Ayush Goyal

190905522 CSE D Roll 62

ES LAB 1 (Week 1)

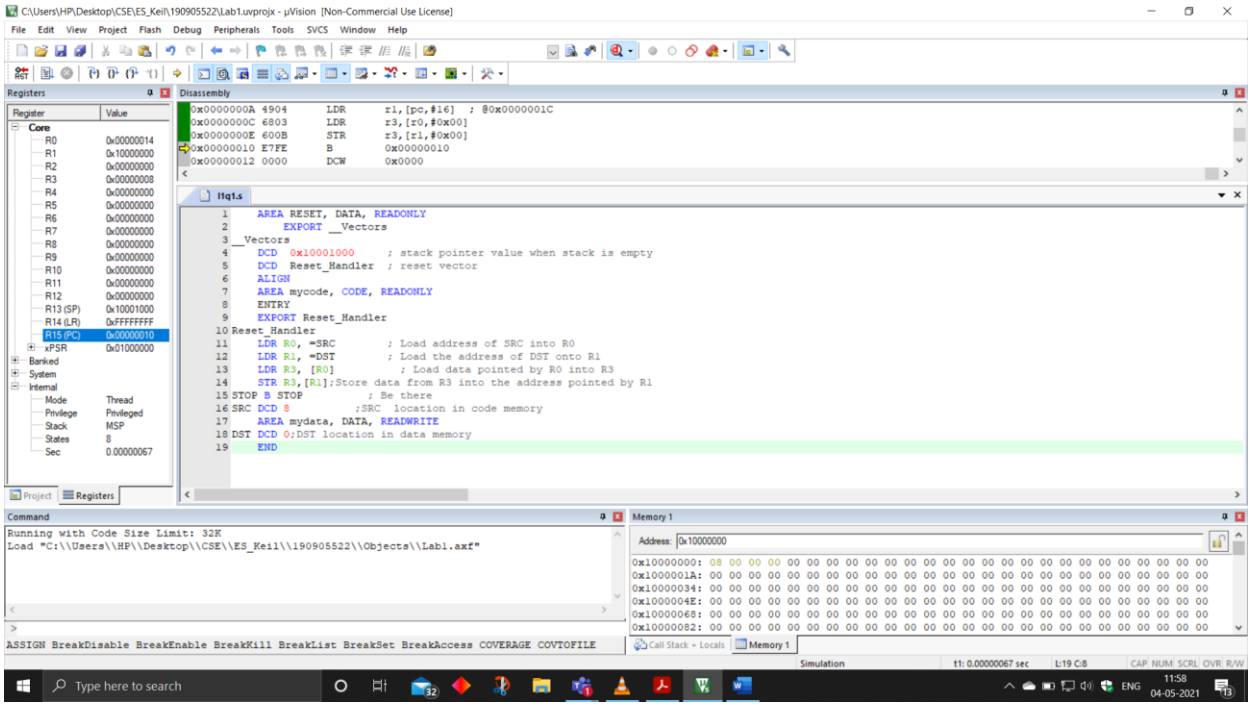
Solved Exercise

Q1) Write an ARM assembly language program to copy 32 bit data from code memory to data memory.

CODE:

```
        AREA RESET, DATA, READONLY
                EXPORT __Vectors
__Vectors
        DCD  0x10001000     ; stack pointer value when stack is empty
        DCD  Reset_Handler  ; reset vector
        ALIGN
        AREA mycode, CODE, READONLY
        ENTRY
        EXPORT Reset_Handler
Reset_Handler
        LDR R0, =SRC       ; Load address of SRC into R0
        LDR R1, =DST       ; Load the address of DST onto R1
        LDR R3, [R0]       ; Load data pointed by R0 into R3
        STR R3,[R1];Store data from R3 into the address pointed by R1
STOP B STOP         ; Be there
SRC DCD 8         ;SRC  location in code memory
        AREA mydata, DATA, READWRITE
DST DCD 0;DST location in data memory
        END
```

OUTPUT :



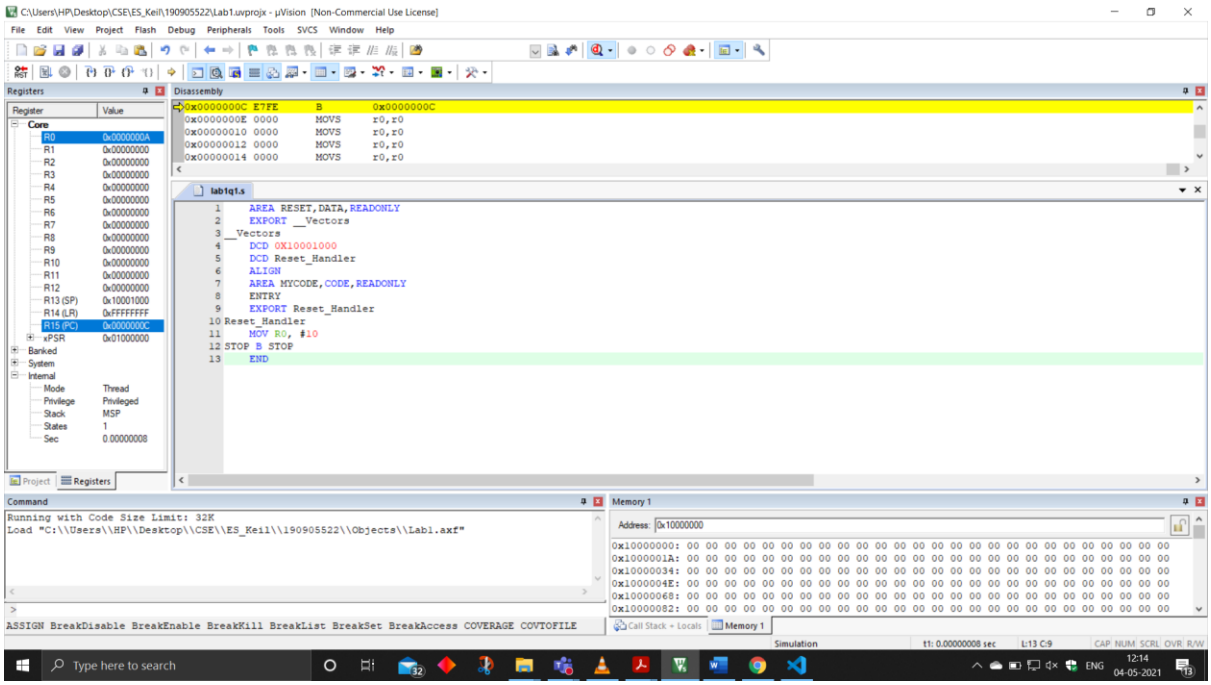
LAB EXERCISES:

Q1) Write an ARM assembly language program to store data into general purpose registers.

CODE :

```
        AREA RESET,DATA,READONLY
        EXPORT __Vectors
__Vectors
        DCD 0X10001000
        DCD Reset_Handler
        ALIGN
        AREA MYCODE,CODE,READONLY
        ENTRY
        EXPORT Reset_Handler
Reset_Handler
        MOV R0, #10
STOP B STOP
        END
```

OUTPUT:



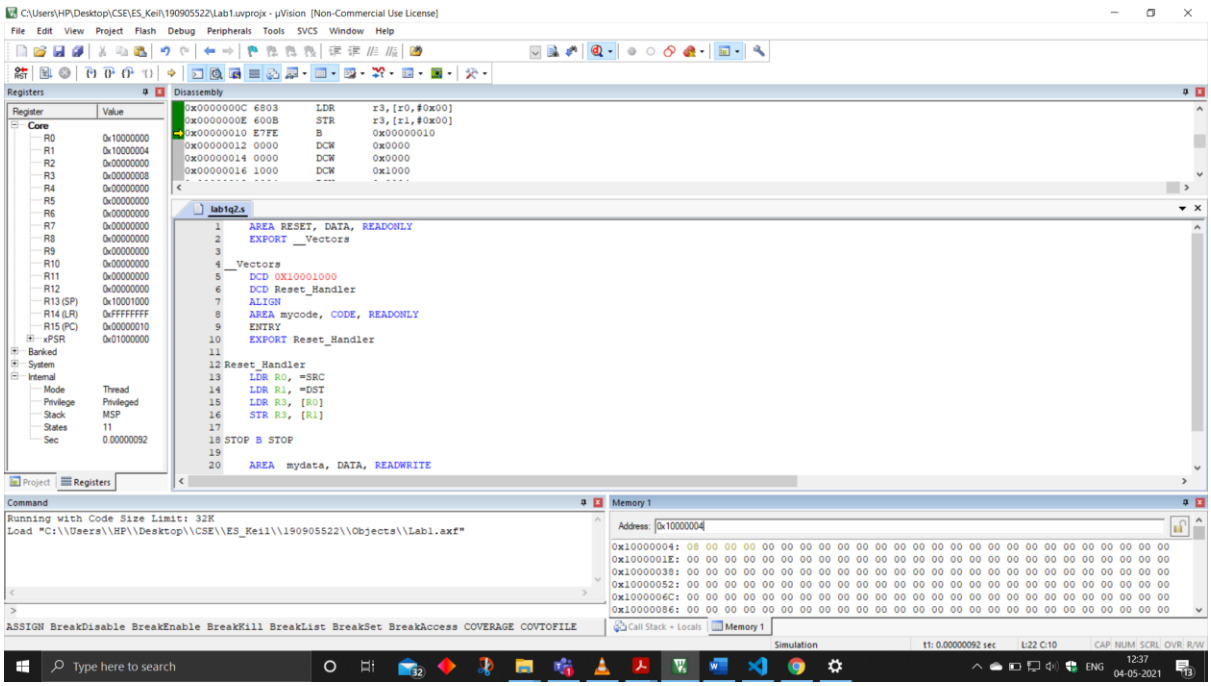
Q2) Write an ARM assembly language program to transfer a 32 bit number from one location in the data memory to another location in the data memory.

CODE:

```
        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, =DST
```

```
        LDR R3, [R0]
        STR R3, [R1]

STOP B STOP

        AREA  mydata, DATA, READWRITE
SRC DCD 8
DST DCD 0
        END
```

OUTPUT:

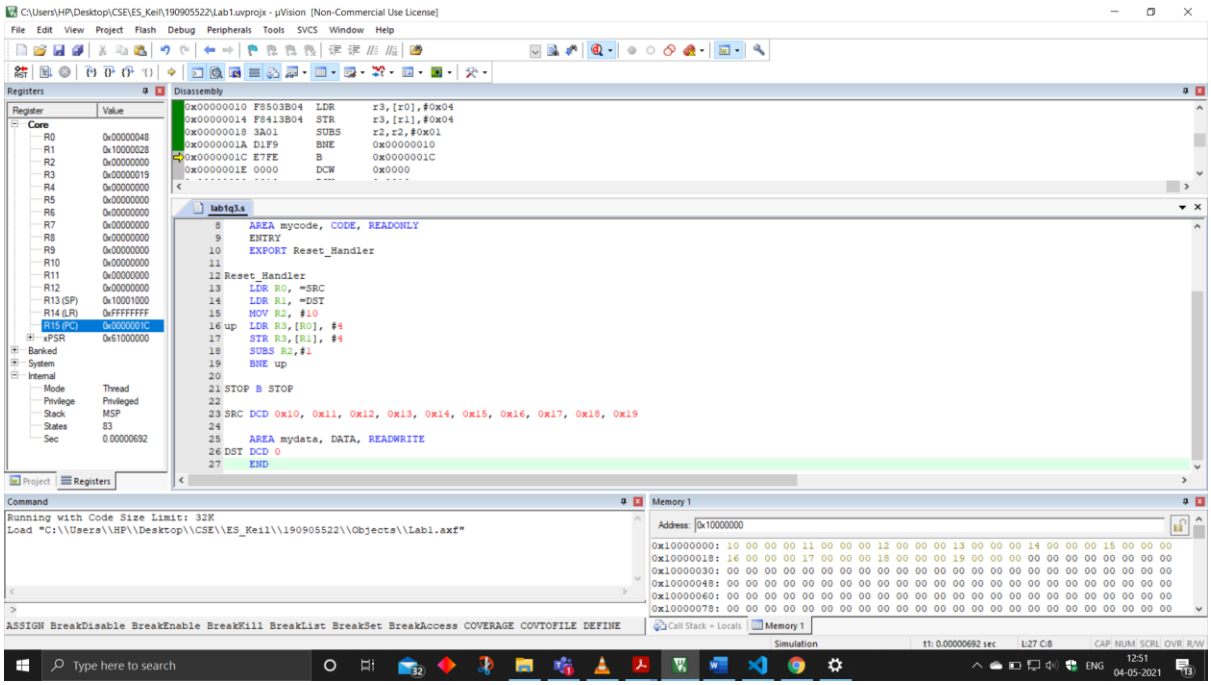


Q3) Write an ARM Assembly language program to transfer block of ten 32 bit numbers from code memory to data memory when the source and destination blocks are non-overlapping.

CODE:

```
        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, =DST
        MOV R2, #10
up      LDR R3,[R0], #4
        STR R3,[R1], #4
        SUBS R2,#1
        BNE up

STOP B STOP

SRC DCD 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19

        AREA mydata, DATA, READWRITE
DST DCD 0
        END
```

OUTPUT:



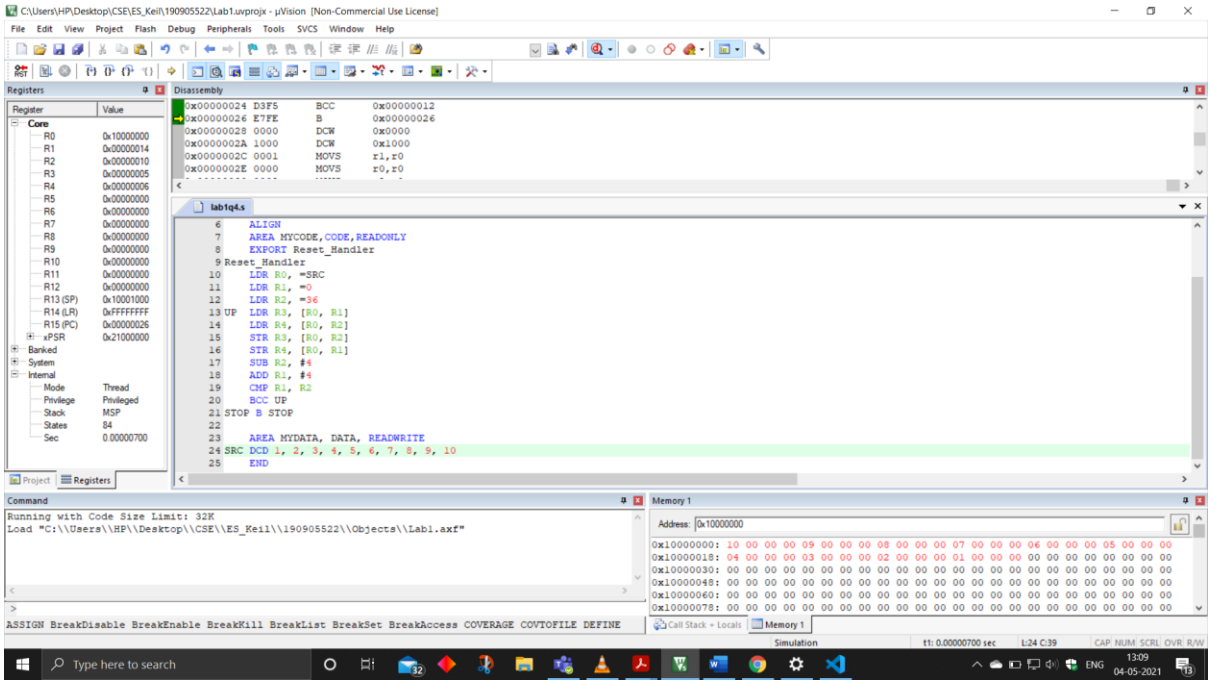
Q4) Reverse an array of ten 32 bit numbers in the memory.

CODE:

```
        AREA RESET,DATA,READONLY
        EXPORT __Vectors
__Vectors
        DCD 0X10001000
        DCD Reset_Handler
        ALIGN
        AREA MYCODE,CODE,READONLY
        EXPORT Reset_Handler
Reset_Handler
        LDR R0, =SRC
        LDR R1, =0
        LDR R2, =36
UP      LDR R3, [R0, R1]
```

```
        LDR R4, [R0, R2]
        STR R3, [R0, R2]
        STR R4, [R0, R1]
        SUB R2, #4
        ADD R1, #4
        CMP R1, R2
        BCC UP
STOP B STOP

        AREA MYDATA, DATA, READWRITE
SRC DCD 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
        END
```

OUITPUT:



END OF LAB 1