LABORATORY MANUAL

MICROPROCESSOR & MICROCONTROLLER LAB

B.Tech. (Electrical Engineering), 6th Semester

Department of Electrical Engineering
Veer Surendra Sai University of Technology,
BURLA

Vision

To be recognized as a center of excellence in education and research in the field of Electrical Engineering by producing innovative, creative and ethical Electrical Engineering professionals for socio-economic development of society in order to meet the global challenges.

Mission

Electrical Engineering Department of VSSUT Burla strives to impart quality education to the students with enhancement of their skills to make them globally competitive through:

M1. Maintaining state of the art research facilities to provide enabling environment to create, analyze, apply and disseminate knowledge.

M2. Fortifying collaboration with world class R& D organizations, educational institutions, industry and alumni for excellence in teaching, research and consultancy practices to fulfill 'Make in India' policy of the Government.

M3. Providing the students with academic environment of excellence, leadership, ethical guidelines and lifelong learning needed for a long productive career.

Program Educational Objectives

The program educational objectives of B.Tech. in Electrical Engineering program of VSSUT Burla are to prepare its graduates:

1. To have basic and advanced knowledge in Electrical Engineering with specialized knowledge in design and commissioning of electrical systems/renewable energy systems comprising of generation, transmission and distribution to become eminent, excellent and skillful engineers.
2. To succeed in getting engineering position with electrical design, manufacturing industries or in software and hardware industries, in private or government sectors, at Indian and in Multinational organizations.
3. To have a well-rounded education that includes excellent communication skills, working effectively on team-based projects, ethical and social responsibility.
4. To have the ability to pursue study in specific area of interest and be able to become successful entrepreneur.
5. To have broad knowledge serving as foundation for lifelong learning in multidisciplinary areas to enable career and professional growth in top academic, industrial and government/corporate organizations.

LIST OF EXPERIMENTS

1. Verification of basic instruction set of 8085 microprocessors.
2. Addition, Subtraction, Multiplication and Division of two 8-bit numbers resulting 8/16-bit numbers using 8085.
3. (a) Find smallest and largest number among ‘n’ numbers in a given data array using 8085.
   (b) Write an assembly language program of binary to gray code conversion in 8085.
4. Write a program to generate square waves of different frequencies on all lines of 8255 by the help of delay program.
5. Study of stepper motor and its operations (clockwise, anti-clockwise, angular movement and rotation in various speeds).
6. Study of different addressing modes of 8051 microcontrollers.

COURSE OUTCOMES

Upon completion of the course, the students will be able to:

CO1. Perform different mathematical operations using microprocessor.
CO2. Demonstrate an ability to write assembly language programming.
CO3. Perform different tasks using programmable devices and work with different interfacing circuits.
CO4. Demonstrate stepper motor using microprocessor.
CO5. Produce different types of waveforms.

INTRODUCTION TO MICROPROCESSOR 8085

ARCHITECTURE OF 8085 MICROPROCESSOR

a) General purpose registers

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 counters (PC)

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

d) Stack pointer (SP)

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

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

i) Carry flag (C)

ii) Parity flag (P)

iii) Zero flag (Z)

iv) Auxiliary carry flag (AC)

v) 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.

PIN Description

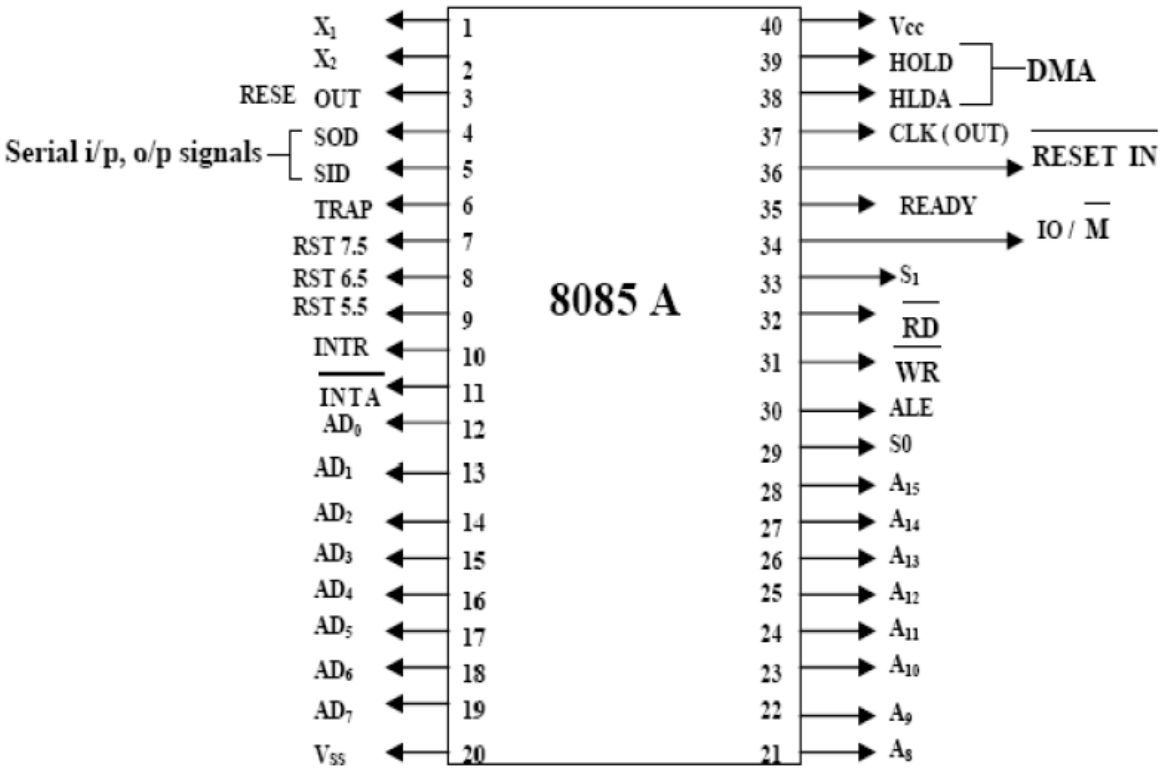


Fig. 1. 8085 Microprocessor Pin Layout and Signal Groups

Address Bus

1. The pins A0 – A15 denote the address bus.
2. They are used for most significant bit

Address / Data Bus

1. AD0 – AD7 constitutes the address / Data bus
2. These pins are used for least significant bit

ALE: (Address Latch Enable)

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

IO / M

1. This distinguishes whether the address is for memory or input.
2. When this pins go high; the address is for an I/O device.

S0 – S1

S0 and S1 are status signal which provides different status and functions.

RD

1. This is an active low signal
2. This signal is used to control READ operation of the microprocessor.

WR

1. WR is also an active low signal
2. 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

1. HLDA is the acknowledgement signal for HOLD
2. It indicates whether the hold signal is received or not.

INTR

1. INTE is an interrupt request signal
2. IT can be enabled or disabled by using software

INTA

1. Whenever the microprocessor receives interrupt signal
2. It has to be acknowledged.

TRAP

1. Trap is the only non-maskable interrupt
2. 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.

X1, X2

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

RST 5.5, 6.5, 7.5

1. These are nothing but the restart interrupts
2. They insert an internal restart junction automatically.

VCC and VSS

1. VCC is +5V supply pin
2. VSS is ground pin

Specifications

1. Processors: Intel 8085 at E144 MHz clock
2. Memory: RAM: 2000 – 3FFF

EPROM :0000 to 1FFF

3. Input / Output:

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

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: LCD display.

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

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

IC's Used

8085 - 8-bitMicroprocessor

8253 - Programmable internal timer

8255 - Programmable peripheral interface

8279 - Programmable key boards / display interface

8251 - Programmable communication interface

2764 - 8 K EPROM

6264 - 8K STATIC RAM

7414 - Hex inverter

7432 - Quad 21/p OR GATE

7409 - Quad 21/p AND GATE

7400 - NAND Gate

7404 - Dual D-FF

74373 - Octal 'D' Latch

74139 - Dual 2 to 4-line decoder

74138 - 3 to 8-line decoder

Experiment 1

AIM OF THE EXPERIMENT:

Verification of Instructions of 8085 Microprocessor.

(A) Data Transferred Group

(B) Arithmetic Group

(C) Logical Group

(D) Branch Control Group

(E) I/O & machine Control Group

APPARATUS REQUIRED:

1. Advanced LCD 8085 Microprocessor kit.
2. Key board.

INSTRUCTIONS TO BE VERIFIED:

(A) Data Transferred Group

1. MOV r1, r2 (Move the content of one register to another)

Program

MVI B, data	direct the data will be loaded to register B
MOV A, B	Data will be transferred from B to A.
STA 2400H	Data will be stored at memory location 2400H from ACC.
RST 5	Stop.

Address	Instruction	Op-Code
2000	MVI B, 20	06
2001		20
2002	MOV A, B	78
2003	STA 2400	32
2004		00
2005		24
2006	RST 5	EF

INPUT		OUTPUT	
Address	Data	Address	Data
2001H	20H	2400H	20H

2. MOV r, M (Move the content of memory to register)

Program

LXI H, 2500H	Load H-L pair direct by 2500H.
MOV B, M	Move the content of memory location 2500H to B.
INX H	increment the register pair.
MOV M, B	Move the data from B to memory location 2501.
RST 5	Stop.

Address	Instruction	Op-Code
2000	LXI H, 2500	21
2001		00
2002		25
2003	MOV B, M	46
2004	INX H	23
2005	MOV M, B	70
2006	RST 5	EF

INPUT		OUTPUT	
Address	Data	Address	Data
2500H	20H	2501H	20H

3. MVI r, Data (Move immediate data to memory)

(1^{St}. program can be used for checking)

4. LXI rp, data (Load register pair immediate)

(Instruction is used in program-2 to load immediate data to H-L pair by 2500)

5. LDA addr. (Load accumulator direct)

Program

LDA 2500H	Data will be loaded to accumulator by picking up the date from 2500H.
STA 2502	Data will be stored at memory location 2501H from ACC.
RST 5	Stop.

Address	Instruction	Op-Code
2000	LDA 2500	3A
2001		00

2002		25
2003	STA 2502	32
2004		02
2005		25
2006	RST 5	EF

INPUT		OUTPUT	
Address	Data	Address	Data
2500H	20H	2502H	20H

6. STA addr. (Store Accumulator)

(Instruction is verified in the above program)

7. LHLD addr (Load H-L pair direct)

[L] ← [addr] , [H] ← [addr+1]

Program

Address	Instruction	Op-Code
2000	LHLD 2500	2A
2001		00
2002		25
2003	SHLD 2502	22
2005		02
2005		25
2006	RST 5	EF

(N.B.: - Give the data at memory location 2500H and 2501H. and data will be stored at 2502H and 2503H)

INPUT		OUTPUT	
Address	Data	Address	Data
2500H	25H	2502H	25H
2501H	26H	2503H	26H

8. SHLD addr (Store H-L pair direct)

[addr] ← [L] [addr+1] ← [H]

(check the program 7)

(B) ARITHMETIC GROUP :-

9. ADD r (Add register to accumulator)

[A] ←[A] + [B]

Program

LXI H,	2400H Load H-L pair direct by 2400H
MOV A, M	Move the content of memory location 2400H to Accumulator.
INX H	increment the register pair.
MOV B, M	Move the data from B to memory location 2401
ADD B	Add content of register B to accumulator
STA 2500H	Data will be stored at memory location 2500H from ACC
RST 5	Stop.

INPUT		OUTPUT	
Address	Data	Address	Data
2400H	04H	2500H	0AH
2401H	06H		

10. ADD M (Add memory to accumulator)

Program

LXI H,	2400H Load H-L pair direct by 2500H
MOV A, M	Move the content of memory location 2500H to Accumulator.
INX H	increment the register pair.
ADD M	Add content of memory to accumulator
STA 2500H	Data will be stored at memory location 2500H from ACC
RST 5	Stop.

INPUT		OUTPUT	
Address	Data	Address	Data
2400H	05H	2500H	0BH
2401H	06H		

11. ADI data (Add immediate data to accumulator)

Program

LDA 2400	Data will be loaded to accumulator by picking up the date from 2400H
ADI data	Add immediate data to accumulator
STA 2401	Data will be stored at memory location 2401H from Accumulator
RST 5	Stop.

12. DAD rp (Add registrar pair to H-L Pair)

Program

LHLD 2500H	Load H-L pair direct
XCHG	Exchange data [H-L] pair to [D-E] pair
LHLD 2502	Load [H-L] pair direct.
DAD D	Add registrar pair to H-L Pair
SHLD 2004H	Store H-L pair direct

RST 5 Stop

INPUT		OUTPUT	
Address	Data	Address	Data
2500HH	03H	2600H	08H
2502H	05H		

13. DAA (Decimal Adjust Accumulator)

Program

LDA 2400H	Data will be loaded to accumulator by picking up the date from 2400H
MOV B, A	Data will be transferred from A to B.
LDA 2401	Data will be loaded to accumulator by picking up the date from 2401H
ADD B	Add content of register B to accumulator
DAA	Decimal Adjust Accumulator
STA 2402	Data will be stored at memory location 2401H from Accumulator
RST 5	Stop

Data is in BCD

Result is in Decimal equivalent.

INPUT		OUTPUT	
Address	Data	Address	Data
2400HH	22	2402H	54
2401H	32		

(C) Logical Group

14. ANA r (AND register with accumulator)

[A] ← [A] ∧ [r]

Program

LDA 2500H	Data will be loaded to accumulator by picking up the date from 2400H
MOV B, A	Data will be transferred from A to B.
LDA 2501H	Data will be loaded to accumulator by picking up the date from 2401H
ANA B	AND register with accumulator
STA 2402H	Data will be stored at memory location 2401H from Accumulator
RST 5	Stop

INPUT		OUTPUT	
Address	Data	Address	Data
2500H	02H	2402H	00H
2501H	01H		

15 ANA M (AND memory with accumulator)

Program

LXI H 2400H	Load H-L pair direct by 2400H
MOV A, M	Data will be transferred from Memory to Accumulator.
INX H	Increment the register pair
ANA M	AND memory with accumulator
STA 2402H	Data will be stored at memory location 2401H from Accumulator
RST 5	Stop.

16. ANI Data (AND immediate data with accumulator)

Program

LDA 2400H	Data will be loaded to accumulator by picking up the date from 2400H
ANI Data	AND immediate data with accumulator
STA 2401H	Data will be stored at memory location 2401H from Accumulator
RST 5	Stop.

17. ORA r (Or registrar with accumulator)

[A] [A] V [r]

Program

LDA 2400H	Data will be loaded to accumulator by picking up the date from 2400H
MOV B, A	Data will be transferred from A to B.
LDA 2401H	Data will be loaded to accumulator by picking up the date from 2401H
ORA B	Or registrar with accumulator

STA 2402H Data will be stored at memory location 2402H from Accumulator

RST 5 Stop.

INPUT		OUTPUT	
Address	Data	Address	Data
2400H	07H	2402H	07H
2401H	06H		

18. CMC (Compliment carry status)

[CS] ← [CS]

19. STC (Set carry status)

[CS] ← 1

Program

LXI H 2400H Load H-L pair direct by 2400H

MOV A, M Data will be transferred from Memory to Accumulator

STC Set Carry, 1

CMC Compliment carry status

RAR Rotate accumulator left

INX H Increment the register pair

MOV M, A Data available at 2401

RST 5 Stop.

INPUT		OUTPUT	
Address	Data	Address	Data
2400H	04H	2401H	02H

20 RLC (Rotate Accumulator left)

$[A_{n+1}]$ $[A_n]$, $[A_0]$ $[A_7]$, [CS] [A7]

Program

LXI H 2400H Load H-L pair direct by 2400H

MOV A, M Data will be transferred from Memory to Accumulator

RLC Rotate Accumulator left

STA 2401H Data will be stored at memory location 2401H from Accumulator

RST 5 Stop

INPUT		OUTPUT	
Address	Data	Address	Data
2400H	06H	2401H	0CH

21 RAR (Rotate accumulator right through carry)

(It is checked in 20th. Instruction checking the program.)

$[A_n] \leftarrow [A_{n+1}]$, $[CS] \leftarrow [A_0]$ $[A_7] \leftarrow [CS]$

(D)Branch Control Group

22. JMP addr (Label): - (Unconditional jump to the instruction specified by the operand)

```
    Program
    LXI H 2400H
    MOV A, M
    INX H
    JMP GO
    ADD M
GO  STA 2402H
    RST 5
```

(N.B.: - No addition will be performed. Data present at the accumulator will be stored location 2402H. i.e., the data given at 2400)

Address		Instruction	Op-Code
2000		LXI H, 2400	21
2001			00
2002			24
2003		MOV A,M	7E
2004		INX H	23
2005		JMP **GO**	C3
2006			08
2007			20
2008		ADD M	86
2009	**GO**	STA 2402	32
200A			02
200B			24
200C		RST 5	EF

INPUT		OUTPUT	
Address	Data	Address	Data
2400H	04H	2402H	04H
2401H	06H		

23. JZ adder (Label): - (Jump if the result is zero)

Program

```
        LXI H 2400H
        MOV A, M
        INX H
        CMP M
        JZ      FORWARD   (Is the present no. the smallest one? Yes, go to FORWARD)
FORWARD STA 2502    (So the smallest one will be stored at the location 2502)
        RST 5
```

INPUT		OUTPUT	
Address	Data	Address	Data
2400H	06H	2402H	DDH
2401H	07H		

24. JNZ addr (Label) (Jump if the result is not zero)

25. JC addr (Label) (Jump if there is a carry)

Program

```
        LXI H 2500H
        MOV A, M
        INX H
        CMP M
        JC      AHEAD
        MOV A, M
AHEAD   STA 2503H
        RST 5
```

INPUT		OUTPUT	
Address	Data	Address	Data
2500H	06H	2502H	06H
2501H	07H		

(E) Stack, I/O and M/C Control Group

26. Push rp and Poprp: -

Push rp - Push the content of register pair to the stack.

Pop rp – Pop the content of resister pair. Which was saved from the stack.

Program

```
LXI SP, 2500H
LXI H, 2400H
PUSH H
RST 5
```

PUSH rp

N.B.: - Check the location 24FE and 24FF to get the contents of L and H i.e., “00” and “24” respectively.

POP rp

N.B.: - Check the contents of H & L to get the desired result. i.e [rl] ← [[SP]]

[rh] ← [[SP]+1]

Experiment 2

AIM OF THE EXPERIMENT:

Addition, Subtraction, Multiplication and Division of two 8-bit numbers resulting 8/16-bit numbers using 8085.

APPARATUS REQUIRED:

1. Advanced LCD 8085 Microprocessor kit.
2. Key board.

ADDITION OF TWO 8-BIT NUMBERS

Algorithm:

Step 1 : Start the microprocessor

Step 2 : Initialize the carry as 'Zero'

Step 3 : Load the first 8 bit data into the accumulator

Step 4 : Copy the contents of accumulator into the register 'B'

Step 5 : Load the second 8 bit data into the accumulator.

Step 6 : Add the 2 - 8 bit data and check for carry.

Step 7 : Jump on if no carry

Step 8 : Increment carry if there is

Step 9 : Store the added request in accumulator

Step 10 : More the carry value to accumulator

Step 11 : Store the carry value in accumulator

Step 12 : Stop the program execution.

Program

	MVI C 00	Initialize the carry as 'Zero'
	LDA 2500	Load the first 8-bit data into the accumulator
	MOV B, A	Copy the contents of accumulator into the register 'B'
	LDA 2501	Load the second 8-bit data into the accumulator.
	ADD B	Add the 2 8-bit data and check for carry.
	JNC **GO**	Jump on if no carry
	INR C	Increment carry if there is
GO	STA 2502	Store the added request in accumulator
	MOV A, C	More the carry value to accumulator
	STA 2503	Store the carry value in accumulator

	RST 5	Stop.

SUBTRACTING 2 BIT (8) NUMBERS

Algorithm:

Step 1 : Start the microprocessor

Step 2 : Initialize the carry as 'Zero'

Step 3 : Load the first 8-bit data into the accumulator

Step 4 : Copy the contents of contents into the register 'B'

Step 5 : Load the second 8-bit data into the accumulator.

Step 6 : Subtract the 2 8-bit data and check for borrow.

Step 7 : Jump on if no borrow

Step 8 : Increment borrow if there is

Step 9 : 2's compliment of accumulator is found out

Step 10 : Store the result in the accumulator

Step 11 : More the borrow value from 'c' to accumulator

Step 12 : Store the borrow value in the accumulator

Step 13 : Stop program execution

Program

	MVI C 00	Initialize the carry as 'Zero'
	LDA 2500	Load the first 8-bit data into the accumulator
	MOV B, A	Copy the contents of accumulator into the register 'B'
	LDA 2501	Load the second 8-bit data into the accumulator.
	SUB B	Subtract both the value.
	JNC **LOOP**	Jump on if no borrow
	INR C	Increment borrow if there is
	CMA	Compliment of 2nd. data
	ADI 01	Add one to 1's compliment of 2nd Data.
LOOP	STA 2502	Store the result in accumulator
	MOV A, C	More the borrow value from C to accumulator
	STA 2503	Store the borrow value in accumulator
	RST 5	Stop.

Input (Without borrow)

Input Address	Data
2500H	05H
2501H	07H

Output

Output Address	Data
2502H	02H
2503H	00(borrow)

Input (With borrow)

Input Address	Data
2500H	07H
2501H	05H

Output

Output Address	Data
2502H	02H
2503H	01(borrow)

MULTIPLYING TWO 8-BIT NUMBERS

Algorithm:

Step 1 : Start the microprocessor

Step 2 : Get the 1st 8-bit numbers

Step 3 : Move the 1st 8-bit number to register 'B'

Step 4 : Get the 2nd 8-bit number

Step 5 : Move the 2nd 8-bit number to register 'C'

Step 6 : Initialize the accumulator as zero

Step 7 : Initialize the carry as zero

Step 8 : Add both register 'B' value as accumulator

Step 9 : Jump on if no carry

Step 10 : Increment carry by 1 if there is

Step 11 : Decrement the 2nd value and repeat from step 8, till the 2nd value becomes zero.

Step 12 : Store the multiplied value in accumulator

Step 13 : Move the carry value to accumulator

Step 14 : Store the carry value in accumulator

Program

LDA 2500 Load the 1st 8-bit numbers

```
        MOV B, A          Move the 1st 8-bit number to register 'B'
        LDA 2501          Load the 2nd 8-bit number
        MOV C, A          Move the 2nd 8-bit number to register 'C'
        MVIA,00           Initialize the accumulator as zero
        MVI D,00          Initialize the carry as zero
Loop    ADD B             Add both content of register 'B' with accumulator
        JNC      GO       Jump on if no carry
        INR D             Increment carry by 1 if there is
GO      DCR C             Decrement the value C
        JNZ  Loop         Jump if number is zero
        STA 2502          Store the result in accumulator.
        MOV A, D          Move the carry in accumulator
        STA 2503          Store the result in accumulator
        RST 5             Stop.
```

INPUT		OUTPUT	
Address	Data	Address	Data
2500H	05H	2502H	14H
2501H	04H	2506H	00H

DIVISION OF TWO 8 – BIT NUMBERS

Algorithm:

Step 1 : Start the microprocessor

Step 2 : Initialize the Quotient as zero

Step 3 : Load the 1st 8-bit data

Step 4 : Copy the contents of accumulator into register 'B'

Step 5 : Load the 2nd 8-bit data

Step 6 : Compare both the values

Step 7 : Jump if divisor is greater than dividend

Step 8 : Subtract the dividend value by divisor value

Step 9 : Increment Quotient

Step 10 : Jump to step 7, till the dividend becomes zero

Step 11 : Store the result (Quotient) value in accumulator

Step 12 : Move the remainder value to accumulator

Step 13 : Store the result in accumulator

Step 14 : Stop the program execution

Program

	MVI C, 00	Initialize the Quotient as zero
	LDA 2500	Load the 1st 8-bit data
	MOV B, A	Copy the 1st data into register 'B'
	LDA 2501	Get the 2nd data
	CMP B	Compare the 2 values
	JC **Loop 1**	Jump if dividend lesser than divisor
Loop 2	SUB B	Subtract the 1st value by 2nd value
	INR C	Increment Quotient
	CMP B	
	JNC **Loop 2**	Jump to Loop 1 till the value of dividend becomes zero
Loop 1	STA 2502	Store the value in accumulator
	MOV A, C	Move the value of remainder to accumulator
	STA 2503	Store the remainder value in accumulator
	RST 5	Stop.

Input Address	Data	Output Address	Data
2500H	02H (Divisor)	2502H	04H (Remainder)
2501H	09H (Dividend)	2503H	01H (Quotient)

Experiment 3

AIM OF THE EXPERIMENT:

To write a program of

(i) Smallest/Largest number among 'n' number in a given data array.

(ii) Binary to gray code.

APPARATUS REQUIRED:

1. Advanced LCD 8085 Microprocessor kit.
2. Key board.

(A) **Smallest number among 'n' number in a given data array.**

Algorithm:

Step 1 : Start the microprocessor

Step 2 : Initialize the starting address of the n elements

Step 3 : Load the count of the array

Step 4 : Load FF in the accumulator [Accumulator FF is the largest 8-bit no.]

Step 5 : Find the smallest number by comparing the elements given by verifying the carry flag.

Step 6 : Store the address of smallest number.

Step 7 : Stop the program execution.

Program

	LXI H 2500	Get address for count in H-L pair.
	MOV C, M	Count in register C
	MVI FF	Get FF in accumulator
	INX H	Get address of 1st. number in H-L pair
	MOV A, M	1st. number in accumulator.
	DCR C	Decrement Count
Loop	INX H	Address of next number in H-L pair
	CMP M	Compare next number with previous smallest less than next number?
	JC **Ahead**	yes, smaller number in accumulator go to **Ahead.**
	MOV A, M	No, get the next number in accumulator.
Ahead	DCR C	Decrement count.
	JNZ **Loop**	
	STA 2450	Store the smallest number in 2450H
	RST 5	Stop.

Input Address	Data	Output Address	Data
2500H	04H (Count)	2450H	02H
2501H	08H		
2502H	06H		
2503H	09H		
2504H	02H		

(B) Largest number among 'n' number in a given data array.

Program

	LXI H 2500	Get address for count in H-L pair.
	MOV C, M	Count in register C
	SUB A	Get 00 in accumulator
Loop	INX H	Address of next number of series
	CMP M	Compare next number with previous Maximum.
		Is next number >previous maximum.
	JNC **Ahead**	No, Larger number in accumulator go to **Ahead.**
	MOV A, M	No, get the next number in accumulator.
Ahead	DCR C	Decrement count.
	JNZ **Loop**	
	STA 2450	Store the smallest number in 2450H
	RST 5	Stop.

Input Address	Data	Output Address	Data
2500H	04H (Count)	2450H	09H
2501H	07H		
2502H	09H		
2503H	02H		
2504H	08H		

(II) Binary to gray code.

Algorithm

Step 1 : Start the microprocessor

Step 2 **:** The MSB in the gray code is the same as the corresponding bit in a binary number.

Step 3 : Going from left to right, add each adjacent pair of binary digits to get the next gray code digit. Disregard carries.

Step 4 : Stop the program execution.

Program

LXI H 2500	Load H-L pair direct by 2500H
MOV A, M	Data will be transferred from Memory to Accumulator
STC	Set carry
CMC	Complement carry
RAR	Rotate accumulator right through carry
XRA M	Exclusive –OR register with accumulator
STA 2400	Store the result in 2400H
RST 5	Stop

Input Address	Data	Output Address	Data
2500H	05H	2400H	07H

Experiment 4

AIM OF THE EXPERIMENT:

To write a program to generate square waves of different frequencies on all lines of 8255 by the help of delay program.

APPARATUS REQUIRED:

1.Advanced LCD 8085 Microprocessor kit.

2. Key board.

THEORY:

The intel 8255 is a programmable peripheral interface. It has two versions, namely 8255-1 & 8255-2. Each version has two ports; Port A, PortB & port C. According to requirement of a port can be configured to act as an output port. For programming the ports of 8255, a control words register. The address of ports and control word register (CWR) of the two units of 8255 are as follows.

8255-1

Port/ Control Word register	Address
Port A	00
PORT B	01
Port C	02
CWR	03

8255-2

Port/ Control Word register	Address
Port A	08
PORT B	09
Port C	0A
CWR	0B

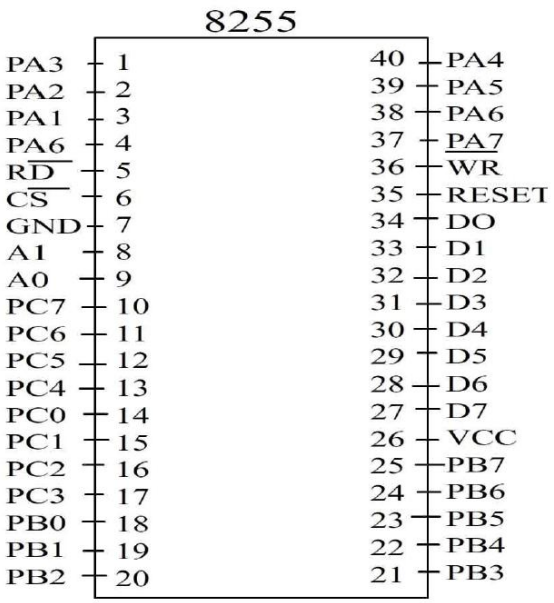


Fig. 2. Pin Diagram of 8255 PPI

PROCEDURE:

Step-1 Select the address of port and Control word register for the version 8255 available in the microprocessor kit.

Step-2 Select a port to act as the output port. From control word for the particular configuration of ports selected, select pin of the output port for taking output.

Step-3 Identify the pin by checking for high and low signal of it.

Step-4 Enter the program with delay subroutine for generating square wave.

Program:

The program given below is for port B as an output port and output is taken from line PB_0.

Program for making PB_0High.

```
MVI A,98      Get control word for 8255.
OUT 03        Initialize ports of 8255-1
MVI A 01
OUT 01        Make PB0 High.
RST 5         Stop.
```

Program for making PB_0Low.

```
MVI A,98      Get control word for 8255.
OUT 03        Initialize ports of 8255-1
MVI A 00
OUT 01        Make PB0 High.
RST 5         Stop.
```

Generation of Square wave.

```
Begin:          LXI SP,     2600H
Label 1         CALL        Set Routine
                CALL        Delay 1
                CALL        Unset Routine
                CALL        Delay
End             JMP         Label 1
Set- Routine    MVI A,      80
                OUT 03
                MVI FF      [All port B are output in high.]
                OUT 01
                RET
Unset- Routine  MVI A,      80
                OUT 03
                MVI 00      [All port B are output in low.]
                OUT 01
                RET
```

```
Delay -1        MVI B,      Data
Label 2         DCR B
                JNZ     Label 2
                RET
Delay -2        MVI B,      Data
                MVI C,      Data
Label 3         DCX B
                JNZ Label 3
                RET
```

Experiment 5

AIM OF THE EXPERIMENT:

Study of stepper motor and its operations (clockwise, anticlockwise, angular movement rotate) in various speeds.

APPARATUS REQUIRED:

1. Advanced LCD 8085 Microprocessor kit.
2. Key board.
3. Stepper motor.

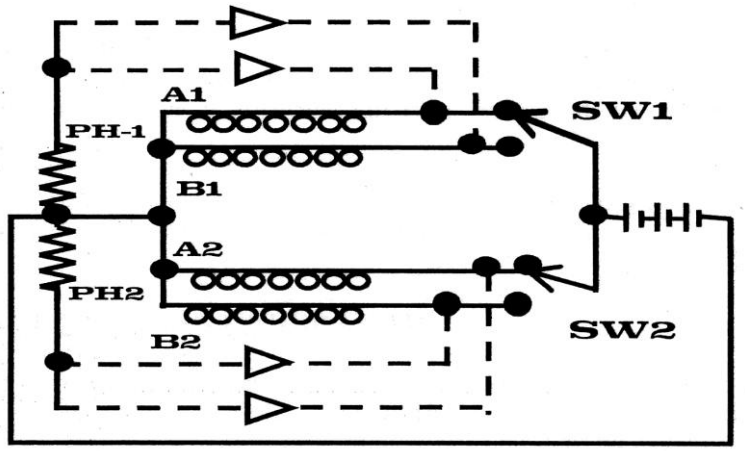
GENERAL DESCRIPTION:

DC STEPPING MOTORS: Stay they are entering into all branches of engineering. There are many systems to monitor various processes and give out control signals in the form of digits but there is only one device to convert these digital pulses into precise incremental motion and that device is stepping motor. Stepping motor is a device, which converts digital pulses into precise angular or linear steps of desired value. The stepping motors in general have the following specifications:

1. Permanent magnet DC stepping Motors, two phase bifiller would, Step angle – 1.8 degree ± 5% Non-commutative. Steps / Revolution: 200.
2. Features:
    - Instantaneous response to control pulses.
    - Holds on to the position infinitely in static condition.
    - No burn-out due to locked rotor.
    - Speed can be varied over a wide margin from 0-10,000 step /-sec. Equivalent to 0-3,000 RPM.
    - High torque to inertia ratio. Can be overdriven without any damage.
    - Can be programmed in three parameters namely, **Speed, direction and number of steps.**

WORKING OF STEPPING MOTOR:

The stepping action is caused by sequential switching of supply to the two phase of the motor as described in switching diagram. All stepping motors are of bifiller type with six leads. Each of the two phases of motor has double winding with a center tap.



Switching Diagram of DC Stepping Motor

Switching the supply from one side to another of a phase causes reversal of magnetic polarity without actually reversing the polarity of supply. Four step-input sequences gives 1.8 degree (Full) step function where as eight step input sequence gives 0.9 (half) step function.

SWITCHING SEQUENCE

Step	SW-1	SW-2	PH-1		PH-2		PHASE-1		PHASE-2	
			A1	B1	A2	B2	A1	B1	A2	B2
1	1	2	0	1	0	1	0	1	0	1
2	1	4	1	0	0	1	0	0	0	1
3	3	4	1	0	1	0	1	0	0	1
4	3	2	0	1	1	0	1	0	0	0
5	1	2	0	1	0	1	1	0	1	0
							0	0	1	0
							0	1	1	0
							0	1	0	0
							0	1	0	1

That the above switching Logic will move the shaft in one direction. To change the direction, read the sequence upward.

The specified torque of any stepping motor is the torque at stand still (holding torque). This torque is directly proportional to the current in the winding, which is governed by the DC resistance of winding. As the switching sequence starts, the inductive reactance of the winding which increases with the frequency of switching opposes the rise of current to desired level within the time given for one step depending upon the speed of level causes drop in torque as the speed increase. In order to improve torque at high speed, it is necessary to maintain current at the desired level. This can be done by one of the following methods:

1. By increasing supply voltage and introducing current limiting resistance in each phase. Introduction of resistance improves the time constant of winding. Seven to nine time the winding resistance in each phase will give very good improvement in torque / speed characteristics.
2. By using a constant current source with or without a chopper instead off using a constant voltage source which will give even better performance.

SETUP FOR THE EXPERIMENT:

This explanation as well as the explanation of the Program under the heading "Description of the Program" is for 8085 LED Kit. However, if you are interfacing the Stepper Motor Module to other Kits, then also refer to the specific instruction before the program listing for that particular Kit also.

1. Connect the ET-SMC I/F module to the 8255-I port connector of the kit using 26-pin flat FRC cable. The pin No.1 of the connector on the module as well as the kit are marked. Please ensure that the pin no. 1 of the connector on the Kit is connected to pin No.1 of the connector on the module.
2. Also connect the +5V to the Module power connector.
3. Enter the program given below from the memory location mentioned in the program.
4. Connect the Power of the Motor to +5V or as specified on the motor and Run the program and observe that the motor is running.

DESCRIPTION OF THE PROGRAM:

The program initializes the 8255 (PPI-1) in order to make port A as output port. The PA_0 to PA_3 is connected through buffer and driving circuit to the winding of the stepper motor. The codes for clockwise movement of stepper motor are FA, F6, F5 and F9 (refer switching sequence). These codes are to be outputted in the sequence they are written. In case of anti-clockwise movement of the stepper motor output codes are F9, F5, F6 and F4. The delay routine is called to generate the delay (max of the about 1 sec.) between the steps. This delay can be changed to make faster steps. The minimum delay depends upon the maximum speed of the stepper motor specified.

The speed of the steps can be varied by changing the constant at 2031 & 32 and 2037 and 2038. These values are taken by registrar pair DE and a corresponding delay is generated. Both the delays are added up to give the final delay. The individual delay can be calculated by 24N + 17) X basic Machine cycle. Here N is the number stored in D register pair.

Note: Listing of program for various models of Microprocessor and Micro controller kits is given below. Please select the model of kit being used before entering the program into the kit.

PROGRAM FOR 8085 KITS HAVING LCD DISPLAY:

Connect the J_1 of the Kit to the Module through 26 pin FRC Cable. Ensure that the pin 1 of the J_1 at the kit end is connected to the pin-1 of the Module connector. Enter the program from address 2000. Execute the program from address 2000.

ADDRESS	OP-CODE	LABEL	MNEMONICS	REMARKS
2000	3E 80		MVI A 80	Initialize All ports of 8255-1 as Output Port
2002	D3 03		OUT 03	
2004	3E **FA**	START	MVI A **FA**	Output code for Step 0
2006	D3 00		OUT 00	
2008	CD 30 20		CALL 2030	Delay between Two Steps
200B	3E **F6**		MVI A **F6**	Output Code for Step 1
200D	D3 00		OUT 00	
200F	CD 30 20		CALL 2030	Delay between Two Steps
2012	3E **F5**		MVI A **F5**	Output Code for Step 2
2014	D3 00		OUT 00	
2016	CD 30 20		CALL 2030	Delay between Two Steps
2019	3E **F9**		MVI A **F9**	Output Code for Step-3
201B	D3 00		OUT 00	
201d	CD 30 20		CALL 2030	Delay between Two Steps
2020	C3 04 20		Jmp 2004	Jump to START

DELAY ROUTINE: NOW ENTER FROM THE ADDRESS 2030 ONWARD

2030	11 00 00		LXI D – 00 00	Generate A Delay
2033	CD A6 03		CALL 03A6	
2036	11 00 00		LXI D – 00 00	Generate A Delay
2039	CD A6 03		CALL 03A6	
203C	C9		RET	Return

Note

1. **To change the Direction, reverse the switching sequence from FA, F6, F5, F9 to F9, F5, F6, FA.**
2. **To vary the speed, change the data at location 2032 and 2038 (Change 00 TO 0F OR 05).**

Experiment 6

AIM OF THE EXPERIMENT:

Study of different addressing modes of 8051 microcontroller.

APPARATUS REQUIRED:

1. 8051 microcontroller trainer
2. keyboard

THEORY:

Addressing mode is a way to address an operand. Operand means the data we are operating upon. It can be a direct address of memory, it can be register names, it can be any numerical data etc. There are 5 addressing modes in 8051:

1. Immediate addressing mode
2. Direct addressing mode
3. Register direct addressing mode
4. Register indirect addressing mode
5. Indexed addressing mode

PROGRAMS:

Immediate addressing mode

Memory address	Mnemonics	Opcode
6000	MOV A, #02	74, 02
6002	MOV B, #02	7A, 02
6004	MOV DPTR, #6200	90, 62, 00
6007	MOVX @DPTR, A	F0
6008	LCALL 05D2	12, 05, D2
600B	SJMP0006	02, 00, 06

Direct addressing mode

Memory address	Mnemonics	Opcode
6000	MOV DPTR, #6200	90,62,00
6003	MOV R2, DPL	AA,82
6005	MOV R3, DPH	AB, 83
6007	LCALL 05 D2	12, 05, D2
600A	SJMP0006	02, 00, 06

Register direct addressing mode

Memory address	Mnemonics	Opcode
6000	MOV A, R2	EA
6001	MOV DPTR, #6200	90, 62, 00
6004	MOVX @DPTR, A	F0
6005	INC A	04
6006	MOVX @DPTR, A	F0
6007	LCALL 05D2	12, 05, D2
600A	SJMP0006	02, 00, 06

Register indirect addressing mode

Memory address	Mnemonics	Opcode
6000	MOV DPTR, #6200	90, 62, 00
6003	MOVX A, @DPTR	E0
6004	INC A	04
6005	MOVX A, @DPTR	E0
6006	ADD A, B	34, 04
6008	INC DPTR	04
6009	MOVX @DPTR, A	F0
600A	LCALL 05D2	12, 05, D2
600D	LJMP0006	02, 00, 06

Indexed addressing mode

Memory address	Mnemonics	Opcode
6000	MOV DPTR, #6200	90, 62, 00
6003	MOV A, 01	E5, 01
6005	MOVC A, @A+DPTR	93
6006	INC A	04
6007	MOVX @DPTR, A	F0
6008	LCALL 05D2	12, 05, D2
600B	SJMP0006	02, 00, 06