LAB REPORT– EXPERIMENT 5

ARM Assembly - Computations in ARM

BY

AJOE GEORGE (EE21B009)

AND

MANDLA SIVA MANOJ (EE21B083)

GROUP 3

AIM:

To (a) learn the architecture of ARM processor (b) learn basics of ARM instruction set, in particular the ARM instructions pertaining to computations (c) go through example programs and (d) write assembly language programs for the given set of (computational) problems

PROBLEM STATEMENT:

Solve the following engineering problems using ARM through assembly programs
1. Compute the factorial of a given number using ARM processor through assembly programming
2. Combine the low four bits of each of the four consecutive bytes beginning at LIST into one 16-bit halfword. The value at LIST goes into the most significant nibble of the result. Store the result in the 32-bit variable RESULT.
3. Given a 32-bit number, identify whether it is an even or odd. (You implementation should not involve division).

CODE FOR FACTORIAL:

```
AREA factorial,CODE
    ENTRY
                           ;program to find factorial
    MOV  R0,#1             ;int c =1
    MOV  R1,#8             ;int n=1
    MOV  R3,#1             ;int fact=8
    BL loop
    B STOP

loop
        MUL R4,R3,R0
        MOV R3,R4
        ADD R0,R0,#1
        CMP R0,R1
        BLE loop
        BX LR
STOP B STOP          ;R4 IS FINAL ANSWER

    END
```

CODE FOR EVEN ODD:

ODD:

```
AREA EVENODD, CODE, READONLY
Entry


__main
    LDR R0, NUM
    LDR R1, NUM
    LSRS R0,#1
    BCS NEXT
    LDR R3,=EVEN
    STR R1, [R3]
    B skip
NEXT LDR R2,=ODD
skip STR R1,[R2]
    SWI &11


NUM DCD 0x0000091
  AREA RESULT, DATA, READWRITE
ODD DCD 0
EVEN DCD 0
    END
```

EVEN:

```
AREA EVENODD, CODE, READONLY
Entry


__main
    LDR R0, NUM
    LDR R1, NUM
    LSRS R0,#1
    BCS NEXT
    LDR R3,=EVEN
    STR R1, [R3]
    B skip
NEXT LDR R2,=ODD
skip STR R1,[R2]
    SWI &11


NUM DCD 0x0000092
  AREA RESULT, DATA, READWRITE
ODD DCD 0
EVEN DCD 0
   END
```

CODE IS SAME FOR EVEN ODD WE JUST CHANGED VALUES 91 & 92

CODE FOR COMBINATION:

```
AREA program, CODE,READONLY
    ENTRY
main
    adr r0,four
    ldr r1,[r0]
    adr r0,three
ldr r2,[r0]
adr r0,two
ldr r3,[r0]
adr r0,one
ldr r4,[r0]
adr r0,mask
ldr r5,[r0]
    mov r0,r1
and r1,r0,r5
mov r0,r2
and r2,r0,r5
MOV R2, R2, LSL#0x4
mov r0,r3
and r3,r0,r5
MOV R3, R3, LSL#0x8
mov r0,r4
and r4,r0,r5
MOV R4, R4, LSL#0x0c
mov r0,0
add r0,r1
add r0,r2
add r0,r3
add r0,r4
MOV R0,R0, LSL#0x10
str r0,result
swi &11

one dcb &A
align
two dcb &2
align
three dcb &3
align
four dcb &4
    align
mask dcd &0000000f
result dcb 0
  END
```

Explanation:

Factorial

- Factorial is the product of an integer and all the integers below it; e.g. factorial four (8!) is equal to 40320.
- We are writing a code in AVR assembly language to find out the factorial of a given number.
- We are defining our input in register R1.
- As we know, the factorial of a number is the product of all integers less than or equal to the given number. Here we are incrementing the value of register R0 from 0 to 9 (we are taking 9 as input) in a loop and multiplying with R3 (initially 1) and storing the value in R3 and R4 at the end of the loop.
- The final answer gets stored in register R4 in the form of a hexadecimal number.

Combining

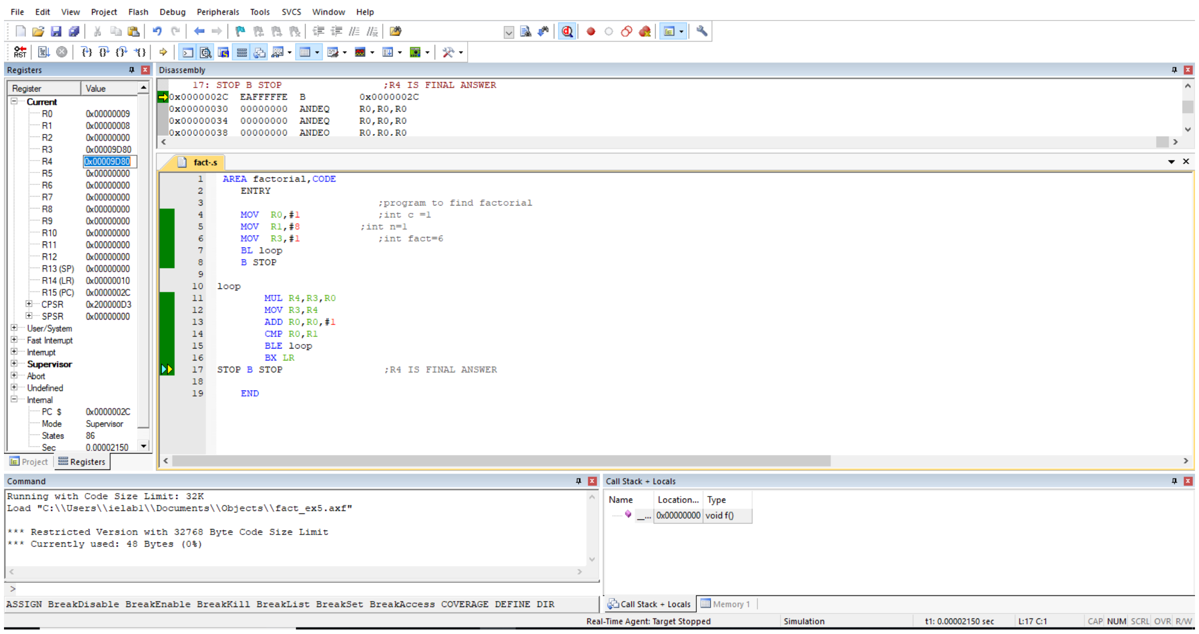
- We are giving four 4-bit numbers as input and expecting a single 16-bit number as output.
- The code works like this: First, the first 4 bits of the number are moved up by 3 places. left. Similarly, the second number is moved two places. Third single digit. And the 4th number doesn't move.
- Then we add all the four numbers after shifting them. This will give us a 16-bit value.
- We will shift the value by four places and this will give us final output.

Even Odd Detection

- In binary if the last digit of the number is 1 then the number is odd, if it is 0 the number is even.
- The code will shift the input by one bit towards the right and the rightmost bit comes out.
- We will check the bit popped out for even or odd.
- If the bit is 0 the number is even and the register R2 will store the value.
- If the bit is 1 the number is odd and the register R3 will store the value.
- In the above code we have taken 92 as input which is even. We can change the input by modifying the code.

RESULTS:

FACTORIAL:



```
Registers
Register      Value
Current
  R0          0x00000009
  R1          0x00000008
  R2          0x00000000
  R3          0x00009D80
  R4          0x00009D80
  R5          0x00000000
  R6          0x00000000
  R7          0x00000000
  R8          0x00000000
  R9          0x00000000
  R10         0x00000000
  R11         0x00000000
  R12         0x00000000
  R13 (SP)    0x00000000
  R14 (LR)    0x00000010
  R15 (PC)    0x0000002C
  CPSR        0x200000D3
  SPSR        0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
Undefined
Internal
  PC $        0x0000002C
  Mode        Supervisor
  States      86
  Sec         0.00002150
```

```
Disassembly
      17: STOP B STOP                    ;R4 IS FINAL ANSWER
0x0000002C  EAFFFFFE  B        0x0000002C
0x00000030  00000000  ANDEQ    R0,R0,R0
0x00000034  00000000  ANDEQ    R0,R0,R0
0x00000038  00000000  ANDEQ    R0,R0,R0
```

fact-.s

```
 AREA factorial,CODE
    ENTRY
                                ;program to find factorial
    MOV  R0,#1                  ;int c =1
    MOV  R1,#8               ;int n=1
    MOV  R3,#1                  ;int fact=6
    BL loop
    B STOP

loop
        MUL R4,R3,R0
        MOV R3,R4
        ADD R0,R0,#1
        CMP R0,R1
        BLE loop
        BX LR
STOP B STOP                     ;R4 IS FINAL ANSWER

    END
```

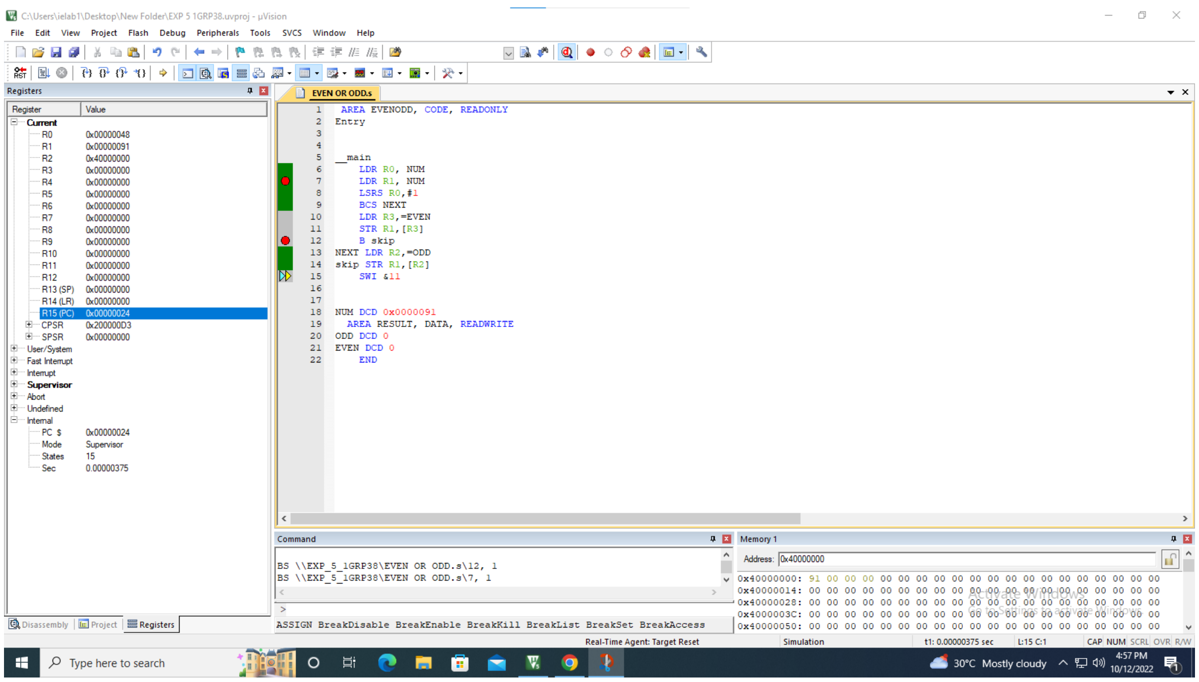
```
Command
Running with Code Size Limit: 32K
Load "C:\\Users\\ielab1\\Documents\\Objects\\fact_ex5.axf"

*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 48 Bytes (0%)
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE DEFINE DIR
```

Name	Location...	Type
_...	0x00000000	void f()

Real-Time Agent: Target Stopped   Simulation   t1: 0.00002150 sec   L:17 C:1

0x00009D80 in decimals is 40320

ODD:



C:\Users\ielab1\Desktop\New Folder\EXP 5 1GRP38.uvproj - µVision

```
Registers
Register      Value
Current
  R0          0x00000048
  R1          0x00000091
  R2          0x40000000
  R3          0x00000000
  R4          0x00000000
  R5          0x00000000
  R6          0x00000000
  R7          0x00000000
  R8          0x00000000
  R9          0x00000000
  R10         0x00000000
  R11         0x00000000
  R12         0x00000000
  R13 (SP)    0x00000000
  R14 (LR)    0x00000000
  R15 (PC)    0x00000024
  CPSR        0x200000D3
  SPSR        0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
Undefined
Internal
  PC $        0x00000024
  Mode        Supervisor
  States      15
  Sec         0.00000375
```

EVEN OR ODD.s

```
 AREA EVENODD, CODE, READONLY
Entry


__main
    LDR R0, NUM
    LDR R1, NUM
    LSRS R0,#1
    BCS NEXT
    LDR R3,=EVEN
    STR R1,[R3]
    B skip
NEXT LDR R2,=ODD
skip STR R1,[R2]
    SWI &11


NUM DCD 0x0000091
  AREA RESULT, DATA, READWRITE
ODD DCD 0
EVEN DCD 0
    END
```

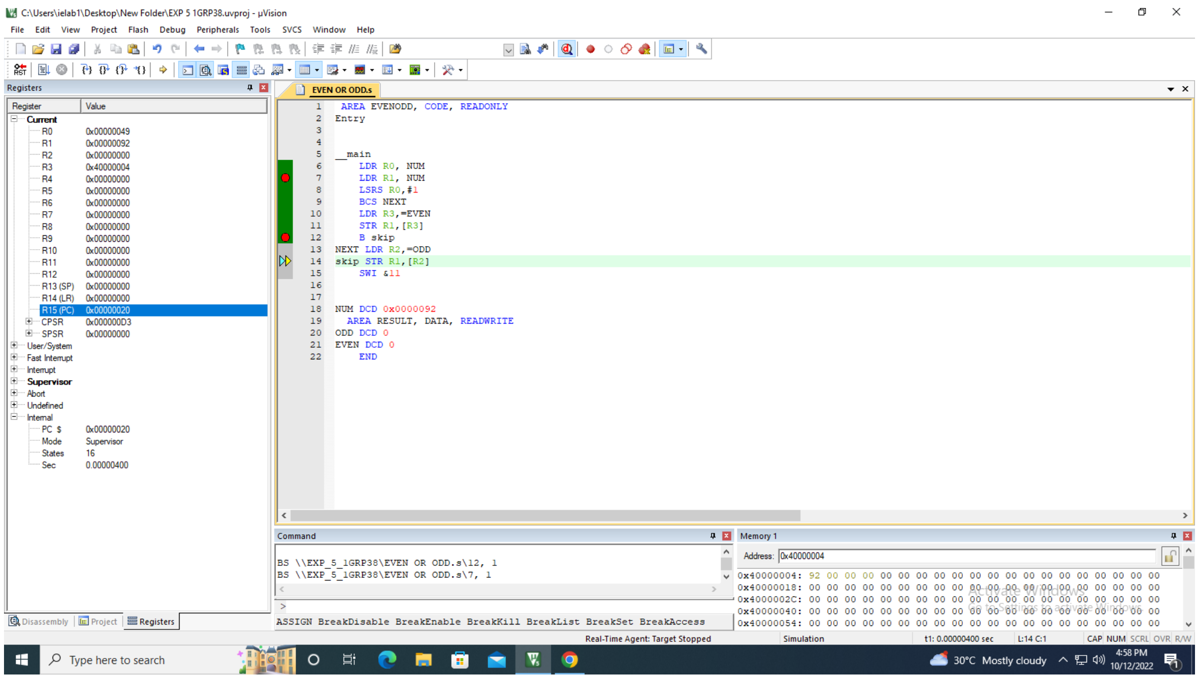
```
Command
BS \\EXP_5_1GRP38\EVEN OR ODD.s\12, 1
BS \\EXP_5_1GRP38\EVEN OR ODD.s\7, 1
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess
```

```
Memory 1
Address: 0x40000000
0x40000000: 91 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000014: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000028: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x4000003C: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Real-Time Agent: Target Reset   Simulation   t1: 0.00000375 sec   L:15 C:1

30°C Mostly cloudy   4:57 PM 10/12/2022

EVEN:



C:\Users\ielab1\Desktop\New Folder\EXP 5 1GRP38.uvproj - µVision

EVEN OR ODD.s

```
 AREA EVENODD, CODE, READONLY
Entry


__main
    LDR R0, NUM
    LDR R1, NUM
    LSRS R0,#1
    BCS NEXT
    LDR R3,=EVEN
    STR R1,[R3]
    B skip
NEXT LDR R2,=ODD
skip STR R1,[R2]
    SWI &11


NUM DCD 0x0000092
  AREA RESULT, DATA, READWRITE
ODD DCD 0
EVEN DCD 0
    END
```

Registers

```
Register      Value
Current
  R0          0x00000049
  R1          0x00000092
  R2          0x00000000
  R3          0x40000004
  R4          0x00000000
  R5          0x00000000
  R6          0x00000000
  R7          0x00000000
  R8          0x00000000
  R9          0x00000000
  R10         0x00000000
  R11         0x00000000
  R12         0x00000000
  R13 (SP)    0x00000000
  R14 (LR)    0x00000000
  R15 (PC)    0x00000020
  CPSR        0x000000D3
  SPSR        0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
Undefined
Internal
  PC $        0x00000020
  Mode        Supervisor
  States      16
  Sec         0.00000400
```

Command

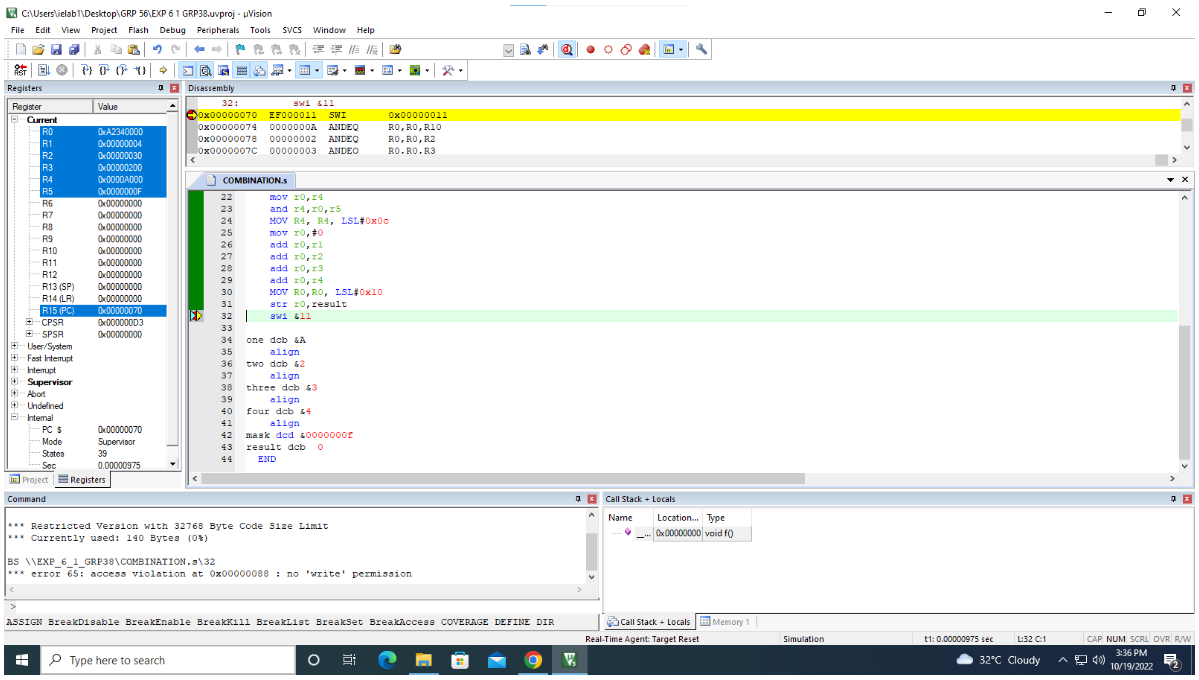
```
BS \\EXP_5_1GRP38\EVEN OR ODD.s\12, 1
BS \\EXP_5_1GRP38\EVEN OR ODD.s\7, 1
```

Memory 1

```
Address: 0x40000004
0x40000004: 92 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000018: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x4000002C: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000054: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Real-Time Agent: Target Stopped Simulation t1: 0.00000400 sec L:14 C:1

COMBINATION:



C:\Users\ielab1\Desktop\GRP 56\EXP 6 1 GRP38.uvproj - µVision

Disassembly

```
    32:          swi &11
0x00000070  EF000011  SWI      0x00000011
0x00000074  0000000A  ANDEQ    R0,R0,R10
0x00000078  00000002  ANDEQ    R0,R0,R2
0x0000007C  00000003  ANDEQ    R0,R0,R3
```

COMBINATION.s

```
    mov r0,r4
    and r4,r0,r5
    MOV R4, R4, LSL#0x0c
    mov r0,#0
    add r0,r1
    add r0,r2
    add r0,r3
    add r0,r4
    MOV R0,R0, LSL#0x10
    str r0,result
    swi &11

one dcb &A
    align
two dcb &2
    align
three dcb &3
    align
four dcb &4
    align
mask dcd &0000000f
result dcb  0
  END
```

Registers

```
Register      Value
Current
  R0          0xA2340000
  R1          0x00000004
  R2          0x00000030
  R3          0x00000200
  R4          0x0000A000
  R5          0x0000000F
  R6          0x00000000
  R7          0x00000000
  R8          0x00000000
  R9          0x00000000
  R10         0x00000000
  R11         0x00000000
  R12         0x00000000
  R13 (SP)    0x00000000
  R14 (LR)    0x00000000
  R15 (PC)    0x00000070
  CPSR        0x000000D3
  SPSR        0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
Undefined
Internal
  PC $        0x00000070
  Mode        Supervisor
  States      39
  Sec         0.00000975
```

Command

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 140 Bytes (0%)

BS \\EXP_6_1_GRP38\COMBINATION.s\32
*** error 65: access violation at 0x00000088 : no 'write' permission
```

Call Stack + Locals

Name	Location...	Type
_...	0x00000000	void f()

Real-Time Agent: Target Reset Simulation t1: 0.00000975 sec L:32 C:1