



**Department of Computer Science &
Engineering
Microprocessor & Computer
Architecture
MPCA-
Laboratory/Assignment/Hands-
on/Project
UE20CS252**

**NAME : VISHWAS M
SEC : F
SRN : PES2UG20CS390
DATE : 31/01/2022**

1. Write a program in ARM7TDMI-ISA to copy a block of N data items from Location A to Location B.
 - a. Use Full word (.word directive)
 - b. Use Half word(.Hword directive)
 - c. Use Byte wise (.Byte directive)

Screenshots:

a) Using full word.

The screenshot shows the ARM Simulator interface with the following components:

- RegistersView:** Displays the state of registers R0 through R15. R2 is highlighted with a value of 4228. R15 (PC) is 70656.
- CodeView:** Shows the assembly code for a program named 'copyw.o'. The code includes:
 - .DATA:** Defines location A as a word containing the sequence 23, 43, 65, 34, 76, 87, 55, 36, 55, 98. Location B is defined as a word containing ten zeros.
 - .TEXT:** Contains the main logic:
 - Initialization: LDR R1, =A and LDR R2, =B.
 - Loop Setup: MOV R5, #1.
 - Loop Body: LDR R3, [R1] (backward), STR R3, [R2] (forward), ADD R1, R1, #4, and ADD R2, R2, #4.
 - Count Increment: ADD R5, R5, #1.
 - Termination: CMP R5, #11, BNE L1, and SWI 0XD11.
- MemoryView:** Displays the memory contents. Address 0000105C is selected, showing a sequence of 16 bytes, all of which are 81818181.

b) using half word.

2. Write a program in ARM7TDMI-ISA to find the sum of N data items in the memory. Store the result in the memory location.

- Use Full word (.word directive)
 - Use Half word(.Hword directive)
 - Use Byte wise (.Byte directive)
- {1,2,3,4,5}

Screenshots:

a) using full word.

The screenshot displays the ARMSim# - The ARM Simulator interface. The main window shows assembly code for a program that calculates the sum of data items. The code is as follows:

```
// Program to transfer a block of data from location A to location B.
.DATA
0000102C:00000001      A: .WORD 1,4,3,6,4,7,4,8,6,10
:00000004
:00000003
:00000006
:00000004

.TEXT
00001000:E59F1020      LDR R1, =A      //INITIALIZATION OF THE BLOCK ADDRESSES

00001004:E3A05001      MOV R5,#1
00001008:E3A06000      MOV R6,#0
0000100C:E5913000      L1:  LDR R3, [R1]      //BACKWARD=PC=PC+4-OFFSET, FORWARD=PC=PC+4+OFFSET
00001010:E0866003      ADD R6, R6,R3
00001014:E2811004      ADD R1, R1,#4      //ADDRESS TO THE NEXT DATA
00001018:E2855001      ADD R5, R5, #1     // INCREMENT THE COUNT REGISTER
0000101C:E355000B      CMP R5, #11        //CHECK WHETHER ALL NUMBERS ARE ADDED
00001020:1AFFFFFFF9      BNE L1             // EKSE BRANCH TO L1 LOCATION
00001024:EF000011      SWI 0X011          //LOGICAL END OF THE PROGRAM
:00000000
```

The left sidebar shows the RegisterView with the following values:

Register	Value
R0	:0
R1	:0
R2	:0
R3	:10
R4	:0
R5	:11
R6	:53
R7	:0
R8	:0
R9	:0
R10(s1)	:0
R11(fp)	:0
R12(ip)	:0
R13(sp)	:70656
R14(lr)	:0
R15(pc)	:70656

The bottom section shows the CPSR Register with the following values:

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

The bottom right section shows the OutputView and WatchView, with the console output displaying the address 0x600000df.

b)Using half word.

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView X

General Purpose Floating Point

Hexadecimal
Unsigned Decimal
Signed Decimal

R0 :0
R1 :0
R2 :0
R3 :10
R4 :0
R5 :11
R6 :53
R7 :0
R8 :0
R9 :0
R10(s1):0
R11(fp):0
R12(ip):0
R13(sp):70656
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

CodeView X

sumh.o

```
// Program to transfer a block of data from location A to location B.
.DATA
0000102C:00040001      A: .HWORD 1,4,3,6,4,7,4,8,6,10
:00060003
:00070004
:00080004
:000A0006

.TEXT
00001000:E59F1020      LDR R1, =A      //INITIALIZATION OF THE BLOCK ADDRESSES

00001004:E3A05001      MOV R5,#1
00001008:E3A06000      MOV R6,#0
0000100C:E1D130B0      L1:  LDRH R3, [R1]      //BACKWARD=PC=PC+4-OFFSET, FORWARD=PC=PC+4+OFFSET
00001010:E0866003      ADD R6, R6,R3
00001014:E2811002      ADD R1, R1,#2      //ADDRESS TO THE NEXT DATA
00001018:E2855001      ADD R5, R5, #1      // INCREMENT THE COUNT REGISTER
0000101C:E355000B      CMP R5, #11        //CHECK WHETHER ALL NUMBERS ARE ADDED
00001020:1AFFFFF9      BNE L1              // EKSE BRANCH TO L1 LOCATION
00001024:EF000011      SWI 0X011           //LOGICAL END OF THE PROGRAM
:00000000
```

OutputView WatchView X

Console stdin/stdout/stderr

c)Using byte.

The screenshot displays the ARMSim# - The ARM Simulator interface. The main window shows the assembly code for a program named 'sumb.o'. The code is as follows:

```
// Program to transfer a block of data from location A to location B.
.DATA
0000102C:06030401      A: .byte 1,4,3,6,4,7,4,8,6,10
                   :08040704
                   :0A06

.TEXT
00001000:E59F1020      LDR R1, =A      //INITIALIZATION OF THE BLOCK ADDRESSES

00001004:E3A05001      MOV R5,#1
00001008:E3A06000      MOV R6,#0
0000100C:E5D13000      L1:  LDRB R3, [R1]      //BACKWARD=PC=PC+4-OFFSET, FORWARD=PC=PC+4+OFFSET
00001010:E0866003      ADD R6, R6,R3
00001014:E2811001      ADD R1, R1,#1      //ADDRESS TO THE NEXT DATA
00001018:E2855001      ADD R5, R5, #1     // INCREMENT THE COUNT REGISTER
0000101C:E355000B      CMP R5, #11        //CHECK WHETHER ALL NUMBERS ARE ADDED
00001020:1AFFFFF9      BNE L1             // EKSE BRANCH TO L1 LOCATION
00001024:EF000011      SWI 0X011         //LOGICAL END OF THE PROGRAM
                   :00000000
```

The left sidebar shows the RegistersView with the following values:

Register	Value
R0	:0
R1	:0
R2	:0
R3	:10
R4	:0
R5	:11
R6	:53
R7	:0
R8	:0
R9	:0
R10 (s1)	:0
R11 (fp)	:0
R12 (ip)	:0
R13 (sp)	:70656
R14 (lr)	:0
R15 (pc)	:4140

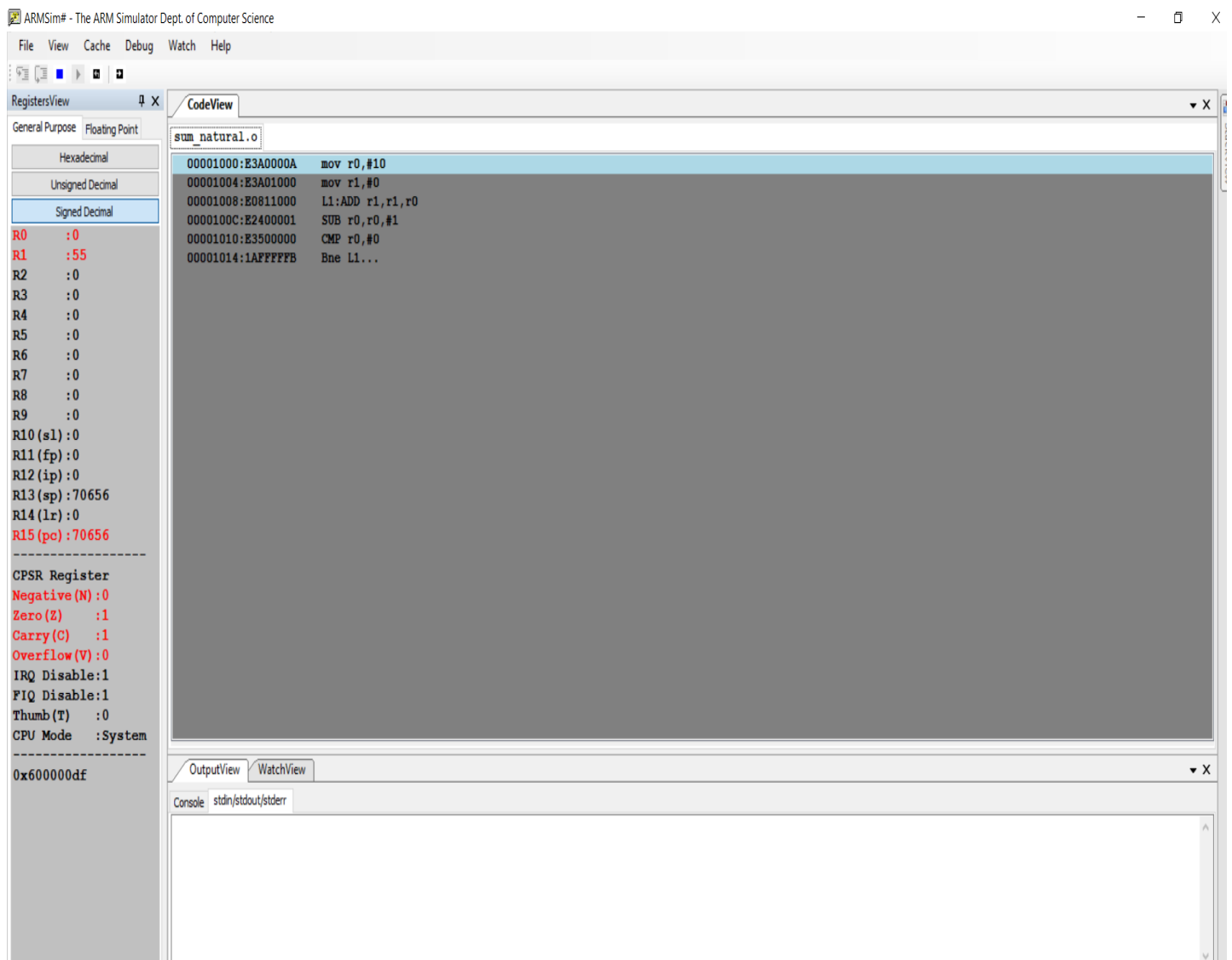
The CPSR Register shows the following status:

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

The bottom of the interface shows the OutputView and WatchView tabs, with the Console tab selected, displaying the address 0x600000df.

3. Write a program in ARM7TDMI-ISA to find the sum of N natural numbers. Store the result in the memory location.

Screenshot:



5. Convert the following statement in C language into an ALP using ARM7TDMI – ISA.
- IF([A]==[B]) then C=[A]+[B];

ELSE IF ([B]==[C]) D=[A]-[B];
ELSE E=[A]*[B]

Where A,B C, D & E are memory locations.

Screenshots:

a)when A and B are equal:

The screenshot displays the ARMSim# ARM Simulator interface. On the left, the 'RegistersView' pane shows the state of various registers. The 'General Purpose' registers R0 through R15 are listed, with R15 (PC) at address 70656. The 'CPSR Register' is also shown with fields like Negative (N), Zero (Z), Carry (C), and Overflow (V). The 'CPU Mode' is set to 'System'. The main 'CodeView' pane shows the assembly code for 'if_else_condition.o'. It includes a '.DATA' section with variables A, B, C, D, and E, each a 65-bit word. The '.TEXT' section contains assembly instructions: LDR R1, =A; LDR R2, =B; LDR R3, =C; LDR R4, =D; LDR R9, =E; LDR R5, [R1]; LDR R6, [R2]; LDR R7, [R3]; LDR R8, [R4]; LDR R10, [R9]; CMP R5, R6; BEQ L1; CMP R6, R7; BEQ L2; MUL R10, R5, R6; B L3; L1: ADDS R7, R5, R6; B L3; L2: SUBS R8, R5, R6; L3: SWI 0X011. The 'OutputView' and 'WatchView' panes are empty.

```
if_else_condition.o

.DATA
00001064:00000041      A: .WORD 65
00001068:00000041      B: .WORD 65
0000106C:0000002D      C: .WORD 45
00001070:00000000      D: .WORD 0
00001074:00000000      E: .WORD 0

.TEXT
00001000:E59F1048      LDR R1, =A
00001004:E59F2048      LDR R2, =B
00001008:E59F3048      LDR R3, =C
0000100C:E59F4048      LDR R4, =D
00001010:E59F9048      LDR R9, =E
00001014:E5915000      LDR R5, [R1]
00001018:E5926000      LDR R6, [R2]
0000101C:E5937000      LDR R7, [R3]
00001020:E5948000      LDR R8, [R4]
00001024:E599A000      LDR R10, [R9]
00001028:E1550006      CMP R5, R6
0000102C:0A000003      BEQ L1
00001030:E1560007      CMP R6, R7
00001034:0A000003      BEQ L2
00001038:E00A0695      MUL R10, R5, R6
0000103C:EA000002      B L3
00001040:E0957006      L1: ADDS R7, R5, R6
00001044:EA000000      B L3
00001048:E0558006      L2: SUBS R8, R5, R6
0000104C:EF000011      L3: SWI 0X011
00001050:00000000
```

b) when B and C are equal:

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView Floating Point

General Purpose Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 0

R1 : 4196

R2 : 4200

R3 : 4204

R4 : 4208

R5 : 45

R6 : 65

R7 : 65

R8 : -20

R9 : 4212

R10 (s1) : 0

R11 (fp) : 0

R12 (ip) : 0

R13 (sp) : 70656

R14 (lr) : 0

R15 (pc) : 70656

CPSR Register

Negative (N) : 1

Zero (Z) : 0

Carry (C) : 0

Overflow (V) : 0

IRQ Disable : 1

FIQ Disable : 1

Thumb (T) : 0

CPU Mode : System

0x800000df

CodeView

if_else_condition.o

```

.DATA
00001064:0000002D      A: .WORD 45
00001068:00000041      B: .WORD 65
0000106C:00000041      C: .WORD 65
00001070:00000000      D: .WORD 0
00001074:00000000      E: .WORD 0

.TEXT
00001000:E59F1048      LDR R1, =A
00001004:E59F2048      LDR R2, =B
00001008:E59F3048      LDR R3, =C
0000100C:E59F4048      LDR R4, =D
00001010:E59F9048      LDR R9, =E
00001014:E5915000      LDR R5, [R1]
00001018:E5926000      LDR R6, [R2]
0000101C:E5937000      LDR R7, [R3]
00001020:E5948000      LDR R8, [R4]
00001024:E59A0000      LDR R10, [R9]
00001028:E1550006      CMP R5, R6
0000102C:0A000003      BEQ L1
00001030:E1560007      CMP R6, R7
00001034:0A000003      BEQ L2
00001038:E00A0695      MUL R10, R5, R6
0000103C:EAD00002      B L3
00001040:E0957006      L1: ADDS R7, R5, R6
00001044:EAD00000      B L3
00001048:E0558006      L2: SUBS R8, R5, R6
0000104C:EF000011      L3: SWI 0X011

00001050:00000000

```

OutputView WatchView

Console stdin/stdout/stderr

c) When A,B,C are different:

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView Floating Point

General Purpose Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 0

R1 : 4196

R2 : 4200

R3 : 4204

R4 : 4208

R5 : 45

R6 : 55

R7 : 65

R8 : 0

R9 : 4212

R10 (s1) : 2475

R11 (fp) : 0

R12 (ip) : 0

R13 (sp) : 70656

R14 (lr) : 0

R15 (pc) : 70656

CPSR Register

Negative (N) : 1

Zero (Z) : 0

Carry (C) : 0

Overflow (V) : 0

IRQ Disable : 1

FIQ Disable : 1

Thumb (T) : 0

CPU Mode : System

0x800000df

CodeView

if_else_condition.o

```

.DATA
00001064:0000002D      A: .WORD 45
00001068:00000037      B: .WORD 55
0000106C:00000041      C: .WORD 65
00001070:00000000      D: .WORD 0
00001074:00000000      E: .WORD 0

.TEXT
00001000:E59F1048      LDR R1, =A
00001004:E59F2048      LDR R2, =B
00001008:E59F3048      LDR R3, =C
0000100C:E59F4048      LDR R4, =D
00001010:E59F9048      LDR R9, =E
00001014:E5915000      LDR R5, [R1]
00001018:E5926000      LDR R6, [R2]
0000101C:E5937000      LDR R7, [R3]
00001020:E5948000      LDR R8, [R4]
00001024:E59A0000      LDR R10, [R9]
00001028:E1550006      CMP R5, R6
0000102C:0A000003      BEQ L1
00001030:E1560007      CMP R6, R7
00001034:0A000003      BEQ L2
00001038:E00A0695      MUL R10, R5, R6
0000103C:EAD00002      B L3
00001040:E0957006      L1: ADDS R7, R5, R6
00001044:EAD00000      B L3
00001048:E0558006      L2: SUBS R8, R5, R6
0000104C:EF000011      L3: SWI 0X011

00001050:00000000

```

OutputView WatchView

Console stdin/stdout/stderr

6. Write a program in ARM7TDMI-ISA to find the factorial of a number.

Screenshot:

ARMSim# - The ARM Simulator Dept. of Computer Science

File View Cache Debug Watch Help

RegistersView CodeView

General Purpose Floating Point

Hexadecimal
Unsigned Decimal
Signed Decimal

R0 : 0
R1 : 0
R2 : 0
R3 : 5040
R4 : 4152
R5 : 0
R6 : 0
R7 : 0
R8 : 0
R9 : 0
R10 (s1) : 0
R11 (fp) : 0
R12 (ip) : 0
R13 (sp) : 70656
R14 (lr) : 0
R15 (pc) : 4152

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

factorial.o

```
.DATA
00001034:00000007      A: .WORD 7
00001038:00000000      FACT: .WORD 0

.TEXT
00001000:E59F1024      LDR R1, =A
00001004:E5912000      LDR R2, [R1]
00001008:E59F4020      LDR R4, =FACT
0000100C:E3A03001      MOV R3,#1

00001010:E3520000      LOOP: CMP R2,#0
00001014:0A000002      BEQ EXIT
00001018:E0030392      MUL R3,R2,R3
0000101C:E2422001      SUB R2,R2,#1
00001020:1AFFFFFA      BNE LOOP

00001024:E5843000      EXIT: STR R3,[R4]
00001028:EF000011      SWI 0X011...
          :00000000
          :00000004
```

OutputView WatchView

Console stdin/stdout/stderr

Execute question 1,2,3,5,6