Microprocessor and Computer Architecture

UE23CS251B

4th Semester, Academic Year 2024-25

Date:01/02/2024

Name: Keerthan P.V	SRN: PES2UG23CS272	Section: 4E

Week#____3_______ Program Number: ____1___

Title of the Program

Given a dividend and a divisor ,Write an ALP using ARM7TDMI to find the remainder obtained after division without using DIV instructions.

(Hint: Use repeated subtraction)

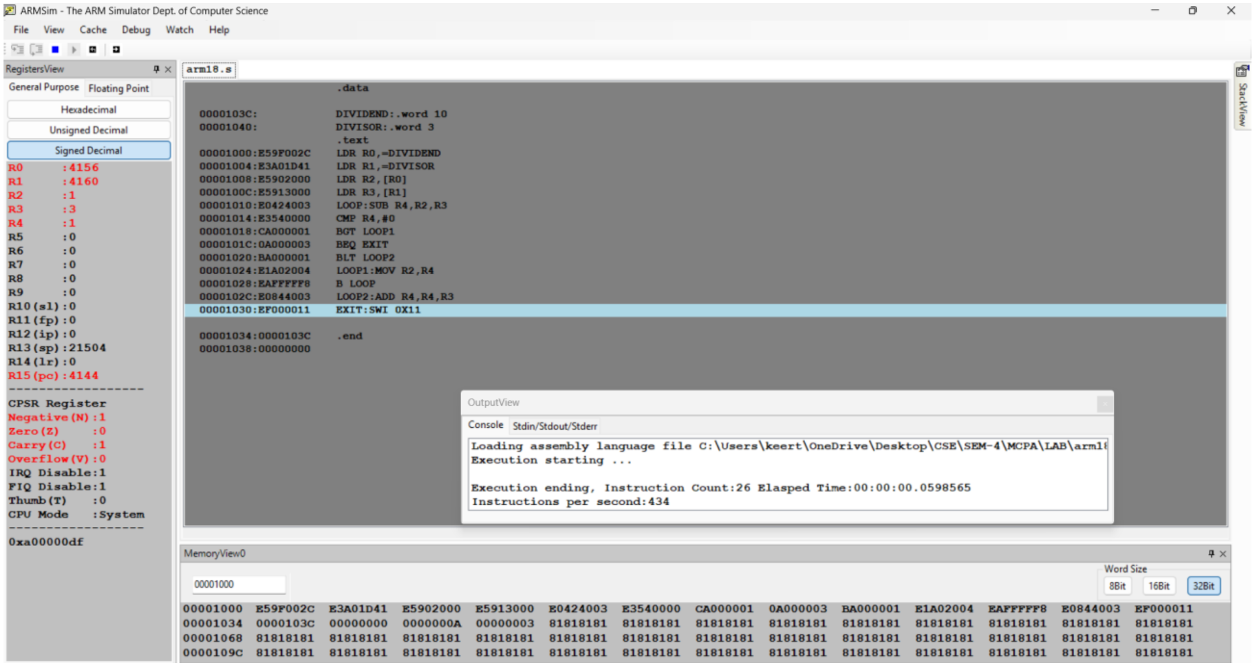
I. ARM Assembly Code(1)

Typed Code to be Included. Screenshot of Code not permitted

```
.data

DIVIDEND:.word 10
DIVISOR:.word 3
.text
LDR R0,=DIVIDEND
LDR R1,=DIVISOR
LDR R2,[R0]
LDR R3,[R1]
LOOP:SUB R4,R2,R3
CMP R4,#0
BGT LOOP1
BEQ EXIT
BLT LOOP2
LOOP1:MOV R2,R4
B LOOP
LOOP2:ADD R4,R4,R3
EXIT:SWI 0X11

.end
```

II. Output Screen Shots (1)

The result should be clearly visible in the screenshots. The screenshot should include the code window, register window, memory window and console window



```
R0        :4156
R1        :4160
R2        :1
R3        :3
R4        :1
R5        :0
R6        :0
R7        :0
R8        :0
R9        :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):4144
------------------
CPSR Register
Negative(N):1
Zero(Z)     :0
Carry(C)    :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)    :0
CPU Mode    :System
------------------
0xa00000df
```

```
                        .data

0000103C:               DIVIDEND:.word 10
00001040:               DIVISOR:.word 3
                        .text
00001000:E59F002C       LDR R0,=DIVIDEND
00001004:E3A01D41       LDR R1,=DIVISOR
00001008:E5902000       LDR R2,[R0]
0000100C:E5913000       LDR R3,[R1]
00001010:E0424003       LOOP:SUB R4,R2,R3
00001014:E3540000       CMP R4,#0
00001018:CA000001       BGT LOOP1
0000101C:0A000003       BEQ EXIT
00001020:BA000001       BLT LOOP2
00001024:E1A02004       LOOP1:MOV R2,R4
00001028:EAFFFFF8       B LOOP
0000102C:E0844003       LOOP2:ADD R4,R4,R3
00001030:EF000011       EXIT:SWI 0X11

00001034:0000103C       .end
00001038:00000000
```

```
Loading assembly language file C:\Users\keert\OneDrive\Desktop\CSE\SEM-4\MCPA\LAB\arml8
Execution starting ...

Execution ending, Instruction Count:26 Elasped Time:00:00:00.0598565
Instructions per second:434
```

```
00001000

00001000  E59F002C  E3A01D41  E5902000  E5913000  E0424003  E3540000  CA000001  0A000003  BA000001  E1A02004  EAFFFFF8  E0844003  EF000011
00001034  0000103C  00000000  0000000A  00000003  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181
00001068  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181
0000109C  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181
```

Week#____3______ Program Number: ____2___

Title of the Program

Write an ALP using ARM7TDMI to search for an element in an array of 16 bit each using Linear search technique

I. ARM Assembly Code(1)

Typed Code to be Included. Screenshot of Code not permitted

a)FOR key in array:

```
.data
A:.hword 1,2,3,4,5,6,7,8,9

.text
LDR R0,=A
MOV R1,#5
MOV R2,#0
MOV R3,#9
LOOP:LDRH R4,[R0]
ADD R0,R0,#2
CMP R1,R4
BEQ LOOP1
SUB R3,R3,#1
CMP R3,#0
BNE LOOP
B EXIT
```

```
LOOP1:ADD R2,R2,#1
EXIT:SWI 0X11

.end
```

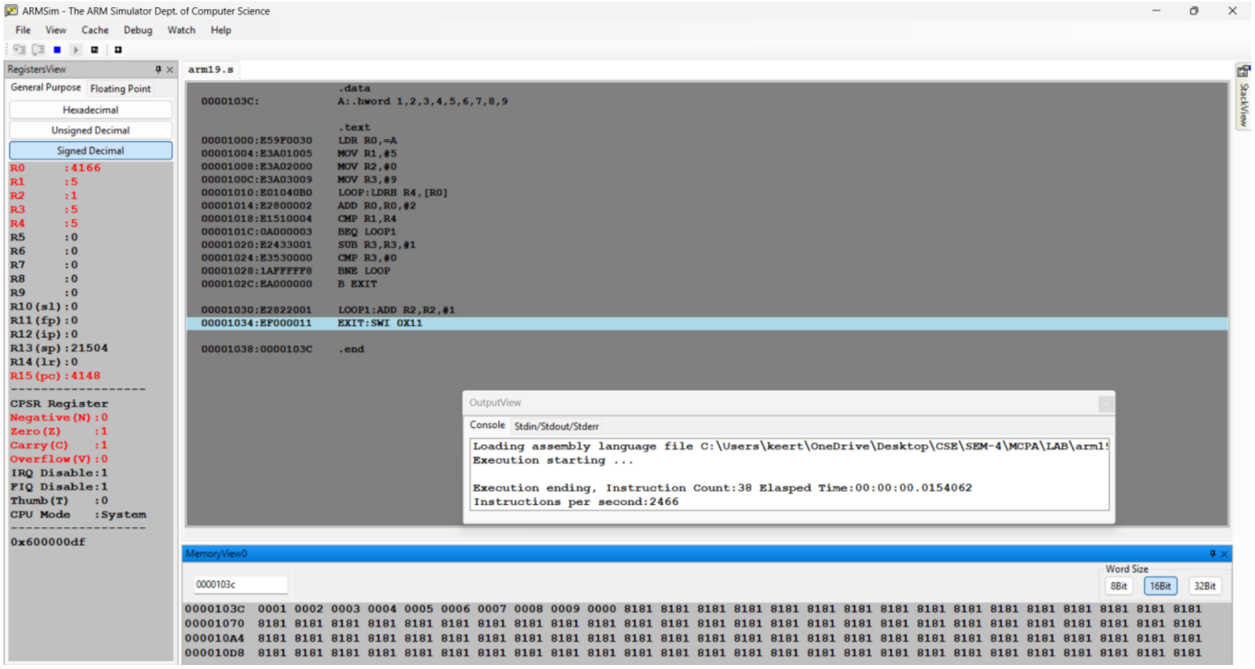
b)For key not in array:

```
.data

A:.hword 1,2,3,4,5,6,7,8,9

.text
LDR R0,=A
MOV R1,#10
MOV R2,#0
MOV R3,#9
LOOP:LDRH R4,[R0]
ADD R0,R0,#2
CMP R1,R4
BEQ LOOP1
SUB R3,R3,#1
CMP R3,#0
BNE LOOP
B EXIT
LOOP1:ADD R2,R2,#1
EXIT:SWI 0X11

.end
```

II. Output Screen Shots (2)

Show the results for both cases KEY FOUND and KEY NOT FOUND. The results should be clearly visible in the screenshots. The screenshot should include the code window, register window, memory window and console window

a)For key in array :



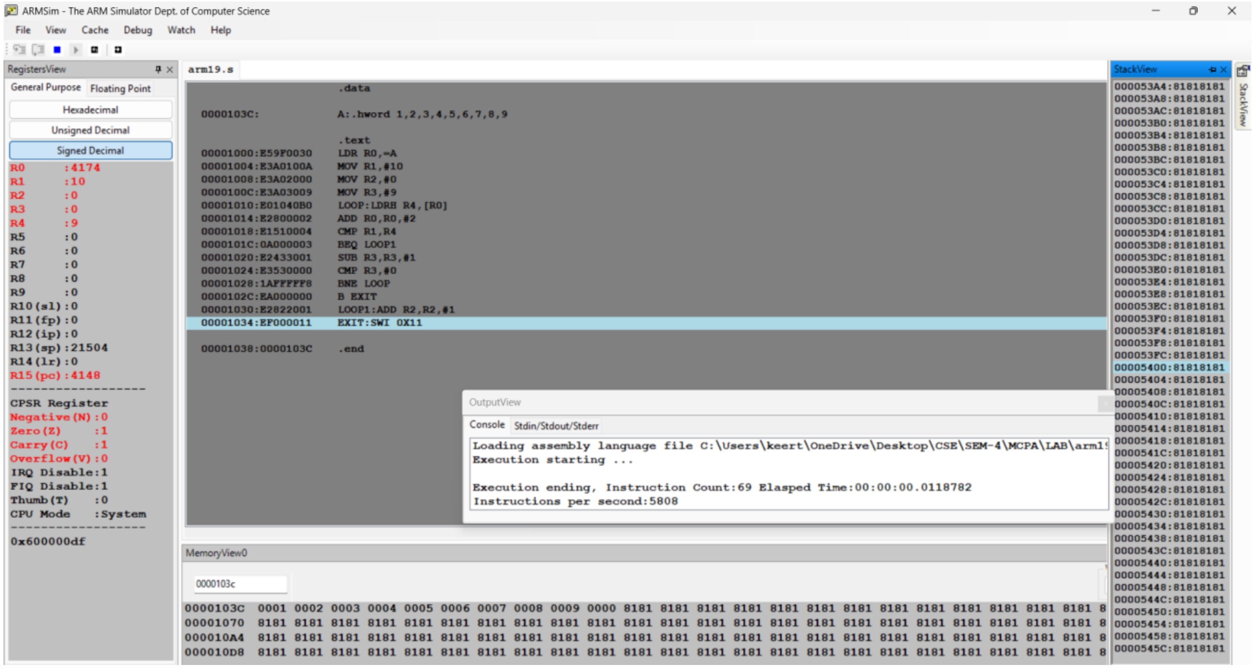
```
R0        :4166
R1        :5
R2        :1
R3        :5
R4        :5
R5        :0
R6        :0
R7        :0
R8        :0
R9        :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):4148
------------------
CPSR Register
Negative(N):0
Zero(Z)     :1
Carry(C)    :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)    :0
CPU Mode    :System
------------------
0x600000df
```

```
                          .data
0000103C:                 A:.hword 1,2,3,4,5,6,7,8,9

                          .text
00001000:E59F0030         LDR R0,=A
00001004:E3A01005         MOV R1,#5
00001008:E3A02000         MOV R2,#0
0000100C:E3A03009         MOV R3,#9
00001010:E01040B0         LOOP:LDRH R4,[R0]
00001014:E2800002         ADD R0,R0,#2
00001018:E1510004         CMP R1,R4
0000101C:0A000003         BEQ LOOP1
00001020:E2433001         SUB R3,R3,#1
00001024:E3530000         CMP R3,#0
00001028:1AFFFFF8         BNE LOOP
0000102C:EA000000         B EXIT

00001030:E2822001         LOOP1:ADD R2,R2,#1
00001034:EF000011         EXIT:SWI 0X11

00001038:0000103C         .end
```

```
Loading assembly language file C:\Users\keert\OneDrive\Desktop\CSE\SEM-4\MCPA\LAB\arm1
Execution starting ...

Execution ending, Instruction Count:38 Elasped Time:00:00:00.0154062
Instructions per second:2466
```

```
0000103c

0000103C  0001 0002 0003 0004 0005 0006 0007 0008 0009 0000 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
00001070  8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
000010A4  8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
000010D8  8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181
```

b)For key not in array:


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
arm19.s
OutputView
Console Stdin/Stdout/Stderr
MemoryView0
StackView

```
R0        :4174
R1        :10
R2        :0
R3        :0
R4        :9
R5        :0
R6        :0
R7        :0
R8        :0
R9        :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):4148
------------------
CPSR Register
Negative(N):0
Zero(Z)     :1
Carry(C)    :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)    :0
CPU Mode    :System
------------------
0x600000df
```

```
                        .data

0000103C:               A:.hword 1,2,3,4,5,6,7,8,9

                        .text
00001000:E59F0030       LDR R0,=A
00001004:E3A0100A       MOV R1,#10
00001008:E3A02000       MOV R2,#0
0000100C:E3A03009       MOV R3,#9
00001010:E01040B0       LOOP:LDRH R4,[R0]
00001014:E2800002       ADD R0,R0,#2
00001018:E1510004       CMP R1,R4
0000101C:0A000003       BEQ LOOP1
00001020:E2433001       SUB R3,R3,#1
00001024:E3530000       CMP R3,#0
00001028:1AFFFFF8       BNE LOOP
0000102C:EA000000       B EXIT
00001030:E2822001       LOOP1:ADD R2,R2,#1
00001034:EF000011       EXIT:SWI 0X11

00001038:0000103C       .end
```

```
Loading assembly language file C:\Users\keert\OneDrive\Desktop\CSE\SEM-4\MCPA\LAB\arm1
Execution starting ...

Execution ending, Instruction Count:69 Elasped Time:00:00:00.0118782
Instructions per second:5808
```

```
0000103c

0000103C  0001 0002 0003 0004 0005 0006 0007 0008 0009 0000 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8
00001070  8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8
000010A4  8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8
000010D8  8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8181 8
```

```
000053A4:81818181
000053A8:81818181
000053AC:81818181
000053B0:81818181
000053B4:81818181
000053B8:81818181
000053BC:81818181
000053C0:81818181
000053C4:81818181
000053C8:81818181
000053CC:81818181
000053D0:81818181
000053D4:81818181
000053D8:81818181
000053DC:81818181
000053E0:81818181
000053E4:81818181
000053E8:81818181
000053EC:81818181
000053F0:81818181
000053F4:81818181
000053F8:81818181
000053FC:81818181
00005400:81818181
00005404:81818181
00005408:81818181
0000540C:81818181
00005410:81818181
00005414:81818181
00005418:81818181
0000541C:81818181
00005420:81818181
00005424:81818181
00005428:81818181
0000542C:81818181
00005430:81818181
00005434:81818181
00005438:81818181
0000543C:81818181
00005440:81818181
00005444:81818181
00005448:81818181
0000544C:81818181
00005450:81818181
00005454:81818181
00005458:81818181
0000545C:81818181
```

Week#____3_______ Program Number: ____3___

Title of the Program

Write an ALP using ARM7TDMI to copy a block 128 bytes of data from location A to location B if the rate of data transfer rate is 16 bytes, LDM and STM instructions.

and

For the same transfer the block with auto-indexing.

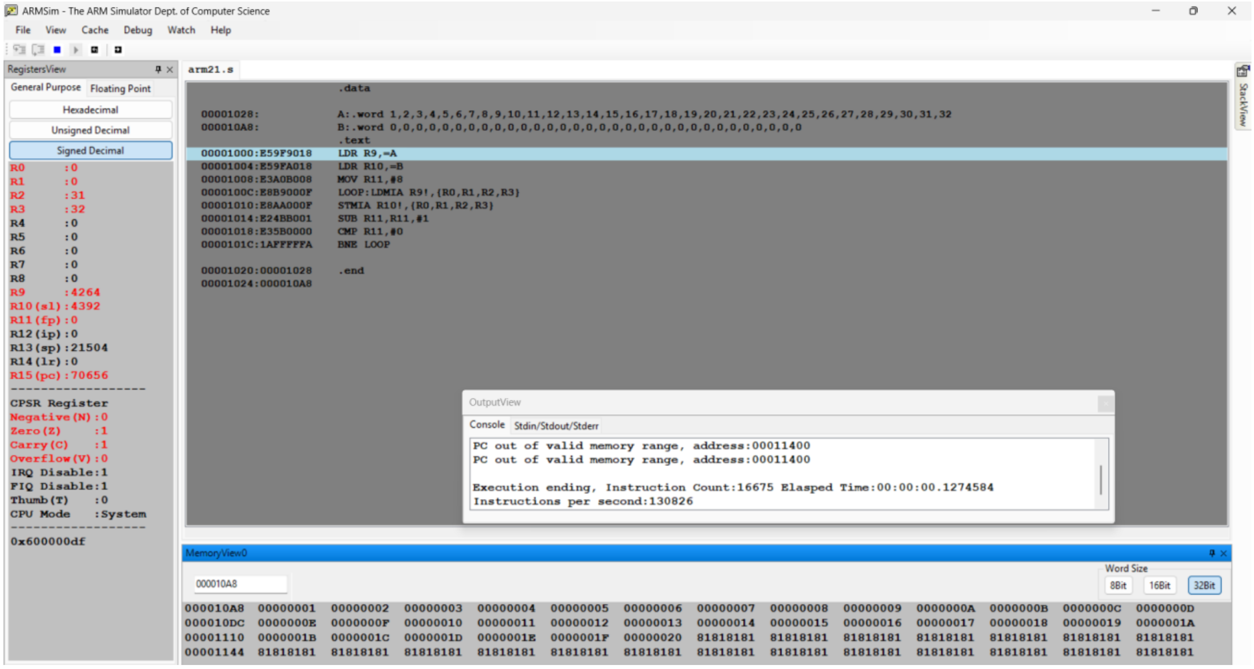
I. ARM Assembly Code(1)

Typed Code to be Included. Screenshot of Code not permitted

```
.data

A:.word
1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,
27,28,29,30,31,32
B:.word 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
.text
LDR R9,=A
LDR R10,=B
MOV R11,#8
LOOP:LDMIA R9!,{R0,R1,R2,R3}
STMIA R10!,{R0,R1,R2,R3}
SUB R11,R11,#1
CMP R11,#0
BNE LOOP

.end
```

II. Output Screen Shots (1)

The results should be clearly visible in the screenshots. The screenshot should include the code window, register window, memory window and console window



```
                    .data

 00001028:          A:.word 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
 000010A8:          B:.word 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
                    .text
 00001000:E59F9018  LDR R9,=A
 00001004:E59FA018  LDR R10,=B
 00001008:E3A0B008  MOV R11,#8
 0000100C:E8B9000F  LOOP:LDMIA R9!,{R0,R1,R2,R3}
 00001010:E8AA000F  STMIA R10!,{R0,R1,R2,R3}
 00001014:E24BB001  SUB R11,R11,#1
 00001018:E35B0000  CMP R11,#0
 0000101C:1AFFFFFA  BNE LOOP

 00001020:00001028  .end
 00001024:000010A8
```

```
R0        :0
R1        :0
R2        :31
R3        :32
R4        :0
R5        :0
R6        :0
R7        :0
R8        :0
R9        :4264
R10(sl):4392
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):70656
-------------------
CPSR Register
Negative(N):0
Zero(Z)     :1
Carry(C)    :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)    :0
CPU Mode    :System
-------------------
0x600000df
```

```
PC out of valid memory range, address:00011400
PC out of valid memory range, address:00011400

Execution ending, Instruction Count:16675 Elasped Time:00:00:00.1274584
Instructions per second:130826
```

```
000010A8

000010A8  00000001  00000002  00000003  00000004  00000005  00000006  00000007  00000008  00000009  0000000A  0000000B  0000000C  0000000D
000010DC  0000000E  0000000F  00000010  00000011  00000012  00000013  00000014  00000015  00000016  00000017  00000018  00000019  0000001A
00001110  0000001B  0000001C  0000001D  0000001E  0000001F  00000020  81818181  81818181  81818181  81818181  81818181  81818181  81818181
00001144  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181
```

Week#____3_______ Program Number: ____4___

Title of the Program

Write an ALP using ARM7TDMI, for the given matrix arranged in row major order, find the index of an element if coordinates of a matrix is given and also find the address of the indexed element. (Using MLA instruction)

I. ARM Assembly Code(1)

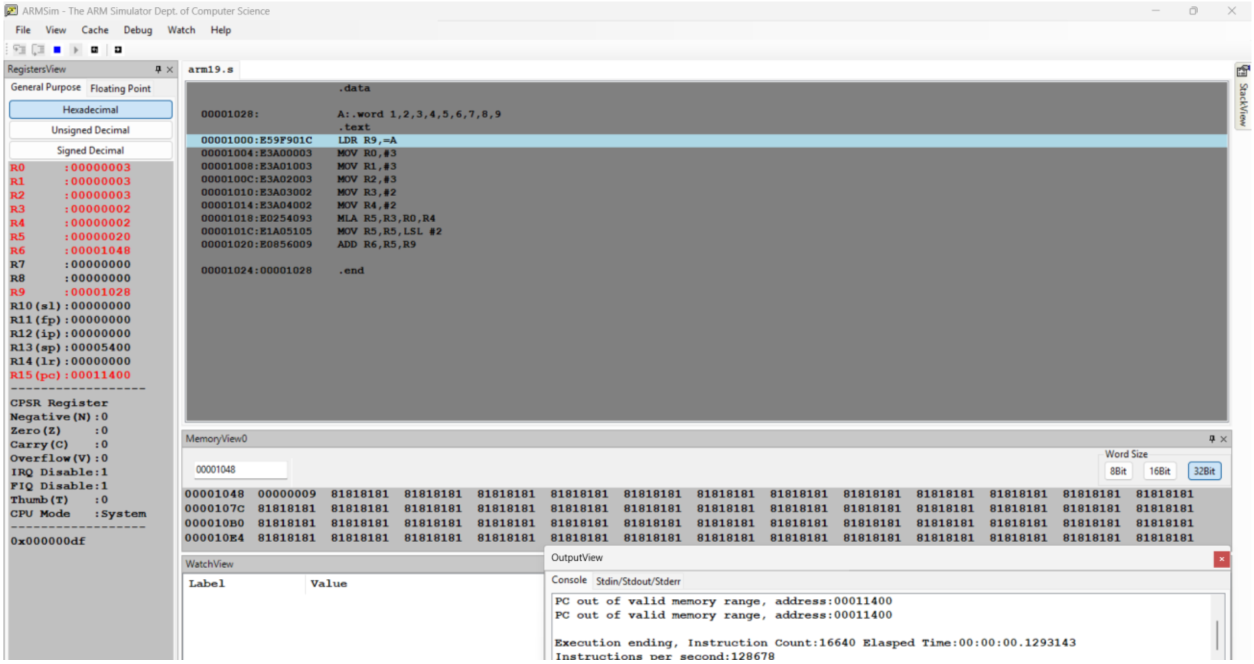
Typed Code to be Included. Screenshot of Code not permitted

```
.data

A:.word 1,2,3,4,5,6,7,8,9
.text
LDR R9,=A
MOV R0,#3
MOV R1,#3
MOV R2,#3
MOV R3,#2
MOV R4,#2
MLA R5,R3,R0,R4
MOV R5,R5,LSL #2
ADD R6,R5,R9
```

.end

II. Output Screen Shot (1)

The results should be clearly visible in the screenshots. The largest number can be finally stored in a register. The screenshot should include the code window, register window, memory window and console window



```
                        .data

00001028:               A:.word 1,2,3,4,5,6,7,8,9
                        .text
00001000:E59F901C       LDR R9,=A
00001004:E3A00003       MOV R0,#3
00001008:E3A01003       MOV R1,#3
0000100C:E3A02003       MOV R2,#3
00001010:E3A03002       MOV R3,#2
00001014:E3A04002       MOV R4,#2
00001018:E0254093       MLA R5,R3,R0,R4
0000101C:E1A05105       MOV R5,R5,LSL #2
00001020:E0856009       ADD R6,R5,R9

00001024:00001028       .end
```

```
R0       :00000003
R1       :00000003
R2       :00000003
R3       :00000002
R4       :00000002
R5       :00000020
R6       :00001048
R7       :00000000
R8       :00000000
R9       :00001028
R10(sl)  :00000000
R11(fp)  :00000000
R12(ip)  :00000000
R13(sp)  :00005400
R14(lr)  :00000000
R15(pc)  :00011400
------------------
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   :System
------------------
0x000000df
```

```
00001048
00001048 00000009 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
0000107C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010B0 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010E4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
```

```
PC out of valid memory range, address:00011400
PC out of valid memory range, address:00011400

Execution ending, Instruction Count:16640 Elasped Time:00:00:00.1293143
Instructions per second:128678
```

Disclaimer:

- The programs and output submitted is duly written, verified and executed by me.
- I have not copied from any of my peers nor from the external resource such as internet.

- If found plagiarized, I will abide with the disciplinary action of the University.

Signature: Keerthan P.V
Name: Keerthan P.V
SRN: PES2UG23CS272
Section: 4E
Date: 01/02/2025