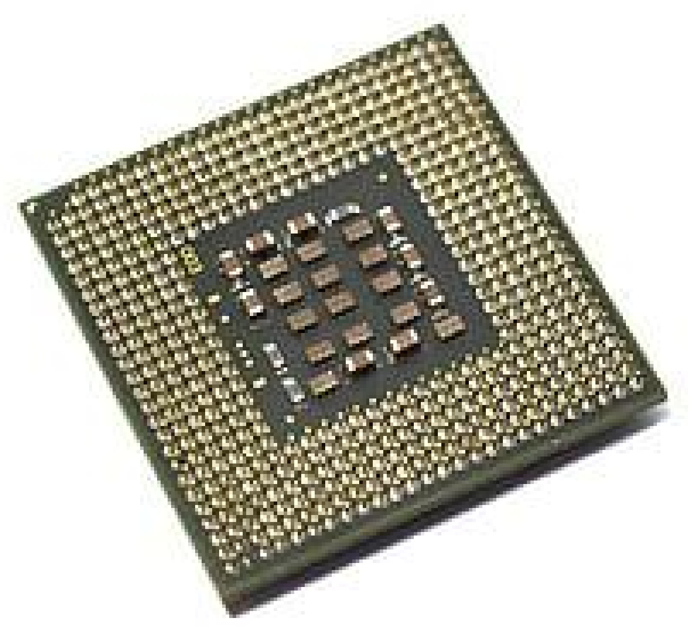
Microprocessor and Microcontroller & Interfacing Techniques Portfolio

ASSIGNMENT-1



KANISHK K U (RA2011004010226)
B.Tech Electronics and Communication Engineering
SRM Institute of Science and Technology

Problems

1. Draw a flowchart and write an 8086 ALP to detect a word is a palindrome or not using string instructions. If palindrome, it should store FFh in location 1200h. Else 00h in the location 1200h (10 marks)

2. Identify the addressing modes of the instructions listed below (5 marks)

   i. TEST [BX][DI], CX

   ii. JMP 1000H:4050H

   iii. AND AX,0007H

   iv. OUT 03H, AL

   v. MUL BX

3. Analyze the below program and express the operation of the program with sample data. (5 marks)

   MOV AX, Data1

   MOV BX, AX

   MUL BX

   MOV [1200], AX

   MOV [1202], DX

   HLT

4. Interface two 16k x 8 EPROMs and two 16k x 8 RAMs chips with 8086. Select suitable address mapping

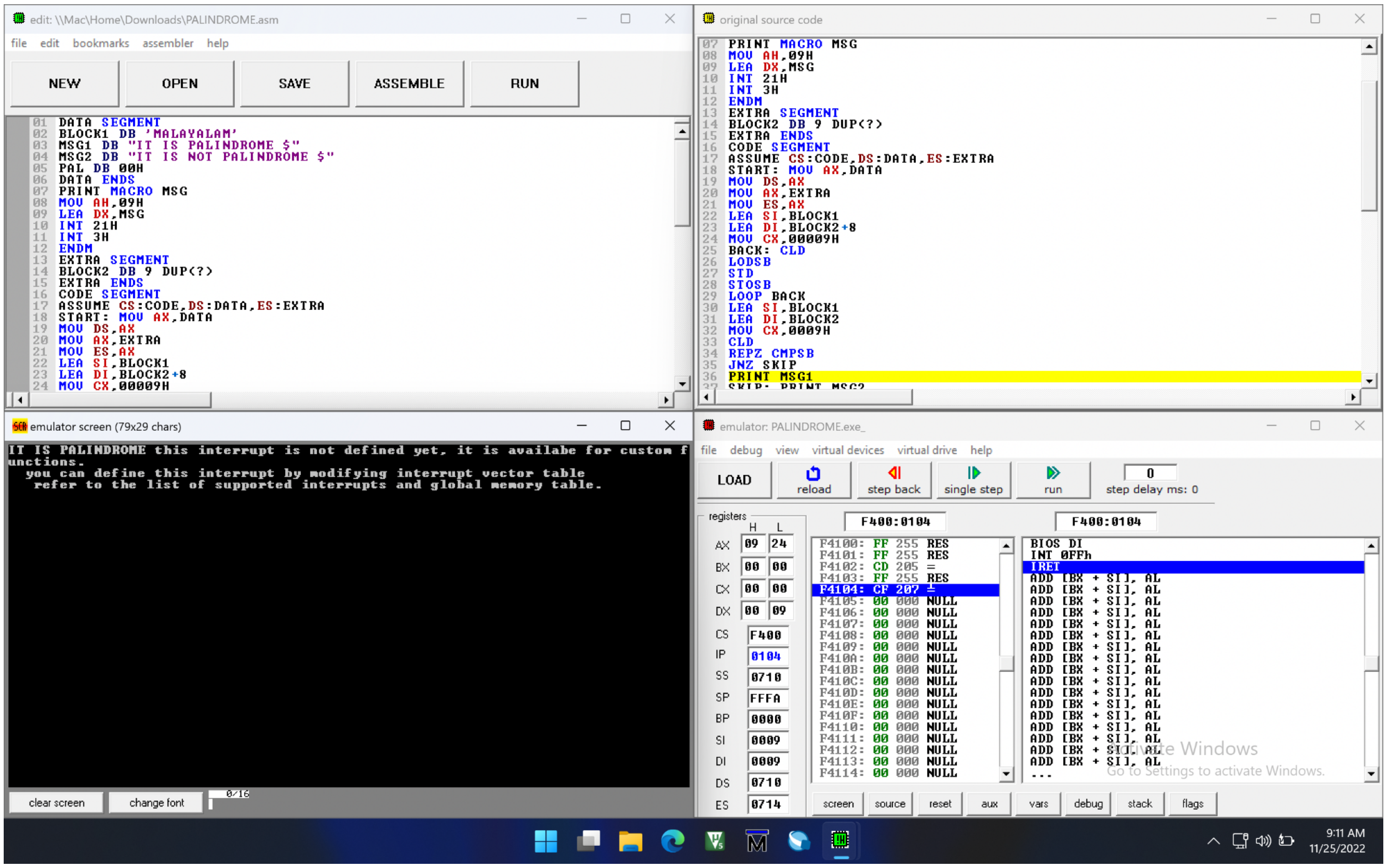
1. Draw a flowchart and write an 8086 ALP to detect a word is a palindrome or not using string instructions. If palindrome, it should store FFh in location 1200h. Else 00h in the location 1200h

Code

```
DATA SEGMENT
BLOCK1 DB 'KANISHK'//'MALAYALAM'
MSG1 DB "IT IS PALINDROME $"
MSG2 DB "IT IS NOT PALINDROME $"
PAL DB 00H
DATA ENDS
PRINT MACRO MSG
MOV AH,09H
LEA DX,MSG
INT 21H
INT 3H
ENDM
EXTRA SEGMENT
BLOCK2 DB 9 DUP(?)
EXTRA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA,ES:EXTRA
START: MOV AX,DATA
MOV DS,AX
MOV AX,EXTRA
MOV ES,AX
LEA SI,BLOCK1
LEA DI,BLOCK2+8
MOV CX,00009H
BACK: CLD
LODSB
STD
STOSB
LOOP BACK
```

```
LEA SI,BLOCK1
LEA DI,BLOCK2
MOV CX,0009H
CLD
REPZ CMPSB
JNZ SKIP
PRINT MSG1
SKIP: PRINT MSG2
CODE ENDS
END START
//RA2011004010226
//KANISHK K U
```

SIMULATION OUTPUT



```
DATA SEGMENT
BLOCK1 DB 'MALAYALAM'
MSG1 DB "IT IS PALINDROME $"
MSG2 DB "IT IS NOT PALINDROME $"
PAL DB 00H
DATA ENDS
PRINT MACRO MSG
MOV AH,09H
LEA DX,MSG
INT 21H
INT 3H
ENDM
EXTRA SEGMENT
BLOCK2 DB 9 DUP(?)
EXTRA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA,ES:EXTRA
START: MOV AX,DATA
MOV DS,AX
MOV AX,EXTRA
MOV ES,AX
LEA SI,BLOCK1
LEA DI,BLOCK2+8
MOV CX,00009H
```

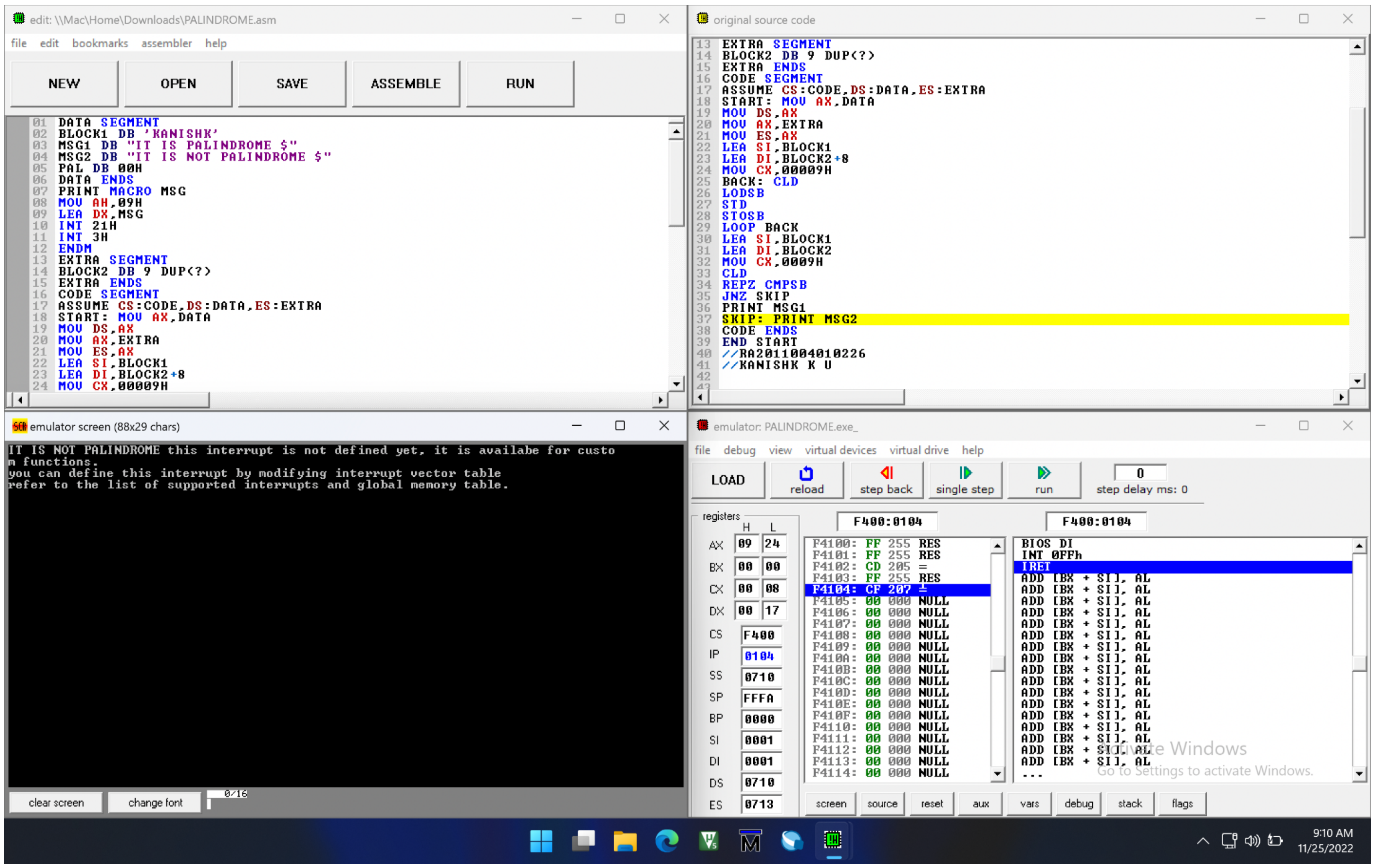
```
PRINT MACRO MSG
MOV AH,09H
LEA DX,MSG
INT 21H
INT 3H
ENDM
EXTRA SEGMENT
BLOCK2 DB 9 DUP(?)
EXTRA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA,ES:EXTRA
START: MOV AX,DATA
MOV DS,AX
MOV AX,EXTRA
MOV ES,AX
LEA SI,BLOCK1
LEA DI,BLOCK2+8
MOV CX,00009H
BACK: CLD
LODSB
STD
STOSB
LOOP BACK
LEA SI,BLOCK1
LEA DI,BLOCK2
MOV CX,0009H
CLD
REPZ CMPSB
JNZ SKIP
PRINT MSG1
```

```
IT IS PALINDROME this interrupt is not defined yet, it is availabe for custom f
unctions.
  you can define this interrupt by modifying interrupt vector table
   refer to the list of supported interrupts and global memory table.
```

```
AX 09 24   CS F400   F400:0104
BX 00 00   IP 0104
CX 00 00   SS 0710
DX 00 09   SP FFFA
           BP 0000
           SI 0009
           DI 0009
           DS 0710
           ES 0714

F4100: FF 255 RES      BIOS DI
F4101: FF 255 RES      INT 0FFh
F4102: CD 205 =        IRET
F4103: FF 255 RES      ADD [BX + SI], AL
F4104: CF 207 ±        ADD [BX + SI], AL
F4105: 00 000 NULL     ADD [BX + SI], AL
F4106: 00 000 NULL     ADD [BX + SI], AL
F4107: 00 000 NULL     ADD [BX + SI], AL
F4108: 00 000 NULL     ADD [BX + SI], AL
F4109: 00 000 NULL     ADD [BX + SI], AL
F410A: 00 000 NULL     ADD [BX + SI], AL
F410B: 00 000 NULL     ADD [BX + SI], AL
F410C: 00 000 NULL     ADD [BX + SI], AL
F410D: 00 000 NULL     ADD [BX + SI], AL
F410E: 00 000 NULL     ADD [BX + SI], AL
F410F: 00 000 NULL     ADD [BX + SI], AL
F4110: 00 000 NULL     ADD [BX + SI], AL
F4111: 00 000 NULL     ADD [BX + SI], AL
F4112: 00 000 NULL     ADD [BX + SI], AL
F4113: 00 000 NULL     ADD [BX + SI], AL
F4114: 00 000 NULL     ...
```

PALINDROME



```
DATA SEGMENT
BLOCK1 DB 'KANISHK'
MSG1 DB "IT IS PALINDROME $"
MSG2 DB "IT IS NOT PALINDROME $"
PAL DB 00H
DATA ENDS
PRINT MACRO MSG
MOV AH,09H
LEA DX,MSG
INT 21H
INT 3H
ENDM
EXTRA SEGMENT
BLOCK2 DB 9 DUP(?)
EXTRA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA,ES:EXTRA
START: MOV AX,DATA
MOV DS,AX
MOV AX,EXTRA
MOV ES,AX
LEA SI,BLOCK1
LEA DI,BLOCK2+8
MOV CX,00009H
```

```
EXTRA SEGMENT
BLOCK2 DB 9 DUP(?)
EXTRA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA,ES:EXTRA
START: MOV AX,DATA
MOV DS,AX
MOV AX,EXTRA
MOV ES,AX
LEA SI,BLOCK1
LEA DI,BLOCK2+8
MOV CX,00009H
BACK: CLD
LODSB
STD
STOSB
LOOP BACK
LEA SI,BLOCK1
LEA DI,BLOCK2
MOV CX,0009H
CLD
REPZ CMPSB
JNZ SKIP
PRINT MSG1
SKIP: PRINT MSG2
CODE ENDS
END START
//RA2011004010226
//KANISHK K U
```

```
IT IS NOT PALINDROME this interrupt is not defined yet, it is availabe for custo
m functions.
you can define this interrupt by modifying interrupt vector table
refer to the list of supported interrupts and global memory table.
```

```
registers
     H   L
AX   09  24
BX   00  00
CX   00  08
DX   00  17
CS   F400
IP   0104
SS   0710
SP   FFFA
BP   0000
SI   0001
DI   0001
DS   0710
ES   0713
```

```
F400:0104
F4100:  FF 255 RES
F4101:  FF 255 RES
F4102:  CD 205 =
F4103:  FF 255 RES
F4104:  CF 207 ≟
F4105:  00 000 NULL
F4106:  00 000 NULL
F4107:  00 000 NULL
F4108:  00 000 NULL
F4109:  00 000 NULL
F410A:  00 000 NULL
F410B:  00 000 NULL
F410C:  00 000 NULL
F410D:  00 000 NULL
F410E:  00 000 NULL
F410F:  00 000 NULL
F4110:  00 000 NULL
F4111:  00 000 NULL
F4112:  00 000 NULL
F4113:  00 000 NULL
F4114:  00 000 NULL
```

```
F400:0104
BIOS DI
INT 0FFh
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

edit: \\Mac\Home\Downloads\PALINDROME.asm
file edit bookmarks assembler help
NEW
OPEN
SAVE
ASSEMBLE
RUN
original source code
emulator screen (88x29 chars)
clear screen
change font
emulator: PALINDROME.exe_
file debug view virtual devices virtual drive help
LOAD
reload
step back
single step
run
step delay ms: 0
screen
source
reset
aux
vars
debug
stack
flags
Activate Windows
Go to Settings to activate Windows.
9:10 AM
11/25/2022

NOT A PALINDROME

2. Identify the addressing modes of the instructions listed below (5 marks)

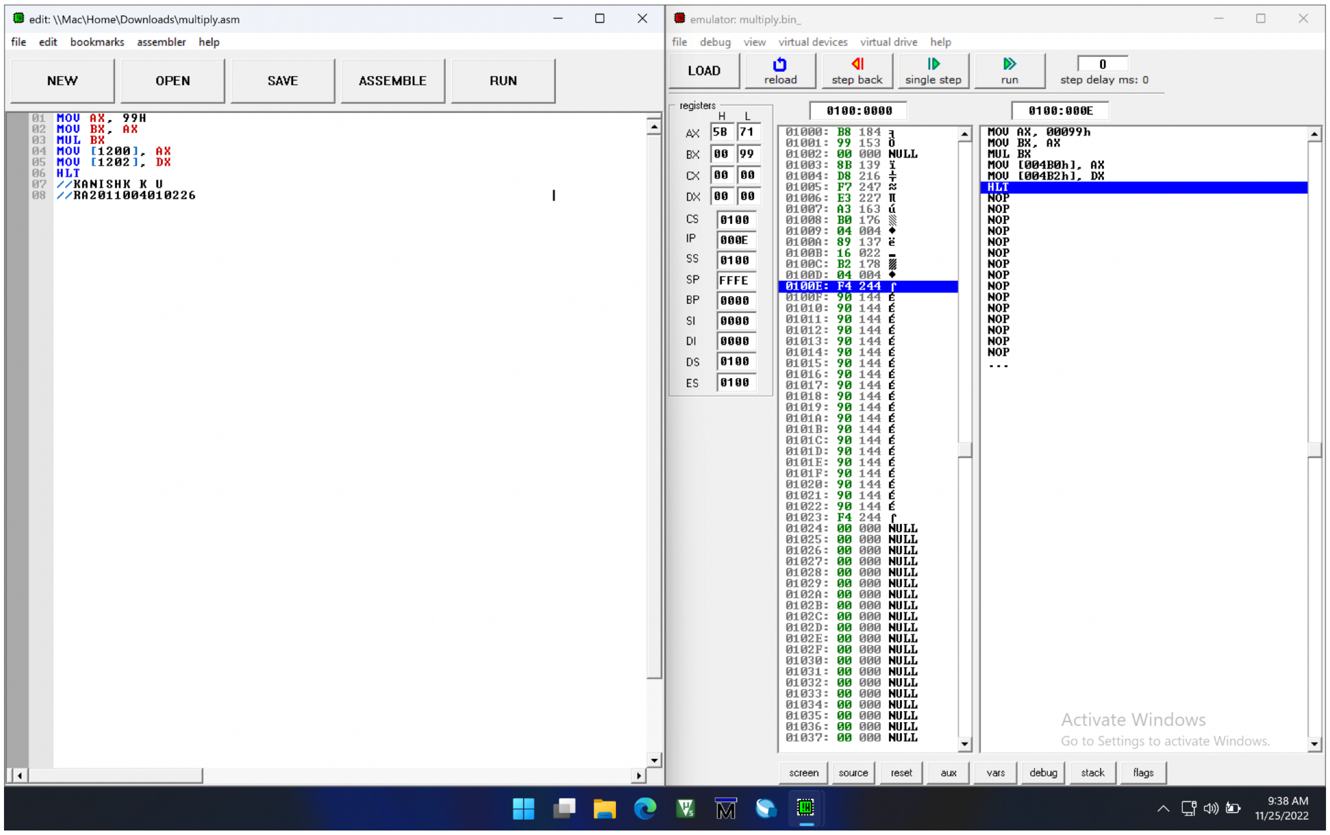
i. TEST [BX][DI], CX	- Based Indexed Addressing Mode
ii. JMP 1000H:4050H	- Direct Addressing Mode
iii. AND AX,0007H	- Immediate Addressing Mode
iv. OUT 03H, AL	- Immediate Addressing Mode
v. MUL BX	- Register Addressing Mode

3. Analyze the below program and express the operation of the program with sample data. (5 marks)

```
MOV AX, Data1
MOV BX, AX
MUL BX
MOV [1200], AX
MOV [1202], DX
HLT
```

Solution:

- Place data in register AX starting at offset 500. (first number)
- Transfer data from register BX to offset 501. (second number)
- Multiply them together (AX=AX*BX).
- Save the result (register AX's content) at offset 600.
- Stop


edit: \\Mac\Home\Downloads\multiply.asm
file edit bookmarks assembler help
NEW OPEN SAVE ASSEMBLE RUN
emulator: multiply.bin_
file debug view virtual devices virtual drive help
LOAD reload step back single step run
step delay ms: 0
screen source reset aux vars debug stack flags
Activate Windows
Go to Settings to activate Windows.
9:38 AM
11/25/2022

```
MOV AX, 99H
MOV BX, AX
MUL BX
MOV [1200], AX
MOV [1202], DX
HLT
//KANISHK K U
//RA2011004010226
```

```
registers
     H   L
AX   5B  71
BX   00  99
CX   00  00
DX   00  00
CS   0100
IP   000E
SS   0100
SP   FFFE
BP   0000
SI   0000
DI   0000
DS   0100
ES   0100
```

```
0100:0000
01000: B8 184 ╕
01001: 99 153 Ö
01002: 00 000 NULL
01003: 8B 139 ï
01004: D8 216 ╪
01005: F7 247 ≈
01006: E3 227 π
01007: A3 163 ú
01008: B0 176 ░
01009: 04 004 ♦
0100A: 89 137 ë
0100B: 16 022 ▬
0100C: B2 178 ▓
0100D: 04 004 ♦
0100E: F4 244 ⌠
0100F: 90 144 É
01010: 90 144 É
01011: 90 144 É
01012: 90 144 É
01013: 90 144 É
01014: 90 144 É
01015: 90 144 É
01016: 90 144 É
01017: 90 144 É
01018: 90 144 É
01019: 90 144 É
0101A: 90 144 É
0101B: 90 144 É
0101C: 90 144 É
0101D: 90 144 É
0101E: 90 144 É
0101F: 90 144 É
01020: 90 144 É
01021: 90 144 É
01022: 90 144 É
01023: F4 244 ⌠
01024: 00 000 NULL
01025: 00 000 NULL
01026: 00 000 NULL
01027: 00 000 NULL
01028: 00 000 NULL
01029: 00 000 NULL
0102A: 00 000 NULL
0102B: 00 000 NULL
0102C: 00 000 NULL
0102D: 00 000 NULL
0102E: 00 000 NULL
0102F: 00 000 NULL
01030: 00 000 NULL
01031: 00 000 NULL
01032: 00 000 NULL
01033: 00 000 NULL
01034: 00 000 NULL
01035: 00 000 NULL
01036: 00 000 NULL
01037: 00 000 NULL
```

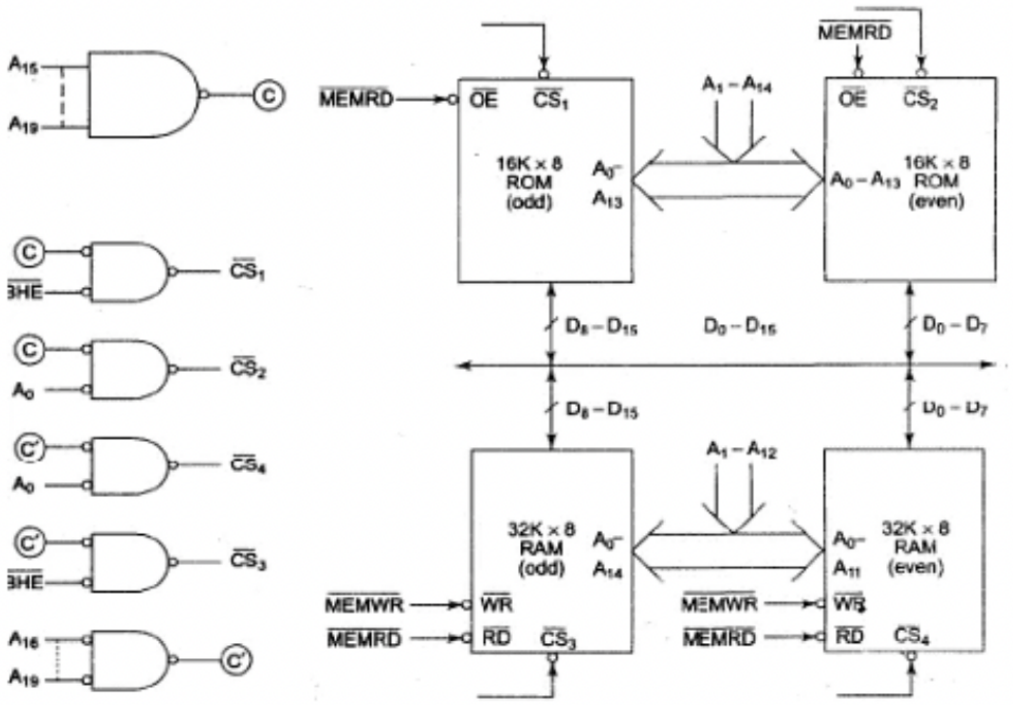
```
0100:000E
MOV AX, 00099h
MOV BX, AX
MUL BX
MOV [004B0h], AX
MOV [004B2h], DX
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
```

4. Interface two 16k x 8 EPROMs and two 16k x 8 RAMs chips with 8086. Select suitable address mapping

Addresses	A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_{09}	A_{08}	A_{07}	A_{06}	A_{05}	A_{04}	A_{03}	A_{02}	A_{01}	A_{00}
FFFFFH	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
						32KB		EPROM												
F8000H	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

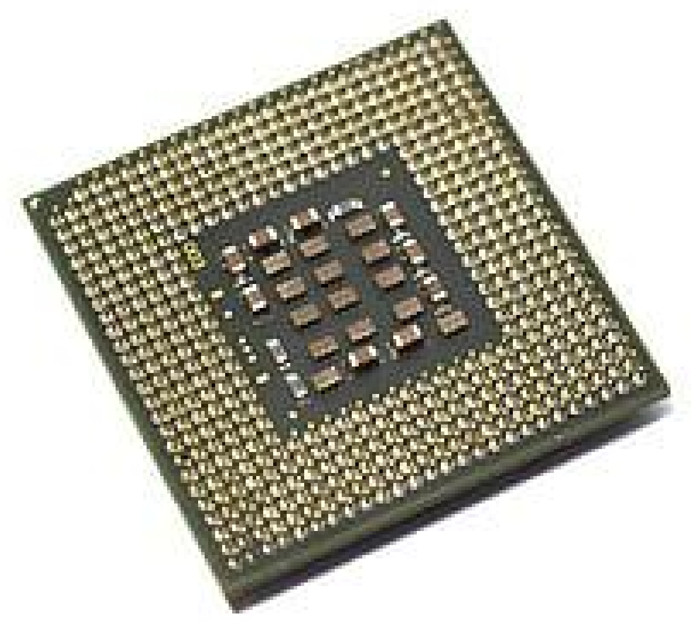
FDFFFH	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
					64KB RAM															
00000H	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Address Map



Microprocessor and Microcontroller & Interfacing Techniques Portfolio

ASSIGNMENT-2



KANISHK K U (RA2011004010226)
B.Tech Electronics and Communication Engineering
SRM Institute of Science and Technology

Problems

1. Write an ALP to copy the value 12H into RAM memory location 50H to 5FH using a) Direct addressing mode b) Register indirect addressing mode without a loop, and c) Register indirect addressing mode with a loop

2. Write an ALP to get the x value from PORT1 and send (x+5)*2 to PORT2, continuously.

3. Design an 8051 based system to display “SRMIST” in 16x2 LCD display.

1. Write an ALP to copy the value 12H into RAM memory location 50H to 5FH using a) Direct addressing mode b) Register indirect addressing mode without a loop, and c) Register indirect addressing mode with a loop

a) Direct addressing mode

```
MOVA, #12h;

MOV 50h, A;

MOV 51h, A ;

MOV 52h, A ;

MOV 53h, A ;

MOV 5fh, A ;
```

b) Register indirect addressing mode without a loop

```
MOVA, #12h;

MOVR0, #50h;

MOV@R0, A;

MOV@R0, A;

MOV@R0, A;

MOV@R0, A;

MOV@R0, A;
```

c) Register indirect addressing mode with a loop

```
MOV A, MOVR0, #50h;

MOV R2, #05;

LOOP: MOV @R0, A;

DJNZ R2, LOOP;
```

2. Write an ALP to get the x value from PORT1 and send (x+5)*2 to PORT2, continuously.

```
ORG 0;
MOV DPTR, #300H;
MOV A, #0FFH;
MOV P1, A;
LOOP: MOV A, P1;
MOVC A, @A+DPTR;
MOV P2, A
SJMP LOOP
ORG 300H
END
```

3. Design an 8051 based system to display “SRMIST” in 16x2 LCD display.

```
MOV A, #38H
ACALL LCD
MOV A, #0EH
ACALL LCD
MOV A, #0LH
ACALL LCD
MOV A, #82H
ACALL LCD
MOV DPTR, #STR
BACK: MOV A, #00H
MOVC A,@A+DPTR
JZ EXIT
ACALL LCD_DATA
INC DPTR
SJMP BACK
EXIT: SJMP EXIT
LCD:
MOV P2,A;
CLR P0.5;
CLR P0.6;
SETR P0.7;
CLR P0.7;
LCD_DATA:
MOV P2,A
SETB P0.5;
CLR P0.6;
SET B P0.7;
CLR P0.7;
RET;
STR:DB ’SRMIST’,0;
```