Lab 01: 8051 Assembly Language Programming using MOV and ADD Instructions

Last, First Name: Khan, Umar , Last 4 Student ID:7331.

Purpose:

The purpose of this lab is to continue to help students know the software development tool Keil uVision, learn how to use Keil to edit, compile, simulate an Assembly program starting to use two simple arithmetic instructions MOV and ADD. This project also aims to understand the usage of registers, PC, machine code, and ROM address

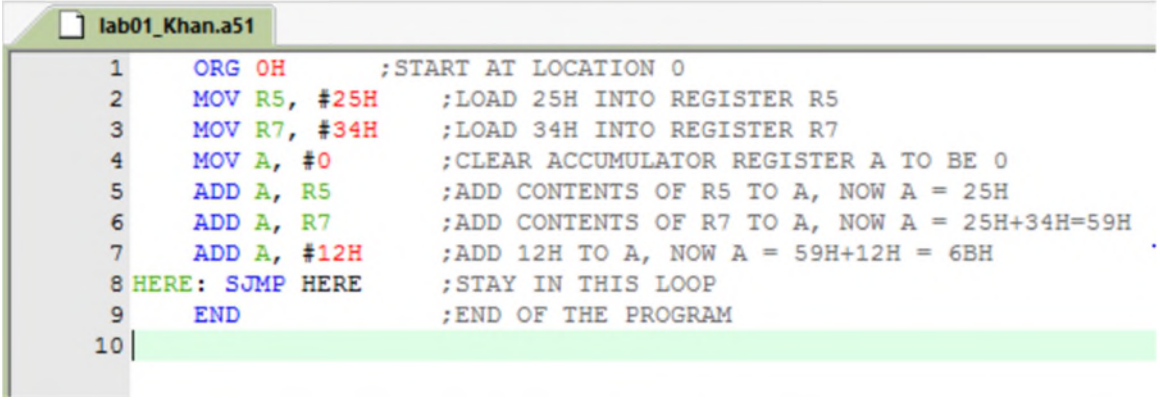
Tasks:

1. Follow the tutorial, Keil tutorial.pdf posted on Beachboard Labs folder; finish all the steps up to simulation. When you create your source file, use "lab01" shown below. Run the program on the simulation and demonstrate it to the instructor.

```
==============================================================
; This program is called "lab01".
; 1. Type in the program, compile and simulate it.
==============================================================
        ORG    0H             ;START AT LOCATION 0
        MOV R5, #25H          ;LOAD 25H INTO REGISTER R5
        MOV R7, #34H          ;LOAD 34H INTO REGISTER R7
        MOV A, #0             ;CLEAR ACCUMULATOR REGISTER A TO BE 0
        ADD    A, R5          ;ADD CONTENTS OF R5 TO A, NOW A = 25H
        ADD A, R7             ;ADD CONTENTS OF R7 TO A, NOW A = 25H+34H=59H
        ADD A, #12H           ;ADD 12H TO A, NOW A = 59H+12H = 6BH
HERE: SJMP HERE               ;STAY IN THIS LOOP
        END                           ;END OF THE PROGRAM
```

2. Verify the execution of the program
3. Upload your report into the dropbox in beachboard with the following items

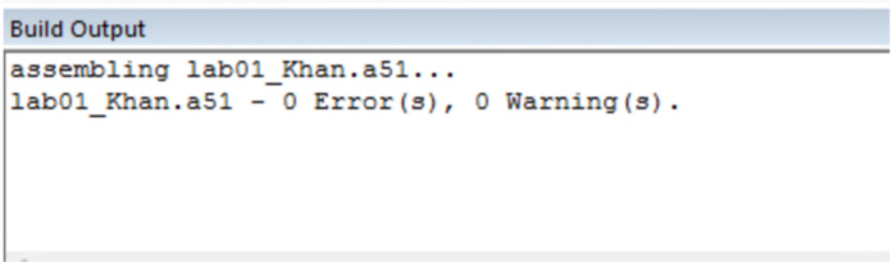
   a. Source code



lab01_Khan.a51

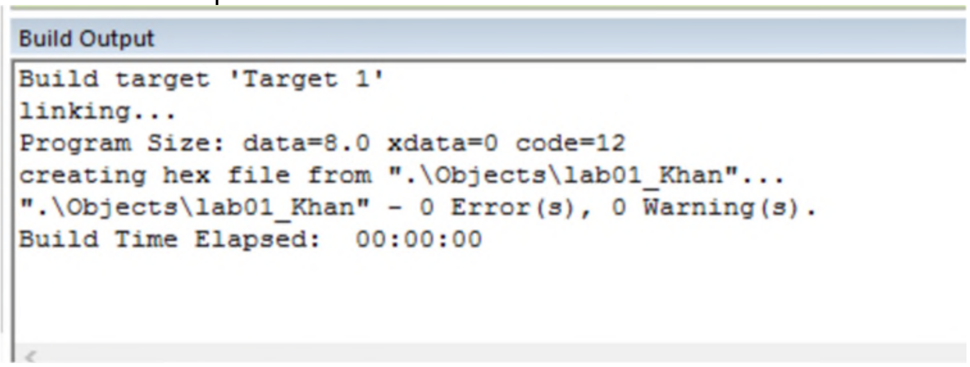
```
     ORG 0H       ;START AT LOCATION 0
     MOV R5, #25H     ;LOAD 25H INTO REGISTER R5
     MOV R7, #34H     ;LOAD 34H INTO REGISTER R7
     MOV A, #0        ;CLEAR ACCUMULATOR REGISTER A TO BE 0
     ADD A, R5        ;ADD CONTENTS OF R5 TO A, NOW A = 25H
     ADD A, R7        ;ADD CONTENTS OF R7 TO A, NOW A = 25H+34H=59H
     ADD A, #12H      ;ADD 12H TO A, NOW A = 59H+12H = 6BH
HERE: SJMP HERE       ;STAY IN THIS LOOP
     END              ;END OF THE PROGRAM
```

b. Build output after the Translate



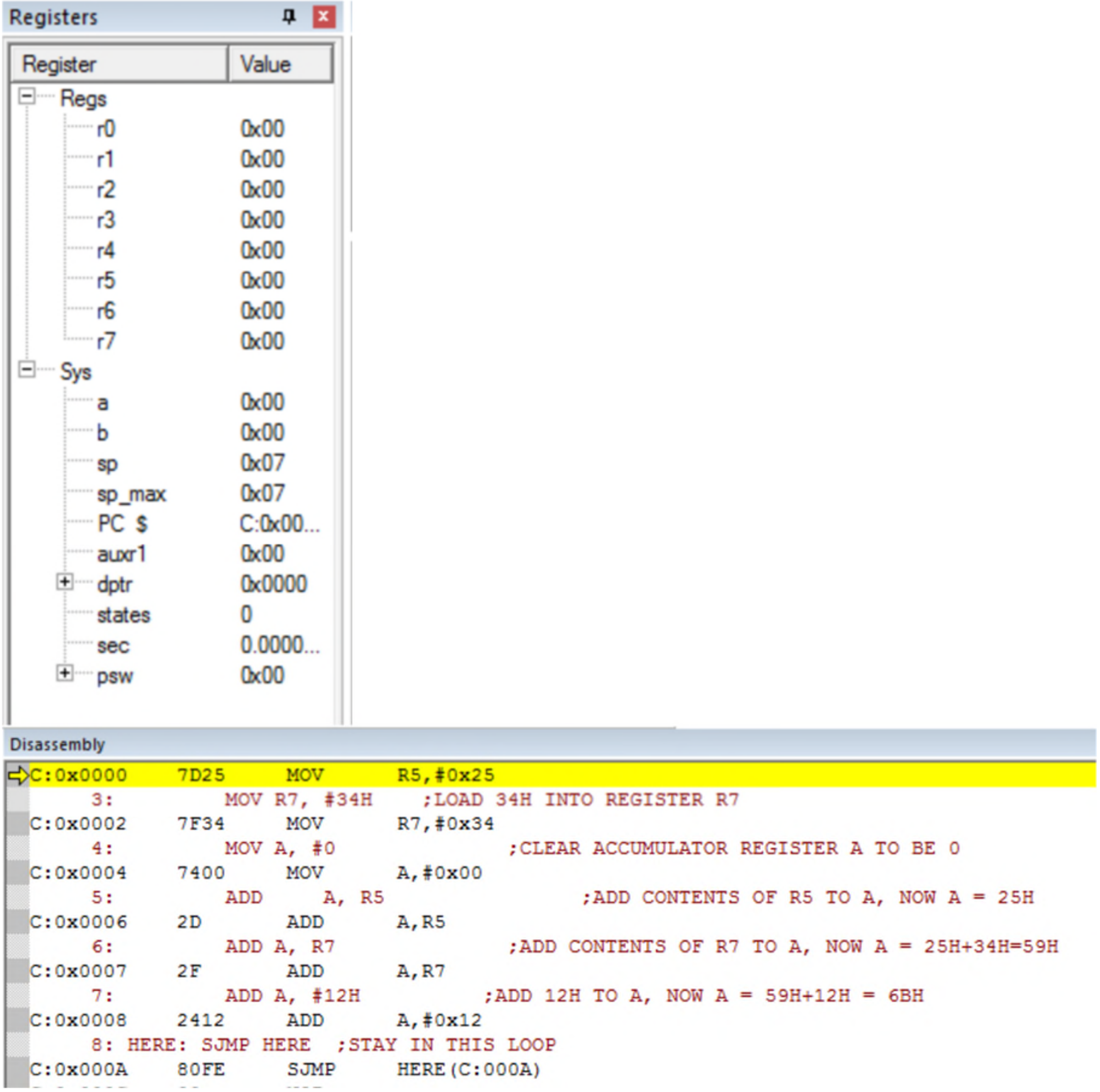
```
assembling lab01_Khan.a51...
lab01_Khan.a51 - 0 Error(s), 0 Warning(s).
```

c. Build output after the Build



```
Build target 'Target 1'
linking...
Program Size: data=8.0 xdata=0 code=12
creating hex file from ".\Objects\lab01_Khan"...
".\Objects\lab01_Khan" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:00
```

d. Register Window and Disassembly Window after the Debug



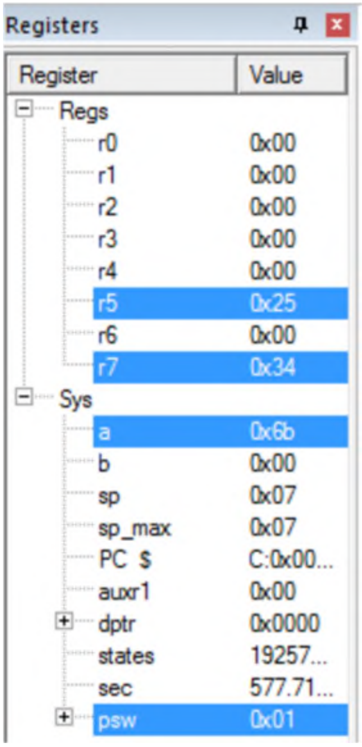
```
Register        Value
⊟ Regs
    r0          0x00
    r1          0x00
    r2          0x00
    r3          0x00
    r4          0x00
    r5          0x00
    r6          0x00
    r7          0x00
⊟ Sys
    a           0x00
    b           0x00
    sp          0x07
    sp_max      0x07
    PC  $       C:0x00...
    auxr1       0x00
  ⊞ dptr        0x0000
    states      0
    sec         0.0000...
  ⊞ psw         0x00
```

```
C:0x0000    7D25     MOV      R5,#0x25
     3:         MOV R7, #34H    ;LOAD 34H INTO REGISTER R7
C:0x0002    7F34     MOV      R7,#0x34
     4:         MOV A, #0              ;CLEAR ACCUMULATOR REGISTER A TO BE 0
C:0x0004    7400     MOV      A,#0x00
     5:         ADD     A, R5                ;ADD CONTENTS OF R5 TO A, NOW A = 25H
C:0x0006    2D       ADD      A,R5
     6:         ADD A, R7              ;ADD CONTENTS OF R7 TO A, NOW A = 25H+34H=59H
C:0x0007    2F       ADD      A,R7
     7:         ADD A, #12H          ;ADD 12H TO A, NOW A = 59H+12H = 6BH
C:0x0008    2412     ADD      A,#0x12
     8: HERE: SJMP HERE  ;STAY IN THIS LOOP
C:0x000A    80FE     SJMP     HERE(C:000A)
```

e. Address 0003 contains which opcode or which operand?

The MOV operand

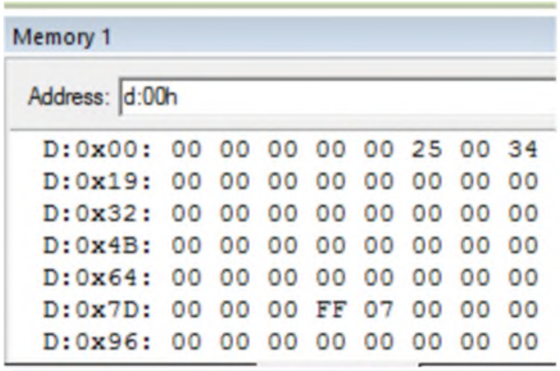
f. What is the values of registers R5, R7, and A when PC is pointed to address 0004? Attach corresponding screenshots here.



Registers

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x25
r6	0x00
r7	0x34
Sys	
a	0x6b
b	0x00
sp	0x07
sp_max	0x07
PC $	C:0x00...
auxr1	0x00
dptr	0x0000
states	19257...
sec	577.71...
psw	0x01

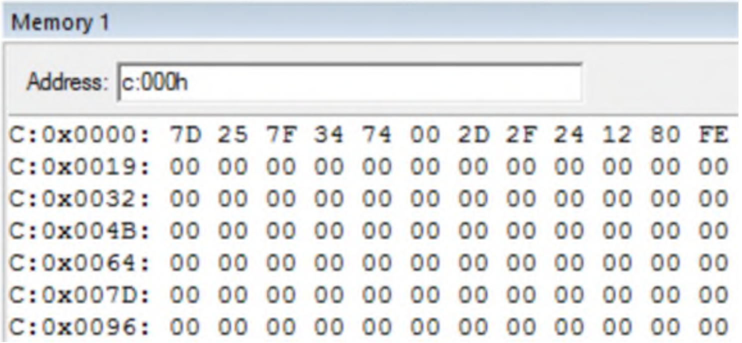
g. Memory window



Memory 1

Address: d:00h

```
D:0x00: 00 00 00 00 00 25 00 34
D:0x19: 00 00 00 00 00 00 00 00
D:0x32: 00 00 00 00 00 00 00 00
D:0x4B: 00 00 00 00 00 00 00 00
D:0x64: 00 00 00 00 00 00 00 00
D:0x7D: 00 00 00 FF 07 00 00 00
D:0x96: 00 00 00 00 00 00 00 00
```



Memory 1

Address: c:000h

```
C:0x0000: 7D 25 7F 34 74 00 2D 2F 24 12 80 FE
C:0x0019: 00 00 00 00 00 00 00 00 00 00 00 00
C:0x0032: 00 00 00 00 00 00 00 00 00 00 00 00
C:0x004B: 00 00 00 00 00 00 00 00 00 00 00 00
C:0x0064: 00 00 00 00 00 00 00 00 00 00 00 00
C:0x007D: 00 00 00 00 00 00 00 00 00 00 00 00
C:0x0096: 00 00 00 00 00 00 00 00 00 00 00 00
```