

Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date: 5/2/2021

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

Week# 3

Program Number: 1

Write an ALP to add two 64 bit numbers loaded from memory and store the result in memory.

ARM Assembly Code:

.text

LDR R0,=A

LDR R1,=B

LDR R2,[R0],#4

LDR R3,[R0]

LDR R4,[R1],#4

LDR R5,[R1]

ADDS R7,R3,R5

ADC R6,R2,R4

LDR R8,=C

STR R6,[R8],#4

STR R7,[R8]

LDR R9,=C

LDR R10,[R9,#4]

LDR R11,[R9]

SWI 0x11

.data

A:.WORD 0x00000001,0xa0000002

B:.WORD 0x00000003,0xa0000004

C:.WORD

Output:

The screenshot displays a debugger interface with three main panels: RegistersView, AssemblyView, and MemoryView2.

RegistersView: Shows the state of various registers. R0 through R9 are in hexadecimal. R10 (s1) is 40000006, R11 (fp) is 00000005, R12 (ip) is 00000000, R13 (sp) is 00005400, R14 (lr) is 00000000, and R15 (pc) is 00001038. The CPSR Register shows Negative (N) : 0, Zero (Z) : 0, Carry (C) : 1, Overflow (V) : 1, IRQ Disable : 1, FIQ Disable : 1, Thumb (T) : 0, and CPU Mode : System.

AssemblyView: Shows assembly code for file 1.s. The .text section includes instructions from 00001000 to 00001037, ending with SWI 0x11 at 00001038. The .data section starts at 0000104C with labels A, B, and C.

MemoryView2: Shows a memory dump starting at address 0000105C. The first row shows values 00000005, 40000006, and several 81818181s.

OutputView: Shows the Console tab selected, with no output visible.

Week#__3__

Program Number: __2__

Write an ALP to copy n numbers from Memory Location A to Memory Location B.

ARM Assembly Code:

.text

LDR R0,=A

LDR R1,=B

MOV R2,#10

L1:

LDR R3,[R0]

STR R3,[R1]

SUB R2,R2,#1

ADD R0,R0,#4

ADD R1,R1,#4

CMP R2,#0

BNE L1

SWI 0x11

.data

A:.WORD 2,4,6,40,10,12,14,16,18,20

B:.WORD

Output:

RegistersView

General Purpose

Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 0000105c
R1 : 00001084
R2 : 00000000
R3 : 00000014
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) : 00001028

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

2.s

```

.text
00001000:E59F0024      LDR R0,=A
00001004:E59F1024      LDR R1,=B
00001008:E3A0200A      MOV R2,#10
0000100C:              L1:
0000100C:E5903000      LDR R3,[R0]
00001010:E5813000      STR R3,[R1]
00001014:E2422001      SUB R2,R2,#1
00001018:E2800004      ADD R0,R0,#4
0000101C:E2811004      ADD R1,R1,#4
00001020:E3520000      CMP R2,#0
00001024:1AFFFFF8      BNE L1
00001028:EF000011      SWI 0x11

.data
00001034:              A: .WORD 2,4,6,40,10,12,14,16,18,20
0000105C:              B: .WORD

```

MemoryView2

0000105C

0000105C 00000002 00000004 00000006 00000028 0000000A 0000000C 0000000E 00000010 00000012 00000014 81818181 81818181
00001090 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010C4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010F8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181

OutputView

Week# 3 Program Number: 3

Write an ALP to find smallest number in an array of n – 32 bit numbers.

ARM Assembly Code:

.text

LDR R0,=A

LDR R1,=B

LDR R4,[R0],#4

LDR R3,[R1]

```

CMP R3,#1
BEQ L1
SUB R3,R3,#1
MOV R6,R4           ; Smallest number to be stored in R6
L2:
    LDR R4,[R0],#4
CMP R4,R6
MOVLT R6,R4
SUB R3,R3,#1
CMP R3,#0
BNE L2
L1:
    SWI 0x11

```

.data

```
A:.WORD 5,13,21,4,3,19,30,8,16,42
```

```
B:.WORD 10
```

Output:

RegistersView

General Purpose

Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 0000106c
R1 : 0000106c
R2 : 00000000
R3 : 00000000
R4 : 0000002a
R5 : 00000000
R6 : 00000003
R7 : 00000000
R8 : 00000000
R9 : 00000000
R10 (s1) : 00000000
R11 (fp) : 00000000
R12 (ip) : 00000000
R13 (sp) : 00005400
R14 (lr) : 00000000
R15 (pc) : 00001038

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

3.s

```

.text
00001000:E59F0034    LDR R0,=A
00001004:E59F1034    LDR R1,=B
00001008:E4904004    LDR R4,[R0],#4
0000100C:E5913000    LDR R3,[R1]
00001010:E3530001    CMP R3,#1
00001014:0A000007    BEQ L1
00001018:E2433001    SUB R3,R3,#1
0000101C:E1A06004    MOV R6,R4           ; Smallest number to be stored in R6
00001020:                L2:
00001020:E4904004    LDR R4,[R0],#4
00001024:E1540006    CMP R4,R6
00001028:B1A06004    MOVLT R6,R4
0000102C:E2433001    SUB R3,R3,#1
00001030:E3530000    CMP R3,#0
00001034:1AFFFFF9    BNE L2
00001038:                L1:
00001038:EF000011    SWI 0x11

.data
00001044:                A: .WORD 5,13,21,4,3,19,30,8,16,42
0000106C:                B: .WORD 10

```

MemoryView2

0000105c

0000105C 0000001E 00000008 00000010 0000002A 0000000A 81818181 81818181 81818181
00001090 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010C4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010F8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181

Week# 3 Program Number: 4a

Write an ALP to count the number of 1's and 0's in a given 32 bit number.

ARM Assembly Code:

.text

LDR R0,=0b11110010100101001101001100110111

MOV R1,#32

MOV R5,#0

MOV R6,#0

L1:

AND R2,R0,#1

CMP R2,#1

ADDEQ R5,R5,#1 ;Number of 1's is stored in R5

ADDNE R6,R6,#1 ;Number of 0's is stored in R6

MOV R0,R0,LSR #1

SUB R1,R1,#1

CMP R1,#0

BNE L1

SWI 0x11

.end

Output:

The screenshot displays a debugger interface with three main panels:

- RegistersView:** Shows the state of ARM registers. R0 through R15 are listed with their decimal values. R15 (PC) is 4144. The CPSR register is also shown with various flags like Negative (N), Zero (Z), Carry (C), Overflow (V), IRQ Disable, FIQ Disable, Thumb (T), and CPU Mode (System).
- Assembly View:** Displays the assembly code for file 4a.s. The code includes initialization of R0, R1, R5, and R6, followed by a loop labeled L1 that increments R5 and R6 based on the parity of R2, decrements R1, and branches back to L1 until R1 reaches 0. It concludes with a software interrupt (SWI 0x11) and an end-of-file marker.
- MemoryView2:** Shows a memory dump starting at address 0000105c. The memory contains a sequence of 81818181 values, which is a common pattern used for debugging or testing.

Week#____3_____

Program Number: __4b__

Write an ALP to find the number of zeroes, positive and negative numbers in a given array.

ARM Assembly Code:

.text

LDR R1,=A

LDR R2,=B

LDR R3,[R2]

MOV R7,#0

MOV R8,#0

MOV R9,#0

L1:

LDR R4,[R1],#4

CMP R4,#0

ADDEQ R7,R7,#1 ;Number of 0's is stored in R7

ADDLT R8,R8,#1 ;Number of negative numbers is stored in

R8

ADDGT R9,R9,#1 ;Number of positive numbers is stored in

R9

SUB R3,R3,#1

CMP R3,#0

BNE L1

SWI 0x11

.data

A:.WORD -1,-2,-3,0,1,2,0,4,5,6

B:.WORD 10

Output:

The screenshot displays a debugger window with the following components:

- RegistersView:** Shows the state of 16 registers. R0-R15 are listed with their values. R15 (pc) is highlighted in red. Below the registers, the CPSR Register status is shown: Negative(N):0, Zero(Z):1, Carry(C):1, Overflow(V):0, IRQ Disable:1, FIQ Disable:1, Thumb(T):0, and CPU Mode: System.
- Assembly View:** Displays assembly code for a file named '4b.s'. The code includes instructions like LDR, MOV, CMP, ADDEQ, ADDLT, ADDGT, SUB, and SWI. Comments explain the purpose of some instructions, such as counting zeros, negative numbers, and positive numbers.
- MemoryView:** Shows a memory dump starting at address 0000105C. The dump consists of a grid of hexadecimal values, with the first row containing zeros and subsequent rows containing the value 81818181.

Week#___3_____

Program Number: __5__

Write an ALP to check whether a given number is present in array using Linear Search (Without SWI 0x02), if found move +1 to R6 and key position to R7 else move -1 to R6 (if number not found).

Case 1(Element found):

ARM Assembly Code:

.text

LDR R0,=A

MOV R1,#10

MOV R3,R1

L1:

LDR R2,[R0],#4

CMP R2,#25 ;Element 25 is to be searched in the array

BEQ L2

SUBS R1,R1,#1

CMP R1,#0

BNE L1

MOV R6,#-1

SWI 0x11

L2:

MOV R6,#1

SUB R1,R1,#1

SUB R7,R3,R1

SWI 0x11

.data

A:.WORD 2,4,6,8,12,20,25,30,40,50

Output:

The screenshot displays a debugger interface with three main panels:

- RegistersView:** Shows the state of ARM registers. R0 is 4188, R1 is 3, R2 is 25, R3 is 10, and R15 (PC) is 4152. The CPSR register shows Negative (N) as 0, Zero (Z) as 1, Carry (C) as 1, and Overflow (V) as 0.
- Assembly View:** Displays ARM assembly code. The current instruction is `SWI 0x11` at address 00001038. The code includes a loop to search for the value 25 in an array.
- MemoryView2:** Shows a memory dump starting at address 0000105C, displaying hexadecimal values.

```
.text
00001000:E3A00D41    LDR R0,=A
00001004:E3A0100A    MOV R1,#10
00001008:E1A03001    MOV R3,R1
0000100C:                L1:
0000100C:E4902004    LDR R2,[R0],#4
00001010:E3520019    CMP R2,#25      ;Element 25 is to be searched in the array
00001014:0A000004    BEQ L2
00001018:E2511001    SUBS R1,R1,#1
0000101C:E3510000    CMP R1,#0
00001020:1AFFFFF9    BNE L1
00001024:E3E06000    MOV R6,#-1
00001028:EF000011    SWI 0x11
0000102C:                L2:
0000102C:E3A06001    MOV R6,#1
00001030:E2411001    SUB R1,R1,#1
00001034:E0437001    SUB R7,R3,R1
00001038:EF000011    SWI 0x11

.data
00001040:                A:.WORD 2,4,6,8,12,20,25,30,40,50

MemoryView2
0000105C 0000001E 00000028 00000032 81818181 81818181 81818181 81818181 81818181 81818181
00001090 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010C4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010F8 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
```

Case 2 (Element NOT found):

ARM Assembly Code:

.text

LDR R0,=A

MOV R1,#10

MOV R3,R1

L1:

LDR R2,[R0],#4

CMP R2,#70 ;Element 70 is to be searched in the array

BEQ L2

SUBS R1,R1,#1

CMP R1,#0

BNE L1

MOV R6,#-1

SWI 0x11

L2:

MOV R6,#1

SUB R1,R1,#1

SUB R7,R3,R1

SWI 0x11

.data

A:.WORD 2,4,6,8,12,20,25,30,40,50

Output:

RegistersView

General Purpose

Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 4200

R1 : 0

R2 : 50

R3 : 10

R4 : 0

R5 : 0

R6 : -1

R7 : 0

R8 : 0

R9 : 0

R10 (s1) : 0

R11 (fp) : 0

R12 (ip) : 0

R13 (sp) : 21504

R14 (lr) : 0

R15 (pc) : 4136

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

5.s

```

.text
00001000:E3A00D41      LDR R0,=A
00001004:E3A0100A      MOV R1,#10
00001008:E1A03001      MOV R3,R1
0000100C:              L1:
0000100C:E4902004      LDR R2,[R0],#4
00001010:E3520046      CMP R2,#70      ;Element 70 is to be searched in the array
00001014:0A000004      BEQ L2
00001018:E2511001      SUBS R1,R1,#1
0000101C:E3510000      CMP R1,#0
00001020:1AFFFFF9      BNE L1
00001024:E3E06000      MOV R6,#-1
00001028:EF000011      SWI 0x11
0000102C:              L2:
0000102C:E3A06001      MOV R6,#1
00001030:E2411001      SUB R1,R1,#1
00001034:E0437001      SUB R7,R3,R1
00001038:EF000011      SWI 0x11
.data
00001040:              A: .WORD 2,4,6,8,12,20,25,30,40,50

```

MemoryView2

0000105c

0000105C

0000001E

00000028

00000032

81818181

81818181

81818181

81818181

81818181

Week#____3_____

Program Number: __6__

Write an ALP to generate Fibonacci Series and store them in an array.

ARM Assembly Code:

.text

LDR R1,=A

MOV R2,#0

STR R2,[R1]

ADD R1,R1,#4

MOV R3,#1

STR R3,[R1]

MOV R5,#8 ; 8 fibonacci numbers will be stored after 0 and
1(So,total=10) in the array

L1:

ADD R4,R2,R3

ADD R1,R1,#4

STR R4,[R1]

MOV R2,R3

MOV R3,R4

SUBS R5,R5,#1

MOV R4,#0

BNE L1

SWI 0x11

A:.WORD

Output:

RegistersView

General Purpose

Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

6.s

R0 : 00000000

R1 : 00001068

R2 : 00000015

R3 : 00000022

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) : 0000103c

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

.text

00001000:E59F1038 LDR R1,=A

00001004:E3A02000 MOV R2,#0

00001008:E5812000 STR R2,[R1]

0000100C:E2811004 ADD R1,R1,#4

00001010:E3A03001 MOV R3,#1

00001014:E5813000 STR R3,[R1]

00001018:E3A05008 MOV R5,#8 ; 8 fibonacci numbers will be stored after 0 and 1(So,total=10) in the array

0000101C: L1:

0000101C:E0824003 ADD R4,R2,R3

00001020:E2811004 ADD R1,R1,#4

00001024:E5814000 STR R4,[R1]

00001028:E1A02003 MOV R2,R3

0000102C:E1A03004 MOV R3,R4

00001030:E2555001 SUBS R5,R5,#1

00001034:E3A04000 MOV R4,#0

00001038:1AFFFFF7 ENE L1

0000103C:EF000011 SWI 0x11

.data

00001044: A:.WORD

MemoryView3

00001044

Word Size

8Bit

16Bit

32Bit

00001044 00000000 00000001 00000001 00000002 00000003 00000005 00000008 0000000D 00000015 00000022 81818181 81818181 81818181

00001078 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181

000010AC 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181

000010E0 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181

OutputView