Microprocessor and Computer Architecture UE22CS251B

4th Semester, Academic Year 2023-24

Date:

Name: ANKITH GOWDA B S	SRN:PES2UG22CS077	Section : B

LAB #____4_____ Program Number: ____1___

Title of the Program

1. a)Write an ALP to perform Convolution using MUL instruction (Addition of multiplication of respective numbers of loc A and loc B)

b) Write an ALP to perform Convolution using MLA instruction (Addition of multiplication of respective numbers of loc A and loc B).

I. ARM Assembly

Code CODE: .data a:

.word 1,2,3,4 b:

.word 2,3,4,5

c: .word 0

```
.text LDR
R0,=a
LDR R1,=b
MOV R2,#4
MOV R5,#0
LDR R8,=c
LOOP: LDR R3,[R0],#4
    LDR R4,[R1],#4
    MUL R6,R3,R4
    ADD R5,R5,R6
    SUB R2,R2,#1
    CMP R2,#0
    BNE LOOP
    STR R5,[R8]
    SWI 0X11
.end
```

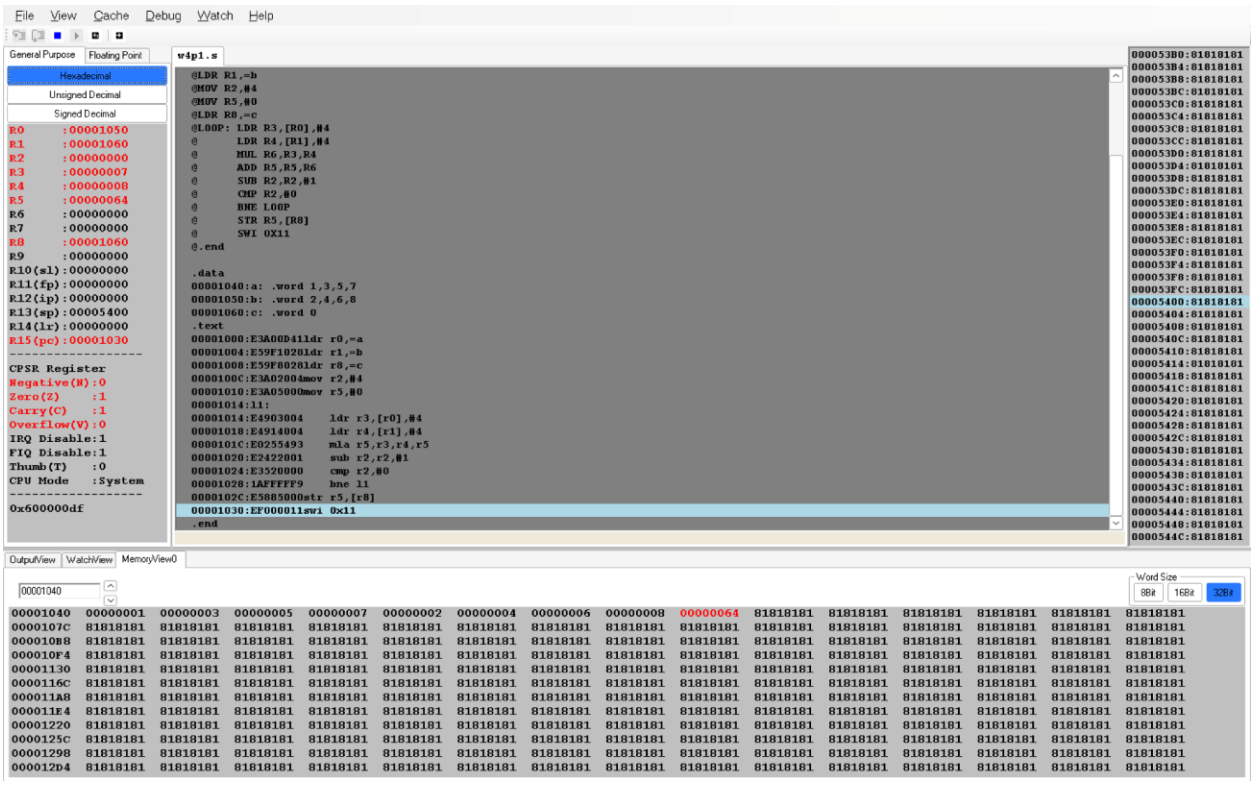
B:

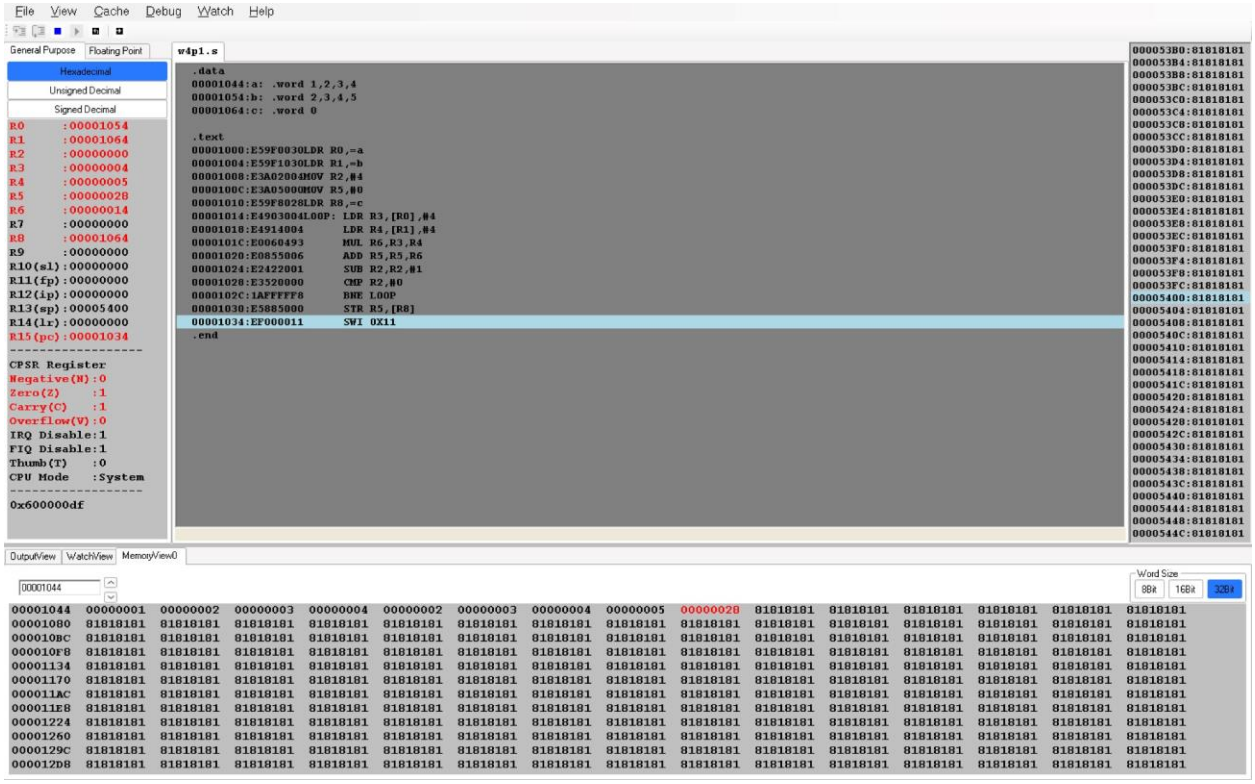
```
@.data
@a: .word 1,3,5,7
@b: .word 2,4,6,8
@c: .word 0
@.text
```

```
@ldr r0,=a
@ldr r1,=b
@ldr r8,=c
@mov r2,#4
@mov r5,#0
@l1:
@    ldr r3,[r0],#4
@ ldr r4,[r1],#4
 2 mla r5,r3,r4,r5
 @ sub r2,r2,#1
 @ cmp r2,#0
@     bne l1
@str r5,[r8]
@swi 0x11
@.end
```

II. Output Screen Shots

(Two Screenshots including Register Window,Memory Window and Code Window)


File View Cache Debug Watch Help
General Purpose
Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :00001050
R1 :00001060
R2 :00000000
R3 :00000007
R4 :00000008
R5 :00000064
R6 :00000000
R7 :00000000
R8 :00001060
R9 :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):00000000
R15(pc):00001030
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
w4p1.s
@LDR R1,=b
@MOV R2,#4
@MOV R5,#0
@LDR R8,=c
@LOOP: LDR R3,[R0],#4
@ LDR R4,[R1],#4
@ MUL R6,R3,R4
@ ADD R5,R5,R6
@ SUB R2,R2,#1
@ CMP R2,#0
@ BNE LOOP
@ STR R5,[R8]
@ SWI 0X11
@.end
.data
00001040:a: .word 1,3,5,7
00001050:b: .word 2,4,6,8
00001060:c: .word 0
.text
00001000:E3A00D41ldr r0,=a
00001004:E59F1028ldr r1,=b
00001008:E59F8028ldr r8,=c
0000100C:E3A02004mov r2,#4
00001010:E3A05000mov r5,#0
00001014:11:
00001014:E4903004 ldr r3,[r0],#4
00001018:E4914004 ldr r4,[r1],#4
0000101C:E0255493 mla r5,r3,r4,r5
00001020:E2422001 sub r2,r2,#1
00001024:E3520000 cmp r2,#0
00001028:1AFFFFF9 bne 11
0000102C:E5885000str r5,[r8]
00001030:EF000011swi 0x11
.end
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
OutputView
WatchView
MemoryView0
00001040
Word Size
8Bit
16Bit
32Bit
00001040 00000001 00000003 00000005 00000007 00000002 00000004 00000006 00000008 00000064 81818181 81818181 81818181 81818181 81818181 81818181
0000107C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010B8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010F4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001130 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
0000116C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000011A8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000011E4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001220 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
0000125C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001298 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000012D4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181


File View Cache Debug Watch Help
General Purpose
Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :00001054
R1 :00001064
R2 :00000000
R3 :00000004
R4 :00000005
R5 :0000002B
R6 :00000014
R7 :00000000
R8 :00001064
R9 :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):00000000
R15(pc):00001034
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
w4p1.s
.data
00001044:a: .word 1,2,3,4
00001054:b: .word 2,3,4,5
00001064:c: .word 0
.text
00001000:E59F0030LDR R0,=a
00001004:E59F1030LDR R1,=b
00001008:E3A02004MOV R2,#4
0000100C:E3A05000MOV R5,#0
00001010:E59F8028LDR R8,=c
00001014:E4903004LOOP: LDR R3,[R0],#4
00001018:E4914004 LDR R4,[R1],#4
0000101C:E0060493 MUL R6,R3,R4
00001020:E0855006 ADD R5,R5,R6
00001024:E2422001 SUB R2,R2,#1
00001028:E3520000 CMP R2,#0
0000102C:1AFFFFF8 BNE LOOP
00001030:E5885000 STR R5,[R8]
00001034:EF000011 SWI 0X11
.end
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
OutputView
WatchView
MemoryView0
00001044
Word Size
8Bit
16Bit
32Bit
00001044 00000001 00000002 00000003 00000004 00000002 00000003 00000004 00000005 00000028 81818181 81818181 81818181 81818181 81818181 81818181
00001080 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010BC 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010F8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001134 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001170 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000011AC 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000011E8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001224 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001260 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
0000129C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000012D8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181

Microprocessor and Computer Architecture

UE22CS251B

4th Semester, Academic Year 2023-24

Date:

Name: ANKITH GOWDA B S	SRN:PES2UG22CS077	Section :B

LAB #____4_____ Program Number: ____2___

Title of the Program

Write an ALP to implement Sum[i]+=a[i][j]

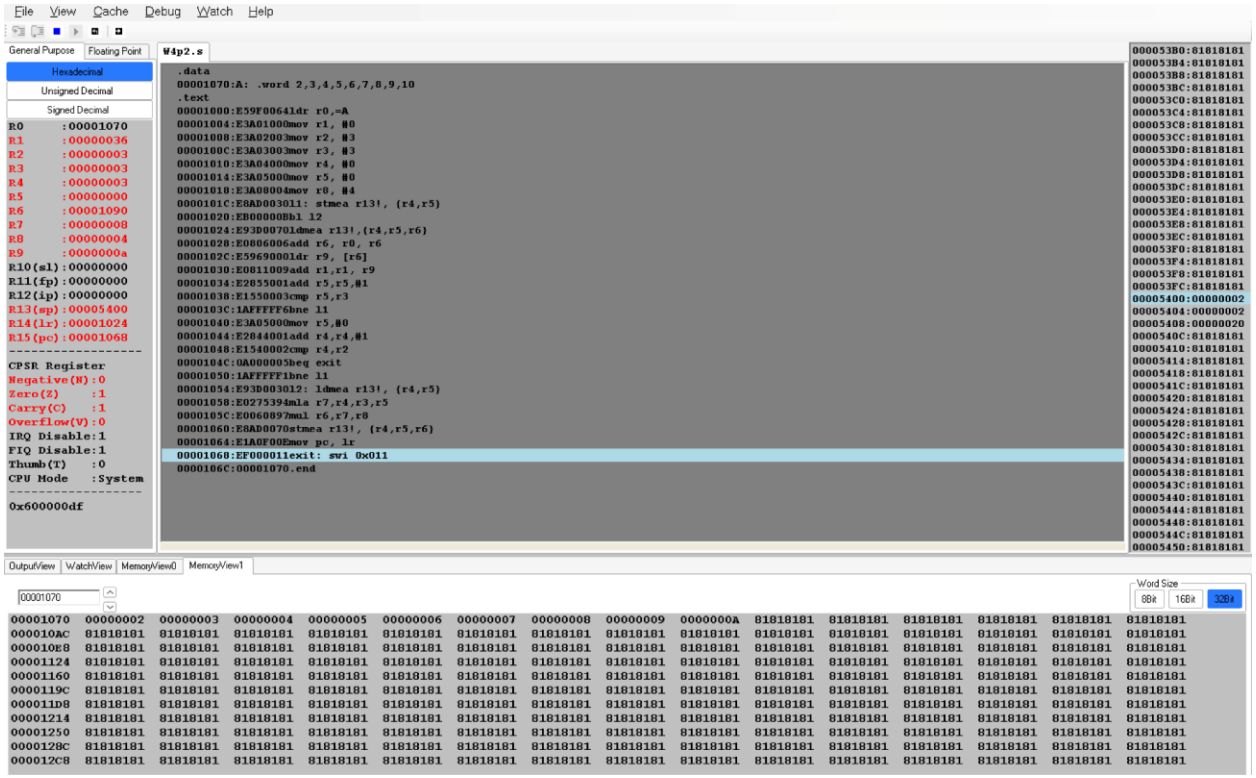
I. ARM Assembly

```
Code CODE: .data
A: .word 2,3,4,5,6,7,8,9,10
.text ldr
r0,=A mov
r1, #0
mov r2,
#3 mov
r3, #3
mov r4,
#0 mov
r5, #0
mov r8,
#4 l1:
```

```
stmea
r13!,
{r4,r5} bl
l2
ldmea r13!,{r4,r5,r6}
add r6, r0, r6 ldr r9,
[r6] add r1,r1, r9 add
r5,r5,#1 cmp r5,r3
bne l1 mov r5,#0 add
r4,r4,#1 cmp r4,r2
beq exit bne l1 l2:
ldmea r13!, {r4,r5}
mla r7,r4,r3,r5 mul
r6,r7,r8 stmea r13!,
{r4,r5,r6} mov pc, lr
exit: swi 0x011
.end
```

II. Output Screen Shots

(One Screenshot including Register Window,Memory Window and Code Window)



Microprocessor and Computer Architecture

UE22CS251B

4th Semester, Academic Year 2023-24

Date:

Name: ANKITH GOWDA B S	SRN:PES2UG22CS077	Section :B

LAB #____4_____ Program Number: ____3___

Title of the Program

Write an ALP to find the length of a given string

I. ARM Assembly

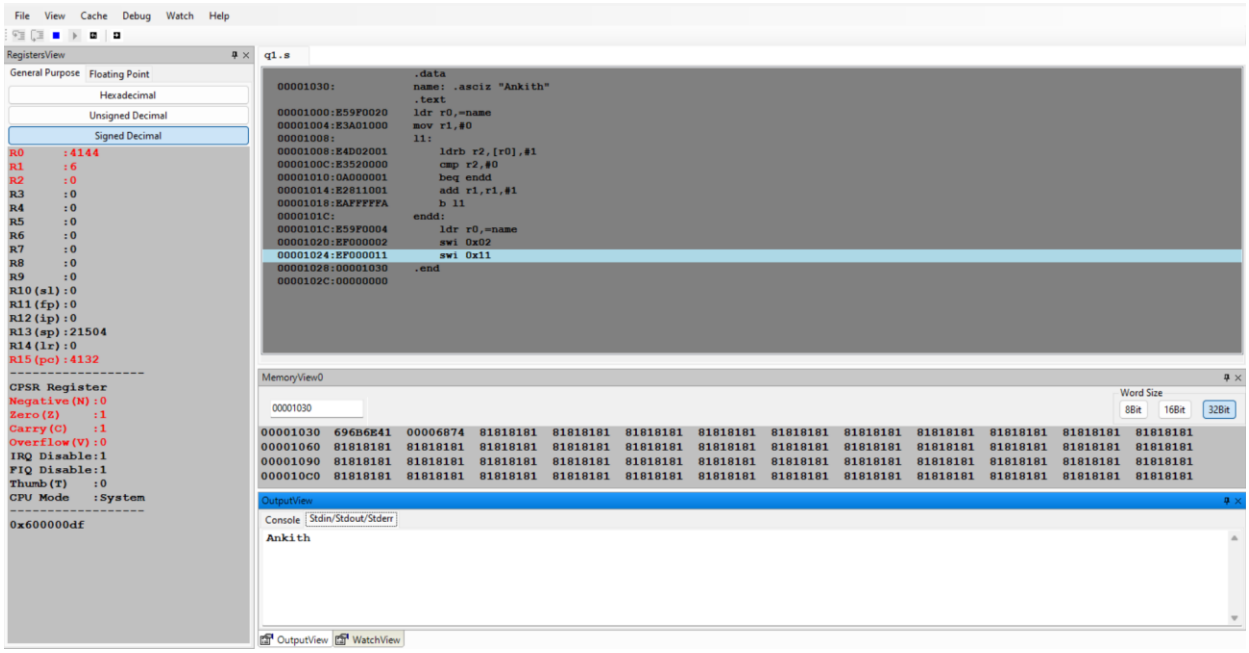
Code CODE:

```
.data
name: .asciz
"Ankith"
.text
ldr r0,=name
mov r1,#0
l1:
   ldrb r2,[r0],#1
   cmp r2,#0
   beq endd
   add r1,r1,#1
   b l1
endd:
   ldr r0,=name
```

```
        swi 0x02
        swi 0x11
    .end
```

II. Output Screen Shot

(One Screenshot including Register Window,Memory Window and Code Window, Output Window with string displayed)



Microprocessor and Computer Architecture

UE22CS251B 4th Semester, Academic Year 2023-24

Date:

Name: ANKITH GOWDA B S	SRN:PES2UG22CS077	Section :B

LAB #____4_____ Program Number: ____4___

Title of the Program

Write an ALP to copy string from one location to another

I. ARM Assembly Code

CODE:

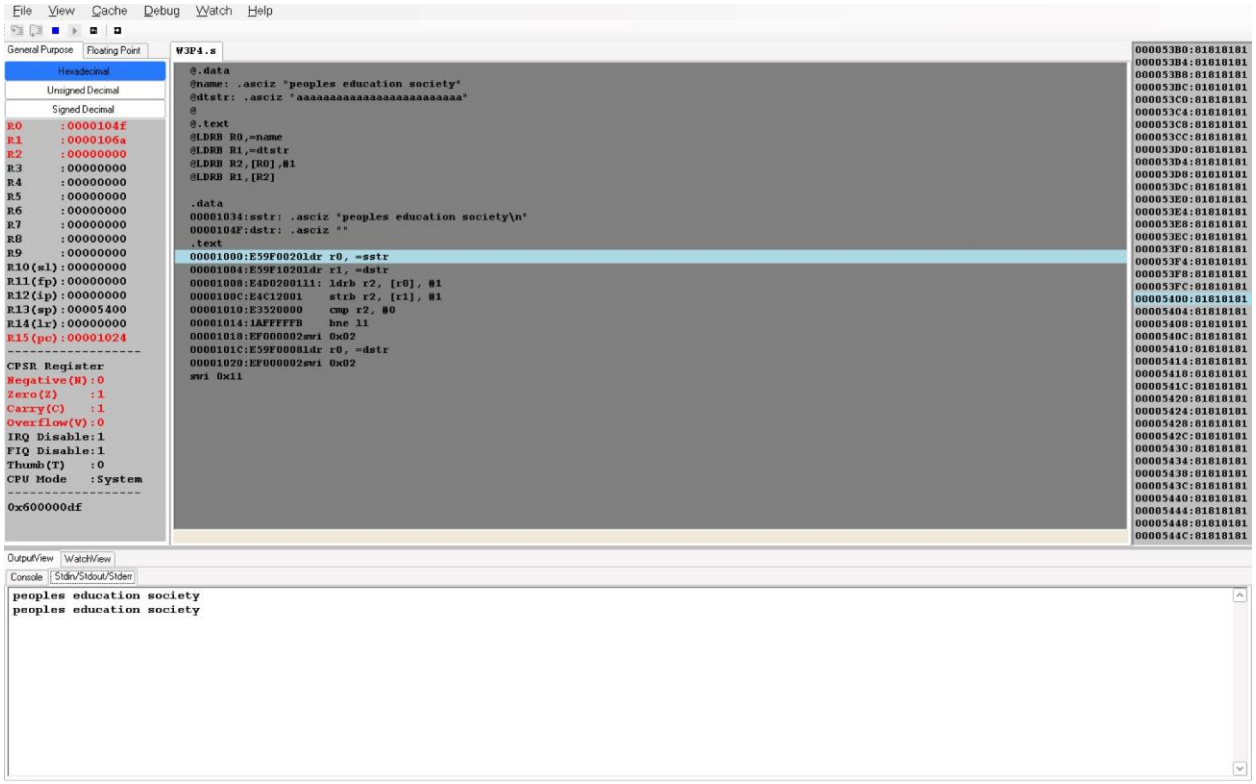
```
.data
sstr: .asciz "peoples education society\n"
dstr: .asciz "" .text
ldr r0, =sstr ldr r1,
=dstr l1: ldrb r2,
[r0], #1
   strb r2, [r1],
   #1 cmp r2, #0
   bne l1
swi 0x02
ldr r0,
=dstr swi
```

0x02 swi

0x11

II. Output Screen Shots

(One Screenshot including Register Window,Memory Window and Code Window,Output Window with strings displayed)



UE22CS251B

4th Semester, Academic Year 2023-24

Date:

Name: ANKITH GOWDA B S	SRN:PES2UG22CS077	Section :B

LAB #____4_____ Assignment Question 1

Title of the Program

Write an ALP to find whether a given character is present in a string. If present, find how many times the given character is present in a string.

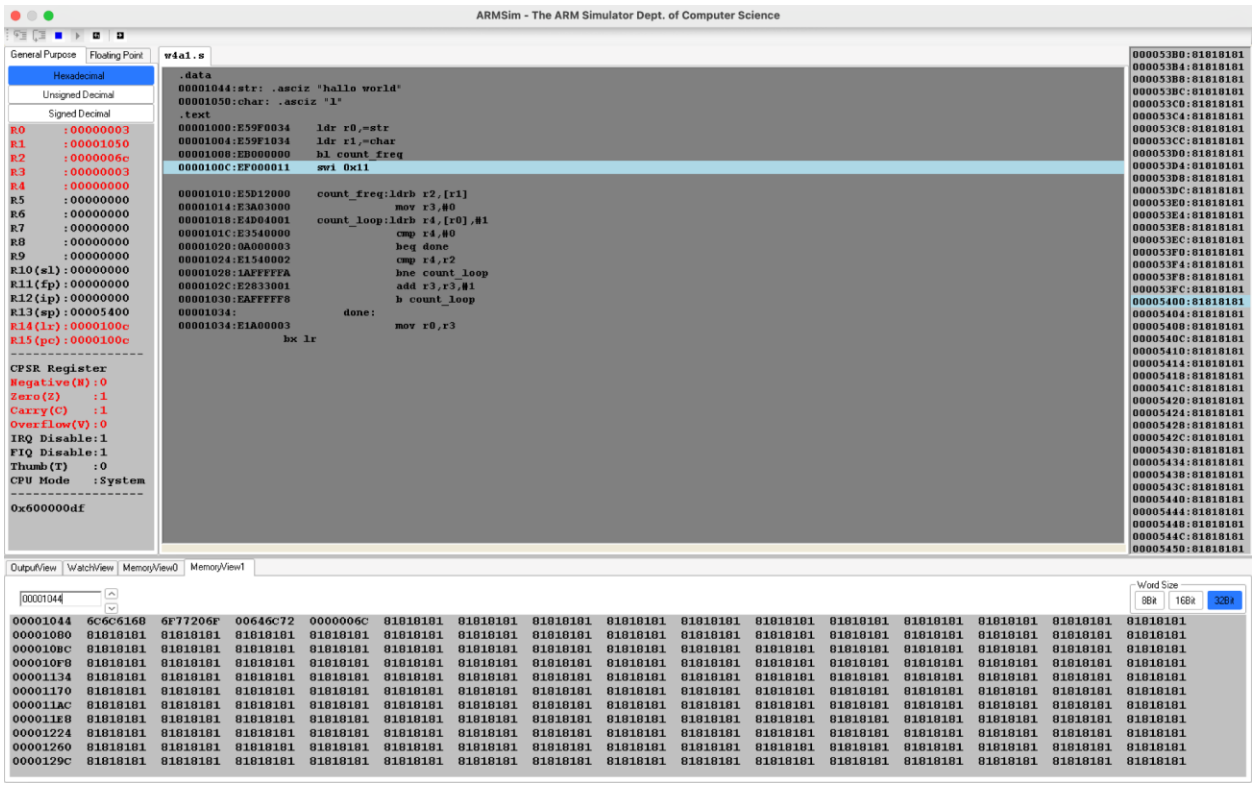
I. ARM Assembly Code Code: .data

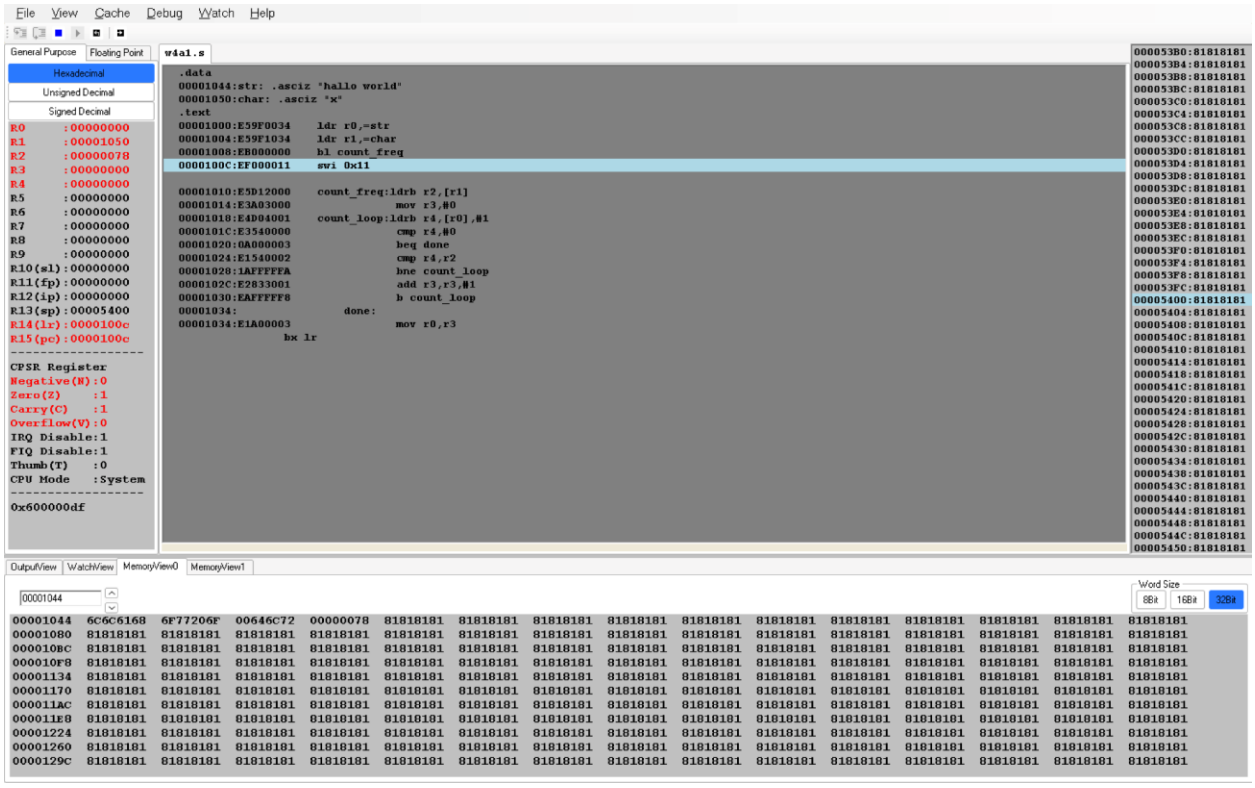
```
str: .asciz "hallo world"
char: .asciz "x"
.text ldr r0,=str ldr
   r1,=char bl count_freq
   swi 0x11
   count_freq:ldrb r2,[r1]
   mov r3,#0
   count_loop:ldrb
         r4,[r0],#1 cmp
         r4,#0 beq done
```

```
cmp r4,r2 bne
count_loop add
r3,r3,#1 b
count_loop done:
mov r0,r3 bx lr
```

II. Output Screen Shots

(Two Screenshots-Character Present,Character not Present,screenshot including Register Window,Memory Window and Code Window)


ARMSim - The ARM Simulator Dept. of Computer Science
General Purpose
Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :00000003
R1 :00001050
R2 :0000006c
R3 :00000003
R4 :00000000
R5 :00000000
R6 :00000000
R7 :00000000
R8 :00000000
R9 :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):0000100c
R15(pc):0000100c
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
w4a1.s
.data
00001044:str: .asciz "hallo world"
00001050:char: .asciz "l"
.text
00001000:E59F0034 ldr r0,=str
00001004:E59F1034 ldr r1,=char
00001008:EB000000 bl count_freq
0000100C:EF000011 swi 0x11
00001010:E5D12000 count_freq:ldrb r2,[r1]
00001014:E3A03000 mov r3,#0
00001018:E4D04001 count_loop:ldrb r4,[r0],#1
0000101C:E3540000 cmp r4,#0
00001020:0A000003 beq done
00001024:E1540002 cmp r4,r2
00001028:1AFFFFFA bne count_loop
0000102C:E2833001 add r3,r3,#1
00001030:EAFFFFF8 b count_loop
00001034: done:
00001034:E1A00003 mov r0,r3
bx lr
OutputView
WatchView
MemoryView0
MemoryView1
00001044
Word Size
8Bit
16Bit
32Bit
00001044 6c6c6168 6f77206f 00646c72 0000006c 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
00001080 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
000010BC 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
000010F8 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
00001134 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
00001170 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
000011AC 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
000011E8 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
00001224 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
00001260 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
0000129C 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101 01010101
000053B0:01010101
000053B4:01010101
000053B8:01010101
000053BC:01010101
000053C0:01010101
000053C4:01010101
000053C8:01010101
000053CC:01010101
000053D0:01010101
000053D4:01010101
000053D8:01010101
000053DC:01010101
000053E0:01010101
000053E4:01010101
000053E8:01010101
000053EC:01010101
000053F0:01010101
000053F4:01010101
000053F8:01010101
000053FC:01010101
00005400:01010101
00005404:01010101
00005408:01010101
0000540C:01010101
00005410:01010101
00005414:01010101
00005418:01010101
0000541C:01010101
00005420:01010101
00005424:01010101
00005428:01010101
0000542C:01010101
00005430:01010101
00005434:01010101
00005438:01010101
0000543C:01010101
00005440:01010101
00005444:01010101
00005448:01010101
0000544C:01010101
00005450:01010101


File
View
Cache
Debug
Watch
Help
General Purpose
Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :00000000
R1 :00001050
R2 :00000078
R3 :00000000
R4 :00000000
R5 :00000000
R6 :00000000
R7 :00000000
R8 :00000000
R9 :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):0000100c
R15(pc):0000100c
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
w4a1.s
.data
00001044:str: .asciz "hallo world"
00001050:char: .asciz "x"
.text
00001000:E59F0034 ldr r0,=str
00001004:E59F1034 ldr r1,=char
00001008:EB000000 bl count_freq
0000100C:EF000011 swi 0x11
00001010:E5D12000 count_freq:ldrb r2,[r1]
00001014:E3A03000 mov r3,#0
00001018:E4D04001 count_loop:ldrb r4,[r0],#1
0000101C:E3540000 cmp r4,#0
00001020:0A000003 beq done
00001024:E1540002 cmp r4,r2
00001028:1AFFFFFA bne count_loop
0000102C:E2833001 add r3,r3,#1
00001030:EAFFFFF8 b count_loop
00001034: done:
00001034:E1A00003 mov r0,r3
bx lr
OutputView
WatchView
MemoryView0
MemoryView1
00001044
Word Size
8Bit
16Bit
32Bit
00001044 6c6c6168 6f77206f 00646c72 00000078 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001080 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010BC 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010F8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001134 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001170 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000011AC 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000011E8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001224 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001260 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
0000129C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000053B0:01010101
000053B4:01010101
000053B8:01010101
000053BC:01010101
000053C0:01010101
000053C4:01010101
000053C8:01010101
000053CC:01010101
000053D0:01010101
000053D4:01010101
000053D8:01010101
000053DC:01010101
000053E0:01010101
000053E4:01010101
000053E8:01010101
000053EC:01010101
000053F0:01010101
000053F4:01010101
000053F8:01010101
000053FC:01010101
00005400:01010101
00005404:01010101
00005408:01010101
0000540C:01010101
00005410:01010101
00005414:01010101
00005418:01010101
0000541C:01010101
00005420:01010101
00005424:01010101
00005428:01010101
0000542C:01010101
00005430:01010101
00005434:01010101
00005438:01010101
0000543C:01010101
00005440:01010101
00005444:01010101
00005448:01010101
0000544C:01010101
00005450:01010101

Microprocessor and Computer Architecture

UE22CS251B

4th Semester, Academic Year 2023-24

Date:

Name: ANKITH GOWDA B S	SRN:PES2UG22CS077	Section : B

LAB #____4_____ Assignment Question 2

Title of the Program

Write a program in ARM7TDMI-ISA to generate a diagonal matrix.

;Note: do not read the matrix elements.

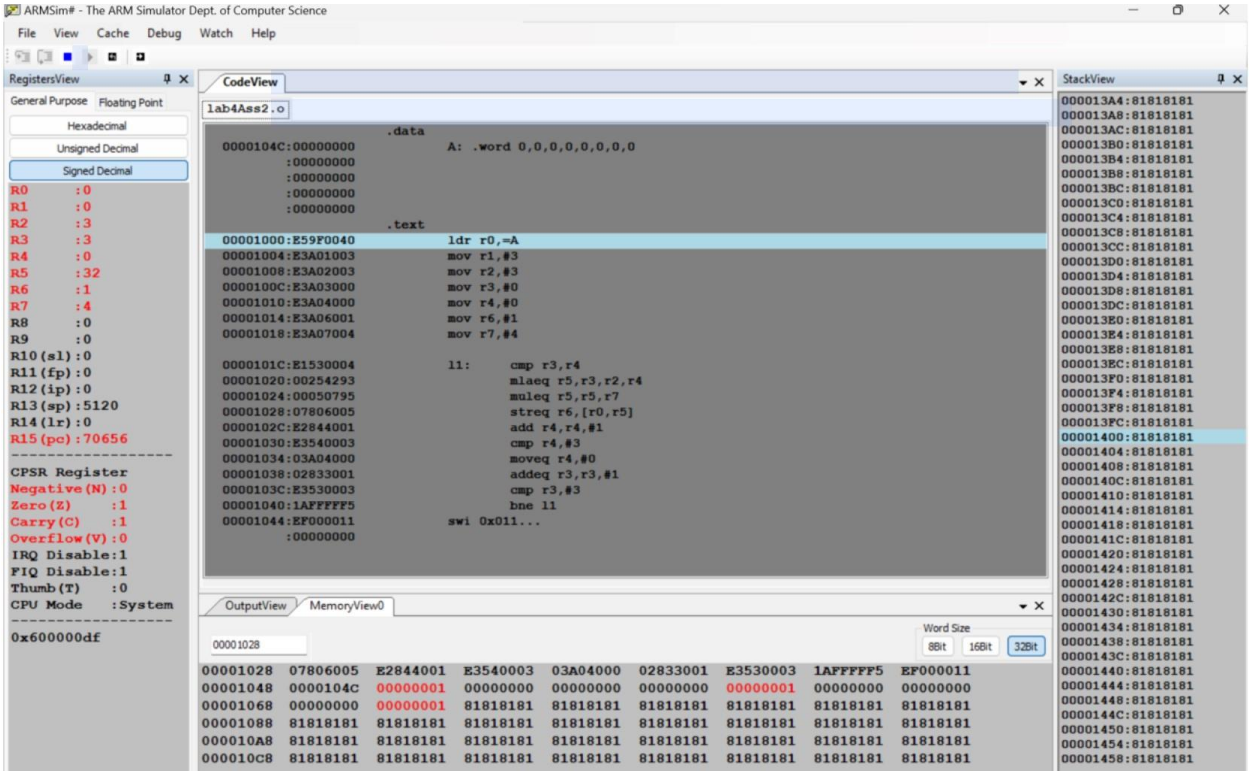
I. ARM Assembly Code

```
code:
.data
A: .word 0,0,0,0,0,0,0,0,0
.text
        ldr r0,=A mov
        r1,#3 mov r2,#3
        mov r3,#0 mov
        r4,#0 mov r6,#1
```

```
mov r7,#4 l1:
cmp r3,r4 mlaeq
r5,r3,r2,r4
muleq r5,r5,r7
streq r6,[r0,r5]
add r4,r4,#1
cmp r4,#3
moveq r4,#0
addeq r3,r3,#1
cmp r3,#3 bne l1
swi 0x011
```

II. Output Screen Shots

(One Screenshot including Register Window,Memory Window and Code Window)



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.
-

  Name: ANKITH GOWDA B S
  SRN:PES2UG22CS077
  Section: B
  Date:20/02/2024