

# Microprocessor and Computer Architecture Laboratory

UE19CS256

4th Semester, Academic Year 2020-21

Date:29/1/21

Name: Kartik Soni	SRN: PES1UG19CS212	Section D
-------------------	-----------------------	--------------

Week# 2 Program Number: 5

Based on the value of the number in R0, Write an ALP to store 1 in R1 if R0 is zero, Store 2 in R1 if R0 is positive, Store 3 in R1 if R0 is negative.

## I. ARM Assembly Code

```
mov r0, #0
cmp r0, #0
beq L1
bmi L2
mov r1, #2
L1: mov r1, #1
L2: mov r1, #3
```

## II. Output Screen Shot

File View Cache Debug Watch Help

RegistersView

General Purpose Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 00000000  
**R1 : 00000001**  
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) : 00001018**

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

1.s

```
00001000:E3A00000  mov r0, #0
00001004:E3500000  cmp r0, #0
00001008:0A000001  beq L1
0000100C:4A000001  bmi L2
00001010:E3A01002  mov r1, #2
00001014:E3A01001  L1: mov r1, #1
00001018:E3A01003  L2: mov r1, #3
```

OutputView

Console Stdin/Stdout/Stderr

Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s

## Case1

RegistersView

General Purpose Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 00000005  
**R1 : 00000002**  
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) : 00001014**

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

0x200000df

1.s

```
00001000:E3A00005  mov r0, #0x05
00001004:E3500000  cmp r0, #0
00001008:0A000001  beq L1
0000100C:4A000001  bmi L2
00001010:E3A01002  mov r1, #2
00001014:E3A01001  L1: mov r1, #1
00001018:E3A01003  L2: mov r1, #3
```

OutputView

Console Stdin/Stdout/Stderr

Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s

## Case2

RegistersView

General Purpose

Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 :ffffffe

R1 :00000003

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) :0000101c

-----

CPSR Register

Negative (N) :1

Zero (Z) :0

Carry (C) :1

Overflow (V) :0

IRQ Disable:1

FIQ Disable:1

Thumb (T) :0

CPU Mode :System

-----

0x00000000

1.s

00001000:E3E00001 mov r0, #-2

00001004:E3500000 cmp r0,#0

00001008:0A000001 beq L1

0000100C:4A000001 bmi L2

00001010:E3A01002 mov r1,#2

00001014:E3A01001 L1: mov r1,#1

00001018:E3A01003 L2: mov r1,#3

OutputView

Console

Stdin/Stdout/Stderr

Loading assembly language file C:\Users\Kartik\Desktop\College\MFCA Lab\1.s

## Case3

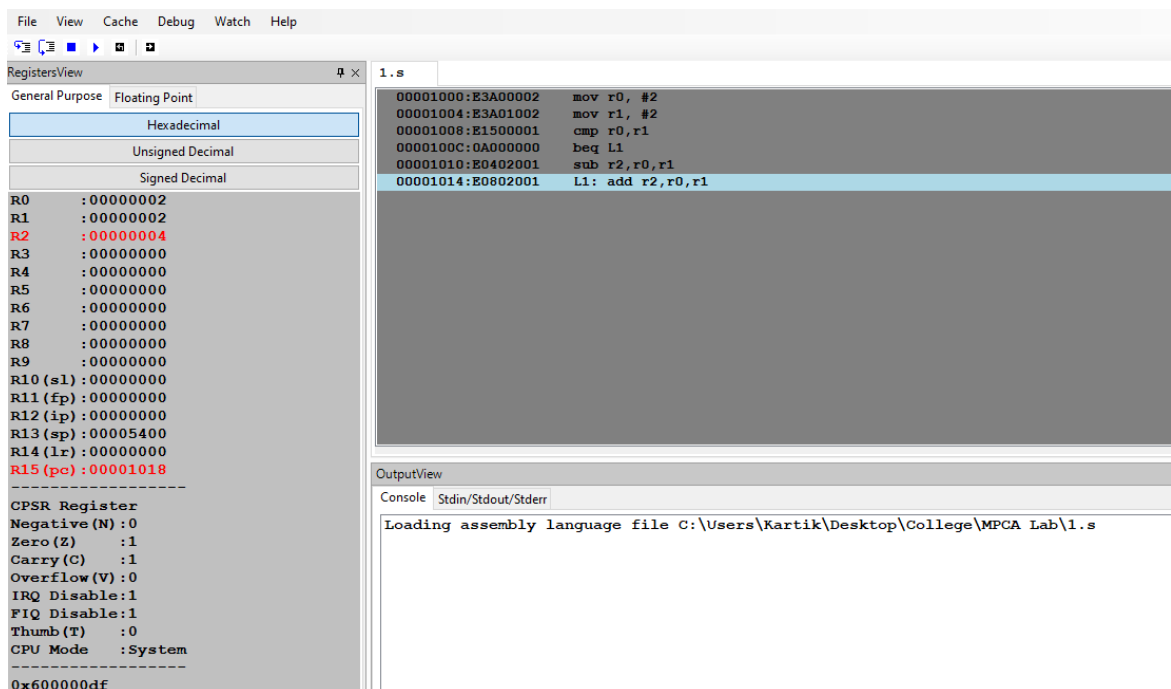
Week# \_\_\_\_2\_\_\_\_ Program Number: \_\_\_\_6\_\_\_\_

Write an ALP to compare the value of R0 and R1, add if R0 = R1, else subtract

### I. ARM Assembly Code

```
mov r0, #3
mov r1, #2
cmp r0,r1
beq L1
sub r2,r0,r1
L1: add r2,r0,r1
```

### II. Output Screen Shot



## Case1

File
View
Cache
Debug
Watch
Help

RegistersView
1.s

General Purpose
Floating Point

Hexadecimal
Unsigned Decimal
Signed Decimal

R0 :00000003  
R1 :00000002  
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) :00001014  
-----  
CPSR Register  
Negative (N) :0  
Zero (Z) :0  
Carry (C) :1  
Overflow (V) :0  
IRQ Disable:1  
FIQ Disable:1  
Thumb (T) :0  
CPU Mode :System  
-----  
0x200000df

00001000:E3A00003 mov r0, #3  
00001004:E3A01002 mov r1, #2  
00001008:E1500001 cmp r0,r1  
0000100C:0A000000 beq L1  
00001010:E0402001 sub r2,r0,r1  
00001014:E0802001 L1: add r2,r0,r1

OutputView

Console Stdin/Stdout/Stderr

Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s

## Case2

Week#\_\_\_\_2\_\_\_\_\_Program Number: \_\_\_\_7\_\_\_\_

Write an ALP to find the factorial of a number stored in R0.  
Store the value in R1 (without using LDR and STR instructions).Use only registers.

### I. ARM Assembly Code

```
mov r0, #5
mov r1, #1
mov r2, #1
L1: mul r3 , r1 , r2
    add r2,r2,#1
    mov r1,r3
    cmp r2,#6
    bne L1
    swi 0x011
```

### II. Output

The screenshot displays an ARM assembly debugger interface. The top menu bar includes File, View, Cache, Debug, Watch, and Help. Below the menu is a toolbar with icons for file operations, execution, and breakpoints. The main window is divided into three panes:

- RegistersView:** Shows the state of various registers. The 'General Purpose' tab is selected, displaying a list of registers (R0 to R15) with their current values in hexadecimal. R0 is 5, R1 is 120, R2 is 6, and R3 is 120. Other registers (R4 to R15) are 0. The CPSR Register is also shown with various flags (Negative, Zero, Carry, Overflow, IRQ Disable, FIQ Disable, Thumb) and CPU Mode set to Supervisor.
- Assembly Code:** Displays the assembly code being executed, with addresses and instructions. The code is as follows:

```
00001000:E3A00005  mov r0, #5
00001004:E3A01001  mov r1, #1
00001008:E3A02001  mov r2, #1
0000100C:E0030291  L1: mul r3 , r1 , r2
00001010:E2822001  add r2,r2,#1
00001014:E1A01003  mov r1,r3
00001018:E3520006  cmp r2,#6
0000101C:1AFFFFFA  bne L1
00001020:EF000011  swi 0x011
```
- OutputView:** Shows the execution output in the Console. The output includes the following text:

```
Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s
Execution starting ...
PC out of valid memory range, address:00000008

Execution ending, Instruction Count:0 Elapsed Time:00:00:00.0229456
Instructions per second:0
```

Week#\_\_\_\_2\_\_\_\_\_Program Number: \_\_\_\_8a\_\_\_\_

Write an ALP to add two 32 bit numbers loaded from memory and store the result in memory.

I. ARM Assembly Code

.Data

A: .WORD 12345678

B: .WORD 01342110

C: .WORD 0

.Text

LDR r0,=A

LDR r1,=B

LDR r2,=C

LDR r4,[r0]

LDR r5,[r1]

add r3,r4,r5

STR r3,[r2]

## II. Output

The screenshot displays a debugger window with the following components:

- RegistersView:** Shows the state of 16 registers (R0-R15) and the CPSR register. R0-R15 are in hexadecimal format. R0 is 00001028, R1 is 0000102c, R2 is 00001030, R3 is 00c22596, R4 is 00bc614e, R5 is 0005c448, R6-R9 are 00000000, R10 (s1) is 00000000, R11 (fp) is 00000000, R12 (ip) is 00000000, R13 (sp) is 00005400, R14 (lr) is 00000000, and R15 (pc) is 00011400. The CPSR register shows 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.
- Memory View:** Shows the memory layout for file 1.s. It includes a .Data section with labels A (00001028: 12345678), B (0000102C: 01342110), and C (00001030: 0). It also includes a .Text section with instructions: LDR r0,=A (00001000:E59F0014), LDR r1,=B (00001004:E59F1014), LDR r2,=C (00001008:E59F2014), LDR r4,[r0] (0000100C:E5904000), LDR r5,[r1] (00001010:E5915000), add r3,r4,r5 (00001014:E0843005), and STR r3,[r2] (00001018:E0843005).
- OutputView:** Shows the console output. It includes the message "Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s", "Execution starting ...", "PC out of valid memory range, address:00011400", "Execution ending, Instruction Count:0 Elapsed Time:00:00:00.0778000", and "Instructions per second:0".



Week# \_\_\_\_2\_\_\_\_ Program Number: \_\_\_\_8b\_\_\_\_

Write an ALP to add two 16 bit numbers loaded from memory and store the result in memory.

## I. ARM Assembly Code

### .Data

A: .HWORD 1234

B: .HWORD 5678

C: .HWORD 0

### .Text

LDR r0,=A

LDR r1,=B

LDR r2,=C

LDRH r4,[r0]

LDRH r5,[r1]

add r3,r4,r5

STRH r3,[r2]

## II. Output

The screenshot displays an ARM assembly debugger interface. The top menu bar includes File, View, Cache, Debug, Watch, and Help. The main window is divided into three panes:

- RegistersView:** Shows the state of 16 ARM registers. R0 through R15 are listed with their values in hexadecimal. R15 (PC) is highlighted in red and shows the address 00011400. Below the registers, the CPSR register is shown with various flags (Negative, Zero, Carry, Overflow, IRQ Disable, FIQ Disable, Thumb) and the CPU Mode set to System.
- 1.s:** Displays the assembly code for the file 1.s. It shows the .Data section with variables A, B, and C, and the .Text section with the assembly instructions: LDR r0,=A; LDR r1,=B; LDR r2,=C; LDRH r4,[r0]; LDRH r5,[r1]; add r3,r4,r5; STRH r3,[r2].
- OutputView:** Shows the execution log. It indicates that the assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s was loaded, and execution started. It also shows a warning: "PC out of valid memory range, address:00011400". The execution ended with an instruction count of 0 and an elapsed time of 00:00:00.0777748.

Week#\_\_\_\_2\_\_\_\_\_Program Number: \_\_\_\_9a\_\_\_\_

Write an ALP to find GCD of two numbers (without using LDR and STR instructions). Both numbers are in registers.

Use only registers.

I. ARM Assembly Code

```
mov r0,#200
```

```
mov r1,#40
```

```
mov r2,r0
```

```
mov r3,r1
```

```
L1: cmp r3,r2
```

```
    beq L2
```

```
    bmi L3
```

```
    sub r3,r3,r2
```

```
    b L1
```

```
L3: sub r2,r2,r3
```

```
    b L1
```

```
L2: swi 0x011
```

# Output

File View Cache Debug Watch Help

RegistersView

General Purpose Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 200  
R1 : 40  
R2 : 40  
R3 : 40  
R4 : 0  
R5 : 0  
R6 : 0  
R7 : 0  
R8 : 0  
R9 : 0  
R10 (s1) : 0  
R11 (fp) : 0  
R12 (ip) : 0  
R13 (sp) : 0  
R14 (lr) : 4144  
R15 (pc) : 8

-----  
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 : Supervisor  
-----  
0x600000d3

1.s

```
00001000:E3A000C8    mov r0,#200
00001004:E3A01028    mov r1,#40
00001008:E1A02000    mov r2,r0
0000100C:E1A03001    mov r3,r1
00001010:E1530002    L1: cmp r3,r2
00001014:0A000004            beq L2
00001018:4A000001            bmi L3
0000101C:E0433002            sub r3,r3,r2
00001020:EAFFFFFA            b L1
00001024:E0422003    L3: sub r2,r2,r3
00001028:EAFFFFF8            b L1
0000102C:EF000011    L2: swi 0x011
```

OutputView

Console Stdin/Stdout/Stderr

Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s  
Execution starting ...  
PC out of valid memory range, address:00000008

Execution ending, Instruction Count:0 Elapsed Time:00:00:00.0189503  
Instructions per second:0

Week#\_\_\_\_2\_\_\_\_\_Program Number: \_\_\_\_9b\_\_\_\_

Write an ALP to find the GCD of given numbers (both numbers in memory). Store result in memory.

I. ARM Assembly Code

.Data

A: .WORD 100

B: .WORD 400

C: .WORD 0

.Text

LDR r0,=A

LDR r1,=B

LDR r4,=C

LDR r2,[r0]

LDR r3,[r1]

L1: cmp r3,r2

beq L2

bmi L3

sub r3,r3,r2

b L1

L3: sub r2,r2,r3

b L1

L2: STR r3,[r4]

swi 0x011

## II.Output

RegistersView

General Purpose Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 4164  
R1 : 4168  
R2 : 100  
R3 : 100  
R4 : 4172  
R5 : 0  
R6 : 0  
R7 : 0  
R8 : 0  
R9 : 0  
R10 (s1) : 0  
R11 (fp) : 0  
R12 (ip) : 0  
R13 (sp) : 0  
R14 (lr) : 4152  
R15 (pc) : 8

-----  
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 : Supervisor  
-----  
0x600000d3

1.s

.Data  
00001044: A: .WORD 100  
00001048: B: .WORD 400  
0000104C: C: .WORD 0

.Text  
00001000:E59F0030 LDR r0,=A  
00001004:E59F1030 LDR r1,=B  
00001008:E59F4030 LDR r4,=C  
0000100C:E5902000 LDR r2,[r0]  
00001010:E5913000 LDR r3,[r1]  
00001014:E1530002 L1: cmp r3,r2  
00001018:0A000004 beq L2  
0000101C:4A000001 bmi L3  
00001020:E0433002 sub r3,r3,r2  
00001024:EAF00000 b L1  
00001028:E0422003 L3: sub r2,r2,r3  
0000102C:EAF00000 b L1  
00001030:E5843000 L2: STR r3,[r4]  
swi 0x011

OutputView

Console Stdin/Stdout/Stderr

Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s  
Execution starting ...  
PC out of valid memory range, address:00000008

Execution ending, Instruction Count:0 Elapsed Time:00:00:00.0305387  
Instructions per second:0

A<B

RegistersView

General Purpose Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 4164  
R1 : 4168  
R2 : 400  
R3 : 400  
R4 : 4172  
R5 : 0  
R6 : 0  
R7 : 0  
R8 : 0  
R9 : 0  
R10 (s1) : 0  
R11 (fp) : 0  
R12 (ip) : 0  
R13 (sp) : 0  
R14 (lr) : 4152  
R15 (pc) : 8

-----  
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 : Supervisor  
-----  
0x600000d3

1.s

.Data  
00001044: A: .WORD 400  
00001048: B: .WORD 400  
0000104C: C: .WORD 0

.Text  
00001000:E59F0030 LDR r0,=A  
00001004:E59F1030 LDR r1,=B  
00001008:E59F4030 LDR r4,=C  
0000100C:E5902000 LDR r2,[r0]  
00001010:E5913000 LDR r3,[r1]  
00001014:E1530002 L1: cmp r3,r2  
00001018:0A000004 beq L2  
0000101C:4A000001 bmi L3  
00001020:E0433002 sub r3,r3,r2  
00001024:EAF00000 b L1  
00001028:E0422003 L3: sub r2,r2,r3  
0000102C:EAF00000 b L1  
00001030:E5843000 L2: STR r3,[r4]  
swi 0x011

OutputView

Console Stdin/Stdout/Stderr

Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s  
Execution starting ...  
PC out of valid memory range, address:00000008

Execution ending, Instruction Count:0 Elapsed Time:00:00:00.0209442  
Instructions per second:0

A=B



Week#\_\_\_\_2\_\_\_\_\_Program Number: \_\_\_\_10a\_\_\_\_

Write an ALP to add an array of ten 32 bit numbers from  
memory

## 1.Assembly Code

.Data

A:.word 10,20,30,40,50,60,70,80,90,11

.Text

LDR R0,=A

mov R1,#10

mov R3,#0

L1: LDR R2,[R0]

ADD R0,R0,#4

ADD R3,R3,R2

SUB R1,R1,#1

cmp R1,#0

bne L1

swi 0x011

## 2.Output

FileViewCacheDebugWatchHelp

RegistersView

General PurposeFloating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0: 00001054

R1: 00000000

R2: 0000000b

R3: 000001cd

R4: 00000000

R5: 00000000

R6: 00000000

R7: 00000000

R8: 00000000

R9: 00000000

R10 (s1): 00000000

R11 (fp): 00000000

R12 (ip): 00000000

R13 (sp): 00000000

R14 (lr): 00001028

R15 (pc): 00000008

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: Supervisor

0x600000d3

1.s

.Data

0000102C: A: .word 10,20,30,40,50,60,70,80,90,11

00001054: B: .word 0

.Text

00001000:E59F0020 LDR R0,=A

00001004:E3A0100A mov R1,#10

00001008:E3A03000 mov R3,#0

0000100C:E5902000 L1: LDR R2,[R0]

00001010:E2800004 ADD R0,R0,#4

00001014:E0833002 ADD R3,R3,R2

00001018:E2411001 SUB R1,R1,#1

0000101C:E3510000 cmp R1,#0

00001020:1AFFFFF9 bne L1

00001024:EF000011 swi 0x011

00001028:0000102C

OutputView

ConsoleStdin/Stdout/Stderr

Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s  
Execution starting ...  
PC out of valid memory range, address:00000008  
  
Execution ending, Instruction Count:0 Elapsed Time:00:00:00.0189525  
Instructions per second:0



Week#\_\_\_\_2\_\_\_\_\_Program Number: \_\_\_\_10b\_\_\_\_

Write an ALP to add an array of five 16 bit numbers from  
memory

I. ARM Assembly Code

.Data

A:.hword 10,20,30,40,50

.Text

```
LDR R0,=A
mov R1,#5
mov R3,#0
L1: LDRH R2,[R0]
ADD R0,R0,#2
ADD R3,R3,R2
SUB R1,R1,#1
cmp R1,#0
bne L1
swi 0x011
```

## II. Output Screen Shot

The screenshot displays a debugger window with the following components:

- RegistersView:** Shows the state of 16 registers (R0-R15) and the CPSR register. R15 (PC) is highlighted in red and shows the value 00001024.
- Assembly View:** Displays the assembly code for file 1.s. The current instruction is `swi 0x011` at address 00001024.
- OutputView:** Shows the console output, which includes the message "Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s".

**RegistersView Details:**

Register	Value
R0	00001036
R1	00000000
R2	00000032
R3	00000096
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)	00001024

**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

**Assembly View Details:**

```
0000102C: .Data      A: .hword 10,20,30,40,50
00001000:E59F0020  LDR R0,=A
00001004:E3A01005  mov R1,#5
00001008:E3A03000  mov R3,#0
0000100C:E01020B0  L1: LDRH R2,[R0]
00001010:E2800002  ADD R0,R0,#2
00001014:E0833002  ADD R3,R3,R2
00001018:E2411001  SUB R1,R1,#1
0000101C:E3510000  cmp R1,#0
00001020:1AFFFFF9    bne L1
00001024:EF000011    swi 0x011
00001028:0000102C
```

**OutputView Details:**

```
Loading assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s
```

Week# \_\_\_\_2\_\_\_\_ Program Number: \_\_\_\_10c\_\_\_\_

Write an ALP to add an array of five 8 bit numbers  
from memory

## I. ARM Assembly Code

.Data

A:.byte 1,2,3,4,5

.Text

```
LDR R0,=A
mov R1,#5
mov R3,#0
L1: LDRB R2,[R0]
ADD R0,R0,#1
ADD R3,R3,R2
SUB R1,R1,#1
cmp R1,#0
bne L1
swi 0x011
```

## II. Output

The screenshot displays an ARM assembly debugger interface. The top menu bar includes File, View, Cache, Debug, Watch, and Help. The main window is divided into three panes:

- RegistersView:** Shows the state of 16 registers (R0-R15) and the CPSR register. R0 is highlighted with a value of 00001031. The CPSR register shows various flags like Negative (N), Zero (Z), Carry (C), Overflow (V), IRQ Disable, FIQ Disable, Thumb (T), and CPU Mode (Supervisor).
- Memory View:** Displays the memory layout. The .Data section contains the array A with values 1, 2, 3, 4, 5. The .Text section contains the assembly code, with the instruction LDR R0,=A highlighted.
- OutputView:** Shows the execution log. It indicates that the assembly language file C:\Users\Kartik\Desktop\College\MPCA Lab\1.s was loaded and execution started. A message states "PC out of valid memory range, address:00000008". The execution ended with an instruction count of 0 and an elapsed time of 00:00:00.0241286 seconds.