

Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of CSE
Tutorial-1

Programme: B.E
Course: Computer Organization

Term: Jan to May 2018
Course Code: CS45

Name: <i>Kaushik Kampli</i>	Marks: <i>10/10</i>	Date: <i>1/2/20</i>
USN: <i>1MS18CS057</i>	Signature of the Faculty:	<i>[Signature]</i>

Activity I: Assembling and disassembling of a computer

Objective: To demonstrate the functional units of a system.

Assembling of a system: A PC computer is a modular type of computer, it can be assembled using hardware components made by different manufacturers, so as to have a custom built computer according to one's specific needs.

Disassembling of a system: When referring to hardware, **disassemble** is the process of breaking down a device into separate parts. A device may be disassembled to help determine a problem, to replace a part, or to take the parts and use them in another device or to sell them individually.

Activity to be performed by students: Identify the different parts of the system including its interconnection. Observe the assembly and disassembly procedure.

Answer the following questions.

1. Write down the detailed procedure to assemble a system.
2. Explain how troubleshooting a system helps to trace and correct the faults in a system.
3. List out the procedure to install extra memory card to a system.
4. With a diagram explain different cables used to connect function units in a system.
5. Discuss the safety precautions one should take while removing components of a system.



MARKS :

Name :	Kavitha Kampli	Branch:	CSB
USN/Roll No. :	1MS18CS057	Sem/Sec:	III / B
Subject :	Computer Organization	Subject Code:	

- 1) Write down detailed procedure to assemble a system.

The procedure is :-

- 1) Make sure to have all the parts ready
 - 2) Install the power supply : some cans come with power supply pre installed, if not follow the power supply will go near the top or bottom area of the can
 - 3) Add components to the motherboard
- Attach the processor to the motherboard by finding the processor slot and placed it in the right orientation, which is indicated by an arrow
 - Attach RAM to the motherboards by snapping them into the slots, a little pressure may be necessary.
 - Apply thermal paste to the processor if necessary

- Attach the heat sink : This varies from heat sink to heat sink.
 - Solder the motherboard into place by the screws provided and make necessary connectors
 - Install your hard drive: Plug the hard drive's sata cable into the sata slot on the motherboard
 - Install any peripheral hardware such as graphic card
network card
etc
 - Plug the monitor to the cpu and install the os of your choice and you are good to go
- 2) Explain how troubleshooting a system helps to trace and correct failure in a system.
- we first describe the symptoms of malfunction and troubleshooting is the process of identifying the causes of these symptoms

It is the process of elimination and usually requires confirmation that one is returning the product to its working state.

It demands critical thinking. A basic principle is to start from simplest and most probable problems first.

Troubleshooting can also take the form of systematic checklist, procedure or flowchart made before the problem occurs.

One of the core principles of troubleshooting is that reproducible problems can be reliably isolated and resolved.

When talking about replacement, one must focus on 'adjustment or other modification' & not literally replace components.

3) List out the procedure to install extra memory card to a system.

1) Determine the mode and amount of RAM, the clock speed and memory.

- disconnect cables : Unplug the AC power card and disconnect all peripheral devices.
- Ground yourself : To discharge the built up static electricity.
- Check existing expansion slots : You will have to remove the old card, if no new slots are available.
- Uninstall old RAM : Remove by pushing outward on white ejector tabs.
- Insert new memory : Insert the stick such that it is perpendicular to the motherboard.
- Lock the memory stick into its place.
- Check the system properties and verify if everything is proper.
- Close the cover properly.



MARKS:

Name :	Kaushik Kampli	Branch:	CS5
USN/Roll No. :	IMS18CS057	Sem/Sec:	IV/B
Subject :	Computer Organization	Subject Code:	

Q) Write a neat labelled diagram; explain different cables used to connect to the function units in a system.

different cables are:-

i) VGA cable:

Also known as D-sub cable, analog video cable. Connect one end to computer monitor, television. Connect other end to VGA port.

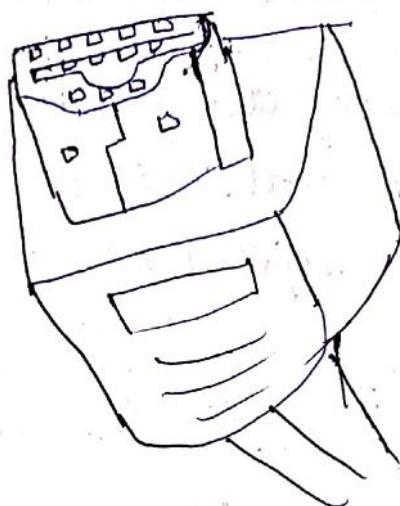


(a)



(b)

3) HDMI cable : Used to transmit display and sound to TV wires to make TV as an external monitor with sound speakers on TV.

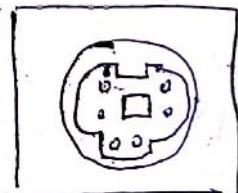
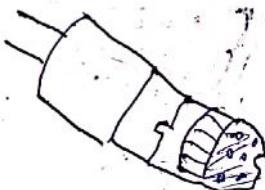


HDMI

3) PS/2 cable

Purple ps/2 cable : keyboard

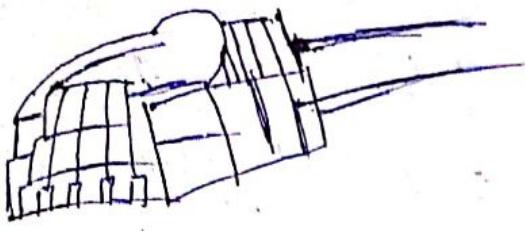
Green ps/2 port : mouse



4) Ethernet (RJ45 cable)

One end to : router, network switch

other end to : ethernet port on computer.



- 5) discuss the safety precautions one should take while removing components of a system.
- 1) workshop apparel: Avoid acrylic or wool sweater when working with electronic parts.
 - 2) unplug all computer equipment and peripherals before opening any case (only exception is if you were working w/o any anti-static mat.)
 - 3) retain all screws during disassembly
 - 4) do not forcefully remove components, as you may end up damaging the parts
 - 5) power supplies produce several voltage levels.

6) Do not damage any case or cover during disassembly, may cause problems during refitting of the system and its component.

Tutorial -II

Programme: B.E
Course: Computer Organization

Term: Jan to May 2020
Course Code: CS45

Name: <u>Kavashik Kamath</u>	Marks: <u>10</u> /10	Date: <u>1/1/2020</u>
USN: <u>18S18CS067</u>	Signature of the Faculty:	

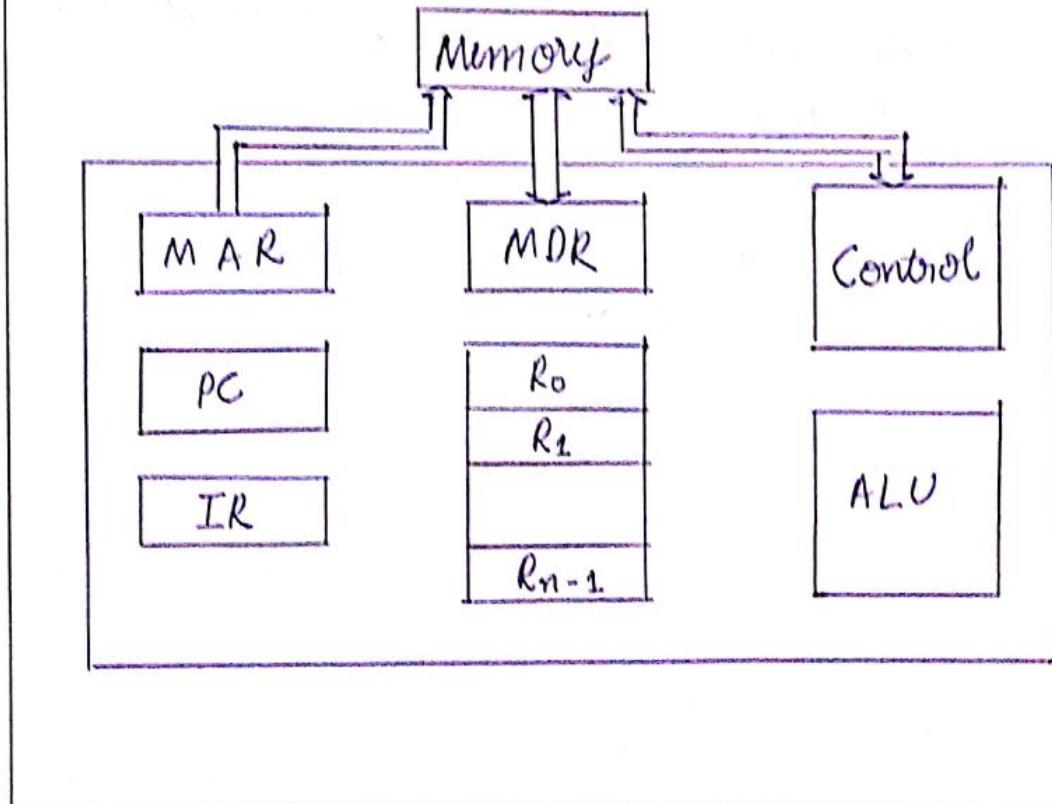
Activity II: Demonstrating Datapath and instruction execution stages using MarieSim Simulator

Objective: To simulate inter communication between CPU and memory.

Simulator Description: MarieSim is a computer architecture simulator based on the MARIE architecture. It provides users with interactive tools and simulations to help them deepen their understanding of the operation of a simple computer. One can observe how assembly language statements affect the registers and memory of a computer system.

Activity to be performed by students:

1. Draw the interconnection between memory and a processor.



- 2) The steps required to execute an instruction are
- 1) Fetching an instruction : The instruction is fetched from main memory. It is fetched and stored in IR.
 - 2) Decode instruction : During this cycle the encoded instruction is interpreted by the decoder in the IR
 - 3) ALU operation : The operations will be carried out based on the operator and operands
 - 4) Access memory : Only 2 kinds of instructions access memory, LOAD copies a value from memory to AC and STOU copies a register value to memory.
 - 5) Update Registers File : Result of ALU operation is written back to the register file to update it.
 - 6) Update the PC : The PC must be updated to the address of the next instruction, so that the program can be repeated.

2. List out the steps required to execute an instruction.

3. Write and execute assembly language program to compute

$$\text{i) } f = (g+h)*(i+y)$$

$$\text{ii) } d = b^2 - 4ac$$

4. Describe the factors affecting the performance of a processor

The major factors affecting performance are:-

- 1) Design of the computer : Speed is improved by pipelining, as simultaneous instructions can be executed.
- 2) The machine instruction set : The instructions used to write machine code also affect performance.
- 3) Design of hardware : The speed at which memory is recalled by the processor through a bus reduces drastically if the copy of the memory is placed in cache. It is read directly from cache.

Minor Factors are

- 1) Elapsed Time : Time taken for whole operation
- 2) Processor Time : Time during which the processor is active ($T = N \times S/R$)
- 3) Processor clock : Higher the clk rate, higher the no of basic steps per clock cycle.

3. Results and Snapshots:

3) i) $f = (g+h)^*(i+g)$ ii) $b^2 - 4ac$

LOAD G
 ADD H
 STORE A
 LOAD I
 ADD Y
 STORE B
 loop, LOAD A
 ADD F
 STORE F
 LOAD B
 SUBT one
 STORE B
 SKIPCOND 400
 JUMP loop

LOAD F
 OUTPUT
 HALT

G, DEC 9
 H, DEC 5
 Y, DEC 8
 I, DEC 2
 A, DEC 0
 B, DEC 0
 F, DEC 0
 one DEC 1

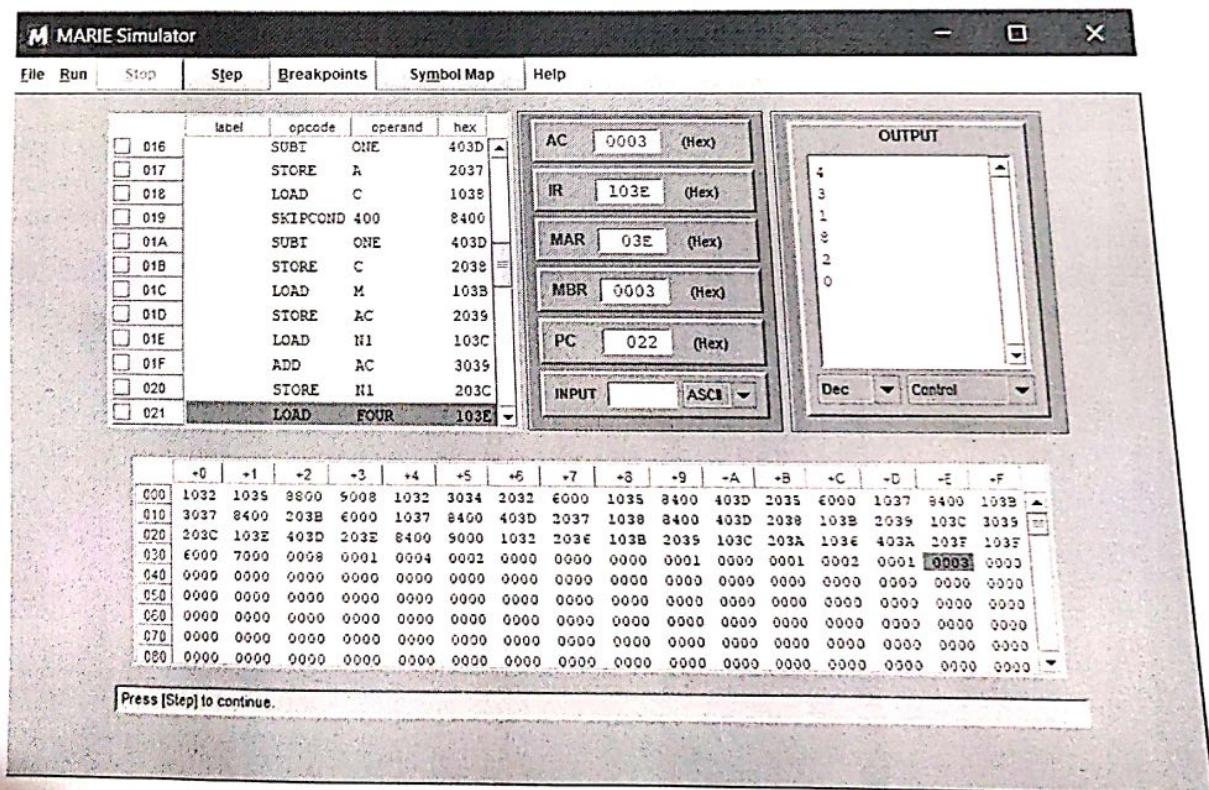
LOAD B
 STORE O
 first, LOAD B
 ADD X
 STORE X
 LOAD O
 SUBT one
 STORE O.
 SKIPCOND 400
 JUMP first

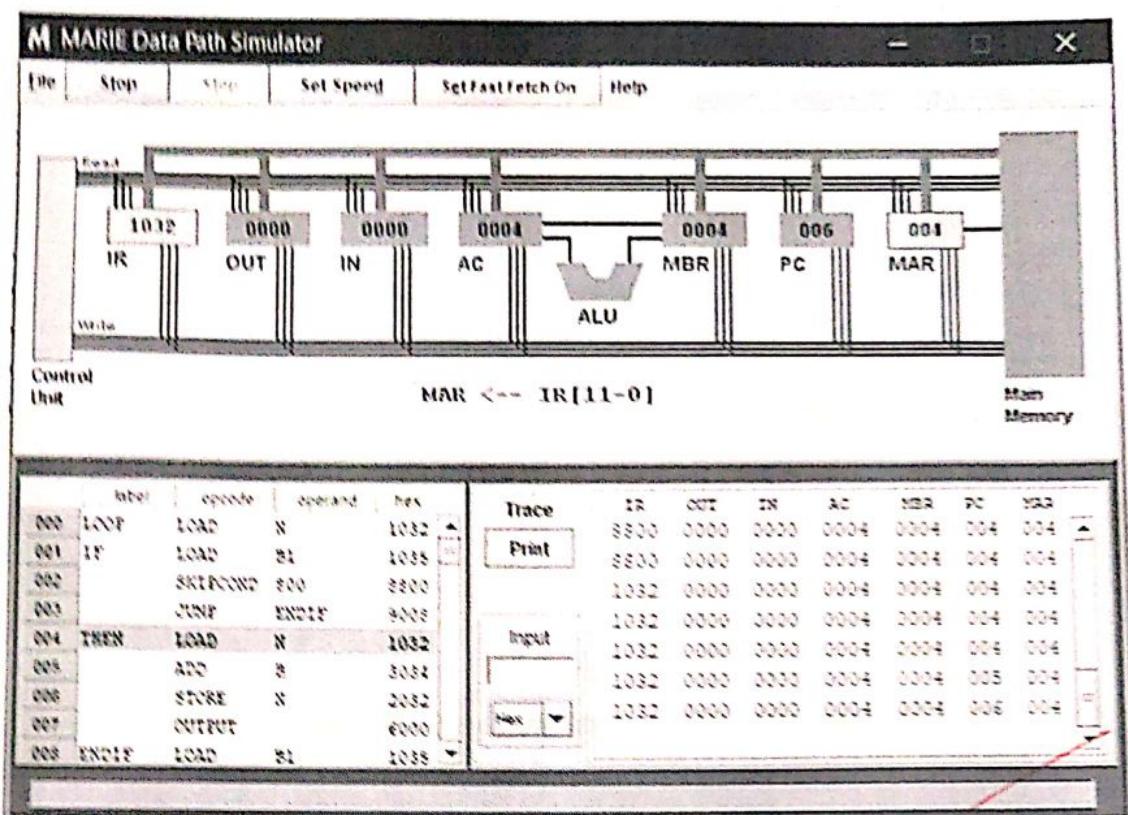
second, LOAD A
 ADD Y
 STORE Y
 LOAD C
 SUBT one
 STORE C
 SKIPCOND 400
 JUMP second

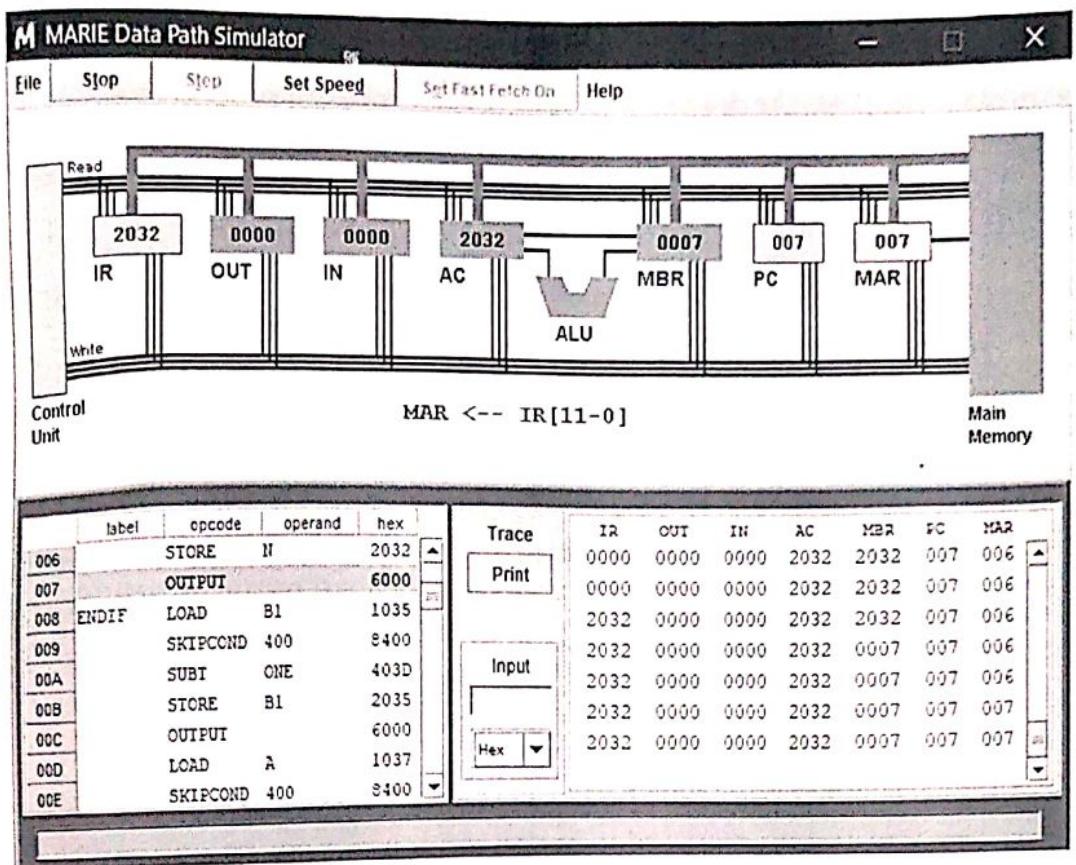
third, LOAD four
 ADD Z
 STORE Z
 LOAD Y
 SUBT one
 STORE Y
 SKIPCOND 400
 JUMP third

LOAD D
 ADD X
 SUBT Z
 OUTPUT
 HALT

$$f = (g + h)^* (i + y)$$







$$d = b^2 - h^+ a^+ c$$

MARIE Assembler Code Editor

```

File Edit Assemble Help
LOAD G
ADD H
STORE G
LOAD I
ADD I
STORE I
LOAD G
LOOP,   LOAD N
        ADD G
        STORE N

        LOAD I
        SUBT ONE
        STORE I

        SKIPCOND 400
        JUMP LOOP

LOAD N
STORE F
HALT
N,      DEC 0
G,      DEC 1
H,      DEC 1
I,      DEC 2
Y,      DEC 2
F,      DEC 0
ONE,    DEC 1

```

C:\Users\Manas Shankar\Desktop\MarieSim\multAdd\multAdd.mas Assembly successful.

M Assembly Listing for multAdd.mas

```

600 1013 I LOAD G
601 3014 I ADD H
602 2012 I STORE G
603 3015 I LOAD I
604 3014 I ADD Y
605 2015 I STORE I
606 1012 I LOAD G
607 2012 I STORE H
608 3013 I ADD G
609 2012 I STORE I

610 1015 I LOAD I
611 3018 I SUBT ONE
612 2015 I STORE I

613 8400 I SKIPCOND 400
614 9007 I CND 500F

615 1012 I LOAD H
616 2017 I STORE F
617 7000 I HALT
618 9005 I E DEC 0
619 9005 I E DEC 1
620 9005 I E DEC 1
621 9005 I E DEC 2
622 9005 I E DEC 2
623 9005 I E DEC 0
624 9005 I E DEC 1
625 9005 I E DEC 1
626 9005 I E DEC 2
627 9005 I E DEC 0
628 9005 I E DEC 1

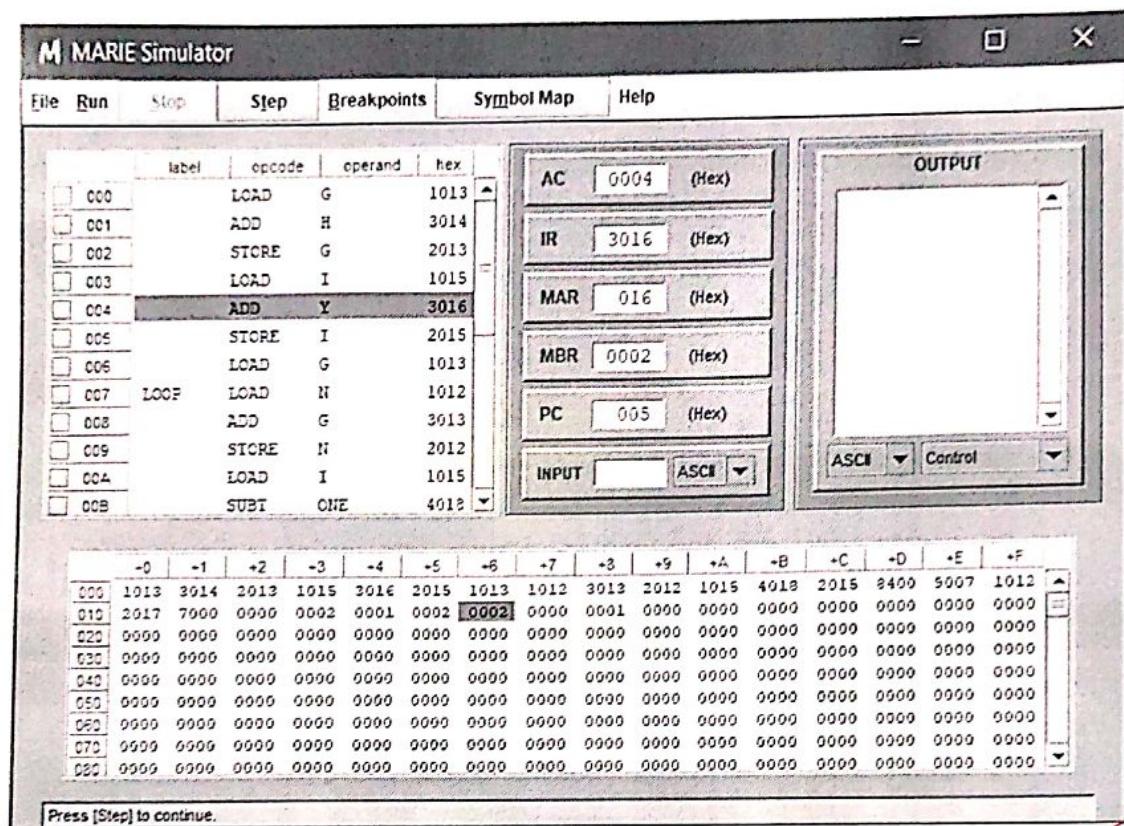
```

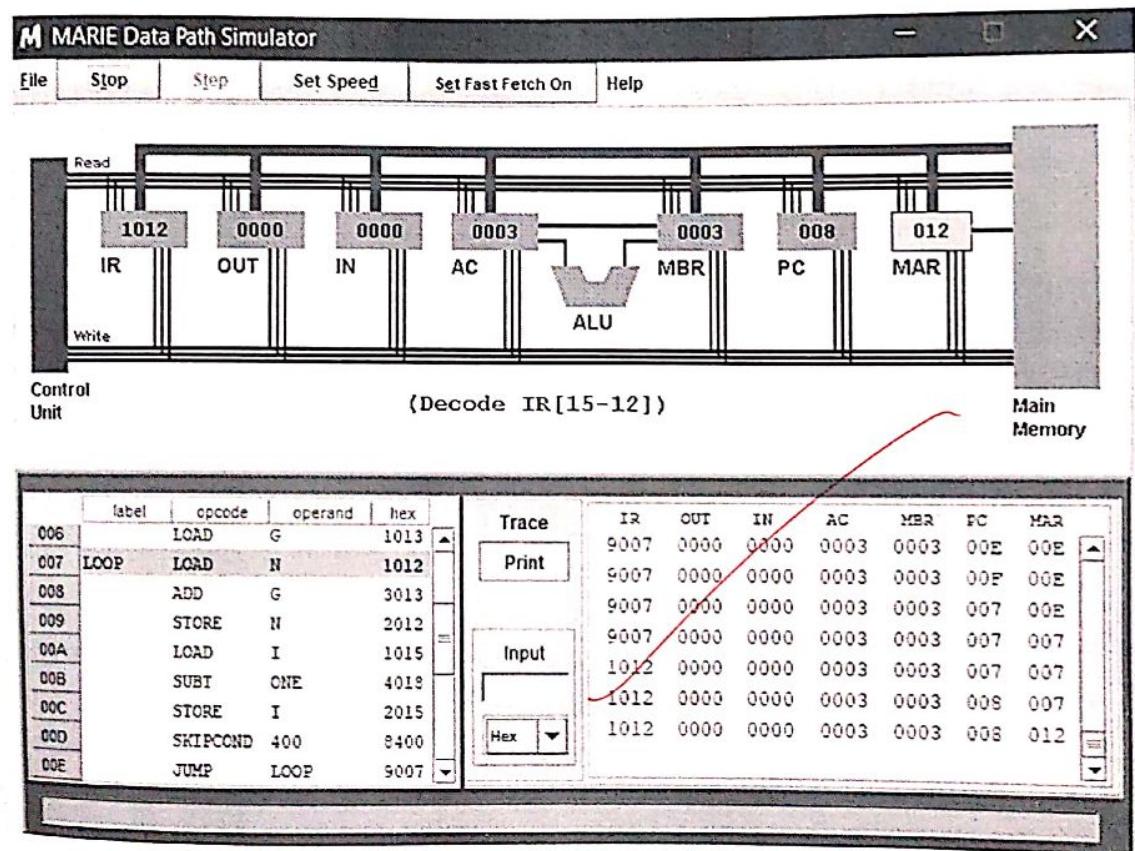
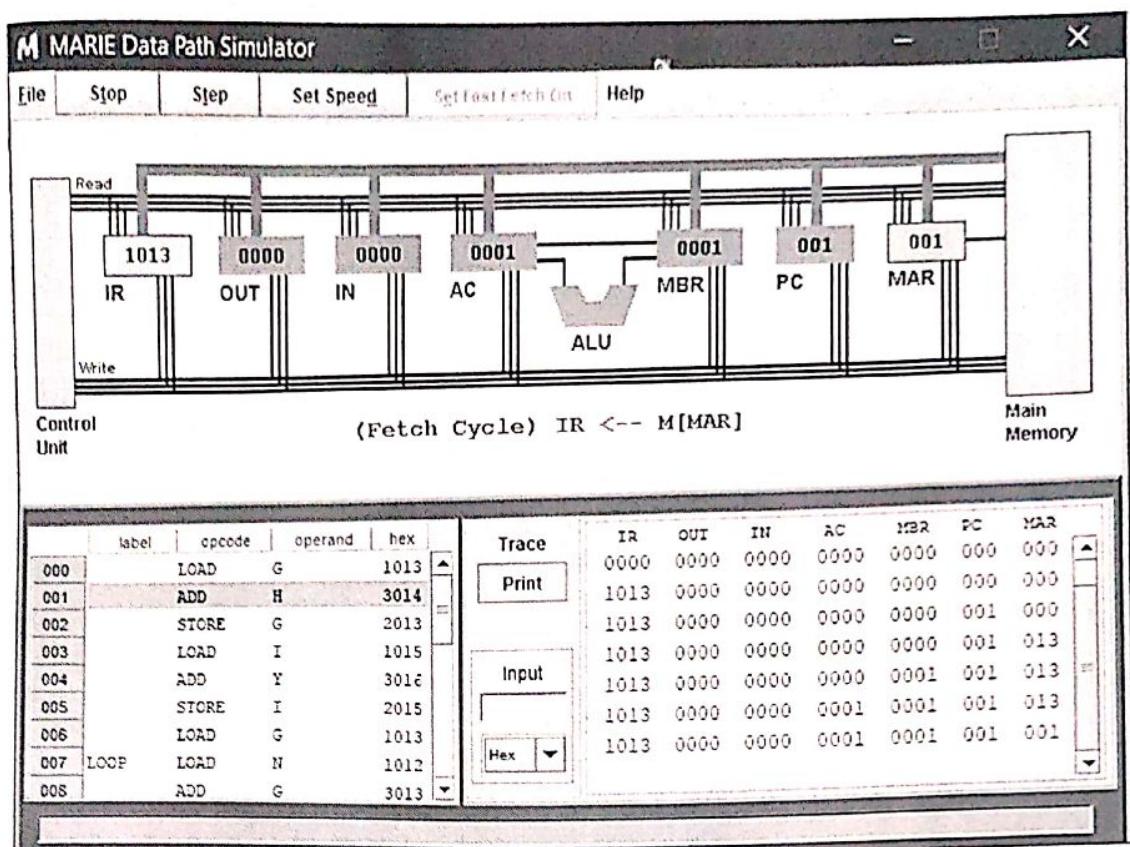
Assembly successful.

SIMDOS TABLE

Symbol	Defined	References
F	I 017 I 010	
G	I 012 I 000, 001, 004, 005	
H	I 014 I 001	
I	I 013 I 003, 005, 014, 006	
LOOP	I 007 I 022	
ONE	I 012 I 037, 009, 008	
Y	I 018 I 028	
Z	I 016 I 004	

Print Close





Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of CSE

Tutorial -III

Programme: B.E

Term: Jan to May 2020

Course: Computer Organization Course Code: CS45

Name: Kaushik Kampli	Marks: 10 /10	Date:
USN: 1MS1KCS057	Signature of the Faculty:	UP 12/12/2022

Objective: To simulate ARM Instruction set using ARMSim simulator.

Simulator Used: ARMSim 1.91 is a desktop application running in a Windows environment. It allows users to simulate the execution of ARM assembly language programs on a system based on the ARM7TDMI processor.

ARM enables the users both to debug ARM assembly programs and to monitor the state of the system while a program executes.

Activity to be performed by students:

- 1) Write an ARM program to perform basic arithmetic operations.
- 2) Write an ARM program to demonstrate the working of load and store instructions.
- 3) Write an ARM program to evaluate expression $f=(g+h)-(i+j)$
- 4) Write an ARM program to find the sum of all elements of an array.
- 5) Write an ARM program to find the factorial of a number.

Programs and the snapshots:

MARKS :

Name :	Kaushik Kamphli	Branch:	CSE
USN/Roll No. :	IMS18CS057	Sem/Sec:	IV/B
Subject :	Computer Organization	Subject Code:	

Tutorial - III

- 1) ARM program to perform basic arithmetic operations

```

MOV R1, #0x00000020
MOV R2, #2
MOV R3, #1
MOV R4, #1
ADD R1, R1, R2
ADD R3, R3, R4
SUB R1, R1, R3
  
```

SWI 0x11

- 2) ARM program to demonstrate load and store instructions

```

MOV R1, #0x00000030
MOV R3, #0
MOV R4, #5
  
```

STR R4, [R1, R3]
LDR R5, [R1, R3]

SWI 0x11

3) write an ARM program to evaluate expression
 $f = (g+h)-(i+j)$

MOV R3, #h1
MOV R4, #h5
MOV R7, #35
MOV R8, #38

LDR R1, =0x00000064
LDR R2, =0x00000062
LDR R9, =0x00000074
LDR R10, =0x00000078

STR R3, [R1]
STR R4, [R2]
STR R7, [R9]
STR R8, [R10]

LDR R5, [R1]
LDR R6, [R2]
LDR R11, [R9]
LDR R12, [R10]

ADD R5, &R5, R6
ADD R11, R11, R12
SUB R0, R5, R11

SWI 0x11

4) ARM program to find the sum of all elements in an array

```
MOV R0, #5  
LDR R1, =array  
loop: LDR R2, [R1], #4  
      ADD R3, R3, R2  
      SUB R0, R0, #1  
      CMP R0, #00  
      BNE loop  
      SWI 0x11
```

.data
array: word 0x00000001, 0x0000000A, 0x000000AB
 0x0000000B, 0x00000008

5) write an ARM program to find the factorial of a number.

```
LDR R1, =array  
LDR R2, [R1]  
LDR R3, [R1]  
loop: SUB R2, R2, #1  
      CMR R2, #0  
      BEQ end  
      MUL R0, R2, R3  
      MOV R3, R0
```

CMP R2, #0

BNE loop

SWI 0x11

• data

fact: .word 0x00000005,

end fact

fact end

ARMsim - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

Registers

General Purpose: R0-R3 R4-R12 R13-R15

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 0
R1 : 32
R2 : 12
R3 : 12
R4 : 1
R5 : 0
R6 : 0
R7 : 0
R8 : 0
R9 : 0
R10(:el):0
R11(fp):0
R12(ip):0
R13(ep):17408
R14(lr):0
R15(pc):28

CPSR Register

Negative(N):0
Zero(Z):0
Carry(C):0
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T):10
CPU Mode :SY
0x000000df

Memory

EXT.s

```
00000000:EX00017200 MOV R1, #0x40000000
00000004:EX00017201 MOV R2, #1
00000008:EX00017202 MOV R3, #1
0000000C:EX00017203 MOV R4, #1
00000010:EX0001720F ADD R1, R1, R2
00000014:EX00017204 ADD R3, R3, R4
00000018:EX00017203 SUB R1, R1, R3
0000001C:EX00000111 SWI $x11
```

Outputs

Console Stdin Stdout Stderr

Loading assembly language file C:\Users\INNOVITIA-PC-01\Desktop\kaus\EXP.s

12:42 PM 04-03-2020

ARMsim - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

Registers

General Purpose: R0-R3 R4-R12 R13-R15

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 0
R1 : 48
R2 : 0
R3 : 0
R4 : 5
R5 : 15
R6 : 0
R7 : 0
R8 : 0
R9 : 0
R10(:el):0
R11(fp):0
R12(ip):0
R13(ep):17408
R14(lr):0
R15(pc):120

CPSR Register

Negative(N):0
Zero(Z):0
Carry(C):0
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T):0
CPU Mode :SY
0x000000df

Memory

STORE.s

```
00000000:EX00014000 MOV R1, #0x00000000
00000004:EX00014000 MOV R3, #0
00000008:EX00014003 MOV R4, #3
0000000C:EX00014003 STR R4, [R1,R3]
00000010:EX00014003 Ldr R3, [R1,R3]
00000014:EX00000111 SWI $x11
```

Outputs

Console Stdin Stdout Stderr

Loading assembly language file C:\Users\INNOVITIA-PC-01\Desktop\kaus\STORE.s

12:39 PM 04-03-2020

The screenshot shows the ARMsim interface with the following details:

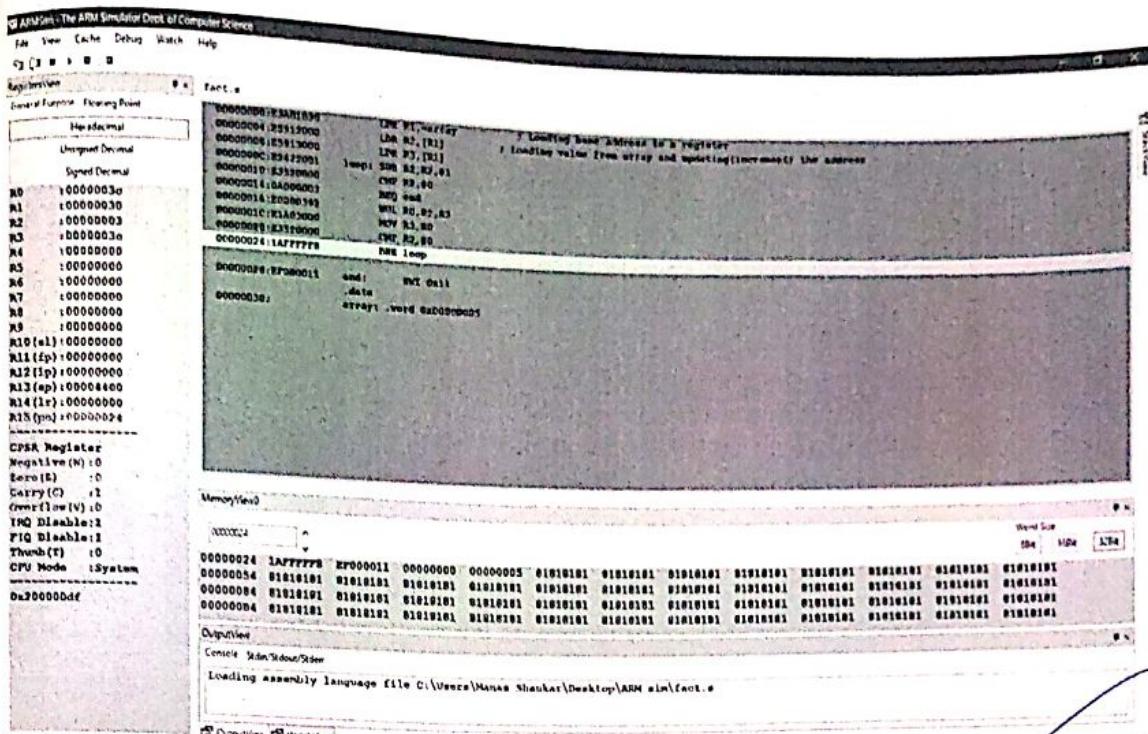
- Registers:** Shows the state of various registers (R0-R15, PC, CPSR) and memory locations.
- Assembly Code:**

```

MOV R2, #0           ; Initializing counter register
LDR R1, =array       ; Loading base address to a register
loop: LDR R2,[R1],#4  ; Loading value from array and updating(increment) the address
      ADD R1,R1,#4
      SDR R0,R2,0        ; Sum is stored in R0 register
      SUB R0,R0,#1        ; Decreasing counter value
      CMP R0, #0           ; Checking counter value

    BNE loop
    SMC Call
    .data
array: .word 0x00000001, 0x0000000A, 0x0000000B, 0x0000000C, 0x0000000D

```
- CPSR Register:** Shows flags: Negative(N), Zero(Z), Carry(C), Overflow(V), IRQ Disable, FIQ Disable, Thumb(T), and CPU Mode (set to System).
- Memory Dump:** A window showing memory starting at address 0x00000004, displaying the values 00000001, 0000000A, 0000000B, 0000000C, 0000000D, followed by four rows of data.
- Output Window:** Displays the message "Loading assembly language file C:\Users\Manas Shekhar\Desktop\ARM sim\array.s".



Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)

Department of CSE

Programme: B.E
Course: Computer Organization

Tutorial IV

Term: Jan to May 2019
Course Code: CS45

Activity IV: Executing ARM programs using ARMsim simulator.

Name: <i>Kausik Kampli</i>	Marks: /10	Date: 11/02/2020
USN: <i>1MS18CS057</i>	Signature of the Faculty:	



Objective: To simulate ARM Instruction set using ARMsim simulator.

Simulator Used: ARMSim 1.91 is a desktop application running in a Windows environment. It allows users to simulate the execution of ARM assembly language programs on a system based on the ARM7TDMI processor.

ARM enables the users both to debug ARM assembly programs and to monitor the state of the system while a program executes.

Activity to be performed by students:

- 1) Write an ARM program to generate Fibonacci Series.
- 2) Write an ARM to search an element in an array and print Y if found and print N if not found.
- 3) Write an ARM program to find the length of a string and copying one string to another.





MARKS :

Name :	Kaushile Kamthli	Branch:	CSB
USN/Roll No. :	IMS18CS057	Sem/Sec:	IV/B
Subject :	CO Lab IV	Subject Code:	

ARM Program to generate Fibonacci sequence

```
    mov  r0,#0
    mov  r1,#1
    mov  r2,#20
    mov  r3,#0
    ldr  r4,=0x00002000
    mov  r5,#0
loop: str  r0,[r4,r5]
        add  r6,r0,r1
        mov  r0,r1
        mov  r1,r6
        add  r5,r5,#1
        add  r3,r3,#1
        cmp  r3,r2
        blt loop
        swi  0x22
        swi  0x11
```

Program to search an element in an array

```
mov r8, #4  
mov r4, #'n'  
mov r1, #16  
ldr r0, =0x00002000  
mov r5, #4  
loop : ldr r2, [r0, r5]  
      sub r8, r8, #1  
      add r5, r5, #4  
      cmh r1, r2  
      bne phistly  
      cmh r8, #0  
      bne phintn  
      bne loop  
charint : str r4, [r0]  
      ldr r0, [r0]  
      swi 0x00  
      b end  
charinty : str r3, [r0]  
      ldr r0, [r0]  
      swi 0x00
```

Program
mov
mov
loop

Program to copy a string

```
mov d0, #0x00001000  
mov d1, #'a'  
loop: stib d1, [d0, #0]  
      mov d2, #'b'  
      std b d2, [d0, #1]  
      mov d3, #'c'  
      std b d3, [d0, #2]  
      mov d4, #0  
      std b d4, [d0, #3]  
      mov d7, #0  
      mov d8, #0  
      mov d5, #10x00003000  
      mov d9, #0
```

```
li:  
      ddib d6, [d0, d7]  
      std b d6, [d5, d8]  
      add d7, d7, #1  
      add d8, d8, #1  
      add d9, d9, #1  
      cmph d9, #5  
      bne li  
      swi 0x11
```

Deployment		Resource	Count
Resource Type	Running Process		
Process			
Unjoined Devices			
Joined Devices			
R0	122		
R1	119		
R2	118		
R3	123		
R4	119		
R5	124		
R6	120		
R7	19		
R8	18		
R9	123 (123.0)		
R10	123 (123.1)		
R11	123 (123.2)		
R12	123 (123.3)		
R13	123 (123.4)		
R14	123 (123.5)		
R15	123 (123.6)		
R16	123 (123.7)		
R17	123 (123.8)		
R18	123 (123.9)		
R19	123 (123.10)		
R20	123 (123.11)		
R21	123 (123.12)		
R22	123 (123.13)		
R23	123 (123.14)		
R24	123 (123.15)		
R25	123 (123.16)		
R26	123 (123.17)		
R27	123 (123.18)		
R28	123 (123.19)		
R29	123 (123.20)		
R30	123 (123.21)		
R31	123 (123.22)		
R32	123 (123.23)		
R33	123 (123.24)		
R34	123 (123.25)		
R35	123 (123.26)		
R36	123 (123.27)		
R37	123 (123.28)		
R38	123 (123.29)		
R39	123 (123.30)		
R40	123 (123.31)		
R41	123 (123.32)		
R42	123 (123.33)		
R43	123 (123.34)		
R44	123 (123.35)		
R45	123 (123.36)		
R46	123 (123.37)		
R47	123 (123.38)		
R48	123 (123.39)		
R49	123 (123.40)		
R50	123 (123.41)		
R51	123 (123.42)		
R52	123 (123.43)		
R53	123 (123.44)		
R54	123 (123.45)		
R55	123 (123.46)		
R56	123 (123.47)		
R57	123 (123.48)		
R58	123 (123.49)		
R59	123 (123.50)		
R60	123 (123.51)		
R61	123 (123.52)		
R62	123 (123.53)		
R63	123 (123.54)		
R64	123 (123.55)		
R65	123 (123.56)		
R66	123 (123.57)		
R67	123 (123.58)		
R68	123 (123.59)		
R69	123 (123.60)		
R70	123 (123.61)		
R71	123 (123.62)		
R72	123 (123.63)		
R73	123 (123.64)		
R74	123 (123.65)		
R75	123 (123.66)		
R76	123 (123.67)		
R77	123 (123.68)		
R78	123 (123.69)		
R79	123 (123.70)		
R80	123 (123.71)		
R81	123 (123.72)		
R82	123 (123.73)		
R83	123 (123.74)		
R84	123 (123.75)		
R85	123 (123.76)		
R86	123 (123.77)		
R87	123 (123.78)		
R88	123 (123.79)		
R89	123 (123.80)		
R90	123 (123.81)		
R91	123 (123.82)		
R92	123 (123.83)		
R93	123 (123.84)		
R94	123 (123.85)		
R95	123 (123.86)		
R96	123 (123.87)		
R97	123 (123.88)		
R98	123 (123.89)		
R99	123 (123.90)		
R100	123 (123.91)		
R101	123 (123.92)		
R102	123 (123.93)		
R103	123 (123.94)		
R104	123 (123.95)		
R105	123 (123.96)		
R106	123 (123.97)		
R107	123 (123.98)		
R108	123 (123.99)		
R109	123 (123.100)		
R110	123 (123.101)		
R111	123 (123.102)		
R112	123 (123.103)		
R113	123 (123.104)		
R114	123 (123.105)		
R115	123 (123.106)		
R116	123 (123.107)		
R117	123 (123.108)		
R118	123 (123.109)		
R119	123 (123.110)		
R120	123 (123.111)		
R121	123 (123.112)		
R122	123 (123.113)		
R123	123 (123.114)		
R124	123 (123.115)		
R125	123 (123.116)		
R126	123 (123.117)		
R127	123 (123.118)		
R128	123 (123.119)		
R129	123 (123.120)		
R130	123 (123.121)		
R131	123 (123.122)		
R132	123 (123.123)		
R133	123 (123.124)		
R134	123 (123.125)		
R135	123 (123.126)		
R136	123 (123.127)		
R137	123 (123.128)		
R138	123 (123.129)		
R139	123 (123.130)		
R140	123 (123.131)		
R141	123 (123.132)		
R142	123 (123.133)		
R143	123 (123.134)		
R144	123 (123.135)		
R145	123 (123.136)		
R146	123 (123.137)		
R147	123 (123.138)		
R148	123 (123.139)		
R149	123 (123.140)		
R150	123 (123.141)		
R151	123 (123.142)		
R152	123 (123.143)		
R153	123 (123.144)		
R154	123 (123.145)		
R155	123 (123.146)		
R156	123 (123.147)		
R157	123 (123.148)		
R158	123 (123.149)		
R159	123 (123.150)		
R160	123 (123.151)		
R161	123 (123.152)		
R162	123 (123.153)		
R163	123 (123.154)		
R164	123 (123.155)		
R165	123 (123.156)		
R166	123 (123.157)		
R167	123 (123.158)		
R168	123 (123.159)		
R169	123 (123.160)		
R170	123 (123.161)		
R171	123 (123.162)		
R172	123 (123.163)		
R173	123 (123.164)		
R174	123 (123.165)		
R175	123 (123.166)		
R176	123 (123.167)		
R177	123 (123.168)		
R178	123 (123.169)		
R179	123 (123.170)		
R180	123 (123.171)		
R181	123 (123.172)		
R182	123 (123.173)		
R183	123 (123.174)		
R184	123 (123.175)		
R185	123 (123.176)		
R186	123 (123.177)		
R187	123 (123.178)		
R188	123 (123.179)		
R189	123 (123.180)		
R190	123 (123.181)		
R191	123 (123.182)		
R192	123 (123.183)		
R193	123 (123.184)		
R194	123 (123.185)		
R195	123 (123.186)		
R196	123 (123.187)		
R197	123 (123.188)		
R198	123 (123.189)		
R199	123 (123.190)		
R200	123 (123.191)		
R201	123 (123.192)		
R202	123 (123.193)		
R203	123 (123.194)		
R204	123 (123.195)		
R205	123 (123.196)		
R206	123 (123.197)		
R207	123 (123.198)		
R208	123 (123.199)		
R209	123 (123.200)		
R210	123 (123.201)		
R211	123 (123.202)		
R212	123 (123.203)		
R213	123 (123.204)		
R214	123 (123.205)		
R215	123 (123.206)		
R216	123 (123.207)		
R217	123 (123.208)		
R218	123 (123.209)		
R219	123 (123.210)		
R220	123 (123.211)		
R221	123 (123.212)		
R222	123 (123.213)		
R223	123 (123.214)		
R224	123 (123.215)		
R225	123 (123.216)		
R226	123 (123.217)		
R227	123 (123.218)		
R228	123 (123.219)		
R229	123 (123.220)		
R230	123 (123.221)		
R231	123 (123.222)		
R232	123 (123.223)		
R233	123 (123.224)		
R234	123 (123.225)		
R235	123 (123.226)		
R236	123 (123.227)		
R237	123 (123.228)		
R238	123 (123.229)		
R239	123 (123.230)		
R240	123 (123.231)		
R241	123 (123.232)		
R242	123 (123.233)		
R243	123 (123.234)		
R244	123 (123.235)		
R245	123 (123.236)		
R246	123 (123.237)		
R247	123 (123.238)		
R248	123 (123.239)		
R249	123 (123.240)		
R250	123 (123.241)		
R251	123 (123.242)		
R252	123 (123.243)		
R253	123 (123.244)		
R254	123 (123.245)		
R255	123 (123.246)		
R256	123 (123.247)		
R257	123 (123.248)		
R258	123 (123.249)		
R259	123 (123.250)		
R260	123 (123.251)		
R261	123 (123.252)		
R262	123 (123.253)		
R263	123 (123.254)		
R264	123 (123.255)		
R265	123 (123.256)		
R266	123 (123.257)		
R267	123 (123.258)		
R268	123 (123.259)		
R269	123 (123.260)		
R270	123 (123.261)		
R271	123 (123.262)		
R272	123 (123.263)		
R273	123 (123.264)		
R274	123 (123.265)		
R275	123 (123.266)		
R276	123 (123.267)		
R277	123 (123.268)		
R278	123 (123.269)		
R279	123 (123.270)		
R280	123 (123.271)		
R281	123 (123.272)		
R282	123 (123.273)		
R283	123 (123.274)		
R284	123 (123.275)		
R285	123 (123.276)		
R286	123 (123.277)		
R287	123 (123.278)		
R288	123 (123.279)		
R289	123 (123.280)		
R290	123 (123.281)		
R291	123 (123.282)		
R292	123 (123.283)		
R293	123 (123.284)		
R294	123 (123.285)		
R295	123 (123.286)		
R296	123 (123.287)		
R297	123 (123.288)		
R298	123 (123.289)		
R299	123 (123.290)		
R300	123 (123.291)		
R301	123 (123.292)		
R302	123 (123.293)		
R303	123 (123.294)		
R304	123 (123.295)		
R305	123 (123.296)		
R306	123 (123.297)		
R307	123 (123.298)		
R308	123 (123.299)		
R309	123 (123.300)		
R310	123 (123.301)		
R311	123 (123.302)		
R312	123 (123.303)		
R313	123 (123.304)		
R314	123 (123.305)		
R315	123 (123.306)		
R316	123 (123.307)		
R317	123 (123.308)		
R318	123 (123.309)		
R319	123 (123.310)		
R320	123 (123.311)		
R321	123 (123.312)		
R322	123 (123.313)		
R323	123 (123.314)		
R324	123 (123.315)		
R325	123 (123.316)		
R326	123 (123.317)		
R327	123 (123.318)		
R328	123 (123.319)		
R329	123 (123.320)		
R330	123 (123.321)		
R331	123 (123.322)		
R332	123 (123.323)		
R333	123 (123.324)		
R334	123 (123.325)		
R335	123 (123.326)		
R336	123 (123.327)		
R337	123 (123.328)		
R338	123 (123.329)		
R339	123 (123.330)		
R340	123 (123.331)		
R341	123 (123.33		

Lab 5

Kaushik Kampli
IMSI18CS057
IV 'B'

Activity V : Designing an ALU to perform arithmetic and logical functions using logicsim simulator

Objective : To simulate working of ALU using simulator.

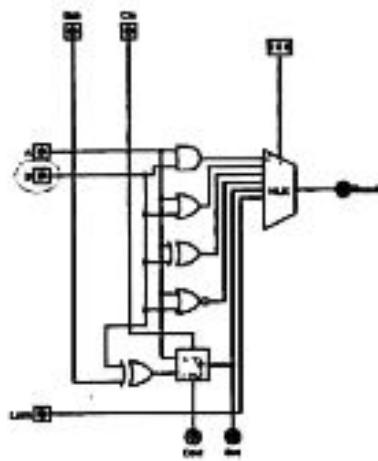
Activity to be performed by students :

List out the steps in designing ALU :

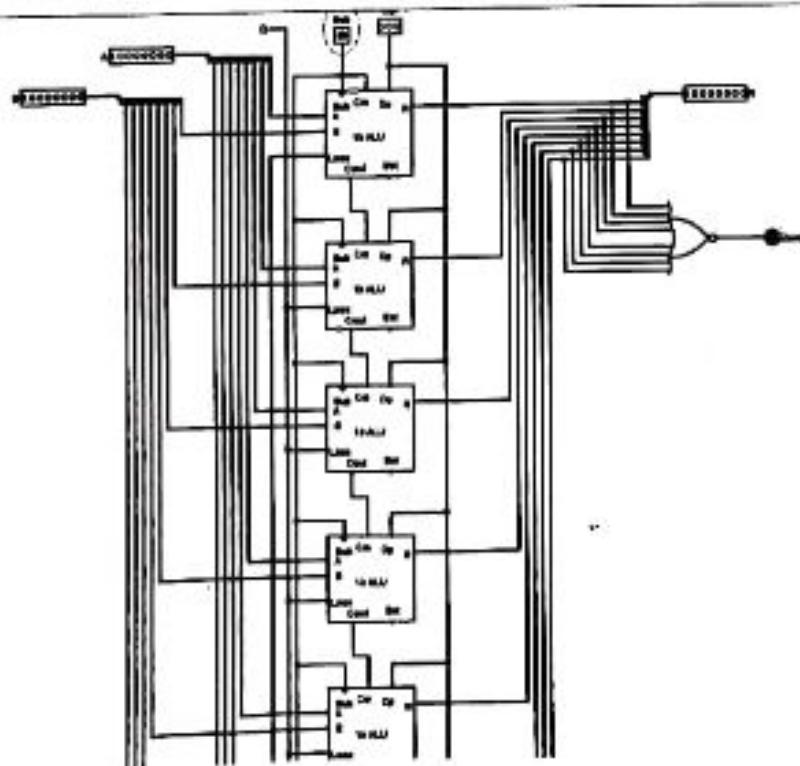
1. Add the two i/p pins. Name them A and B .
2. Add OR, AND, XOR, NOR gates and a 1-bit adder .
3. Connect the A's and B's of all the gates to their respective pins
4. Add an output pin and name it Result .
5. Add a 1-bit multiplier with 3 select bits
6. Connect outputs of all gates to the MUX
7. Connect 3-bit i/p pin to MUX
8. Add i/p pin to C_m and o/p pin to C_{out} .
9. Add an XOR gate, connect its o/p to C_{out} . The first i/p must be connected to B and second to another i/p pin sub .
10. Add another i/p and name it d_s . Connect it to the MUX
11. Add an output pin and name it out , connect it to the o/p of adder unit .

Snapshots :

1-bit Arithmetic Logic Unit (ALU)



8-bit Arithmetic Logic Unit (ALU)



Lab - 6

Kaushik Kampli
1MS18CS057
IV (B)

Activity VI : To stimulate writing operation on memory and hence to design a memory system using logic simulation.

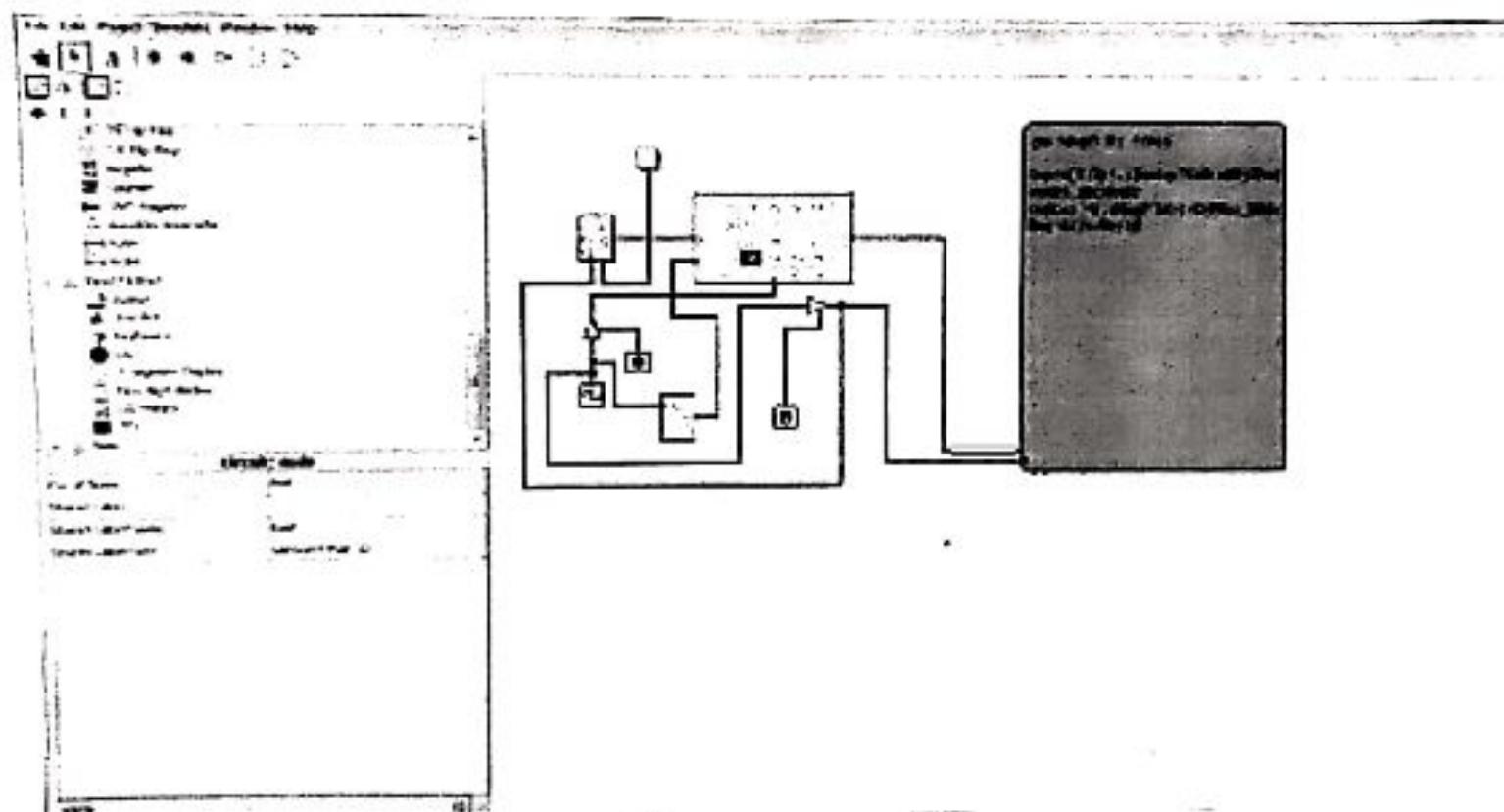
Objective : To stimulate writing operation on memory.

Action to be performed by students:

List out the steps in designing memory system :

1. Add a RAM with separate load and store selected.
2. Add a counter and connect Q to A of the RAM.
3. Add a controller buffer and connect its o/p to the RAM.
4. Add a clock and connect to this i/p of the buffer.
5. Add a TTY unit with 32 rows and columns
make a connection with the RAM.
6. Add a 7-bit random number generator, connected to D.
7. Add another controlled buffer, connect it to TTY.
Also add an i/p pin to the buffer.
8. Connect o/p of the second buffer to the counter
9. Connect a button to the counter

Observations and Snapshots:



Scanned with CamScanner

Scanned with CamScanner