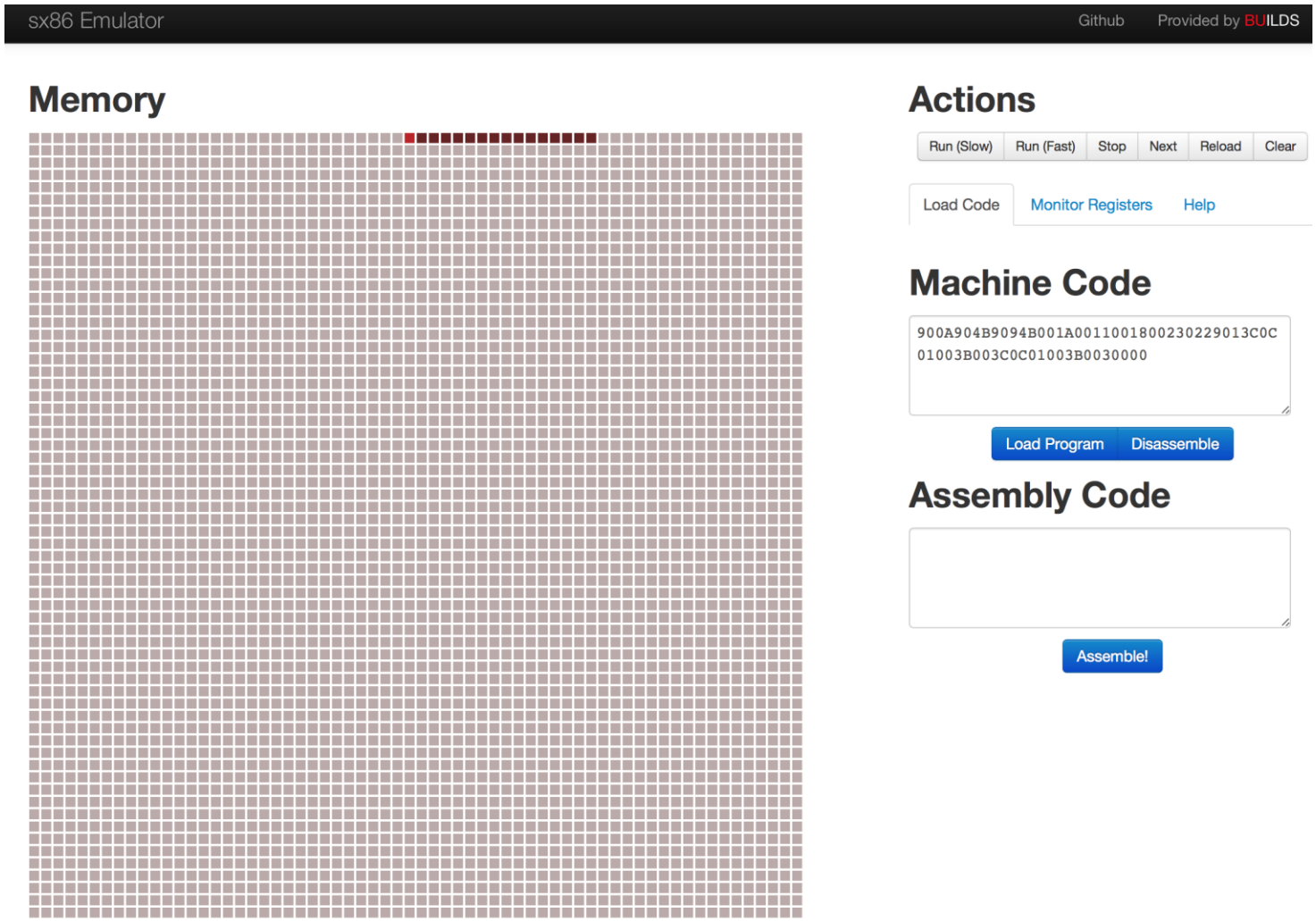
# HOW TO USE THE EC 327 SX86 ASSEMBLY SIMULATOR

This simulator has been developed by one of our former students at BU, Christopher Woodall. It can help you test your answers for HW1 before submission. EC 327 staff highly recommends you use the simulator prior to submission.

1) The simulator can be found here:

   http://cjwoodall.com/misc/sx86-emulator/

   Once it starts, you should see the screen like this:



2) Let's run the code for the following problem:

**Problem:** Use memory locations 10-19 to store numbers 0-9, respectively.
**Solution:** The code is below. Try to reason your way through it and see if it makes sense.

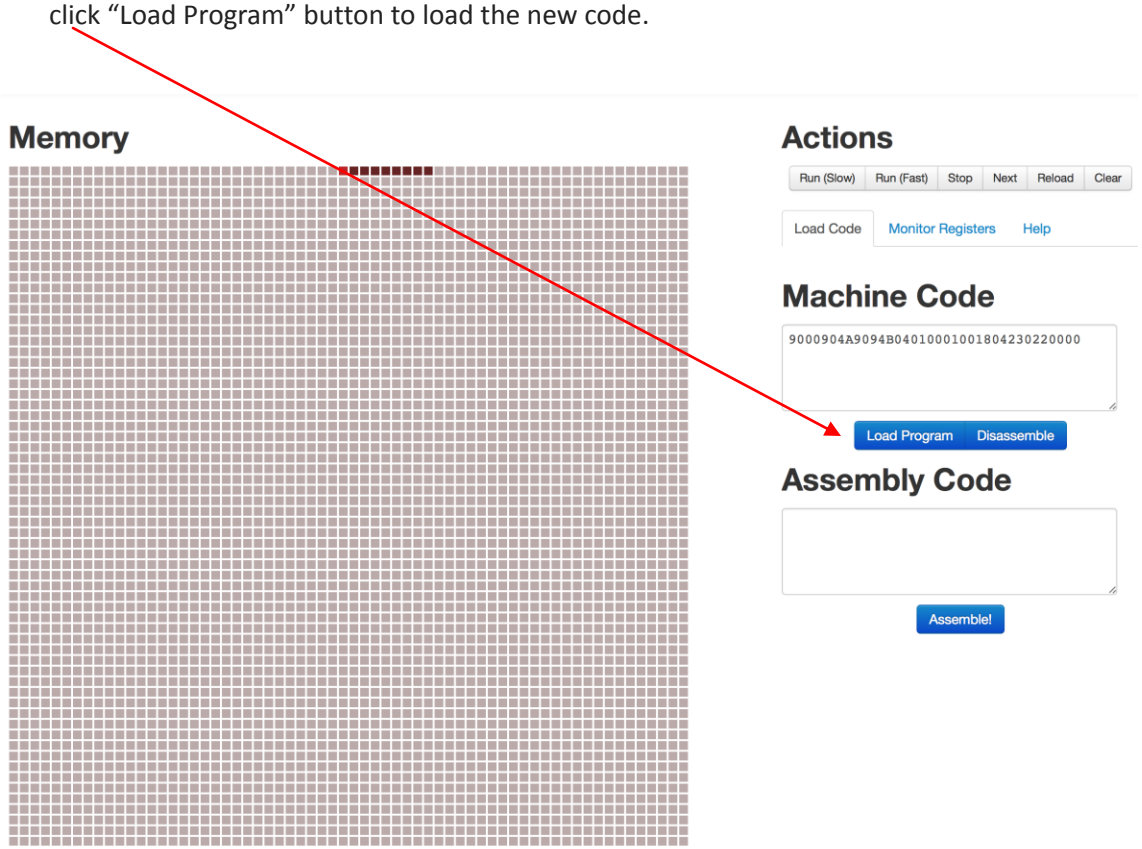
```
Address                Machine code                  Assembly              Comments

31              1001 000000 000000 (Hex: 9000)       mov R0, 0       ;R0 holds 0
32              1001 000001 001010 (Hex: 904A)       mov R1, 10      ;R1 hold 10
33              1001 000010 010100 (Hex: 9094)       mov R2, 20      ;R2 holds 20
34 theLoop:     1011 000001 000000 (Hex: B040)       mov [R1], R0    ;perform the move,in the loop
35              0001 000000 000000 (Hex: 1000)       inc R0          ;R0 increases by 1
36              0001 000000 000001 (Hex: 1001)       inc R1          ;R1 increases by 1
37              1000 000001 000010 (Hex: 8042)       cmp R1, R2      ;compare two registers
38              0011 000000 100010 (Hex: 3022)       jne theLoop     ;jump to address 34 = 0x22
39              0000 000000 000000 (Hex: 0000)       halt            ;halt the program
```

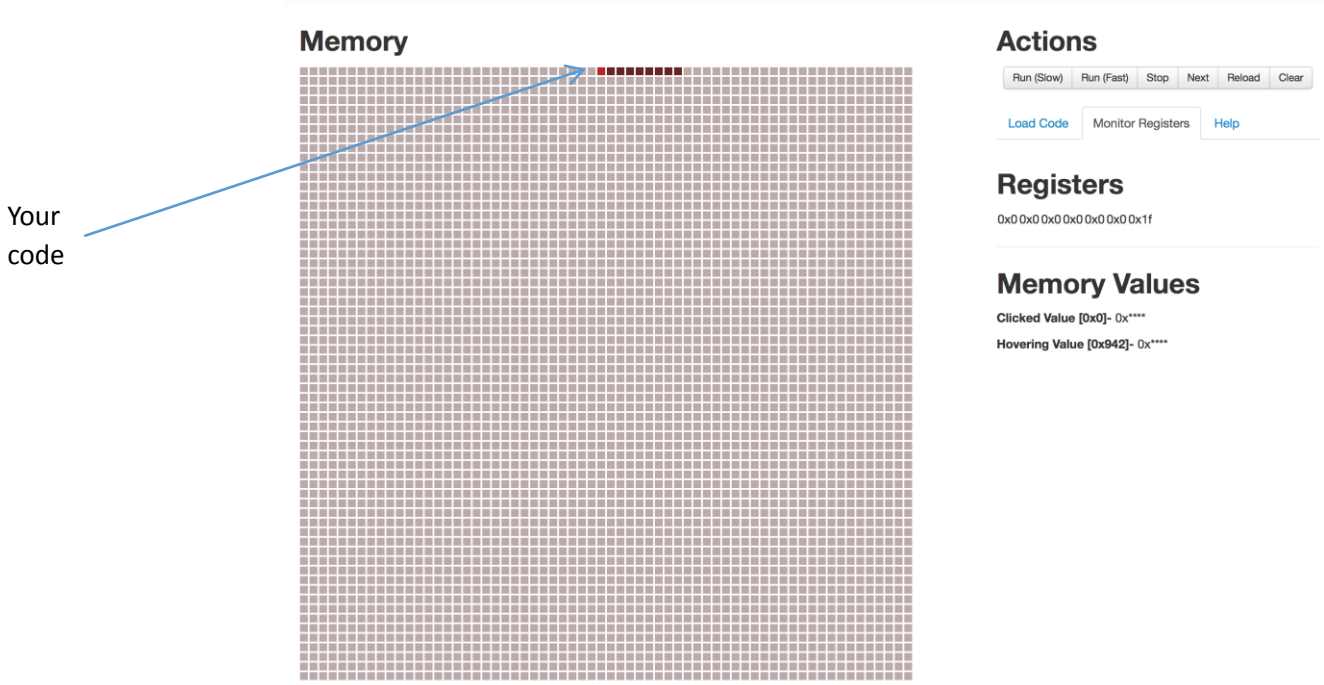
3) While the assembly code is useful for a human to read, we need to feed the simulator the hexadecimal string of each instruction concatenated together. This is:

9000904A9094B0401000100180423022000 0

4) Now, in the simulator, copy/paste this whole line into the "Machine Code" box. Make sure to click "Load Program" button to load the new code.

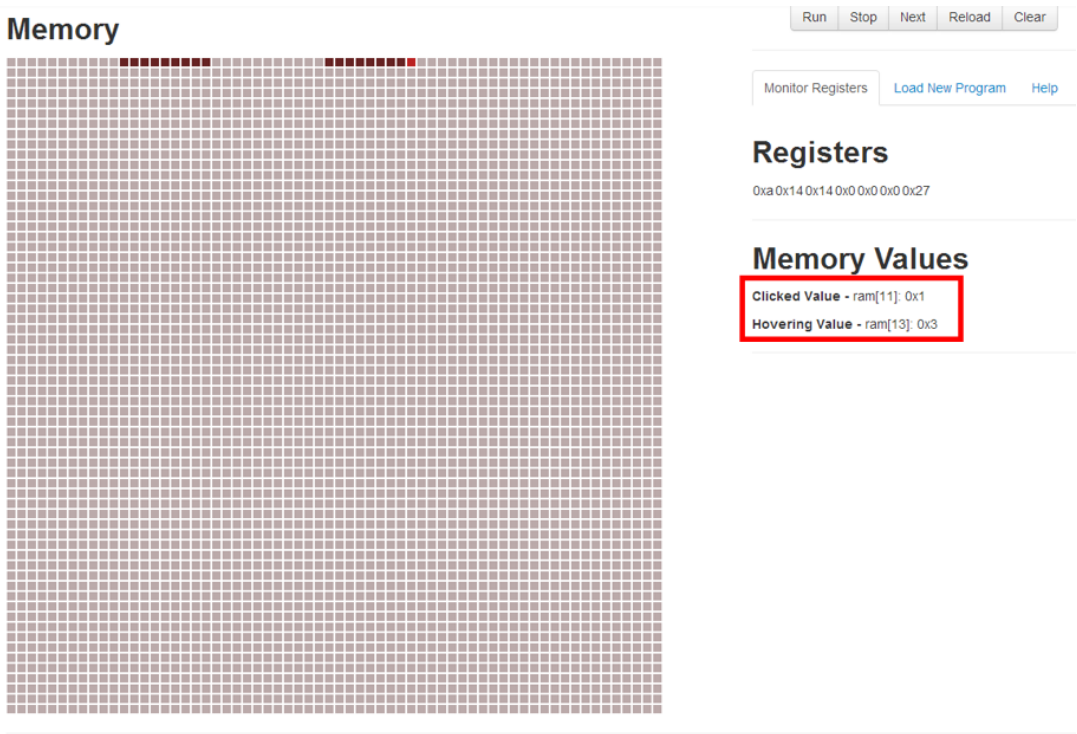


5) If successful, the code will load at memory location 31 (this can't be changed), and the screen will look like this:




6) You can run the code step by step (i.e. one instruction at a time) by clicking "Next" button, or the whole code at once using "Run(Slow)" button, and follow the changes in registers and memory. "Run(Fast)" will run the code to completion immediately.

7) When code finishes, if correct, the screen shot should look like below, with the memory locations 10-19 filled up with numbers 0-9 respectively (you can check that by hovering the mouse cursor over the squares in memory or clicking each square).




8) If the result is not correct, make any necessary changes to the code, click “Load Program”, and start over.

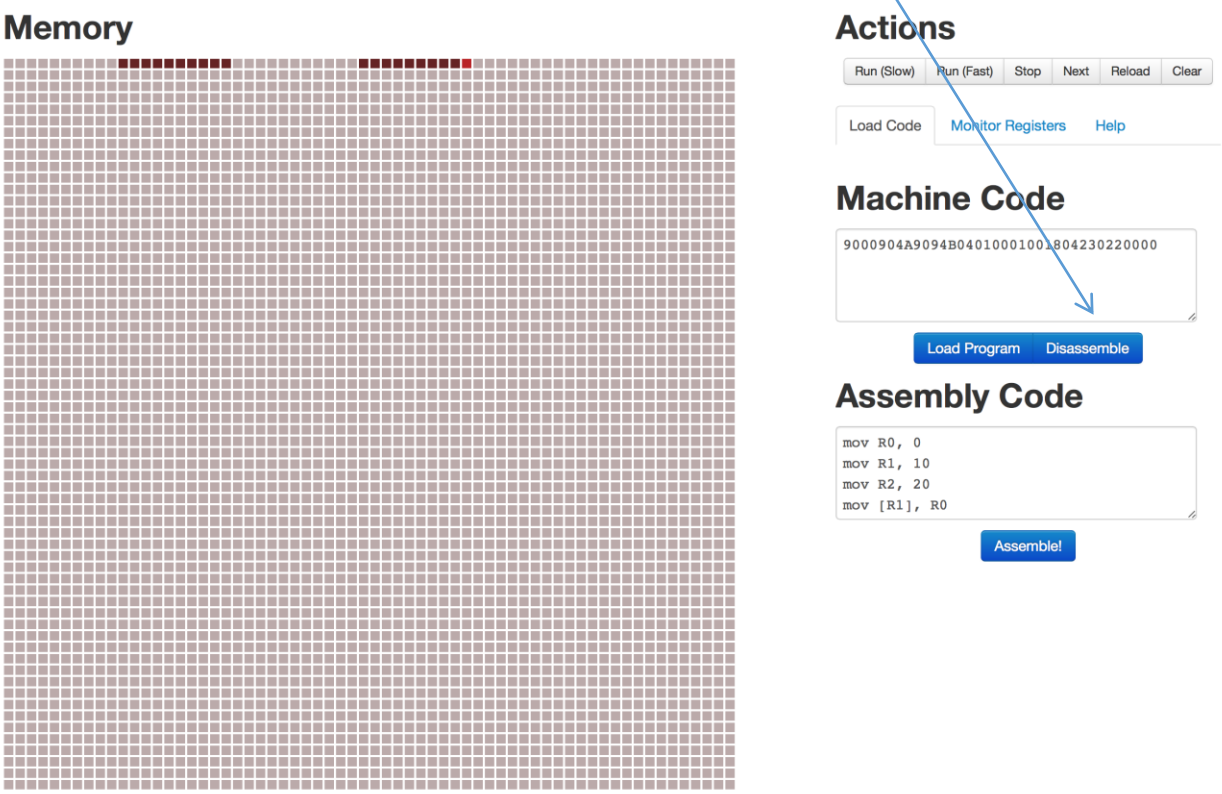
9) That is pretty much it for the simulator. Feel free to try more examples like one-line string:

900090409095B0401000100180423022000 0 - to populate memory locations 0-20 with 0-20
90009040909FB0401000100180423022000 0 - to populate memory locations 0-30 with 0-30

10) If you have any questions about the simulator, please email the staff (Prof. Densmore or any UTF). Very likely, someone around you can help you too.

11) Finally, note that the first command is in memory location 31. Your answers should take that into account, especially when converting loops from assembly to machine code.

12) New to the emulator is the option to “Disassemble” machine code. Click “Disassemble” to turn machine code into Assembly Code; much more readable!




13) Likewise, you can paste valid Assembly Code into the Assembler to generate working machine code. Refresh the page and paste the assembly below. Click “Assemble!” and watch the Machine Code box update with a new program.

```
mov R0, 0
mov R1, 10
mov R2, 20
mov [R1], R0
inc R0
inc R1
cmp R1, R2
jne 34
halt
```