Lab Week 2- 01/10/2014
Eng. Muhammad Sami

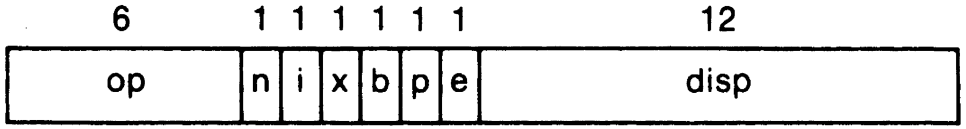
1. Go to the course website: http://moodle.manalhelal.com/course/view.php?id=5
2. Download the "sicsim.jar" from Week 0 material, and run the file.
3. Download the "add.asm" in a folder in your computer.
4. In the "sicsim.jar", click "Load asm" button, and navigate to where you saved the "add.asm" file and load it.
5. The file contains the following code:

```
ADD    START    3000
       STA      SHELLAD
       LDA      FIRST
       ADD      SECOND
       STA      THREE
       LDL      SHELLAD
       RSUB
.      HLT


FIRST      WORD     5
SECOND     WORD     7
SHELLAD    RESW     1
THREE      RESW     1
           END  ADD
```

6. The columns in the CPU view window are:
   a. First Column: Memory Address in Hexadecimal. You will find the memory of the first instruction 00BB8, which is the hexadecimal value 3000 as per the first instruction that loaded the program in a particular memory location.
   b. Second Column: Instruction Code in Hexadecimal using Format 3 as follows:

Format 3 (3 bytes):



c. Third column translates the operand memory address using the given flags. For example the second instruction "`STA SHELLAD`", has the instruction code 0F2015, which in format 3 in binary form becomes:

0000 1111 0010 0000 0001 0101

The instruction code is shown in red bits. In this example it is 0C, which is STA according to the SIC Instruction set shown in appendix A in the book.

The nixbpe black bits are the flags used to decide the addressing mode. In this example: n = 1, i = 1, therefore simple addressing is used, x = 0, therefore the address in the Index Register X is not used, b = 0, therefore the address is not base relative (doesn't add the address in the B register), p = 1, therefore the addressing is PC relative (add the content of the Program Count (PC) register) to the address in disp field; e = 0, therefore it is not format 4 as wee see from the length of the instruction.

The displacement field blue bits are the three bytes used in addressing as operand, value, address, … etc based on the flags used. In this example it is 015m

Translating this command, it becomes "STA (PC)+15"

d. Fourth column list the instruction "STA" in this example.
e. Fifth column translates the address to "STA (PC)+15" in this example.
f. Sixth column shows the

7. Watch the Registers contents change as you load, execute, and store results, by doing the following.
   a. Click the "Step" button to execute the first instruction and move the PC to the following instruction.
   b. First "Step" will store the value in the Accumulator in the memory address given which is (PC) + 15 <- A, which is "00BD0" , will be zero in first run.

c. Click the "Step" button again will load the memory content of 00BCA into Accumulator A, which is the constant declared in the line:
   FIRST WORD 5
d. Click the "Step" button again will add the value in Accumulator A to the memory content of 00BCD, which is the constant declared in the line:
   SECOND WORD 7
e. Click the "Step" button again will store the result from the Accumulator A (which is C) to the memory content of 00BD3, which is the variable declared in the line:
   THREE RESW 1

f. Click the "Step" button again will load the memory content of BD0 to the L register (which is 0) to the such that the following command RSUB, return from subroutine to this address.
g. Click the "Step" button return from the subroutine to the address in the L register (which is 0).

8. Check the memory view as well by going to 00BB8 where the program started.

After studying the instruction codes and how the memory changes after each execution, change the program to subtract two numbers and submit your code in moodle in lab1 assignment.

Goood luck.