DEPARTMENT OF
ELECTRONICS & COMMUNICATION ENGINEERING

LABORATORY MANUAL
FOR
MICROPROCESSORS & INTERFACING
DEVICES
(III B.Tech. - II - Sem.)

BALAJI INSTITUTE OF TECHNOLOGY & SCIENCE
Laknepally, Narsampet, Warangal
LAB: - MICRO PROCESSOR & INTERFACING DEVICES LAB

REGULATION: R 13

List of Experiments:

1. Arithmetic operations (addition, subtraction, multiplication and division)
2. Addition of two BCD numbers
3. Ascending order descending order of an array of numbers
4. Finding largest smallest number in an array of numbers
5. Generation of fibonacci series
6. Hexadecimal to decimal conversion
7. ASCII to decimal conversion
8. Program for sorting an array for 8086
9. Program for searching for a number of characters in an array for 8086
10. Program for String Manipulations for 8086

MASM PROGRAMING

1. Arithmetic operations (addition, subtraction, multiplication and division)
2. Addition of two BCD numbers
3. Ascending order descending order of an array of numbers
4. Finding largest smallest number in an array of numbers
5. Generation of fibonacci series
6. Hexadecimal to decimal conversion
8051 MICRO CONTROLLER

1. Arithmetic operations (addition, subtraction, multiplication and division)
2. Addition of two BCD numbers
3. Ascending order descending order of an array of numbers
4. Finding largest smallest number in an array of numbers
5. Generation of fibonacci series
6. Masking of bits
7. Hexadecimal to decimal conversion

INTERFACING WITH 8086 MICRO PROCESSOR:

1. Stepper Motor Interfacing to 8086
2. Traffic light controller interfacing to 8086
3. Elevator simulator interfacing to 8086
4. Seven-segment display interfacing to 8086
5. Tone generator interfacing to 8086
6. Interfacing ADC and DAC to 8086
7. SRAM and DRAM interfacing to 8086
8. Digit key-interfacing to 8086.
1. Programs for 16 bit Arithmetic Operations for 8086 (Using Various Addressing Modes)

1.1) AIM: TO WRITE 8086 ALP TO ADD, SUB, MUL, DIV TWO 16-BIT NUMBERS.

**APPARATUS:**

1. 8086 mp kit -1& Adopter-1
2. System-1
3. RS 232 Serial cable.
4. +5v Supply
5. 86/88e Driver Software.

**PROGRAM CODE:**

XOR AX, AX  
MOV ES, AX  
MOV DI, 3000  
XOR BX, BX  
XOR CX, CX  
XOR DX, DX  
MOV AX, 5555  
MOV BX, 2222  
ADD AX, BX  
MOV [DI], AX  
INC DI  
INC DI  
MOV AX, 5555  
SUB AX, BX  
MOV [DI], AX  
INC DI  
INC DI  
MOV AX, 5555  
MUL BX
MOV [DI], AX
INC DI
INC DI
INC DI
INC DI
MOV AX, 5555
XOR DX, DX
DIV BX
MOV [DI], AX
INC DI
INC DI
MOV [DI], DX
INT 03

I/P: AX = 5555   BX = 2222
O/P: ES: DI
0000:3000 77
0000:3001 77 } Add(AX)
0000:3002 33}
0000:3003 33} Sub(AX)
0000:3004 4A}
0000:3005 9F} AX
0000:3006 60}
0000:3007 0B } DX  MUL(AX, DX)

0000:3008 02}
0000:3009 00} AX Quotient}

0000:300A 11}
0000:300B 11} DX Reminder } DIV
Flow chart:

1. Start
2. Get the first number
3. Get the second number
4. ADD the two numbers
5. Is carry flag set?
   - Carry=0: Store result and carry in memory
   - Carry=1: Continue to next step
6. Stop
1.2 MULTIPICATION OF TWO 16-BIT DATA

**AIM:** To multiply two 16-bit data (Multibyte multiplication) using 8086 microprocessor.

**APPARATUS:**

1. 8086 Trainer kit
2. Key board
3. SMPS

**PROGRAM CODE:**

```
MOV AX,[0300]
MOV BX,[0302]
MUL BX
INT A5
```

**RESULT:**

Input Data (Before Execution)

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:0300</td>
<td>44</td>
</tr>
<tr>
<td>0000:0301</td>
<td>44</td>
</tr>
<tr>
<td>0000:0302</td>
<td>11</td>
</tr>
<tr>
<td>0000:0303</td>
<td>11</td>
</tr>
</tbody>
</table>

Output Data (After Execution)

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>44</td>
</tr>
<tr>
<td>AH</td>
<td>44</td>
</tr>
<tr>
<td>DL</td>
<td>00</td>
</tr>
<tr>
<td>DH</td>
<td>00</td>
</tr>
</tbody>
</table>
1.3. DIVISION OF TWO 16-BIT DATA

**AIM:** To multiply two 16-bit data (Multibyte division) using 8086 microprocessor.

**APPARATUS:**

1. 8086 Trainer kit
2. Key board
3. SMPS

**PROGRAM CODE:**

```
MOV AX,[0300]
MOV BX,[0302]
DIV BX
INT A5
```

**RESULT:**

Input Data (Before Execution)

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:0300</td>
<td>44</td>
</tr>
<tr>
<td>0000:0301</td>
<td>44</td>
</tr>
<tr>
<td>0000:0302</td>
<td>22</td>
</tr>
<tr>
<td>0000:0303</td>
<td>22</td>
</tr>
</tbody>
</table>

Output Data (After Execution)

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>22</td>
</tr>
<tr>
<td>AH</td>
<td>22</td>
</tr>
<tr>
<td>DL</td>
<td>00</td>
</tr>
<tr>
<td>DH</td>
<td>00</td>
</tr>
</tbody>
</table>
Program: 1.4  MULTIPLICATION OF SIGNED NUMBERS

AIM: To multiply two signed numbers (8-bit data) using 8086 microprocessors.

APPARATUS:
1. 8086 Trainer kit
2. Key board
3. SMPS

PROGRAM CODE:

MOV AX,0200
MOV DS,AX
MOV AL,[1500]
NEG AL
MOV BL,[1501]
MUL BL
MOV [1505],AX
INT A5

RESULT:

Input Data (Before Execution)

2000:1500  10
2000:1501  15

Output Data (After Execution)

2000:1505  B0
2000:1506  13
2. ADDITION OF TWO 16-BIT BCD NUMBERS

AIM: TO WRITE 8086 ALP TO ADD TWO 16-BIT BCD NUMBERS.

APPARATUS:

1. 8086 mp kit -1 & Adopter-1
2. System-1
3. RS 232 Serial cable.
4. +5v Supply
5. 86/88e Driver Software.

PROGRAM

DATA SEGMENT
MESS1 DB 0AH,0DH,'ENTER FIRST NUMBER:','$'
MESS2 DB 0AH,0DH,'ENTER SECOND NUMBER:','$'
MESS3 DB 0AH,0DH,'SUM OF TWO 16-BIT NUMBER IS:','$'
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:MOV AX,DATA
MOV DS,AX
LEA DX,MESS1
MOV AH,09H
INT 21H
CALL READ
MOV BX,DX
LEA DX,MESS2
MOV AH,09H
INT 21H
CALL READ
MOV CL,00H
MOV AL,BL
ADD AL,DL
DAA
MOV BL,AL
MOV AL,BH
ADC AL,DH
DAA
JNC NEXT
INC CL
NEXT:MOV BH,AL
CALL DISP
MOV AH,4CH
INT 21H
READ PROC NEAR
PUBLIC READ
MOV CH,02H
R3:MOV AH,01H
INT 21H
MOV CL,04H
MOV DL,AL
SUB DL,30H
CMP DL,0AH
JC R1
SUB DL,07H
R1:SHL DL,CL
MOV AH,01H
INT 21H
SUB AL,30H
CMP AL,0AH
JC R2
SUB AL,07H
AND AL,0FH
R2:OR DL,AL
DEC CH
JZ R4
MOV DH,DL
JMP R3
R4:RET
READ ENDPDISP PROC NEAR PUBLIC DISP
LEA DX, MESS3
MOV AH, 09H
INT 21H
MOV DL, CL
ADD DL, 30H
MOV AH, 06H
INT 21H
MOV CH, 02H
L3: MOV CL, 04H
MOV DL, BH
SHR DL, CL
CMP DL, 0AH
JC L1
ADD DL, 07H
L1: ADD DL, 30H
MOV AH, 06H
INT 21H
AND BH, 0FH
CMP BH, 0AH
JC L2
ADD BH, 07H
L2: ADD BH, 30H
MOV DL, BH
MOV AH, 06H
INT 21H
DEC CH
JZ L4
MOV BH, BL
JMP L3
L4: RET
DISP END
PCODE END
SEND START
OUTPUT:
ENTER FIRST NUMBER : 6987
ENTER SECOND NUMBER : 3991
SUM OF TWO 16-BIT NUMBER IS : 10978
ENTER FIRST NUMBER : 9999
ENTER SECOND NUMBER : 8888
SUM OF TWO 16-BIT NUMBER IS : 18887
3. ASCENDING ORDER DESCENDING ORDER OF AN ARRAY OF NUMBERS

**AIM:** To write a program to sort a given string of a number in ascending/descending 8086 microprocessor programming.

**APPARATUS:**
1. 8086 Trainer kit
2. Keyboard
3. SMPS

**PROGRAM CODE:**

```
MOV CX, 0005  
DEC CX  
Again
MOV DX, CX  
MOV SI, 0200  
Up
MOV AL, [SI]  
INC SI  
MOV BL, [SI]  
CMP AL, BL  
JLE/JGE Next  
XCHG AL, BL  
MOV [SI], BL  
DEC SI  
MOV [SI], AL  
INC SI  
Next
DEC DX  
JNZ Up  
DEC CX  
JNZ Again  
INT A5
```
RESULT

Input Data (Before Execution)

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:0200</td>
<td>38</td>
</tr>
<tr>
<td>0000:0201</td>
<td>47</td>
</tr>
<tr>
<td>0000:0202</td>
<td>02</td>
</tr>
<tr>
<td>0000:0203</td>
<td>11</td>
</tr>
<tr>
<td>0000:0204</td>
<td>29</td>
</tr>
</tbody>
</table>

Output Data (After Execution)

<table>
<thead>
<tr>
<th>Address</th>
<th>Ascending order</th>
<th>Descending order</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:0200</td>
<td>02</td>
<td>47</td>
</tr>
<tr>
<td>0000:0201</td>
<td>11</td>
<td>38</td>
</tr>
<tr>
<td>0000:0202</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>0000:0203</td>
<td>38</td>
<td>11</td>
</tr>
<tr>
<td>0000:0204</td>
<td>47</td>
<td>02</td>
</tr>
</tbody>
</table>
4. FINDING LARGEST SMALLEST NUMBERS IN AN ARRAY OF NUMBERS

AIM: Write a Program to find smallest number from a given array of numbers.

APPARATUS:
1. 8086 Trainer kit
2. Key board
3. SMPS

PROGRAM CODE:

XOR AX,AX
MOV DS,AX
MOV SI,5000
MOV CL,06
MOV AL,[SI]
L1: INC SI
    MOV BL,[SI]
    CMP AL,BL
    JL 7014(L2)
    XCHG AL,BL
L2: LOOP 700B(L1)
    INT 03

RESULT: The smallest number is given array of number is:

AX=009D     FL=F097
4:2 To find the Greatest number

AIM: Write a Program to find Greatest number from a given array of numbers

APPARATUS:
1. 8086 Trainer kit
2. Keyboard
3. SMPS

PROGRAM CODE:

XOR AX,AX
MOV DS,AX
MOV SI,5000
MOV CL,06
MOV AL,[SI]
L1: INC SI
    MOV BL,[SI]
    CMP AL,BL
    JG 7014(L2)
    XCHG AL,BL
L2: LOOP 700B(L1)
    INT 03

RESULT: The smallest number is given array of number is:

AX=007C  FL=F006
5 GENERATION OF FIBONACCI SERIES

AIM: TO WRITE 8086 ALP TO GENERATE FIBONACCI SERIES.

APPARATUS:

1. 8086 mp kit -1 & Adopter-1
2. System-1
3. RS 232 Serial cable.
4. +5v Supply
5. 86/88e Driver Software.

Assume cs: code
Code segment
Start: XOR AX,AX
XOR BX,BX
XOR CX,CX
XOR DX,DX
MOV CL,09
MOV DS,AX
MOV SI,3000
MOV AL,00
MOV [SI],AL
INC SI
MOV BL,01
MOV [SI],BL
L1: ADD AL,BL
MOV DL,AL
INC SI
MOV [SI],DL
MOV AL,BL
MOV BL,DL
LOOP L1
INT 03
Code ends
End start
6. HEXADECIMAL TO DECIMAL CONVERSION

AIM: TO WRITE 8086 ALP TO CONVERT HEXADECIMAL TO DECIMAL

APPARATUS:
1. 8086 mp kit -1 & Adopter-1
2. System-1
3. RS 232 Serial cable.
4. +5v Supply
5. 86/88e Driver Software.

PROGRAM:
DATA SEGMENT
   NUM DW 1234H
   RES DB 10 DUP ('$')
DATA ENDS
START:
   MOV AX, DATA
   MOV DS, AX
   MOV AX, NUM
   LEA SI, RES
   CALL HEX2DEC
   LEA DX, RES
   MOV AH, 9
   INT 21H
   MOV AH, 4CH
   INT 21H
CODE ENDS
HEX2DEC PROC NEAR
   MOV CX, 0
   MOV BX, 10
HEX2DEC ENDP
LOOP1: MOV DX,0
    DIV BX
    ADD DL,30H
    PUSH DX
    INC CX
    CMP AX,9
    JG LOOP1

    ADD AL,30H
    MOV [SI],AL

LOOP2: POP AX
    INC SI
    MOV [SI],AL
    LOOP LOOP2
    RET
HEX2DEC ENDP

END START
7. ASCII TO DECIMAL

AIM: TO WRITE 8086 ALP TO CONVERT ASCII TO DECIMAL

APPARATUS:

1. 8086 mp kit - 1 & Adopter - 1
2. System - 1
3. RS 232 Serial cable.
4. +5v Supply
5. 86/88e Driver Software.

PROGRAM:

    MOV AX, DATA
    MOV DS, AX

    MOV AH, 0
    MOV AL, NUM

    LEA SI, RES
    CALL HEX2DEC

    LEA DX, RES
    MOV AH, 9
    INT 21H

    MOV AH, 4CH
    INT 21H

CODE ENDS
HEX2DEC PROC NEAR
    MOV CX, 0
    MOV BX, 10

LOOP1: MOV DX, 0
    DIV BX
ADD DL,30H
PUSH DX
INC CX
CMP AX,9
JG LOOP1

ADD AL,30H
MOV [SI],AL

LOOP2: POP AX
INC SI
MOV [SI],AL
LOOP LOOP2
RET
HEX2DEC ENDP
8. PROGRAM FOR SORTING AN ARRAY FOR 8086

**AIM:** To find the sum of squares of data string by using 8086 assembly language program.

**APPARATUS:**

1. 8086 Trainer kit
2. Key board
3. SMPS

**PROGRAM CODE:**

```assembly
MOV SI,[0300]
MOV CL,[SI]
MOV DI,0220
MOV BX,0000
MOV AH,00
INC SI
MOV AL,[SI]
MUL AL
ADD [DI],AX
DEC CL
JNZ Up
INT A5
```

**RESULT**

Input Data

```
0000:0300  05
0000:0301  01
0000:0302  02
0000:0303  03
0000:0304  04
0000:0305  05
```

Output Data
8.2 SUM OF CUBES OF ‘N’ NUMBERS

**AIM:** To find the sum of cubes of an array of size 10 by using 8086 assembly language program.

**APPARATUS**

1. 8086 Trainer kit
2. Key board
3. SMPS

**PROGRAM CODE:**

```
MOV SI,0200
MOV DI,0220
MO CL,0A
MOV AX,0000
MOV [DI],AX
MOV AL,[SI]
MOV BL,AL
MUL AL
MUL BL
ADD [DI],AX
```

**RESULT**

<table>
<thead>
<tr>
<th>Input Data</th>
<th>Output Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:0200</td>
<td>01</td>
</tr>
<tr>
<td>0000:0201</td>
<td>02</td>
</tr>
<tr>
<td>0000:0202</td>
<td>03</td>
</tr>
<tr>
<td>0000:0203</td>
<td>04</td>
</tr>
<tr>
<td>0000:0204</td>
<td>05</td>
</tr>
<tr>
<td>0000:0205</td>
<td>06</td>
</tr>
<tr>
<td>0000:0206</td>
<td>07</td>
</tr>
<tr>
<td>0000:0207</td>
<td>08</td>
</tr>
<tr>
<td>0000:0208</td>
<td>09</td>
</tr>
<tr>
<td>0000:0209</td>
<td>0A</td>
</tr>
<tr>
<td>0220</td>
<td>D1</td>
</tr>
<tr>
<td>0221</td>
<td>0b</td>
</tr>
</tbody>
</table>
9. Program for Searching A for A Number in a Sting for 8086.

**AIM:** PAM for searching a number or a character in a string.

**APPARATUS:**
1. 8086 Trainer kit
2. Keyboard
3. SMPS

**PROGRAM CODE:**
```
XORW AX, AX
XORW BX, BX
MOVW DS, AX
MOVW SI, 6000
MOVW CX, 0007
MOVB AL,09
MOVB BL,[SI]
CMP AL, BL
JZ : L1
INCW SI
LOOP :L2
MOV DX, SI
INT 03
```

**RESULT:**

Input Data (Before Execution)

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:6000</td>
<td>26</td>
</tr>
<tr>
<td>0000:6001</td>
<td>86</td>
</tr>
<tr>
<td>0000:6002</td>
<td>95</td>
</tr>
<tr>
<td>0000:6003</td>
<td>09</td>
</tr>
<tr>
<td>0000:6004</td>
<td>48</td>
</tr>
<tr>
<td>0000:6005</td>
<td>43</td>
</tr>
<tr>
<td>0000:6006</td>
<td>91</td>
</tr>
</tbody>
</table>
Output Data (After Execution): register Contents:
AX=0009  BX=0009  CX=0004  DX=0000  SP=0100  SI=6003
FL=F046

RESULT: A PROGRAM FOR SEARCHING OF AN NUMBER (OR) CHARACTER IN A STRING HAS BEEN PERFORMED.
9.2 TO SEARCH A STRING USING SCAS-SCAN THE STRING.

AIM: WAP to find the required string (or) number from the given string using SCAS Required element is EE and mentions the address in the register ‘DX’.

APPARATUS:
1. 8086 Trainer kit
2. Keyboard
3. SMPS

PROGRAM CODE:
XOR DX, DX
MOV AX, AX
MOV ES, AX
MOV DI, 6000
MOV CL, 07
MOV AL, 0EE
REPNZ
SCASB
DEC DI
MOV DX, DI
INT 03

RESULT:
Input Data (Before Execution)

<table>
<thead>
<tr>
<th>Address:0000-0000</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:6000</td>
<td>55</td>
</tr>
<tr>
<td>0000:6001</td>
<td>AB</td>
</tr>
<tr>
<td>0000:6002</td>
<td>D5</td>
</tr>
<tr>
<td>0000:6003</td>
<td>FD</td>
</tr>
<tr>
<td>0000:6004</td>
<td>9L</td>
</tr>
<tr>
<td>0000:6005</td>
<td>EE</td>
</tr>
<tr>
<td>0000:6006</td>
<td>-34</td>
</tr>
<tr>
<td>0000:6007</td>
<td>-66</td>
</tr>
</tbody>
</table>

Here DX IS Not equal to zero and the required element is present in the register Dx having the Address 6005.
10. PROGRAM FOR STRING MANIPULATIONS FOR 8086 COMPARISION OF TWO STRINGS.

**AIM:** To compare two data stings using 8086 microprocessor programming

**APPARATUS:**
1. 8086 Trainer kit
2. Key board
3. SMPS

**PROGRAM CODE:**

```
MOV AX,2000
MOV DS,AX
MOV ES,AX
MOV SI,0100
MOV DI,0200
MOV CX,0005
MOV BX,0000
CLD
REP CMPSB
JE Last
MOV BX,FFFF
INT A5
```

**RESULT**

Input Data (Before Execution)

<table>
<thead>
<tr>
<th>DS:SI</th>
<th>ES:DI</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000:0100 01</td>
<td>2000:0200 01</td>
</tr>
<tr>
<td>2000:0101 02</td>
<td>2000:0201 02</td>
</tr>
<tr>
<td>2000:0102 03</td>
<td>2000:0202 03</td>
</tr>
<tr>
<td>2000:0103 04</td>
<td>2000:0203 04</td>
</tr>
<tr>
<td>2000:0104 05</td>
<td>2000:0204 05</td>
</tr>
</tbody>
</table>

Output Data (After Execution)

```
BX FFFF
```
10.2 LENGTH OF A DATA STRING

**AIM:** To find the length of a given string of data using 8086 microprocessor programming.

**APPARATUS:**
1. 8086 Trainer kit
2. Keyboard
3. SMPS

**PROGRAM CODE:**

```
MOV AX,2000
MOV ES,AX
MOV DI,0100
MOV CX,0000
MOV AL,00
CLD
Again SCASB
JZ Last
INC CX
JMP Again
Last INT A5
```

**RESULT**

**Input Data (Before Execution)**

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000:0100</td>
<td>44</td>
</tr>
<tr>
<td>2000:0101</td>
<td>67</td>
</tr>
<tr>
<td>2000:0102</td>
<td>49</td>
</tr>
<tr>
<td>2000:0103</td>
<td>20</td>
</tr>
<tr>
<td>2000:0104</td>
<td>00</td>
</tr>
</tbody>
</table>

**Output Data (After Execution)**

<table>
<thead>
<tr>
<th>CX</th>
</tr>
</thead>
<tbody>
<tr>
<td>0004</td>
</tr>
</tbody>
</table>
10.3 MOVING A STRING OF DATA

**AIM:** To move a sting or a block of data from one segment to the other segment using 8086 microprocessor programming.

**APPARATUS:**
1. 8086 Trainer kit
2. Key board
3. SMPS

**PROGRAM CODE:**

```
MOV AX,2100
MOV DS,AX
MOV AX,2200
MOV ES,AX
MOV SI,2000
MOV DI,0000
MOV CX,0005
CLD
REP MOVSB
INT A5
```

**RESULT**

Input Data (Before Execution)

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000:2000</td>
<td>23</td>
</tr>
<tr>
<td>2000:2001</td>
<td>45</td>
</tr>
<tr>
<td>2000:2002</td>
<td>21</td>
</tr>
<tr>
<td>2000:2003</td>
<td>78</td>
</tr>
<tr>
<td>2000:2004</td>
<td>69</td>
</tr>
</tbody>
</table>

Output Data (After Execution)

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>2200:0000</td>
<td>23</td>
</tr>
<tr>
<td>2000:0001</td>
<td>45</td>
</tr>
<tr>
<td>2000:0002</td>
<td>21</td>
</tr>
<tr>
<td>2000:0003</td>
<td>78</td>
</tr>
<tr>
<td>2000:0004</td>
<td>69</td>
</tr>
</tbody>
</table>
10.4 REVERSE OF A STRING

**AIM:** To write a program to reverse of a string 8086 microprocessor programming.

**APPARATUS:**
1. 8086 Trainer kit
2. Keyboard
3. SMPS

**PROGRAM CODE:**

```
MOV AX,2000
MOV DS,AX
MOV CX,0005
MOV SI,0200
MOV DI,0209

Next  MOV AL,[SI]
       XCHG AL,[DI]
       MOV [SI],AL
       INC SI
       DEC DI
       JNZ  Next
       INT A5
```

**RESULT**

<table>
<thead>
<tr>
<th>Input Data (Before Execution)</th>
<th>Output Data (After Execution)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000:0200 00</td>
<td>2000:0200 09</td>
</tr>
<tr>
<td>2000:0201 01</td>
<td>2000:0201 08</td>
</tr>
<tr>
<td>2000:0202 02</td>
<td>2000:0202 07</td>
</tr>
<tr>
<td>2000:0203 03</td>
<td>2000:0203 06</td>
</tr>
<tr>
<td>2000:0204 04</td>
<td>2000:0204 05</td>
</tr>
<tr>
<td>2000:0205 05</td>
<td>2000:0205 04</td>
</tr>
<tr>
<td>2000:0206 06</td>
<td>2000:0206 03</td>
</tr>
<tr>
<td>2000:0207 07</td>
<td>2000:0207 02</td>
</tr>
<tr>
<td>2000:0208 08</td>
<td>2000:0208 01</td>
</tr>
<tr>
<td>2000:0209 09</td>
<td>2000:0209 00</td>
</tr>
</tbody>
</table>
MASM PROGRAMMING

Introduction to MASM /TASM

MASM: (Microsoft assembler)

To Create Source File: An editor is a program which allows you to create a file containing the assembly language statements for your program. This file is called a source file.

Command to create a source file

C:\MASM\BIN> Edit filename. Asm

The next step is to process the source file with an assembler. When you run the assembler, it reads the source file of your program. On the first pass through the source program, the assembler determines the displacement of named data items, the offset labels, etc. and puts this information in a symbol table. On the second pass through the source program the assembler produces the binary code for each instruction and inserts the offsets, etc. that it calculated during first pass.

C:\MASM\BIN > Masm filename. asm X, Z

With this command assembler generates three files.
1. The first file (X) called the object file, is given the extension .OBJ. The object file contains the binary codes for the instructions and information about the addresses of the instructions.
2. The third file (Z) generated by this assembler is called the cross-reference file and is given the extension .CRF. The cross-reference file lists all labels and pertinent information required for cross – referencing.

NOTE: The Assembler only finds syntax errors : It will not tell you whether program does what it is supposed to do. To determine whether your program works, you have to run the program and test it.

Next step is to process the object file with linker.

C:\MASM\BIN>LINK filename. obj

Run File [Filename1.exe] : “filename1.exe”
List file [nul.map]: NUL
Libraries [.lib]: library name
Definitions File [nul.def]:

**Creation of Library: Refer Modular Programming Section**

A Linker is a program used to join several object files into one layer object file.

**NOTE:** On IBM PC – type Computers, You must run the LINK program on your .OBJ file even if it contains only one assembly module. The linker produces a link file with the .EXE extension (an execution file). Next Run C:\MASM\BIN> filename
1. Programs for 16 bit arithmetic operations for 8086 (using Various Addressing Modes).

a) Addition:
  i) 16 bit addition:

  AIM: - To write an assembly language program for Addition of two 16-bit numbers.

  APPARATUS: 1. 8086 microprocessor kit/MASTM ----1
               2. RPS (+5V) ----1

  PROGRAM:
  i) By using MASM:
  Assume cs: code

  Code segment
  Start: MOV AX, 4343
  MOV BX, 1111
  ADD AX, BX
  INT 3
  Code ends
  End start

  OUTPUT:
  Input
  output
  Register Data
b) **Subtraction:**

i) **16 bit subtraction:**

**AIM:** - To write an assembly language program for subtraction of two 16-bit numbers.

**APPARATUS:**
1. 8086 microprocessor kit/MASM ----1
2. RPS (+5V) ----1

**PROGRAM:**

**k) By using MASM:**

Assume cs: code

Code segment

Start: MOV AX, 4343
MOV BX, 1111
SUB AX, BX
INT 3

Code ends

End start

**OUTPUT:**

Input output    Register Data    Register Data
ii) 16 bit multiplication (signed numbers)

**AIM:** To write an assembly language program for multiplication of two 16-bit signed numbers.

**APPARATUS:**
1. 8086 microprocessor kit/MASM ----1
2. RPS(+5V) ---- 1

**PROGRAM:** By using MASM:

```
Assume cs: code
Code segment
Start: MOV SI, 2000
MOV DI, 3000
MOV AX, [SI]
ADD SI, 02
MOV BX, [SI]
IMUL BX
MOV [DI], AX
ADD DI, 02
MOV [DI], DX
INT 3
Code ends
End start
```

**OUTPUT:**

<table>
<thead>
<tr>
<th>Input MEMORY LOCATION</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>E4(-28)</td>
</tr>
<tr>
<td>2001</td>
<td>E4(-28)</td>
</tr>
<tr>
<td>2002</td>
<td>3B(+59)</td>
</tr>
<tr>
<td>2003</td>
<td>3B(+59)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output MEMORY LOCATION</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>3000</td>
<td>8C</td>
</tr>
<tr>
<td>3001</td>
<td>4C</td>
</tr>
<tr>
<td>3002</td>
<td>F5</td>
</tr>
<tr>
<td>3003</td>
<td>34</td>
</tr>
</tbody>
</table>
d) Division:

I) 16 bit division:

AIM: - To write an assembly language program for multiplication of two 16-bit numbers.

APPARATUS: 1. 8086 microprocessor kit/MASM ----1
2. RPS (+5V) ----1

PROGRAM:

A) By using MASM:
Assume cs: code

Code segment
Start: MOV AX,4343
MOV BX,1111
MUL BX
INT 3
Code ends
End start

OUTPUT:

<table>
<thead>
<tr>
<th>Input Register Data</th>
<th>Output Register Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>AX 4343</td>
<td>AX EA73</td>
</tr>
<tr>
<td>BX 1111</td>
<td>DX 047B</td>
</tr>
</tbody>
</table>

RESULT: 16 bit arithmetical operations are performed by using different addressing modes.
2. ADDITION OF TWO 16-BIT BCD NUMBERS

**AIM:** To write an assembly language program for Addition of two 16-bit Bcd numbers.

**APPARATUS:**
1. 8086 microprocessor kit/MASM ----1
2. RPS (+5V) ----1

**Program:**

```assembly
DATA SEGMENT
MESS1 DB 0AH,0DH,'ENTER FIRST NUMBER:','$'
MESS2 DB 0AH,0DH,'ENTER SECOND NUMBER:','$'
MESS3 DB 0AH,0DH,'SUM OF TWO 16-BIT NUMBER IS:','$'
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:MOV AX,DATA
MOV DS,AX
LEA DX,MESS1
MOV AH,09H
INT 21H
CALL READ
MOV BX,DX
LEA DX,MESS2
MOV AH,09H
INT 21H
CALL READ
MOV CL,00H
MOV AL,BL
ADD AL,DL
DAA
MOV BL,AL
MOV AL,BH
ADC AL,DH
DAA
```

JNC NEXT
INC CL
NEXT:MOV BH,AL
CALL DISP
MOV AH,4CH
INT 21H
READ PROC NEARPUBLIC READ
MOV CH,02H
R3:MOV AH,01H
INT 21H
MOV CL,04H
MOV DL,AL
SUB DL,30H
CMP DL,0AH
JCR1
SUB DL,07H
R1:SHL DL,CL
MOV AH,01H
INT 21H
SUB AL,30H
CMP AL,0AH
JCR2
SUB AL,07H
AND AL,0FH
R2:OR DL,AL
DEC CH
JZ R4
MOV DH,DL
JMP R3
R4:RET
READ ENDP
DISP PROC NEARPUBLIC DISP
LEA DX,MESS3
MOV AH,09H
INT 21H
MOV DL, CL
ADD DL, 30H
MOV AH, 06H
INT 21H
MOV CH, 02H
L3: MOV CL, 04H
MOV DL, BH
SHR DL, CL
CMP DL, 0AH
JC L1
ADD DL, 07H
L1: ADD DL, 30H
MOV AH, 06H
INT 21H
AND BH, 0FH
CMP BH, 0AH
JC L2
ADD BH, 07H
L2: ADD BH, 30H
MOV DL, BH
MOV AH, 06H
INT 21H
DEC CH
JZ L4
MOV BH, BL
JMP L3L4: RET
DISP END
PCODE END
SEND START
OUTPUT:
ENTER FIRST NUMBER : 6987
ENTER SECOND NUMBER : 3991
SUM OF TWO 16-BIT NUMBER IS : 10978
ENTER FIRST NUMBER : 9999
ENTER SECOND NUMBER : 8888
SUM OF TWO 16-BIT NUMBER IS : 18887
3. ASCENDING ORDER DESCENDING ORDER OF AN ARRAY OF NUMBERS

AIM:- Program to sort the given numbers in ascending order

APPARATUS: 1. 8086 microprocessor kit/MASM ---- 1
2. RPS (+5V) ---- 1

PROGRAM:

A) By using MASM:

ASSUME CS: CODE
CODE SEGMENT
START: MOV AX, 0000H
MOV CH, 0004H
DEC CH
UP1 : MOV CL, CH
MOV SI, 2000
UP: MOV AL, [SI]
INC SI
CMP AL, [SI]
JC DOWN
XCHG AL, [SI]
DEC SI
MOV [SI], AL
INC SI
DOWN: DEC CL
JNZ UP
DEC CH
JNZ UP1
INT 3
CODE ENDS
END START

ii) DESCENDING ORDER

AIM:- Program to sort the given numbers in descending order

APPARATUS: 1. 8086 microprocessor kit/MASM ---- 1
2. RPS (+5V) ---- 1

**PROGRAM:**

A) By using MASM:

ASSUMEx CS: CODE

CODE SEGMENT

START: MOV AX, 0000H

MOV CH, 0004H

DEC CH

UP1 : MOV CL, CH

MOV SI, 2000

UP: MOV AL, [SI]

INC SI

CMP AL, [SI]

JNC DOWN

XCHG AL, [SI]

DEC SI

MOV [SI], AL

INC SI

DOWN: DEC CL

JNZ UP

DEC CH

JNZ UP1

INT 3

CODE ENDS

END START

**RESULT:** Program for sorting an array performed by using masm software and trainer kit.

<table>
<thead>
<tr>
<th>I/p Memory location</th>
<th>Data</th>
<th>O/p Memory location</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>03</td>
<td>2000</td>
<td>07</td>
</tr>
<tr>
<td>2001</td>
<td>06</td>
<td>2001</td>
<td>06</td>
</tr>
<tr>
<td>2002</td>
<td>07</td>
<td>2002</td>
<td>04</td>
</tr>
<tr>
<td>2003</td>
<td>04</td>
<td>2003</td>
<td>03</td>
</tr>
</tbody>
</table>
4. FINDING LARGEST & SMALLEST NUMBER OF AN ARRAY OF NUMBERS

AIM: To write an Assembly Language Program (ALP) to find the largest and Smallest number in a given array.

APPARATUS REQUIRED:

1. Microprocessor kit 8086 1
2. Power Supply +5 V dc 1

PROGRAM :

MOV SI,1200H
MOV CL,[SI]
INC SI
MOV AL,[SI]
DEC CL
L2 INC SI
CMP AL,[SI]
JNB L1 MOV AL,[SI]
L1 DEC CL
JNZ L2
MOV DI,1300H
MOV [DI],AL
HLT

SMALLEST : MOV SI,1200H
MOV CL,[SI]
INC SI
MOV AL,[SI]
DEC CL
L2 INC SI
CMP AL,[SI]
JB L1
MOV AL,[SI]
L1 DEC CL
JNZ L2
MOV DI,1300H
MOV [DI],AL
HLT

RESULT:
Thus largest and smallest number is found in a given array
5 GENERATION OF FIBONACCI SERIES

AIM: - To write an assembly language program for generation of fibonacci series.

APPARATUS:
1. 8086 microprocessor kit/MASTM ---- 1
2. RPS (+5V) ---- 1

Program:
Assume cs: code
Code segment
Start: XOR AX,AX
XOR BX,BX
XOR CX,CX
XOR DX,DX
MOV CL,09
MOV DS,AX
MOV SI,3000
MOV AL,00
MOV [SI],AL
INC SI
MOV BL,01
MOV [SI],BL
L1:ADD AL,BL
MOV DL,AL
INC SI
MOV [SI],DL
MOV AL,BL
MOV BL,DL
LOOP L1
INT 03
Code ends
End start
6. HEXADECIMAL TO DECIMAL CONVERSION

**AIM:** - To write an assembly language program to covert hexadecimal to decimal conversion.

**APPARATUS:**
1. 8086 microprocessor kit/MASM ---- 1
2. RPS (+5V)

**PROGRAM**

DATA SEGMENT
   NUM DW 1234H
   RES DB 10 DUP ('$')
DATA ENDS

START:
   MOV AX,DATA
   MOV DS,AX
   MOV AX,NUM
   LEA SI,RES
   CALL HEX2DEC
   LEA DX,RES
   MOV AH,9
   INT 21H

   MOV AH,4CH
   INT 21H

CODE ENDS

HEX2DEC PROC NEAR
   MOV CX,0
   MOV BX,10
   LOOP1: MOV DX,0
DIV BX
ADD DL,30H
PUSH DX
INC CX
CMP AX,9
JG LOOP1

ADD AL,30H
MOV [SI],AL

LOOP2: POP AX
INC SI
MOV [SI],AL
LOOP LOOP2
RET
HEX2DEC ENDP

END START
PROGRAMMING WITH 8051
1 PROGRAMMING USING ARITHMETIC, LOGICAL AND BIT MANIPULATION INSTRUCTIONS OF 8051

1.1 ADDITION OF TWO NUMBERS

**AIM:** To add two numbers by using 8051 microcontroller.

**APPARATUS:**
1. 8051 Trainer kit
2. Keyboard
3. SMPS

**PROGRAM CODE:**

```
MOV A,#24
MOV F0,#42
ADD A,F0
RET
```

**RESULT**

Input Data

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>24</td>
</tr>
<tr>
<td>B</td>
<td>42</td>
</tr>
</tbody>
</table>

Output Data

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>66</td>
</tr>
</tbody>
</table>
1.2 SUBTRACTION OF TWO NUMBERS

AIM: To subtraction two numbers by using 8051 microcontroller

APPARATUS:

1. 8051 Trainer kit
2. Keyboard
3. SMPS

PROGRAM CODE:

```
MOV A,#44
MOV F0,#37
CLR C
SUB A,F0
RET
```

RESULT

Input Data

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>44</td>
</tr>
<tr>
<td>B</td>
<td>37</td>
</tr>
</tbody>
</table>

Output Data

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0D</td>
</tr>
</tbody>
</table>
1.3 MULTIPLICATION OF TWO NUMBERS

**AIM:** To multiply the given two numbers by using 8051 microcontroller

**APPARATUS:**

1. 8051 Trainer kit
2. Key board
3. SMPS

**PROGRAM CODE:**

```
MOV A,#22
MOV F0,#11
MUL AB
RET
```

**RESULT**

**Input Data**

<table>
<thead>
<tr>
<th>A</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>11</td>
</tr>
</tbody>
</table>

**Output Data**

<table>
<thead>
<tr>
<th>A</th>
<th>42</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>02</td>
</tr>
</tbody>
</table>
1.4 DIVISION OF TWO NUMBERS

AIM: To multiply the given numbers by using 8051 microcontroller

APPARATUS:

1. 8051 Trainer kit
2. Key board
3. SMPS

PROGRAM CODE:

MOV A,#22
MOV F0,#11
DIV A
RET

RESULT

Input Data
A  22
B  11

Output Data
A  02
B  00
1.5 USE OF SWAP INSTRUCTION

AIM: To show the use of SWAP instruction of 8051 microcontroller

APPARATUS:

1. 8051 Trainer kit
2. Keyboard
3. SMPS

PROGRAM CODE:

```
MOV A,#50
SWAP A
MOV R0,A
RET
```

RESULT:

Input Data
   A  50

Output Data
   A  05
2 ADDITION OF TWO BCD NUMBERS

**AIM:** To find the addition of two bcd numbers using 8051 microcontroller.

**APPARATUS:**

1. 8051 Trainer kit
2. Keyboard
3. SMPS

**PROGRAM:**

```
mov r1,#46h
mov r2,#29h
mov psw,#00h
mov a,#99h
subb a,r2
add a,r1
da a
addc a,#00h
mov r0,a
end
```
3 ASCENDING ORDER/DESCENDING ORDER OF AN ARRAY OF NUMBERS

AIM: To arrange the given numbers in ascending order.

APPARATUS:
8051 Trainer kit
Keyboard
SMPS

PROGRAM CODE:

ORG 000H
SJMP START
START : MOV R1,#05H
AGAIN UP : INC R0
    MOV B,@R0
    CLR C
    SUBB A,B
    JC SKIP
    MOV B,@R0
    DEC R0
    MOV A,@R0
    MOV @R0,B
    INC R0
    MOV @R0,A
SKIP : DJNZ R2,UP
    DJNZ R1,AGAIN
STOP : SJMP STOP : MOV A,R1
    MOV R2,A
    MOV R0,#30H
    MOV A,@R0
DESCENDING ORDER

**AIM:** To arrange the given numbers in descending order.

**APPARATUS:**
- 8051 Trainer kit
- Keyboard
- SMPS

**PROGRAM CODE:**

```
MOV R6, #07H
START:   MOV R7, #07H
          MOV R0, #30H
          MOV A, #00H
BACK:    MOV A, @R0
          INC R0
          CJNE A, @R0, CARRY
          SJMP DECREMENTC
CARRY:   JC DECREMENTC
          MOV B, @R0
          MOV @R0, A
          DEC R0
          MOV A, B
          MOV @R0, A
DECREMENTC:  INC R0
            DJNZ R7, BACK
            DJNZ R6, START
END
```
4 FINDIND LARGEST/SMALLEST NUMBERSIN AN ARRAY
SMALLEST OF TWO NUMBERS

AIM: To find the smallest of two numbers using 8051 microcontroller.

APPARATUS:
1. 8051 Trainer kit
2. Key board
3. SMPS

PROGRAM CODE:

    MOV R0,#05
    MOV R1,#08
    MOV A,R0
    CLR C
    SUBB A,R1
    JC Down
    MOV A,R1
    RET

    Down MOV A,R0
    RET

RESULT:

Input Data

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R0</td>
<td>05</td>
</tr>
<tr>
<td>R1</td>
<td>08</td>
</tr>
</tbody>
</table>

Output Data

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>05</td>
</tr>
</tbody>
</table>
LARGEST OF TWO NUMBERS

**AIM:** To find the largest of two numbers using 8051 microcontroller.

**APPARATUS:**
4. 8051 Trainer kit
5. Key board
6. SMPS

**PROGRAM CODE:**

```
MOV R0,#05
MOV R1,#08
MOV A,R0
CLR C
SUBB A,R1
JNC Down
MOV A,R1
RET
Down
MOV A,R0
RET
```

**RESULT:**

Input Data

<table>
<thead>
<tr>
<th>R0</th>
<th>05</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>08</td>
</tr>
</tbody>
</table>

Output Data

| A  | 08 |
4. FIBONACCI SERIES

**AIM:** To write Fibonacci series using 8051 microcontroller.

**APPARATUS:**
1. 8051 Trainer kit
2. Keyboard
3. SMPS

**PROGRAM:**

```
BEGIN: MOV R1,30H
MOV R7,#40H
MOV R7,#00H
INC R7
MOV @R7,#00H
INC R7
MOV @R7,#01H
MOV R5,#42H
DEC R1
DEC R1
DEC R7.
LOOP: MOV A, @R7.
INC R7
ADD A, @R7
MOV @R5,A
```
INC R5

DJNZ R1,LOOP

STOP: SJMP

STOP
6. MASKING OF BITS

**AIM:** To perform MASKING of bits.

**APPARATUS:**
- 8051 Trainer kit
- Keyboard
- SMPS

**PROGRAM CODE:**
- ORG 00h
- MOV A, #35h
- ANL A, #0FH

**RESULT**
- 35H 0011 0101
- 0FH 0000 1111
- 05H 0000 0101
7. HEXADECIMAL TO DECIMAL

**AIM:** To perform conversion of Hexadecimal number to decimal number.

**APPARATUS:**
- 8051 Trainer kit
- Keyboard
- SMPS

**Program:**

```
mov b,#100d
div ab
mov hun,a;

mov a,b;
mov b,#10d;
div ab
mov tens,a;

mov units,a;
ret
```
INTERFACING WITH 8086 MICROPROCESSOR
1. STEPPER MOTOR INTERFACE TO 8086

**AIM:** Write a program to rotate stepper motor in clock wise direction.

**APPARATUS:**

1. 8086 Trainer kit
2. Keyboard
3. SMPS
4. Stepper motor interfacing kit

**PROGRAM CODE:**

```
MOV DX, 8006
MOV AL, 80
OUT DX, AL
MOV CL, 01
MOV DX, 8000
MOV AL, 88
OUT DX, AL
CALL Delay
ROR AL, CL
JMP Up
INT A5
```

**DELAY PROGRAM**

```
Delay MOVCX, FFFF
INT AA
RET
```
2 TRAFFIC LIGHT INTERFACING TO 8086

**AIM:** Write a program for traffic light interfacing.

**APPARATUS:**

1. 8086 Trainer kit
2. Keyboard
3. SMPS

**Program:**

```asm
MODEL SMALL
.STACK 100
.DATA
CWR EQU 0FFC6 H
PORTA EQU 0FFC0 H
PORTB EQU 0FFC2 H
PORTC EQU 0FFC4 H
.CODE
START:
MOV AX,@DATA
MOV DS,AX
MOV AL,80H
MOV DX,CWR
OUT DX,AL
MOV AL,F3H
MOV DX,PORTC
OUT DX,AL
MOV AL,FFH
MOV DX,PORTB
OUT DX,AL
MOV AL,FFH
MOV DX,PORTA
OUT DX,AL
MOV AL,FFH
MOV DX,PORTB
OUT DX,AL
```

Balaji Institute of Technology & Science, Narsampet
OUT DX,AL
MOV CL,03H
CALL DELAY
TOP:
MOV AL,EEH
MOV DX,PORTA
OUT DX,AL
MOV AL,EEH
MOV DX,PORTB
OUT DX,AL
MOV CL,02H
CALL DELAY

MOV AL,FCH
MOV DX,PORTC
OUT DX,AL
MOV AL,7DH
MOV DX,PORTA
OUT DX,AL
MOV AL,57H
MOV DX,PORTB
OUT DX,AL
MOV CL,15H
CALL DELAY

MOV AL,E7H
MOV DX,PORTB
OUT DX,AL
MOV AL,FDH
MOV DX,PORTA
OUT DX,AL
MOV AL,EDH
MOV DX,PORTA
OUT DX,AL
MOV CL,02H
CALL DELAY

MOV AL,F7H
MOV DX,PORTB
OUT DX,AL
MOV AL,F0H
MOV DX,PORTC
OUT DX,AL
MOV AL,F1H
MOV DX,PORTA
OUT DX,AL
MOV CL,15H
CALL DELAY

MOV AL,FBH
MOV DX,PORTA
OUT DX,AL
MOV AL,FBH
MOV DX,PORTB
OUT DX,AL
MOV AL,50H
MOV DX,PORTC
OUT DX,AL
MOV CL,15H
CALL DELAY

MOV AL,FEH
MOV DX,PORTA
OUT DX,AL
MOV AL,FEH
MOV DX,PORTB
OUT DX,AL
MOV CL,03H
CALL DELAY

MOV AL,FFH
MOV DX,PORTA
OUT DX,AL
MOV AL,AFH
MOV DX,PORTC
OUT DX,AL
MOV AL,EEH
MOV DX,PORTA
OUT DX,AL
MOV AL,EEH
MOV DX,PORTB
OUT DX,AL
MOV CL,02H
CALL DELAY

MOV AL,BFH
MOV DX,PORTA
OUT DX,AL
MOV AL,BFH
MOV DX,PORTB
OUT DX,AL
MOV CL,15H
CALL DELAY
JMP TOP

DELAY:
MOV BX,10H
D1:
MOV CX,0FFFFH
D2:
LOOP D2
DEC BX
PROCEDURE:

1. Connect power supply 5V & GND to both microprocessor trainer kit & Traffic light controller interfacing kit.

2. Connect data bus between microprocessor trainer kit & Traffic light controller interfacing kit.

3. Enter the program to control Traffic light.

4. Execute the program by typing GO E000:0B80 ENTER.

5. Observe the LED’s on traffic light controller PCB.
3 ELEVATOR CONTROLLER

**AIM:** Write a program for elevator controller.

**APPARATUS:**

1. 8086 Trainer kit
2. Keyboard
3. SMPS
4. Elevator controller interfacing kit.

**Program:**

```assembly
CODE SEGMENT:0000H
FCODE EQU   2100H
FCLR EQU   2104H

MOV DX,0FFE6
MOV AL,82
OUT DX,AL
XOR AX,AX
LOOP1: MOV AL,AH
OR AL,0F0
MOV DX,0FFE0
OUT DX,AL
MOV DX,0FFE2
LOOP2: IN AL,DX
AND AL,0F
CMP AL,0F
JZ 2013
MOV SI,00
FINDF: ROR AL,1
JNC 2024
INC SI
JMP 201D
```
FOUND: MOV AL, [SI]2100
CMP AL, AH
JA 2038
JB 204F
CLEAR: MOV AL, [SI]2104
MOV DX, 0FFE0
OUT DX, AL
JMP 2008
Goup: CALL 2066
INC AH
XCHG AL, AH
OR AL, 0F0
MOV DX, 0FFE0
OUT DX, AL
AND AL, 0F
XCHG AH, AL
CMP AL, AH
JNZ 2038
JMP 202E
GODN: CALL 2066
DEC AH
XCHG AH, AL
OR AL, 0F0
MOV DX, 0FFE0
OUT DX, AL
AND AL, 0F
XCHG AL, AH
CMP AL, AH
JNZ 204F
JMP 202E
Delay: MOV CX, 0800
HR1: LOOP 2069
HR2: LOOP 206B
ORG 2100H
4. SEVEN SEGMENT DISPLAY INTERFACE

**AIM:** Write a program for seven segment display interfacing.

**APPARATUS:**

1. 8086 Trainer kit
2. Key board
3. SMPS
4. 7-Segment Display interfacing kit.

**Program:**

DATA SEGMENT
PORTA EQU 120H
PORTB EQU 121H
PORTC EQU 122H
CWRD EQU 123H
TABLE DB 8CH,0C7H,86H,89H DATA
ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START: MOV AX,DATA
MOV DS,AX
MOV AL,80H
MOV DX,CWRD
OUT DX,AL
MOV BH,04
LEA SI,TABLE
NEXTDIGIT:MOV CL,08
MOV AL,[SI]
NEXTBIT: ROL AL,01
MOV CH,AL
MOV DX,PORTB
OUT DX,AL
MOV AL,01
MOV DX,PORTC
OUT DX,AL
DEC AL
MOV DX,PORTC
OUT DX,AL
MOV AL,CH
DEC CL
JNZ NEXTBIT
DEC BH
INC SI
JNZ NEXTDIGIT
MOV AH,4CH
INT 21H
CODE ENDS
END
START
5 TONE GENERATOR

AIM: Write a program for tone Generator..

APPARATUS:

1. 8086 Trainer kit
2. Key board
3. SMPS
4. Tone Generator interfacing kit.

Program:

MOV AX,0000
MOV ES,AX
MOV DX,0FFE6
MOV AL,80
OUT DX,AL
GETKEY: CALLS 0FE00
MOV BX,0000
MOV SI,2500
CMP AL,41
JB 200B
CMP AL,50
JG 200B
MOV DX,0FFE4
SUB AL,41
MOV BL,AL
MOV CL,4F
FREQ: MOV AL,00
MOV DX,0FFE4
OUT DX,AL
MOV CH,[BX][SI]
NXTPL: NOP
NOP
NOP
NOP
DEC CH
JNZ 202F
MOV AL,0FF
OUT DX,AL
MOV CH,[BX][SI]
NXTPH: NOP
NOP
NOP
NOP
DEC CH
JNZ 203C
DEC CL
JNZ 2027
JMP 200B
6. INTERFACING ADC AND DAC TO 8086

**AIM:** To interface analog to digital converter with 8086 microprocessor through 8255 and display the digital equivalent of the analog input voltage.

**APPARATUS:**
1. 8086 Trainer kit
2. Key board
3. SMPS
4. CRO
5. Interfacing cable with probe

**PROGRAM CODE:**

```assembly
MOV DX,8807
MOV AL,81
OUT DX,AL
MOV DX,8803
MOV AL,00
OUT DX,AL
MOV DX,8807
MOV AL,09
OUT DX,AL
MOV AL,08
OUT DX,AL
MOV DX,8805
UP
IN AL,DX
```

Repeat
```assembly
MOV DX,8807
MOV AL,0D
OUT DX,AL
MOV AL,0C
OUT DX,AL
MOV DX,8805
UP
IN AL,DX
```
AND AL,02
JNZ        Up
Again
IN AL,DX
AND AL,02
JZ        Again
MOV AL,0B
MOV DX,8807
OUT DX,AL
MOV DX,8803
IN AL,DX
MOV CL,AL
MOV DX,8807
MOV AL,0A
OUT DX,AL
INT AB
MOV AL,02
MOV DX,CX
NOP
MOV DH,00
INT AE
MOV AH,0B
INT A1
AND AL,FF
JZ        Repeat
INT A3

RESULT: 8255 PPI is interfaced with 8086 in mode 0 with port A, B, and C as output ports.

<table>
<thead>
<tr>
<th>8255 address</th>
<th>PORT A</th>
<th>8801</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PORT B</td>
<td>8803</td>
</tr>
<tr>
<td></td>
<td>PORT C</td>
<td>8805</td>
</tr>
<tr>
<td></td>
<td>CWR</td>
<td>8807</td>
</tr>
</tbody>
</table>

Output:
8. DIGIT KEY INTERFACING

**AIM:** Write a program for digital key interfacing.

**APPARATUS:**

1. 8086 Trainer kit
2. Key board
3. SMPS

**Program:**

```
DATA SEGMENT
PORTA EQU 120H
PORTC EQU 122H
CWRD EQU 123H
ARRAY DB '0123456789.+-*/%ACK=MMMM'
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE, DS: DATA
START: MOV AX, DATA
MOV DS, AX ; initialise data segment
MOV AL, 90H ; initialise 8255 porta as i/p and portc as o/p
MOV DX, CWRD
OUT DX, AL
REPEAT: MOV DX, PORTC ; make first row of the keyboard high through pc0
MOV AL, 01
OUT DX, AL
MOV DX, PORTA
IN AL, DX ; input contents of porta and check if key is pressed-
CMP AL, 00 ; in first row.
JZ NEXT
JMP FIRSTROW
NEXT: MOV DX, PORTC ; if key not found in first row, check if key is in
; second row
```
MOV AL,02
OUT DX,AL
MOV DX,PORTA IN
AL,DX
CMP AL,00
JNZ SECONDOVER
MOV AL,04 ; if key not found then check for key closure in
; third row
MOV DX,PORTC
OUT DX,AL
MOV DX,PORTA IN
AL,DX
CMP AL,00H
JNZ THIRDOVER
JMP REPEAT
FIRSTROW: CALL DELAY ; check all the keys one by one in first row
LEA SI,ARRAY
-30-
UP: SHR AL,1
JC DISPLAY ; if key found jump to the display subroutine
INC SI
JMP UP
JMP DISPLAY
SECONDOVER: CALL DELAY
LEA SI,ARRAY+08H ; second row keys from array +08
UP1: SHR AL,1
JC DISPLAY ; if key found jump to the display subroutine
INC SI
JMP UP1
THIRDOVER: CALL DELAY
LEA SI,ARRAY+10H ; third row keys from array +16(dec)
UP2: SHR AL,1
JC DISPLAY ; if key found jump to the display subroutine
INC SI
JMP UP2
JMP DISPLAY
DISPLAY: MOV DL,[SI]
CMP DL,97 ;24 in decimal. 8x3rows = 24keys
JZ EXIT
MOV AH,02H ; display key no in ascii
INT 21H
JMP REPEAT
DELAY: MOV BX,0FFFFH
L1: MOV CX,0FFFH L2:
DEC CX
JNZ L2
DEC BX
JNZ L1
RET
CODE ENDS
END START
EXIT:MOV AH,4CH
INT 21H
-31-