

H M Mythreya

PES2UG20CS130

MPCA-Lab Week-2

Task 1: Write a program to copy a block of N data items from location A to B.

a) Full word(.word directive)

```
.DATA
    A: .WORD 2,4,6,8,10
    B: .WORD 0,0,0,0,0

.TEXT
    LDR R1,=A
    LDR R2,=B

    MOV R4,#0

LOOP:
    LDR R3,[R1]
    STR R3,[R2]
    ADD R1,R1,#4
    ADD R2,R2,#4
    ADD R4,R4,#1
    CMP R4,#5
    BNE LOOP
    SWI 0X011

.END
```

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

RegistersView (Left Panel):

- Registers R0 through R15 are listed with their current values in hexadecimal.
- R15 (pc) is highlighted in red, indicating the current instruction pointer.
- Below the registers, the CPSR Register is shown with various flags: Negative (N): 1, Zero (Z): 0, Carry (C): 0, Overflow (V): 0, IRQ Disable: 1, FIQ Disable: 1, Thumb (T): 0, and CPU Mode: System.
- The value 0x800000df is shown at the bottom.

CodeView (Right Panel):

- The assembly code is displayed, with the current instruction 'STR R3, [R2]' highlighted in blue.
- The code is organized into sections: .DATA, .TEXT, and .END...
- The .DATA section contains the definitions for variables A and B.
- The .TEXT section contains the main logic of the program, including the LOOP label and the SWI instruction.

b) Half word(.Hword directive)

.DATA

A: .HWORD 2,4,6,8,10

B: .HWORD 0,0,0,0,0

.TEXT

LDR R1,=A

LDR R2,=B

MOV R4,#0

LOOP:

LDRH R3,[R1]

STRH R3,[R2]

ADD R1,R1,#2

ADD R2,R2,#2

ADD R4,R4,#1

CMP R4,#5

BNE LOOP

SWI 0X011

.END

RegistersView

General Purpose Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

| | |
|----------|-----------|
| R0 | :00000000 |
| R1 | :00001034 |
| R2 | :0000103e |
| 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) | :00011400 |
| R14 (lr) | :00000000 |
| R15 (pc) | :0000100c |

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

q1.o

.DATA

00001034:00040002 A: .HWORD 2,4,6,8,10

:00080006

:000A

0000103E:00000000 B: .HWORD 0,0,0,0,0

:00000000

:0000

.TEXT

00001000:E59F1024 LDR R1,=A

00001004:E59F2024 LDR R2,=B

00001008:E3A04000 MOV R4,#0

LOOP:

0000100C:E1D130B0 LDRH R3,[R1]

00001010:E1C230B0 STRH R3,[R2]

00001014:E2811002 ADD R1,R1,#2

00001018:E2822002 ADD R2,R2,#2

0000101C:E2844001 ADD R4,R4,#1

00001020:E3540005 CMP R4,#5

00001024:1AFFFFF8 BNE LOOP

00001028:EF000011 SWI 0X011

0000102C:00000000 .END...

:0000000A

c) Byte word(.Hword directive)

.DATA

A: .BYTE 2,4,6,8,10

B: .BYTE 0,0,0,0,0

.TEXT

LDR R1,=A

LDR R2,=B

MOV R4,#0

LOOP:

LDRB R3,[R1]

STRB R3,[R2]

ADD R1,R1,#2

ADD R2,R2,#2

ADD R4,R4,#1

CMP R4,#5

BNE LOOP

SWI 0X011

.END

RegistersView

General Purpose
Floating Point

Hexadecimal
Unsigned Decimal
Signed Decimal

| | |
|---------|-----------|
| R0 | :00000000 |
| R1 | :00001038 |
| R2 | :0000103d |
| R3 | :0000000a |
| R4 | :00000002 |
| 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) | :00001010 |

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

q1.o

.DATA

00001034:08060402 A: .BYTE 2,4,6,8,10
:0A

00001039:00000000 B: .BYTE 0,0,0,0,0
:00

.TEXT

00001000:E59F1024 LDR R1,=A

00001004:E59F2024 LDR R2,=B

00001008:E3A04000 MOV R4,#0

LOOP:

0000100C:E5D13000 LDRB R3,[R1]

00001010:E5C23000 STRB R3,[R2]

00001014:E2811002 ADD R1,R1,#2

00001018:E2822002 ADD R2,R2,#2

0000101C:E2844001 ADD R4,R4,#1

00001020:E3540005 CMP R4,#5

00001024:1AFFFFF8 BNE LOOP

00001028:EF000011 SWI 0X011

0000102C:00000000 .END...
:00000005

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

a) Full word(.word directive)

.DATA

A: .WORD 5,10,15,20

B: .WORD 0

.TEXT

LDR R1,=A

LDR R2,=B

MOV R3,#0

MOV R4,#0

LOOP:

LDR R5,[R1]

ADD R4,R4,R5

ADD R1,R1,#4

ADD R3,R3,#1

CMP R3,#4

BNE LOOP

SWI 0x011

STR R2,[R4]

.END

The screenshot displays a debugger interface with two main panels. The left panel, titled 'RegistersView', shows the state of ARM registers. The 'General Purpose' tab is selected, and the 'Signed Decimal' view is chosen. Registers R0 through R15 are listed, with R4 highlighted in red showing a value of 30, and R15 (PC) highlighted in red showing a value of 4120. Below the registers, the CPSR Register is shown with various flags like Negative (N), Zero (Z), Carry (C), Overflow (V), IRQ Disable, FIQ Disable, Thumb (T), and CPU Mode (System). The right panel, titled 'CodeView', shows the assembly code for a program named 'q2.o'. The code is organized into sections: .DATA, .TEXT, and .END. The .TEXT section contains the main logic of the program, including a loop that calculates the sum of elements in memory. The 'ADD R1, R1, #4' instruction is highlighted in blue. The .DATA section defines variables A and B, and the .END instruction marks the end of the program.

| Register | Value |
|----------|-------|
| R0 | 0 |
| R1 | 4160 |
| R2 | 4168 |
| R3 | 2 |
| R4 | 30 |
| R5 | 15 |
| 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) | 4120 |

CPSR Register

| Flag | Value |
|--------------|--------|
| 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

q2.o

```
.DATA
00001038:00000005      A: .WORD 5,10,15,20
          :0000000A
          :0000000F
          :00000014
00001048:00000000      B: .WORD 0

.TEXT
00001000:E59F1028      LDR R1,=A
00001004:E59F2028      LDR R2,=B
00001008:E3A03000      MOV R3,#0
0000100C:E3A04000      MOV R4,#0

LOOP:
00001010:E5915000      LDR R5,[R1]
00001014:E0844005      ADD R4,R4,R5
00001018:E2811004      ADD R1,R1,#4
0000101C:E2833001      ADD R3,R3,#1
00001020:E3530004      CMP R3,#4
00001024:1AFFFFF9      BNE LOOP
00001028:EF000011      SWI 0x011
0000102C:E5842000      STR R2,[R4]

00001030:00000000      .END
          :00000010
```

B) Half word(.Hword directive)

.DATA

A: .HWORD 5,10,15,20

B: .HWORD 0

.TEXT

LDR R1,=A

LDR R2,=B

MOV R3,#0

MOV R4,#0

LOOP:

LDRH R5,[R1]

ADD R4,R4,R5

ADD R1,R1,#2

ADD R3,R3,#1

CMP R3,#4

BNE LOOP

SWI 0x011

STRH R2,[R4]

.END

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

RegistersView: This window shows the state of the processor registers. The **General Purpose** tab is selected, and the **Signed Decimal** format is chosen. The registers are listed as follows:

| Register | Value |
|----------|-------|
| R0 | 0 |
| R1 | 4158 |
| R2 | 4160 |
| R3 | 3 |
| R4 | 30 |
| R5 | 20 |
| 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) | 4116 |

Below the registers, the **CPSR Register** is shown with the following flags:

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

The **CodeView** window shows the assembly code for the program **q2.o**. The code is organized into sections: **.DATA**, **.TEXT**, **LOOP:**, and **.END**.

.DATA:

| Address | Hex Value | Label |
|----------|-----------|----------------------|
| 00001038 | 000A0005 | A: .HWORD 5,10,15,20 |
| | :0014000F | |
| 00001040 | 0000 | B: .HWORD 0 |

.TEXT:

| Address | Hex Value | Label |
|----------|-----------|-----------|
| 00001000 | E59F1028 | LDR R1,=A |
| 00001004 | E59F2028 | LDR R2,=B |
| 00001008 | E3A03000 | MOV R3,#0 |
| 0000100C | E3A04000 | MOV R4,#0 |

LOOP:

| Address | Hex Value | Label |
|----------|-----------|--------------|
| 00001010 | E1D150B0 | LDRH R5,[R1] |
| 00001014 | E0844005 | ADD R4,R4,R5 |
| 00001018 | E2811002 | ADD R1,R1,#2 |
| 0000101C | E2833001 | ADD R3,R3,#1 |
| 00001020 | E3530004 | CMP R3,#4 |
| 00001024 | 1AFFFFF9 | BNE LOOP |
| 00001028 | EF000011 | SWI 0x011 |
| 0000102C | E1C420B0 | STRH R2,[R4] |

.END:

| Address | Hex Value | Label |
|----------|-----------|-------|
| 00001030 | 00000000 | .END |
| | :00000008 | |

C) Byte Wise(.BYTE directive)

.DATA

A: .BYTE 5,10,15,20

B: .BYTE 0

.TEXT

LDR R1,=A

LDR R2,=B

MOV R3,#0

MOV R4,#0

LOOP:

LDRB R5,[R1]

ADD R4,R4,R5

ADD R1,R1,#2

ADD R3,R3,#1

CMP R3,#4

BNE LOOP

SWI 0x011

STRB R2,[R4]

.END

The screenshot displays a debugger interface with two main panes. The left pane, titled 'RegistersView', shows the state of the ARM registers. The right pane, titled 'CodeView', shows the assembly code with memory addresses and disassembled instructions.

RegistersView:

- General Purpose registers: R0 to R15. R15 (PC) is highlighted in red and shows the value 4128.
- 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.
- Memory dump: 0x800000df.

CodeView:

- Address 00001038: 140F0A05: A: .BYTE 5,10,15,20
- Address 0000103C: 00: B: .BYTE 0
- Address 00001000: E59F1028: LDR R1,=A
- Address 00001004: E59F2028: LDR R2,=B
- Address 00001008: E3A03000: MOV R3,#0
- Address 0000100C: E3A04000: MOV R4,#0
- Address 00001010: E5D15000: LOOP: LDRB R5,[R1]
- Address 00001014: E0844005: ADD R4,R4,R5
- Address 00001018: E2811001: ADD R1,R1,#1
- Address 0000101C: E2833001: ADD R3,R3,#1
- Address 00001020: E3530004: CMP R3,#4
- Address 00001024: 1AFFFFFF9: BNE LOOP
- Address 00001028: EF000011: SWI 0x011
- Address 0000102C: E5C42000: STRB R2,[R4]
- Address 00001030: 00000000: .END
- Address 00001034: 00000004: (blank instruction)

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

```
.DATA
    SUM: .WORD 0
```

```
.TEXT
    LDR R1,=SUM
    MOV R2,#10
```

```
LOOP:
    LDR R3,[R1]
    ADD R3,R3,R2
    STR R3,[R1]
    SUB R2,R2,#1
    CMP R2,#0
    BNE LOOP
    SWI 0X011
```

```
.END
```

The screenshot displays a debugger interface with two main panes. The left pane, titled 'RegistersView', shows the state of various registers. The 'General Purpose' tab is selected, and the 'Signed Decimal' view is chosen. Registers R0 through R15 are listed, with R2, R15, and R15 (pc) highlighted in red. R2 contains the value 6, and R15 (pc) contains 4120. Below the registers, the CPSR (Current Program Status Register) is shown with fields for Negative (N), Zero (Z), Carry (C), Overflow (V), IRQ Disable, FIQ Disable, Thumb (T), and CPU Mode (System). The right pane, titled 'CodeView', shows the assembly code for the file 'N-Natural.o'. The code is organized into sections: .DATA, .TEXT, and .END. The .TEXT section contains the main logic of the program, including the initialization of R1 to the address of SUM, setting R2 to 10, and the LOOP section which calculates the sum of natural numbers up to R2. The current instruction being executed is 'CMP R2,#0' at address 00001018.

| Register | Value |
|----------|-------|
| R0 | 0 |
| R1 | 4136 |
| R2 | 6 |
| R3 | 34 |
| R4 | 0 |
| 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) | 4120 |

CPSR Register

| Field | Value |
|--------------|--------|
| 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

```
.DATA
SUM: .WORD 0

.TEXT
LDR R1,=SUM
MOV R2,#10

LOOP:
LDR R3,[R1]
ADD R3,R3,R2
STR R3,[R1]
SUB R2,R2,#1
CMP R2,#0
BNE LOOP
SWI 0X011

.END...
```

4) Write a program to find the product of two 32-bit numbers using a barrel shifter.

Code:

```
.DATA
    A: .WORD 0X87654321
    B: .WORD 65
    C: .WORD 0

.TEXT
    LDR R1,=A
    LDR R2,=B
    LDR R3,=C

    LDR R4,[R1]
    ADDS R5,R4,R4,LSL #6
    STR R5,[R3]
    SWI 0X011
```

The screenshot displays a debugger interface with two main panes: **RegistersView** on the left and **CodeView** on the right.

RegistersView: The 'General Purpose' tab is selected. The 'Hexadecimal' view is chosen. It lists registers R0 through R15, with R15 (PC) highlighted in red. Below the registers, the CPSR Register status is shown, including flags like Negative (N), Zero (Z), Carry (C), Overflow (V), IRQ Disable, FIQ Disable, Thumb (T), and CPU Mode (System). The PC value is 0x300000df.

CodeView: The file 'barrelshiftmult.o' is loaded. It shows the assembly code for the program, with the same instructions as provided in the code block above. The code is organized into sections: **.DATA** (variables A, B, C), **.TEXT** (main logic), and **.SWI** (system call). The instruction 'SWI 0X011...' is highlighted in blue.

5) Convert the following statement in C Language into an ALP

```
IF([A]==[B]) then C=[A]+[B]
ELSE IF ([B]==[C]) D=[A]-[B]
ELSE E=[A]*[B]
```

Code:

```
.DATA
    A: .word 5
    B: .word 5
    C: .word 0
    D: .word 0
    E: .word 0

.TEXT
    LDR R0,=A
    LDR R1,=B
    LDR R2,=C
    LDR R3,=D
    LDR R4,=E

    LDR R5,[R0]
    LDR R6,[R1]
    CMP R5,R6
    BNE TWO
    ADD R7,R5,R6
    STR R7,[R2]
    B DONE

TWO:
    LDR R7,[R2]
    CMP R6,R7
    BNE THREE
    SUB R7,R5,R6
    STR R7,[R3]
    B DONE

THREE:
    MUL R7,R5,R6
    STR R7,[R4]
    B DONE

DONE:
    SWI 0X011

.END
```

RegistersView

General Purpose Floating Point

Hexadecimal

Unsigned Decimal

Signed Decimal

R0 : 4204

R1 : 4208

R2 : 4212

R3 : 4216

R4 : 4220

R5 : 5

R6 : 5

R7 : 10

R8 : 0

R9 : 0

R10 (s1) : 0

R11 (fp) : 0

R12 (ip) : 0

R13 (sp) : 70656

R14 (lr) : 0

R15 (pc) : 4184

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

q5.o

.DATA

0000106C:00000005 A: .word 5

00001070:00000005 B: .word 5

00001074:00000000 C: .word 0

00001078:00000000 D: .word 0

0000107C:00000000 E: .word 0

.TEXT

00001000:E59F0050 LDR R0,=A

00001004:E59F1050 LDR R1,=B

00001008:E59F2050 LDR R2,=C

0000100C:E59F3050 LDR R3,=D

00001010:E59F4050 LDR R4,=E

00001014:E5905000 LDR R5,[R0]

00001018:E5916000 LDR R6,[R1]

0000101C:E1550006 CMP R5,R6

00001020:1A000002 BNE TWO

00001024:E0857006 ADD R7,R5,R6

00001028:E5827000 STR R7,[R2]

0000102C:EA000008 B DONE

TWO:

00001030:E5927000 LDR R7,[R2]

00001034:E1560007 CMP R6,R7

00001038:1A000002 BNE THREE

0000103C:E0457006 SUB R7,R5,R6

00001040:E5837000 STR R7,[R3]

00001044:EA000002 B DONE

THREE:

00001048:E0070695 MUL R7,R5,R6

0000104C:E5847000 STR R7,[R4]

00001050:EAF00000 B DONE

DONE:

00001054:EF000011 SWI 0X011

00001058:00000000 .END...

:00000004

:00000008

:0000000C

:00000010

6) Write a program to find the factorial of a number.

Code:

```
MOV r1,#5
MOV r2,#1
B fact
fact:
    MUL r2,r1,r2
    SUB r1,#1
    CMP r1,#1
    BNE fact
    SWI 0X011
```

.END

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

RegistersView (Left Pane):

- General Purpose: Floating Point
- Hexadecimal
- Unsigned Decimal
- Signed Decimal
- Registers: R0 to R15 (pc). R1 is highlighted in red with value 1. R15 (pc) is 4116.
- 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.
- Address: 0x200000df

CodeView (Right Pane):

- File: factorial.o
- Assembly code:

```
00001000:E3A01005    MOV r1,#5
00001004:E3A02001    MOV r2,#1
00001008:EAF0000F    B fact
fact:
0000100C:E0020291        MUL r2,r1,r2
00001010:E2411001        SUB r1,#1
00001014:E3510001        CMP r1,#1
00001018:1AF000FB        BNE fact
0000101C:EF000011        SWI 0X011

.END...
```