

Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:

Name: TUSHAR Y S	SRN: PES1UG19CS545	Section I
------------------	-----------------------	--------------

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:

.text

;adding 2 numbers

MOV R0,#10

MOV R1,#20

ADD R2,R0,R1

;subtracting 2 numbers

MOV R0,#10

MOV R1,#20

SUB R3,R1,R0

SUB R4,R0,R1

.end

Output:

The screenshot displays a debugger window with two main panes. The left pane, titled 'RegistersView', shows the state of various registers. The right pane, titled 't1.s', displays the assembly code being executed.

RegistersView:

- General Purpose: Floating Point
- Hexadecimal (selected)
- Unsigned Decimal
- Signed Decimal
- R0: 0000000a
- R1: 00000014
- R2: 0000001e
- R3: 0000000a
- R4: ffffffff6
- R5: 00000000
- R6: 00000000
- R7: 00000000
- R8: 00000000
- R9: 00000000
- R10 (s1): 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

t1.s:

```
.text
;adding 2 numbers
00001000:E3A0000A    MOV R0,#10
00001004:E3A01014    MOV R1,#20
00001008:E0802001    ADD R2,R0,R1

;subtracting 2 numbers
0000100C:E3A0000A    MOV R0,#10
00001010:E3A01014    MOV R1,#20
00001014:E0413000    SUB R3,R1,R0
00001018:E0404001    SUB R4,R0,R1
.end
```

OutputView:

Console Stdin/Stdout/Stderr

Execution ending, Instruction Count:0 Elapsed Time:00:00:00.2122845
Instructions per second:0

Test case:

.text

;adding 2 numbers

MOV R0,#2

MOV R1,#4

ADD R2,R0,R1

;subtracting 2 numbers

MOV R0,#2

MOV R1,#4

SUB R3,R1,R0

SUB R4,R0,R1

.end

Output:

RegistersView

General PurposeFloating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0:00000002

R1:00000004

R2:00000006

R3:00000002

R4:fffffffe

R5:00000000

R6:00000000

R7:00000000

R8:00000000

R9:00000000

R10(s1):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

t1a.s

.text

;adding 2 numbers

00001000:E3A00002MOV R0,#2

00001004:E3A01004MOV R1,#4

00001008:E0802001ADD R2,R0,R1

;subtracting 2 numbers

0000100C:E3A00002MOV R0,#2

00001010:E3A01004MOV R1,#4

00001014:E0413000SUB R3,R1,R0

00001018:E0404001SUB R4,R0,R1

.end

OutputView

ConsoleStdin/Stdout/Stderr

Execution ending, Instruction Count:0 Elapsed Time:00:00:00.2239820

Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:

Name: TUSHAR Y S	SRN: PES1UG19CS545	Section I
------------------	-----------------------	--------------

Week# ____1____ Program Number: ____2____

Title of the Program

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

ARM Assembly Code:

.text

;To demonstrate logical operations

MOV R0,#5

MOV R1,#6

AND R2,R0,R1 ;AND operation

ORR R3,R0,R1 ;OR operation

EOR R4,R0,R1 ;EX-OR operation

MVN R5,R0 ;NOT operation

.end

Output:

The screenshot shows a debugger window with two main panes. The left pane, titled 'RegistersView', displays the state of 16 ARM registers (R0-R15) and the CPSR register. The right pane, titled 't2.s', shows the assembly code being executed.

RegistersView:

- General Purpose: Floating Point
- Hexadecimal: (selected)
- Unsigned Decimal
- Signed Decimal
- R0: 00000005
- R1: 00000006
- R2: 00000004
- R3: 00000007
- R4: 00000003
- R5: ffffffff
- R6: 00000000
- R7: 00000000
- R8: 00000000
- R9: 00000000
- R10 (s1): 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

t2.s:

```
.text
;To demonstrate logical operations
00001000:E3A00005    MOV R0,#5
00001004:E3A01006    MOV R1,#6
00001008:E0002001    AND R2,R0,R1 ;AND operation
0000100C:E1803001    ORR R3,R0,R1 ;OR operation
00001010:E0204001    EOR R4,R0,R1 ;EX-OR operation
00001014:E1E05000    MVN R5,R0    ;NOT operation
.end
```

OutputView:

Console Stdin/Stdout/Stderr

Execution ending, Instruction Count:0 Elapsed Time:00:00:00.2210657
Instructions per second:0

Test case:

.text

;To demonstrate logical operations

 MOV R0,#4

 MOV R1,#8

AND R2,R0,R1 ;AND operation

ORR R3,R0,R1 ;OR operation

EOR R4,R0,R1 ;EX-OR operation

MVN R5,R0 ;NOT operation

.end

Output:

The screenshot displays an ARM assembly editor interface. On the left, the 'RegistersView' window shows the state of 16 registers (R0-R15) and the CPSR register. R0 is 00000004, R1 is 00000008, R2 is 00000000, R3 is 0000000c, R4 is 0000000c, R5 is ffffffff, and R15 (PC) is 00011400. The CPSR register shows flags: Negative (N): 0, Zero (Z): 0, Carry (C): 0, Overflow (V): 0, IRQ Disable: 1, FIQ Disable: 1, Thumb (T): 0, and CPU Mode: System. The main editor window shows the assembly code for 't2a.s', which includes instructions for MOV, AND, ORR, EOR, and MVN, followed by a .end directive. The output window at the bottom is empty.

Register	Value
R0	00000004
R1	00000008
R2	00000000
R3	0000000c
R4	0000000c
R5	fffffff
R6	00000000
R7	00000000
R8	00000000
R9	00000000
R10 (s1)	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

```
.text
;To demonstrate logical operations
00001000:E3A00004    MOV R0,#4
00001004:E3A01008    MOV R1,#8

00001008:E0002001    AND R2,R0,R1 ;AND operation
0000100C:E1803001    ORR R3,R0,R1 ;OR operation
00001010:E0204001    EOR R4,R0,R1 ;EX-OR operation
00001014:E1E05000    MVN R5,R0 ;NOT operation

.end
```

Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:

Name: TUSHAR Y S	SRN: PES1UG19CS545	Section I
------------------	-----------------------	--------------

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:

.text

;Adding 5 numbers

MOV R0,#5

MOV R1,#6

MOV R2,#7

MOV R3,#6

MOV R4,#15

ADD R5,R0,R1

ADD R5,R5,R2

ADD R5,R5,R3

ADD R5,R5,R4

.end

Output:

The screenshot displays a debugger interface with two main panes. The left pane, titled 'RegistersView', shows the state of various registers. The right pane, titled 't3.s', shows the assembly code being executed.

RegistersView:

- General Purpose: Floating Point
- Hexadecimal
- Unsigned Decimal
- Signed Decimal
- R0: 00000005
- R1: 00000006
- R2: 00000007
- R3: 00000006
- R4: 0000000f
- R5: 00000027
- R6: 00000000
- R7: 00000000
- R8: 00000000
- R9: 00000000
- R10 (s1): 00000000
- R11 (fp): 00000000
- R12 (ip): 00000000
- R13 (sp): 00005400
- R14 (lr): 00000000
- R15 (pc): 00011400

Assembly Code (t3.s):

```
.text
;Adding 5 numbers
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

00001014:E0805001    ADD R5,R0,R1
00001018:E0855002    ADD R5,R5,R2
0000101C:E0855003    ADD R5,R5,R3
00001020:E0855004    ADD R5,R5,R4

.end
```

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

Test Case:

.text

;Adding 5 numbers

MOV R0,#1

MOV R1,#2

MOV R2,#3

MOV R3,#4

MOV R4,#5

ADD R5,R0,R1

ADD R5,R5,R2

ADD R5,R5,R3

ADD R5,R5,R4

.end

Output:

RegistersView

General Purpose

Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 00000001

R1 : 00000002

R2 : 00000003

R3 : 00000004

R4 : 00000005

R5 : 0000000f

R6 : 00000000

R7 : 00000000

R8 : 00000000

R9 : 00000000

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

t3a.s

.text

;Adding 5 numbers

00001000:E3A00001 MOV R0,#1

00001004:E3A01002 MOV R1,#2

00001008:E3A02003 MOV R2,#3

0000100C:E3A03004 MOV R3,#4

00001010:E3A04005 MOV R4,#5

00001014:E0805001 ADD R5,R0,R1

00001018:E0855002 ADD R5,R5,R2

0000101C:E0855003 ADD R5,R5,R3

00001020:E0855004 ADD R5,R5,R4

.end

Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:

Name: TUSHAR Y S	SRN: PES1UG19CS545	Section I
------------------	-----------------------	--------------

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:

Case 1:

.text

MOV R1,#6

ANDs R2,R1,#1

BEQ Location1

MOV R0,#0xFF

B quit

Location1:

MOV R0,#0x00

quit:

.end

Output:

The screenshot displays a debugger interface with two main panels. The left panel, titled 'RegistersView', shows the state of various registers. The right panel, titled 't4.s', shows the assembly code being executed.

RegistersView Panel:

- General Purpose: Floating Point
- Hexadecimal
- Unsigned Decimal
- Signed Decimal
- R0 : 00000000
- R1 : 00000006
- R2 : 00000000
- R3 : 00000000
- R4 : 00000000
- R5 : 00000000
- R6 : 00000000
- R7 : 00000000
- R8 : 00000000
- R9 : 00000000
- R10 (s1) : 00000000
- R11 (fp) : 00000000
- R12 (ip) : 00000000
- R13 (sp) : 00005400
- R14 (lr) : 00000000
- R15 (pc) : 00011400
-
- CPSR Register
- Negative (N) : 0
- Zero (Z) : 1
- Carry (C) : 0
- Overflow (V) : 0
- IRQ Disable : 1
- FIQ Disable : 1
- Thumb (T) : 0
- CPU Mode : System
-
- 0x400000df

t4.s Panel:

```
.text
00001000:E3A01006    MOV R1,#6
00001004:E2112001    ANDs R2,R1,#1
00001008:0A000001    BEQ Location1
0000100C:E3A000FF    MOV R0,#0xFF
00001010:EA000000    B quit
00001014:                Location1:
00001014:E3A00000    MOV R0,#0x00
00001018:                quit:

.end
```

Case 2:

.text

MOV R1,#5

ANDs R2,R1,#1

BEQ Location1

MOV R0,#0xFF

B quit

Location1:

MOV R0,#0x00

quit:

.end

Output:

RegistersView

General Purpose
Floating Point

Hexadecimal
Unsigned Decimal
Signed Decimal

R0 : 000000ff
R1 : 00000005
R2 : 00000001
R3 : 00000000
R4 : 00000000
R5 : 00000000
R6 : 00000000
R7 : 00000000
R8 : 00000000
R9 : 00000000
R10 (s1) : 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

t4a.s

```

.text
00001000:E3A01005    MOV R1,#5
00001004:E2112001    ANDs R2,R1,#1
00001008:0A000001    BEQ Location1
0000100C:E3A000FF    MOV R0,#0xFF
00001010:EA000000    B quit
00001014:                Location1:
00001014:E3A00000    MOV R0,#0x00
00001018:                quit:

.end

```

Test cases:

Case 1:

.text

MOV R1,#10

ANDs R2,R1,#1

BEQ Location1

MOV R0,#0xFF

B quit

Location1:

MOV R0,#0x00

quit:

.end

Output:

The screenshot displays a debugger interface with two main panels. The left panel, titled 'RegistersView', shows the state of various registers. The right panel, titled 't4b.s', shows the assembly code being executed.

RegistersView Panel:

- General Purpose: Floating Point
- Hexadecimal
- Unsigned Decimal
- Signed Decimal
- R0 : 00000000
- R1 : 0000000a
- R2 : 00000000
- R3 : 00000000
- R4 : 00000000
- R5 : 00000000
- R6 : 00000000
- R7 : 00000000
- R8 : 00000000
- R9 : 00000000
- R10 (s1) : 00000000
- R11 (fp) : 00000000
- R12 (ip) : 00000000
- R13 (sp) : 00005400
- R14 (lr) : 00000000
- R15 (pc) : 00011400
-
- CPSR Register
- Negative (N) : 0
- Zero (Z) : 1
- Carry (C) : 0
- Overflow (V) : 0
- IRQ Disable: 1
- FIQ Disable: 1
- Thumb (T) : 0
- CPU Mode : System
-
- 0x400000df

t4b.s Panel:

```
.text
00001000:E3A0100A    MOV R1,#10
00001004:E2112001    ANDs R2,R1,#1
00001008:0A000001    BEQ Location1
0000100C:E3A000FF    MOV R0,#0xFF
00001010:EA000000    B quit
00001014:                Location1:
00001014:E3A00000    MOV R0,#0x00
00001018:                quit:

.end
```

Case 2:

.text

MOV R1,#19

ANDs R2,R1,#1

BEQ Location1

MOV R0,#0xFF

B quit

Location1:

MOV R0,#0x00

quit:

.end

Output:

The screenshot displays a debugger interface with two main panels. The left panel, titled 'RegistersView', shows the state of various registers and the CPSR. The right panel shows the assembly code for 't4c.s'.

RegistersView:

- General Purpose: Floating Point
- Hexadecimal
- Unsigned Decimal
- Signed Decimal
- R0: 000000ff
- R1: 00000013
- R2: 00000001
- R3: 00000000
- R4: 00000000
- R5: 00000000
- R6: 00000000
- R7: 00000000
- R8: 00000000
- R9: 00000000
- R10 (s1): 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

t4c.s:

```
.text
00001000:E3A01013    MOV R1,#19
00001004:E2112001    ANDs R2,R1,#1
00001008:0A000001    BEQ Location1
0000100C:E3A000FF    MOV R0,#0xFF
00001010:EA000000    B quit
00001014:                Location1:
00001014:E3A00000    MOV R0,#0x00
00001018:                quit:

.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: Tushar

Name: TUSHAR Y S

SRN: PES1UG19CS545

Section: I

Date: 26/1/2021