

Madhav Institute of Technology and Science, Gwalior

A Govt. Aided UGC Autonomous & NAAC Accredited Institute

(Affiliated to R.G.P.V. Bhopal)

Department of Electronics Engineering



Lab Manual

MICROPROCESSOR & INTERFACING

**(ELECTRONICS VI SEM / ELECTRONICS &
TELECOMMUNICATION V SEM)**

(140601/200503)

Subject Name: Microprocessors and Interfacing
Subject Code: 140601/200503

L	T	P	C
-	-	2	1

Course Objectives

This course gives the ability to students to learn the assembly language programming of 8085, 8086 microprocessor and 8051 microcontroller and their interfacing with different peripherals.

List of Experiments

1. Write an assembly language program to perform different arithmetic operation on 8 bit numbers.
2. Write an assembly language program to find whether the number is even or odd.
3. Write an assembly language program to interface stepper motor with 8085.
4. Write an assembly language program to generate standard waveforms using DAC and display waveforms on CRO with 8085.
5. Write an assembly language program to obtain 2's complement of a number.
6. Write an assembly language program to perform arithmetic operations of two BCD numbers.
7. Write an assembly language program to interfacing 8253 Timer with 8086 in different modes.
8. Write an assembly language program to interface ADC card with 8051 Microcontroller and display the digital value of the LCD.

Value added Experiments:

9. Write an assembly language program to interfacing temperature measurement card with 8086 and program to display the temperature on LCD.
10. Write an assembly language program to interface 7 segment display with 8051 Microcontroller.

Course Outcomes:

After successful completion of the course, students will be able to:

- CO1. **Develop** the assembly language programs for the various arithmetic, logical, and string operations.
- CO2. Design interfacing circuits for different I/O devices using PPIs with 8085, 8086 microprocessors and 8051 microcontroller.

PROCEDURE FOR OPERATING DYNA 85LU 8085 KIT:

Procedure: -

- Step 1. Connect the power supply and keyboard to 8085 kit.
- Step 2. Turn ON the power supply, the LCD on the board will display as **DYNA85>**
- Step 3. Now Specify the location in the memory where the program would be entered as: **DYNA85> A [Memory location]**. e.g. **DYNA85> A C000**.
- Step 4. Now enter the program through keyboard. For next line press enter.
- Step 5. After entering the complete program the last line should be entered as **RST1**.
- Step 6. Now press Enter two times to come out of program entering mode and LCD should display **DYNA85>**
- Step 7. Before executing the program if the data needs to be saved in the memory then following command needs to be executed:
DYNA85> E [Memory location] and enter the data at that location.
- Step 8. Now, to execute the program:
DYNA85> G [Starting address] [Ending address] press enter and space bar.
- Step 9. To view the result in registers. Press R as **DYNA85>R**. This will show the value of all registers.

EXPERIMENT NO : 1

AIM:- Write an assembly language program to perform different arithmetic and logical operations on 8 bit and 16 bit operations.

APPARATUS:- Dyna 8085 microprocessor kit.

Program:

(a) Using Immediate Data (8 Bit):

ADDRESS	OPCODE	MNEMONIC	OPERAND	COMMENTS
C000	3E Data1	MVI	A, Data1	Get first byte in the accumulator
C002	06 Data2	MVI	B, Data2	Get Second byte in register B
C004	88	ADD	B	Addition of Data1 (A) + Data2 (B). Result is saved in accumulator A.
C008	76	HLT (RST 1)		Halt the program

- Press Space and Enter
- For results check the contents of the register A as:
Press R as **DYNA85>R.**

Example:

Data 1	Data 2	Result
A = 2	B = 3	A = 5 (Accumulator)

(b) Program to add two 8 bit numbers using data stored in memory:

Algorithm:

- Save the memory location of first number (i.e. C0B1) in HL register pair.
- Move first number/data into accumulator
- Increment the content of HL register pair to get the memory location of second data. (i.e. C0B2)
- Add the second data with accumulator
- Store the result in memory location C0B3H.
- Store the carry in memory location C0B4H.

ADDRESS	OPCODE	LABEL	MNEMONIC	OPERAND	COMMENTS
C000	21,B1,C0		LXI	H, C0B1 H	Get address of 1 st number in H-L pair
C003	7E		MOV	A, M	Shift 1 st no. in accumulator
C004	23		INX	H	Increment content of H-L Pair to point to 2 nd number.
			MOV	B, M	Move 2 nd no into Reg. B
C005	86		ADD	B	1 st no. + 2 nd no., result is stored in accumulator
C006	23		INX	H	Increment content of H-L Pair
C007	77		MOV	M, A	Move result at address C0B3 H
C00E	23		INX	H	Move to next location
C00F	00 36		MVI	M, 00H	Store 0 if carry is not generated
C010			JNC	Label1 (C014)	
C012	01 36		MVI	M, 01H	Store 1 as carry
C014	76	Label1:	HLT	RST 1	Halt

Data given at specified memory locations:

Address	Data	Comments	Example
C0B1	DATA1	1 st no. to be added	5
C0B2	DATA2	2 nd no. to be added	2
C0B3	RESULT	Data1 + Data2	3
C0B4	CARRY	Carry is stored as either 0 or 1	0

Viva Questions:

1. Name the important CPU registers in the 8085 microprocessor?
2. Which are the addressing modes of 8085?
3. What will be the status of carry and zero flags when instruction SUB A is executed?
4. What is the use of ALE pin of 8085?
5. What is Clock Speed ?

EXPERIMENT NO.: -2

AIM: - Write an assembly language program to find whether the number is even or odd.

Apparatus: - 8085 microprocessor kit.

Algorithm –

1. Load the content of memory location C0B0 in accumulator A.
2. Perform AND operation with 01 in value of accumulator A by the help of ANI instruction.
3. Check if zero flag is set, i.e. if ZF = 1 then store 0E in accumulator A otherwise store 0D in A.
4. Store the value of A in memory location C0B1

Program:

ADDRESS	OPCODE	LABEL	MNEMONIC	OPERAND	COMMENTS
C000			LDA	C0B0	Move the number from memory location C0B1 to accumulator
C003			ANI	01	Perform logical ANDING operation of number with 01
C005			JZ	C00D	Jump to location C00D if ZF = 1 i.e. number is Even
C008			MVI	A, 0D	Move '0D' in A
C00A			JNZ	C00F	Jump to location C00F if ZF = 1 i.e. number is ODD
C00D			MVI	A, 0E	Move '0E' in A
C00F			STA	C0B1	Store the contents of A at location C0B1

C010			HLT	RST 1	
------	--	--	-----	-------	--

Result: - Check the contents of accumulator, if A = 0D then the given number is odd and if A=0E it is even.

Viva Questions:

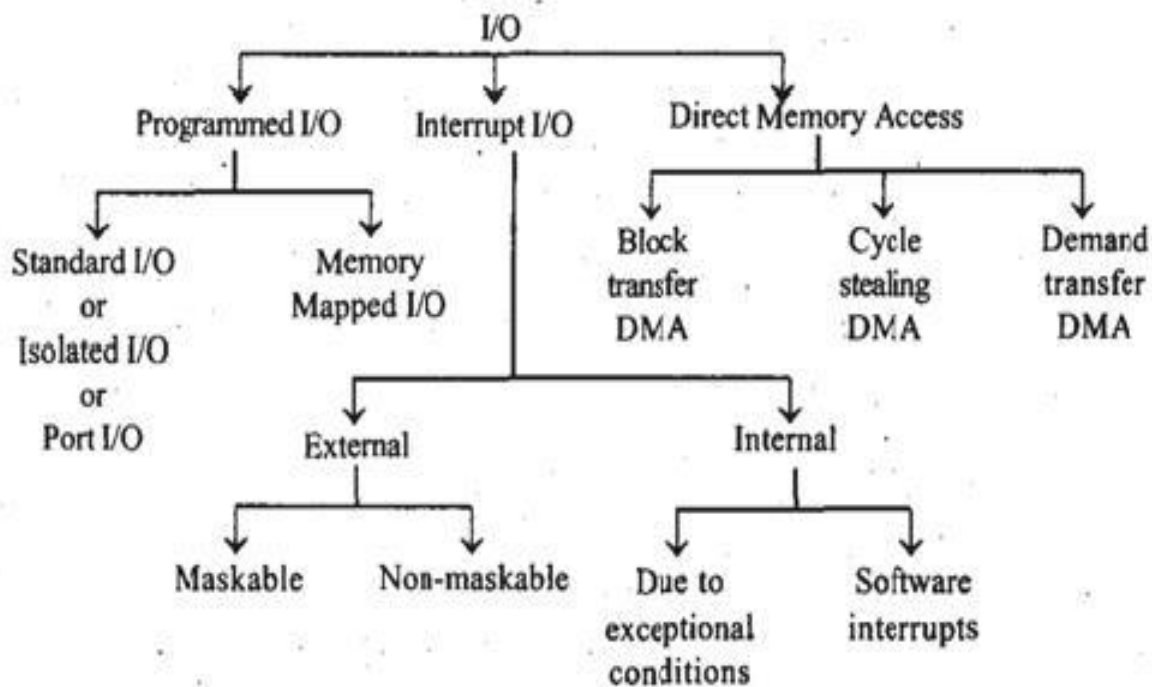
1. State the use of LDA instruction?
2. Which Stack is used in 8085?
3. In 8085 name the 16 bit registers?
4. What are the various flags used in 8085?
5. What is Stack Pointer?

I/O INTERFACING WITH 8085

I/O STRUCTURE OF A TYPICAL MICROCOMPUTER:

There are three major types of data transfer between the microcomputer and an I/O device. They are,

- Programmed I/O : In programmed I/O the data transfer is accomplished through an I/O port and controlled by software.
- Interrupt driven I/O : In interrupt driven I/O, the I/O device will interrupt the processor, and initiate data transfer.
- Direct memory access (DMA) : In DMA, the data transfer between memory and I/O can be performed by bypassing the microprocessor.



EXPERIMENT NO.: 3

AIM: - Write an assembly language program to interface stepper motor with 8085.

Apparatus: - 8085 microprocessor kit.

Stepper motor Interfacing/Control using 8085

Stepper Motor

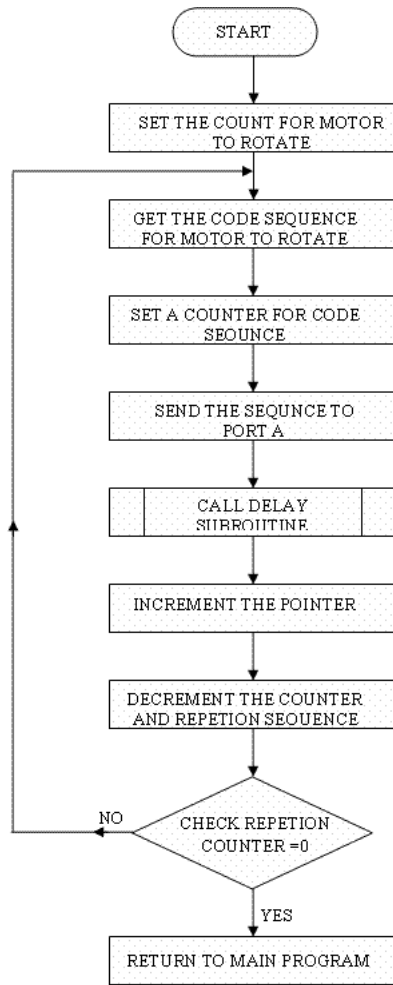
A stepper motor is a device that translates electrical pulses into mechanical movement in steps of fixed step angle.

- The stepper motor rotates in steps in response to the applied signals.
- It is mainly used for position control.
- It is used in disk drives, dot matrix printers, plotters and robotics and process control circuits.

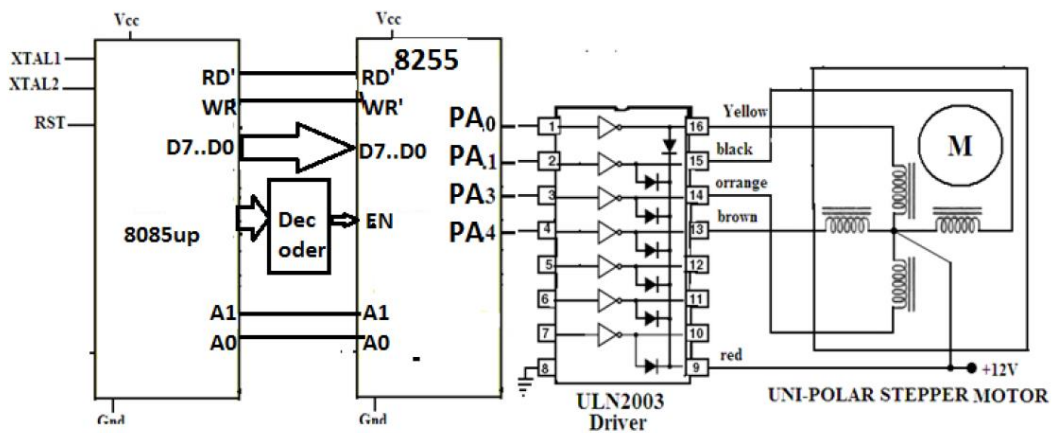
Operation

The important parameter of a stepper motor is the step angle. It is the minimum angle through which the motor rotates in response to each excitation pulse. In a four phase motor if there are 200 steps in one complete rotation then then the step angle is $360/200 = 1.8^{\circ}$. So to rotate the stepper motor we have to apply the excitation pulse. For this the controller should send a hexa decimal code through one of its ports. The hex code mainly depends on the construction of the stepper motor. So, all the stepper motors do not have the same Hex code for their rotation.

The flow graph for the process of interfacing is shown as below:



Flow graph of the interfacing program



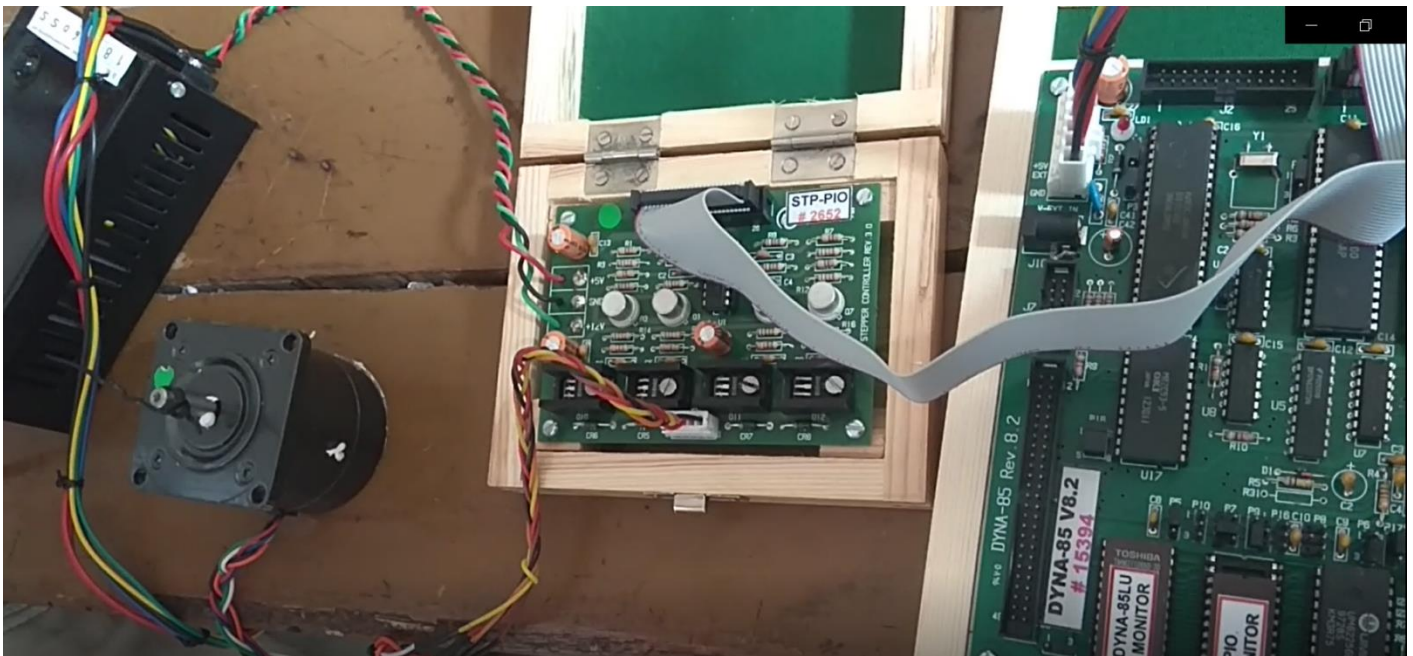
Stepper Motor interface - Schematic Diagram for (8085)

Procedure:

Locate 26 pin connector J3 on DYNA-85LU kit. All the lines of the 8255 PIO are brought out on this connector.

The PIO cards are interfaced through 26 pin FRC flat cable.

1. Connect the PIO card to DYNA-85LU (J3) through 26 pin FRC flat cable as shown below:



2. The program for all the PIO cards are given in an EPROM (2732) labeled PIO-MON. Place EPROM in socket U3 of DYNA_85LU kit. Do the appropriate jumper setting for 2732 EPROM.
3. The Program can be entered in two ways:
 - a. Copy the program from the specified RAM location as:
DYNA-85> C 4700 49FF C000 Enter
 - b. Enter the program given as below:
4. Run the program from C000.

PROGRAM:

The assembly program, when executed will display the message “*Stepper mode (0 or 1)*”. If 0 is given, the motor will run continuously and if 1 is given, the motor will run for a specified no. of steps, direction and speed.

0 Continuous Mode:

In this mode, motor will run continuously at the maximum speed and in the clockwise direction. To stop it, Press ESC key.

1 Step Mode:

The program will display message 'stepper mode (0 or 1)'. Enter 1 to switch to step mode. Then the program will display 'rP (0 or 1)' asking for the Rotational pattern. Here 0 is for clockwise direction and 1 is for anti-clockwise direction. Then 'steps' will be displayed asking the user to enter the no. of steps (0000-FFFF). Then it will display 'Speed', asking for the speed (00-FF). After entering the value of rotational motion, steps and speed, the motor will run at the defined rotational motion, no. of steps and speed, and then it will stop.

In the program 'cts' routine controls the motor in continuous Mode and 'step' routine controls the motor in continuous mode and 'step' routine controls the Motor in Continuous Mode and 'step' routine controls the motor in clockwise direction and 'anti' routine runs it in anticlockwise direction. To display all the different messages corresponding ASCII codes for each message is stored in the lookup table. The program is given below:

Program:

ADDRESS	OPCODE	LOCATION	COMMAND	REAMRKS
C000	21 00 C1		MVI A, 80	80H → Control word to configure 8255 PA,PB,PC in O/P mode
C003	CD 44 18		OUT	Write control word in CWR of 8255
C006	Cd 1f 06	UP:	MVI A, 77	Code for the Phase 1
C009	79		OUT	sent to motor via port A of 8255
C00A	FE 30		CALL DELAY	Delay subroutine
C00C	CA 94 C0		MVI A, BB	Code for the Phase II
C00F	FE 31		OUT	sent to motor via port A of 8255
C011	CA 1C C0		CALL DELAY	Delay subroutine.
C014	FE 1B		MVI A, DD	Code for the Phase III
C016	C2 06 C0		OUT	sent to motor via port A of 8255
C019	CD C4 0F		CALL DELAY	Delay subroutine
C01C	21 80 1C		MVI A, EE H	Code for the Phase 1
C01F	CD 44 18		OUT	sent to motor via port A of 8255
C022	CD 26 06		CALL DELAY	Delay subroutine
C025	79		JMP UP	Keep the motor rotating continuously
Delay subroutine				

C029	78		MVI C, FF	Load C with FF -- Change it for the
C02A	32 31 C2		LOOP1: MVI	Load D with FF
C02D	21 40 C1		LOOP2: DCR D	Decrement D
C030	CD 44 18		JNZ LOOP2	Continue decrementing till D=0
C033	CD 26 06		DCR C	Decrement C
C036	79		JNZ LOOP1	Continue decrementing till C=0
C037	32 32 C2		RET	Return to main program

Result:

1. After proper interfacing and execution of the program the User will be asked to choose option 'stepper mode (0 or 1)'. Press '0' or '1' from keyboard accordingly.	
<u>'0' Continuous Mode:</u>	<u>'1' Step Mode:</u>
Motor will run continuously at the maximum speed and in the clockwise direction	<p>2. The user will be asked to choose the rotation pattern as 'rP (0 or 1)'. Press '0' for Clock-wise or '1' for anti-clockwise rotation.</p> <p>3. Then user will be prompted for the step options as 'STEPS'. The user can enter the number of steps from range '0000' to 'FFFF'. Enter Value through keyboard and press Enter.</p> <p>4. Further, user will be prompted for the speed option as 'SPEED'. The user can enter the speed range from '00' to 'FF'. Press Enter.</p>
2. The motor will run for specified steps, speed and direction.	

Viva Questions:

1. What are the internal devices of 8255?
2. What is a programmable peripheral device?
3. What are the information that can be obtained from the status word of 8255?
4. What is stepper motor?
5. What are the different modes of operation of stepper motor?

EXPERIMENT NO.: -4(a)

AIM: - To generate positive going ramp (or sawtooth) using 8085 microprocessor.

Apparatus: - 8085 microprocessor kit.

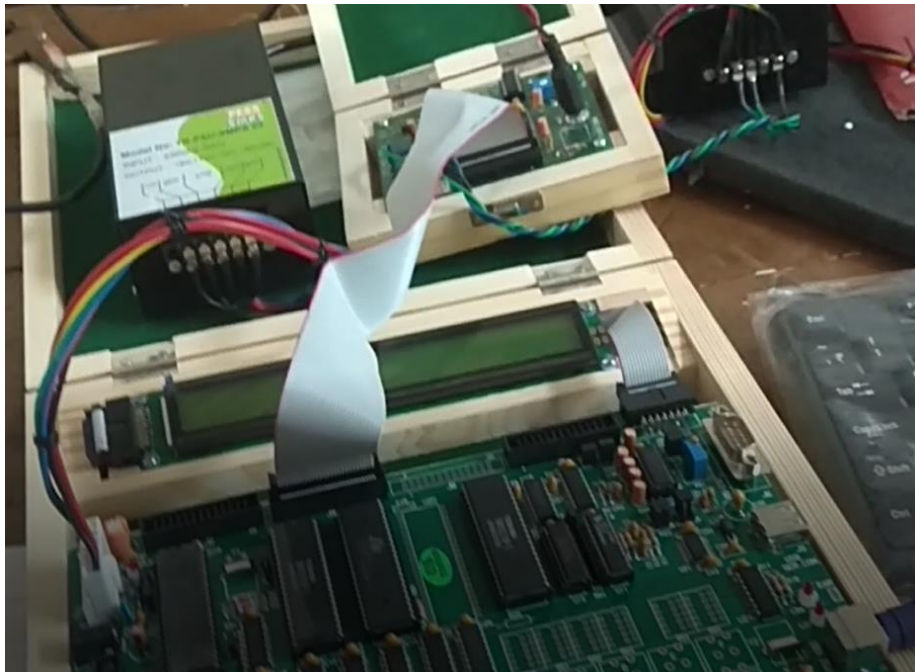
Theory:- This program generates a stair case. If we continuously increment a 8 bit counter which is connected to ADC, then it will generate 255 voltage steps of a staircase at its analog output as the counter starts from 0 to 255. After 255th count , the counter returns to zero and again starts incrementing 0 to 255, so DAC generates 255 steps staircase (or ramp), which can be observed on the oscilloscope.

Procedure:

There is a 26 pin connector J3. All the lines of 8255 PIO are brought out on this connector.

The PIO cards are interfaced through 26 pin FRC flat cable.

1. Connect the PIO card to DYNA -85L (J3) through 26 pin FRC flat cable as shown below:



2. The program for all the PIO cards are given in an EPROM (2732) labeled PIO-MON. Place EPROM in socket U3 of DYNA_85L kit. Do the appropriate jumper setting for 2732 EPROM (Check DYNA-85L Manual for details).
3. The Program can be entered in two ways:
 - c. Copy the program from the specified RAM location as:
DYNA-85> C 4E00 4EFF C000 Enter
 - d. Enter the program given as below:
4. Run the program from C000.

Code for display the message “Program for ramp generation”

DYNA85LU> E C050 [Press Enter]

C050: 0D 0A 50 52 4F 47 52 41 4D 20 46 4F 52 20 44 41
 C060: 43 0D 0A 52 41 4D 50 20 47 45 4E 45 52 41 54 4F
 C070: 52 00

[Press Enter]

PROGRAM:

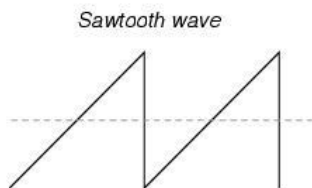
DYNA85LU> A C000 [Press Enter]

ADDRESS	OPCODE	LABEL	Mnemonics	Remarks
C000	21 50 C0		LXI H, C050	Load Message in HL
C002	CD 44 18		CALL 1844	Call display subroutine.
C006	3E 89		MVI A,89h	8255 mode set to MODE 0
C008	D3 13		OUT 13	
C00A	3E 01		MVI A,01	Initialize A with 01
C00C	D3 11		OUT 11	
C00E	3C	UP:	INR A	Increment step
C00F	D3 10		OUT 10	Output to DAC
C011	C3 0E C0		JMP UP	Go to label UP
C014			RST1	Hold the program

Press Enter Key

DYNA85LU> G C000 [Press Enter Key]

WAVEFORM:



CALCULATION:

Amplitude:

Time Period:

RESULT: The Sawtooth Waveform was generated and displayed on CRO as:



EXPERIMENT NO.: -4(b)

AIM: - To generate Triangular waveform using 8085 Microprocessor.

Code for display the message “Program for triangular wave generation”

DYNA85LU> E C050 [Press Enter]

```
C080 0D 0A 50 52 4F 47 52 41 4D 20 46 4F 52 20 44 41  
C090 43 0D 0A 54 52 49 41 4E 47 55 4C 41 52 20 57 41  
C0A0 56 45 20 47 45 4E 45 52 41 54 4F 52 00
```

[Press Enter]

Procedure:

There is a 26 pin connector J3. All the lines of 8255 PIO are brought out on this connector.

The PIO cards are interfaced through 26 pin FRC flat cable.

1. Connect the PIO card to DYNA -85L (J3) through 26 pin FRC flat cable as shown below:



2. The program for all the PIO cards are given in an EPROM (2732) labeled PIO-MON. Place EPROM in socket U3 of DYNA_85L kit. Do the appropriate jumper setting for 2732 EPROM (Check DYNA-85L Manual for details).
3. The Program can be entered in two ways:
 - e. Copy the program from the specified RAM location as:
DYNA-85> C 4E00 4EFF C000 Enter
 - f. Enter the program given as below:
4. Run the program from C020.

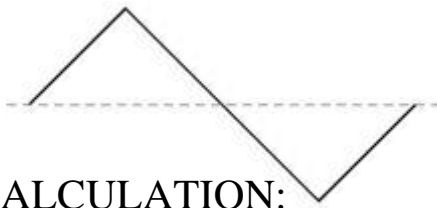
PROGRAM:

DYNA85LU> A C020 [Press Enter]

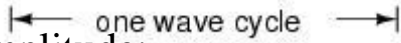
ADDRESS	OPCODE	LABEL	Mnemonics	Remarks
C020	21 80 C0		LXI H, C080	Load Message in HL
C023	CD 44 18		CALL 1844	Call display subroutine.
C026	3E 89		MVI A,89h	8255 mode set.
C028	D3 13		OUT 13	MODE 0
C02A	3E 01		MVI A,01	Initialize A by 0
C02C	D3 11		OUT 11	Move the value to 8255
C02E	3E 00		MVI A, 00	Clear ACC.
C030	3C	UP:	INR A	Increment step counter Up to FF
C031	D3 10		OUT 10	
C033	FE FF		CPI FF	
C035	C2 30 C0		JNZ UP	Decrement Step Counter
C038	3D	NEXT:	DCR A	
C039	D3 10		OUT 10	From FFH to 00H
C03B	FE 00		CPI 00	
C03D	C2 38 C0		JNZ NEXT	
C040	C3 30 C0		JMP UP	Do again
C043			RST1	Hold the program
Press Enter Key				
DYNA85LU> G C000 [Press Enter Key]				

WAVEFORM:

Triangle wave



CALCULATION:

Amplitude:  ← one wave cycle →

Time Period:

RESULT: The Triangular Waveform was generated and viewed in the CRO as:



Viva Questions:

1. What are the internal devices of 8255?
2. What is a programmable peripheral device?
3. What are the information that can be obtained from the status word of 8255?
4. What is stepper motor?
5. What are the different modes of operation of stepper motor?

8086 Based Programs

Steps to Insert and Execute a program in M86-02 Dynalog Trainer kit

Steps:-

1. Power on the trainer.
2. A message display on the display “[M86-02]”
3. Now first press “Reset” button then “enter” key.
4. The menu display “> A. D. F. G. I. M. P. T. U.”
5. This kit is working in two modes “ASSEMBLE MODE AND DESSEMBLE MODE” for assemble mode press “A” then starting add. Of the program then “ENTER” starting address of your program, by default Ram starting address start from “0400H”.
6. Now enter UP TO END of the program. Your program and press “ENTER” and the memory location automatically increment.
7. Now Press entering input of program press “D” key from keyboard and input data add. Memory then “enter” key. “GO” from keyboard then starting add. Of the program is 0400.then enter key.
8. To now to execute program examine result observe the memory locations where data is manipulated. Press “D” address out data location and press “enter” data is displayed.
9. *If we want to write the program through opcode*
10. Just enter “D”then starting address of program and after that press “ENTER” key from keyboard up to end address of the programmed and also give the input data add. of the program.
11. A Press “GO” from keyboard then starting add. Of the program is 400 then enter key.
12. Now to execute program examine result observe the memory locations where data is manipulated. Press “D” address out data location and press “enter” data is displayed.

EXPERIMENT NO: 5

AIM: - Write an assembly language program to find out the 2' S complement.

APPARATUS: - Dyna 8086 microprocessor kit.

Program:

- Press RES
- Press SEG (EB/AX) 0100 then Press INR
- OFF 0100 then Press INR
- Start entering the Op Codes as:
B8 03 03 F7 D8 CC 00
- Press INR after entering each byte.
- The program corresponding to the above mentioned Op Codes is as below:

Address	Op Code	Mnemonic Operand	Comments
0100	B8 03 03	MOV AX, 0303H	Load Accumulator with data as 0303H
0103	F7 D8	NEG AX	Compute the 2's complement of AX and Store in AX
0105	CC	HLT (INT 3)	Halt the program.

- Press EXEC
- Press GO
- SEG 0100 Press EXEC
- START 0100 Press EXEC
- Br 0105 Press EXEC
- F will be displayed
- View Result in register by pressing REG and AX

Result:

Data	Output
0303H	FCFDH (2's complement of 0303H)

EXPERIMENT NO: 6

AIM: - Write an assembly language program to perform arithmetic operations of two BCD numbers.

APPARATUS: - Dyna 8086 microprocessor kit.

Theory:- Here multiplication of two BCD numbers is demonstrated.

Program:

- Press RES
- Press SEG (EB/AX) 0100 then Press INR
- OFF 0100 then Press INR
- Start entering the Op Codes as:
B0 05 B7 09 F6 E7 D4 0A CC 00
- Press INR after entering each byte.
- The program corresponding to the above mentioned Op Codes is as below:

Address	Op Code	Mnemonic Operand	Comments
0100	B0 05	MOV AL, 5	Load unpacked BCD 05H in AL
0102	B7 09	MOV BH, 9	Load unpacked BCD 09H in BH
0104	F6 E7	MUL BH	Multiply AL with BH, result in AX
0106	D4 0A	AAM	AAM (BCD Adjust After Multiply) instruction adjusts the BCD after multiply.
0108	CC	INT 3 (Hold)	Load Data bytes (which are in location 0500 and 0501 in 16 bit ACC. i.e. (0500) – AH (0501) – AL.

- Press EXEC
- Press GO
- SEG 0100 Press EXEC
- START 0100 Press EXEC
- Br 0108 Press EXEC
- F will be displayed
- View Result in register by pressing REG and AX

Result:**Data**

AL=05H (Unpacked BCD)
AH=09H (Unpacked BCD)

Output

AX=45H (Packed BCD)

Viva Questions:

1. Briefly explain how instruction operations in 8086 can be classified?
2. Briefly explain the pointers and index group of registers?
3. List out the differences between isolated I/O and memory mapped I/O?
4. What is logical address and effective address in 8086?
5. What is the function of BIU and EU in 8086 architecture?

Programs of 8086 Interfacing:

EXPERIMENT NO: 7

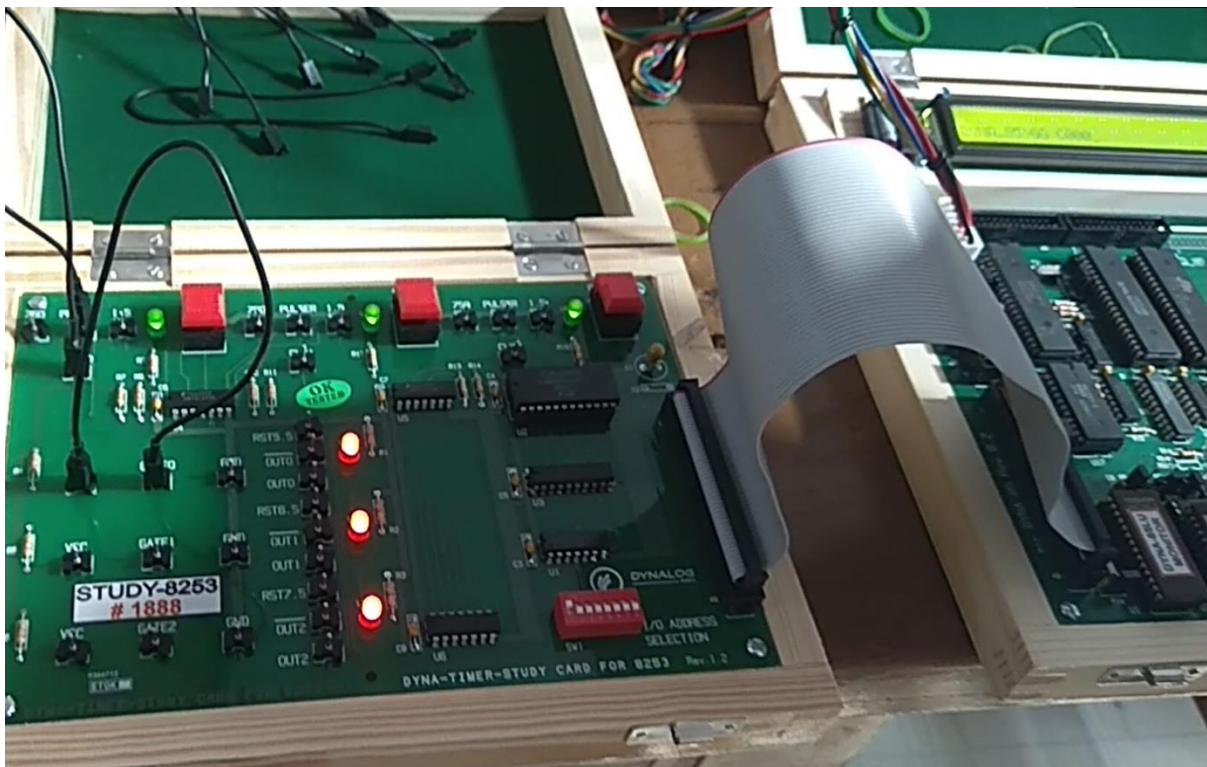
AIM:-Write an assembly language program to interfacing 8253 Timer with 8086 in mode 0.

The Intel **8253** is a Programmable Interval Timers (PITs), which perform timing and counting functions.

The 8253/54 solves one of most common problem in any microcomputer system, the generation of accurate time delays under software control. Instead of setting up timing loops in system software, the programmer configures the 8253 to match his requirements, initializes one of the counters of the 8253 with the desired quantity, then upon command the 8253 will count out the delay and interrupt the CPU when it has completed its tasks. 8253/54 can be operated in 6 different modes with the use of Control word Register

PROCEDURE:

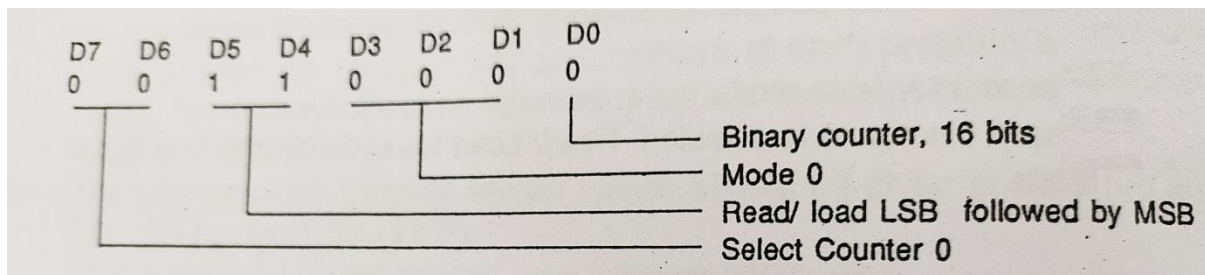
Step 1: “7FH” is the command used to unmask IRQ7. Do the connection as follows:



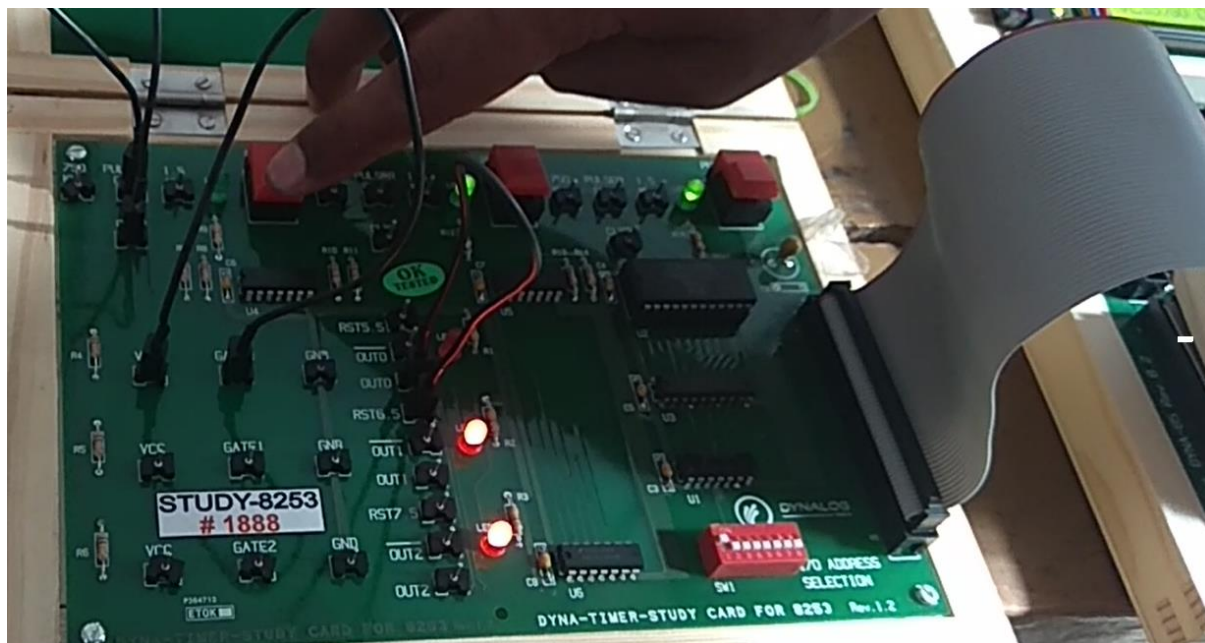
Step 2: “30H” is the control word for 8253. Binary counter 0 is selected. Timer mode is 0. Lower 8-bit count should be loaded first and then higher 8-bit count should be loaded.

Step3: 0005H is the 16-bit count that is loaded into the counter. First the LSB count should be loaded followed by the LSB count.

Step4: “00H” is the control word for 8253 to latch the count. If the 4th and 5th bits of control words are 0 then the count can be latched. MSB and LSB count can be stored in a reg. pair so that it can be read.



Step 5: A pulse can be given to the pulser clock as shown in figure and the output can be observed at the OUT0 pin.



Step 6: After 5 pulses OUT0 goes high generating IRQ7.

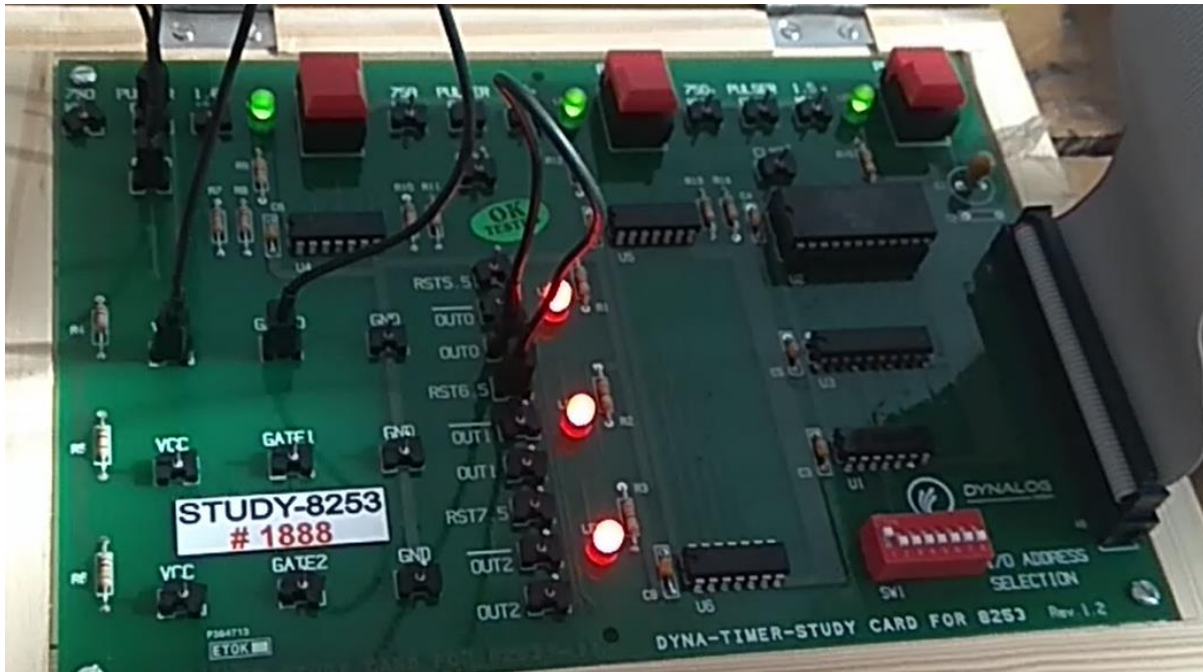
APPARATUS: - Dyna 8086 microprocessor kit and PPI card (8253).

Program:

Address	Op Code	Mnemonic Operand	Comments
1000	B0 7F	MOV AL, 07FH	Unmask IRQ 7
1002	E6 02	OUT 02 (OCW1), AL	Send OCW1
1004	FB	STI	Enable Interrupts
1005	B0 30	MOV AL, 30H	Binary counter_0 selected, mode 0 read / load LSB first & then MSB
1007	E6 33	OUT 33 (CMD_PORT_53), AL	Out the content of AL to CMD_PORT_53
1009	B0 05	MOV AL, 05H	Load 05 to 16 bit counter
100B	E6 30	OUT 30 (COUNTER_0), AL	Counter_0 LSB
100D	B0 00	MOV AL, 00H	Load control word 00H
100F	E6 30	OUT 30 (COUNTER_0), AL	Counter_0 MSB
1011	E4 30	IN AL, 30 (COUNTER_0)	Read LSB
1013	8A D0	MOV DL, AL	Move AL to DL
1015	E4 30	IN AL, 30 (COUNTER_0)	Read MSB
1017	8A F0	MOV DH, AL	Move AL to DH
1019	EB 0D 10	JMP 1101	Jump to address 1011

Result:

LED1 will glow after 5 pulser compression key as shown in figure below.



Viva Questions:

1. State the number of counters that are present in the programmable timer device 8253?
2. Name the mode that is used to interrupt the processor by setting a suitable terminal count is?
3. Name the counter which is selected when in control word register is $SC1=0$ and $SC0=1$?
4. What operation is performed by 8254 when in control word register, $RL1=1$, $RL0=1$?
5. Which mode of 8253 can provide pulse width modulation?

8051 Microcontroller Programming:

PROCEDURE FOR WRITING AND COMPILING PROGRAM THROUGH μ Vision Keil :

Software environment and microcontroller description:

Procedure to write the program in μ Vision Keil:

1. Create a New Project.
2. Give an appropriate name to the project.
3. Select Target as Atmel AT89C52.
4. After creation of project open a new file. Start writing your program.
5. If the program needs to be included then include the program in Target1 as :
Right click on “Source Group1” to add the file as:
(Add file to group “Source Group 1”)
6. Now, file will be included in the target.
7. Right click on Target1 → click option for Target1 → Output Tab → tick the check box “Create the Hex file”.
8. Now build the target & rebuild it.
9. Debug the errors (if any).
10. Open the Flash Magic.
11. Select Device “89LV51RD2” as written on the microcontroller.
12. Connect the microcontroller through USB port.
13. IMP: Check the COM port each time we connect the microcontroller through USB. e.g. ‘COM12’ (Check from device manager).
14. Select the same COM port in the COM port drop down box in the flash magic as given in step 13.
15. Select the baud rate as 9600.
16. Select the check box “Interface Non (ISP)”.
17. Check “Erase all Flash”.
18. Select the Hex file from the directory
19. IMP: make sure go to options → Advance options → Hwdr Config. --> “Use DTR to control RST ” box is unchecked.

20. Now, Make sure that jumper settings are as per 4-bit and 8-bit operation.
(Refer Dyna51 User Manual).
21. Press Start.
22. At bottom of the Flash Magic window “Finished ” will be displayed.

EXPERIMENT NO.: 8(a)

AIM: Write a program to ADD, SUBSTRACT, DIVIDE, MULTIPLY two numbers.

APPRATUS:

1. 8051 micro controller kit.
2. PC for connecting the DYNA51 kit.
3. Keil-U-Vision software for compilation of program.

Program :

```
Org 0000h
mov a,#10h
mov b,#5h
add a,b
mov r1,a
clr a
mov b,#00h
mov a,#20h
mov b,#10h
subb a,b
mov r2,a
clr a
mov b,#00h
mov a,#2h
mov b,#2h
mul ab
mov r3,a
clr a
mov b, #00h
mov a,#21h
mov b,#2h
div ab
mov r4,a
mov r5,b
end
```


EXPERIMENT NO.: 8(a)

AIM: Write an assembly language program to interface LCD and display a string with 8051 Microcontroller.

APPARATUS:

4. 8051 micro controller kit.
5. PC for connecting the DYNA51 kit.
6. Keil-U-Vision software for compilation of program.
7. LCD PIO CARD for DYNA51.

Theory: Read an analog voltage at the input of ADC given as the knob position using 8051 microcontrollers. An Analog to Digital Converter (ADC) is a very useful feature that converts an analog voltage on a pin to a digital number.

The binary counter is initially reset to 0000; the output of integrator reset to 0V and the input to the ramp generator or integrator is switched to the unknown analog input voltage V_A . The analog input voltage V_A is integrated by the inverting integrator and generates a negative ramp output. The output of comparator is positive and the clock is passed through the AND gate. This results in counting up of the binary counter.

OPERATING PROCEDURE FOR ADC CARDS:

1. Connect ADC card to 26 pin frc connector of Dyna-51EB.
2. Connect +12V , -12V and GND supply lines to the ADC card.
3. Connect 16X2 LCD card (J1) to PORT 3 of Dyna-51EB in 4 bit mode.
4. Download the program adc.hex (given in the CD) through flash magic.
5. Apply the analog input at the phono jack on the ADC card.
6. Output will display on LCD card.

PROGRAM:

```
ORG 0000H
LJMP MAIN
ORG 0030H
MAIN:
NOP
EN EQU P2.0
RS EQU P2.2
// RW EQU P1.1
DAT EQU P2
LCALL LCD_INT
LCALL CLEAR
LCALL LINE1
MOV DPTR, #MYDATA
LCALL LOOP
LCALL LINE2
MOV DPTR, #MYDAT2
LCALL LOOP
LCALL LINE5
lcall dispH
again: lcall adconvert
LCALL LINE3
lcall adisph
LCALL LINE4
lcall adispl

SJMP again
```

```
;=====
=====
```

```
W_NIB:      PUSH ACC          ;Save A for low nibble
            ORL DAT,#0F0h    ;Bits 4..7 <- 1
            ORL A,#0Fh      ;Don't affect bits 0-3
            ANL DAT,A        ;High nibble to display
            SETB EN
            CLR EN
            POP ACC          ;Prepare to send
            SWAP A           ;...second nibble
            ORL DAT,#0F0h    ; Bits 4..7 <- 1
            ORL A,#0Fh      ; Don't affect bits 0...3
            ANL DAT,A        ;Low nibble to display
            SETB EN
            CLR EN
            RET
```

```
;=====
=====
```

```
LCD_INT:   CLR RS
//         CLR RW
           CLR EN
           SETB EN
```

```

MOV DAT,#028h
CLR EN
LCALL SDELAY
MOV A,#28h
LCALL COM
MOV A,#0Ch
LCALL COM
MOV A,#06h
LCALL COM
LCALL CLEAR
MOV A,#080H
LCALL COM
RET

```

```

;=====
=====

```

```

CLEAR:          CLR RS
                MOV A,#01h
                LCALL COM
                RET

```

```

;=====
=====

```

```

DATAW:          SETB RS
//              CLR RW
                LCALL W_NIB
                LCALL LDELAY
                RET

```

```

;=====
=====

```

```

SDELAY:         MOV R6,#1
HERE2:          MOV R7,#255
HERE:           DJNZ R7,HERE
                DJNZ R6,HERE2
                RET

```

```

;=====
=====

```

```

LDELAY:         MOV R6,#1
HER2:           MOV R7,#255
HER:            DJNZ R7,HER
                DJNZ R6,HER2
                RET

```

```

;=====
=====

```

```

COM: CLR RS
//           CLR RW
            LCALL W_NIB
            LCALL SDELAY
            RET

```

```

;=====
=====

```

```

LINE1:          MOV A,#80H
                LCALL COM
                RET
LINE2:          MOV A,#0C0H
                LCALL COM
                RET
LINE3:          MOV A,#0c9H
                LCALL COM
                RET
LINE4:          MOV A,#0cah
                LCALL com
                RET
LINE5:          MOV A,#0cbh
                LCALL com
                RET

```

```

;=====
=====

```

```

LOOP: CLR A
        MOVC A,@A+DPTR
        JZ GO_B2
        LCALL DATAW
        LCALL SDELAY
        INC DPTR
        SJMP LOOP
GO_B2:  RET

```

```

ADisph:
        MOV a,r3
        LCALL DATAW
        LCALL SDELAY
        ret

```

```

ADispl:
        MOV a,r4
        LCALL DATAW
        LCALL SDELAY
        ret

```

```

;=====
=====

```

```

MYDATA:          DB " PIO-ADC-01 ",0
MYDAT2:          DB " ADC i/p= ",0
dispH:
        MOV a,#48H ;ascii for H
        LCALL DATAW
        LCALL SDELAY
        ret

```

```

;=====
=====

```

```

adconvert:

```

```

;-----
START      EQU  P1.0          ; Pin 6 Start
EOC        EQU  P1.3          ; Pin 7 EOC
OE         EQU  P1.1          ; Pin 9 Output Enable
ALE        EQU  P1.2          ; Pin 22 ALE
adata     EQU  P0             ; Data Lines

;-----
; Read one byte of data from adc.
; Performs a analog conversion cycle.
; address of channel in register "ADDRESS",
; Returns data in BUFFER
; Destroys A.
      MOV  adata,#0FFH        ; Data lines for input
      SETB OE                 ; Disable output
      SETB ALE                ; Latch the address
      NOP
      nop
      nop
      NOP
      SETB START              ; Start the conversion
      NOP
      NOP
      NOP
      CLR START
      NOP
      NOP
EOCLOOP:
      JNB EOC, EOCLOOP        ; Do until EOC high
      CLR OE                  ; Output Enable
      MOV a,adata             ; Get data in buffer
      SETB OE
      CLR ALE

```

Result:

The Digital data corresponding to the analog input is shown in the LCD.

Viva Questions:

1. Why 8051 is called 8 bit microcontroller?
2. How much on chip ram is available on 8051?
3. What is special Function Registers (SFR)?
4. Which bit of the flag register is set when output overflows to the sign bit?
5. Explain whether Port 0 of 8051 can be used as input output port?

Value added experiments

EXPERIMENT NO: 12

AIM: - Write an assembly language program to interfacing temperature measurement card with 8086 and program to display the temperature on LCD.

A thermocouple is an electrical device consisting of two dissimilar electrical conductors forming electrical junctions at differing temperatures. A thermocouple produces a temperature-dependent voltage as a result of the thermoelectric effect, and this voltage can be interpreted to measure temperature.

APPARATUS: - Dyna 8086 microprocessor kit and thermocouple controller card.

Address	Op Code	Mnemonic Operand	Comments
5313	B2 01	MOV DL,1	Move 1 to DL
5315	BB E0 4B	MOV BX,4BE0H	Move 4BE0H to BX
5318	E8 86 FF	CALL DISP 1	Call subroutine display
531B	B0 91	MOV AL,91H	Move 91H to AL
531D	E6 67	OUT CNTL_PORT,AL	Out the content of AL to port CNTL_PORT
531F	B0 80	MOV AL,80H	
5321	E6 65	OUT PORT_C,AL	Out the content of AL to port PORT_C
5323	B0 00	START : MOV AL,00H	Clear the accumulator
5325	B1 05	MOV CL,05H	Load the counter with value 05
5327	D2 C0	ROL AL,CL	Rotate the accumulator value left by CL
5329	24 E0	AND AL,E0H	Perform logical AND of AL with

			E0H
532B	0C 16	OR AL,16H	Perform logical OR of AL with 16H
532D	8A E0	MOV AH,AL	Move content of AL to AH
532F	E6 63	OUT PORT-B,AL	
5331	B9 FF FF	MOV CX,FFFFH	Mask the CX register
5334	E8 86 00	CALL DELAY	Call the subroutine delay
5337	8A C4	MOV AL,AH	Move Content of AH to AL
5339	0C 01	OR AL,01H	Perform logical OR of AL with 01H
533B	8A F0	MOV AH,AL	
533D	E6 63	OUT PORT_B,AL	Out the content of AL to PORT_B
533F	B9 FF 0F	MOV CX,0FFFH	Move 0FFFH to CX
5342	E8 78 00	CALL DELAY	Call delay subroutine
5345	8A C4	MOV AL,AH	
5347	24 FE	AND AL,FEH	Perform logical AND of AL with FEH
5349	8A E0	MOV AH,AL	
534B	E6 63	OUT PORT_B,AL	Out the content of AL to PORT_B
534D	B9 FF 00	MOV CX,00FFH	Move 00FFH to CX
5350	E8 6A 00	CALL DELAY	Call delay subroutine
5353	E4 65	UP : IN AL,PORT_C	Get the input from Port C
5355	24 01	AND AL,01H	
5357	3C 01	CMP AL,01H	Compare the content of AL with 01H
5359	75 F8	JNZ UP	If the result is not zero , then jump

			to UP label
535B	8A C4	MOV AL,AH	Move the content of AH into AL
535D	24 FD	AND AL,FDH	
535F	8A E0	MOV AH,AL	
5361	E6 63	OUT PORT_B,AL	Output the content of AL to PORT_B
5363	B9 FF 0F	MOV CX,0FFFH	Move 0FFFH to CX
5366	E8 54 00	CALL DELAY	Call delay subroutine
5369	E4 61	IN AL, PORT-A	
536B	8A C8	MOV CL,AL	Perform logical AND of AL with FEH
536D	E8 55 00	CALL CHK_SETPT	
5370	B5 00	MOV CH,00H	Out the content of AL to PORT_B
5372	BB 00 00	MOV BX,0000H	Move 00FFH to CX
5375	87 CB	XCHG CX,BX	Call delay subroutine
5377	BA 00 00	MOV DX,0000H	Get the input from Port C
537A	8A C7	UP 1 : MOV AL,BH	
537C	0A C3	OR AL,BL	Compare the content of AL with 01H
537E	74 17	JZ DOWN	If Zero flag is set then start the execution from DOWN
5380	4B	DEC BX	Decrement the content of register pair BC
5381	32 C0	XOR AL,DL	EXOR of AL and DL
5383	8A C2	MOV AL,DL	Move DL into AL

5385	FE C0	INC AL	Increase the AL by 1
5387	27	DAA	Decimal adjust the value of A
5388	8A D0	MOV DL,AL	
538A	73 EE	JNC UP 1	Out the content of AL to PORT_B
538C	32 C0	XOR AL,AL	Move 00FFH to CX
538E	8A C6	MOV AL,DH	Call delay subroutine
5390	FE C0	INC AL	Get the input from Port C
5392	27	DAA	
5393	8A F0	MOV DH,AL	Compare the content of AL with 01H
5395	73 E3	JNC UP 1	If Zero flag is set then start the execution from DOWN
5397	E8 1B 00	DOWN : CALL DIS	Decrement the content of register pair BC
539A	E8 2A FF	CALL KEYCHK 1	EXOR of AL and DL
539D	3C FF	CMP AL,FFH	Move DL into AL
539F	74 04	JZ DOWN 1	Increase the AL by 1
53A1	3C 0F	CMP AL,0FH	Decimal adjust the value of A
53A3	77 0D	JA DOWN 2	Move 0FFFH to CX
53A5	B9 FF FF	DOWN 1 : MOV CX,FFFFH	Call delay subroutine
53A8	90	UP 2: NOP	
53A9	90	NOP	Perform logical AND of AL with FEH
53AA	90	NOP	
53AB	90	NOP	Out the content of AL to PORT_B
53AC	90	NOP	Move 00FFH to CX

53AD	E2 F9	LOOP UP 2	Call delay subroutine
53AF	E9 71 FF	JMP START	Get the input from Port C
53B2	E9 1C F9	DOWN 2: JMP MAIN_MENU	

Result:

Measured **Temperature** will be display on LCD.

EXPERIMENT NO.: -10

AIM: Write an assembly language program to interface 7 segment display with 8051 Microcontroller.

APPARATUS:

1. 8051 micro controller kit.
2. PC for connecting the DYNA51 kit.
3. Keil-U-Vision software for compilation of program.
4. 7 segment HEX PIO CARD for DYNA51.

PROCEDURE:

1. Connect port-2(J2) of 7-segment display to PORT-1(J2) of Dyna51EB & port-1(J1) of 7-segment display to PORT-2(J3) of Dyna-51EB.
2. Download the program (seginterface.hex) through flash magic & observe the output on 7segment display.

PROGRAM:

;SEVEN SEGMENT 8-DIGIT COMMON ANODE DISPLAY

```
CPU "8051.TBL" ;Processor declaration
HOF "INT8" ;Intel 8-bit hexcode
```

```
ZERO: EQU 3fH
ONE: EQU 06H
TWO: EQU 5bH
THREE: EQU 4fH
FOUR: EQU 66H
FIVE: EQU 6dH
SIX: EQU 7dH
SEVEN: EQU 07H
EIGHT: EQU 7fH
NINE: EQU 0e7H
;DOT: EQU 7FH
adata: equ 77h
```

```
bdata: equ 7ch
cdata: equ 39h
ddata: equ 5eh
edata: equ 79h
fdata: equ 71h
seg_port: equ 0a0h ; port p2
digit_port: equ 90h ; port p1
```

```
ORG 0000H
```

```
MOV seg_port,#00H
mov digit_port,#0ffh
mov a,#0feh
```

```
LOOP:      mov digit_port,a
MOV seg_port,#ZERO
CALL DELAYS
MOV seg_port,#ONE
CALL DELAYS

MOV seg_port,#TWO
CALL DELAYS

MOV seg_port,#THREE
CALL DELAYS

MOV seg_port,#FOUR
CALL DELAYS

MOV seg_port,#FIVE
CALL DELAYS

MOV seg_port,#SIX
CALL DELAYS

MOV seg_port,#SEVEN
CALL DELAYS

MOV seg_port,#EIGHT
CALL DELAYS

MOV seg_port,#NINE
CALL DELAYS

;MOV seg_port,#DOT
;CALL DELAYS
```

```
MOV seg_port,#adata
CALL DELAYS
```

```
MOV seg_port,#bdata
CALL DELAYS
```

```
MOV seg_port,#cdata
CALL DELAYS
```

```
MOV seg_port,#ddata
CALL DELAYS
```

```
MOV seg_port,#edata
CALL DELAYS
```

```
MOV seg_port,#fdata
CALL DELAYS
```

```
rl a
AJMP loop
```

DELAYS:

```
MOV R5,#2 ;200ms delay
```

D1:

```
CALL DELAY
DJNZ R5,D1
RET
```

```
DELAY: ;100ms DELAY
MOV R7,#200
```

D2:

```
MOV R6,#100
```

D3:

```
NOP
NOP
NOP
DJNZ R6,D3
DJNZ R7,D2
RET
```

END

Result:

The desired character will be displayed of 7-segment display.