

# **Microprocessor and Computer Architecture**

**UE24CS251B**

**4th Semester, Academic Year 2025-26**

Date:

Name: Harshit Chandak	SRN: PES2UG24CS185	Section C
-----------------------	-----------------------	--------------

LAB # 5

Program Number: 1

Title of the Program

**Write an ARM assembly language program to compute the sum of all elements in a 3×3 matrix stored in memory.**

**The matrix elements are stored in row-major order as a one-dimensional array. The program should:**

- **Use nested loops to iterate over rows and columns.**
- **Compute the effective memory address of each matrix element using a separate subroutine.**
- **Load each matrix element from memory and accumulate the total sum in a register.**
- **Store or maintain the final sum in a designated register before program termination.**

**The program must use:**

- A subroutine to calculate the address of `a[i][j]`
- Stack operations (STMFD, LDMFD) to preserve register values
- Multiply-Accumulate (MLA) for address computation

I. Typed ARM Assembly Code

II. Output Screen Shot

*( Screenshot including Register Window, Output Window and Code Window)*

**.DATA**

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

**SUM: .WORD 0**

**.TEXT**

**LDR R8,=A**

**MOV R0,#0**

**MOV R7,#0**

**MOV R3,#3**

**LDR R9,=SUM**

**ROW:**

**CMP R0,#3**

**BEQ EXIT**

**MOV R1,#0**

**COLUMN:**

**CMP R1,#3**

**BEQ NEXT\_ROW**

**BL ADDRESS**

**LDR R6,[R5]**

**ADD R7,R7,R6**

**ADD R1,R1,#1**

**B COLUMN**

**NEXT\_ROW:**

**ADD R0,R0,#1**

**B ROW**

**ADDRESS:**

**STMFD R13!,{R3,R4,LR}**

**MLA R4,R0,R3,R1**

**MOV R4, R4, LSL #2**

**ADD R5,R4,R8**

**LDMFD R13!,{R3,R4,LR}**

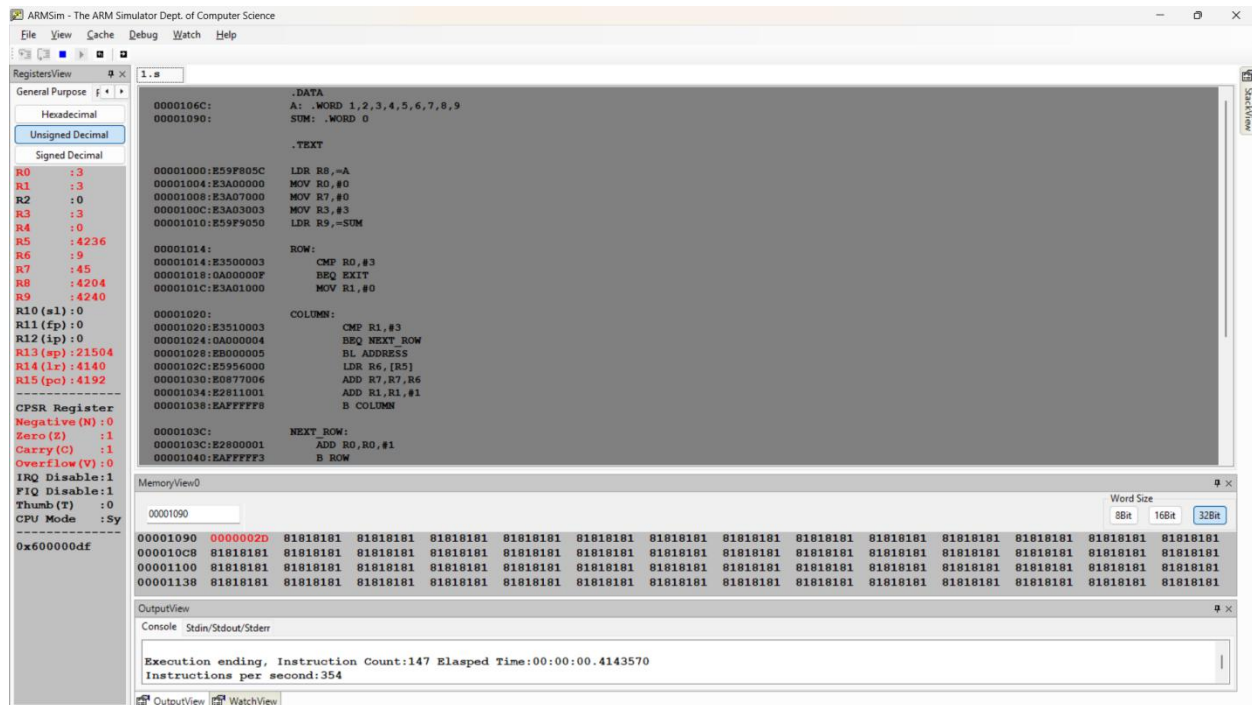
**MOV PC,LR**

**EXIT:**

**STR R7,[R9]**

**SWI 0x11**

**.END**



# Microprocessor and Computer Architecture

UE24CS251B

4th Semester, Academic Year 2025-26

Date:

Name: Harshit Chandak	SRN: PES2UG24CS185	Section C
-----------------------	-----------------------	--------------

LAB # 5

Program Number: 2

## Title of the Program

**Write an ARM assembly program to transfer a 3×3 matrix stored in memory location A to memory location C. Use a subroutine to compute element addresses and stack operations to save and restore registers.**

**The source matrix is stored as a one-dimensional array in row-major order. The program should:**

**Use nested loops to traverse rows and columns of the matrix.**

**Compute the effective address of each matrix element using a separate subroutine.**

**Load each element from the source matrix and store it into the destination matrix.**

**Use stack operations to preserve register values during subroutine calls.**

**Ensure that all matrix elements are copied correctly before program termination.**

- I. Typed ARM Assembly Code
- II. Output Screen Shot

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

**.DATA**

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

**B: .WORD 0,0,0,0,0,0,0,0,0**

**.TEXT**

**MOV R0,#0**

**MOV R2,#3**

**LDR R3,=A**

**LDR R4,=B**

**ROW:**

**CMP R0,#3**

**BEQ EXIT**

**MOV R1,#0**

**COLUMN:**

**CMP R1,#3**

**BEQ NEXT\_ROW**

**BL ADDRESS**

**LDR R7,[R6]**

**STR R7,[R4]**

**ADD R4,R4,#4**

**ADD R1,R1,#1**

**B COLUMN**

**NEXT\_ROW:**

**ADD R0,R0,#1**

**B ROW**

**ADDRESS:**

**STMFD R13!,{R2,R1,LR}**

**MLA R5,R0,R2,R1**

**MOV R5, R5, LSL #2**

**ADD R6,R3,R5**

**LDMFD R13!,{R2,R1,LR}**

**MOV PC,LR**

**EXIT:**



SWI 0x11

.END

The screenshot displays the ARMSim ARM simulator interface. The main window shows assembly code with labels like .DATA, .TEXT, and .ROW. The left sidebar contains the RegistersView, showing the CPSR register with flags like Negative (N), Zero (Z), Carry (C), and Overflow (V). The bottom section shows the MemoryView and OutputView, which displays the execution ending message and instruction count.

Microprocessor and Computer Architecture

UE24CS251B

4th Semester, Academic Year 2025-26

Date:

Name: Harshit Chandak

SRN:  
PES2UG24CS185

Section  
C

LAB #\_\_\_\_5\_\_\_\_\_

Program Number: \_\_\_\_3\_\_\_\_

Title of the Program

**Generate the first N Fibonacci numbers and store them in memory while using stack operations for register preservation.**

- I. ARM Assembly Code
- II. Output Screen Shot  
*(One Screenshot including Register Window, Memory Window and Code Window)*

**.DATA**

**FIB: .WORD 0,0,0,0,0**

**.TEXT**

**LDR R4, =FIB**

**MOV R2, #10**

**MOV R0, #0**

**MOV R1, #1**

**STMFD SP!, {R0,R1}**

**STR R0, [R4], #4**

**STR R1, [R4], #4**

**SUB R2, R2, #2**

**LOOP:**

**ADD R3, R0, R1**

**MOV R0, R1**

**MOV R1, R3**

**STMFD SP!, {R3}**

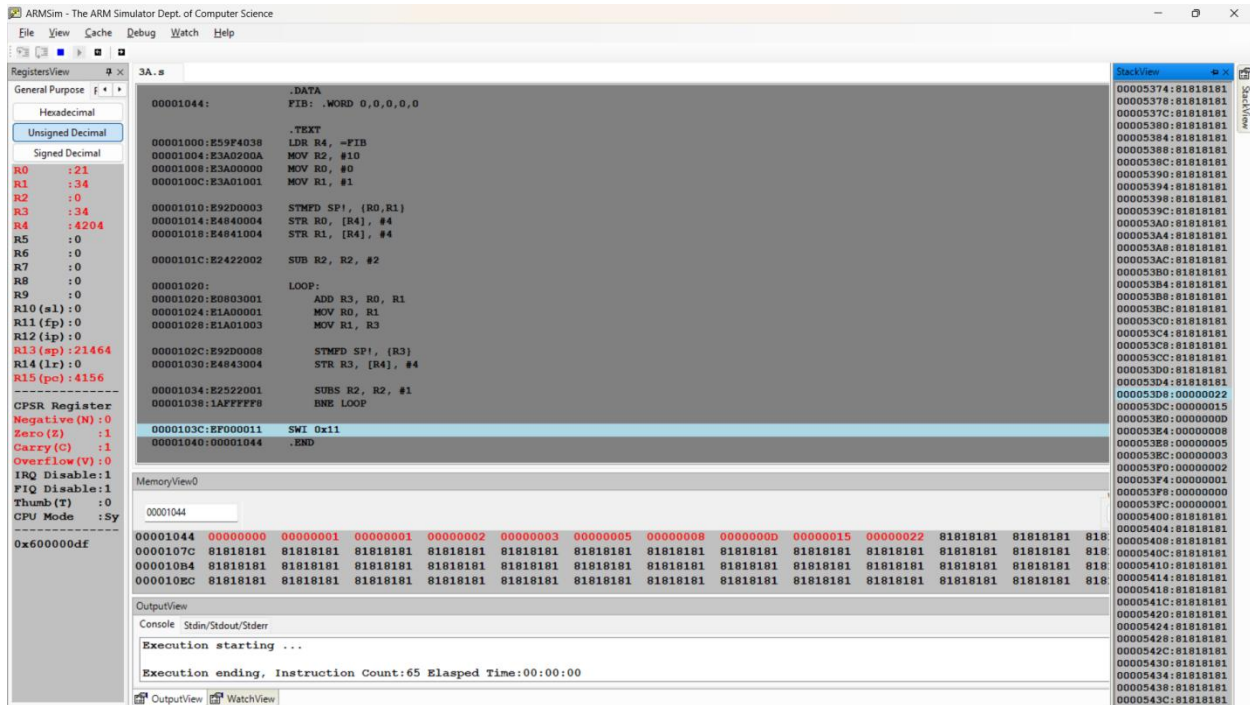
**STR R3, [R4], #4**

**SUBS R2, R2, #1**

**BNE LOOP**

**SWI 0x11**

**.END**



# Microprocessor and Computer Architecture

UE22CS251B

4th Semester, Academic Year 2023-24

Date:

Name:Harshit Chandak	SRN: PES2UG24CS185	Section C
----------------------	-----------------------	--------------

LAB # \_\_\_\_5\_\_\_\_

Program Number: \_\_\_\_4\_\_\_\_

Title of the Program

**Write an ARM ALP to count the number of words in a string.**

- I. ARM Assembly Code
- II. Output Screen Shot  
*(One Screenshot including Register Window, Memory Window ,Output Window, Code Window)*

**.DATA**

**MSG: .ASCIZ "ARM MPCA PES"**

**COUNT: .WORD 0**

**.TEXT**

**LDR R0,=MSG**

**MOV R1,#0**

**MOV R2,#1**

**LOOP:**

**LDRB R3,[R0],#1**

**CMP R3,#0**

**BEQ EXIT**

**CMP R3,#' '**

**BEQ SPACE**

**CMP R2,#1**

**ADDEQ R1,R1,#1**

**MOVEQ R2,#0**

**B LOOP**

**SPACE:**

**MOV R2,#1**

**B LOOP**

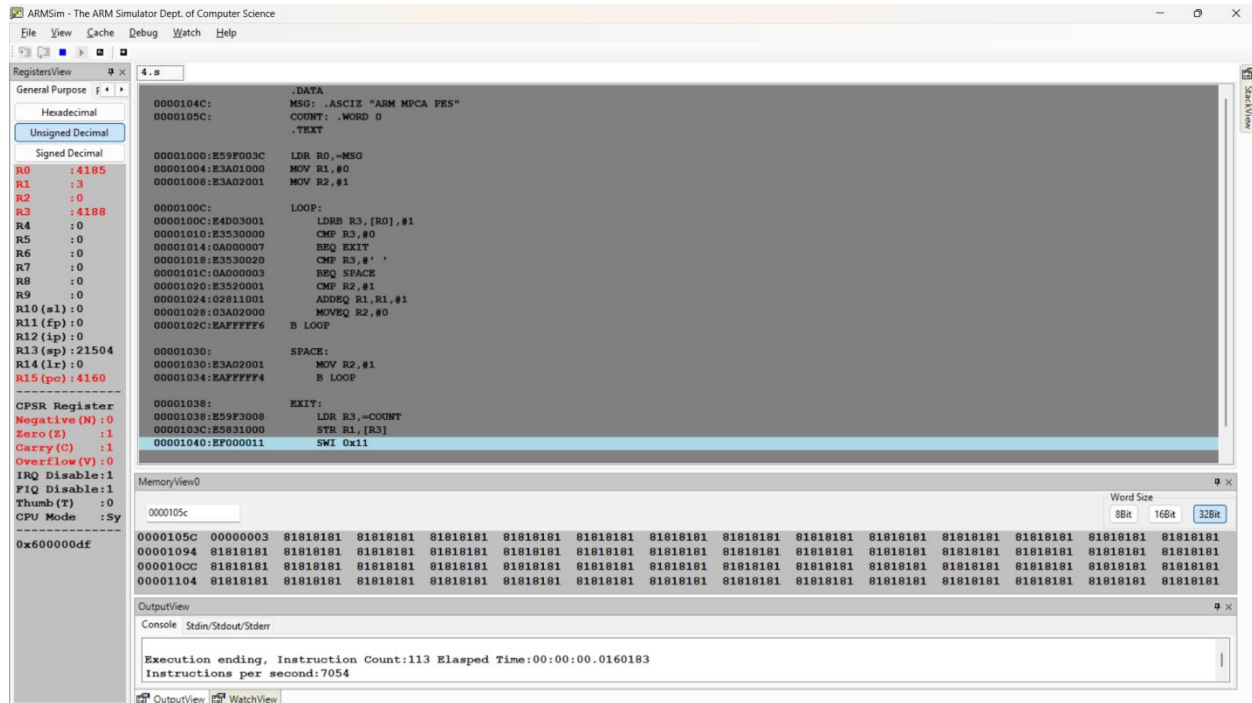
**EXIT:**

**LDR R3,=COUNT**

**STR R1,[R3]**

**SWI 0x11**

**.END**



# Microprocessor and Computer Architecture

UE22CS251B

4th Semester, Academic Year 2023-24

Date:

Name: Harshit Chandak	SRN: PES2UG24CS185	Section C
-----------------------	-----------------------	--------------

LAB #\_\_\_\_5\_\_\_\_\_

Assignment Question 1

Title of the Program

**Write an ALP using conditional ARM instructions to sort an array of numbers using Bubble Sort Algorithm. Use STMFD to save registers before swap and LDMFD to restore the registers**

I. ARM Assembly Code

II. Output Screen Shot

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

**.DATA**

**A: .WORD 3,4,1,6,5**

**.TEXT**

**LDR R1,=A**

**MOV R5,#5**



**MOV R0,#0**

**LOOP:**

**CMP R0,#4**

**BEQ EXIT**

**MOV R2,#0**

**INNERLOOP:**

**MOV R4, R5**

**SUB R4, R4, R0**

**SUB R4, R4, #1**

**CMP R2, R4**

**BEQ NEXT\_OUTER**

**BL ADDRESS**

**LDR R6,[R3]**

**LDR R7,[R3,#4]**

**CMP R6,R7**

**STRGT R7,[R3]**

**STRGT R6,[R3,#4]**

**ADD R2,R2,#1**

**B INNERLOOP**

**NEXT\_OUTER:**

**ADD R0,R0,#1**

**B LOOP**

**ADDRESS:**

**STMFD R13!,{R0,R2,LR}**

**ADD R3,R1,R2,LSL #2**

**LDMFD R13!,{R0,R2,LR}**

**MOV PC,LR**

EXIT:

SWI 0x11

.END

The screenshot displays the ARMSim interface with the following components:

- RegistersView:** Shows registers R0 through R15. R0 is 4, R1 is 4204, R2 is 1, R3 is 4204, R4 is 1, R5 is 5, R6 is 1, R7 is 3, R8 is 0, R9 is 0, R10 is 0, R11 is 0, R12 is 0, R13 is 21504, R14 is 4144, and R15 is 4196. The CPSR register shows Negative(N):0, Zero(Z):1, Carry(C):1, Overflow(OV):0, IRQ Disable:1, FIQ Disable:1, Thumb(T):0, and CPU Mode:Sy.
- Assembly Code:** The main window shows assembly instructions. The instruction at address 00001064 is highlighted: `SWI 0x11`. Other instructions include `SUB R4, R4, #1`, `CMP R2, R4`, `REQ NEXT_OUTER`, `BL ADDRESS`, `LDR R6, [R3]`, `LDR R7, [R3, #4]`, `CMP R6, R7`, `STROT R7, [R3]`, `STROT R6, [R3, #4]`, `ADD R2, R2, #1`, `B INNERLOOP`, `NEXT_OUTER:`, `ADD R0, R0, #1`, `B LOOP`, `ADDRESS:`, `STMPD R13!, (R0, R2, LR)`, `ADD R3, R1, R2, LSL #2`, `LMPD R13!, (R0, R2, LR)`, `MOV PC, LR`, and `.END`.
- MemoryView0:** Shows a memory dump starting at address 0000106C. The data is mostly 0x18181818.
- OutputView:** Shows the execution ending with an instruction count of 216 and an elapsed time of 00:00:00.3993240. It also shows instructions per second as 540.

## Microprocessor and Computer Architecture

UE22CS251B

4th Semester, Academic Year 2023-24

Date:

Name: Harshit Chandak

SRN:  
PES2UG24CS185

Section  
2

--	--	--

LAB # 3

### Assignment Question 2

Title of the Program

Write an ALP using ARM7TDMI to find the length of string

- I. ARM Assembly Code
- II. Output Screen Shots  
*(One Screenshot including Register Window, Memory Window and Code Window)*

**.DATA**

**A: .ASCIZ "HARSHITCHANDAK"**

**COUNT: .WORD 0**

**.TEXT**

**LDR R0,=A**

**MOV R3,#0**

**LOOP:**

**LDRB R1,[R0],#1**

**CMP R1,#0**

**BEQ EXIT**

**ADD R3,R3,#1**

**BNE LOOP**

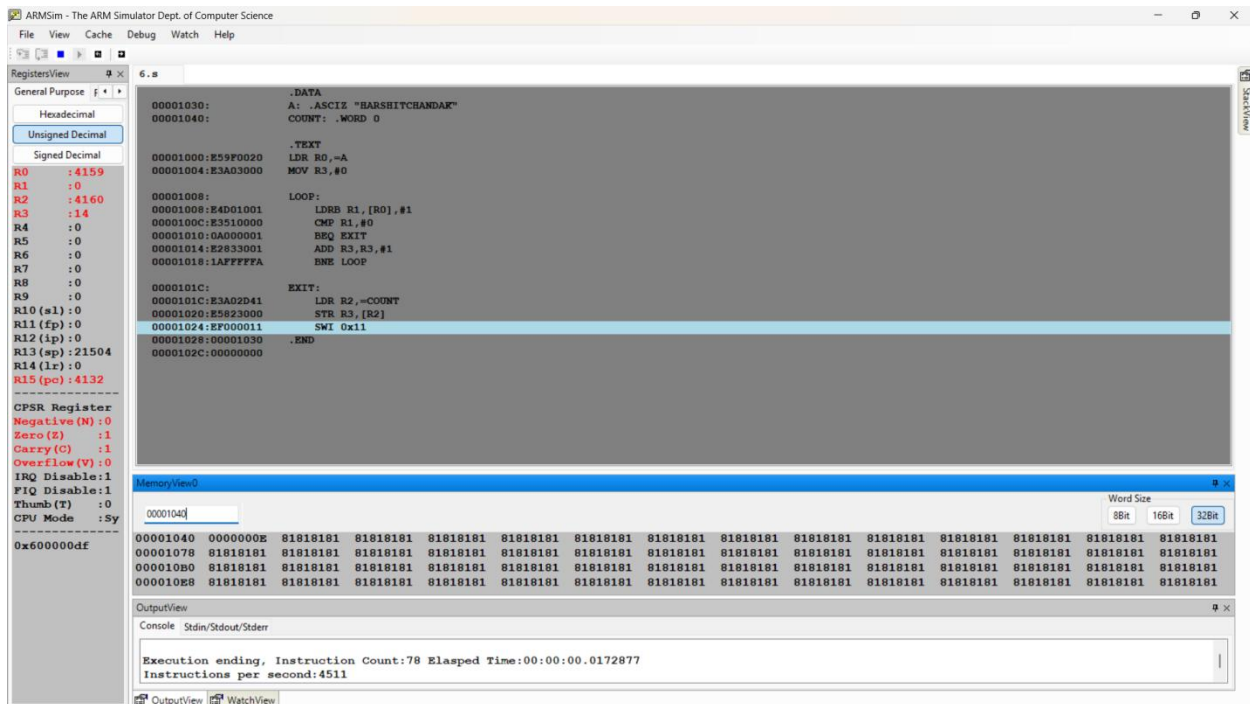
**EXIT:**

**LDR R2,=COUNT**

**STR R3,[R2]**

**SWI 0x11**

**.END**



### **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:

Name:

SRN:

Section:

Date: