Creating a Timing Function on the ATmega168

I'll break the creation of a timing function on the ATmega168 into two steps, calculating the number of clock cycles to count and setting up the registers. Timer 2 (an 8-bit timer) will be used in this example. Porting the code to Timer 1 (a 16-bit timer) should be straight forward using the datasheet.

Timer 0, 1 and 2 are actually timers / counters. We'll be using Timer 2 as a counter in counting clock cycles. Since we are working with an 8-bit counter, the highest number of clock cycles we can count is 256 before the counter rolls over to 0. A roll over triggers an interrupt enabling us to count them in a straight forward way. Our first order of business then will be determining the relationship between these interrupts and time.

Our chip operates at about 8 MHz using its internal RC circuit (which, by the way, is very inaccurate). We'll use a prescaler of 8 which will reduce the frequency of Timer 2 to 1 MHz, giving us a clock cycle of

$$\frac{1}{10^6 Hertz} = 1\ microsecond$$

Since our there are 256 cycles for each roll over, the time between roll overs is

$$256 * 1\ microsecond = 256\ microseconds$$

This makes it a little awkward to use so we'll make a change in the counter so that it only counts 250 cycles between roll overs vice 256. To do this, I'll setup the counter to start at 6, not 0, before counting up giving us exactly 250 cycles. Therefore, the time between roll overs will be

$$250 * 1\ microsecond = 250\ microseconds$$

If I count 4 roll overs, I will then have a unit of time that is easy to use:

$$4 * 250\ microseconds = 1000\ microseconds = 1\ millisecond$$

This will be the basis for setting up our routine:

- Use Timer 2
- Set the prescaler to 8
- Begin the counter at 6 vice 0
- Enable the roll over interrupt
- For every 4 calls to the interrupt, update a variable called mseconds.

The code for this follows.

```
#include <avr/io.h>
#include <avr/interrupt.h>

unsigned int timecount = 0;
unsigned int mseconds = 0;

ISR (TIMER2_OVF_vect)
{

        TCNT2  =  6;                    /* start counting at 6 for 250 clock cycles */

        if (++timecount  ==  4)         // every 4 interrupts equals one millisecond
        {
                timecount = 0;
                ++ mseconds;
        }

}

int main(void)
{
        TCCR2B  =  (2 << CS20);         //Set prescaler to 8
        TCNT2  = 0;                     //Counter starts at 0 (will change in Interrupt)

        TIMSK2  =  (1<<TOIE2);          //Enables timer overflow interrupt

        SREG  =  (1 << SREG_I);         // Enable global interrupts

        while(1);
}
```

To get the time in another function, you would just use the variable “mseconds”. Notice that this is not a function as it was on the Handy Board (e.g., mseconds()).