MICROPROCESSOR AND INTERFACING LAB
MICROPROCESSOR AND INTERFACING LAB

LIST OF EXPERIMENTS
V SEM. (ECE, CSE, IT, BME)

1. STUDY ARCHITECTURE OF 8085 & FAMILIARIZATION WITH ITS HARDWARE, COMMANDS & OPERATION OF MICROPROCESSOR KIT.

2. WRITE A PROGRAM USING 8085 & VERIFY FOR:
   (A) ADDITION OF TWO 8-BIT NUMBERS.
   (B) ADDITION OF TWO 16-BIT NUMBERS (WITH CARRY)

3. WRITE A PROGRAM USING 8085 & VERIFY FOR:
   (A) SUBTRACTION OF TWO 8-BIT NUMBERS (DISPLAY OF BARROW)
   (B) SUBTRACTION OF TWO 16-BIT NUMBERS (DISPLAY OF BARROW)

4. WRITE A PROGRAM USING 8085 FOR MULTIPLICATION OF TWO 8-BIT NUMBERS BY REPEATED ADDITION METHOD CHECK MINIMUM NUMBER OF ADDITION & TEST FOR TYPICAL DATA.

5. WRITE A PROGRAM USING 8085 FOR MULTIPLICATION OF TWO 8-BIT NUMBERS BY BIT ROTATION METHOD & VERIFY.

6. WRITE A PROGRAM USING 8085 FOR DIVISION OF TWO 8-BIT NUMBERS BY REPEATED SUBTRACTION METHOD & TEST FOR TYPICAL DATA.

7. WRITE A PROGRAM USING 8085 FOR DIVISION OF TWO 8-BIT NUMBERS BY BIT ROTATION METHOD & TEST FOR TYPICAL DATA.

8. WRITE A PROGRAM USING 8085 FOR FINDING SQUARE OF A NUMBER USING LOOK-UP TABLE & VERIFY.

9. WRITE A PROGRAM USING 8085 FOR FINDING SQUARE-ROOT OF A NUMBER.

10. STUDY OF 8086 MICROPROCESSOR KIT.

11. WRITE A PROGRAM USING 8086 FOR ADDITION OF TWO 16 BIT NUMBERS.

12. WRITE A PROGRAM USING 8086 FOR DIVISION OF A DEFINED DOUBLE WORD BY ANOTHER WORD & VERIFY.

13. WRITE A PROGRAM USING 8086 FOR COPYING 12 BYTES OF DATA FROM SOURCE TO DESTINATION & VERIFY

14. WRITE A PROGRAM USING 8086 & VERIFY FOR FINDING THE LARGEST NUMBER FROM AN ARRAY.

15. WRITE A PROGRAM USING 8086 FOR ARRANGING AN ARRAY OF NUMBERS IN DESCENDING ORDER & VERIFY.
EXPERIMENT NO. 1

AIM: STUDY OF 8085-MICROPROCESSOR KIT.

APPARATUS: 8085 microprocessor kit.

THEORY:

Intel 8085 is an 8-bit microprocessor. It is 40-pin IC package fabricated on a single LSI chip. It uses a single +5 V supply. Its clock speed is about 3 MHz. It consists of three main sections:

1. ALU (Arithmetic and logic unit):

   The ALU performs the arithmetic and logical operation, addition, subtraction, logical AND, OR, EX-OR, Complement, Increment, Decrement, shift, clear.

2. Timing and Control Unit:

   It generates timing and control signals, which are necessary for the execution of instruction.

3. Registers:

   These are used for temporary storage of data and instruction. INTEL 8085 has following registers:
   
   i) One 8 bit accumulator
   ii) Six 8 bit registers (B, C, D, E, H, L)
   iii) One 16 bit stack pointer, SP
   iv) One 16 bit program counter, PC
   v) Instruction register
   vi) Status register
   vii) Temporary registers

   PC contains the address of next instruction.
   IR holds the instruction until it is decoded.
   SP holds the address of the stack top.
   Accumulator is used during execution of program for temporary storage of data.

   Status flags are as follows:
   
   i) Carry (CS)
   ii) Zero (Z)
   iii) Sign (S)
   iv) Parity (P)
   v) Auxiliary Carry (AC)

   PSW

   This 8-bit program status word includes status flags and three undefined bits.

Data and Address bus

   Data bus is 8-bit wide and 8 bits of data can be transmitted in parallel. It has 16-bit wide address bus as the memory addresses are of 16 bits.
PIN CONFIGURATION

A8-A15 (Output):-
These are address bus and used for the most significant bits of memory address.

AD0-AD7 (Input/Output):-
These are time multiplexed address data bus. These are used for the least significant 8 bits of the memory address during first clock cycle and then for data during second and third clock cycle.

ALE (Address Latch Enable)
It goes high during the 1st clock cycle of a machine. It enables the lower 8 bits of address to be latched either in the memory or external latch.

IO/M
It is status signal, when it goes high; the address on address bus is for I/O device, otherwise for memory.

So, S1
These are status signals to distinguish various types of operation

<table>
<thead>
<tr>
<th>S1</th>
<th>So</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Halt</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Write</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Read</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Fetch</td>
</tr>
</tbody>
</table>

RD (output)
It is used to control read operation.

WR (output)
It is used to control write operation.

HOLD (input)
It is used to indicate that another device is requesting the use of address & data bus.
**HLDA (output)**
It is acknowledgement signal used to indicate HOLD request has been received.

**INTR (input)**
When it goes high, microprocessor suspends its normal sequence of operations.

**INTA (output)**
It is interrupt acknowledgement signal sent by microprocessor after INTR is received.

**RST 5,5,6,5,7,5 and TRAP**
These are various interrupt signals. Among them TRAP is having highest priority

**RESET IN (input)**
It resets the PC to zero.

**RESET OUT (output)**
It indicates that CPU is being reset.

**X1, X2 (input)**
This circuitry is required to produce a suitable clock for the operation of microprocessor.

**Clk (output)**
It is clock output for user. Its frequency is same at which processor operates.

**SID (input)**
It is used for data line for serial input.

**SOD (output)**
It is used for data line for serial output.

**Vcc**
+5 volts supply

**Vss**
Ground reference
EXPERIMENT NO. 2(A)

AIM: WRITE A PROGRAM USING 8085 & VERIFY FOR:
(a) ADDITION OF TWO 8-BIT NUMBERS.

APPARATUS: 8085 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program)

<table>
<thead>
<tr>
<th>Memory address</th>
<th>Machine code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td>21,01,75</td>
<td>LXI</td>
<td>H,7501</td>
<td>Get address of 1st no. in HL pair</td>
</tr>
<tr>
<td>7003</td>
<td>7E</td>
<td>MOV</td>
<td>A,M</td>
<td>Move 1st no. in accumulator</td>
</tr>
<tr>
<td>7004</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td>HL points the address 7502H</td>
</tr>
<tr>
<td>7005</td>
<td>86</td>
<td>ADD</td>
<td>M</td>
<td>Add the 2nd no.</td>
</tr>
<tr>
<td>7006</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td>HL points 7503H</td>
</tr>
<tr>
<td>7007</td>
<td>77</td>
<td>MOV</td>
<td>M,A</td>
<td>Store result in 7503H</td>
</tr>
<tr>
<td>7008</td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Terminate</td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:-

```
START

Get the first No.

Get the second No.

Add. Two numbers

Store the result

END
```
PROCEDURE:-

ANSHUMAN
S
Enter Enter
Program Address
Write Program

Execution Steps

Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

SCIENTECH
Reset
Exmem
Starting Address
Next
Write Program

Execution Steps

Reset
GO
Starting Address
Fill
Reset
Exmem
Result Address

INPUT DATA
7501- 13H
7502- 12H

OUTPUT DATA
7503- 25H

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 2(B)

**AIM**: WRITE A PROGRAM USING 8085 & VERIFY FOR:
(b) ADDITION OF TWO 16-BIT NUMBERS(WITH CARRY).

**APPARATUS**: 8085 microprocessor kit, 5V power supply, Keyboard.

**THEORY (Program)**

<table>
<thead>
<tr>
<th>Memory address</th>
<th>Label</th>
<th>Machine code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td></td>
<td>2A,01,76</td>
<td>LHLD</td>
<td>7601H</td>
<td>Get 1st no. in HL pair from memory (7601)</td>
</tr>
<tr>
<td>7003</td>
<td></td>
<td>EB</td>
<td>XCHG</td>
<td></td>
<td>Exchange cont. of DE ↔ HL</td>
</tr>
<tr>
<td>7004</td>
<td></td>
<td>2A,03,76</td>
<td>LHLD</td>
<td>7603H</td>
<td>Get 2nd no. in HL pair from location 7603</td>
</tr>
<tr>
<td>7007</td>
<td></td>
<td>0E,00</td>
<td>MVI</td>
<td>C.00H</td>
<td>Clear reg. C.</td>
</tr>
<tr>
<td>7009</td>
<td></td>
<td>19</td>
<td>DAD</td>
<td>D</td>
<td>Get HL+DE &amp; store result in HL</td>
</tr>
<tr>
<td>700A</td>
<td></td>
<td>D2,12,70</td>
<td>JNC</td>
<td>7012(loop)</td>
<td>If no carry move to loop/if carry then move to next step.</td>
</tr>
<tr>
<td>700D</td>
<td></td>
<td>0C</td>
<td>INR</td>
<td>C</td>
<td>Increment reg C</td>
</tr>
<tr>
<td>700E</td>
<td></td>
<td>79</td>
<td>MOV</td>
<td>A,C</td>
<td>Move carry from reg. C to reg. A</td>
</tr>
<tr>
<td>7011</td>
<td></td>
<td>32,02,75</td>
<td>STA</td>
<td>7502</td>
<td>Store carry at 7502H</td>
</tr>
<tr>
<td>7012</td>
<td>loop</td>
<td>22,00,75</td>
<td>SHLD</td>
<td>7500</td>
<td>Store result in 7500H.</td>
</tr>
<tr>
<td>7015</td>
<td></td>
<td>CF</td>
<td>RST1</td>
<td></td>
<td>Terminate</td>
</tr>
</tbody>
</table>
CIRCUIT DIAGRAM / BLOCK DIAGRAM:-

START

- Get the first No. in HL
- Exchange DE <-> HL
- Get the second No. in HL
- Add. Two numbers
- If carry
  - Yes: Increment reg. C
  - No: Go to previous step
- Move C to A
- Store Accumulator
- Store HL result

END

PROCEDURE:-

ANSHUMAN

Enter Enter

Program Address

Write Program

Execution Steps

Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

SCIENTECH

Reset
Exmem
Starting Address
Next
Write Program
Execution Steps

Reset
GO
Starting Address
Fill
Reset
Exmem
Result Address
INPUT DATA
7601 : 13H
7602 : 31H
7603 : 12H
7604 : 10H

OUTPUT DATA
7500 : 25H
7501 : 41H
7502 : 00H

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 3(A)

AIM: WRITE A PROGRAM USING 8085 & VERIFY FOR:
(A) SUBTRACTION OF TWO 8-BIT NUMBERS. (DISPLAY OF BARROW).

APPARATUS: 8085 microprocessor kit, 5V power supply, Keyboard.

THEORY(Program):

<table>
<thead>
<tr>
<th>Memory address</th>
<th>Opcode</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td>21,01,75</td>
<td>LXI</td>
<td>H, 7501</td>
<td>Get address of Ist no. in HL pair</td>
</tr>
<tr>
<td>7003</td>
<td>7E</td>
<td>MOV</td>
<td>A, M</td>
<td>Move Ist no. in accumulator</td>
</tr>
<tr>
<td>7004</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td>HL points 7502H.</td>
</tr>
<tr>
<td>7005</td>
<td>96</td>
<td>SUB</td>
<td>M</td>
<td>Subtract 2nd no. from Ist no.</td>
</tr>
<tr>
<td>7006</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td>HL points 7503 H.</td>
</tr>
<tr>
<td>7007</td>
<td>77</td>
<td>MOV</td>
<td>M, A</td>
<td>Move contents of acc. to memory</td>
</tr>
<tr>
<td>7008</td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM :-

START

Get the first No.

Get the second No.

Subtract second number from first number

Store the result

END
PROCEDURE:-

ANSHUMAN
S
Enter Enter
Program Address
Write Program

Execution Steps

Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

INPUT DATA

7501  :  20H
7502  :  10H

OUTPUT DATA

7503  :  10H

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 3 (B)

AIM : WRITE A PROGRAM USING 8085 & VERIFY FOR :
(B) SUBTRACTION OF TWO 16-BIT NUMBERS. (DISPLAY OF BARROW)

APPARATUS : 8085 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program) :

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td>2A, 01,75</td>
<td>LHLD</td>
<td>7501 H</td>
<td>Get 1st 16 bit no. in HL pair</td>
</tr>
<tr>
<td>7003</td>
<td>EB</td>
<td>XCHG</td>
<td></td>
<td>Exchange HL pair with DE.</td>
</tr>
<tr>
<td>7004</td>
<td>2A, 03,75</td>
<td>LHLD</td>
<td>7503 H</td>
<td>Get 2nd 16 bit no. in HL pair</td>
</tr>
<tr>
<td>7007</td>
<td>7B</td>
<td>MOV</td>
<td>A, E</td>
<td>Get lower byte of 1st no.</td>
</tr>
<tr>
<td>7008</td>
<td>95</td>
<td>SUB</td>
<td>L</td>
<td>Subtract lower byte of 2nd no.</td>
</tr>
<tr>
<td>7009</td>
<td>6F</td>
<td>MOV</td>
<td>L, A</td>
<td>Store the result in reg. L</td>
</tr>
<tr>
<td>700A</td>
<td>7A</td>
<td>MOV</td>
<td>A, D</td>
<td>Get higher byte of 1st no.</td>
</tr>
<tr>
<td>700B</td>
<td>96</td>
<td>SBB</td>
<td>H</td>
<td>Subtract higher byte of 2nd no. with borrow</td>
</tr>
<tr>
<td>700C</td>
<td>67</td>
<td>MOV</td>
<td>H,A</td>
<td>Move from acc. To H</td>
</tr>
<tr>
<td>700D, E, F</td>
<td>22, 05,75</td>
<td>SHLD</td>
<td>7505H</td>
<td>Store 16 bit result at 7505&amp;7506</td>
</tr>
<tr>
<td>7010</td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Terminate</td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM :-

```
START

Get the lower byte of first No.

Get the lower byte of second No.

Sub. lower byte of second No. from lower byte of first No.

Get the higher byte of first No.

Get the higher byte of second

Sub. higher byte of second No. and borrow from the previous sub.

Store the result

END
```
PROCEDURE:-

ANSHUMAN
S
Enter Enter
Program Address
Write Program

Execution Steps

Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter -2
Register Name

INPUT DATA

7501 : 30H
7502 : 40H
7503 : 10H
7504 : 20H

OUTPUT DATA

7505 : 20H
7506 : 20H

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 4

AIM: WRITE A PROGRAM USING 8085 FOR MULTIPLICATION OF TWO 8-BIT NUMBERS BY REPEATED ADDITION METHOD CHECK MINIMUM NUMBER OF ADDITION & TEST FOR TYPICAL DATA

APPARATUS: 8085 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program):

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Label</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td>0E,25</td>
<td>MVI</td>
<td>C,25</td>
<td></td>
<td>Move the no. in reg. C</td>
</tr>
<tr>
<td>7002</td>
<td>1E,05</td>
<td>MVI</td>
<td>E,05</td>
<td></td>
<td>Move the no. in reg. E</td>
</tr>
<tr>
<td>7004</td>
<td>06,00</td>
<td>MVI</td>
<td>B,00</td>
<td></td>
<td>Clear reg. B</td>
</tr>
<tr>
<td>7006</td>
<td>21,00,00</td>
<td>LXI</td>
<td>H,0000</td>
<td>Initial Product=0000</td>
<td></td>
</tr>
<tr>
<td>7009</td>
<td>UP1: 09</td>
<td>DAD</td>
<td>B</td>
<td>HL+BC=&gt;HL</td>
<td></td>
</tr>
<tr>
<td>700A</td>
<td>1D</td>
<td>DCR</td>
<td>E</td>
<td>Decrement reg. E</td>
<td></td>
</tr>
<tr>
<td>700B</td>
<td>C2,09,70</td>
<td>JNZ</td>
<td>UP1(7009)</td>
<td>Jump if not zero to location up1</td>
<td></td>
</tr>
<tr>
<td>700E</td>
<td>22,00,75</td>
<td>SHLD</td>
<td>7500</td>
<td>Store HL at 7500</td>
<td></td>
</tr>
<tr>
<td>7011</td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Terminate</td>
<td></td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:-

START

Get the first No. in reg. C

Get the second No. in reg. E (Counter)

Initial Product in HL =0000

Add first No. in initial Product

Is Counter in reg. E=0

NO

Yes

Store Product in HL pair

END
PROCEDURE:-

ANSHUMAN

S
Enter Enter
Program Address
Write Program

Execution Steps

Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter
Name
Register

SCIENTECH

Reset
Exmem
Starting Address
Next
Write Program

Execution Steps

Reset
GO
Starting Address
Fill
Reset
Exmem
Result Address

INPUT DATA

1) Reg.C : 25H
   Reg.E : 05H
   Reg.B : 00H

OUTPUT DATA

HL pair : 00B9H

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 5

AIM : WRITE A PROGRAM USING 8085 FOR MULTIPLICATION OF TWO 8-BIT NUMBERS BY BIT ROTATION METHOD & VERIFY.

APPARATUS : 8085 microprocessor kit, 5 V power supply, Keyboard.

THEORY (Program)

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Label</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td></td>
<td>2A,01,75</td>
<td>LHLD</td>
<td>7501 H</td>
<td>Get Multiplicand in H-L pair.</td>
</tr>
<tr>
<td>7003</td>
<td>EB</td>
<td>XCHG</td>
<td></td>
<td></td>
<td>Exchange HL pair with DE pair</td>
</tr>
<tr>
<td>7004</td>
<td></td>
<td>3A,03,75</td>
<td>LDA</td>
<td>7503 H</td>
<td>Get 2nd no. in acc.</td>
</tr>
<tr>
<td>7007</td>
<td></td>
<td>21,00,00</td>
<td>LXI</td>
<td>H,0000</td>
<td>Initial product in HL=00</td>
</tr>
<tr>
<td>700A</td>
<td>Loop</td>
<td>0E,08</td>
<td>MVI</td>
<td>C,08H</td>
<td>Count=08 in reg .C</td>
</tr>
<tr>
<td>700C</td>
<td></td>
<td>19</td>
<td>DAD</td>
<td>H</td>
<td>Shift partial product left by 1 bit</td>
</tr>
<tr>
<td>700D</td>
<td></td>
<td>17</td>
<td>RAL</td>
<td></td>
<td>Rotate multiplication by 1 bit. Is multiplier = 1?</td>
</tr>
<tr>
<td>700E</td>
<td></td>
<td>D2,12,70</td>
<td>JNC</td>
<td>Ahead(7012)</td>
<td>No, go to ahead</td>
</tr>
<tr>
<td>7011</td>
<td></td>
<td>19</td>
<td>DAD</td>
<td>D</td>
<td>Product=Product + Multiplicand</td>
</tr>
<tr>
<td>7012</td>
<td>Ahead</td>
<td>0D</td>
<td>DCR</td>
<td>C</td>
<td>Decrement Count</td>
</tr>
<tr>
<td>7013</td>
<td></td>
<td>C2,0C,70</td>
<td>JNZ</td>
<td>Loop(700C)</td>
<td></td>
</tr>
<tr>
<td>7016</td>
<td></td>
<td>22,04,75</td>
<td>SHLD</td>
<td>7504</td>
<td>Store result</td>
</tr>
<tr>
<td>7019</td>
<td></td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Terminate</td>
</tr>
</tbody>
</table>
Get Multiplicand, Get Multiplier

Initial value of Product = 00
Count = 08

Shift Product left one Bit
Shift Multiplier left one Bit

IS Carry from Multiplier?

YES
Product = Product + Multiplicand
Count = Count - 1

NO

IS Count = 0?

YES
Store Product

END
PROCEDURE:-

ANSHUMAN
S
Enter Enter
Program Address
Write Program

Execution Steps

Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

SCIENTECH
Reset
Exmem
Starting Address
Next
Write Program

Execution Steps

Reset
GO
Starting Address
Fill
Reset
Exmem
Result Address

INPUT DATA

7501- 25H
7502- 00H
7503- 05H

OUTPUT DATA

7504- B9H
7505- 00H

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 6

AIM: WRITE A PROGRAM USING 8085 FOR DIVISION OF TWO 8-BIT NUMBERS BY REPEATED SUBTRACTION METHOD & TEST FOR TYPICAL DATA.

APPARATUS: 8085 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program):

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Label</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td></td>
<td>3A,01,75</td>
<td>LDA</td>
<td>Divisor(7501)</td>
<td></td>
</tr>
<tr>
<td>7003</td>
<td></td>
<td>47</td>
<td>MOV B,A</td>
<td></td>
<td>Take divisor in reg.B</td>
</tr>
<tr>
<td>7004</td>
<td></td>
<td>3A,02,75</td>
<td>LDA</td>
<td>Dividend(7502)</td>
<td>Take dividend in reg.A</td>
</tr>
<tr>
<td>7007</td>
<td></td>
<td>0E,00</td>
<td>MVI</td>
<td>C.00</td>
<td>Quotient=00</td>
</tr>
<tr>
<td>7009</td>
<td></td>
<td>B8</td>
<td>CMP</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>700A</td>
<td>loop1</td>
<td>DA,13,70</td>
<td></td>
<td>JC</td>
<td>Loop(7013)</td>
</tr>
<tr>
<td>700D</td>
<td></td>
<td>90</td>
<td>SUB</td>
<td>B</td>
<td>Dividend-divisor=&gt;A</td>
</tr>
<tr>
<td>700E</td>
<td></td>
<td>0C</td>
<td>INR</td>
<td>C</td>
<td>C=C+1</td>
</tr>
<tr>
<td>700F</td>
<td></td>
<td>B8</td>
<td>CMP</td>
<td>B</td>
<td>Is dividend &lt; divisor</td>
</tr>
<tr>
<td>7010</td>
<td>loop</td>
<td>D2,0D,70</td>
<td>JNC</td>
<td>Loop1(700D)</td>
<td>If not, go back</td>
</tr>
<tr>
<td>7013</td>
<td></td>
<td>32,03,75</td>
<td>STA</td>
<td>Remainder(7503)</td>
<td>Store Remainder</td>
</tr>
<tr>
<td>7016</td>
<td></td>
<td>79</td>
<td>MOV</td>
<td>A,C</td>
<td></td>
</tr>
<tr>
<td>7017</td>
<td></td>
<td>32,04,75</td>
<td>STA</td>
<td>Quotient(7504)</td>
<td>Store Quotient</td>
</tr>
<tr>
<td>701A</td>
<td></td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Terminate.</td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:-

START


Take quotient in reg. C= 00

IS Dividend > Divisor?

YES

A= Dividend - Divisor

Increment Counter by 1

NO

Quotient is in reg. C

Reminder is in acc.

END
PROCEDURE:-

ANSHUMAN
S
Enter Enter
Program Address
Write Program

Execution Steps
Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

INPUT DATA
7501- Divisor
7502-Dividend

OUTPUT DATA
7503-Remainder
7504-Quotient

PRECAUTIONS:-
Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 7

AIM : WRITE A PROGRAM USING 8085 FOR DIVISION OF TWO 8-BIT NUMBERS BY BIT ROTATION METHOD & TEST FOR TYPICAL DATA.

APPARATUS : 8085 microprocessor kit, Keyboard, and 5V Power Supply.

THEORY (Program)

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Label</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td>2A, 01,75</td>
<td>LHLD</td>
<td>7501 H</td>
<td></td>
<td>Enter the 16 bit address in HL pair</td>
</tr>
<tr>
<td>7003</td>
<td>3A, 03,75</td>
<td>LDA</td>
<td>7503 H</td>
<td></td>
<td>Get divisor from 7503</td>
</tr>
<tr>
<td>7006</td>
<td>47</td>
<td>MOV</td>
<td>B, A</td>
<td></td>
<td>Divisor in register B</td>
</tr>
<tr>
<td>7007</td>
<td>0E, 08</td>
<td>MVI</td>
<td>C, 08</td>
<td></td>
<td>Count = 08 in register C.</td>
</tr>
<tr>
<td>7009</td>
<td>Loop</td>
<td>29</td>
<td>DAD H</td>
<td></td>
<td>Shift dividend and quotient left by one bit.</td>
</tr>
<tr>
<td>700A</td>
<td>7C</td>
<td>MOV</td>
<td>A, H</td>
<td></td>
<td>Most significant bits of dividend in acc.</td>
</tr>
<tr>
<td>700B</td>
<td>90</td>
<td>SUB</td>
<td>B</td>
<td></td>
<td>Subtract divisor from MSB of dividend.</td>
</tr>
<tr>
<td>700C</td>
<td>DA, 11, 70</td>
<td>JC</td>
<td>Ahead(7011)</td>
<td>Is MSB of dividend&gt;divisor? No, go to AHEAD.</td>
<td></td>
</tr>
<tr>
<td>700F</td>
<td>67</td>
<td>MOV</td>
<td>H, A</td>
<td></td>
<td>MSB of dividend in reg. H</td>
</tr>
<tr>
<td>7010</td>
<td>2C</td>
<td>INR</td>
<td>L</td>
<td></td>
<td>Yes, add 1 to quotient.</td>
</tr>
<tr>
<td>7011</td>
<td>Ahead</td>
<td>0D</td>
<td>DCR C</td>
<td></td>
<td>Decrement count.</td>
</tr>
<tr>
<td>7012</td>
<td>C2, 09,70</td>
<td>JNZ</td>
<td>Loop(7009)</td>
<td>Is count=0? No, jump to loop.</td>
<td></td>
</tr>
<tr>
<td>7015</td>
<td>22,04,75</td>
<td>SHLD</td>
<td>7504 H</td>
<td></td>
<td>Store quotient in 7504 and remainder in 7505 H.</td>
</tr>
<tr>
<td>7018</td>
<td>CF</td>
<td>RST1</td>
<td></td>
<td></td>
<td>Stop.</td>
</tr>
</tbody>
</table>
START

Get Dividend, Get Divisor
Count = 08, Quotient = 00

Shift Dividend left one Bit
Shift Quotient left one Bit

IS MSB Of Dividend > Divisor?

NO

YES

Quotient = Quotient + 1

8 MSBs of Dividend = 8 MSBs of Dividend - Divisor

Count = Count - 1

IS Count = 0?

NO

YES

Store Product

END

IS MSB Of Dividend > Divisor?
PROCEDURE:-

ANSHUMAN
S
Enter Enter
Program Address
Write Program

Execution Steps
Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

INPUT DATA
7501- LSB of dividend
7502- MSB of dividend
7503- Divisor

OUTPUT DATA
7504- Quotient
7505- Remainder

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 8

AIM: WRITE A PROGRAM USING 8085 FOR FINDING SQUARE OF A NUMBER USING LOOK-UP TABLE & VERIFY

APPARATUS: 8085 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program):

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>3A,00,25</td>
<td>LDA</td>
<td>2500 H</td>
<td>Get 1st no. in acc.</td>
</tr>
<tr>
<td>2003</td>
<td>6F</td>
<td>MOV</td>
<td>L,A</td>
<td>Move From A into reg. L</td>
</tr>
<tr>
<td>2004</td>
<td>26,26</td>
<td>MVI</td>
<td>H,26H</td>
<td>Get 26 in reg H</td>
</tr>
<tr>
<td>2006</td>
<td>7E</td>
<td>MOV</td>
<td>A,M</td>
<td>Square of data in accumulator</td>
</tr>
<tr>
<td>2007</td>
<td>32,01,25</td>
<td>STA</td>
<td>2501 H</td>
<td>Store square in 2501 H</td>
</tr>
<tr>
<td>200A</td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>

LOOK-UP TABLE

<table>
<thead>
<tr>
<th>Address</th>
<th>Square</th>
</tr>
</thead>
<tbody>
<tr>
<td>2600</td>
<td>00</td>
</tr>
<tr>
<td>2601</td>
<td>01</td>
</tr>
<tr>
<td>2602</td>
<td>04</td>
</tr>
<tr>
<td>2603</td>
<td>09</td>
</tr>
<tr>
<td>2604</td>
<td>16</td>
</tr>
<tr>
<td>2605</td>
<td>25</td>
</tr>
<tr>
<td>2606</td>
<td>36</td>
</tr>
<tr>
<td>2607</td>
<td>49</td>
</tr>
<tr>
<td>2608</td>
<td>64</td>
</tr>
<tr>
<td>2609</td>
<td>81</td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:-

START

Initialize lookup table

Initialize source memory pointer
Initialize destination memory pointer

Get the number

Find the square

Store square in the destination memory location

END
PROCEDURE:-

ANSHUMAN
  S
Enter Enter
Program Address
Write Program
  Execution Steps
Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

INPUT DATA
  2500- 07H

OUTPUT DATA
  2501- 49H

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 9

AIM: WRITE A PROGRAM USING 8085 FOR FINDING SQUARE-ROOT OF A NUMBER

APPARATUS: 8085 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program):

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Label</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td></td>
<td>0E,01</td>
<td>MVI</td>
<td>C,01H</td>
<td>Place 01 in reg.C</td>
</tr>
<tr>
<td>2002</td>
<td></td>
<td>06,01</td>
<td>MVI</td>
<td>B,01H</td>
<td>Place odd number 1 in reg.B</td>
</tr>
<tr>
<td>2004</td>
<td></td>
<td>3E,36</td>
<td>MVI</td>
<td>A,36</td>
<td>Load accumulator with the given number</td>
</tr>
<tr>
<td>2006</td>
<td>Loop</td>
<td>90</td>
<td>SUB</td>
<td>B</td>
<td>Subtract odd number from the accumulator</td>
</tr>
<tr>
<td>2007</td>
<td></td>
<td>CA,10,20</td>
<td>JZ</td>
<td>Ahead(2010)</td>
<td>If accumulator contents are zero, go to Ahead</td>
</tr>
<tr>
<td>200A</td>
<td></td>
<td>0C</td>
<td>INR</td>
<td>C</td>
<td>Increment reg. C</td>
</tr>
<tr>
<td>200B</td>
<td></td>
<td>04</td>
<td>INR</td>
<td>B</td>
<td>Increment odd number</td>
</tr>
<tr>
<td>200C</td>
<td></td>
<td>04</td>
<td>INR</td>
<td>B</td>
<td>Increment odd number</td>
</tr>
<tr>
<td>200D</td>
<td></td>
<td>C3,06,20</td>
<td>JMP</td>
<td>Loop(2006)</td>
<td>Repeat subtraction</td>
</tr>
<tr>
<td>2010</td>
<td>Ahead</td>
<td>79</td>
<td>MOV</td>
<td>A,C</td>
<td>Move the contents of C reg. to accumulator</td>
</tr>
<tr>
<td>2011</td>
<td></td>
<td>32,50,20</td>
<td>STA</td>
<td>2050H</td>
<td>Store the result in the memory location 2050H.</td>
</tr>
<tr>
<td>2014</td>
<td>CF</td>
<td>RST1</td>
<td></td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:

START

1. Place 01 in the subtraction counter register
2. Place odd numbers 1 in any one of the register

Load the accumulator with the given

Subtract odd number from the contents of accumulator

Is remainder=0
Or negative

YES

NO

Increment subtraction counter register

Odd number=odd no. +2

Store the result

END
PROCEDURE:-

ANSHUMAN                                                            SCIENTECH
    S                                                                 Reset
Enter Enter                                                        Exmem
Program Address                                                    Starting Address
Write Program                                                      Next
Write Program                                                      Write Program

Execution Steps
Esc                                                               Execution Steps
G                                                                Reset
Enter-enter                                                        GO
Prog. Address                                                       Starting Address
Enter                                                            Fill
S                                                                Reset
Enter                                                            Exmem
Any key-2                                                          Result Address
Enter
Name
Register

INPUT DATA

    2500-10H
    2501-00H

OUTPUT DATA

    2550-04H

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO .10

AIM : STUDY OF 8086 MICROPROCESSOR KIT.

APPARATUS: 8086 microprocessor kit.

THEORY : The 8086 is a 16-bit, N-channel, HMOS microprocessor. The term HMOS is used for “high-speed MOS”. The 8086 uses 20 address lines and 16 data lines. It can directly address up to $2^{20} = 1$ Mbytes of memory. The 16-bit data word is divided into a low-order byte and a high-order byte. The 20 address lines are time multiplexed lines. The 16 low-order address lines are time multiplexed with data, and the 4 high-order address lines are time multiplexed with status signals.

OPERATING MODES OF 8086

There are two modes of operation for Intel 8086, namely the minimum mode and the maximum mode. When only one 8086 CPU is to be used in a microcomputer system the 8086 is used in the minimum mode of operation. In this mode the CPU issues the control signals required by memory and I/O devices. In case of maximum mode of operation control signals are issued by Intel 8288 bus controller which is used with 8086 for this very purpose. When MN/MX is high the CPU operates in the minimum mode. When it is low the CPU operates in the maximum mode.

Pin Description For Minimum Mode

For the minimum mode of operation the pin MN/MX is connected to 5V d.c supply. The description of the pins from 24 to 31 for the minimum mode is as follows:

**INTA (Output):** Pin no. 24. Interrupt acknowledge. On receiving interrupt signal the processor issues an interrupt acknowledge signal. It is active LOW.

**ALE (Output):** Pin no. 25. Address latch enable. It goes HIGH during T1. The microprocessor sends this signal to latch the address into the Intel 8282/8283 latch.

**DEN (Output):** Pin no. 26. Data enable. When Intel 8286/8287 octal bus transceiver is used this signal acts as an output enable signal. It is active LOW.

**DT/R (Output):** Pin no. 27. Data Transmit/Receive. When Intel 8286/8287 octal bus transceiver is used this signal controls the direction of data flow through the transceiver. When it is High data are sent out. When it is LOW data are received.

**M/IO (Output):** Pin no. 28. Memory or I/O access. When it is HIGH the CPU wants to access memory. When it is LOW the CPU wants to access I/O device.

**WR (Output):** Pin no. 29. Write. When it is LOW the CPU performs memory or I/O write Operation.

**HLDA (Output):** Pin no. 30. HOLD acknowledge. It is issued by the processor when it receives HOLD signal. It is active HIGH signal. When HOLD request is removed HLDA goes LOW.

**HOLD (Output):** Pin no. 31. Hold. when another device in microcomputer system wants to use the address and data bus, it sends a HOLD request to CPU through this pin. It is an active HIGH signal.

Pin Description For Maximum Mode

For the maximum mode of operation the pin MN/MX is made LOW. It is grounded. The description of the pins from 24 to 31 is as follows:

**QS1,QS0 (Output):** Pin no. 24, 25. Instruction Queue status. Logic are given below:

<table>
<thead>
<tr>
<th>QS1</th>
<th>QS0</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>No operation</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1st byte of opcode from queue</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Empty the queue</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Subsequent byte from queue</td>
</tr>
</tbody>
</table>
**S0, S1, S2 (Output)**: Pin nos. 26, 27, 28. Status signals. These signals are connected to the bus controller Intel 8288. The bus controller generates memory and I/O access control signals. Table for status signals is:

<table>
<thead>
<tr>
<th>S2</th>
<th>S1</th>
<th>S0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**LOCK (Output)**: Pin no. 29. It is an active LOW signal. When it is LOW all interrupts are masked and no HOLD request is granted. In a multiprocessor system all other processors are informed by this signal that they should not ask the CPU for relinquishing the bus control.

**RQ / GT₁, RQ / GT₀ (Bidirectional)**: Pin no. 30, 31. Local bus Priority control. Other processors ask the CPU through these lines to release the local bus. **RQ / GT₁** has higher priority than **RQ / GT₀**.

**FUNCTIONAL UNITS OF 8086**: The 8086 contains two functional units: a bus interface unit (BIU) and an execution unit (EU). The general purpose registers, stack pointer, base pointer and index registers, ALU, flag register (FLAGS), instruction decoder and timing and control unit constitute execution unit (EU). The segment registers, instruction pointer and 6-byte instruction queue are associated with the bus interface unit (BIU).
The Intel 8086 contains the following registers:

a) General Purpose Register
b) Pointer and Index Registers
c) Segment Registers
d) Instruction Registers
e) Status Flags
EXPERIMENT NO. 11

AIM: WRITE A PROGRAM USING 8086 FOR ADDITION OF TWO 16 BIT NUMBERS.

APPARATUS: 8086 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program):

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Opcodes</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>B8,34,12</td>
<td>MOV</td>
<td>AX,1234</td>
<td>Load 1234 in AX</td>
</tr>
<tr>
<td>1003</td>
<td>BA,65,87</td>
<td>MOV</td>
<td>DX,8765</td>
<td>Load 8765 in DX</td>
</tr>
<tr>
<td>1006</td>
<td>03,C2</td>
<td>ADD</td>
<td>AX,DX</td>
<td>Add DX with AX</td>
</tr>
<tr>
<td>1008</td>
<td>8B,C8</td>
<td>MOV</td>
<td>CX,AX</td>
<td>Move answer to CX</td>
</tr>
<tr>
<td>1009</td>
<td>CD,A5</td>
<td>INT A5</td>
<td></td>
<td>Jump to command mode saving all registers.</td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:

```
START

Get the first No.

Get the second No.

Add. Two numbers

Store the result

END
```

PROCEDURE:-

<table>
<thead>
<tr>
<th>ANSHUMAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
</tr>
<tr>
<td>Enter</td>
</tr>
<tr>
<td>Enter</td>
</tr>
<tr>
<td>SRC-SEGM Address</td>
</tr>
<tr>
<td>Enter</td>
</tr>
<tr>
<td>Program Address</td>
</tr>
<tr>
<td>Write Program</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCIENTECH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
</tr>
<tr>
<td>O</td>
</tr>
<tr>
<td>EB/AX</td>
</tr>
<tr>
<td>Starting Address</td>
</tr>
<tr>
<td>Next</td>
</tr>
<tr>
<td>Write Program</td>
</tr>
</tbody>
</table>
**Execution Steps**

Esc
G
Enter-enter
SRC-SEGM Add
Enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

**Execution Steps**

Reset
GO
Starting Address
Fill
Reset
O
EB/AX
Result Address

**INPUT DATA**

1000-1234(H)
1001-8765(H)

**OUTPUT DATA**

AX -9999(H)

**PRECAUTIONS:**

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 12

AIM: WRITE A PROGRAM USING 8086 FOR DIVISION OF A DEFINED DOUBLE WORD BY ANOTHER WORD & VERIFY.

APPARATUS: 8086 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program)

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>B8,78,56</td>
<td>MOV</td>
<td>AX,5678H</td>
<td>Move 5678 to AX</td>
</tr>
<tr>
<td>1003</td>
<td>BA,34,12</td>
<td>MOV</td>
<td>DX,1234H</td>
<td>Move 1234 to DX</td>
</tr>
<tr>
<td>1006</td>
<td>B9,25,25</td>
<td>MOV</td>
<td>CX,2525</td>
<td>Move 2525 to CX</td>
</tr>
<tr>
<td>1009</td>
<td>F7,F1</td>
<td>DIV</td>
<td>CX</td>
<td>Divide AX&amp;DX by CX</td>
</tr>
<tr>
<td>100b</td>
<td>CD,A5</td>
<td>INT</td>
<td>A5</td>
<td></td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:--

START

Get LSB of Dividend

Get MSB of Dividend

Get Divisor in CX

Divide & Store result

END

PROCEDURE:--

SCIENTECH

ANSHUMAN

Reset

S

O

Enter Enter

EB/AX

SRC-SEGMENT Address

Starting Address

Enter

Next

Program Address

Write Program

Write Program
**Execution Steps**

Esc
G
Enter-enter
SRC-SEGM Add
Enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

**Execution Steps**

Reset
GO
Starting Address
Fill
Reset
O
EB/AX
Result Address

**INPUT DATA**

\[
\begin{align*}
AX &: 5678H \\
DX &: 1234H \\
CX &: 2525H
\end{align*}
\]

**OUTPUT DATA**

\[
\begin{align*}
AX &: 7D77(Quotient) \\
DX &: 0145(Remainder)
\end{align*}
\]

**PRECAUTIONS:-**

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 13

AIM: WRITE A PROGRAM USING 8086 FOR COPYING 12 BYTES OF DATA FROM SOURCE TO DESTINATION & VERIFY

APPARATUS: 8086 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program)

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Label</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101</td>
<td>FC</td>
<td>CLD</td>
<td></td>
<td></td>
<td>Clear direction flag DF</td>
</tr>
<tr>
<td>0102</td>
<td>BE,00,03</td>
<td>MOV</td>
<td>SI,0300</td>
<td>Source address in SI</td>
<td></td>
</tr>
<tr>
<td>0105</td>
<td>BF,02,02</td>
<td>MOV</td>
<td>DI,0202</td>
<td>Destination address in DI</td>
<td></td>
</tr>
<tr>
<td>0108</td>
<td>8B,0C</td>
<td>MOV</td>
<td>CX,[SI]</td>
<td>Count in CX</td>
<td></td>
</tr>
<tr>
<td>010A</td>
<td>46</td>
<td>INC</td>
<td>SI</td>
<td>Increment SI</td>
<td></td>
</tr>
<tr>
<td>010B</td>
<td>46</td>
<td>INC</td>
<td>SI</td>
<td>Increment SI</td>
<td></td>
</tr>
<tr>
<td>010C</td>
<td>BACK</td>
<td>A4</td>
<td>MOV</td>
<td>SB Move byte</td>
<td></td>
</tr>
<tr>
<td>010D</td>
<td>E2,FD</td>
<td>LOOP</td>
<td>BACK</td>
<td>Jump to BACK until CX becomes zero</td>
<td></td>
</tr>
<tr>
<td>010F</td>
<td>CC</td>
<td>INT</td>
<td></td>
<td>Interrupt program</td>
<td></td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:-

START

Get source address

Get destination address

Get Count in CX

Increment in SI   Increment in SI

Move bytes

IS Count =0?

NO

YES

END
INPUT DATA
0300 : 0B
0301 : 00
0302 : 03
0303 : 04
0304 : 05
0305 : 06
0306 : 15
0307 : 07
0308 : 12
0309 : 08
030A : 09
030B : 0A
030C : 0B
030D : 0E

OUTPUT DATA
0202 : 03
0203 : 04
0204 : 05
0205 : 06
0206 : 15
0207 : 07
0208 : 12
0209 : 08
020A : 09
020B : 0A
020C : 0B
020D : 0E

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
EXPERIMENT NO. 14

AIM: WRITE A PROGRAM USING 8086 & VERIFY FOR FINDING THE LARGEST NUMBER FROM AN ARRAY.

APPARATUS: 8086 microprocessor kit, 5V power supply, Keyboard.

THEORY (Program)

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Label</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101</td>
<td>B0,00,00</td>
<td>MOV AX,0000</td>
<td>Initial value for comparison</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0104</td>
<td>BE,00,02</td>
<td>MOV SI,0200</td>
<td>Memory address in SI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0107</td>
<td>8B,0C</td>
<td>MOV CX,[SI]</td>
<td>Count in CX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0109 BACK</td>
<td>46</td>
<td>INC SI</td>
<td>Increment SI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>010A A</td>
<td>46</td>
<td>INC SI</td>
<td>Increment SI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>010B B</td>
<td>3B,04</td>
<td>CMP AX,[SI]</td>
<td>Compare previous largest number with next number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>010D D</td>
<td>73,02</td>
<td>JAE GO</td>
<td>Jump if number in AX is larger i.e. CF=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>010F F</td>
<td>8B,04</td>
<td>MOV AX,[SI]</td>
<td>Save next larger number in AX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0111</td>
<td>GO</td>
<td>E2,F6 LOOP BACK</td>
<td>Jump to BACK until CX becomes zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0113</td>
<td>A3,51,02</td>
<td>MOV (0251),AX</td>
<td>Store largest number in memory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0116</td>
<td>CC</td>
<td>INT3</td>
<td>Interrupt program</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:-

START

Initialize for comparison
Initialize SI (mem. Address)

Set counter

Increment SI

Is AX > next No?

YES

Replace 2nd largest No. in AX.

NO

Is Count =0?

Yes

Store largest No.

END
PROCEDURE:-

ANSHUMAN
S
Enter Enter
Program Address
Write Program

Execution Steps
Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

INPUT DATA
0200 : 05H
0201 : 00H
0202 : 41H
0203 : 83H
0204 : 58H
0205 : 72H
0206 : 39H
0207 : 46H
0208 : 53H
0209 : 84H
020A : 30H
020B : 96H

OUTPUT DATA
251 : 30H
252 : 96H

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
Experiment No. 15

**AIM**: WRITE A PROGRAM USING 8086 FOR ARRANGING AN ARRAY OF NUMBERS IN DESCENDING ORDER & VERIFY.

**APPARATUS**: 8086 microprocessor kit, 5V power supply, Keyboard.

**THEORY** (Program)

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Label</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>0200</td>
<td></td>
<td>BE,00,03</td>
<td>MOV</td>
<td>SI,0300</td>
<td>Initialize SI Reg. with Memory Location. 0300.</td>
</tr>
<tr>
<td>0203</td>
<td></td>
<td>8B,1C</td>
<td>MOV</td>
<td>BX,[SI]</td>
<td>BX has no. of bytes</td>
</tr>
<tr>
<td>0205</td>
<td></td>
<td>4B</td>
<td>DEC</td>
<td>BX</td>
<td>Decrement the no. of bytes by one</td>
</tr>
<tr>
<td>0206</td>
<td>(3)</td>
<td>8B 0C</td>
<td>MOV</td>
<td>CX (SI)</td>
<td>Move no. of bytes in CX</td>
</tr>
<tr>
<td>0208</td>
<td></td>
<td>49</td>
<td>DEC</td>
<td>CX</td>
<td>Decrement the no. of bytes by one</td>
</tr>
<tr>
<td>0209</td>
<td></td>
<td>BE,02,03</td>
<td>MOV</td>
<td>SI,0303</td>
<td>Initialize SI reg. with the starting address of string</td>
</tr>
<tr>
<td>020C</td>
<td>(2)</td>
<td>8A,04</td>
<td>MOV</td>
<td>AL,[SI]</td>
<td>Move first data byte of string into AL</td>
</tr>
<tr>
<td>020E</td>
<td></td>
<td>46</td>
<td>INC</td>
<td>SI</td>
<td>Point at the next bytes of the string</td>
</tr>
<tr>
<td>020F</td>
<td></td>
<td>3A,04</td>
<td>COMP</td>
<td>AL,[SI]</td>
<td>Compare the two bytes of string.</td>
</tr>
<tr>
<td>0211</td>
<td></td>
<td>73,06</td>
<td>JAE</td>
<td>(1)</td>
<td>If two bytes are equal or 1st byte is above that the second byte branch to (1)</td>
</tr>
<tr>
<td>0213</td>
<td></td>
<td>86,04</td>
<td>XCHG</td>
<td>AL,[SI]</td>
<td>Else</td>
</tr>
<tr>
<td>0215</td>
<td></td>
<td>4E</td>
<td>DEC</td>
<td>SI</td>
<td>Second byte is less than first byte and swap the two bytes.</td>
</tr>
<tr>
<td>0216</td>
<td></td>
<td>88,04</td>
<td>MOV</td>
<td>[SI],AL</td>
<td>Point at next location of string</td>
</tr>
<tr>
<td>0218</td>
<td></td>
<td>46</td>
<td>INC</td>
<td>SI</td>
<td>Point at next location of string</td>
</tr>
<tr>
<td>0219</td>
<td>(1)</td>
<td>E2,F1</td>
<td>LOOP</td>
<td>(2)</td>
<td>Loop if CX is not zero</td>
</tr>
<tr>
<td>021B</td>
<td></td>
<td>4B</td>
<td>DEC</td>
<td>BX</td>
<td></td>
</tr>
<tr>
<td>021C</td>
<td></td>
<td>BE,00,03</td>
<td>MOV</td>
<td>SI,0300</td>
<td></td>
</tr>
<tr>
<td>021F</td>
<td></td>
<td>75,E5</td>
<td>JNZ</td>
<td>(3)</td>
<td></td>
</tr>
<tr>
<td>0221</td>
<td></td>
<td>F4</td>
<td>HLT</td>
<td>Halt.</td>
<td></td>
</tr>
</tbody>
</table>
CIRCUIT DIAGRAM / BLOCK DIAGRAM:

START

Initialize reg.SI (mem. Location)

Set the counter BX

Decrement the counter BX

Move count in CX

Decrement CX

Initialize reg.SI (Starting Location)

Get first no. in AL

Increment SI

Is first No. < next No

Yes

No

Swap the two bytes

Is Count =0?

No

Decrement Count

Compare rest of no.

Is Count =0?

Yes

END
PROCEDURE:-

ANSHUMAN
S
Enter Enter
Program Address
Write Program

Execution Steps

Esc
G
Enter-enter
Prog. Address
Enter
S
Enter
Any key-2
Enter-2
Register Name

SCIENTECH
Reset
Exmem
Starting Address
Next
Write Program

Execution Steps

Reset
GO
Starting Address
Fill
Reset
Exmem
Result Address

INPUT DATA
0300 : 05
0301 : 00
0302 : 20
0303 : 25
0304 : 28
0305 : 15
0306 : 07

OUTPUT DATA
0302 : 28
0303 : 25
0304 : 20
0305 : 15
0306 : 07

PRECAUTIONS:-

Make sure that all the machine codes should be as per specified in the program.
Questions-Answers based on practicals

Q.1 Explain MOV r,M?
Q.2 How many T-state are in MOV instruction?
Q.3 Explain the addressing mode of MOV r,M?
Q.4 How many machine cycles are in MOV instruction?
Q.5 What is MOV M,r?
Q.6 Which flag is affected in MOV instruction?
Q.7 What is MVI r,data?
Q.8 How many T-state are in MVI instruction?
Q.9 Explain the addressing mode of MVI r,data?
Q.10 How many machine cycles are in MVI instruction?
Q.11 Explain LXI rp,data 16?
Q.12 How many T-state are in LXI instruction?
Q.13 Explain the addressing mode of LXI rp,data?
Q.14 How many machine cycles are in LXI instruction?
Q.15 What is LDA addr?
Q.16 How many T-state are in LDA instruction?
Q.17 Explain the addressing mode of LDA addr?
Q.18 How many machine cycles are in LDA instruction?
Q.19 What is STA addr?
Q.20 How many T-state are in STA instruction?
Q.21 Explain the addressing mode of STA addr?
Q.22 How many machine cycles are in STA instruction?
Q.23 What is LHLD addr?
Q.24 How many T-state are in LHLD instruction?
Q.25 Explain the addressing mode of LHLD addr?
Q.26 How many machine cycles are in LHLD instruction?
Q.27 What is SHLD addr?
Q.28 How many T-state are in SHLD instruction?
Q.29 Explain the addressing mode of SHLD addr?
Q.30 How many machine cycles are in SHLD instruction?
Q.31 What is LDAX rp?
Q.32 How many T-state are in LDAX instruction?
Q.33 Explain the addressing mode of LDAX rp?
Q.34 How many machine cycles are in LDAX instruction?
Q.35 What is STAX rp?
Q.36 How many T-state are in STAX instruction?
Q.37 Explain the addressing mode of STAX rp?
Q.38 How many machine cycles are in STAX instruction?
Q.39 What is XCHG?
Q.40 How many T-state are in XCHG instruction?
Q.41 Explain the addressing mode of XCHG?
Q.42 How many machine cycles are in XCHG instruction?
Q.43 What is ADD r?
Q.44 How many T-state are in ADD instruction?
Q.45 Explain the addressing mode of ADD?
Q.46 How many machine cycles are in ADD instruction?
Q.47 What is ADC r?
Q.48 How many T-state are in ADC r instruction?
Q.49 Explain the addressing mode of ADC?
Q.50 How many machine cycles are in ADC instruction?
Q.51 Explain ADI data?
Q.52 How many T-states are in ADI instruction?
Q.53 Explain the addressing mode of ADI?
Q.54 How many machine cycles are in ADI instruction?
Q.55 Explain DAD rp?
Q.56 How many T-states are in DAD instruction?
Q.57 Explain the addressing mode of DAD.
Q.58 How many machine cycles are in DAD instruction?
Q.59 Explain DAA.
Q.60 What is INX rp?
**Answers**

A.1 Move the content of memory to register
A.2 Four T-state
A.3 Register indirect
A.4 two machine cycle
A.5 move the content of register to memory
A.6 none
A 7 move immediate data to register
A.8 seven T-states
A.9 immediate
A.10 three machine cycles
A.11 load register pair immediate
A.12 ten T-states
A.13 immediate
A.14 three machine cycles
A.15 load accumulator direct
A.16 thirteen T-states
A.17 direct
A.18 four
A.19 store accumulator direct
A.20 thirteen T-states
A.21 direct
A.22 four
A.23 load H-L pair direct
A.24 sixteen T-states A25 direct
A.26 five
A.27 store H-L pair direct
A.28 sixteen T-states
A.29 direct
A.30 five
A.31 Load accumulator indirect
A.32 seven
A.33 register indirect
A.34 two
A.35 Store accumulator indirect
A.36 seven
A.37 register indirect
A.38 two
A.39 Exchange the contents of H-L pair with D-E pair
A.40 four
A.41 register
A.42 one
A.43 Add register to accumulator
A.44 four
A.45 register
A.46 two
A.47 Add register with carry to accumulator
A.48 four
A.49 register
A.50 one
A.51 Add immediate data to accumulator
A.52 seven T-states
A.53 immediate
A.54 two
A.55 Add register pair to HL pair
A.56 ten
A.57 register
A.58 three
A.59 Decimal adjust accumulator
A.60 Increment register pair