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: MANAS.P.S	**Marks:** 10/10	**Date:** 21/1/2020
USN: 1MS18CS065	**Signature of the Faculty:**	

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

RAMAIAH
Institute of Technology

Data Sheet

MARKS :

Name :	MANAS. P. S	Branch:	CSE
USN/Roll No. :	1MS18CS065	Sem/Sec:	III / B
Subject :	Computer Org. & Architecture	Subject Code:	

(1) Write down the detailed procedure to assemble a system.

The procedure is as follows :

* Remove side Panels on case : The panels are removed from this case with thumb screws. The model's manual provides more insight into this, for mounting motherboard, thread into corresponding holes in the case.
* Insert motherboard : The I/O panel faceplate needs to be snapped into location in the back of case. After tightening the screws, install components into motherboard.
* check clearances : Make sure there is enough space for each of the components to be installed.

* Front panel connections : Now attach the connections for buttons, lights, USB ports and audio connections. Refer to the manual. Do not use force.
* Install Power supply: The supply is modular and is screwed into the back panel by 4 screws, though some cases have a clamp.
* Power Motherboard: Run the motherboard cable first and connect it first.
* Installing optical drive : It is handy to have an optical device (DVD/CD) installed using the tool-less design of the case.
* Installing hard drives : A computer may use 4 drives, two in raid and rest for a main drive and miscellaneous storage. The direction of placement may differ, and using clips can help.
* Connect cables : The cables are keyed so they will only fit in one direction into the board.
* Install RAM, Graphics card : Place in slots

closest to the CPU, making sure the notch is lined up. The manual should tell us if a 6-pin or 8-pin cable is needed for the graphics cards.

* Cable Management: Orienting the cables is an important task. Make sure the back panel does not leave a large gap if you have several cables running over others.

(2) Explain how troubleshooting a system helps to trace and correct failure in a system.

Troubleshooting is the diagnosis of "trouble" in the management flow of a system caused by some failure. We first describe the symptoms of malfunction and troubleshooting is the process of remedying the causes of these symptoms

It is a process of elimination and usually requires confirmation that the solution restores the product to its working state. It demands critical thinking. A basic principle is to start from the simplest and most probable problems first. It helps to try starting from

a known good state, like a computer reboot. Comprehensive documentation is also very helpful. Troubleshooting can also take the form of a systematic checklist, procedure or flowchart made before the problem occurs. We can first check for "frequently encountered" conditions first. 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" and not literally replace components.

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

Steps are :

* Determine the mode and amount of RAM your computer needs : We must know the exact specifications and configuration of compatible RAM needed.
* Disconnect cables : Unplug the AC power cord and disconnect all peripheral devices

RAMAIAH
Institute of Technology

Data Sheet

MARKS :

Name :	MANAS. P.S	Branch:	CSE
USN/Roll No. :	1MS18CS065	Sem/Sec:	III / B
Subject :	Computer Org. & Architecture	Subject Code:	

from your computer.

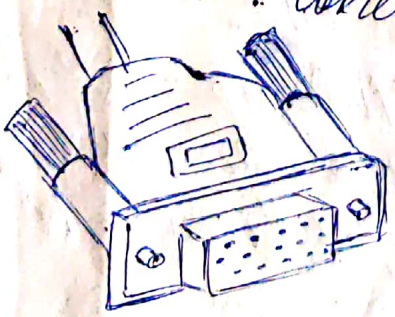
* Open the computer : Remove the cover by a screwdriver or yours fingers.
* Ground yourself : Before touching any electronic component, make sure you first touch an unpainted, grounded metal object to discharge static electricity
* Check the expansion sockets : locate existing memory and expansion sockets on the motherboard. You will need to remove the old card if no new sockets are available.
* Uninstall old RAM : Remove by pushing outward on white ejector tabs.
* Insert new memory : Pick up the card without touching pins or chips. Insert the stick such that it is perpendicular to the motherboard.

- Lock the memory stick: Make sure the small holes on each side fit into the holder. Gently try to pull the stick to ensure locking.
- Install all memories if removed.
- Test and check if the system has detected the new memory. Check the system properties and verify the new RAM size.
- Replace the cover properly.

(4) With a neat diagram, explain different cables used to connect functional units in a ~~cat~~ system.

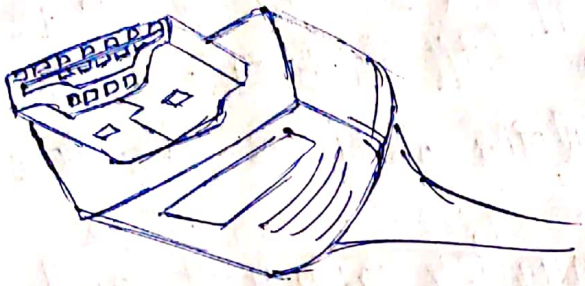
Different cables are:

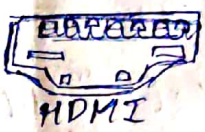
1. VGA cable:
   Also known as D-sub cable, analog video cable. Connect one end to: computer-monitor, television. Connect other end to: VGA port.



2. HDMI cable

Used to transmit display & sound to TV. serves to make TV as an external monitor with sound speakers on TV.

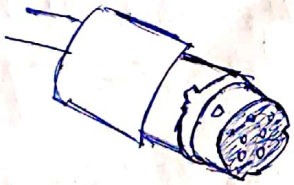


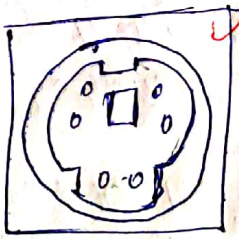


3. PS/2 cable

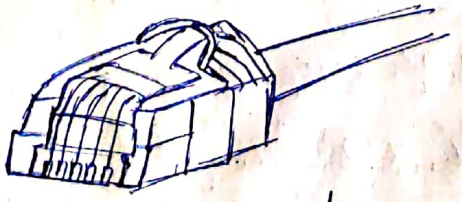
Purple PS/2 port : Keyboard

Green PS/2 port : Mouse



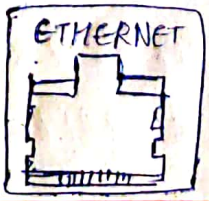


4. Ethernet (RJ45 cable)



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.

Some safety precautions are:

- Wear proper apparel. Avoid acrylic or wool sweaters when working with electronic parts. Do not wear loose fitting clothing, rings, bracelets etc.
- Unplug all computer equipment and peripherals before opening any cases (only exception to this is if you were working without an anti-static mat - keeping cord in would provide a ground).
- Retain all screws during disassembly. Save and sort them in containers such as screw trays, or egg boxes.
- Do not forcefully remove components, as you may end up damaging the ports that attach these components.
- Power supplies produce several voltage levels. Read the supply information carefully before you disconnect and ensure it is appropriate for your system.
- Do not damage any case or cover during disassembly, as it may cause problems during refitting of the system and its components.

MANAS.P.S
1MS18CS065

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

Tutorial -II

Programme: B.E
Course: Computer Organization

Term: Jan to May 2020
Course Code: CS45

Name: ~~1MS18CS065~~ MANAS.P.S	**Marks:** 10/10	**Date:** 28/1/2020
USN: 1MS18CS065	**Signature of the Faculty:**	4/2/2020

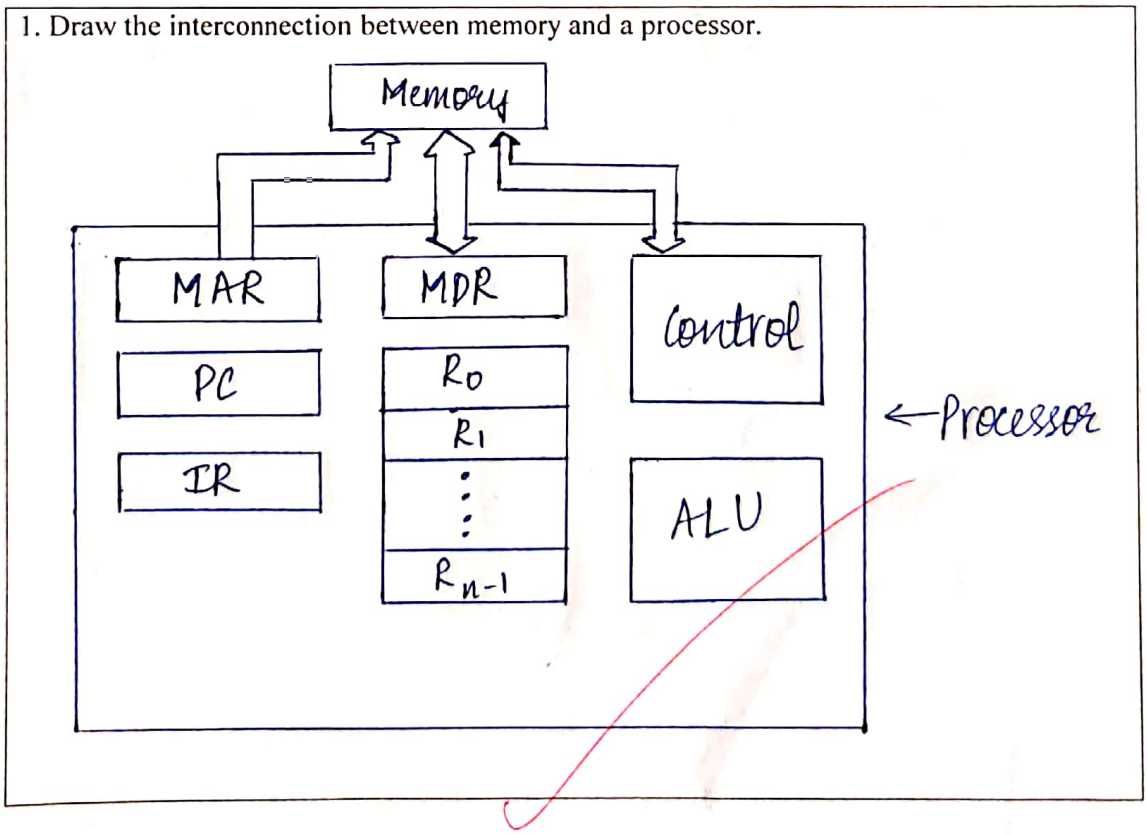
Activity II: Demonstrating Datapath and instruction execution stages using MarieSim Simulator

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. List out the steps required to execute an instruction.

3. Write and execute assembly language program to compute
   i) f=(g+h)*(i+y)
   ii) d=b^2-4ac

4. Describe the factors affecting the performance of a processor

The major factors affecting performance are :

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

Minor factors affecting performance are :

- Elapsed time : Time taken for whole operation
- Processor time : Time during which processor is active (T= NXS/R)
- Processor clock : Higher the clock rate, higher the no. of basic steps per clock cycle.

3. Results and Snapshots:

Attached

RAMAIAH
Institute of Technology

Data Sheet

MARKS :

Name :	MANAS. P. S	Branch:	CSE
USN/Roll No. :	1MS18CS065	Sem/Sec:	IV 'B'
Subject :	Computer Organization & Architecture	Subject Code:	CS45

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

The steps required to execute an instruction are :

- Fetching an instruction : The instruction is fetched from the main memory. The instruct at the current program counter (PC) will be fetched and stored in Instruction register (IR).
- Decode instruction : During this cycle the encoded instruction is interpreted by the decoder int the IR.
- ALU operation : The two operands in the instruction will be operated on given operator in the instructions.
- Access memory : Only 2 kinds of instructions access memory, LOAD copies a value from memory to the Accumulator (AC) and STORE copies a register value to memory.
- Update Register File : Result of the ALU is written back to the register file to update it.
- Update the PC : We need to update the PC to the address of the next instruction, so that we can go back to step1 where the CPU will fetch instruction.

(3) i] Write and execute the assembly language program to compute : $f = (g+h) / (i+y)$

```
        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
```

ii] $d = b^2 - 4ac$

P.T.O.

```
LOAD B
STORE 0
FIRST, LOAD B
       ADD X
       STORE X
       LOAD 0
       SUBT ONE
       STORE 0
       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 0
ADD X
SUBT Z
OUTPUT
HALT
A, DEC 6
B, DEC 3
C, DEC 2
0, DEC 0
X, DEC 0
Y, DEC 0
Z, DEC 0
D, DEC 0
ONE, DEC 1
FOUR, DEC 4
```

IMS18CS065
IV-B
MANAS.P.S

MARIE Assembler Code Editor

File Edit Assemble Help

```
LOAD G
ADD H
STORE G
LOAD I
ADD Y
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.

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

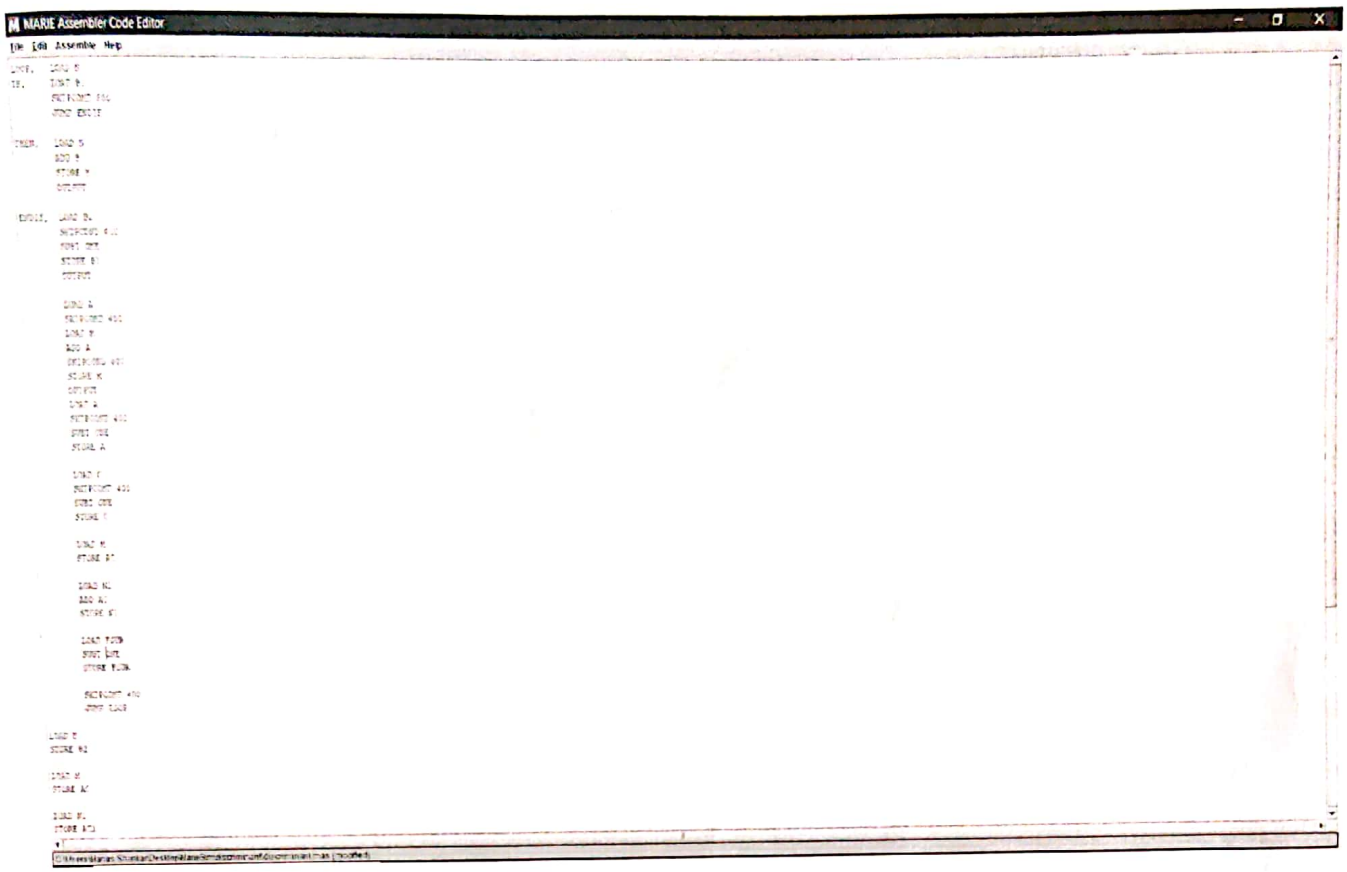
Assembly Listing for multAdd.mas

```
000 1013 |        LOAD G
001 2014 |        ADD H
002 2010 |        STORE G
003 1015 |        LOAD I
004 2016 |        ADD Y
005 2015 |        STORE I
006 1013 |        LOAD G
007 1012 | LOOP   LOAD N
008 2013 |        ADD G
009 2012 |        STORE N
         |
00A 1015 |        LOAD I
00B 4018 |        SUBT ONE
00C 2015 |        STORE I
         |
00D 8400 |        SKIPCOND 400
00E 9007 |        JUMP LOOP
         |
00F 1012 |        LOAD N
010 2017 |        STORE F
011 7000 |        HALT
012 0000 | N      DEC 0
013 0001 | G      DEC 1
014 0001 | H      DEC 1
015 0002 | I      DEC 2
016 0002 | Y      DEC 2
017 0000 | F      DEC 0
018 0001 | ONE    DEC 1

Assembly successful.

        SYMBOL TABLE
-------------------------------------------------
 Symbol | Defined | References
-------------------------------------------------
 F      |  017    | 010
 G      |  013    | 000, 002, 00A, 013
 H      |  014    | 001
 I      |  015    | 003, 005, 00B, 00C
 LOOP   |  007    | 00E
 N      |  012    | 007, 009, 00F
 ONE    |  018    | 00B
 Y      |  016    | 004
-------------------------------------------------
```

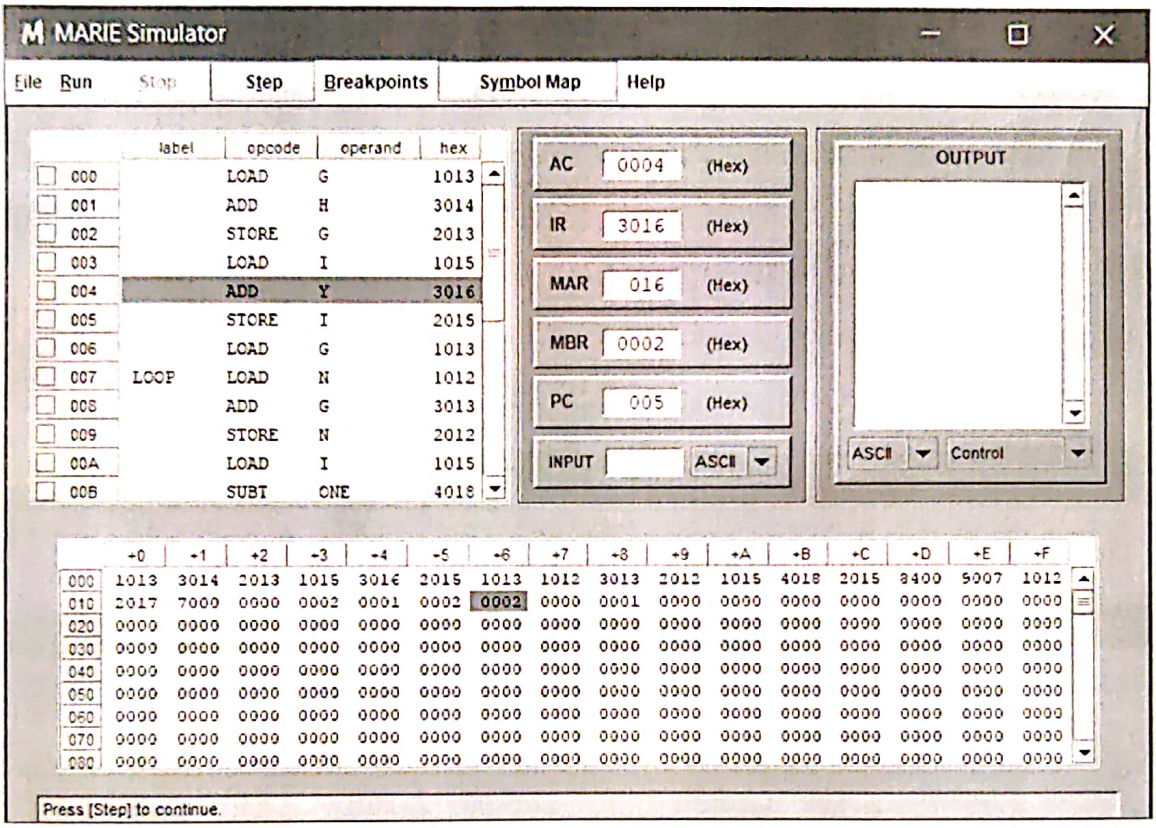
Print

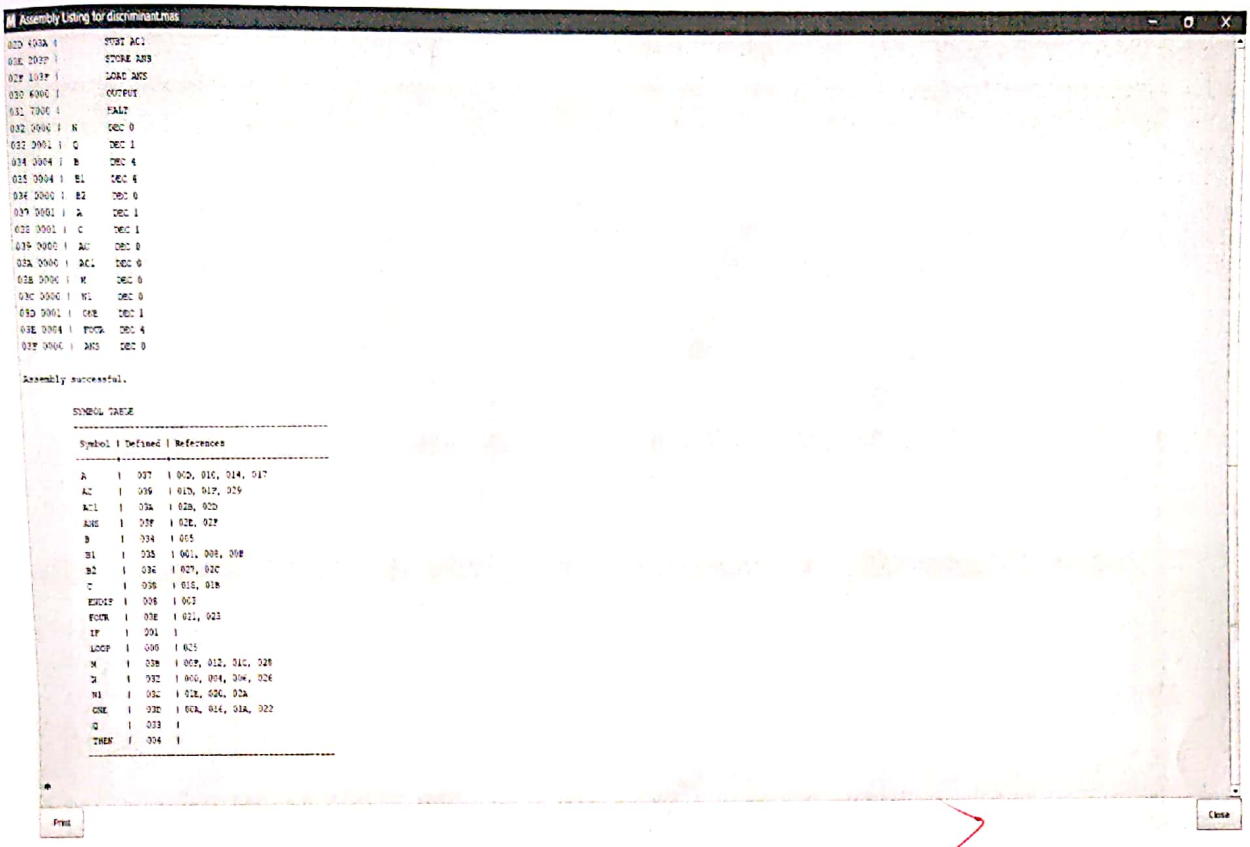
Close

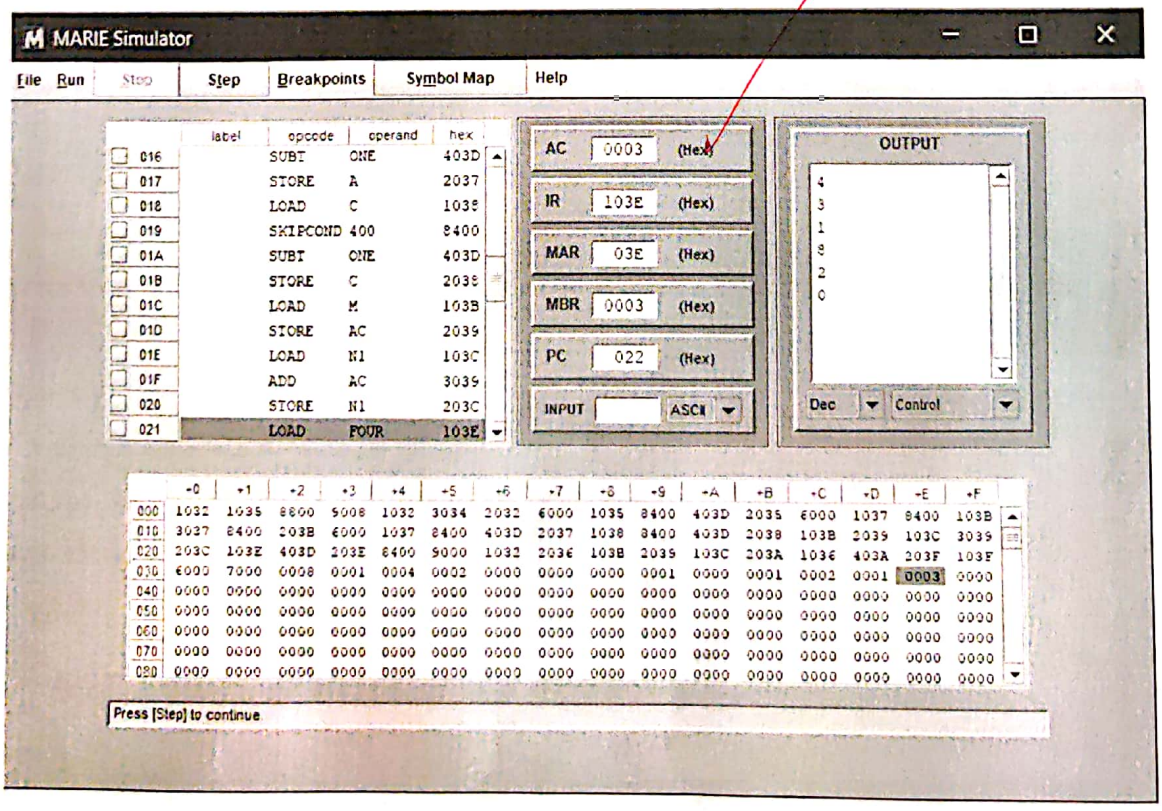
1MS18CS065
IV - B
MANAS. P.S

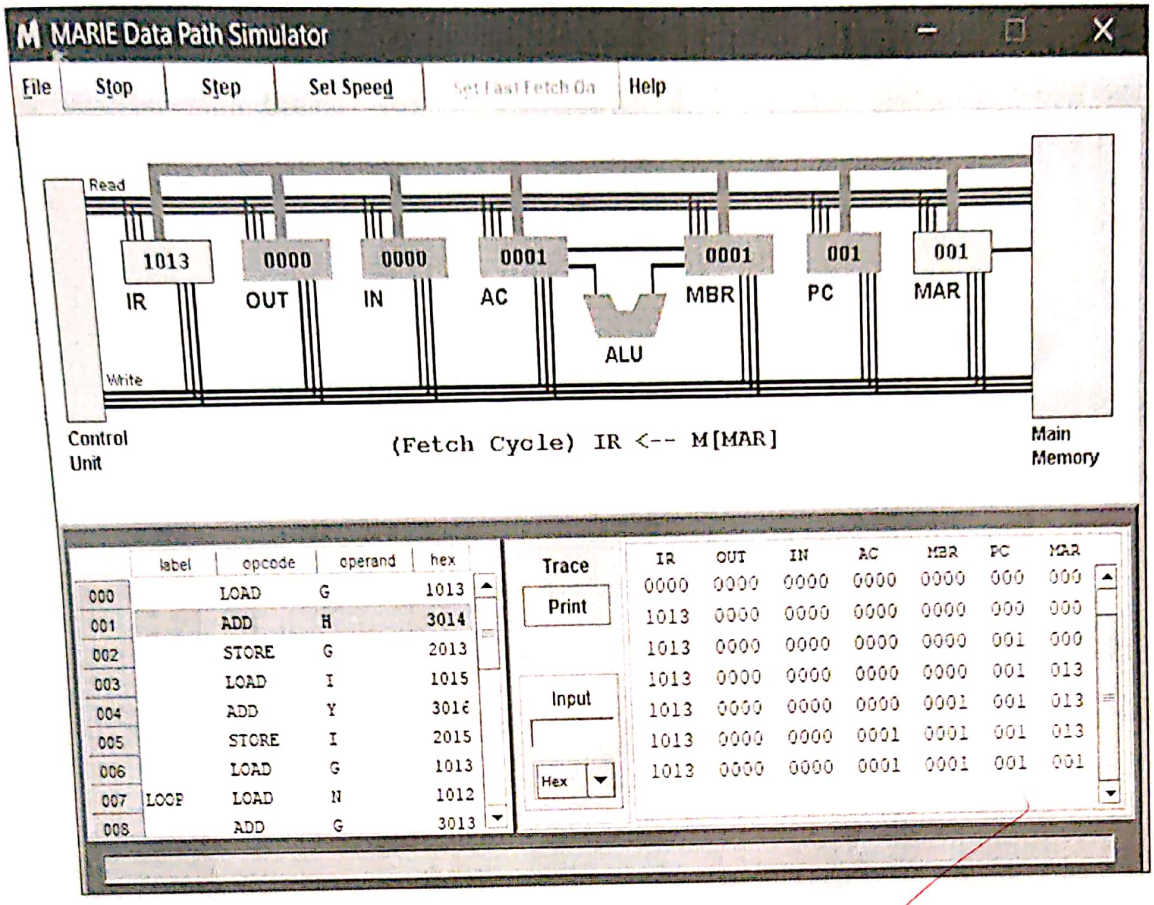


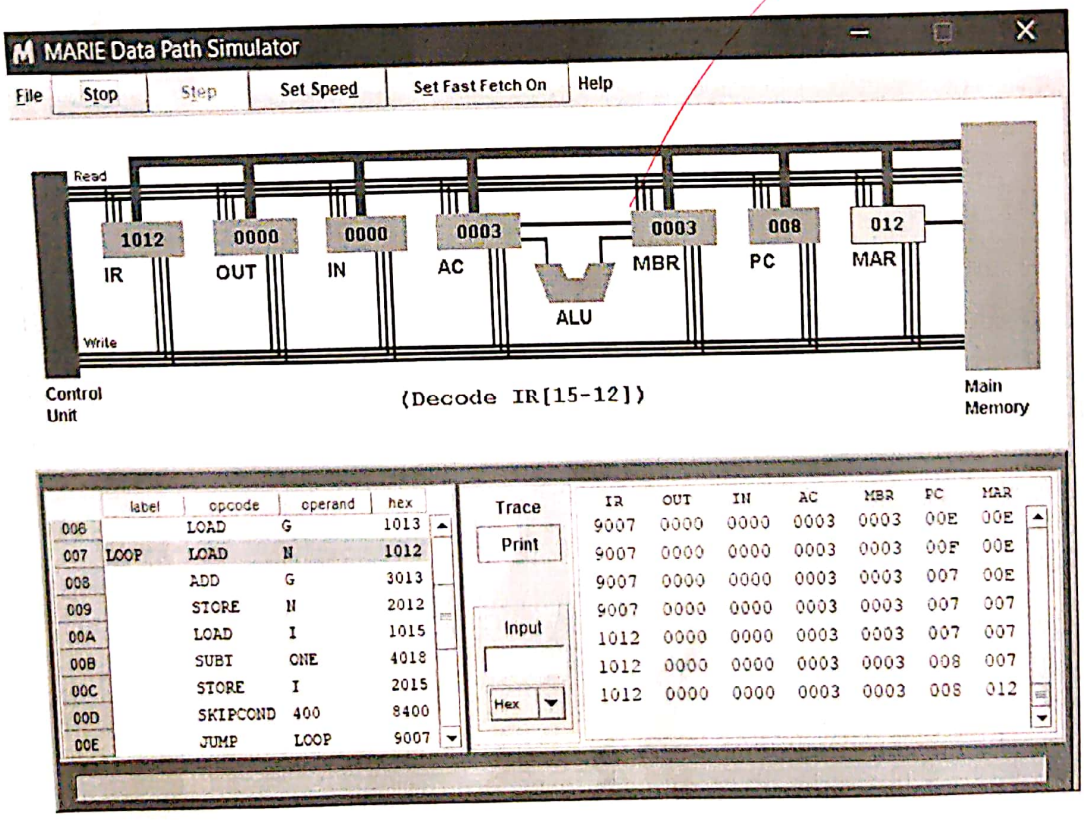
$d = b^2 - 4 * a * c$

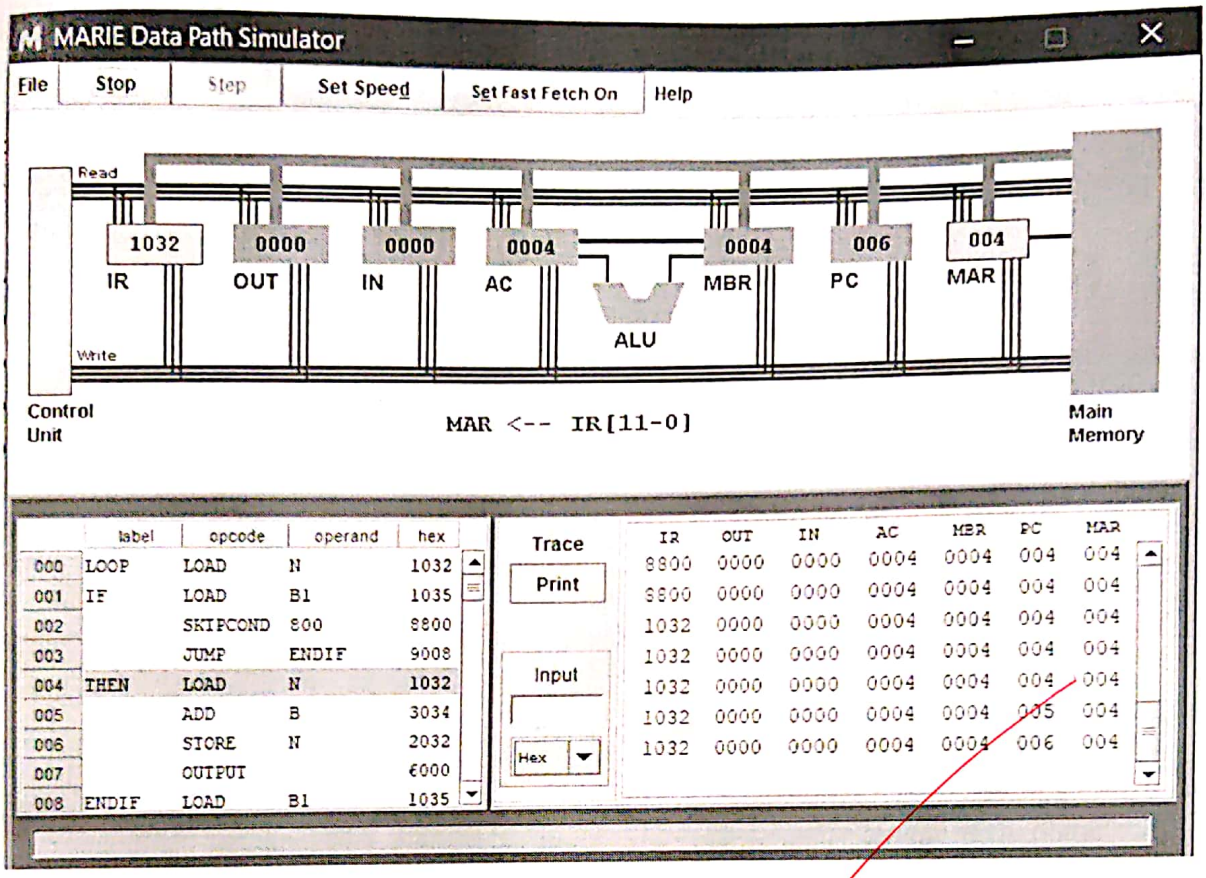












Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of CSE

Tutorial -III

Programme: B.E
Course: Computer Organization Course Code: CS45

Term: Jan to May 2020

Name: MANAS. P.S	**Marks:** 10/10	**Date:** 4/2/2020
USN: 1MS18CS065	**Signature of the Faculty:**	10/2/2020

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:

RAMAIAH
Institute of Technology

Data Sheet

MARKS :

Name :	MANAS.P.S	Branch:	CSE
USN/Roll No. :	1MS18CS065	Sem/Sec:	IV 'B'
Subject :	Computer Organization and Architecture	Subject Code:	CS45

(1) Write an ARM program to perform basic arithmetic operations

```
MOV R3, #24
MOV R4, #25
LDR R1, = 0x00000064
LDR R2, = 0x00000068

STR R3, [R1]
STR R4, [R2]

LDR R5, [R1]
LDR R6, [R2]

ADD R0, R5, R6      ; Basic arithmetic operations
MUL R0, R5, R6
SUB R0, R5, R6

SWI 0x11
```

(2) Write an ARM program to perform and demonstrate working of load and store instructions.

```
MOV R3, #36      ; Values to store
MOV R4, #37
```

```
LDR  R1,=0x00000074      ; load address into
LDR  R2,=0x00000078      ; register

STR  R3,[R1]             ; store R3 and R4 at address in
STR  R4,[R2]             ; R1 and R2

LDR  R5,[R1]             ; load from R1,R2 to R5,R6
LDR  R6,[R2]

ADD  R0,R5,R6
SUB  R0,R5,R6
MUL  R0,R5,R6

SWI  0x11
```

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

```
MOV R3,#41               ; value of g
MOV R4,#45               ; value of h
MOV R7,#35               ; value of i
MOV R8,#38               ; value of j
LDR R1,=0x00000064
LDR R2,=0x00000068
LDR R9,=0x00000074
LDR R10,=0x00000078

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

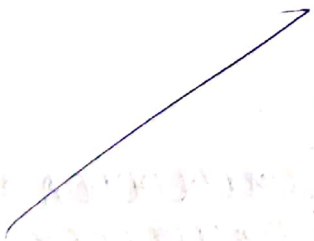
```
LDR R5,[R1]
LDR R6,[R2]
LDR R11,[R9]
LDR R12,[R10]

ADD R5,R5,R6     ; g+h
ADD R11,R11,R12  ; i+j

SUB R0,R5,R11    ; (g+h)-(i+j), store result in R0

SWI 0x11
```

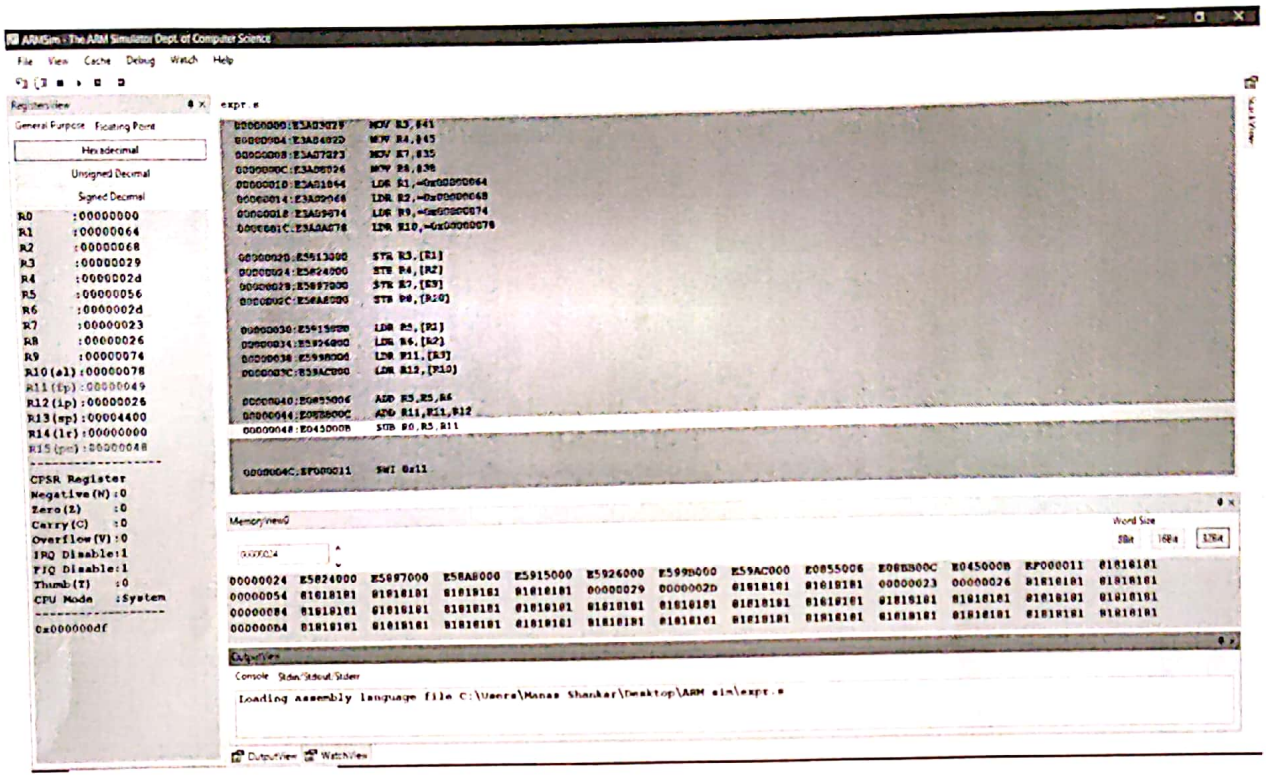
(4) Write an ARM program to find sum of all elements in an array

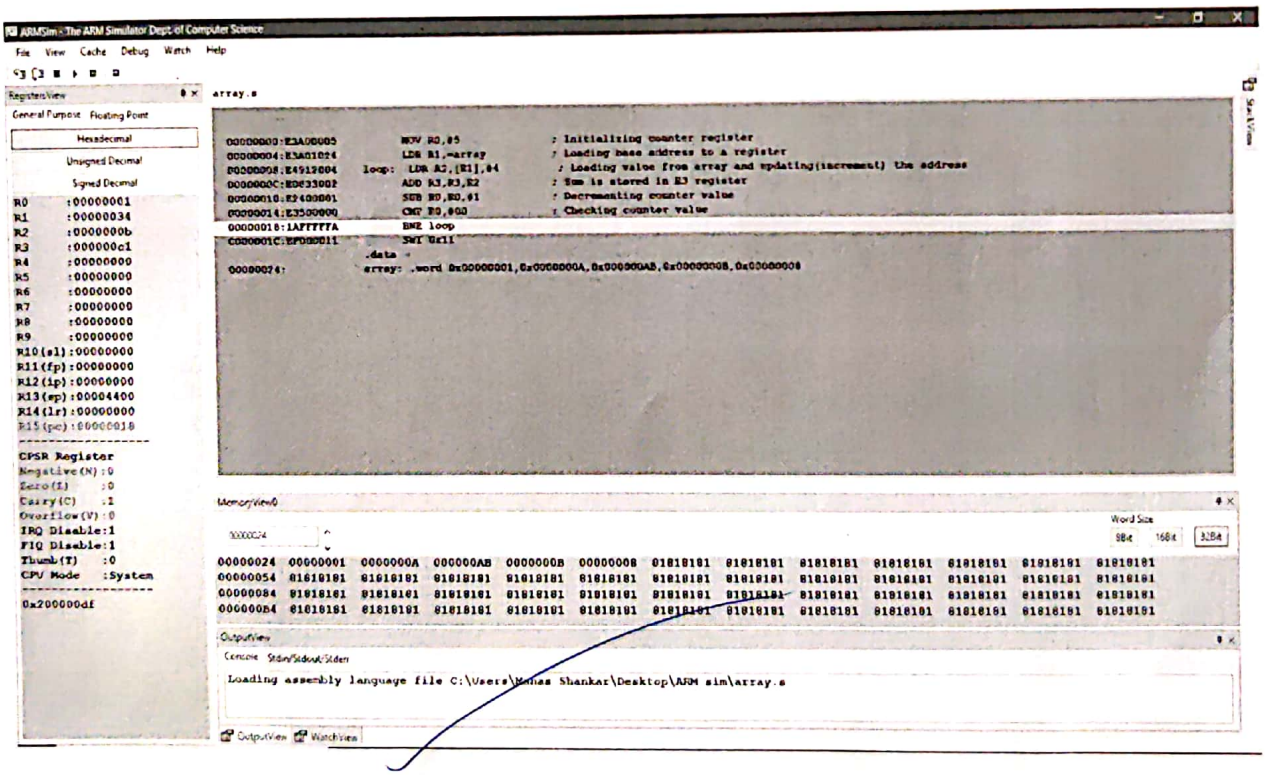
```
MOV R0,#5       ; initialize counter register
LDR R1,=array   ; load base address
loop: LDR R2,[R1],#4  ; load array val and increment
      ADD R3,R3,R2    ; address. Sum stored in R3.
      SUB R0,R0,#1    ; Decrement counter value
      CMP R0,#0       ; checking counter value
      BNE loop
      SWI 0x11
.data
array: .word 0x00000001,0x0000000A,0x000000AB,
             0x0000000B,0x00000008
```

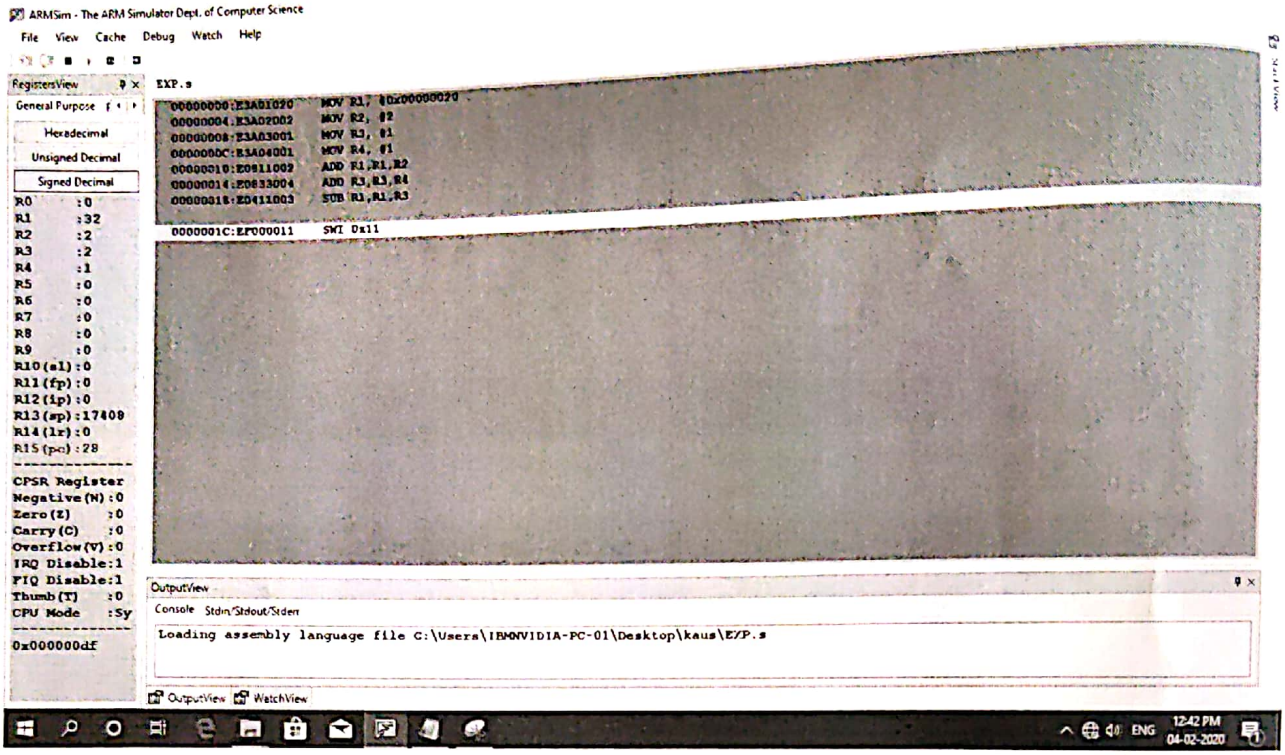
(5) Write an ARM program to find the factorial of a number.

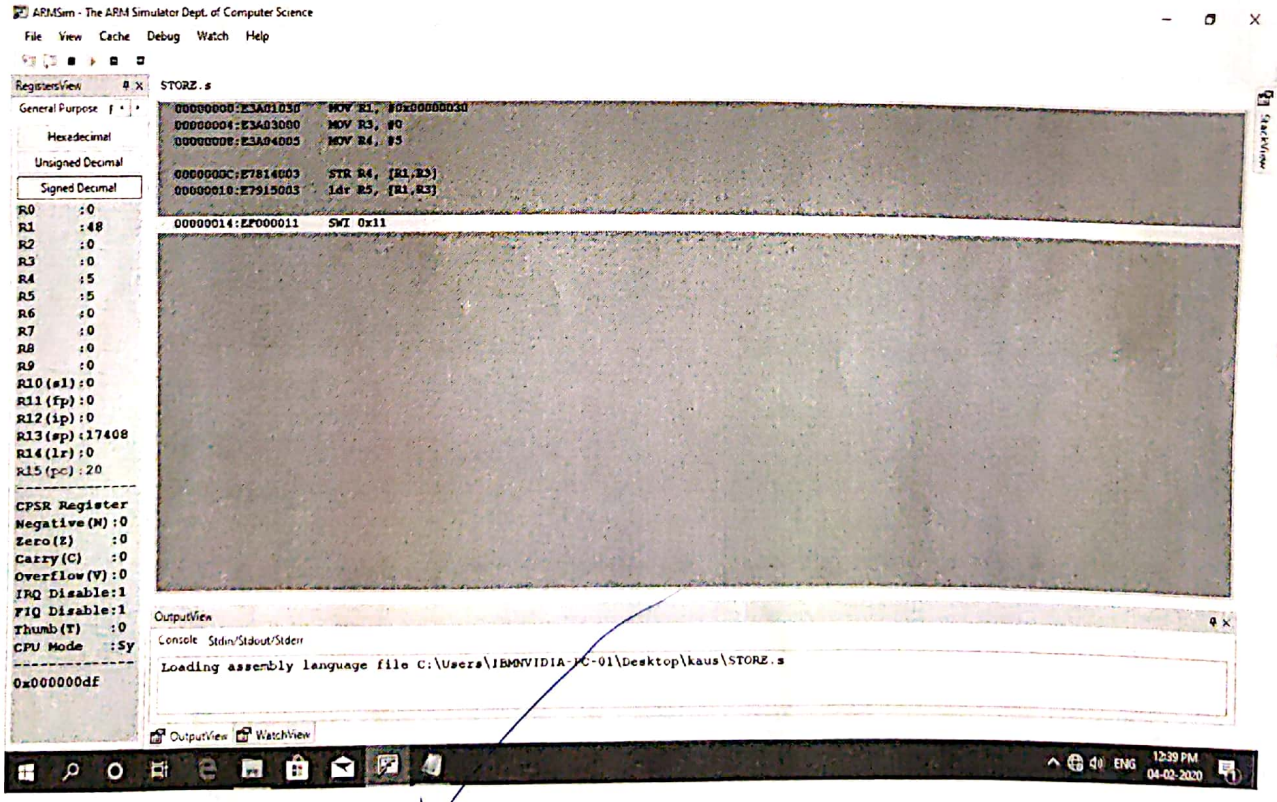
```
LDR R1,=fact   ; load base address of variables
LDR R2,[R1]    ; load value at [R1] into R2 and R3
LDR R3,[R1]
loop : SUB R2,R2,#1    ; decrement R2 value
       CMP R2,#0       ; break to end if R2==0
       BEQ end
       MUL R0,R2,R3    ; calculate product in current
       MOV R3,R0       ; iteration, store into R3 for
       CMP R2,#0       ; updation. check R2 value.
       BNE loop        ; break if equal to zero
end :  SWI 0x11
.data
fact: .word 0x00000005    ; variable with value 5
```

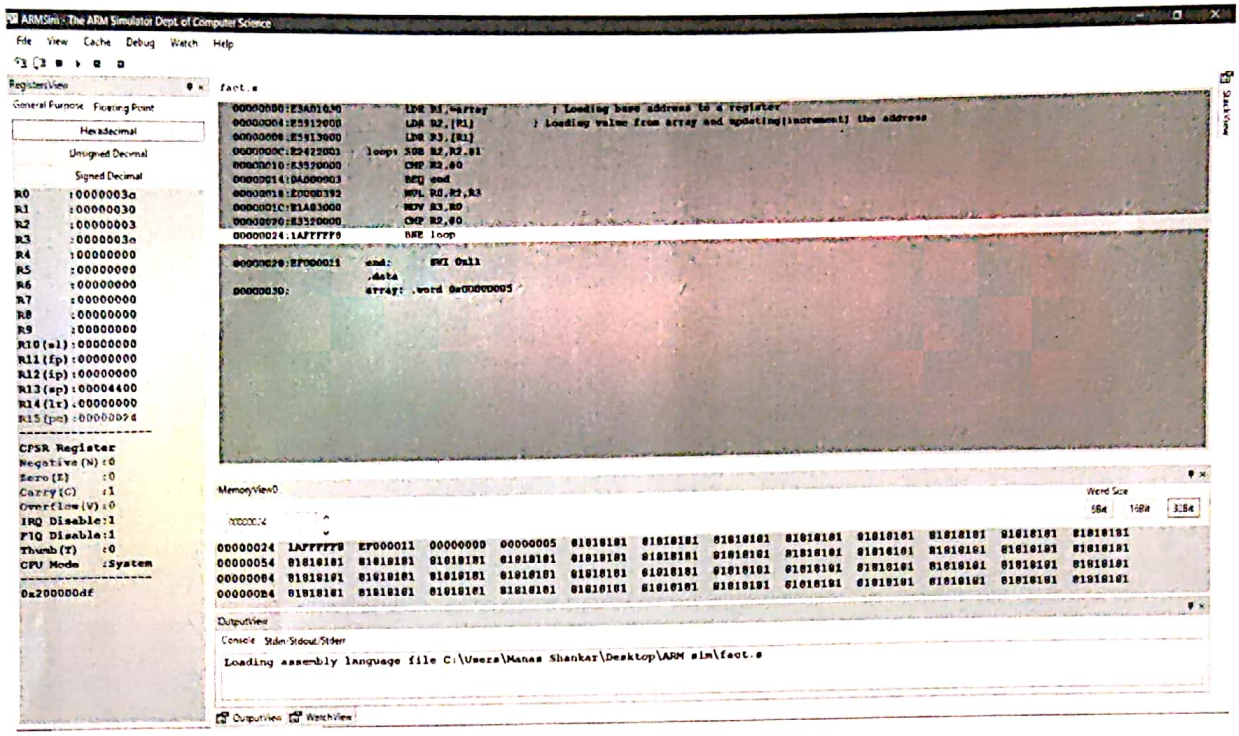












Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)

Department of CSE

Programme: B.E
Course: Computer Organization

Term: Jan to May 2019
Course Code: CS45

Activity IV: Executing ARM programs using ARMsim simulator.

Name: MANAS.P.S	**Marks:** /10	**Date:** 11/2/2020
USN: 1MS18CS065	**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.

Attached in datasheet

Results/Conclusions and Snapshots: Take the snap shot of registers file and memory view

Attached in datasheet

RAMAIAH
Institute of Technology

Data Sheet

MARKS :

Name :	MANAS. P.S	Branch:	CSE
USN/Roll No. :	1MS18CS065	Sem/Sec:	IV 'B'
Subject :	Computer Organization and Architecture	Subject Code:	CS45

(1) Write an ARM program to generate Fibonacci series

```
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,#4
      add r3,r3,#1
      cmp r3,r2
      blt loop
swi 0x22
swi 0x11
```

(2) Write an ARM program to search an element in an array and print y if found and print n if not found

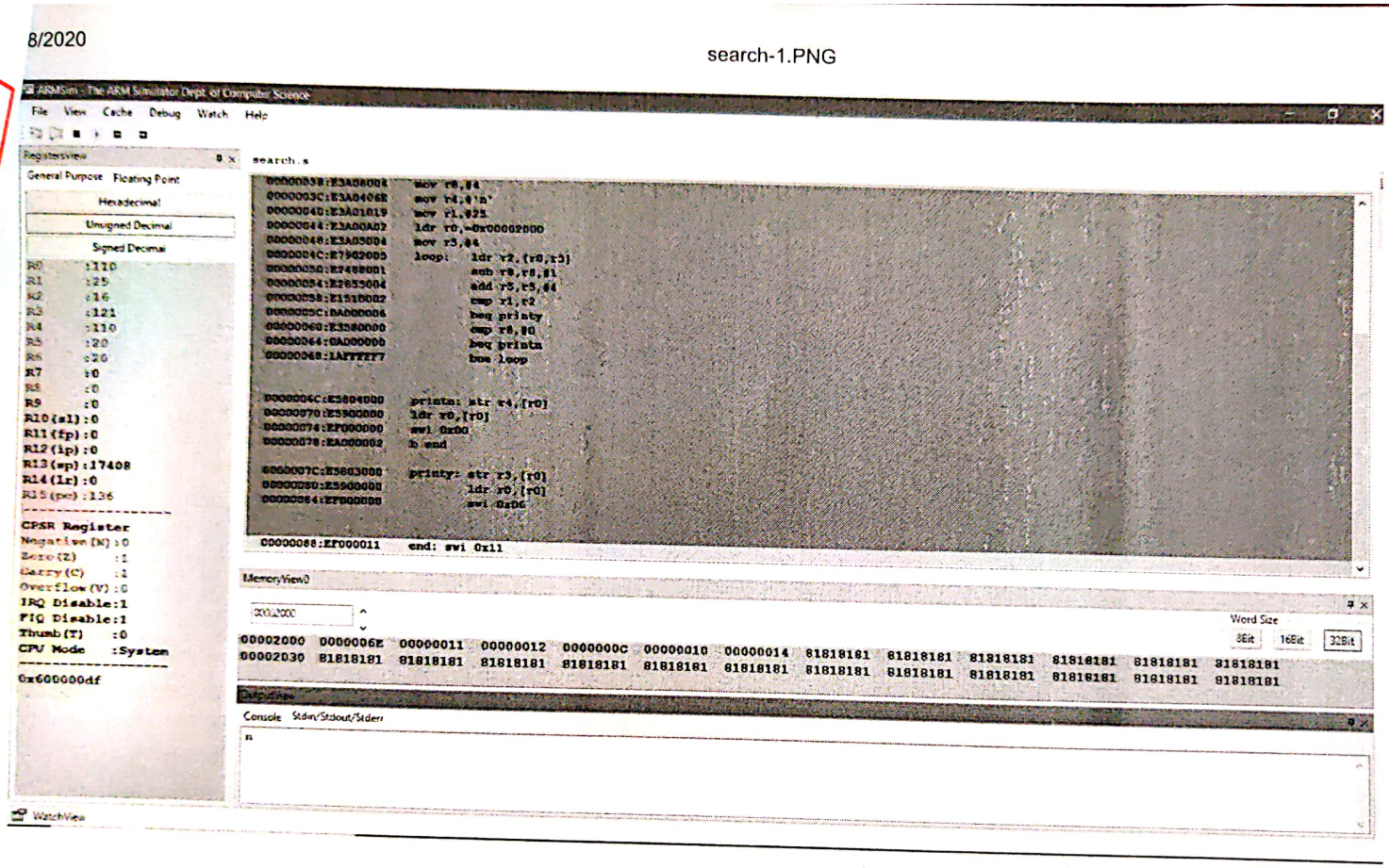
```
ldr r0,=0x00002000
mov r1,#14
mov r2,#17
mov r3,#18
mov r4,#12
mov r5,#16
mov r6,#20
str r1,[r0]
str r2,[r0,#4]
str r3,[r0,#8]
str r4,[r0,#12]
str r5,[r0,#16]
str r6,[r0,#20]
mov r3,#'y'
mov r8,#4
mov r4,#'n'
mov r1,#16
ldr r0,=0x00002000
mov r5,#4
loop: ldr r2,[r0,r5]
      sub r8,r8,#1
      add r5,r5,#4
      cmp r1,r2
      beq printy
      cmp r8,#0
      beq printn
      bne loop
printn: str r4,[r0]
        ldr r0,[r0]
        swi 0x00
        b end
printy: str r3,[r0]
        ldr r0,[r0]
        swi 0x00
end: swi 0x11
```

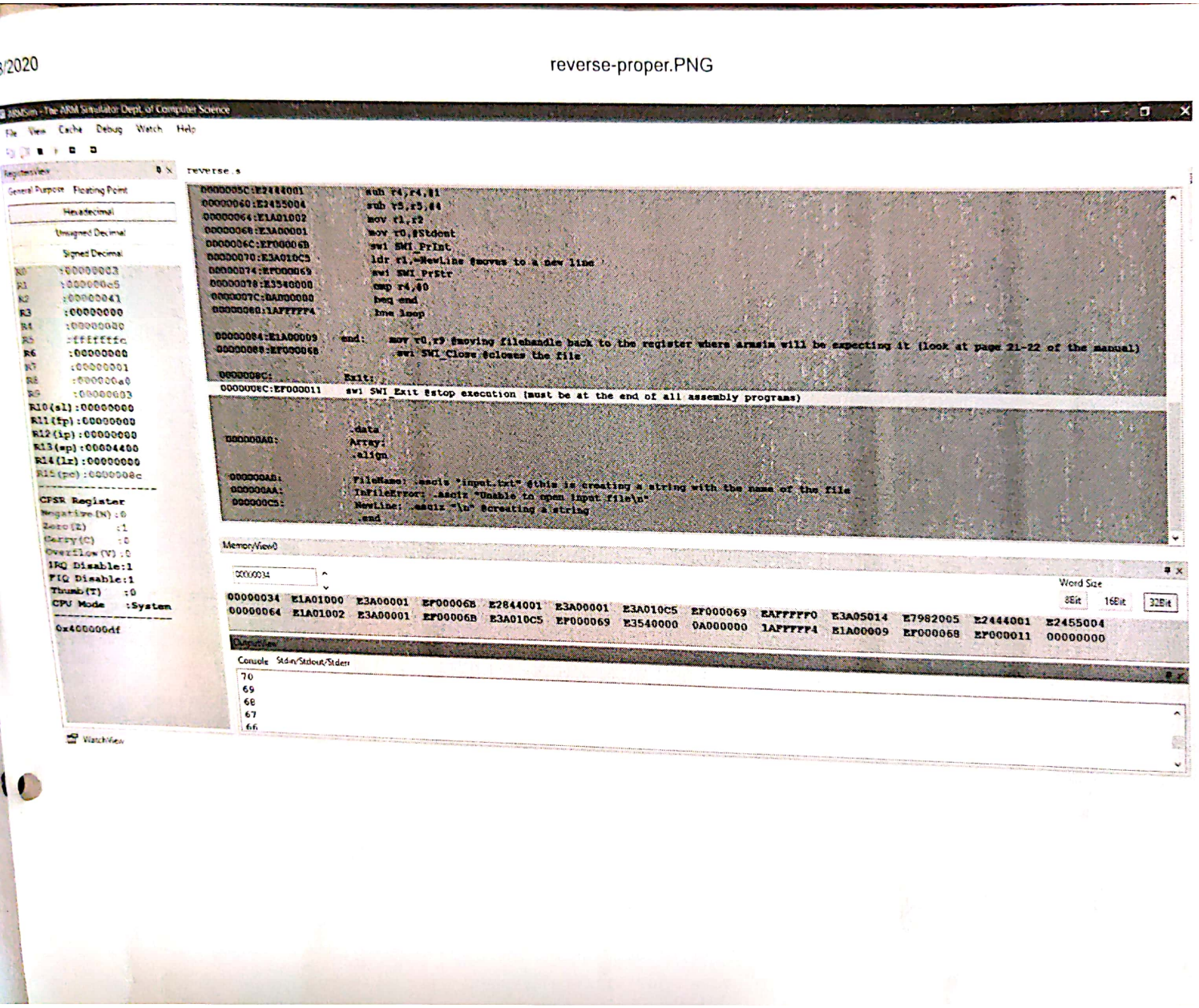
(3) Write an ARM program to find the length of a string and copying one string to another

```
.equ SWI_Open, 0x66
.equ SWI_Close, 0x68
.equ SWI_PrInt, 0x6b
.equ SWI_RdInt, 0x6c
.equ Stdout, 1
.equ SWI_PrStr, 0x69
.equ SWI_Exit, 0x11
.global _start
.text

_start:
ldr r0, =FileName
mov r1, #0
swi SWI_Open
bcs Exit
mov r9, r0
mov r5, #0

loopstart:
mov r7, #Stdout
mov r0, r9
ldr r8, =Array
swi SWI_RdInt
bcs afterloop
str r0, [r8, r5]
add r5, r5, #4
mov r1, r0
mov r0, #Stdout
mov r1, r0
swi SWI_PrInt
add r4, r4, #1
mov r0, #Stdout
ldr r1, =NewLine
swi SWI_PrStr
bal loopstart

afterloop:
mov r5, #20
loop: ldr r2, [r8, r5]
      sub r4, r4, #1
      sub r5, r5, #4
      mov r1, r2
      mov r0, #Stdout
      swi SWI_PrInt
      ldr r1, =NewLine
      swi SWI_PrStr
      cmp r4, #0
      beq end
      bne loop
end: mov r0, r9
     swi SWI_Close
```

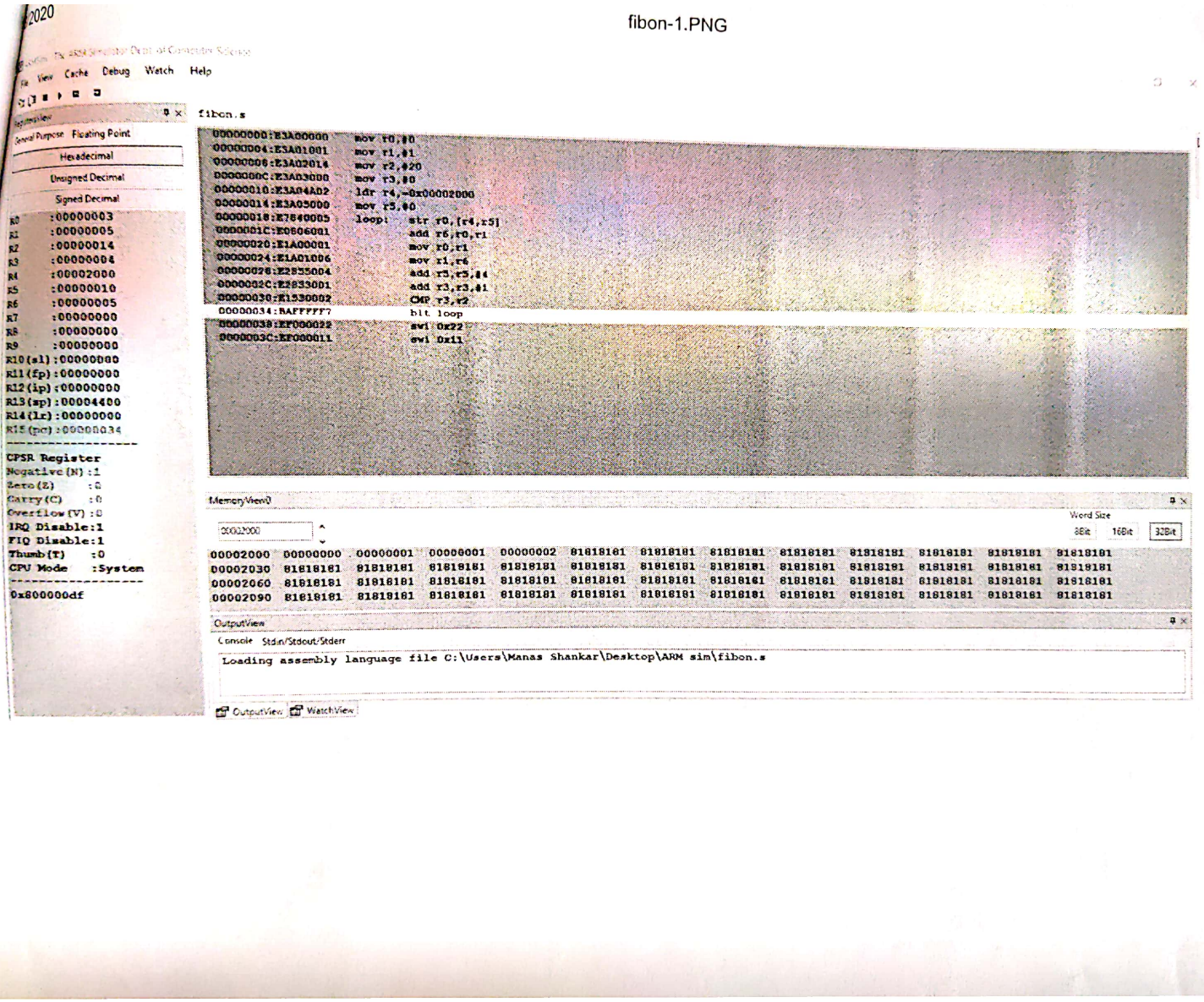
```
Exit :
swi SWI_Exit

.data
Array :
.align
FileName : .asciz "input.txt"
InFileError : .asciz "Unable to open input file \n"
Newline : .asciz "\n"
.end
```

search-1.PNG





fibon-1.PNG



Lab-5

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

Department of CSE

Programme: B.E
Course: Computer Organization

Term: Jan to May 2019
Course Code: CS45

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

Name: MANAS. P.S	**Marks:** /10	**Date:**
USN: 1MS18CS065	**Signature of the Faculty:**	

Objective: To simulate the working of Arithmetic and Logical Unit using simulator.

Simulator Description: Logisim is an educational tool for designing and simulating digital logic circuits. With its simple toolbar interface and simulation of circuits as you build them, it is simple enough to facilitate learning the most basic concepts related to logic circuits. With the capacity to build larger circuits from smaller sub circuits, and to draw bundles of wires with a single mouse drag, Logisim can be used (and is used) to design and simulate entire CPUs for educational purposes.

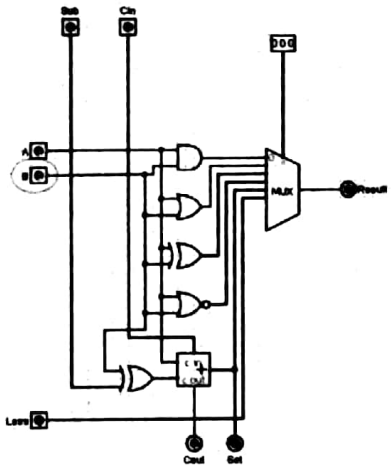
Activity to be performed by students:

List out the steps in designing ALU

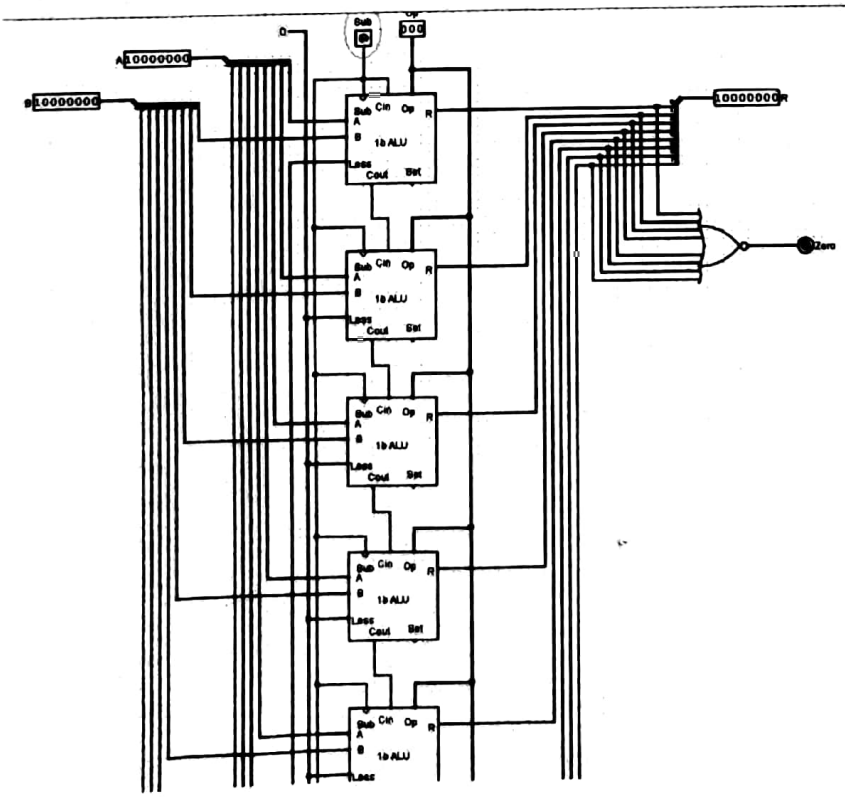
Attached in Datasheet

Snapshots :

1-bit Arithmetic Logic Unit (ALU)



8-bit Arithmetic Logic Unit (ALU).



RAMAIAH Institute of Technology

Lab-5

Data Sheet

MARKS :

Name :	MANAS P S	Branch:	CSE
USN/Roll No. :	1MS18CS065	Sem/Sec:	IV 'B'
Subject :	Computer Organization and Architecture	Subject Code:	CS45

Activity V : Designing an ALU to perform arithmetic and logical functions using logisim 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 multiplexer 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_{in} 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.

P.T.O.

10. Add another i/p and name it less. Connect it to the MUX.

11. Add an output pin and name it Set, connect it to the o/p of adder unit.

Snapshots:

Attached

Lab-6

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

Department of CSE

Programme: B.E **Term: Jan to May 2019**
Course: Computer Organization **Course Code: CS45**

Activity VI: Designing memory system using Logisim simulator.

Name: MANAS.P.S	**Marks: /10**	**Date:**
USN: 1MS18CS065	**Signature of the Faculty:**	

Objective: To simulate the writing operation on memory.

Simulator Description: Logisim is an educational tool for designing and simulating digital logic circuits. With its simple toolbar interface and simulation of circuits as you build them, it is simple enough to facilitate learning the most basic concepts related to logic circuits. With the capacity to build larger circuits from smaller sub circuits, and to draw bundles of wires with a single mouse drag, Logisim can be used (and is used) to design and simulate entire CPUs for educational purposes.

Activity to be performed by students:

List out the steps in designing memory system

Attached in Datasheet

RAMAIAH
Institute of Technology

Data Sheet

Lab-6

MARKS :

Name :	MANAS. P. S	Branch:	CSE
USN/Roll No. :	1MS18CS065	Sem/Sec:	IV 'B'
Subject :	Computer Organization and Architecture	Subject Code:	CS45

Activity VI : To simulate writing operation on memory, and hence to design a memory system using logisim simulator.

Objective : To simulate writing operation on memory.

Activity 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 4-bit random number generator, connect Q 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.

Snapshots:

Attached

Observations and Snapshots: