EE 281 September 27, 2001
Embedded Systems Design Laboratory Handout #4

Laboratory Assignment #1
"Blinkin' Lights"

Due Monday, October 7, 2002

In this first lab you will become familiar with the AVR Studio software development cycle by writing several small programs to blink the lights on the STK200/500 Development Kit. We will provide you with a short program called blink.asm that cycles through a tight loop reading the buttons on Port D and turning on the corresponding LEDs on Port B. Your assignment is to create three variations on blink.asm.

- Variation #1: Invert which button controls which LED. Button 0 controls LED 7, button 1 controls LED 6, and so on.
- Variation #2: Use each button to toggle the corresponding LED. When a button is pressed and released once, the state of the corresponding LED toggles - if it was on, it turns off, if it was off, it turns on. **You must debounce the buttons in software.** Debounce routines should be designed to "filter out" the electrically noisy, bouncy, pushbutton clicks and ensure that the LED toggles only once per user click. The program should work for any number of buttons being pressed at any time and for any reasonable press duration.
- Variation #3: Toggle each LED only upon a double press of the button. A double press is the button being pressed and released twice within a short amount of time, similar to double-clicking a mouse button. As with variation #2, this program should work for any number of presses, and the buttons should be debounced in software.

Each variation should be derived from the original program, not sequentially from each other

1. Download the sample program blink.asm and the associated include file 8515def.inc from the EE 281 WWW page http://www.stanford.edu/class/ee281/labs/lab1 or from the class directory /afs/ir.stanford.edu/class/ee281/WWW/labs/lab1/.
2. Create an AVR Studio project consisting of only the source file blink.asm. Assemble the program and download it into the STK200/500 development board, and convince yourself that it does what it should.
3. Starting with blink.asm, create copies called blink1.asm, blink2.asm, and blink3.asm. Use these copies to create separate AVR studio projects, in separate folders, that implement the three variations described above.
4. To turn in your lab, zip up all of the project directories, including all your .asm files, the .apr project files, and any other .inc include or other files you've created, together with a ReadMe file in the top-level folder describing your solutions. Make sure that your ReadMe contains your name, SUID, email address, and an explanation of how your code works. Also make sure to preserve the directory structure in your zip file. Email the zip file to the TA.

All programs should be written independently by everyone in the class. It is okay to discuss high-level algorithms, but your code should be your own.