Simple simulator lab

This lab works with the toy CPU simulator SimpSim.exe. You should download the program so that you can use it. (It runs on a PC, but not a MAC. Alas.)

Documentation on the Simple machine may be found in chapter 2 and in Appendix C of Brookshear.

Entering data into the Simple simulator

When you start up the program it shows the contents of the 256 main memory locations, the 16 general registers, the PC and the IR (all initially have zeros in them.) Each location contains one 8-bit byte, which is displayed as a two-digit hexadecimal number.

You can type new values into any of these by clicking the mouse cursor over them, then typing the new value. Register RF has the unusual property that when you enter an ASCII code into it, the corresponding character appears on the little screen below. This is an interesting way to see what characters correspond to various hexadecimal codes.

Instructions for the Simple machine

The Simple machine is a sort of desk-calculator, and you can use it to do (very) simple arithmetic. To do so, you need to enter both the data to work with and the calculation commands into main memory. Then you can use combinations of possibly changing) the PC register, the RUN button, and the STEP button to do the calculation.

Here is a very simple problem to add two numbers.

- you enter the following program into main memory at locations as shown

  *[Since each instruction is two bytes long, I've shown the address of the first byte at the left, and the two bytes of the instruction following it, with the **mnemonic explanation** following it on the line. This is the same way the memory contents are explained in the window at the bottom of the Simple machine.]*

```
80:  11,40  load  R2,[40]
82:  12,50  load  R2,[50]
84:  54,12  addi  R4,R1,R2
86:  34,60  store R4,[60]
88:  C0,00  halt
```

- you enter the two numbers to add into main memory locations 40 and 50
- Change the PC register to 80.
- Press the STEP key five times or the RUN key once to run the program.
- The answer will appear in main memory location 60 and in general register R4.
- Once the program is entered, you don't need to retype it to rerun it; you can skip that step.

1. Use the program to find out the sum of 1 and 1, and write that answer here:
2. Run the program again to find out the sum of 23 and 29, and write that answer:
3. Now modify the program so that it adds numbers from locations 20 and 30 instead of 40 and 50.
   - Test your modified program by adding the numbers 88 and 29.
   - You can get a copy of what you've typed into Notepad so you can include it in another document or print it by pressing the DISASM button in the lower right hand corner of the simulator. Do that and print it out to hand in with the lab.
4. Now write a new program which adds four numbers from main memory locations 20, 30, 40, and 50. Test it on the numbers 1,2,3,4.
5. Print out your "add four" program to hand in.

Looping

One of the irritations of the previous program is having to constantly change the PC register when you want to run it. Of course that's an advantage, too, because you can have several different programs ready in the computer, and choose which one to run by changing the PC register. But if you'll be running the same one again and again, there are two different easy ways to avoid that step.

- If you start your program at 00 instead of 80, you can reset the PC with the =0 button.
- If you know that you'll be running the program over and over, you can put a JMP instruction after the halt, so that when you start the second time, the first instruction takes you back to the beginning of the program. For my example program above, I would have put

```
8A:  B0,80  jmpEQ R0=R0,80
```

after the halt instruction. Then if I press RUN again, the program continues after the halt, the jmpEQ instruction changes the PC for me to 80, and it reruns.

For some programs, you don't even want to halt. Here's a program which displaysall the ASCII codes on the screen, again and again:

```
90:  2F,00  load  RF,00      ;... ($00=0)
92:  27,01  load  R7,01      ;... ($01=1)
94:  5F,F7  addi  RF,RF,R7   ;...
96:  B0,94  jmpEQ R0=R0,94   ;...
```

If you set the PC to 90 and run this program, it goes forever... or at least until you stop it. Pressing the BREAK key is a good way to do so. It's kind of interesting to use the STEP key to follow exactly how so many characters end up on the screen.

You can make this program stop by itself by changing main memory byte 96 and onward as follows:

```
96:  BF,9A  jmpEQ RF=R0,9A   ;...
98:  B0,94  jmpEQ R0=R0,94   ;...
9A:  C0,00  halt
```

This comes to a stop when RF gets to have the same value as R0.

- What is the last character shown on the screen when it stops?

Your name in lights

The goal of this exercise is to get your name to print in the little black window of the simulator. To do that, you load the characters into register RF, one at a time. You'll probably want to use the *Load register with bit pattern* instruction, 2, to load up the characters into register RF. For example, the instruction

```
A0:  2F,53 load RF,53
```

draws an upper case "S" on the screen.

Separate your first and last name with a *space* character, which has ASCII code 20.

6. Write a program to display your name, and print out the program.
7. Change the program so it runs forever, drawing your name over and over. Print out the program.