

Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:

Name: Aditya NG	SRN: PES1UG19CS032	Section A
-----------------	-----------------------	------------------

Week# 1 Program Number: 1

Title of the Program

Write an ALP using ARM instruction set to add and subtract two 32 bit numbers .Both numbers are in registers.

ARM Assembly Code

MOV R0, #10

MOV R1, #05

ADD R2, R0, R1

SUB R3, R0, R1

SWI 0X011

Final Output

Case 1

The screenshot shows the ARM Simulator interface. The 'RegistersView' pane on the left displays the state of 16 registers (R0-R15) and CPSR. R15 (PC) is highlighted in red and shows the value 0x0001000c. The 'CodeView' pane on the right shows the assembly code for 'Week1_P1.o'. The code includes two addition programs and two subtraction programs, each with three instructions. The current instruction being executed is 'MOV R0, #23' at address 0000100C.

Register	Value (Hex)
R0	0000000a
R1	00000014
R2	00000014
R3	00000000
R4	00000000
R5	00000000
R6	00000000
R7	00000000
R8	00000000
R9	00000000
R10 (a1)	00000000
R11 (fp)	00000000
R12 (ip)	00000000
R13 (sp)	00011400
R14 (lr)	00000000
R15 (pc)	0000100c

```

.TEXT
/*ADDITION PROGRAM #1*/
00001000:E3A0000A  MOV R0, #10
00001004:E3A01014  MOV R1, #20
00001008:E0812002  ADD R2, R1, R2
/*ADDITION PROGRAM #2*/
0000100C:E3A00017  MOV R0, #23
00001010:E3A01032  MOV R1, #50
00001014:E0812002  ADD R2, R1, R2
/*SUBTRACTION PROGRAM #1*/
00001018:E3A0000A  MOV R0, #10
0000101C:E3A01014  MOV R1, #20
00001020:E0412000  SUB R2, R1, R0
00001024:E0403001  SUB R3, R0, R1
/*SUBTRACTION PROGRAM #2*/
00001028:E3A0000D  MOV R0, #13
0000102C:E3A01022  MOV R1, #34
00001030:E0412002  SUB R2, R1, R2
.END...

```

Case 2

The screenshot shows the ARM Simulator interface. The 'RegistersView' pane on the left displays the state of 16 registers (R0-R15) and CPSR. R15 (PC) is highlighted in red and shows the value 0x0001018. The 'CodeView' pane on the right shows the assembly code for 'Week1_P1.o'. The code includes two addition programs and two subtraction programs, each with three instructions. The current instruction being executed is 'MOV R0, #10' at address 00001018.

Register	Value (Hex)
R0	00000017
R1	00000032
R2	00000046
R3	00000000
R4	00000000
R5	00000000
R6	00000000
R7	00000000
R8	00000000
R9	00000000
R10 (a1)	00000000
R11 (fp)	00000000
R12 (ip)	00000000
R13 (sp)	00011400
R14 (lr)	00000000
R15 (pc)	00001018

```

.TEXT
/*ADDITION PROGRAM #1*/
00001000:E3A0000A  MOV R0, #10
00001004:E3A01014  MOV R1, #20
00001008:E0812002  ADD R2, R1, R2
/*ADDITION PROGRAM #2*/
0000100C:E3A00017  MOV R0, #23
00001010:E3A01032  MOV R1, #50
00001014:E0812002  ADD R2, R1, R2
/*SUBTRACTION PROGRAM #1*/
00001018:E3A0000A  MOV R0, #10
0000101C:E3A01014  MOV R1, #20
00001020:E0412000  SUB R2, R1, R0
00001024:E0403001  SUB R3, R0, R1
/*SUBTRACTION PROGRAM #2*/
00001028:E3A0000D  MOV R0, #13
0000102C:E3A01022  MOV R1, #34
00001030:E0412002  SUB R2, R1, R2
.END...

```

Case 3

RegistersView

General Purpose

Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 0000000a
R1 : 00000014
R2 : 0000000a
R3 : ffffffff6
R4 : 00000000
R5 : 00000000
R6 : 00000000
R7 : 00000000
R8 : 00000000
R9 : 00000000
R10 (s1): 00000000
R11 (fp): 00000000
R12 (ip): 00000000
R13 (sp): 00011400
R14 (lr): 00000000
R15 (pc): 00001028

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

CodeView

Week1_P1.o

.TEXT

/*ADDITION PROGRAM #1*/

00001000:E3A0000A MOV R0, #10

00001004:E3A01014 MOV R1, #20

00001008:E0812002 ADD R2, R1, R2

/*ADDITION PROGRAM #2*/

0000100C:E3A00017 MOV R0, #23

00001010:E3A01032 MOV R1, #50

00001014:E0812002 ADD R2, R1, R2

/*SUBTRACTION PROGRAM #1*/

00001018:E3A0000A MOV R0, #10

0000101C:E3A01014 MOV R1, #20

00001020:E0412000 SUB R2, R1, R0

00001024:E0403001 SUB R3, R0, R1

/*SUBTRACTION PROGRAM #2*/

00001028:E3A0000D MOV R0, #13

0000102C:E3A01022 MOV R1, #34

00001030:E0412002 SUB R2, R1, R2

.END...

Case 4

RegistersView

General Purpose

Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 0000000d
R1 : 00000022
R2 : 00000018
R3 : ffffffff6
R4 : 00000000
R5 : 00000000
R6 : 00000000
R7 : 00000000
R8 : 00000000
R9 : 00000000
R10 (s1): 00000000
R11 (fp): 00000000
R12 (ip): 00000000
R13 (sp): 00011400
R14 (lr): 00000000
R15 (pc): 00001034

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

CodeView

Week1_P1.o

.TEXT

/*ADDITION PROGRAM #1*/

00001000:E3A0000A MOV R0, #10

00001004:E3A01014 MOV R1, #20

00001008:E0812002 ADD R2, R1, R2

/*ADDITION PROGRAM #2*/

0000100C:E3A00017 MOV R0, #23

00001010:E3A01032 MOV R1, #50

00001014:E0812002 ADD R2, R1, R2

/*SUBTRACTION PROGRAM #1*/

00001018:E3A0000A MOV R0, #10

0000101C:E3A01014 MOV R1, #20

00001020:E0412000 SUB R2, R1, R0

00001024:E0403001 SUB R3, R0, R1

/*SUBTRACTION PROGRAM #2*/

00001028:E3A0000D MOV R0, #13

0000102C:E3A01022 MOV R1, #34

00001030:E0412002 SUB R2, R1, R2

.END...

Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:

Name:	SRN:	Section

Week# ____1____ Program Number: ____2____

Title of the Program

Write an ALP to demonstrate logical operations. All operands are in registers.

ARM Assembly Code

MOV R0, #5

MOV R1, #6

AND R2, R0, R1

ORR R3, R0, R1

EOR R4, R0, R1

MVN R5, R0

Final Output

Case 1

RegistersView

General Purpose

Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 00000005

R1 : 00000006

R2 : 00000004

R3 : 00000007

R4 : 00000003

R5 : ffffffff

R6 : ffffffff

R7 : 00000014

R8 : 00000004

R9 : 0000001c

R10 (s1) : 00000000

R11 (fp) : 00000000

R12 (ip) : 00000000

R13 (sp) : 00011400

R14 (lr) : 00000000

R15 (pc) : 00001030

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

CodeView

Week1_P2.o

.TEXT

00001000:E3A00005

MOV R0, #5

00001004:E3A01006

MOV R1, #6

00001008:E3A0600C

MOV R6, #12

0000100C:E3A07014

MOV R7, #20

/*AND Operation*/

00001010:E0002001

AND R2, R0, R1

00001014:E0068007

AND R8, R6, R7

/*OR Operation*/

00001018:E1803001

ORR R3, R0, R1

0000101C:E1869007

ORR R9, R6, R7

/*EX-OR Operation*/

00001020:E0204001

EOR R4, R0, R1

00001024:E0266007

EOR R6, R6, R7

/*NOT Operation*/

00001028:E1E05000

MVN R5, R0

0000102C:E1E06006

MVN R6, R6

.END...

Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:

Name:	SRN:	Section

Week# ____1____ Program Number: ____3____

Title of the Program

Write an ALP to add 5 numbers where values are present in registers.

ARM Assembly Code

MOV R0, #5

MOV R1, #6

MOV R2, #7

MOV R3, #6

MOV R4, #15

ADD R5, R0, R1

ADD R5, R2, R5

ADD R5, R3, R5

ADD R5, R4, R5

MOV R0, #43

MOV R1, #3

MOV R2, #9

MOV R3, #6

MOV R4, #1

ADD R5, R0, R1

ADD R5, R2, R5

ADD R5, R3, R5

ADD R5, R4, R5

Final Output

Case 1

The screenshot displays a debugger interface with two main windows: RegistersView and CodeView.

RegistersView: This window shows the state of the processor registers. The registers R0 through R15 are listed, each with its current value in hexadecimal. R0 is 00000005, R1 is 00000006, R2 is 00000007, R3 is 00000006, R4 is 0000000f, R5 is 0000000b, R6 is 00000012, R7 is 00000018, R8 is 00000027, R9 is 00000000, R10 (s1) is 00000000, R11 (fp) is 00000000, R12 (ip) is 00000000, R13 (sp) is 00011400, R14 (lr) is 00000000, and R15 (pc) is 00001024. Below the registers, the CPSR (Current Program Status Register) is shown with various flags: Negative (N) is 0, Zero (Z) is 0, Carry (C) is 0, Overflow (V) is 0, IRQ Disable is 1, FIQ Disable is 1, Thumb (T) is 0, and CPU Mode is System. The value 0x000000df is also displayed.

CodeView: This window shows the assembly code for the program. The code is organized into sections: .TEXT, .DATA, and .BSS. The .TEXT section contains the following instructions:

```
00001000:E3A00005  MOV R0, #5
00001004:E3A01006  MOV R1, #6
00001008:E3A02007  MOV R2, #7
0000100C:E3A03006  MOV R3, #6
00001010:E3A0400F  MOV R4, #15

/*ADD 5 NUMBERS*/
00001014:E0805001  ADD R5, R0, R1
00001018:E0856002  ADD R6, R5, R2
0000101C:E0867003  ADD R7, R6, R3
00001020:E0878004  ADD R8, R7, R4

/*EXAMPLE #2*/
00001024:E3A0002B  MOV R0, #43
00001028:E3A01003  MOV R1, #3
0000102C:E3A02009  MOV R2, #9
00001030:E3A03006  MOV R3, #6
00001034:E3A04001  MOV R4, #1

/*ADD 5 NUMBERS*/
00001038:E0805001  ADD R5, R0, R1
0000103C:E0856002  ADD R6, R5, R2
00001040:E0867003  ADD R7, R6, R3
00001044:E0878004  ADD R8, R7, R4

.END...
```

Case 2

The image shows a debugger interface with two main windows: **RegistersView** and **CodeView**.

RegistersView: The **General Purpose** tab is selected. The **Signed Decimal** format is chosen. The registers R0 through R15 are listed with their values. R8 is highlighted in red. Below the registers, the **CPSR Register** is shown with the following values:

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

The value **0x000000df** is displayed at the bottom of the register list.

CodeView: The file **Week1_P3.o** is open. The **.TEXT** section is displayed, showing assembly code. The code includes several **MOV** and **ADD** instructions, along with comments like **/*ADD 5 NUMBERS*/** and **/*EXAMPLE #2*/**. The code ends with **.END...**

```
.TEXT
00001000:E3A00005    MOV R0, #5
00001004:E3A01006    MOV R1, #6
00001008:E3A02007    MOV R2, #7
0000100C:E3A03006    MOV R3, #6
00001010:E3A0400F    MOV R4, #15

/*ADD 5 NUMBERS*/
00001014:E0805001    ADD R5, R0, R1
00001018:E0856002    ADD R6, R5, R2
0000101C:E0867003    ADD R7, R6, R3
00001020:E0878004    ADD R8, R7, R4

/*EXAMPLE #2*/
00001024:E3A0002B    MOV R0, #43
00001028:E3A01003    MOV R1, #3
0000102C:E3A02009    MOV R2, #9
00001030:E3A03006    MOV R3, #6
00001034:E3A04001    MOV R4, #1

/*ADD 5 NUMBERS*/
00001038:E0805001    ADD R5, R0, R1
0000103C:E0856002    ADD R6, R5, R2
00001040:E0867003    ADD R7, R6, R3
00001044:E0878004    ADD R8, R7, R4

.END...
```


Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:

Name:	SRN:	Section

Week# 1

Program Number: 4

Title of the Program

Write an ALP using ARM instruction set to check if a number stored in a register is even or odd. If even, store 00 in R0, else store FF in R0

ARM Assembly Code

MOV R1, #6

ANDS R2, R1, #1

CMP R2, #0

BEQ L1

MOV R0, #00

B L2

L1: MOV R2, #0XFF

L2: SWI 0X011

Final Output

Case 1

RegistersView	
General Purpose	Floating Point
Hexadecimal	
Unsigned Decimal	
Signed Decimal	
R0	: 000000ff
R1	: 00000006
R2	: 00000000
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)	: 00011400
R14 (lr)	: 00000000
R15 (pc)	: 00001020

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	
Week1_P4.o	
.TEXT	
00001000:E3A01006	MOV R1, #6
00001004:E2112001	ANDS R2, R1, #1
00001008:E3520000	CMP R2, #0
0000100C:0A000001	BEQ Location1
00001010:E3A00000	MOV R0, #00
00001014:EA000000	B Location2
00001018:E3A000FF	Location1: MOV R0, #0xFF
0000101C:EF000011	Location2: SWI 0x011
.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: NG
Name: Aditya NG
SRN: PES1UG19CS032
Section: L
Date: 27-1-2020