CPE 329
Assignment 1
Derek Nola and Bevin Tang

1. Light Pulses

a. Demonstration of one-second pulse with a minimum of three clock frequencies via a YouTube video link:

https://youtu.be/Kb1UBPaysVc

b. Documentation of accuracy of the one-second pulses with three different clock frequencies.

Figures 1, 2, and 3 each show a one second pulse at different frequencies. Each pulse has two cursors that measure the difference in time between the beginning and end of the pulse. These cursors show that the pulse is up for one second for all three frequencies in question.

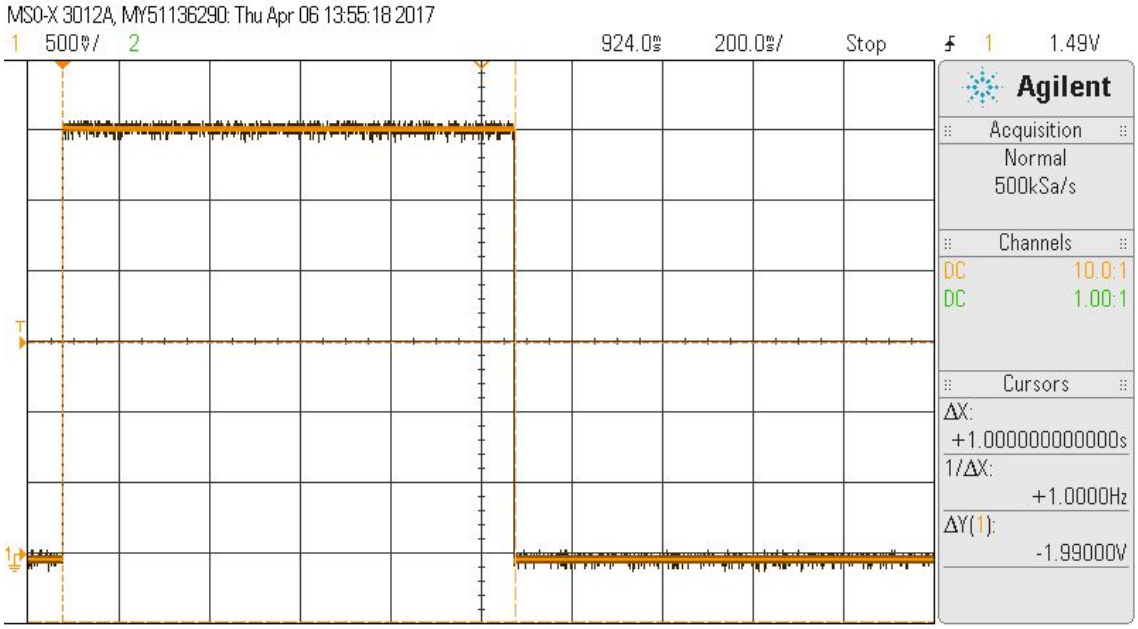

Figure 1: One-second Pulse at 12 MHz Clock Frequency

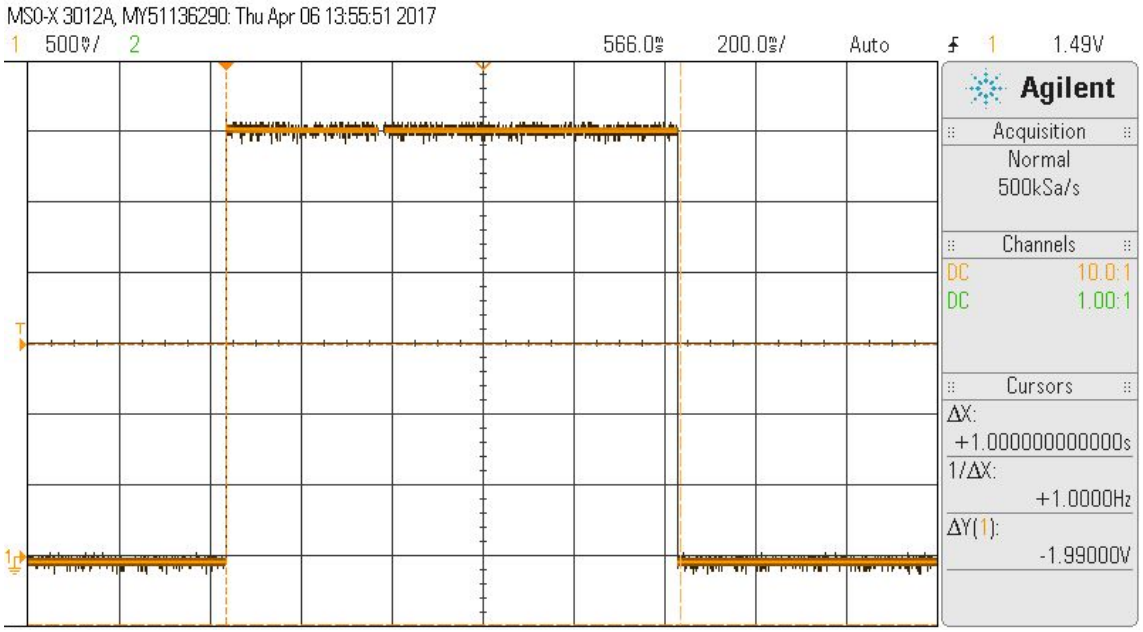

Figure 2: One-second Pulse at 6 MHz Clock Frequency

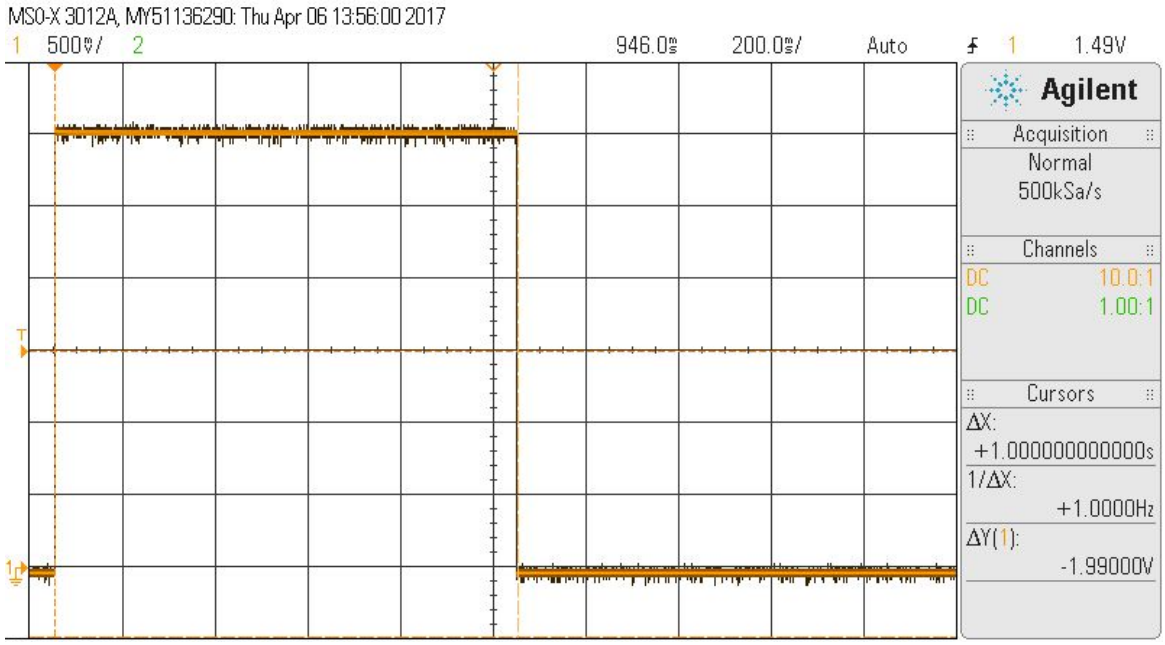

Figure 3: One-second Pulse at 3 MHz Clock Frequency

c. Two 1 microsecond pulses using normal triggering. Confirm that the pulses are accurate to within 5% or better.

The lowest possible pulse we could attain with the device's limits and our algorithm were 14us. But these pulses were not reliable, so decided to create two 100us pulses instead. Figure 4 shows these two pulses.

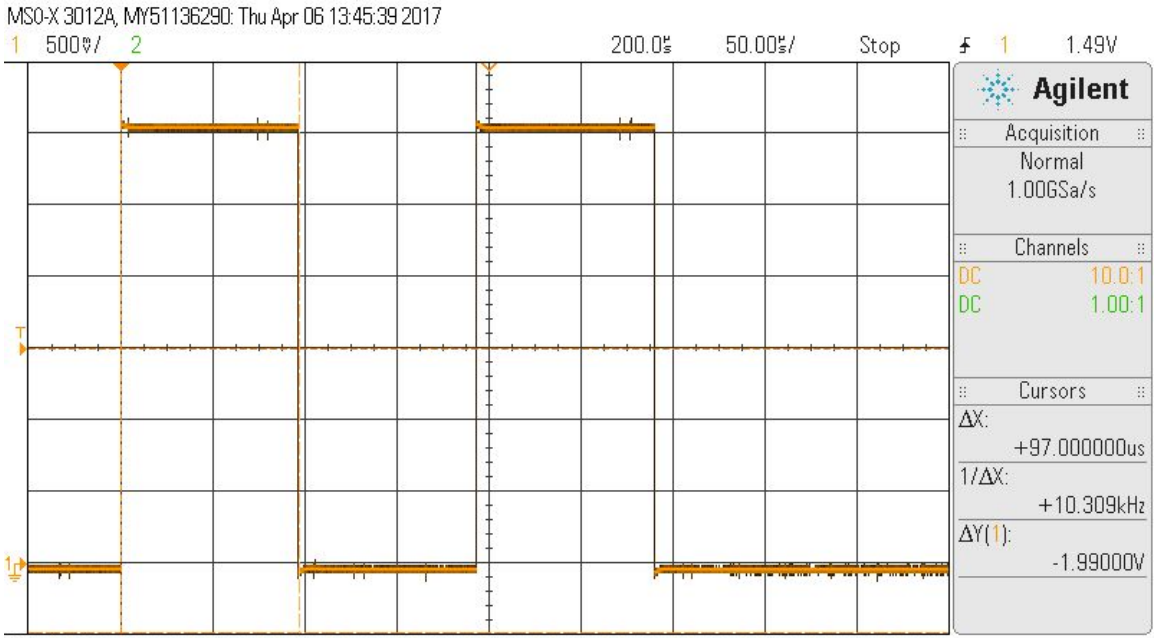

Figure 4: Two, 100 Microsecond Pulses

d. Screen shot of the shortest pulse generated. Confirm accuracy of this pulse.

Figure 5 shows the shortest pulse generated. It is supposed to be a 10.8us pulse, but it is actually a 14.72us pulse. Calculating the error we get:

$$PercentError = \frac{14.7\mu s - 10.8\mu s}{10.8\mu s} = 33\%$$

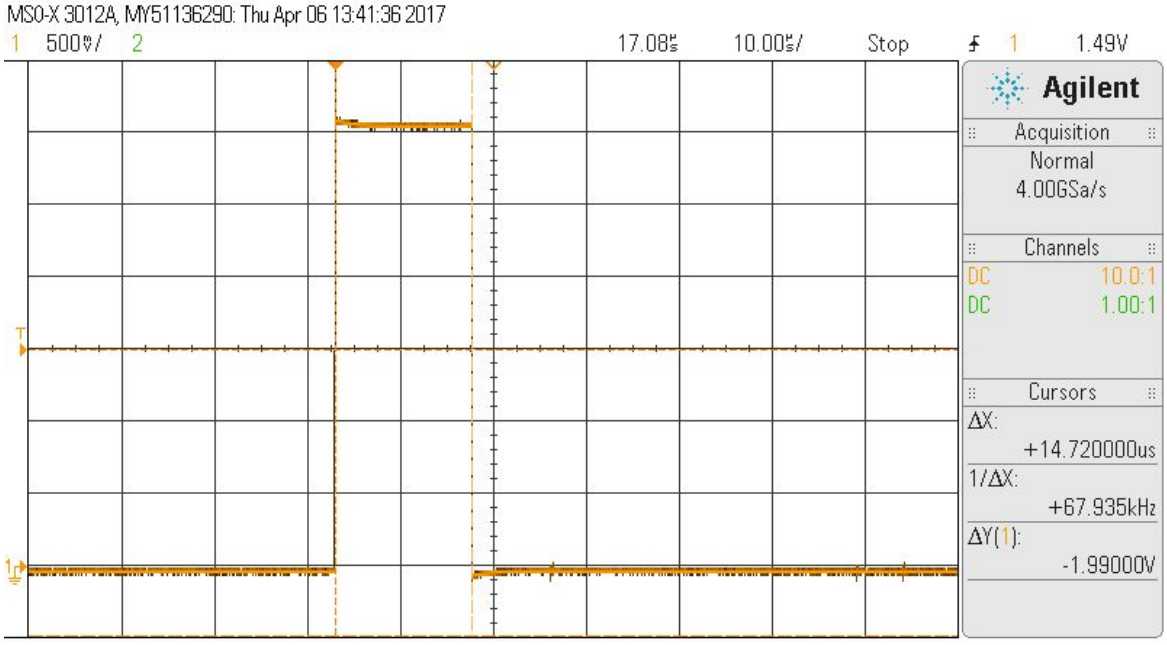

Figure 5: Shortest Pulse of 14.72 Microseconds

2. C-source code for system

Delay.h:

```
#ifndef DELAY_H
#define DELAY_H

typedef unsigned int uint;

#define FREQ_1_5_MHz CS_CTL0_DCORSEL_0
#define FREQ_3_MHz CS_CTL0_DCORSEL_1
#define FREQ_6_MHz CS_CTL0_DCORSEL_2
#define FREQ_12_MHz CS_CTL0_DCORSEL_3
#define FREQ_24_MHz CS_CTL0_DCORSEL_4
#define FREQ_48_MHz CS_CTL0_DCORSEL_5

void setDC0(uint freq){

    if(freq > CS_CTL0_DCORSEL_5){
        fprintf(stderr, "Invalid Clock");
        exit(1);
    }

    // change DC0 from default of 3MHz to 12MHz.
    CS->KEY = CS_KEY_VAL; // unlock CS registers
    CS->CTL0 = 0; // clear register CTL0
    CS->CTL0 = freq; // set DCO = 12 MHz
    // select clock sources
    CS->CTL1 = CS_CTL1_SELA_2 | CS_CTL1_SELS_3 | CS_CTL1_SELM_3;
    CS->KEY = 0; // lock the CS registers
}

/* delay milliseconds when system clock is at 3 MHz for Rev C MCU */
void delay_ms(uint delay, uint freq) {
    int i, j;
    for (j = 0; j < delay; j++)
        for (i = 100*freq; i > 0; i--);      /* Delay 1 ms*/
}

void delay_ns(const int delay, const int freq){
    int i, j;
    const int scale = delay*freq/10800;
    for (j = scale; j; j--);
}
#endif
```

Main.c:

```
#include "msp.h"
#include <stdio.h>
#include <stdlib.h>
#include "delay.h"
```

```
int main(void) {
    uint freq = 12;
    setDC0(FREQ_12_MHz);
    WDTCTL = WDTPW | WDTHOLD;           // Stop watchdog timer

    P2->SEL1 &= ~1;         /* configure P2.0 as simple I/O */
    P2->SEL0 &= ~1;
    P2->DIR |= 1;           /* P2.0 set as output pin */

    P2->SEL1 &= ~4;         /* configure P2.2 as simple I/O */
    P2->SEL0 &= ~4;
    P2->DIR |= 4;           /* P2.2 set as output pin */

    while (1) {

        setDC0(FREQ_12_MHz);
        freq = 12;
        P2->OUT |= 1;       /* turn on P2.0 red LED */
        P2->OUT |= 4;       /* turn on P2.2 blue LED */
        delay_ms(1000, freq);

        P2->OUT &= ~1;      /* turn off P2.0 red LED */
        P2->OUT &= ~4;      /* turn off P2.2 blue LED */
        delay_ms(1000, freq);

        setDC0(FREQ_6_MHz);
        freq = 6;
        P2->OUT |= 1;       /* turn on P2.0 red LED */
        P2->OUT |= 4;       /* turn on P2.2 blue LED */
        delay_ms(1000, freq);

        P2->OUT &= ~1;      /* turn off P2.0 red LED */
        P2->OUT &= ~4;      /* turn off P2.2 blue LED */
        delay_ms(1000, freq);

        setDC0(FREQ_3_MHz);
        freq = 3;
        P2->OUT |= 1;       /* turn on P2.0 red LED */
        P2->OUT |= 4;       /* turn on P2.2 blue LED */
        delay_ms(1000, freq);

        P2->OUT &= ~1;      /* turn off P2.0 red LED */
        P2->OUT &= ~4;      /* turn off P2.2 blue LED */
        delay_ms(1000, freq);
    }
}
```