Lab 2: Introduction to counter/timers

Goals:

The main goal of this lab is to become familiar with counters and timers, use the on-chip timer to generate a square wave output, and to implement an interrupt driven timer.

The ATMega1284p timers

These are arguably the most important, and most commonly used, peripheral devices in the Atmel processors. There are three timers in the 1284p chip, two 8-bit counters and one 16-bit counter. There are many similarities, and some differences, in the counters.

A musical keyboard

In this lab, you will extend a timer example from class to produce a set of square wave outputs with different frequencies, and output a particular frequency when each of a set of buttons is pressed. In short, we will make a simple keyboard. Input will be from the push buttons on the STK500, and output (a different frequency corresponding to each key) will be through the small speakers used in the first lab.

Again, the keys can be input through `PORTD`, and the output will be from `OC0A` (pin PB3). The example in class produced an output at 1760 Hz (the note A). For your implementation, there is a sheet listing the frequencies of musical notes, and the values required for the timer assuming the timer uses CTC mode, and the processor clock is 1 MHz. Look in the "External links to material discussed in class" tab in the course home page.

It is possible to implement the keyboard function using either pin change interrupts or polled input. Decide what you prefer, and justify your decision. In the lab writeup, explain how each option could be implemented.

Optional:

Implement *both*, and comment on under what circumstances you might wish to use either of the implementations.

A real-time clock

Keeping track of *real* or *clock* time is important in many applications, and many of the AVR processors have some facility for this. Atmel has an application note (AVR134: Real-Time Clock using the Asynchronous Timer) on this subject, which you can read. It, too, is referenced in the "External links" page.

In this lab, you will extend the real-time clock example from class to perform a "stop-watch" function — the timer will stop when a button is pressed, and then restart when either the same button is pressed again, or a second button is pressed. (The choice is yours — one button for start/stop, or separate buttons to start and stop. In the first case, care must be taken to "debounce" the button.)

This requires a 32.768 KHz. crystal connected between pins `TOSC1` and `TOSC2` (pins PC6 and PC7). The crystal leads can be inserted in pins 23 and 24 of the socket `EXPANDO` on the STK-500.

Timer output will be using the LEDs on the STK-500. Use `PORTD` for output in this example.

Optional:

Use the two 7-segment displays from the previous lab to display the output in seconds, counting down from 59 seconds.