

Microprocessor and Computer Architecture (MPCA) Laboratory
UE20CS252 4th Semester,
Academic Year 2021-22

Week 5

Vanshika Goel	PES1UG20CS484	H Section
----------------------	----------------------	------------------

1. Write a program in ARM7TDMI-ISA to multiply 2 matrices of order 3.

i.e., implement $c[i][j] = c[i][j] + a[i][j] \times b[i][j]$.

a. Use MLA instruction

b. Use MUL instruction

.DATA

A:.WORD 1,2,3,4,5,6,7,8,9

B:.WORD 1,2,3,4,5,6,7,8,9

C:.WORD 0,0,0,0,0,0,0,0,0

.TEXT

LDR R0,=A

LDR R1,=B

LDR R2,=C

MOV R5,#0

MOV R6,#0

MOV R7,#0

MOV R8,#0

LOOP:

LDR R3,[R0],#4

LDR R4,[R1],#12

MLA R8,R3,R4,R8

ADD R5,R5,#1

CMP R5,#3

BNE LOOP

STR R8,[R2],#4

BL LOOP1

LOOP1:

LDR R3,[R0,#-12]!

LDR R4,[R1,#-32]!

MOV R8,#0

MOV R5,#0

ADD R6,R6,#1

CMP R6,#3

BLT LOOP

LOOP2:

LDR R3,[R0,#12]!

LDR R4,[R1,#-12]!

MOV R8,#0

```

MOV R5,#0
MOV R6,#0
ADD R7,R7,#1
CMP R7,#3
BNE LOOP
SWI 0X011

```

The screenshot shows the cpulab01xz.net emulator interface. The top bar includes navigation buttons like 'Step Into', 'Step Over', 'Step Out', 'Continue', 'Stop', 'Restart', and 'Reload'. The main window is divided into three panes:

- Registers:** A list of registers (r0-r15, sp, lr, pc, cpsr, spsr) with their current values. The 'pc' register is highlighted at 00000020.
- Disassembly (Ctrl-D):** A table showing the assembly code being executed. The address 00000000 is selected. The code includes instructions for loading data from memory, performing arithmetic, and branching.
- Messages:** A log of events, including the compilation of the assembly code into an ELF executable and the successful execution of the program.

The disassembly pane shows the following code:

```

Address      Opcode      Disassembly
-----
fffffffb     aaaaaaaa     bge 0xfeaaaaa8
fffffffc     aaaaaaaa     bge 0xfeaaaaa8
00000000     e59f0074     ldr r0, [pc, #116] ; 0x7c
00000004     e59f1074     ldr r1, [pc, #116] ; 0x80
00000008     e59f2074     ldr r2, [pc, #116] ; 0x84
0000000c     e3a05000     mov r5, #0 ; 0x0
00000010     e3a06000     mov r6, #0 ; 0x0
00000014     e3a07000     mov r7, #0 ; 0x0
00000018     e3a08000     mov r8, #0 ; 0x0
0000001c     e4903004     ldr r3, [r0], #4
00000020     e491400c     ldr r4, [r1], #12
00000024     e0288493     mla r8, r3, r4, r8
00000028     e2855001     add r5, r5, #1 ; 0x1
0000002c     e3550003     cmp r5, #3 ; 0x3
00000030     1afffff9     bne 0x1c (0x1c: LOOP)
00000034     e4828004     str r8, [r2], #4
00000038     ebf00000     bl 0x3c (0x3c: LOOP1)
0000003c     ebf00000     bl 0x3c (0x3c: LOOP1)

```

```

.DATA
A:.WORD 1,2,3,4,5,6,7,8,9
B:.WORD 1,2,3,4,5,6,7,8,9
C:.WORD 0,0,0,0,0,0,0,0,0
.TEXT
    LDR R0,=A
    LDR R1,=B
    LDR R2,=C
LOOP:
    LDR R3,[R0],#4
    LDR R4,[R1],#12
    MUL R8,R3,R4
    ADD R9,R9,R8
    ADD R5,R5,#1
    CMP R5,#3
BNE LOOP
    STR R9,[R2],#4
    BL LOOP1

```

LOOP1:

```
LDR R3,[R0,#-12]!  
LDR R4,[R1,#-32]!  
MOV R8,#0  
MOV R9,#0  
MOV R5,#0  
ADD R6,R6,#1  
CMP R6,#3  
BLT LOOP
```

LOOP2:

```
LDR R3,[R0,#12]!  
LDR R4,[R1,#-12]!  
MOV R8,#0  
MOV R9,#0  
MOV R5,#0  
MOV R6,#0  
ADD R7,R7,#1  
CMP R7,#3
```

BNE LOOP

SWI 0X001

The screenshot displays the cpulab01xz.net web interface. The top navigation bar includes buttons for 'Step Into', 'Step Over', 'Step Out', 'Continue', 'Stop', 'Restart', 'Reload', 'File', and 'Help'. The main interface is divided into three panels:

- Registers:** A list of registers (r0-r15, sp, lr, pc, cpsr, spsr) with their current values. The 'pc' register is highlighted in red.
- Disassembly (Ctrl-D):** A table showing the assembly code being executed. The table has columns for Address, Opcode, and Disassembly. The code includes instructions like 'bge', 'ldr', 'mul', 'add', 'cmp', 'bne', 'str', 'bl', and 'loop1'. The 'loop1' label is highlighted in yellow.
- Messages:** A log of messages showing the compilation process. It includes the command used to compile the code and the resulting assembly output.

The assembly code in the disassembly panel is as follows:

```
Address      Opcode  Disassembly  
fffffffc    bge     0xfeaaaaa4  
fffffffb    bge     0xfeaaaaa8  
fffffffa    bge     0xfeaaaaac  
00000000    e59f0070  ldr r0, [pc, #112] ; 0x78  
00000004    e59f1070  ldr r1, [pc, #112] ; 0x7c  
00000008    e59f2070  ldr r2, [pc, #112] ; 0x80  
0000000c    e4903004  loop1: ldr r3, [r0], #4  
00000010    e491400c  ldr r4, [r1], #12  
00000014    e080493  mul r8, r3, r4  
00000018    e0899008  add r9, r9, r8  
0000001c    e2855001  add r5, r5, #1 ; 0x1  
00000020    e3550003  cmp r5, #3 ; 0x3  
00000024    1afffff8  bne loop1  
00000028    e4829004  str r9, [r2], #4  
0000002c    ebfffff  bl 0x30 (0x30: loop1)  
00000030    e530300c  ldr r3, [r0, #-12]!
```

2. Write a program in ARM7TDMI-ISA to find the NORM of a square matrix of order n.

.DATA

A: .WORD 1,3,5,7,9,11,13,15,17

B: .WORD 0,0,0

C: .WORD 0

.TEXT

LDR R0,=A

LDR R1,=B

LDR R2,=C

MOV R3,#0

MOV R4,#0

MOV R10,#3

MOV R5,#0

MOV R8,#0

SUB R8,R8,#1

LOOP:MLA R11,R4,R10,R3

MOV R11,R11,LSL #2

LDR R6,[R0,R11]

CMP R6,#0

MULMI R6,R8,R6

ADD R5,R5,R6

ADD R4,R4,#1

CMP R4,#3

BNE LOOP

MOV R7,R3,LSL #2

STR R5,[R1,R7]

MOV R4,#0

ADD R3,R3,#1

MOV R5,#0

CMP R3,#3

BNE LOOP

MOV R3,#0

MOV R4,#0

MOV R5,#0

LDR R3,[R1,R4]

MAX:ADD R4,R4,#4

LDR R6,[R1,R4]

CMP R3,R6

MOVLTE R3,R6

ADD R5,R5,#1

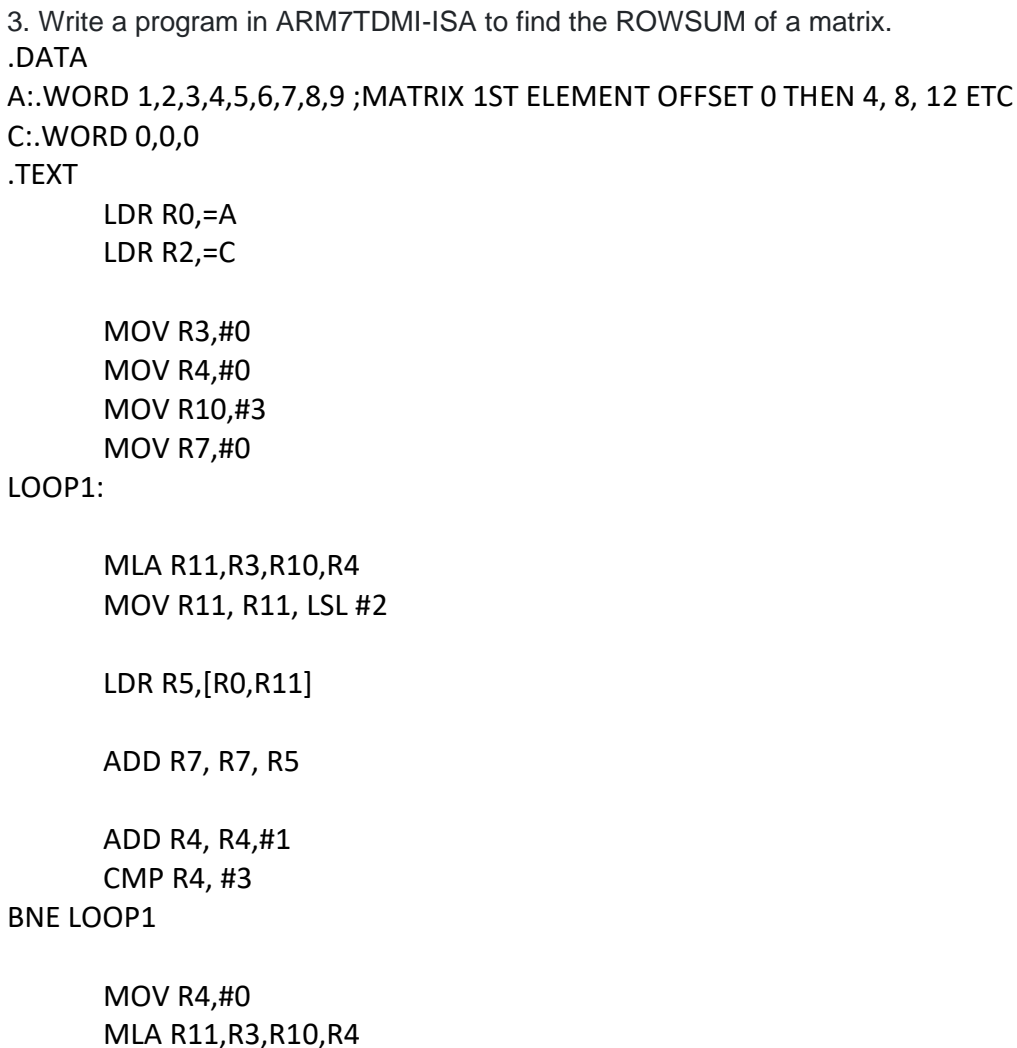
CMP R5,#2

BNE MAX

STR R3,[R2]

SWI 0X011

.END



```

MOV R11, R11, LSL #2
STR R7,[R2,R11]
MOV R7,#0
ADD R3, R3, #1
CMP R3, #3

BNE LOOP1
SWI 0x011

```

The screenshot displays the cpulab01xz.net web-based ARM simulator interface. The top navigation bar includes buttons for 'Stopped', 'Step Into', 'Step Over', 'Step Out', 'Continue', 'Stop', 'Restart', and 'Reload'. Below this, the 'Registers' panel on the left shows the state of various registers, with R11 highlighted at address 00000054. The main 'Disassembly' panel shows the assembly code being executed, with instructions like 'ldr r0, [pc, #80]', 'mov r3, #0', and a loop labeled 'LOOP1'. The 'Messages' panel at the bottom shows the output of the compilation process, including a warning about a missing entry symbol and a successful compile message.