mark origin at the center of the graph sheet.

2. Now mark +ve X-axis as V_F, -ve X-axis as V_R, +ve Y-axis as I_F and –ve Y-axis as I_R.

3. Mark the readings tabulated for forward biased condition in first Quadrant and reverse biased condition in third Quadrant.

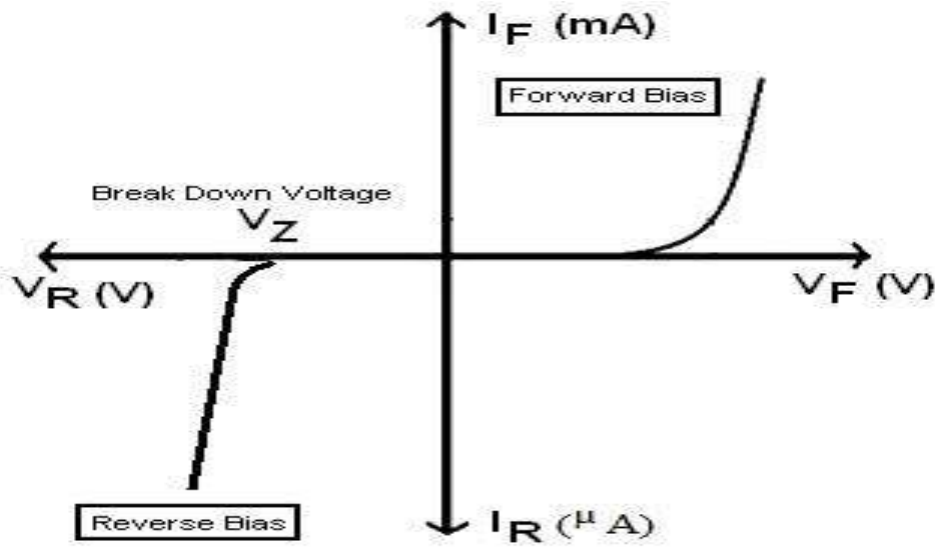


Fig. 3: V-I Characteristics of Zener Diode

Calculations from Graph:

Precautions:

1. While doing the experiment do not exceed the readings of the diode. This may lead to damaging of the diode.
2. Connect voltmeter and ammeter in correct polarities as shown in the circuit diagram.
3. Do not switch ON the power supply unless you have checked the circuit connections as per the circuit diagram.

Results:

1. The Zener Diode Characteristics have been studied.
2. The breakdown voltage of Zener diode in reverse bias was found to be = _____

Half Wave Rectifier

Aim: (i) To study the operation of a Half wave and Full wave rectifier with filters

(ii) To find its:

1. Ripple Factor
2. Percentage Regulation

Components:

Name	Quantity
Diodes 1N4007(Si)	2
Resistor 1K	1
Capacitor 100µF	2
Inductor (35 mH),	1

Equipment:

Name	Range	Quantity
CRO	(0-20)MHz	1
CRO probes		2
Digital Ammeter, Voltmeter	[0-200µA/200mA], [0-20V]	1
Transformer	220V/9V, 50Hz	1
Connecting Wires		

Specifications:

Silicon Diode 1N4007:

Max Forward Current = 1A

Max Reverse Current = 5.0µA

Max Forward Voltage = 0.8V

Max Reverse Voltage = 1000V

Max Power Dissipation = 30mW

Temperature = -65 to 200° C

Circuit Diagram:

Half Wave Rectifier (with L-section filter):

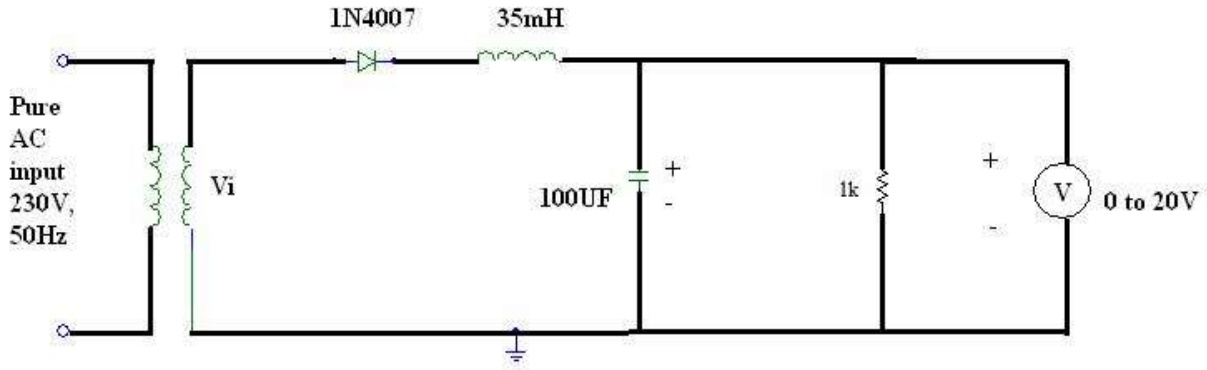


Fig. 1: Half wave rectifier with L-section Filter

Procedure:

PART-I: Half wave rectifier with L-section filter

1. Connect the circuit as shown in the fig.1.
2. Connect the multimeter across the 1kΩ load.
3. Measure the AC and DC voltages by setting multimeter to ac and dc mode respectively.
4. Now calculate the ripple factor using the following formula.

$$\text{Ripple factor } (\gamma) = \frac{V_{AC}}{V_{DC}}$$

5. Connect the CRO channel-1 across input and channel-2 across output i.e load and Observe the input and output Waveforms.
6. Now calculate the peak voltage of input and output waveforms and also the frequency.

PART-II: Full wave rectifier with -section filter

1. Connect the circuit as shown in the fig.2.
2. Repeat the above steps 2-6
3. Plot different graphs for wave forms and calculate ripple factor

Observations:

Half wave rectifier with L-section filter

Load Resistance (R_L)	V_{AC}(V)	V_{DC}(V)	Ripple Factor γ=	Input Signal		Output Signal	
				Vm p-p(v)	Frequency (Hz)	Vm p-p(v)	Frequency (Hz)

Calculations:

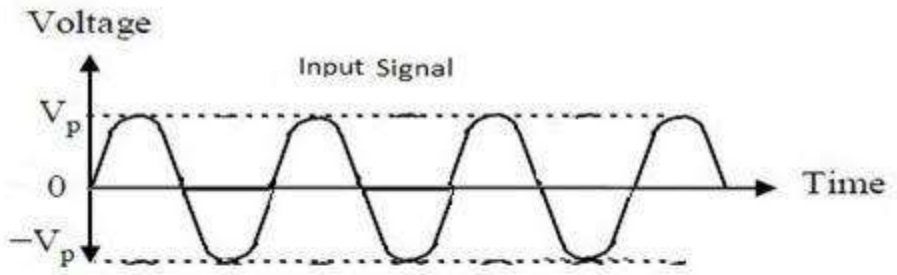
1. Ripple factor :

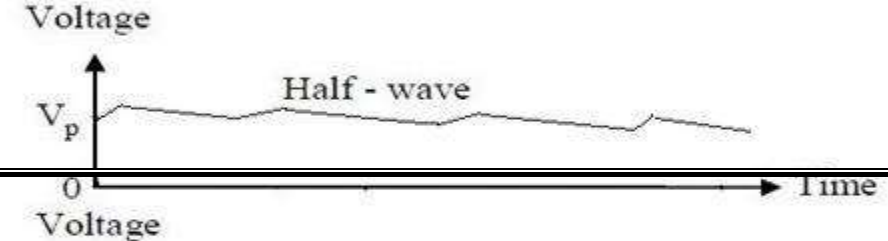
_ =

_ =

2. Percentage Regulation = × 100 %

Expected Waveforms:





Results:

Full Wave rectifier characteristics are studied.

1. Ripple factor of Half wave with L-section filter =
2. Ripple factor of Full wave with π-section filter = ----------
3. Regulation of Half wave with L-section filter = ----------
4. Regulation of Half wave with π -section filter = ----------

Full Wave Rectifier

(i) To study the operation of a Half wave and Full wave rectifier with filters

(ii) To find its:

1. Ripple Factor
2. Percentage Regulation

Components:

Name	Quantity
Diodes 1N4007(Si)	2
Resistor 1K	1
Capacitor 100µF	2
Inductor (35 mH),	1

Equipment:

Name	Range	Quantity
CRO	(0-20)MHz	1
CRO probes		2
Digital Ammeter, Voltmeter	[0-200µA/200mA], [0-20V]	1
Transformer	220V/9V, 50Hz	1
Connecting Wires		

Specifications:

Silicon Diode 1N4007:

Max Forward Current = 1A

Max Reverse Current = 5.0µA

Max Forward Voltage = 0.8V

Max Reverse Voltage = 1000V

Max Power Dissipation = 30mW

Temperature = -65 to 200° C

Circuit Diagram

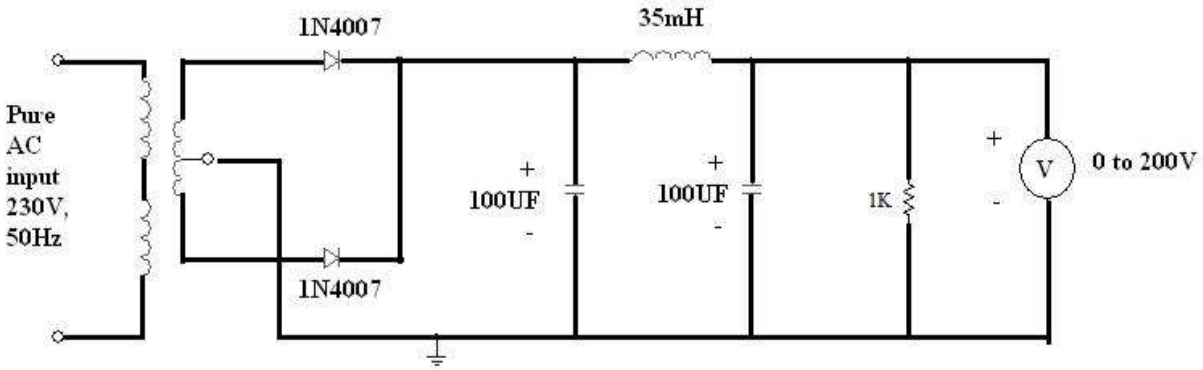
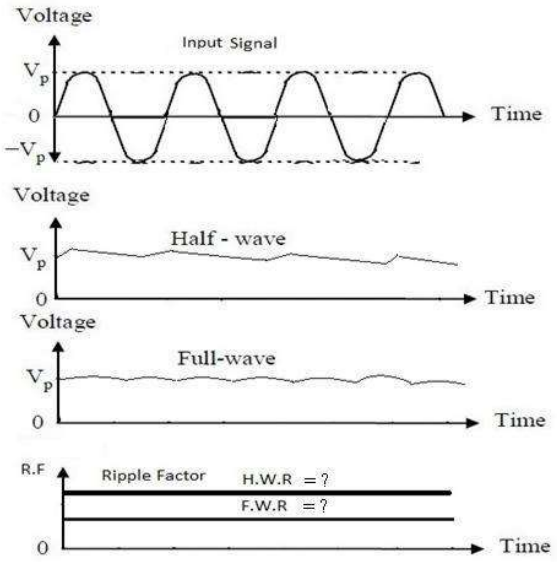


Fig. 2: Full wave rectifier with π-section filter

Expected Output waveforms:-



Load Resistance (R_L)	V_{AC}(V)	V_{DC}(V)	Ripple Factor $\gamma = \frac{\quad}{\quad}$	Input Signal		Output Signal	
				Vm p-p(v)	Frequency (Hz)	Vm p-p(v)	Frequency (Hz)

Calculations:

1. Ripple factor :

$$\_ = \frac{\quad}{\quad}$$

$$\_ = \frac{\quad}{\quad}$$

2. Percentage Regulation = $\frac{\quad - \quad}{\quad} \times 100\,\%$

Results:

Full Wave rectifier characteristics are studied.

Ripple factor of Full wave with Filter = ----------

Hartley & Colpitts Oscillators

Aim: To study the operation of Hartley & Colpitts oscillator circuits and to determine their frequency of oscillations.

Components and Equipments required: HI-Q electronics Hartley oscillator trainer, HI-Q electronics Colpitts oscillator trainer, CRO, CRO probe & connecting patch cords.

.

Circuit Diagram:

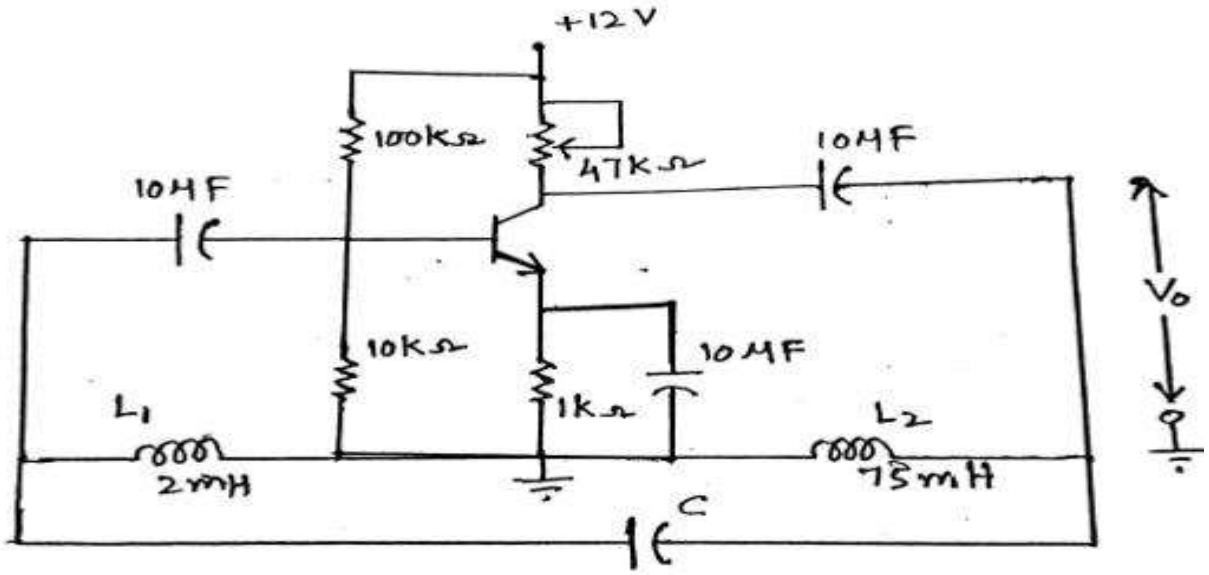


Fig. 1: Hartley oscillator

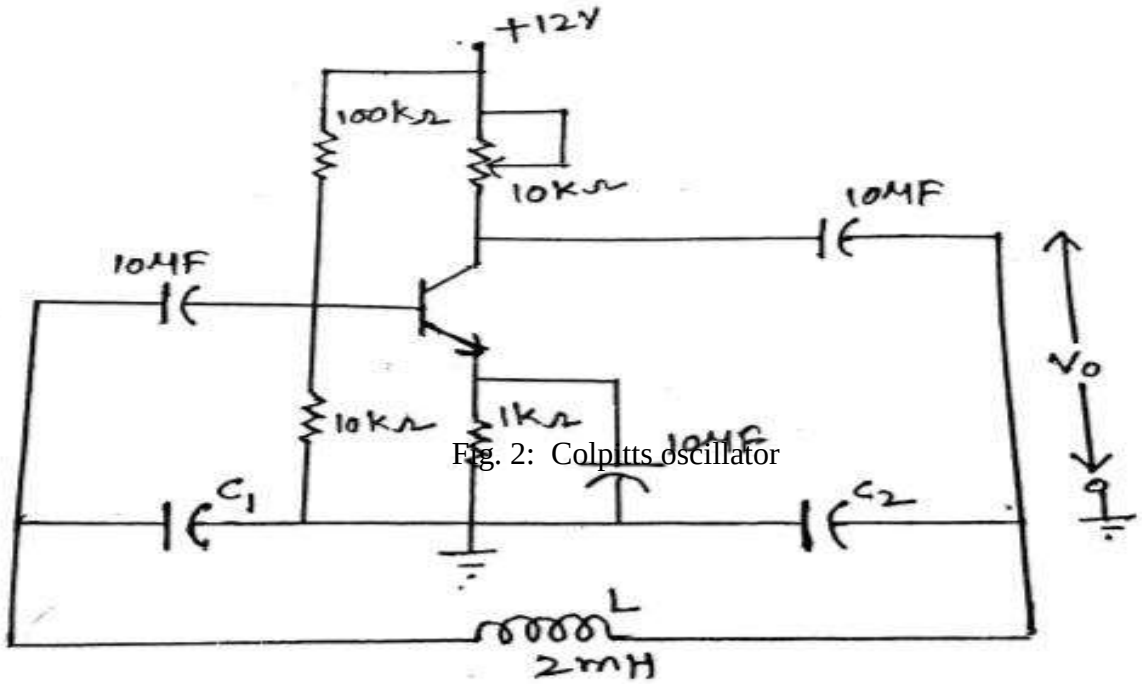


Fig. 2: Colpitts oscillator

Procedure:

Part-1: Hartley Oscillator

1) Complete the circuit as shown in Fig. 1, by connecting the capacitor C1 provided on panel of the trainer.

2) Switch ON the power supply for trainer.

3) Connect the CRO to the output terminals & vary the Rc till the stable oscillations are obtained on the CRO.

4) Now measure the frequency of oscillations practically.

5) Repeat the above steps for different values of capacitors C2, C3 & C4.

6) Draw the oscillations obtained on a graph sheet.

Part-2: Colpitts Oscillator

1) Complete the circuit as shown in Fig. 2, by connecting the capacitors C1 & C2 provided on panel of the trainer.
2) Switch ON the power supply for trainer.
3) Connect the CRO to the output terminals & measure the frequency of oscillations practically.
4) Repeat the above steps by selecting another pair of capacitors C1 & C2.
5) Draw the oscillations obtained on a graph sheet.

Observations:

Hartley Oscillator

S. No.	C	L1	L2	$f_0 = \frac{1}{2\pi\sqrt{(L_1+L_2)C}}$	T	f = 1/T

Colpitts Oscillator

S. No.	C1	C2	L	$f_o = \frac{1}{2\pi}\sqrt{\frac{C_1+C_2}{LC_1C_2}}$	T	f = 1/T

Results: The operation of Hartley & Colpitts oscillator circuits is studied, & their frequency of oscillations is verified with the theoretical values.

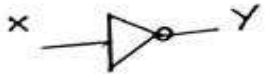
Truth Table verification of Logic Gates

Aim: To verify the truth tables of various logic Gates.

Components and Equipments required: Digital IC trainer kit, IC's (7400, 7402, 7404, 7408, 7432 & 7486) & connecting wires.

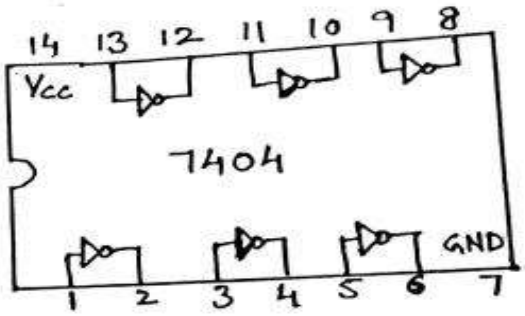
Truth Tables & IC Diagrams:

NOT Gate:



X	Y
0	1
1	0

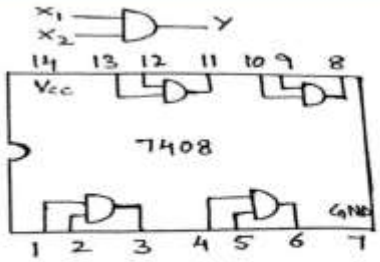
$Y = \overline{X}$



AND Gate:

X_1	X_2	Y
0	0	0
0	1	0
1	0	0
1	1	1

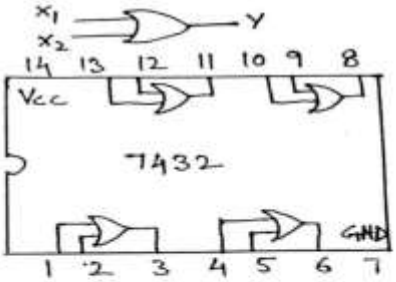
$Y = X_1 \cdot X_2$



OR Gate:

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	1

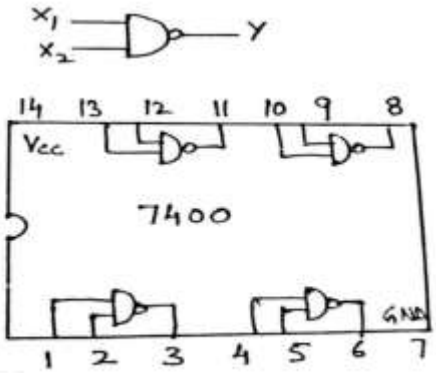
$Y = X_1 + X_2$



NAND Gate:

X_1	X_2	Y
0	0	1
0	1	1
1	0	1
1	1	0

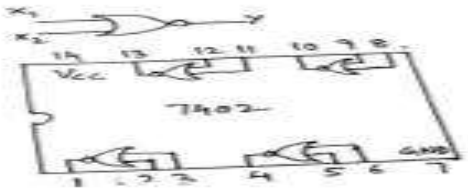
$Y = \overline{X_1 \cdot X_2}$



NOR Gate:

X_1	X_2	Y
0	0	1
0	1	0
1	0	0
1	1	0

$Y = \overline{X_1 + X_2}$

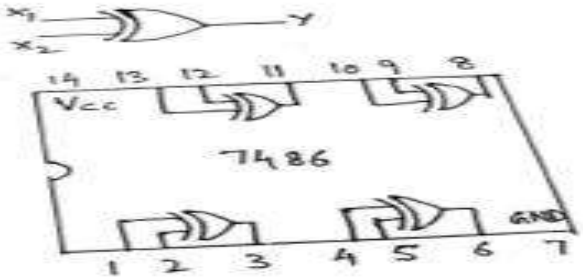


Ex-OR Gate:

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

$Y = A \cdot \overline{B} + \overline{A} \cdot B$

$= A \oplus B$



Procedure:

1. Connect the IC's on the trainer kit.
2. Connect Vcc = 5V & GND to pin 14 & 7 respectively.
3. Apply inputs to the logic gates from switches block of the trainer kit.
4. Verify output of the logic gates at LED indicators of the trainer kit.
5. Repeat the steps 3 & 4 for all the gates present in the IC.

Results: The truth tables of various logic Gates are verified.

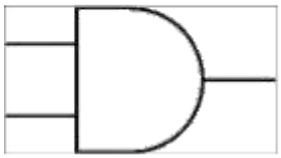
VERIFICATION OF LOGIC GATES USING ICS

AIM:- To study basic gates (AND , OR , NOT) and verify their truth tables.

APPARATUS:- LED, IC‟s , Wires , 5 volt DC supply, Bread Board etc.

THEORY:-

AND Gate



Traditional symbol

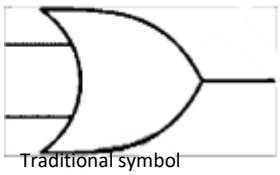
Input A	*Input B*	*Output Q*
0	*0*	*0*
0	*1*	*0*
1	*0*	*0*
1	*1*	*1*

Truth Table

In AND gate circuit it has n input and only one output. Digital signals are applied in input terminal. In the AND gate operation is „t‟ if and only if all the input are „1‟ otherwise zero.

Mathematically :The output Q is true if input A AND input B are both true: **Q = A AND B** An AND gate can have two or more inputs, its output is true if all inputs are true.

OR Gate



Traditional symbol

Truth Table

Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	1

In OR-Gate operation it has also n input and only one output. In OR operation output is one if and only if one or more input are '1'.

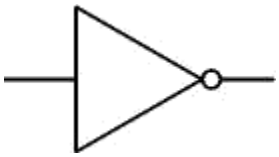
Mathematically

The output Q is true if input A OR input B is true (or both of them are true):

Q = A OR B

An OR gate can have two or more inputs, its output is true if at least one input is true.

NOT Gate (Inverter)



Traditional symbol

Input A	*Output Q*
0	*1*
1	*0*

Truth Table

It is also known as inverter. It has only one input and one output.Mathematically

*The output Q is true when the input A is NOT true, the output is the inverse of the input:**Q = NOT A**. A NOT gate can only have one input. A NOT gate is also called an inverter.*

***RESULT:-** Corresponding truth tables of logic gates are verified.*

PRECAUTIONS:-

1. *Supply should not exceed 5v.*
2. *Connections should be tight and easy to inspect.*
3. *Use L.E.D. with proper sign convention and check it before conneting in circuit.*

FLIPFLOP – RS,JK FLIPFLOPS

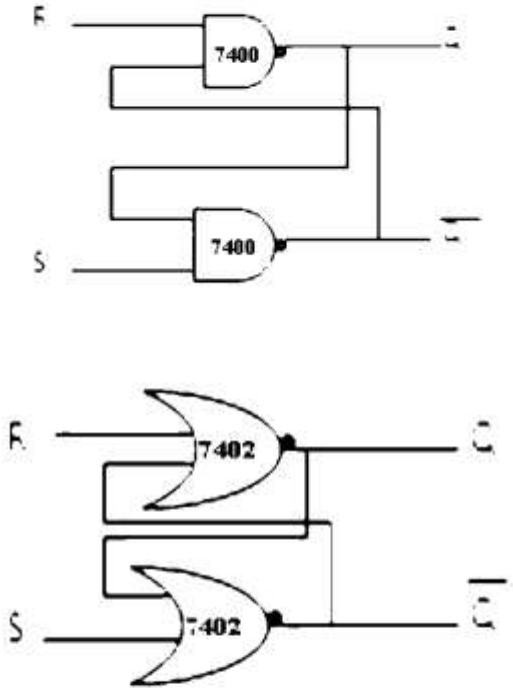
Aim: -To design and construct basic flip-flops R-S ,J-K,J-K Master slave flip-flops using gates and verify their truth tables

Apparatus: -

1 IC"s - 7404, 7402, 7400
2 Electronic circuit designer
3 Connecting patch chords

Circuit Diagrams:-

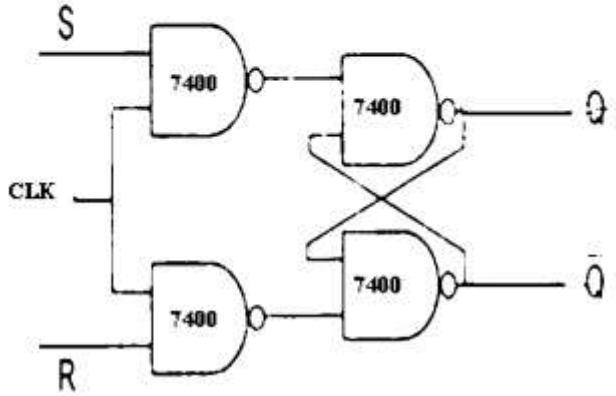
Basic flipflop using NAND gates



Truth Table

S	R	Q
0	0	Forbidden
0	1	1
1	0	0
1	1	No Change

Basic flipflop using NOR gates

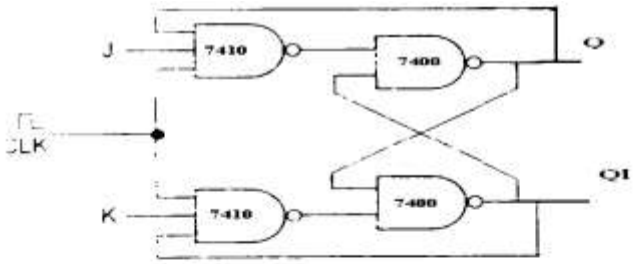


S	R	Q
0	0	No Change
0	1	0
1	0	1
1	1	Forbidden

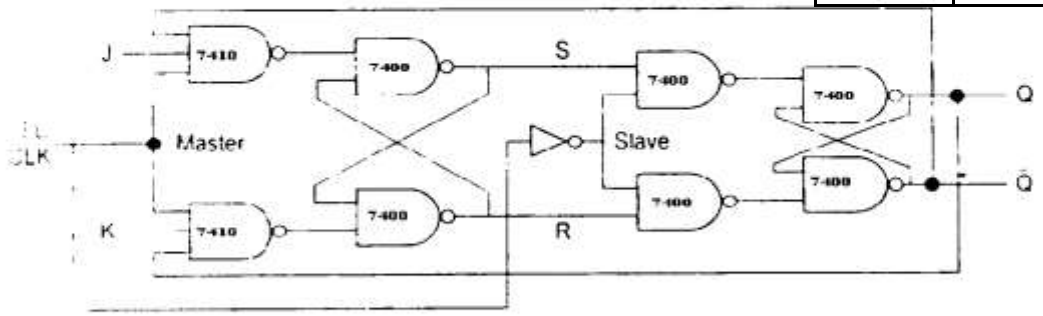
R-S flip-flop using NAND gates

S	R	Q
0	0	No Change
0	1	0
1	0	1
1	1	Forbidden

J-k flip-flop using NAND gates



J	K	Q
0	0	No Change
0	1	0
1	0	1
1	1	Race around



J-K Master Slave using NAND gates

J	K	Q
0	0	
0	1	0
1	0	1
1	1	

Procedure:

1. Connect the Flip-flop circuits as shown above.

2.Apply different combinations of inputs and observe the outputs .

Precautions: All the connections should be made properly.

Result: Different Flip-flops using gates are constructed and their truth tables are verified

ENCODER AND DECODER

AIM: To design and implement encoder and decoder using logic gates and study of IC 7445 and IC 74147.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
1.	3 I/P NAND GATE	IC 7410	2
2.	OR GATE	IC 7432	3
3.	NOT GATE	IC 7404	1
2.	IC TRAINER KIT	-	1
3.	PATCH CORDS	-	27

THEORY:

ENCODER:

An encoder is a digital circuit that performs inverse operation of a decoder. An encoder has 2^n input lines and n output lines. In encoder the output lines generates the binary code corresponding to the input value. In octal to binary encoder it has eight inputs, one for each octal digit and three output that generate the corresponding binary code. In encoder it is assumed that only one input has a value of one at any given time otherwise the circuit is meaningless. It has an ambiguila that when all inputs are zero the outputs are zero. The zero outputs can also be generated when D0 = 1.

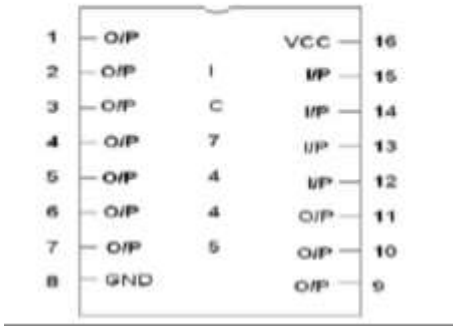
DECODER:

A decoder is a multiple input multiple output logic circuit which converts coded input into coded output where input and output codes are different. The input code generally has fewer bits than the output code. Each input code word produces a different output code word i.e there is one to one mapping can be expressed in truth table. In the block diagram of decoder circuit the

encodedinformation is present as n input producing 2 possible outputs. 2 output values are from 0 through output 2^n-1

PIN DIAGRAM FOR IC 7445:

BCD TO DECIMAL DECODER:



LOGIC DIAGRAM FOR ENCODER

TRUTH TABLE:

INPUT							OUTPUT		
Y1	**Y2**	**Y3**	**Y4**	**Y5**	**Y6**	**Y7**	**A**	**B**	**C**
1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**
0	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**1**
0	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**0**

0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	1	1	1	1

INPUT			OUTPUT			
E	A	B	D0	D1	D2	D3
1	0	0	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

PROCEDURE:

(i) Connections are given as per circuit diagram.
(ii) Logical inputs are given as per circuit diagram.
(iii) Observe the output and verify the truth table.

RESULT:

Thus the design and implementation of encoder and decoder using logic gates and study of IC 7445 and IC 74147 were done.

MULTIPLEXER & DEMULTIPLEXER

AIM: To design and implement Multiplexer and Demultiplexer using logic gates and study of IC 74150 and IC 74154.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
1.	3 I/P AND GATE	IC 7411	2
2.	OR GATE	IC 7432	1
3.	NOT GATE	IC 7404	1
4.	IC TRAINER KIT	-	1
5.	PATCH CORDS	-	

THEORY:

MULTIPLEXER:

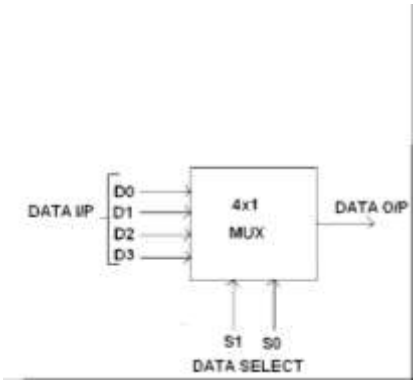
Multiplexer means transmitting a large number of information units over a smaller number of channels or lines. A digital multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line. The selection of a particular input line is controlled by a set of selection lines. Normally there are 2n input line and n selection lines whose bit combination determine which input is selected.

DEMULTIPLEXER:

The function of Demultiplexer is in contrast to multiplexer function. It takes information from one line and distributes it to a given number of output lines. For this reason, the demultiplexer is also known as a data distributor. Decoder can also be used as demultiplexer. In the 1: 4 demultiplexer circuit, the data input line goes to all of the AND gates. The data select lines enable only one gate

at a time and the data on the data input line will pass through the selected gate to the associated data output line.

LOCK DIAGRAM FOR 4:1 MULTIPLEXER:

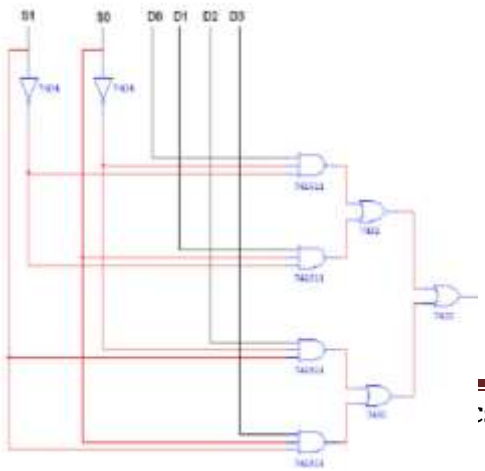


FUNCTION TABLE:

S1	S0	INPUTS Y
0	0	D0 → D0 S1' S0'
0	1	D1 → D1 S1' S0
1	0	D2 → D2 S1 S0'
1	1	D3 → D3 S1 S0

Y = D0 S1' S0' + D1 S1' S0 + D2 S1 S0' + D3 S1 S0

CIRCUIT DIAGRAM FOR MULTIPLEXER:

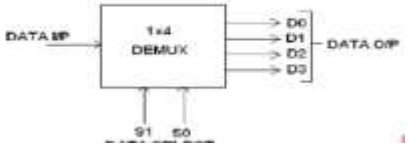


TRUTH TABLE:

S1	S0	Y = OUTPUT
0	0	D0
0	1	D1
1	0	D2
1	1	D3

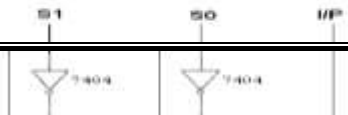
S1	S0	INPUT
0	0	X → D0 = X S1' S0'
0	1	X → D1 = X S1' S0
1	0	X → D2 = X S1 S0'
1	1	X → D3 = X S1 S0

BLOCK DIAGRAM FOR 1:4 DEMULTIPLEXER:



TRUTH TABLE:

Y = X S1' S0' + X S1' S0 + X S1 S0' + X S1 S0



Pin Diagram of 74150:-

PROCEDURE:

(i) Connections are given as per circuit diagram.
(ii) Logical inputs are given as per circuit diagram.
(iii) Observe the output and verify the truth table.

RESULT:

Thus the design and implementation of Multiplexer and Demultiplexer using logic gates and study of IC 74150 and IC 74154 were done

COUNTER

Aim:-To design and construct of 3-bit Synchronous up and down counters,2-bit up/down counter.

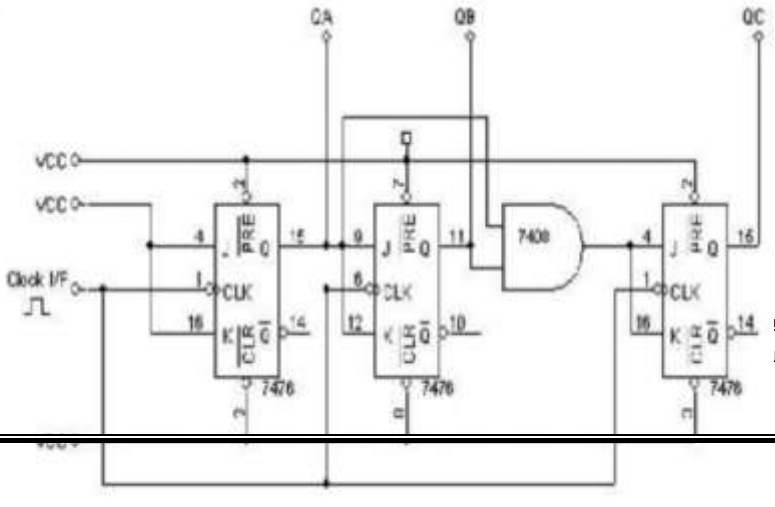
Apparatus:

1 IC"s - 7408,7476,7400,7432
2 Electronic circuit designer
3 Connecting patch chords

Circuit Diagram:

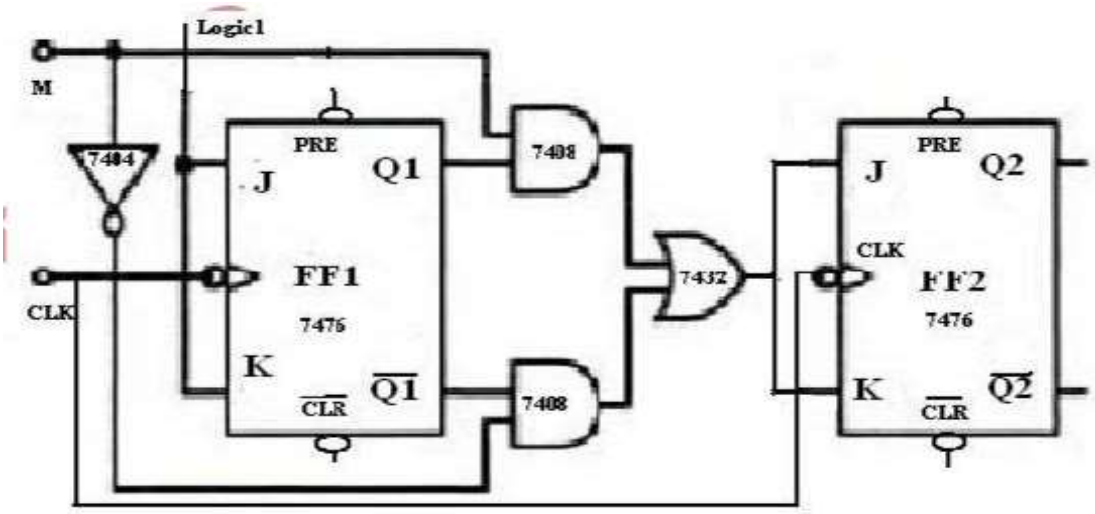
Truth Table

3-bit Synchronous Counter:-



3-bit synchronous up counter			
Clock	QC	QB	QA
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
7	1	1	1
8	0	0	0

Two Bit up/Down Counter using negative edge-triggered flip-flops



WHEN M=1

CLK	Q2	Q1
0	0	0
1	0	1
2	1	0
3	1	1

WHEN M=0

CLK	Q2	Q1
0	1	1
1	1	0
2	0	1
3	0	0

Procedure:

1 Connections are made as per the circuit diagram

2 Switch on the power supply.

3 Apply clock pulses and note the outputs after each clock pulse and note done the out puts.

Result: 3-bit Synchronous up and down counters,2-bit up/down counter are designed and truth tables are verified.

Precautions:

4. All the connections should be made properly.
5. IC should not be reversed.

AIM:- Realize Basic gates (AND,OR,NOT) From Universal Gates(NAND & NOR).

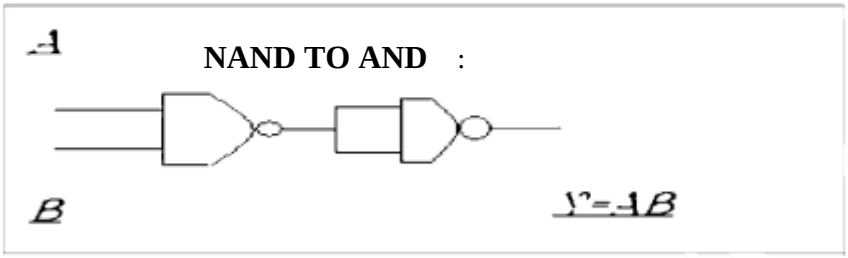
APPARATUS:- L.E.D., Bread-Board, I.C."s, Wires, "5.0" V d.c. supply, etc.

THEORY:-

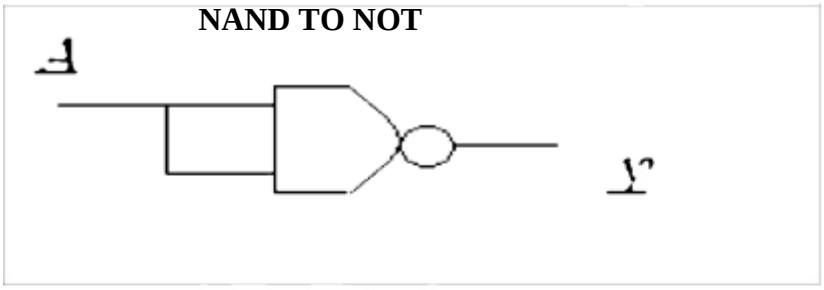
NAND Gates to AND, OR, NOT Gates:-

NAND gates is Universal gate. The Basic gates AND, OR, NOT can be realized from it. The Boolean equations and logic diagrams are as follows :

NAND TO AND :

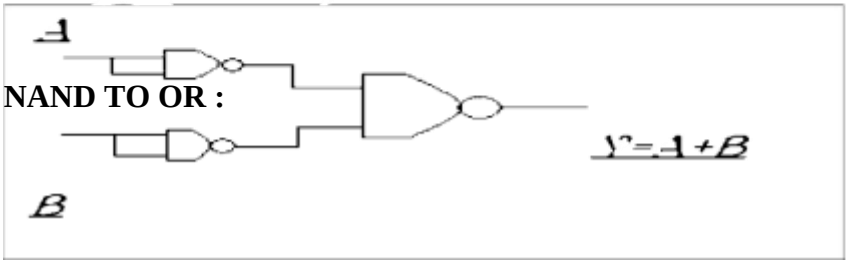


NAND TO NOT



:

NAND TO OR :

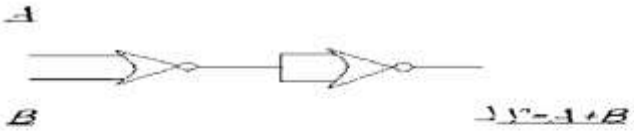


NOR Gate to AND, OR, NOT Gates

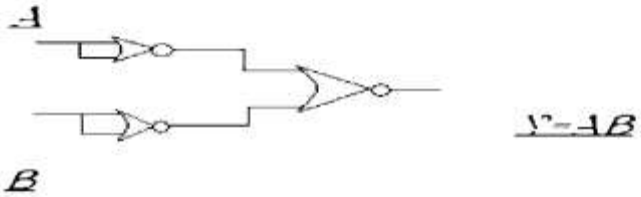
NOR gate is also an Universal gate. The Basic gates AND, OR, NOT can be realized from it.

The Boolean equations and logical diagrams are as follows :

NOR to OR Gate :



NOR to AND Gate



NOR to NOT Gate

Truth tables :

NAND to AND Gate

Inputs		Output
A	**B**	**Y**
0	0	0
0	1	0

1	0	0
1	1	1

NAND to OR Gate

Inputs		**Output**
A	**B**	**Y**
0	0	0
0	1	1
1	0	1
1	1	1

NAND to NOT Gate

input	**Output**
A	Y
0	1
1	0

NOR to AND Gate

Inputs		**Output**
A	**B**	**Y**
0	0	0
0	1	0
1	0	0
1	1	1

NOR to OR Gate

Inputs		**Output**
A	**B**	**Y**
0	0	0
0	1	1
1	0	1
1	1	1

NOR to NOT Gate

Input	**Output**
A	Y
0	1
1	0

RESULT:

The realization of basic gates(AND ,OR ,NOT) from universal gates(NAND &NOR) is successful.& The corresponding truth-tables are also verified.

PRECAUTIONS:-

1) Supply should not exceed 5v.
2) Connections should be tight and easy to inspect.
3) Use L.E.D. with proper sign convention and check it before conneting in circuit.

MICROPROCESSOR LAB

LIST OF EXEPRIMENTS

Sl.No.	Name of the Experiments	Page No.
1	Induction to 8085 Microprocessor	
2	a) Addition of 2 - 8 bit numbers b) Subtraction of 2 - 8 bit numbers	
3	a) Addition of 2 - 16 bit numbers b) Subtraction of 2 – 16 bit numbers	
4	a) Multiplication of 2 - 8 numbers b) Division of 2 - 8 bit numbers	
5	a) Ascending order b) Descending order	
6	a) Fibonaci Series b) Sum of finite series	
7	Factorial of Given Numbers	
8	a) Multiplication of 2 - 16 bit numbers b) Division of 2 - 16 bit numbers	
9	a) Binary to BCD code conversions b) BCD to Binary code conversions	

10	a) Rolling Display b) Flashing Display	
11	Stepper motor rotate forward and reverse direction	
12	Digital analog conversion	
13	Analog digital conversion	
14	Microcontroller a) Addition b) Subtraction c) Multiplication d) Division	

INTRODUCTION TO MICROPROCESSOR 8085

Aim

To study the microprocessor 8085

Architecture of 8085 Microprocessor

a) General purpose register

It is an 8 bit register i.e. B,C,D,E,H,L. The combination of 8 bit register is known as register pair, which can hold 16 bit data. The HL pair is used to act as memory pointer is accessible to program.

b) Accumulator

It is an 8 bit register which hold one of the data to be processed by ALU and stored the result of the operation.

c) Program counter (PC)

It is a 16 bit pointer which maintain the address of a byte entered to line stack.

d) Stack pointer (Sp)

It is a 16 bit special purpose register which is used to hold line memory address for line next instruction to be executed.

e) Arithmetic and logical unit

It carries out arithmetic and logical operation by 8 bit address it uses the accumulator content as input the ALU result is stored back into accumulator.

f) Temporary register

It is an 8 bit register associated with ALU hold data, entering an operation, used by the microprocessor and not accessible to programs.

g) Flags

Flag register is a group of fire, individual flip flops line content of line flag register will change after execution of arithmetic and logic operation. The line states flags are

1 Carry flag (C)
2 Parity flag (P)
3 Zero flag (Z)
4 Auxiliary carry flag (AC)
5 Sign flag (S)

h) Timing and control unit

Synchronous all microprocessor, operation with the clock and generator and control signal from it necessary to communicate between controller and peripherals.

i) Instruction register and decoder

Instruction is fetched from line memory and stored in line instruction register decoder the stored information.

j) Register Array

These are used to store 8 bit data during execution of some instruction.

PIN Description

Address Bus

1 The pins $A_o - A_{15}$ denote the address bus.

2 They are used for most significant bit

Address / Data Bus

1. $AD_0 - AD_7$ constitutes the address / Data bus

2. These pins are used for least significant bit

ALE : (Address Latch Enable)

4. The signal goes high during the first clock cycle and enables the lower order address bits.

IO / M

4 This distinguishes whether the address is for memory or input.
5 When this pins go high, the address is for an I/O device.

$S_0 - S_1$

S_0 and S_1 are status signal which provides different status and functions.

RD

(iv) This is an active low signal
(v) This signal is used to control READ operation of the microprocessor.

WR

(iii) WR is also an active low signal
(iv) Controls the write operation of the microprocessor.

HOLD

1. This indicates if any other device is requesting the use of address and data bus.

HLDA

4 HLDA is the acknowledgement signal for HOLD
5 It indicates whether the hold signal is received or not.

INTR

4 INTE is an interrupt request signal
5 IT can be enabled or disabled by using software

INTA

6. Whenever the microprocessor receives interrupt signal
7. It has to be acknowledged.

RST 5.5, 6.5, 7.5

3. These are nothing but the restart interrupts
4. They insert an internal restart junction automatically.

TRAP

5. Trap is the only non-maskable interrupt
6. It cannot be enabled (or) disabled using program.

RESET IN

1 This pin resets the program counter to 0 to 1 and results interrupt enable and HLDA flip flops.

X_1, X_2

These are the terminals which are connected to external oscillator to produce the necessary and suitable clock operation.

SID

This pin provides serial input data

SOD

This pin provides serial output data

V_{CC} and V_{SS}

4) V_{CC} is +5V supply pin

5) V_{SS} is ground pin

Specifications

1. Processors

Intel 8085 at E144 MHz clock

2. Memory

Monitor RAM:	0000– IFFF
EPROM Expansion:	2000– 3FFF's
	0000– FFF
System RAM:	4000– 5FFF
Monitor data area	4100– 5FFF
RAM Expansion	6000– BFFF

3. Input / Output

Parallel: A8 TTL input timer with 2 number of 32-55 only input timer available in μ-85 EBI.

Serial: Only one number RS 232-C, Compatible, crucial interface using 8281A

Timer: 3 channel -16 bit programmable units, using 8253 channel '0' used for no band late.

Clock generator. Channel '1' is used for single stopping used program.

Display: 6 digit – 7 segment LED display with filter 4 digit for adder display and 2 digit for data display.

Key board: 21 keys, soft keyboard including common keys and hexa decimal keys.

RES: Reset keys allow to terminate any present activity and retain to μ - 85 its on initialize state.

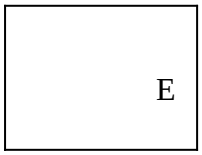
INT: Maskable interrupt connect to CPU's RST 7.5 interrupt

DEC: Decrement the adder by 1

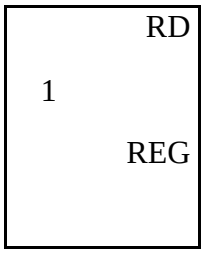
EXEC: Execute line particular value after selecting address through go command.

NEXT: Increment the address by 1 and then display its content.

Key Functions:

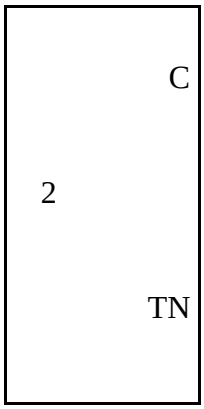


1. Hex entry key '0'
2. Substituting memory content where "next" key is paused immediately after 1, take used to st cutting address.
3. Register key 'E'



i) Hex code entry (1)

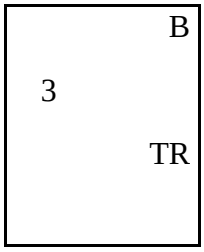
ii) Register key ‘D’



i) Hex code entry ‘2’

ii) Retricre data from data ‘memory’ to data top

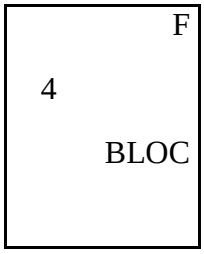
iii) Register key ‘C’



i) Hex code entry ‘3’

ii) Retricre data from memory to top

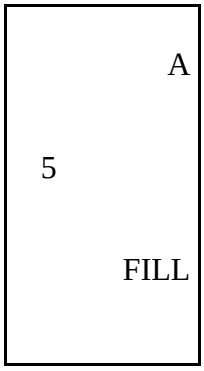
iii) Register key ‘B’



i) Hex key entry ‘C’

ii) Block search from byte

iii) Register key ‘F’

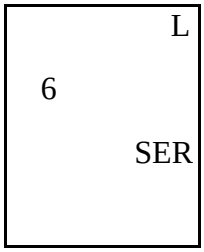


i) Hex key entry ‘5’

ii) Fill block of RAM memory with desired data

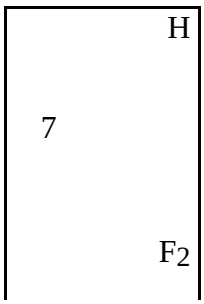
iii) Register key ‘A’

i) Hex key entry ‘6’



ii) TN/Tl used for sending (or) receiving

iii) Register key ‘H’



i) Hex key entry ‘7’

ii) Register key ‘H’

ADDITION OF TWO 8-BIT NUMBERS

Aim:

To write an assembly language for adding two 8 bit numbers by using micro processor kit.

Apparatus required:

8085 micro processor kit

(0-5V) DC battery

Address	Label	Mnemonics	Hex Code	Comments
4100		MVI C,00	OE, 00	Initialize the carry as zero
4102		LDA 4300	3A, (00, 43)	Load the first 8 bit data
4105		MOV, B,A	47	Copy the value of 8 bit data into register B
4106		LDA 4301	3A, (01, 43)	Load the second 8 bit data into the accumulator
4109		ADD B	80	Add the hoo values
410A		JNC	D2, 0E, 41	Jump on if no carry
410D		INR C	OC	If carry is there increment it by one
410E	Loop	STA 4302	32 (02, 43)	Stone the added value in the accumulator
4111		MOV A,C	79	More the value of carry to the accumulator from register C
4112		STA 4303	32 (03, 43)	Store the value of carry in the accumulator
4115		HLT	76	Stop the program execution

Input

Without carry

Input Address	Value
4300	04
4301	02

Output

Output Address	Value
4302	06
4303	00 (carry)

With carry

Input Address	Value
4300	FF
4301	FF

Output Address	Value
4302	FE
4303	01 (carry)

Calculation 1111 1111

1111 1111

---

(1) 1111 1110

=========

FE

Result:

The assembly language program for 8 bit addition of two numbers was executed successfully by using 8085 micro processing kit.

SUBTRACTION OF TWO 8 BIT NUMBERS

Aim:

To write a assembly language program for subtracting 2 bit (8) numbers by using-8085 micro processor kit.

Apparatus required:

8085 micro processor kit

(0-5V) DC battery

Address	Label	Mnemonics	Hex Code	Comments
4100		MVI C,00	OE, 00	Initialize the carry as zero
4102		LDA 4300	3A, (00, 43)	Load the first 8 bit data into the accumulator
4105		MOV, B,A	47	Copy the value into register 'B'
4106		LDA 4301	3A, (01, 43)	Load the 2^{nd} 8 bit data into the accumulator
4109		SUB B	90	Subtract both the values
410A	Loop	INC	D2, 0E, 41	Jump on if no borrow
410D		INR C	OC	If borrow is there, increment it by One
410E	Loop	CMA	2F	Compliment of 2^{nd} data
410F		ADI, 01	6, 01	Add one to 1's compliment of 2^{nd} Data
4111		STA 4302	32,02,43	Store the result in accumulator
4114		MOV A,C	79	Moul the value of borrow into the accumulator
4115		STA 4303	32,03,43	Store the result in accumulator
4118		HLT	76	Stop Program execution

Input

Without borrow

Input Address	Value
4300	05
4301	07

Output

Output Address	Value
4302	02
4303	00 (borrow)

With carry borrow

Input Address	Value
4300	07
4301	05

Output Address	Value
4302	02
4303	01 (borrow)

Calculation 05 – 07

07 – 0111

CMA 1000

ADJ 0.1 0001

1001

05 - 0101

1110 (-2)

Result:

The assembly language program subtraction of two 8 bit numbers was executed successfully by using 8085 micro processing kit.

ADDITION OF TWO 16 – BIT NUMBERS

Aim:

To write an assembly language program for adding two 16 bit numbers using 8085 micro processor kit.

Apparatus required:

8085 micro processor kit

(0-5V) DC battery

Address	Label	Mnemonics		Hex Code	Comments
4500		MVI	C,00	0E	C = 00_H
4501				00	
4502		LHLD	4800	2A	HL – 1^{st} No.
4503				00	
4504				48	
4505		XCHG		EB	HL – DE
4506		LHLD	4802	2A	HL – 2^{nd} No.
4507				02	
4508				48	
4509		DAD	D	19	Double addition DE + HL
450A		JNC	Ahead 450E	D2	If Cy = 0, G0 to 450E
450B				0E	
450C				45	
450D		INR	C	0C	C = C + 01
450E	AHEAD	SHLD	4804	22	HL – 4804 (sum)
450F				04	
4510				48	
4511		MOV	C,A	79	Cy – A

4512		STA	4806	32	Cy – 4806
4513				06	
4514				48	
4515		HLT		76	Stop excution

Input

Without

Input Address	Value
4800	01 (addend)
4801	04
4802	02 (augend)
4803	03 (augend)

Output

Output Address	Value
4804	03 (sum)
4805	07 (sum)
4806	00 (carry)

Calculation 0000 0100 0000 0001

0000 0011 0000 0010

0000 0111 0000 0011

0 7 0 3

With carry

Input Address	Value
4800	FF (addend)
4801	DE (addend)
4802	96 (augend)
4803	DF (augend)

Output Address	Value
4804	95 (sum)
4805	BE (sum)
4806	01 (carry)

Calculation 1101 1110 1111 1111

1101 1111 1001 0101

1011 1110 1001 0101

B E 9 5

Result:

The assembly language program for addition of two 16 bit numbers was executed using 8085 micro processing kit.

SUBTRACTION OF TWO 16 – BIT NUMBERS

Aim:

To write an assembly language program for subtracting two 16 bit numbers using 8085 microprocessor kit.

Apparatus required:

8085 microprocessor kit

(0-5V) DC battery**Address**	**Label**	**Mnemonics**		**Hex Code**	**Comments**
4500		MVI	C,00	0E	C = 00_H
4501				00	
4502		LHLD	4800	2A	L – 1^{st} No.
4503				00	
4504				48	
4505		XLHG		EB	HL – DE
4506		LHLD	4802	2A	HL – 2^{nd} No.
4507				02	
4508				48	
4509		MOV	A,E	7B	LSB of '1' to 'A'
450A		SUB	L	95	A – A – L
450B		STA	4804	32	A – memory
450C				04	
450D				48	
450E		MOV	A,D	7A	MSB of 1 to A
450F		SBB	H	9C	A- A – H
4510		STA	4805	32	A – memory
4511				05	

4512				48	
4513		HLT		76	Stop execution

Input

Without borrow

Input Address	Value
4800	07
4801	08
4802	05
4803	06

Output

Output Address	Value
4804	02
4805	02
4807	00

With borrow

Input Address	Value
4800	05
4801	06
4802	07
4803	08

Output Address	Value
4804	02
4805	02
4806	01

Calculation

05	06	-	07	08				
05	06	0101	0110		07	08	0111	1000
CMA		1010	1001		CMA		1000	0111
ADI		0000	0001		ACI		0000	0001
		---	---				---	---
		1010	1010				1000	1000

05	06	+	07	08
			1010	1010
			1000	1000
			---	---
		(1)	0010	0010
			02	02

Result:

The assembly language program for subtraction of two 16 bit numbers was executed by using 8085 micro processing kit.

MULTIPLICATION OF TWO 8 – BIT NUMBERS

Aim:

To write an assembly language for multiplying two 8 bit numbers by using 8085 micro processor kit.

Apparatus required:

8085 microprocessor kit

(0-5V) DC battery

Address	Label	Mnemonics	Hex Code	Comments
4100		LDA 4500	3A, 00, 45	Load the first 8 bit number
4103		MOV B,A	47	Move the 1st 8 bit data to register 'B'
4104		LDA 4501	3A, 01, 45	Load the 2nd 16 it number
4107		MOV C,A	4F	Move the 2nd 8 bit data to register 'C'
4108		MVI A, 00	3E, 00	Intialise the accumulator as zero
410A		MVI D, 00	16, 00	Intialise the carry as zero
410C		ADD B	80	Add the contents of 'B' and accumulator
410D		INC	D2 11, 41	Jump if no carry
4110		INR D	14	Increment carry if there is
4111		DCR C	OD	Decrement the value 'C'
4112		JNZ	C2 0C, 41	Jump if number zero
4115		STA 4502	32 02, 45	Store the result in accumulator
4118		MOV A,D	7A	Move the carry into

				accumulator
4119		STA 4503	32,03,45	Store the result in accumulator
411C		HLT	76	Stop the program execution

Input

Input Address	Value
4500	04
4501	02

Output

Output Address	Value
4502	08
4503	00

Result:

The assembly language program for multiplication of two 8 bit numbers was executed using 8085 micro processing kit.

DIVISION OF TWO 8 – BIT NUMBERS

Aim:

To write an assembly language program for dividing two 8 bit numbers using microprocessor kit.

Apparatus required:

8085 microprocessor kit

(0-5V) DC battery

Address	Label	Mnemonics	Hex Code	Comments
4100		MVI C, 00	0E, 00	Intialise Quotient as zero
4102		LDA, 4500	3A 00, 45	Get the 1st data
4105		MOV B,A	47	Copy the 1st data into register 'B'
4106		LDA, 4501	3A 01, 45	Get the 2nd data
4109		CMP B	B8	Compare the 2 values
410A		JC (LDP)	DA 12,41	Jump if dividend lesser than divisor
410D	Loop 2	SUB B	90	Subtract the 1st value by 2nd value
410E		INR C	0C	Increment Quotient (410D)
410F		JMP (LDP, 41)	C3, 0D, 41	Jump to Loop 1 till the value of dividend becomes zero
4112	Loop 1	STA 4502	32 02,45	Store the value in accumulator

4115		MOV A,C	79	Move the value of remainder to accumulator
4116		STA 4503	32 03,45	Store the remainder value in accumulator
4119		HLT	76	Stop the program execution

Input

Input Address	Value
4500	09
4501	02

Output

Output Address	Value
4502	04 (quotient)
4503	01 (reminder)

1001

0010 – I

0111

0010 – II

0101

0010 – III

0011

0010 – IV

0001 – carry

Quotient - 04

Carry - 01

Result:

The assembly language program for division of two 8 bit numbers was executed using 8085 micro processing kit.

ASCENDING ORDER

Aim:

To write a program to sort given 'n' numbers in ascending order

Apparatus required:

8085 microprocessor kit

(0-5V) DC battery

Address	Label	Mnemonics	Hex Code	Comments
4100		LDA 4500	3A, 00,45	Load the number of values
4103		MOV B,A	47	Move it 'B' register
4104		DCR B	05	For (N-1) comparisons
4105	Loop 3	LXI H, 4500	21, 00,45	Set the pointer for array
4108		MOV C,M	4E	Count for (N-1) comparisons
4109		DCR C	0D	For (N-1) comparisons
410A		INX H	23	Increment pointer
410B	Loop 2	MOV A,M	7E	Get one data in array 'A'
410C		INX H	23	Increment pointer
410D		CMP M	BE	Compare next with accumulator
410E		JC	DA, 16, 41	If content less memory go ahead
4111		MOV D,M	56	If it is greater than interchange it
4112		MOV M,A	77	Memory content
4113		DCX H	2B	Exchange the content of memory pointed by 'HL' by previous location
4114		MOV M,D	72	One in by 'HL' and previous location

4115		INX H	23	Increment pointer
4116	Loop 1	DCR C	0D	Decrement ‘C’ register
4117		JNZ Loop 1	C2, 0B, 41	Repeat until ‘C’ is zero
411A		DCR B	05	Decrement in ‘B’ values
411B		JNZ Loop 2	C2, 05, 41	Repeat till ‘B’ is zero
411E		HLT	76	Stop the program execution

Input

Input Address	Value
4500	04
4501	AB
4502	BC
4503	01
4504	0A

Output Address & Value

Output Address	Value
4500	04
4501	01
4502	0A
4503	AB
4504	BC

Result:

The assembly language program for sorting numbers in ascending order was executed by microprocessor kit.

DESCENDING ORDER

Aim:

To write a program to sort given 'n' numbers in descending order

Apparatus required:

8085 microprocessor kit

(0-5V) DC battery

Address	Label	Mnemonics	Hex Code	Comments
4100		LDA 4500	3A, 00,45	Load the number of values in accumulator
4103		MOV B,A	47	Move it to 'B' register
4104		DCR B	05	For (N-1) comparisons
4105	Loop 3	LXI H, 4500	21, 00,45	Set the pointer for array
4108		MOV C,M	4E	Count for (N-1) comparisons
4109		DCR C	0D	For (N-1) comparisons
410A		INX H	23	Increment pointer
410B	Loop 2	MOV A,M	7E	Get one data from array
410C		INX H	23	Increment pointer
410D		CMP M	BE	Compare next with number
410E		ICE, Loop 1	D2, 16,41	If content 'A' is greater than content of 'HL' pair
4111		MOV D,M	56	If it is greater than interchange the datas
4112		MOV M,A	77	Accumulator to memory value
4113		DCX H	2B	Decrement memory pointer
4114		MOV M,D	72	Move the old to 'HL' and previous location

4115		INX H	23	Increment pointer
4116	Loop 1	DCR C	0D	Decrement 'C' register
4117		JNZ Loop 2	C2, 0B, 41	Repeat till 'C' is zero
411A		DCR B	05	Decrement in 'B' values
411B		JNZ Loop 3	C2, 05, 41	Jump to loop till the value of 'B' be
411E		HLT	76	Stop the program execution

Input

Input Address	Value
4500	04
4501	AB
4502	BC
4503	01
4504	0A

Output Address & Value

Output Address	Value
4500	04
4501	BC
4502	AB
4503	0A
4504	01

Result:

The assembly language program for sorting '4' numbers in descending order was executed successfully using microprocessor kit.

FIBANOCCI SERIES

Aim:

To write an assembly language program to displace Fibanocci Series.

Apparatus required:

8085 microprocessor kit

(0-5V) DC battery

Address	Label	Mnemonics	Hex Code	Comments
4200		LDA 4300	3A, 00, 43	Store the length of series in 'A'
4203		SUI 02	D6, 02	Decrement 'A' by 02
4205		MOV D,A	57	Move 'A' to 'D' (counter)
4206		LXI H, 4301	21,01,43	Load the starting address of array
4209		MVI M,00	36,00	Intialise 4301 as '00'
420B		INX H	23	Increment pointer
420C		MVI M, 01	36,01	Initialize 2nd as '01'
420E		INX H	23	Increment pointer
420F		MVI B,00	06,00	Intialise 'B' as '00'
4211		MVI, C, 01	0E, 01	Intialise 'C' as '01'
4213	Loop	MOV A,B	78	Move B to A
4214		ADD C	81	Add 'A' and 'C'
4215		MOV B,C	41	Move C to B
4216		MOV C,A	4F	Move A to C
4217		MOV M,A	77	Move the result to memory
4218		INX H	23	Increment pointer
4219		DCR D	15	Decrement counter
421A		JNZ loop	C2, 13,42	If D = 0, jump to loop

421D		HLT	76	Stop the program

Input

Input Address	Value
4300	05

Output

Output Address	Value
4301	00
4302	01
4303	01
4304	02
4305	03

00 + 01 = 01

01+ 01 = 02

02 + 01 = 03

Result:

The assembly language for Fibonaci series was executed successfully using 8085 microprocessor kit.

FACTORIAL OF 8 BIT NUMBER

Aim:

To write an program to calculate the factorial of a number (between 0 to 8)

Apparatus required:

8085 microprocessor kit

(0-5V) power supply

Memory Location	Hex Code	Label	Mnemonics		Comments
			Op code	Operand	
4200 4201 4202	3A 50 42		LDA	4250	Get the number in accumulator
4203 4204	FE 02		CPI	02H	Compare data with 2 and check it is greater than 1
4205 4206 4207	DA 17 42		JC	Loop 1	If cy =1 jump to loop 1 If cy = 0 proceed
4208	5F		MOV	E,A	Move content of A to E
4209 420A	16 00		MVI	D,00	Load this term as a result
420B	3D		DCR	A	Decrement accumulator by 1
420C	4F		MOV	C,A	Move 'A' content to 'C' (counter 1 less than A)
420D 420E 420F	CD 00 46		CALL	Facto	Call sub routine programe Facto

4210	EB		XCHG		Exchange (DE) – (HL)
4211 4212 4213	22 51 42		SHLD	4251	Store content of HL in specified memory location
4214 4215 4216	C3 1D 42		JMP	Loop 3	Jump to Loop 3
4217 4218 4219	21 00 01	Loop 1	LXI	$H,0001_H$	HL is loaded with data 01
421A 421B 421C	22 51 42		SHLD	4251	Store the result in memory
421D	76	Loop 3	HLT		Terminate the program
Sub Routine					
4600 4601 4602	21 00 00	Facto	LXI	H,0000	Initialize HL pair
4603	41		MOV	B,C	Content of 'C' is moved to B
4604	19	Loop 2	DAD	D	Content of DE is added with HL
4605	05		DCR	B	'B' is decremented
4606 4607 4608	C2 04 46		JNZ	Loop 2	Multiply by successive addition till zero flag is set

4609	EB		XCHG		[DE] – [HL]
460A	0D		DCR	C	Decrement counter value
460B 460C 460D	C4 00 46		CNZ	Facto	Call on no zero to facto (i.e repeat process till zero flag for c = 1)
460E	C9		RET		Return to main program

Memory address	Content
4250	04
4251	18

1 x 2 x 3 x 4 = 24

Hexadecimal

16 | 24

1-8

Result:

Thus, factorial program was done successfully

16 – BIT MULTIPLICATION

Aim:

To write an assembly language program for 16 bit multiplication by using 8085 microprocessor kit.

Apparatus required:

8085 microprocessor kit

(0-5V) DC battery

Memory Location	Hex Code	Label	Mnemonics		Comments
			Op code	Operand	
4100	2A,00,42		LHLD	4200	Get the 1st data in HL
4103	F9		SP HL		Save it in stack pointer4106
4106	2A,02,42		LHLD	4202	Get the 2nd data in HL
4107	EB		XCHG		Exchange 'HL' and 'DC'
4108	21,00,00		LXI H	0000	Make HL – 0000
410B	01,00,00		LXI B	0000	Make BC – 0000
410E	39	Next	DAD	SP	Add 'SP' and 'HL'
410F	D2, 13, 41		JNC	Loop	Jump to loop if no carry
4112	03		INX	B	Increment 'BC' by one
4113	1B	Loop	DCX	D	Decrement 'DE' by one
4114	7B		MOV	A,E	Make E – A
4115	B2		ORA	D	'OR' gate between A & D
4116	C2,0E,41		JNZ	Next	Jump on if number zero
4119	22,04,42		SHLD	4204	Store the LSB in

					memory
411C	69		MOV	L,C	Make C to L
411D	60		MOV	H,B	Make B to H
411E	22,06,42		SHLD	4206	Store the MSB in memory
4121	76		HLT		Stop the program

Input

Input Address	Value
4200	04
4201	07
4202	02
4203	01

Output

Output Address	Value
4204	08
4205	12
4206	01
4207	00

Result:

Thus the assembly language program for 16 bit multiplication was executed successfully.

16 – BIT DIVISION

Aim:

To write an assembly language program for 16 bit division in 8085 microprocessor.

Apparatus required:

8085 microprocessor kit

(0-5V) DC battery

Memory Location	Hex Code	Label	Mnemonics		Comments
			Op code	Operand	
4100	2A,00,42		LHLD	4200	Get the 1st data in HL
4103	F9		SP HL		Save it in stack pointer4106
4106	2A,02,42		LHLD	4202	Get the 2nd data in HL
4107	EB		XCHG		Exchange 'HL' and 'DC'
4108	21,00,00		LXI H	0000	Make HL – 0000
410B	01,00,00		LXI B	0000	Make BC – 0000
410E	39	Next	DAD	SP	Add 'SP' and 'HL'
410F	D2, 13, 41		JNC	Loop	Jump to loop if no carry
4112	03		INX	B	Increment 'BC' by one
4113	1B	Loop	DCX	D	Decrement 'DE' by one
4114	7B		MOV	A,E	Make E – A
4115	B2		ORA	D	'OR' gate between A & D
4116	C2,0E,41		JNZ	Next	Jump on if number zero

4119	22,04,42		SHLD	4204	Store the LSB in memory
411C	69		MOV	L,C	Make C to L
411D	60		MOV	H,B	Make B to H
411E	22,06,42		SHLD	4206	Store the MSB in memory
4121	76		HLT		Stop the program

Input

Input Address	Value
4200	04
4201	07
4202	02
4203	01

Output

Output Address	Value
4204	08
4205	12
4206	01
4207	00

Result:

Thus the assembly language program for 16 bit multiplication was executed successfully.

BINARY TO BCD CONVERSION

Aim:

To write an assembly language program to convert an 8 bit binary data to BCD using 8085 microprocessor kit.

Apparatus required:

8085 microprocessor kit

(0-5V) power supply

Memory Location	Hex Code	Label	Mnemonics		Comments
			Op code	Operand	
4100	0E,00		MVI	E,00	Clear 'E' register (Hund)
4102	53		MOV	D,E	Clear 'D' register (tens)
4103	3A,00,42		LDA	4200	Get the data in 'A'
4106	C3,06,41	HUND	CPI	64	Compare the data with 64
4108	DA,11,41		JC	TEN	If content is less jump to ten
410B	D6, 64		SUI	64	Subtract data by 64
410D	IC		INR	E	Increment carry each time
410E	C3,06,41		JMP	HUND	Jump to hundred & repeat
4111	C3, 0A	TEN	CPI	0A	Compare the data with 0A
4113	DA,1C,41		JC	UNIT	If data is less jump to unit
4116	D6, 0A		SUI	0A	Subtract the data by 0A
4118	14		INR	D	Increment 'D' each time

4119	C3,11,41		JMP	TEN	Jump to ten & repeat
411C	4F	UNIT	MOV	4A	Move the value 'A' to 'C'
411D	7A		MOV	A,D	Move the value 'D' to 'A'
411E	07		RLC		Rotate the value of 'A'
411F	07		RLC		Of 'A' so that
4120	07		RLC		Lower and upper niddle
4121	07		RLC		Gets exchanged
4122	81		ADD	C	Add 'A' and 'C'
4123	32,50,42		STA	42,50	Save ten' & units in 'M'
4126	7B		MOV	A,E	Move to E to A
4127	32,51,42		STA	4251	Save hundreds unit in 'A'
412A	76		HLT		Stop the program execution

Input

Input Address	Value
4200	54

Output

Output Address	Value
4250	84
4251	00

Result:

Thus the binary to BCD conversion was executed successfully

BCD TO BINARY

Aim:

To write an assembly language program to convert BCD data to Binary data using 8085 microprocessor kit.

Apparatus required:

8085 microprocessor kit

(0-5V) power supply

Memory Location	Hex Code	Label	Mnemonics		Comments
			Op code	Operand	
4100	3A,00,42		LDA	4200	Get the data in 'A'
4103	5E		MOV	E,A	Save in 'E' register
4104	E6, F0		ANI	F0	Mark the lower nibble
4106	07		RLC		Rotate the upper
4107	07		RLC		To lower nibble
4108	07		RLC		And save in
4109	07		RLC		Register B
410A	47		MOV	B,A	Move it from 'A' to 'B'
410B	AF		XRA	A	Clear the accumulator
410C	0E,0A		MVI	C,0A	Intialise 'C' as '0A'
410E	08		REP		
410F	0D		DCR	C	Decrement 'C' register
4110	C2,0E,41		JNZ		Jump till value 'C' is 0
4113	47		MOV	B,A	Move the value A to B
4114	7B		MOV	A,E	Get the BCD in 'A'
4115	E6, 0F		ANI	0F	Mark the upper nibble
4117	80		ADD	B	Add 'A' and 'B'
4118	32,01,42		STA	4201	Save the binary data
411B	76		HLT		Stoptheprogram execution

Input

Input Address	Value
4200	68

Output

Output Address	Value
4201	44

16 | 68

4-4

Result:

Thus the BCD to binary conversion was executed successfully

ROLLING DISPLAY

Aim:

To write an assembly language program to obtain a rolling display of a particular data by using 8085 microprocessor

Apparatus required:

8085 micro processing kit

(0-5V) power supply

Memory Location	Hex Code	Label	Mnemonics		Comments
			Op code	Operand	
4500		MVI	A,00	3E,00	Initialise A 00
4502		OUT	01	DE, 01	Control word through 8 bit
4504		MVI	A,90	3E, 90	A = RAM cw
4506		OUT	01	DE,01	Output cw through 8 bit port
4508		MVI	A,CC	3E,CC	A = CC
450A		OUT	01	DE,01	Output cw through 8 bit port
450C	Loop 2	LXI	H,5000	21,00,50	Memory -> HL location
450F		MOV	C,M	4E	M -> C
4510	Loop 1	INX	H	23	Increment 'HL'
4511		MOV	A,M	7E	Move 'H' to 'A'
4512		OUT	00	DE, 00	Output the character
4514		CALL	Loop	CD,00,46	Call the subroutine
4517		DCR	C	0D	Decrement 'C' by one
4518		JNZ	Loop 1	C2,10,45	Jump on no zero
451B		JMP	Loop 2	C3,0C,45	Jump to L2
4600	Loop	LXI	D,FFFF	11,FFFF	Load DE-FFFF

4603	Loop 3	DCX	D	1B	Decrement ‘DE’ by 1
4604		MOV	A,D	7A	Move ‘D’ to ‘A’
4605		ORA	E	B3	(A) = (A) check
4606		JNZ	Loop 3	C2,03,46	Jump on no zero
4609		RET		C9	Return to main program

Input

Input Address	Value
5000	06
5001	98
5002	68
5003	7A
5004	C8
5005	1A
5006	2C

Output

HELPUS

Result:

Thus, an assembly language program to obtain rolling display of a particular value written using 8085 microprocessor kit.

FLASHING DISPLAY

Aim:

To write an assembly language program to obtain the following flashing display of a particular data.

Apparatus required:

8085 micro processing kit

(0-5V) power supply

Memory Location	Hex Code	Label	Mnemonics		Comments
			Op code	Operand	
4300		MVI	A,00	3E,00	Intialise 'A' as '00'
4302		OUT	01	DE,01	Out the control word trough 8 bit
4304		MVI	A,90	3E,90	Intialise 'a' with cw for RAM
4306		OUT	01	D3,01	Out the cw
4308		MVI	A,CC	3E,CC	A = CC
430A		OUT	01	0D,01	Out the cw
430C	Loop 2	LXI	H,5000	21,00,50	Load 'HL' with
430F		MOV	C,M	4E	M to C
4310	Loop 1	INX	H	23	Increment 'H' by
4311		MOV	A,M	7E	Move M to A
4312		OUT	00	D3, 00	Out the character
4314		DCR	C	0D	Decrement 'C' by 1
4315		JNZ	Loop 1	C2,10,43	Check for zero
4318		CALL	Delay	C0,00,46	Call subroutine
431B		MVI	A,DC	3E,DC	A <- 0C
431D		OUT	01	D3, 01	A<-01
431F		CALL	Delay	CD,00,46	Call subroutine

4322		JMP	Loop 2	C3 0C,43	Check for zf
4600	Delay	LXI	D,FFFF	11,FF,FF	Intialise DE=FFFF
4603	Loop 3	DCX	D	1B	Decrement DE by 1
4604		MOV	A,E	7B	Move 'E' to 'A'
4605		ORA	D	B2	Check 'De' = '00'
4606		JNZ	Loop 3	C2,03,46	Jump on no zero
4609		RET	C9	C9	Return to main program

Input

Input Address	Value
5000	05
5001	68
5002	68
5003	68
5004	FD
5005	88

Output

EEE – A

Result:

Thus, an assembly language program to obtain flashing display of a particular data was written using 8085 microprocessor kit.

SPEED CONTROL OF STEPPER MOTOR

Aim:

To write an assembly program to make the stepper motor run in forward and reverse direction.

Apparatus required:

Stepper motor

8085 microprocessor kit

(0-5V) power supply

Memory Location	Hex Code	Label	Mnemonics		Comments
			Op code	Operand	
4100	Start	LXI	H,Look up	21,1A,41	Load the 'HL' with data
4103		MVI	B,04	06,04	B = 04
4105	Repeat	MOV	A,M	7E	Memory value to 'A'
4106		OUT	C0	D3, C0	Display it
4108		LXI	D,03,03	11	Load 'DE' with FFFF
410B	Delay	NOP		00	Start delay loop
410C		DCX	D	1B	Decrement DE by 1
410D		MOV	A,E	7B	Move 'E' to 'A'
410E		ORA	D	B2	Check De = 0 or not
410F		JNZ	DELAY	C2, 0B,41	Jump on zero
4112		INX	H	23	Increment HL by 1
4113		DCR	B	05	Decrement B by 1
4114		JNZ	Repeat	C2,05,41	Jump on no zero
4117		JMP	START	C3,00,41	Jump to start

Input

Input Address	Value
411A	0A
411B	06
411C	05
411D	09

Reverse Direction

Output Address	Value
411A	09
411B	05
411C	06
411D	0A

Result:

Thus, an assembly language program to control of stepper motor was written using 8085 microprocessor kit.

ANALOG TO DIGITAL CONVERTER

Aim:

To write an assembly language program to convert analog to digital signal and to display it in 7 segment LED display

Apparatus required:

8085 microprocessor kit

(0-5V) power supply

Memory Location	Hex Code	Label	Mnemonics		Comments
			Op code	Operand	
5000	3E,10		MVI	A,10	Intialise 'a' with 10
5002	D3,C		OUT	C8	Output channel through
5004	3E,18		MVI	A,18	Intialise 'A' with 18
5006	D3, C8		OUT	C8	Output channel through 8 bit port
5008	00		NOP		No operation
5009	00		NOP		No operation
500A	3E,10		MVI	A,10	Intialise 'A' with 2nd signal
500C	D3,C8		OUT	C8	Output channel through 8 bit port
500E	3E,01	L2	MVI	A,01	Intialise 'A' with 2nd
5010	D3,D0		OUT	D0	Output through 8 bit
5012	00		NOP		
5013	00		NOP		
5014	00		NOP		
5015	3E,00		MVI	A,00	
5017	D3,D0		OUT	D0	
5019	DB,D8	L1	IN	D8	
501B	E6,01		ANI	01	

501D	CA,19,50		JZ	L1	
5020	DB,C0		IN	C0	Get input from
5022	47		MOV	B,A	B -> A
5023	E6,0F		ANI	0F	And of with 'A'
5025	32,51,51		STA	5151	Store in 5151
5028	78		MOV	A,B	B -> A
5029	E6,F0		ANI	F0	And F0 with A
502B	0F		RRC		Rotate content 'A'
502C	0F		RRC		
502E	0F		RRC		
502F	32,50,51		STA	550	Store MSB in 5150
5032	3E,03		MVI	A,03	03 -> A
5034	0E,08		MVI	C,08	08 -> C
5036	21,50,51		LXI H	5150	Load 'HL' pair with 5150
5039	CD,05,00		CALL	0005	Call device subroutine
503C	C3,0E,50		JMP	500E	Jump to 500E

Result:

Thus the analog to digital conversion was done microprocessor.

ARTHMETIC OPERATIONS USING 8051

Aim:

To do the arithmetic operations using 8051 microprocessor

Apparatus required:

8085 microprocessor kit

DAC interface kit

Keyboard

Program: 8-bit Addition:

Memory Location	Label	Opcode	Mnemonics	Comments
4100	Start	C3	CLR C	Clear the carry flat
4101		74DA	MOV A, # data 1	Moves data 1 to register A
4103		24DA	ADD A, # data 2	Add content of A and data 2 and store in A
4105		464500	MOV DPTR, # 4500	Moves data 4500 to DPTR
4108		F0	MOV A @ DPTR, A	Moves control of A to location pointed DTPR
4109		80 FE	SJMP 4109	Short jump to 4109

Execution:

Addition:

ML	Input
4103	0L
4109	03

ML	Output
4500	05

Program: 8-bit Subtraction:

Memory Location	Label	Opcode	Mnemonics	Comments
4100	Start	C3	CLR C	Clear the carry flat
4101		74DA	MOV A, # data 1	Moves data 1 to register A
4103		24DA	SUB B, # data 2	Subtract data 2 from content of A and store result in A
4105		464500	MOV DPTR, # 4500	Moves 4500 to DPTR
4108		F0	MOV X @ DPTR, A	Moves result by location by DTPR
4109		80 FE	SJMP 4109	Short jump to 4109

Execution:

Subtraction:

ML	Input
4101	05
4103	02

ML	Output
4500	03

Result:

Thus 8-bit addition, subtraction is performed using 8051.

ARTHMETIC OPERATIONS USING 8051

Aim:

To do the arithmetic operations using 8051 microprocessor

Apparatus required:

8085 microprocessor kit

DAC interface kit

Keyboard

Program: 8-bit Multiplication:

Memory Location	Label	Opcode	Mnemonics	Comments
4100	Start	7403	MOV A, # data 1	Move immediate data to accumulator
4101		75F003	MOV B, # data 2	Move 2nd data to B register
4105		A4	MUL A B	Get the product in A & B
4106		904500	MOV DPTR, # 4500	Load data in 4500 location
4109		F0	MOV X @DPTR, A	Move A t ext RAM
410B		E5F0	MOV A,B	Move 2nd data in A
410D		F0	MOV A @ DPTR	Same the ext RAM
410E		80FE	SJMP 410E	Remain idle in infinite loop

Execution:

Multiplication:

ML	Input
4101	0L

Program: 8-bit Division:

Output Address	Value
4500	08

Memory Location	Label	Opcode	Mnemonics	Comments
4100	Start	7408	MOV A, # data 1	Move immediate data to accumulator
4102		75F002	MOV B, @ data 2 DIV AB	Move immediate to B reg.
4105		84	DIV AB	Divide content of A & B
4106		904500	MOV DPTR, # 4500	Load data pointer with 4500 location
4109		F0	MOV X @ DPTR, A	Move A to ext RAM
410A		A3	INC DPTR	Increment data pointer
410B		ESF0	MOV A,B	Move remainder to A
410D		F0	MOV @ DPTR, A	Move A to ext RAM
410E		80FE	SJMP 410E	Remain idle in infinite loop

Execution:

Division:

ML	Input
4101	08
4103	04

Output Address	Value
4500	02

Result:

Thus 8-bit multiplication & division is performed using 8051