

LAB 3**Lab Exercises:**

1. Write an assembly language program to implement division by repetitive subtraction.

Program:

```
        AREA RESET,DATA,READONLY
        EXPORT __Vectors

__Vectors

        DCD 0X10001000
        DCD Reset_Handler
        AREA mycode,CODE,READONLY
        ENTRY
        EXPORT Reset_Handler

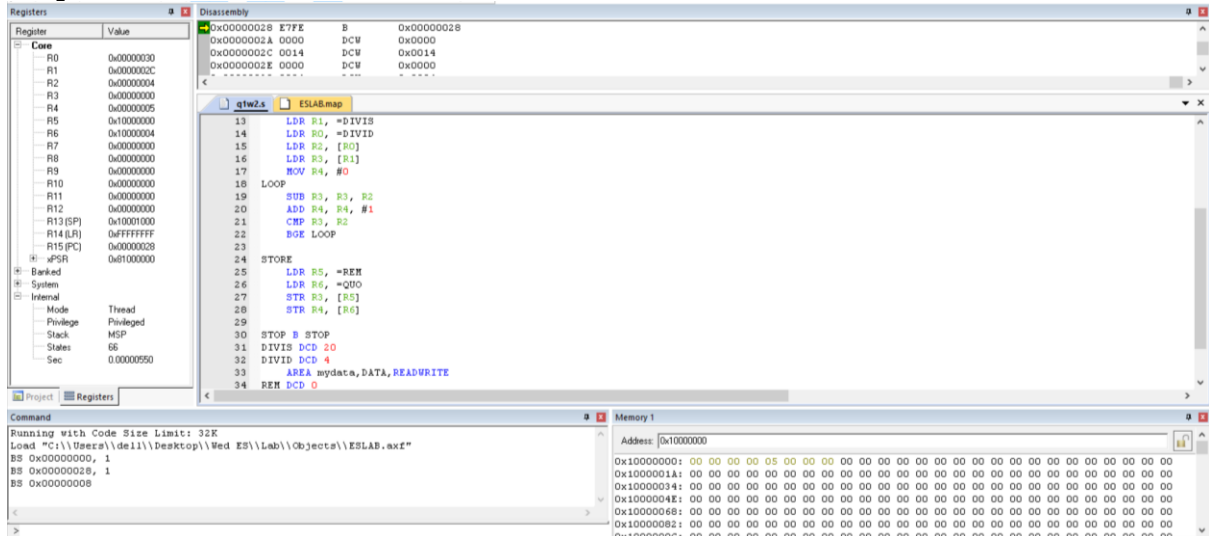
Reset_Handler
        LDR R1, =DIVIS
        LDR R0, =DIVID
        LDR R2, [R0]
        LDR R3, [R1]
        MOV R4, #0

LOOP
        SUB R3, R3, R2
        ADD R4, R4, #1
        CMP R3, R2
        BGE LOOP

STORE
        LDR R5, =REM
        LDR R6, =QUO
        STR R3, [R5]
        STR R4, [R6]

STOP B STOP
DIVIS DCD 20
DIVID DCD 4
        AREA mydata,DATA,READWRITE
REM DCD 0
QUO DCD 0
        END
```

Output:



2. Find the sum of 'n' natural numbers using MLA instruction.

Program:

```
AREA RESET, DATA, READONLY
EXPORT __Vectors
```

__Vectors

```
DCD 0x1001000
DCD Reset_Handler
ALIGN
AREA mycode, CODE, READONLY
ENTRY
EXPORT Reset_Handler
```

Reset_Handler

```
LDR R0,=SRC
LDR R1,=DST
MOV R2,#0
LDR R3,[R0]
MLA R2,R3,R3,R3
LSR R2,#1
STR R2,[R1]
```

STOP

```
B STOP
```

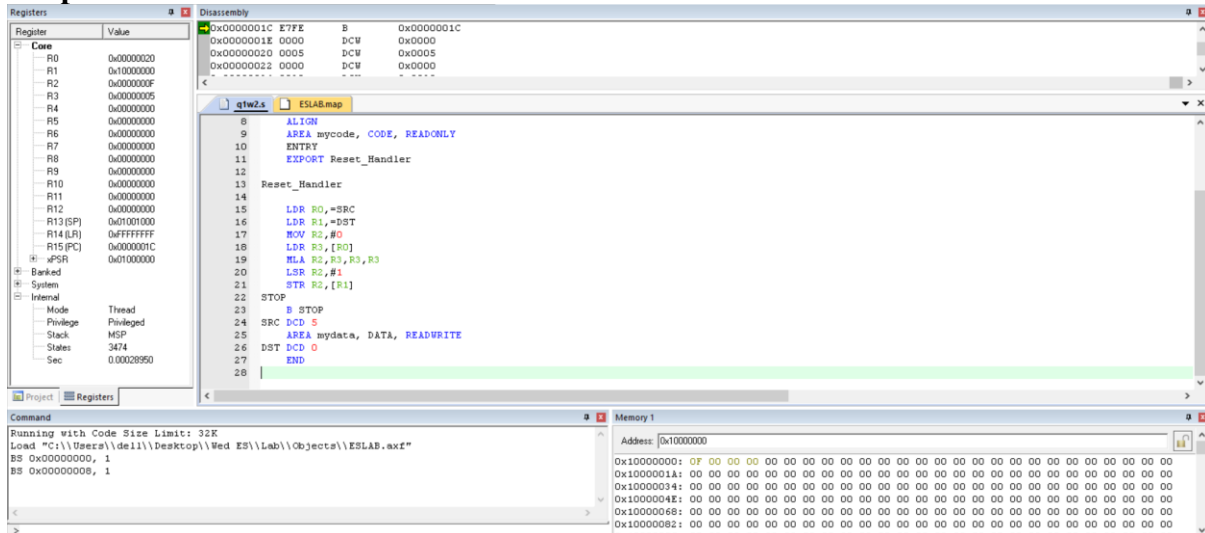
SRC DCD 5 ;n

```
AREA mydata, DATA, READWRITE
```

DST DCD 0

```
END
```

Output:



3. Write an assembly language program to find GCD and LCM of two 8 bit numbers.

Program:

```
AREA RESET, DATA, READONLY
EXPORT __Vectors
```

__Vectors

```
DCD 0X10001000
DCD Reset_Handler
AREA mycode, CODE, READONLY
ENTRY
EXPORT Reset_Handler
```

Reset_Handler

```
LDR R0, =N1
LDR R1, =N2
LDR R2, [R0]
LDR R3, [R1]
```

GCDT

```
CMP R2, R3
SUBLT R3, R3, R2
SUBGT R2, R2, R3
BNE GCDT
```

STRGCD

```
LDR R4, =GCD
STR R2, [R4]
```

LCMT

```
LDR R5, [R0]
LDR R6, [R1]
MOV R4, #0
```

MUL R7, R5, R6

DIVISION

SUB R7, R7, R2

ADD R4, R4, #1

CMP R7, R2

BGE DIVISION

STRLCM

LDR R8, =LCM

STR R4, [R8]

STOP B STOP

N1 DCD 48

N2 DCD 18

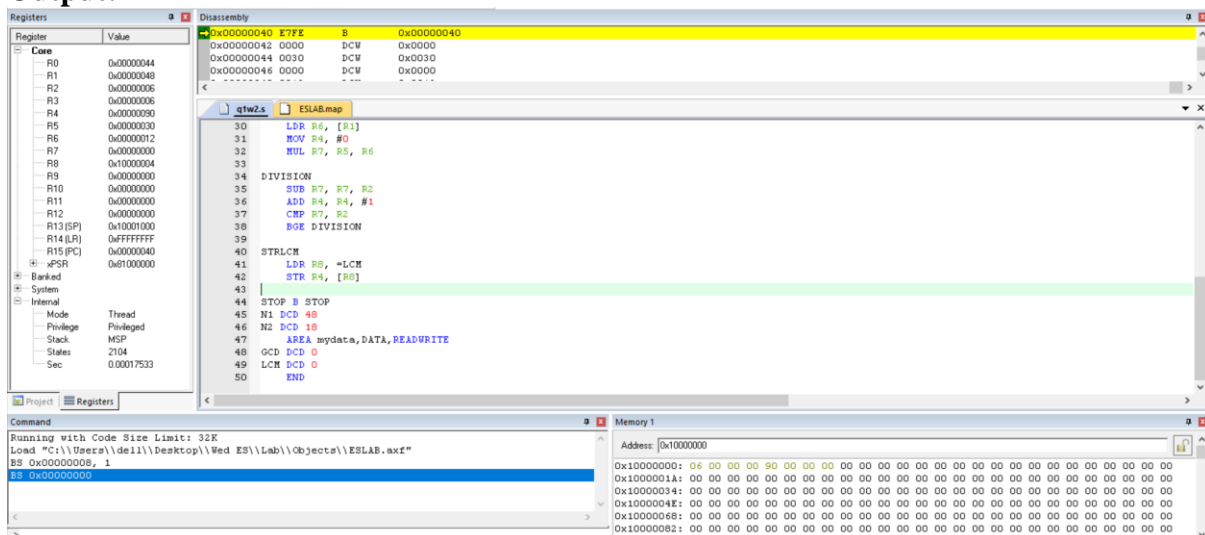
AREA mydata,DATA,READWRITE

GCD DCD 0

LCM DCD 0

END

Output:



4. Write an ARM assembly language program to convert 2-digit hexadecimal number into ascii format.

Program:

AREA RESET,DATA,READONLY
EXPORT __Vectors

__Vectors

DCD 0X10001000

DCD Reset_Handler

ALIGN

AREA mycode,CODE,READONLY

ENTRY

EXPORT Reset_Handler

```

Reset_Handler
    LDR R0, =SRC
    LDR R1, [R0]
    LDR R9, =DST
    MOV R5, #100

BCD
    CMP R1, R5
    BLT DIV2
    MOV R4, #0

DIV1
    SUB R1, R1, R5
    ADD R4, R4, #1
    CMP R1, R5
    BGE DIV1
    ADD R4, R4, #48
    STR R4, [R9], #4

    CMP R5, #10
    MOV R4, #0
    BLT LEND ;loop end

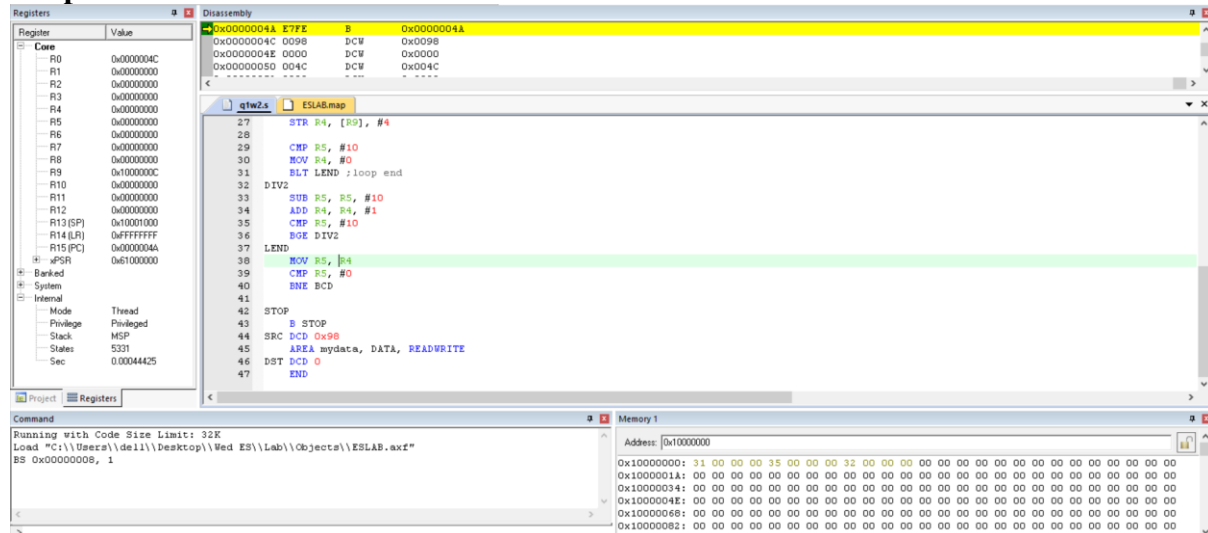
DIV2
    SUB R5, R5, #10
    ADD R4, R4, #1
    CMP R5, #10
    BGE DIV2

LEND
    MOV R5, R4
    CMP R5, #0
    BNE BCD

STOP
    B STOP
SRC DCD 0x98
    AREA mydata, DATA, READWRITE
DST DCD 0
    END

```

Output:



5. Write an ARM assembly language program to convert a 32-bit BCD number in the unpacked form into packed form.

Program:

```

AREA RESET,DATA,READONLY
EXPORT __Vectors
  
```

__Vectors

```

DCD 0X10001000
DCD Reset_Handler
ALIGN
AREA mycode,CODE,READONLY
ENTRY
EXPORT Reset_Handler
  
```

Reset_Handler

```

LDR R0, =SRC
LDR R1, =0x01
MOV R4, #4
  
```

```

UP STR R1,[R0],#1
ADD R1, #2
SUBS R4,#4
BNE UP ;POPULATING UNPACKED
  
```

```

LDR R0,=SRC
LDR R7,=DST
MOV R3,#2
  
```

```

LOOP LDRB R1,[R0],#1
ROR R1,#28
LDRB R2,[R0],#1
ORR R2,R2,R1
STRB R2,[R7],#1
SUBS R3,#1
BNE LOOP
  
```

```

B STOP
AREA mydata, DATA, READWRITE
SRC DCD 0
DST DCD 0
END

```

The screenshot displays the Immunity Debugger interface with three main windows open:

- Registers Window:** Shows the state of various registers. The `R0` register contains the value `0x10000004`. Other registers like `R1` through `R15` and `xPSR` contain `0x00000000`.
- Disassembly Window:** Displays assembly code for a function named `q1w2a`. The code includes a `Reset_Handler` and a `LOOP` section. The `LOOP` section starts with `LDRB R1,[R0],#1` and ends with `B STOP`. The code is disassembled from a file named `ESLAB.axf`.
- Memory Window:** Shows the memory address `0x10000000` and its contents. The memory is filled with `0x00` values, indicating that the memory has been cleared or is uninitialized.