Lab 2 Bitwise Manipulation and Introduction to Assembly Language

Submit lab2.asm by 5:00pm on Jan. 19, 2018. This lab introduces you to the programming environment that we use in CSc 230 lab.

I. Bitwise Manipulation Continues

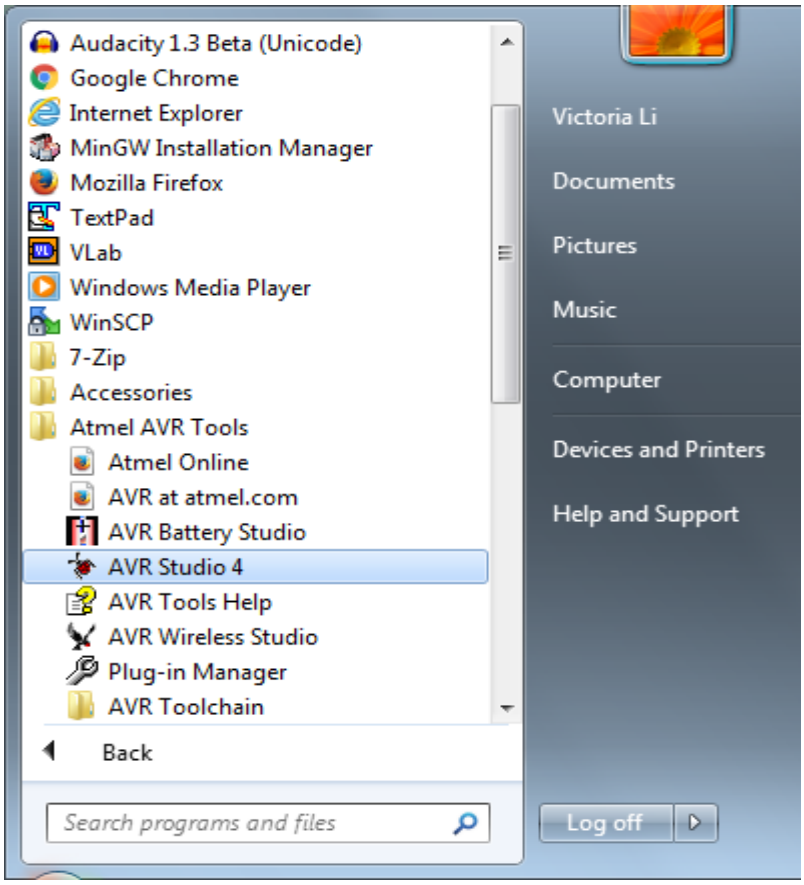
Let's continue with questions 7, 8 and 9 of lab 1. The lab instructor is going to do some demos and you do some exercises.

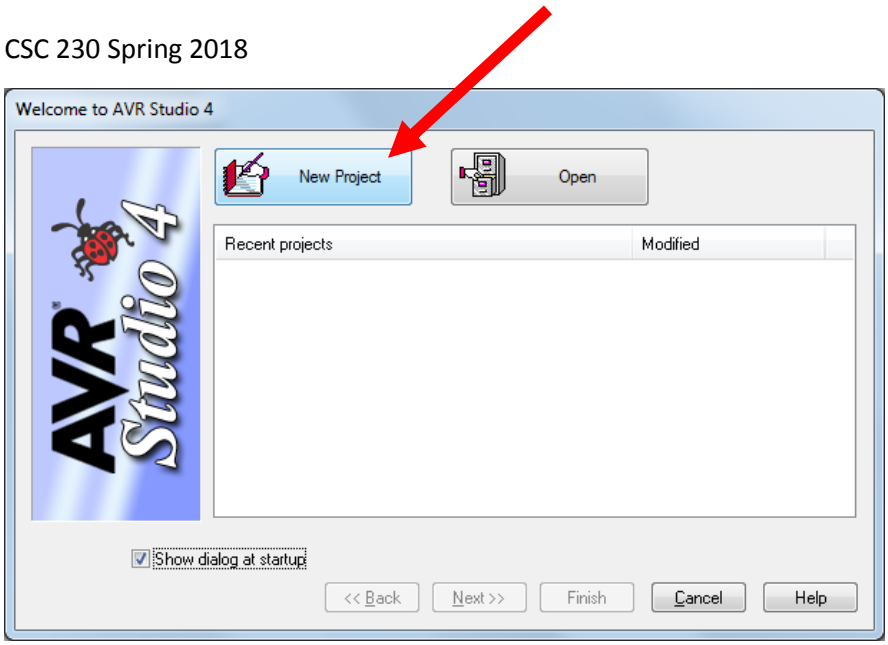
Exercise 4: Determine the result of performing a bitwise OR operation on this pair of nybbles: 0b1001 and 0b 1100 (q23 on page 24 of Margush).

Exercise 5: What mask and operation would be used to zero the upper nibble in a byte? To clear the lower nibble of a byte? (q25 on page 25 of Margush).

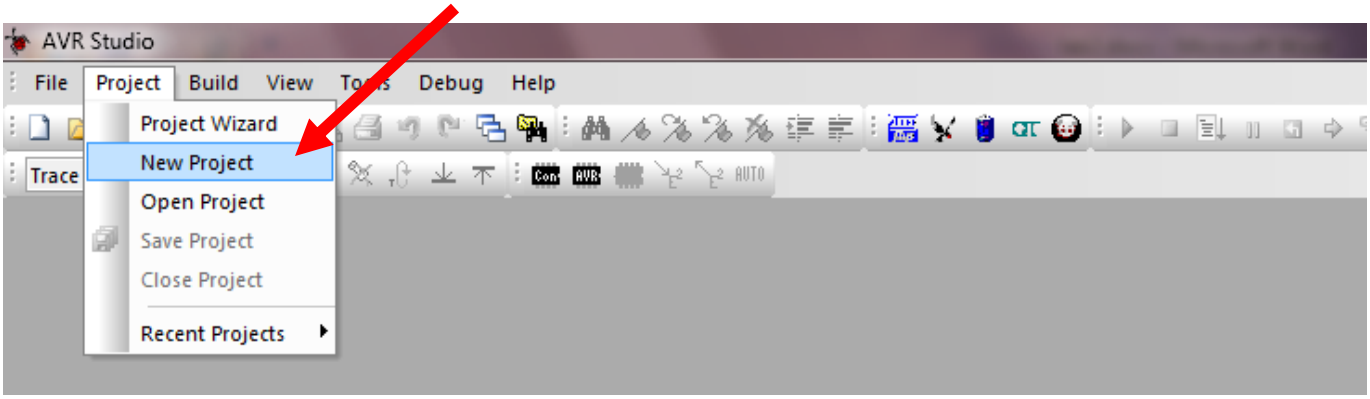
II. Introduction to AVR Studio 4

Launch AVR Studio 4 and create a new project named "lab2"

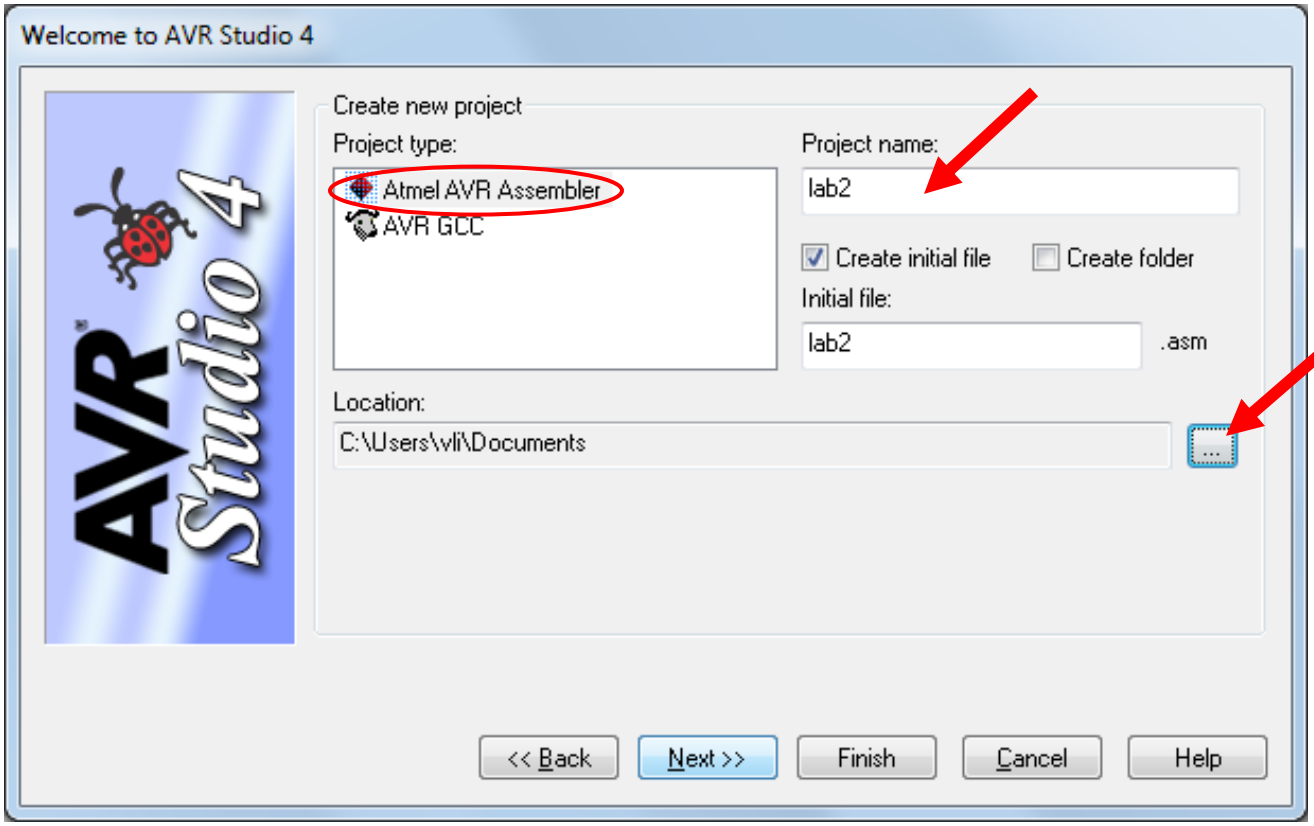




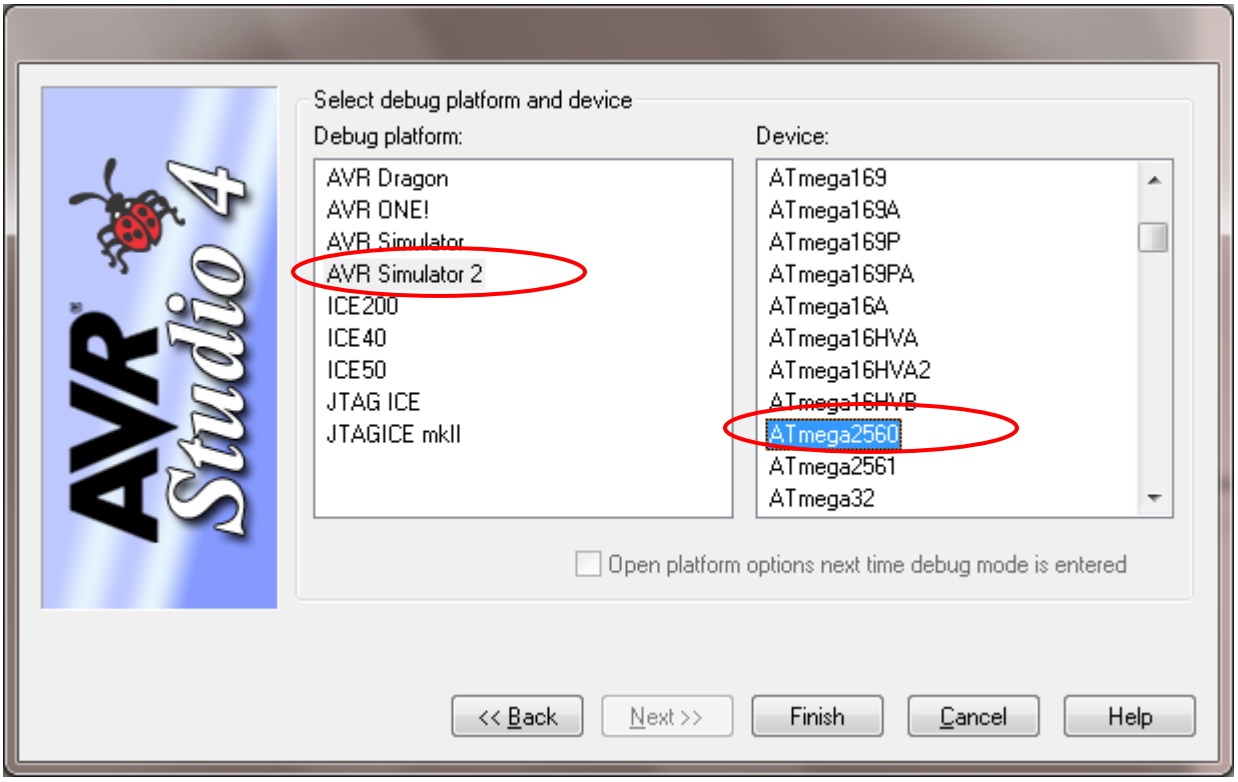
If the program is launched already and you need to create a new project, on the menu, select “Project” -> “New Project”:



Select “Atmel AVR Assembler” for “Project Type”, “lab2” for “Project name” and click on the “…” button to select a directory (suggest H drive).



Click the “Next>>” button and get to this page:
Click on “AVR Simulator 2” for “Debug Platform”
Click on “ATmega2560” for “Device”
Click on “Finish”

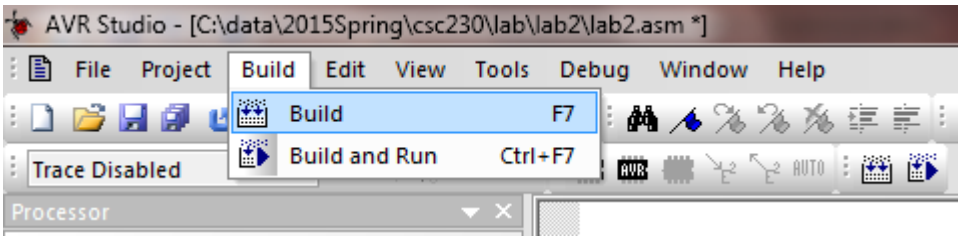


Type the following code:

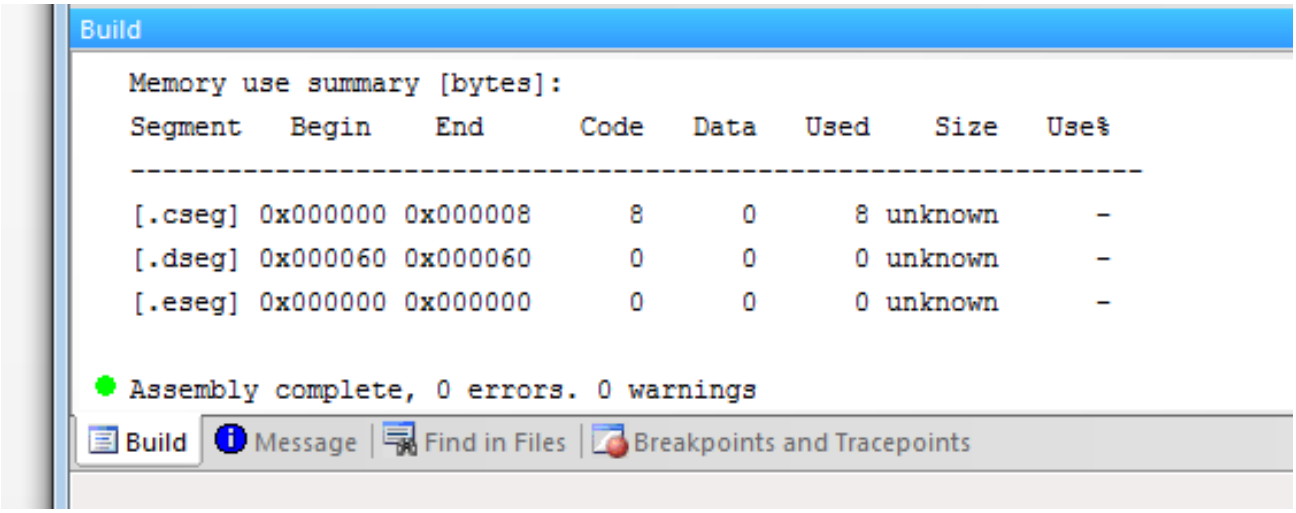
```
.cseg     ;select current segment as code. For more see p56 of Margush
.org 0    ;begin assembling at address 0

;Define symbolic names for resources (eg. registers) used
.def number = r16   ;register 16 holds a number

    ldi number, 0x0A   ;load hex value 0A to register named number
    andi number, 0b00000001   ;clear all the bits except the least significant bit
    done: jmp done
```

Save the code.

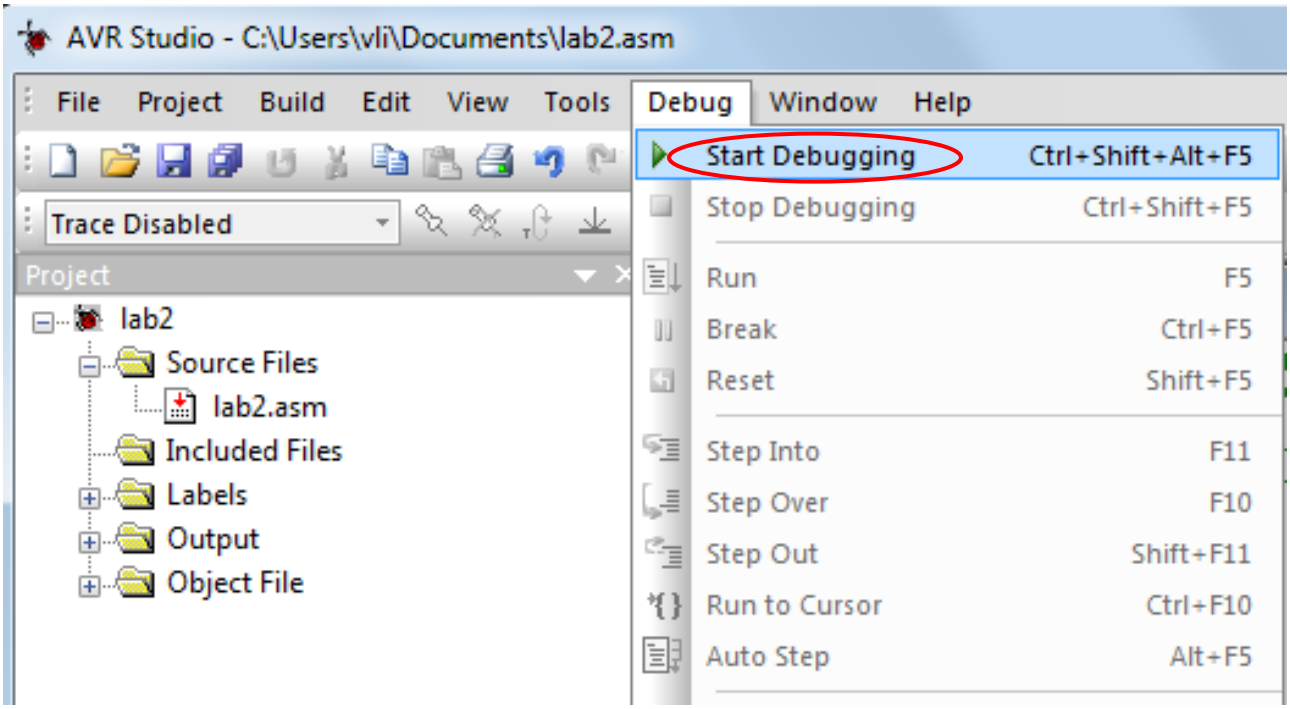
Assemble it (choose Build under the Build menu)



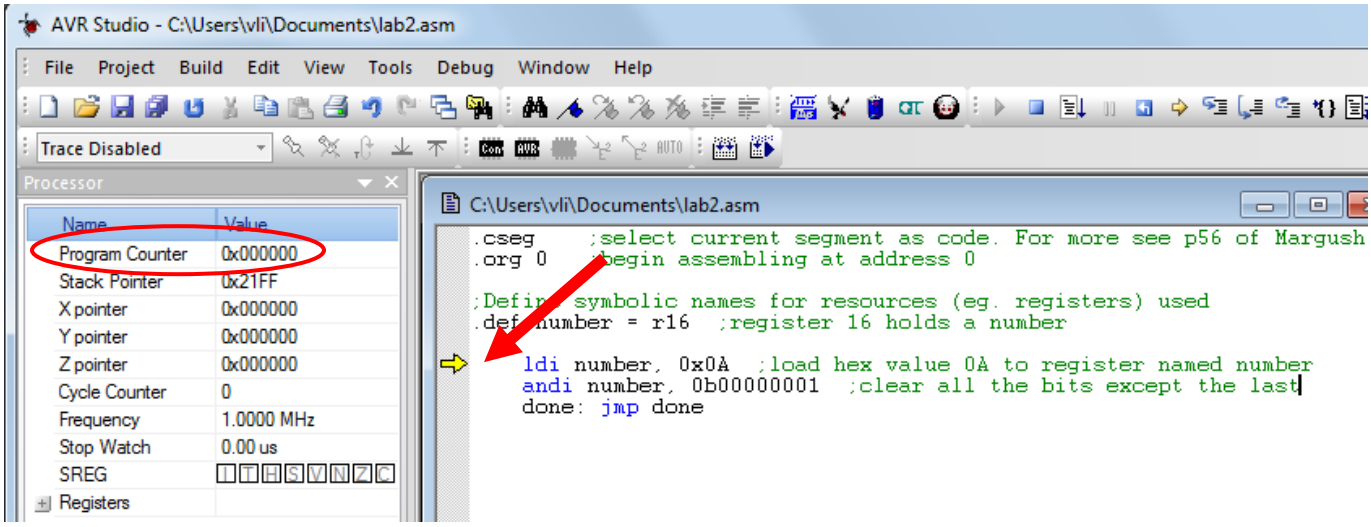
Output from the build at the bottom of the screen:



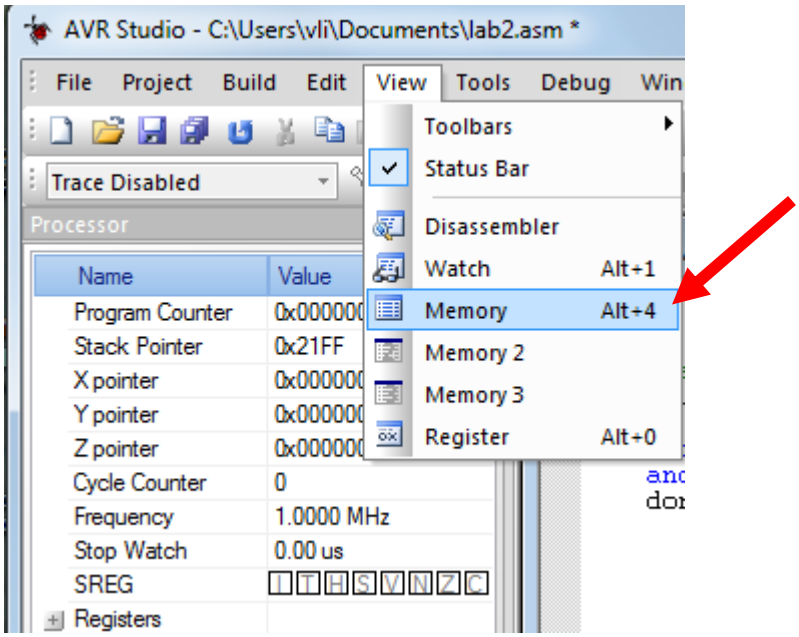
If no errors are detected, start the simulator (debugger). Select "Start Debugging" command under the "Debug" menu.



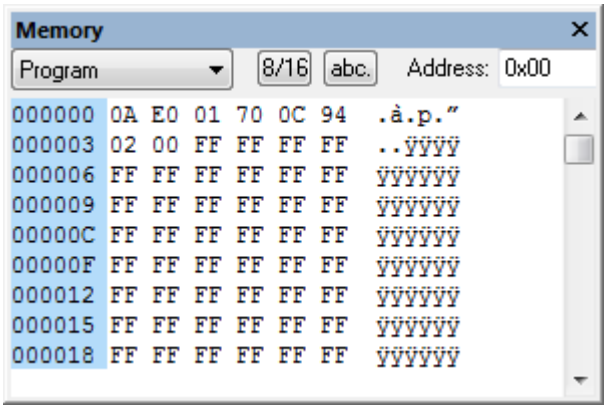
The editor should show a yellow arrow indicating the instruction about to be fetched. The left panel should show the "Processor panel", where you can examine register and processor values.



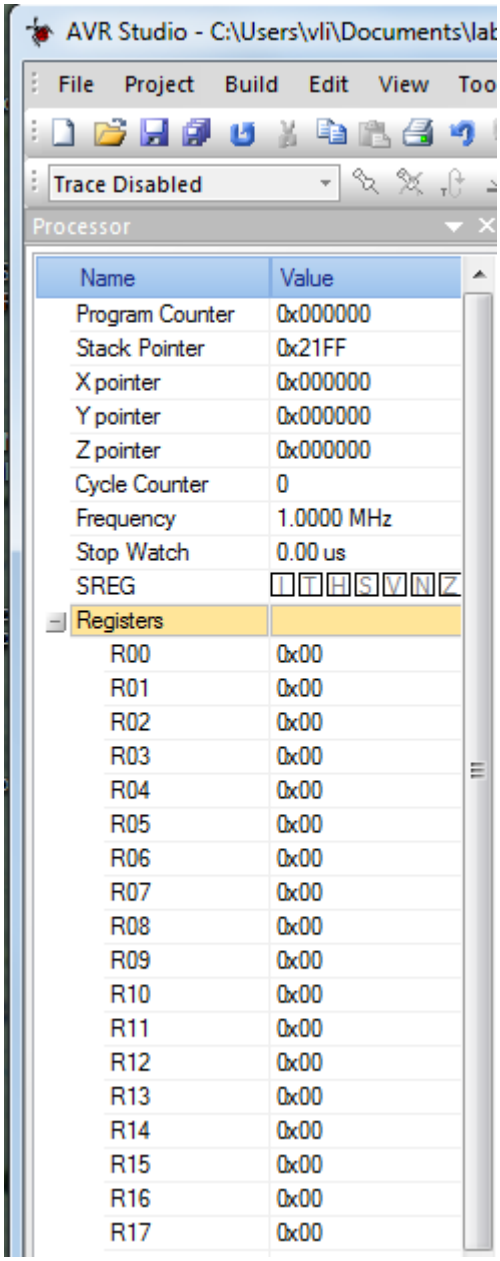
Before executing any instructions, open the "Memory" window (Under the "View" menu):

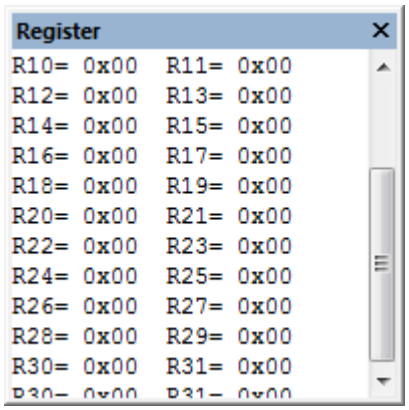


The memory looks like this: (verify the opcode)



Expand the tree node labeled “Registers” (picture on the left), or choose menu “View” -> “Registers” to open a popup window (picture on the right).





Select the "Step Into" option from the "Debug" menu (or press F11) to allow the program to fetch and execute the first instruction. Observe the changes in PC (program counter) and register 16 (r16).

Stop the debugging session by using the "Stop Debugging" command under the "Debug" menu.

Lab exercise: write some opcodes -- such as LSL (logical shift left), LSR (logical shift right), ROL (rotate left), ROR (rotate right), ANDI, ORI, and EOR (exclusive OR), -- and verify their behavior using values store in registers. Play around with masks, too, using these instructions. What other bitwise operations not mentioned here do you find interesting? Play around with those, too. (Pages 276 & 277 of the textbook contain a brief summary of bitwise operations.)

Submit lab2.asm at the end of your lab.

This lab is derived from the Chapter 2 of your textbook (Some Assembly Required by Timothy S. Margush)