LAB 4

Aniket sambher

Reg no-190905466

Section A

Solved Example

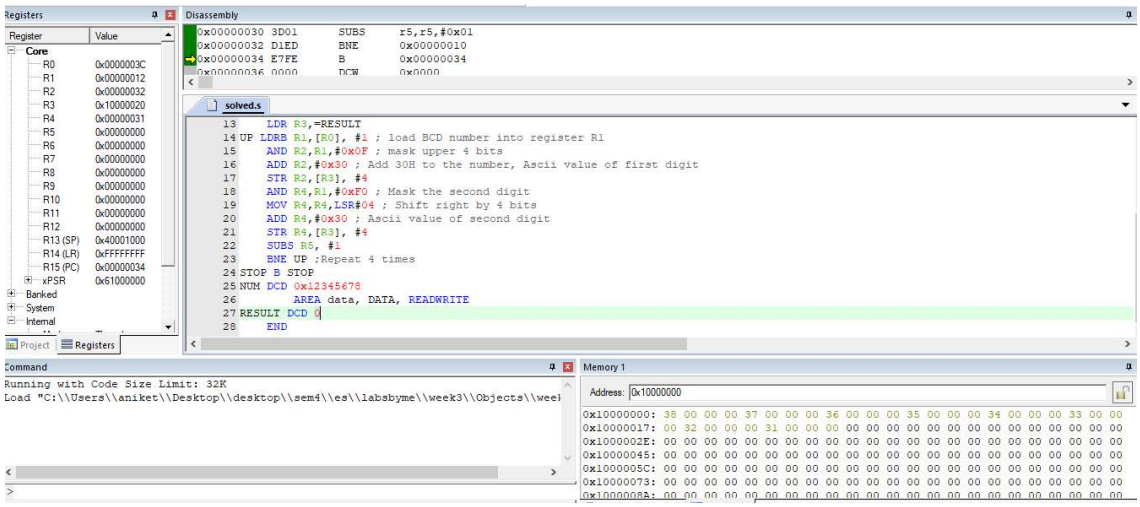
```
                AREA RESET, DATA, READONLY
                EXPORT __Vectors
__Vectors
        DCD 0x40001000 ; stack pointer value when stack is empty
        DCD Reset_Handler ; reset vector
        ALIGN
        AREA mycode, CODE, READONLY
        ENTRY
        EXPORT Reset_Handler
Reset_Handler
        MOV R5, #4
        LDR R0,=NUM
        LDR R3,=RESULT
UP LDRB R1,[R0], #1 ; load BCD number into register R1
        AND R2,R1,#0x0F ; mask upper 4 bits
        ADD R2,#0x30 ; Add 30H to the number, Ascii value of first digit
        STR R2,[R3], #4
        AND R4,R1,#0xF0 ; Mask the second digit
        MOV R4,R4,LSR#04 ; Shift right by 4 bits
        ADD R4,#0x30 ; Ascii value of second digit
        STR R4,[R3], #4
        SUBS R5, #1
        BNE UP ;Repeat 4 times
STOP B STOP
```

NUM DCD 0x12345678

AREA data, DATA, READWRITE

RESULT DCD 0

END

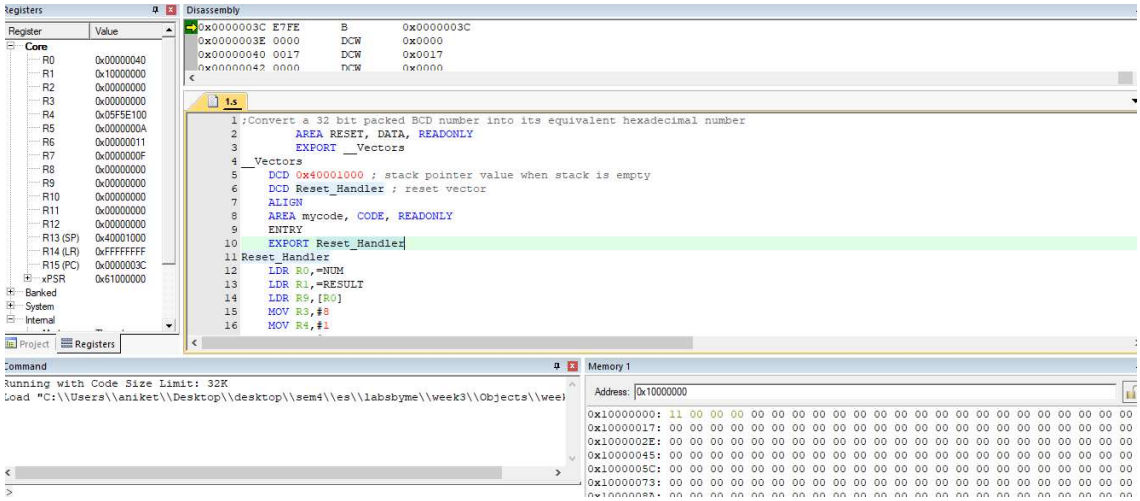


EXCERCISES

1.

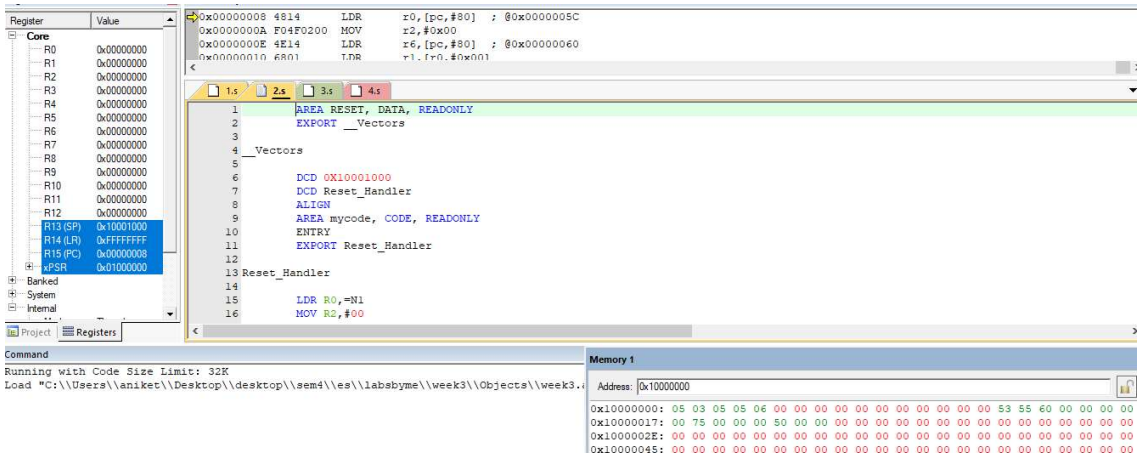
```
;Convert a 32 bit packed BCD number into its equivalent hexadecimal number
                AREA RESET, DATA, READONLY
                EXPORT __Vectors
__Vectors
        DCD 0x40001000 ; stack pointer value when stack is empty
        DCD Reset_Handler ; reset vector
        ALIGN
        AREA mycode, CODE, READONLY
        ENTRY
        EXPORT Reset_Handler
Reset_Handler
        LDR R0,=NUM
        LDR R1,=RESULT
        LDR R9,[R0]
        MOV R3,#8
        MOV R4,#1
        MOV R5,#0xA
        MOV R6,#0
        MOV R7,#0x0F
UP      MOV R8,R9
        AND R8,R7
        MLA R6,R4,R8,R6
        LSR R9,#4
        MUL R4,R5
        SUBS R3,#1
        BNE UP
        STR R6,[R1]
STOP B STOP
NUM DCD 0x17
```

```
        AREA data, DATA, READWRITE
RESULT DCD 0
        END
```



2. Convert a 16 bit hex number into its equivalent packed BCD

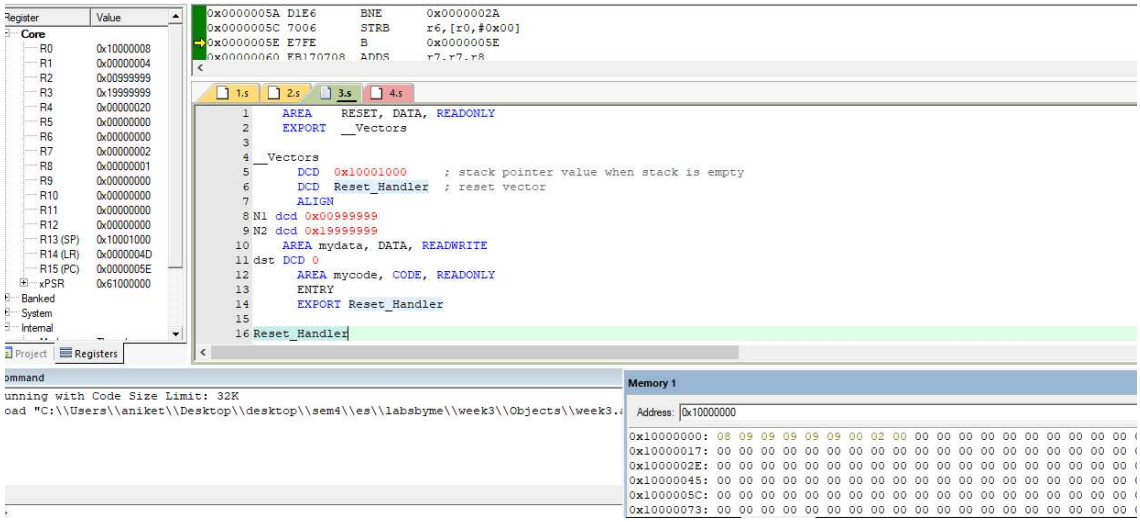
```
                AREA RESET, DATA, READONLY
                EXPORT __Vectors

__Vectors

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

Reset_Handler

                LDR R0,=N1
                MOV R2,#00
                LDR R6,=DST
                LDR R1,[R0]
UP              CMP R1,#0xA
                BCC STORE
                SUB R1,#0xA
                ADD R2,#0X1
                B UP
STORE           STRB R1,[R6],#1
                MOV R1,R2
                MOV R2,#0
                CMP R1,#0xA
                BCS UP
                STRB R1,[R6]

                LDR R0,=DST
                LDR R7,=FIN
                MOV R3,R6
                LDR R6,=DST
                SUB R3,R3,R6
LOOP  LDRB R1,[R0],#1
                ROR R1,#28
                LDRB R2,[R0],#1
                ORR R2,R2,R1
                STRB R2,[R7],#1
                SUB R3,#1
                BNE LOOP
STOP
                B STOP
N1 DCD 0xFFFF
                AREA mydata, DATA, READWRITE
DST DCD 0,0,0,0
FIN DCD 0
        END
```


Register
Value
Core
R0 0x00000000
R1 0x00000000
R2 0x00000000
R3 0x00000000
R4 0x00000000
R5 0x00000000
R6 0x00000000
R7 0x00000000
R8 0x00000000
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x10001000
R14 (LR) 0xFFFFFFFF
R15 (PC) 0x00000008
xPSR 0x01000000
Banked
System
Internal
Project
Registers
0x00000008 4814 LDR r0,[pc,#80] ; @0x0000005C
0x0000000A F04F0200 MOV r2,#0x00
0x0000000E 4E14 LDR r6,[pc,#80] ; @0x00000060
0x00000010 6801 LDR r1,[r0,#0x00]
1.s
2.s
3.s
4.s
1 AREA RESET, DATA, READONLY
2 EXPORT __Vectors
3
4 __Vectors
5
6 DCD 0X10001000
7 DCD Reset_Handler
8 ALIGN
9 AREA mycode, CODE, READONLY
10 ENTRY
11 EXPORT Reset_Handler
12
13 Reset_Handler
14
15 LDR R0,=N1
16 MOV R2,#00
Command
Running with Code Size Limit: 32K
Load "C:\\Users\\aniket\\Desktop\\desktop\\sem4\\es\\labsbyme\\week3\\Objects\\week3.
Memory 1
Address: 0x10000000
0x10000000: 05 03 05 05 06 00 00 00 00 00 00 00 00 00 00 00 53 55 60 00 00 00 00
0x10000017: 00 75 00 00 00 50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x1000002E: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x10000045: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

3.
Add two 32 bit packed BCD numbers and store the result in packed BCD form

```
                AREA    RESET, DATA, READONLY
  EXPORT  __Vectors


__Vectors
    DCD  0x10001000     ; stack pointer value when stack is empty
    DCD  Reset_Handler  ; reset vector
    ALIGN
N1 dcd 0x00999999
N2 dcd 0x19999999
        AREA mydata, DATA, READWRITE
dst DCD 0
    AREA mycode, CODE, READONLY
    ENTRY
    EXPORT Reset_Handler


Reset_Handler
        LDR R0, =N1      ; Load address of SRC into R0
        ldr r2, [r0]
        LDR r0, =N2
        ldr r3, [r0]
        ldr r0, =dst
        mov r5, #8
        mov r6, #0
        mov r9, #0xf
        mov r4, #0
up      mov r1, #0
        mov r7, r2
        mov r8, r3
        and r7, r9
```

```
        lsr r7, r4
        add r7, r6
        mov r6, #0
        and r8, r9
        lsr r8, r4
        bl addn
        add r1, #4
        add r4, #4
        lsl r9, r1
        subs r5, #1
        bne up
        strb r6, [r0]
STOP
        B STOP
addn    adds r7, r8
        cmp  R7, #0xa
        bcc store
        sub r7, #0xa
        ADD R6, #01
store   strb r7, [r0], #1
    bx lr
        END
```


Register
Value
Core
R0 0x10000008
R1 0x00000004
R2 0x00999999
R3 0x19999999
R4 0x00000020
R5 0x00000000
R6 0x00000000
R7 0x00000002
R8 0x00000001
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x10001000
R14 (LR) 0x0000004D
R15 (PC) 0x0000005E
xPSR 0x61000000
Banked
System
Internal
Project
Registers
0x0000005A D1E6 BNE 0x0000002A
0x0000005C 7006 STRB r6,[r0,#0x00]
0x0000005E E7FE B 0x0000005E
1.s
2.s
3.s
4.s
1 AREA RESET, DATA, READONLY
2 EXPORT __Vectors
3
4 __Vectors
5 DCD 0x10001000 ; stack pointer value when stack is empty
6 DCD Reset_Handler ; reset vector
7 ALIGN
8 N1 dcd 0x00999999
9 N2 dcd 0x19999999
10 AREA mydata, DATA, READWRITE
11 dst DCD 0
12 AREA mycode, CODE, READONLY
13 ENTRY
14 EXPORT Reset_Handler
15
16 Reset_Handler
ommand
unning with Code Size Limit: 32K
oad "C:\\Users\\aniket\\Desktop\\desktop\\sem4\\es\\labsbyme\\week3\\Objects\\week3.
Memory 1
Address: 0x10000000
0x10000000: 08 09 09 09 09 09 00 02 00 00 00 00 00 00 00 00 00 00
0x10000017: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x1000002E: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x10000045: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x1000005C: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x10000073: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

4.Multiply two 16 bit packed BCD and store the result in packed BCD form.

```
                AREA RESET, DATA, READONLY
        EXPORT  __Vectors

__Vectors
    DCD  0x10001000
    DCD  Reset_Handler
    ALIGN
N1 DCD 0x9
N2 DCD 0x9
        AREA mydata, DATA, READWRITE
PRODUCT DCD 0,0
TEMP DCD 0
    AREA mycode, CODE, READONLY
    ENTRY
    EXPORT Reset_Handler

Reset_Handler
        LDR R0, =N1
        LDR R2, [R0]
        BL BCD_HEX
        MOV R9, R5
        LDR R0, =N2
        LDR R2, [R0]
        BL BCD_HEX
        MOV R10, R5
        LDR R0, =PRODUCT
        MUL R9, R10
        BL HEX_BCD
```

```
        LDR R9, =TEMP
UP2     LDR R12, [R9], #1
        LDR R11, [R9], #1
        LSL R11, #4
        ORR R12, R11
        STRB R12, [R0], #1
        SUBS R1, #1
        BNE UP2
STOP
        B STOP
BCD_HEX         MOV R3, #1
        MOV R4, #0xa
        MOV R5, #0
        MOV R7, #0xf
UP      MOV R6, R2
        AND R6, R7
        MLA R5, R6, R3, R5
        MUL R3, R4
        LSR R2, #4
        CMP R2, #0
        BNE UP
        BX LR
HEX_BCD
        MOV R8, #0
        LDR R1, =TEMP
UP1     CMP R9, #0xA
        BCC STORE
        SUB R9, #0xA
        ADD R8, #01
        B UP1
STORE
```

```
        STRB R9, [R1], #1
        MOV  R9, R8
        MOV R8, #0
        CMP R9, #0xA
        BCS UP1
    STRB R9, [R1]
            LDR R8, =TEMP
            SUB R1, R8
            BX LR
        END
```