Michael Eller

ECE 3430

Lab 4

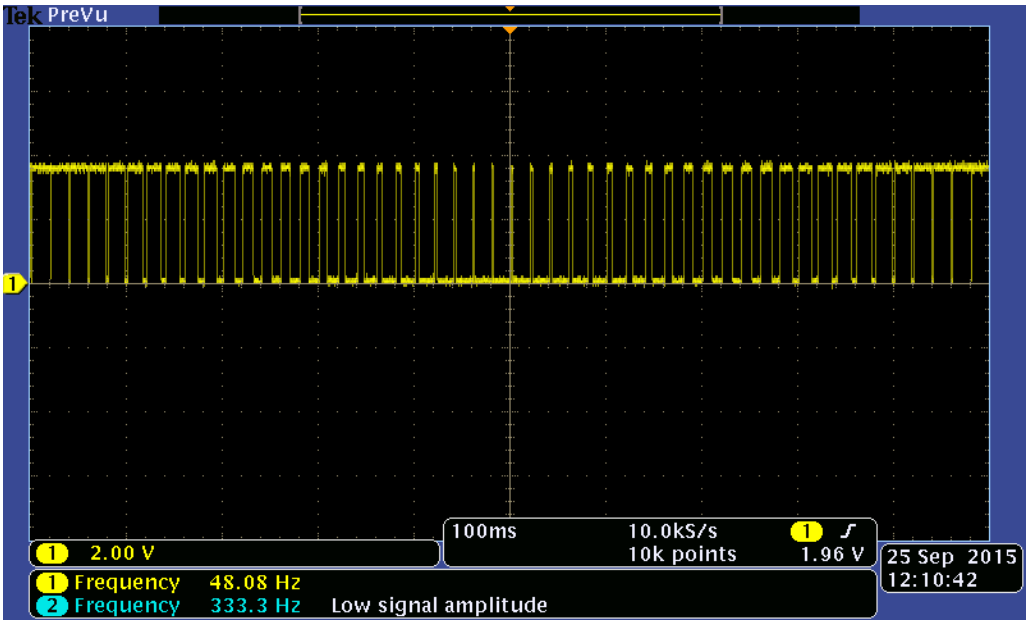
28 September 2015

PWM

Goals:

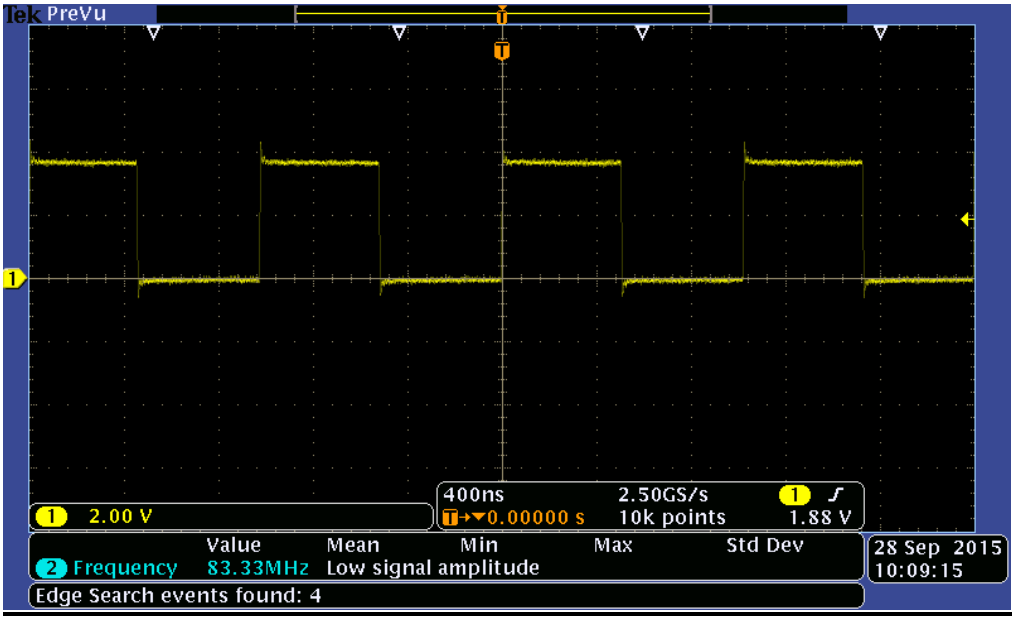
The purpose of this lab was to use the timers on the MSP430 to manually modulate the pulse width of the on-board LED. The button was also used to 'pause' the PWM. Several oscilloscope readings were taken to confirm the laboratory requirements.

In-Lab:

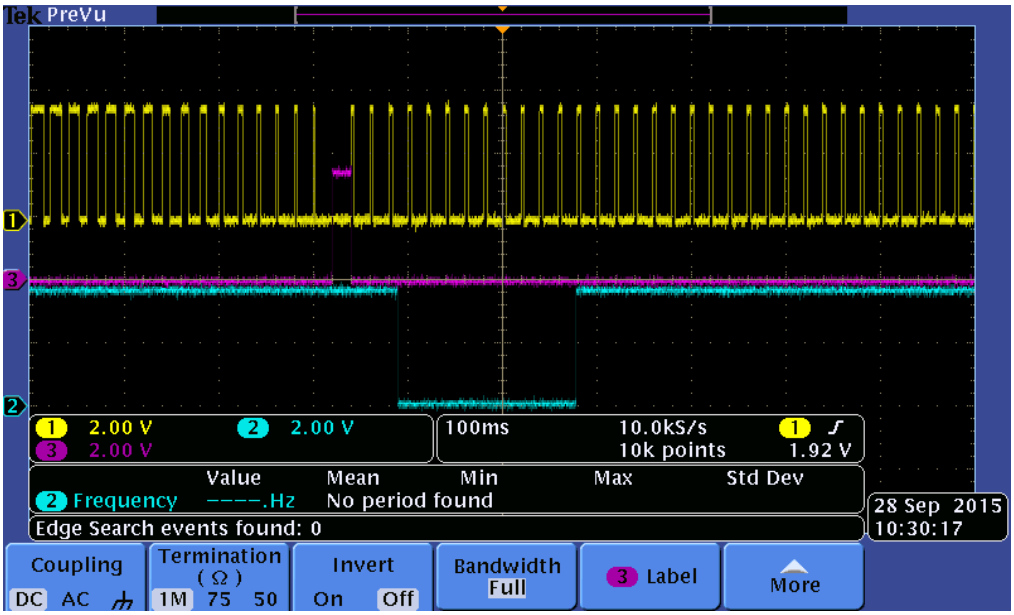


This screenshot shows the duty cycle of the red LED over a one second period. Each division was 100 ms and the volts per division setting was set to 2.0 V.

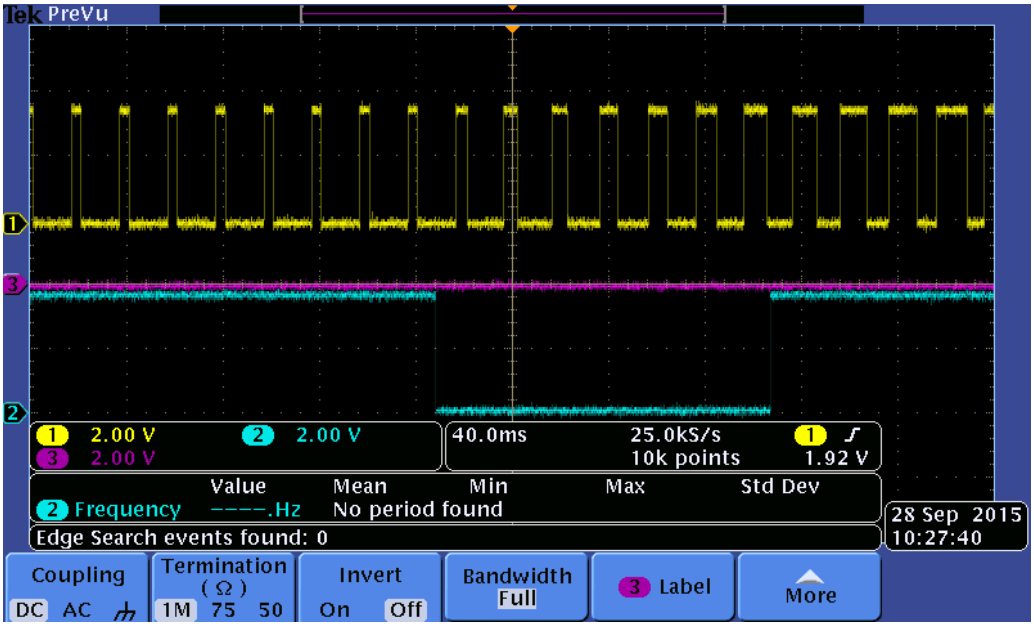
Post-Lab:



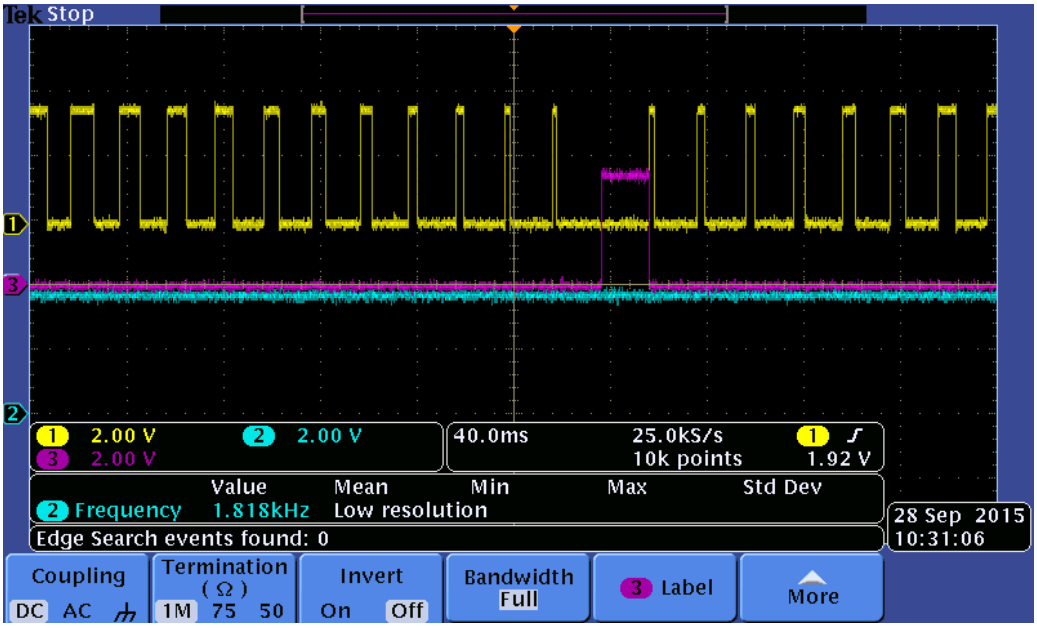
The picture above shows the SMCLK frequency. The time scale was set to 400 ns per division and the volts per division setting was set to 2.0 V. The period was observed to be 1 µs, which corresponds to a clock speed of 1 MHz.



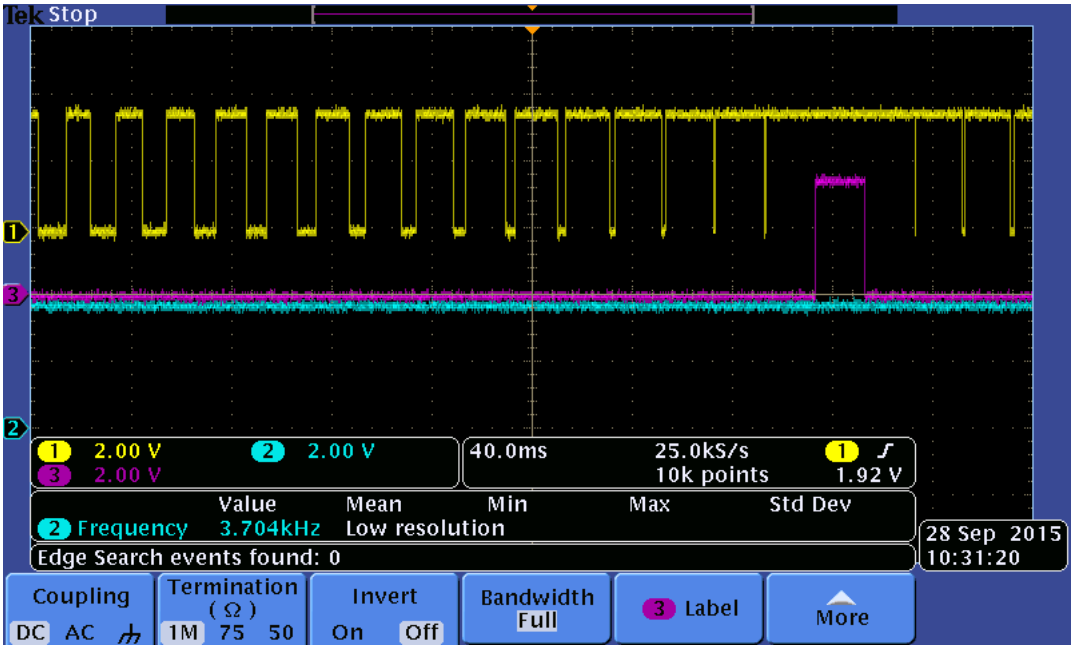
This screenshot shows the ‘pause’ feature of the button interrupt. The time scale was set to 100 ms. The pulse width was decreasing, but when the button (blue line- active LOW) was pressed, the pulse width stayed constant.



This shows the reverse of the last screenshot. When the button was pressed, the pulse width changed from constant to increasing.



This scope shot shows the PWM changing direction. The green LED (purple line) turns on when the modulation changes direction.



This scope shot shows the reverse of the last picture. The green LED pulsed again when the LED shifted from getting brighter to getting dimmer.

Code Explanation:

The main pulse modulation was accomplished in the Timer_A0_routine. Based on a global variable ‘dir’ (1 = brighter, 0 = dimmer), the logic decisions were designed to increase the limit of Timer_A1, which simply turns off the red LED, to the period of the main timer or decrease the limit to 0. A ‘count’ variable was used to make the Timer_A0_routine resemble a ‘for loop’ from 0 to 25, so ‘TACCR1’ (Timer A1 limit) was increased from 0 to 2500 or vice-versa.

The global variables ‘inc’ and ‘count_inc’ were necessary to implement the button feature. When the button was pressed, both of the incrementing values were changed to 0; therefore, the pulse width never changed and ‘count’ was never increased. Pushing the button again set ‘inc’ and ‘count_inc’ to 100 and 1 respectively.

The code was also broken into several header and source files. This is essential for writing quality code since it results in much more organized code. It also makes the code easily transferrable to future projects.