



## **ELEMENTS OF COMPUTER SYSTEMS 1**

### **END SEMESTER PROJECT**

*Submitted by*

**TEAM 4**

ROLL NUMBER	NAME
AM.EN.U4AIE21008	Akarsh S Nair
AM.EN.U4AIE21011	Alfy Alex
AM.EN.U4AIE21048	Nayan M.K
AM.EN.U4AIE21060	Shyamdev Krishnan J
AM.EN.U4AIE21042	Santhosh Mamidisetti

## **TABLE OF CONTENTS**

<b>Sl No</b>	<b>Topic</b>	<b>Page No</b>
<b>1</b>	<b>Pseudo Code</b>	<b>3</b>
<b>2</b>	<b>Hack Assembly Code</b>	<b>6</b>
<b>3</b>	<b>Output snapshots of CPU Emulator</b>	<b>11</b>
<b>4</b>	<b>Insights learned about Hack Assembly code while implementing the project</b>	<b>13</b>

## Pseudo Code:

### Pseudo code

KEYBOARD:

address = SCREEN + 3840

length = 16

column = 0

fill = 8192

if KBD = ! 0, go to MID

goto RESET

MID: // To draw rectangle at middle

RAM [address] = -1

length = length - 1

if length = 0, go to UPDATE

address = address + 32

go to MID,

UPDATE:

column = column + 1

if column = 32, go to RECT 1

address = SCREEN + column + 3840

length = 16

goto MID

RECT 1 :

// To draw rectangle at top left corner

address = SCREEN

length = 60

column = 0

LOOP 1 :

RAM [address] = -1

length = length - 1

if length = 0, go to ~~loop~~ VLOOP 1

address = address + 32

go to LOOP 1

VLOOP 1 :

column = column + 1

if column = 8, goto RECT 2

address = SCREEN + column

length = 60

go to LOOP 1

RECT 2 :

// To draw rectangle at bottom right corner.

address = SCREEN + 6296

length = 60

column = 0

LOOP 2 :

RAM [address] = -1

length = length - 1

if length = 0 , goto VLOOP 2

address = address + 32

goto LOOP 2

VLOOP 2 :

column = column + 1

if column = 8 , go to KEYBOARD

address = ~~address~~ SCREEN + column + 6296

length = 60

go to LOOP 2

RESET :

address = SCREEN

BLANK :

RAM [address] = 0

address = address + 1

mill = mill - 1

if mill > 0 , goto BLANK

go to KEYBOARD

## **Hack Assembly Code :**



project - Notepad

File Edit Format View Help

(KEYBOARD)

@SCREEN

D = A

@3840

D = D + A

@address

M = D               //address = SCREEN + 3840

@16

D = A

@length

M = D               //length = 16

@column

M = 0               //column = 0

@8192

D = A

@nill

M = D               //nill= 8192

@KBD

D = M

@MID

D; JNE               //if KBD != 0 , go to MID

@RESET

0; JMP               //goto RESET

(MID)

@address

A = M

M = -1               //RAM[address]= -1

@length

MD = M - 1

@UPDATE

D; JEQ               |

@32

D = A

@address

```

M = M + D      // address =address +32

@MID
0; JMP

(UPDATE)
@column
M = M + 1      //column= column + 1

@32
D = A
@column
D = D - M
@RECT1
D; JEQ         //if column = 32 ,go to RECT1

@column
D = M
@SCREEN
D = D + A
@3840
D = D + A
@address
M = D          //address = column + SCREEN + 3840

@16
D = A
@length
M = D          // length=16

@MID
0; JMP

(RECT1)        //To draw rectangle at top rigth corner
@SCREEN
D = A
@address
M = D          //address = screen

@60
D = A
@length

```

```

M = D          // length = 60

@column
M = 0

(LLOOP1)
@address
A = M
M = -1        //RAM[address]=-1

@length
MD = M - 1
@UPLLOOP1
D; JEQ        //if length = 0 , goto UPLLOOP

@32
D = A
@address
M = M + D     //address=address + 32

@LOOP1
0; JMP

(UPLLOOP1)
@column
M = M + 1     //column = column + 32

@8
D = A
@column
D = D - M
@RECT2
D; JEQ        //if column = 8 , goto RECT2

@column
D = M
@SCREEN
D = D + A
@address
M = D         //address= SCREEN + column

```

```

@60
D = A
@length
M = D         //length = 60

@LOOP1
0; JMP

(RECT2)        //To draw rectangle at bottom left corner
@SCREEN
D = A
@6296
D = D + A
@address
M = D         //adress = SCREEN + 6296

@60
D = A
@length
M = D

@column
M = 0

(LLOOP2)
@address
A = M
M = -1        //RAM[address]= -1

@length
MD = M - 1
@UPLLOOP2
D; JEQ        // if length = 0 , goto UPLLOOP2

@32
D = A
@address
M = M + D     //address= address + 32

@LOOP2
0; JMP

```



```

(UPLOOP2)
    @column
    M = M + 1    //column = column + 1

    @8
    D = A
    @column
    D = D - M
    @KEYBOARD
    D; JEQ      //if column = 8 , goto KEYBOARD

    @column
    D = M
    @SCREEN
    D = D + A
    @6296
    D = D + A
    @address
    M = D      //address= SCREEN + column + 6296

    @60
    D = A
    @length
    M = D      //length = 60

    @LOOP2
    0; JMP

(RESET)
    @SCREEN
    D = A
    @address
    M = D      //address = SCREEN

(BLANK)
    @address
    A = M
    M = 0      //RAM[address] = 0

    @address
    M = M + 1  //address= address + 1

    @nill
    MD = M - 1 //nill = nill - 1
    @BLANK
    D; JGT      //if nill > 0 , goto BLANK

    @KEYBOARD
    0; JMP

```

## Output snapshots of CPU Emulator :

When keyboard is pressed:

The screenshot displays a CPU Emulator interface with the following components:

- Menu Bar:** File, View, Run, Help
- Toolbar:** Includes icons for file operations, a slider for animation speed (Slow to Fast), and buttons for 'Animate: No animation', 'View: Scr...', and 'Format: D...'.
- ROM Table:**

Address	Instruction
0	@16384
1	D=A
2	@3840
3	D=D+A
4	@16
5	M=D
6	@16
7	D=A
8	@17
9	M=D
10	@18
11	M=0
12	@8192
13	D=A
14	@19
15	M=D
16	@24576
17	D=M
18	@22
19	D;JNE
20	@147
21	0;JMP
22	@16
23	A=M
24	M=-1
25	@17
26	MD=M-1
27	@35
28	D;JEQ
- RAM Table:**

Address	Value
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0
- PC (Program Counter):** 0
- A (Accumulator):** 0
- Keyboard:** A virtual keyboard icon is shown below the RAM table.
- ALU (Arithmetic Logic Unit):**
  - D Input: 0
  - M/A Input: 0
  - ALU output: 0
- Display:** A large black rectangular area representing the screen.
- Status Bar:** Running...

When keyboard is not pressed:

File View Run Help

Print Step Back Step Forward Stop Animate: No animation View: Scr... Format: D...

Slow Fast

ROM

0	@16384
1	D=A
2	@3840
3	D=D+A
4	@16
5	M=D
6	@16
7	D=A
8	@17
9	M=D
10	@18
11	M=0
12	@8192
13	D=A
14	@19
15	M=D
16	@24576
17	D=M
18	@22
19	D;JNE
20	@147
21	0;JMP
22	@16
23	A=M
24	M=-1
25	@17
26	MD=M-1
27	@35
28	D;JEQ

PC 0

RAM

0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	22439
17	16
18	0
19	2137
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

A 0

D 0

ALU

D Input : 0

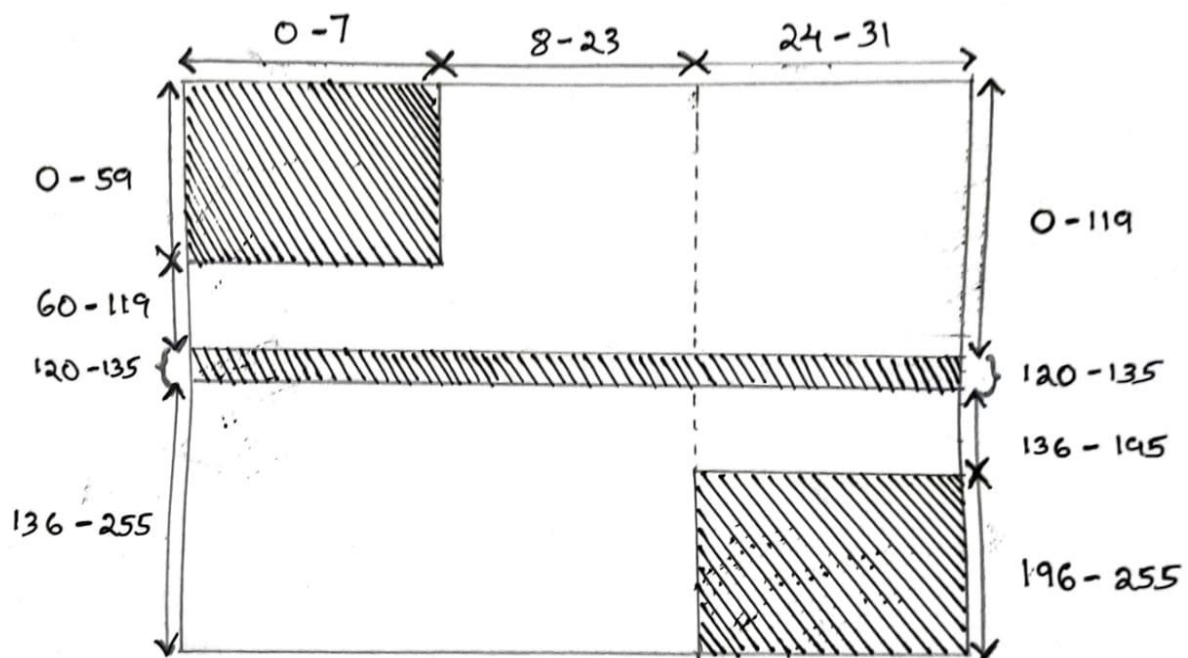
M/A Input : 0

ALU output : 0

Running...

## **Insights learned about Hack Assembly code while implementing the project**

- ❖ Came to know how to address each registers.
- ❖ Understood how to write to each pixel.
- ❖ Applying various conditional and unconditional loops to jump to different parts of the code where ever required.
- ❖ Got an idea about how to access the keyboard memory mapping.
- ❖ Came to know that for turning ON a pixel we write a '1' to the Bit and to turn it OFF we write a '0' to the Bit.
- ❖ Studied more about A and C instructions.
- ❖ We got a clear insight on using pointers to address the screen memory map.
- ❖ Accessing all the pixels on the screen using the screen memory map.
- ❖ Learned how to **on** specified pixels in the screen.
- ❖ Understood that variables are allocated to RAM[16] onward
- ❖ We came to know that it is very good practice to write pseudo code before writing the asm file .Since it helped us to reduce the chance of getting error in the codes .
- ❖ Using labels in the code had made the code more readable and easier to change the program flow.



\*\*\*\*\*