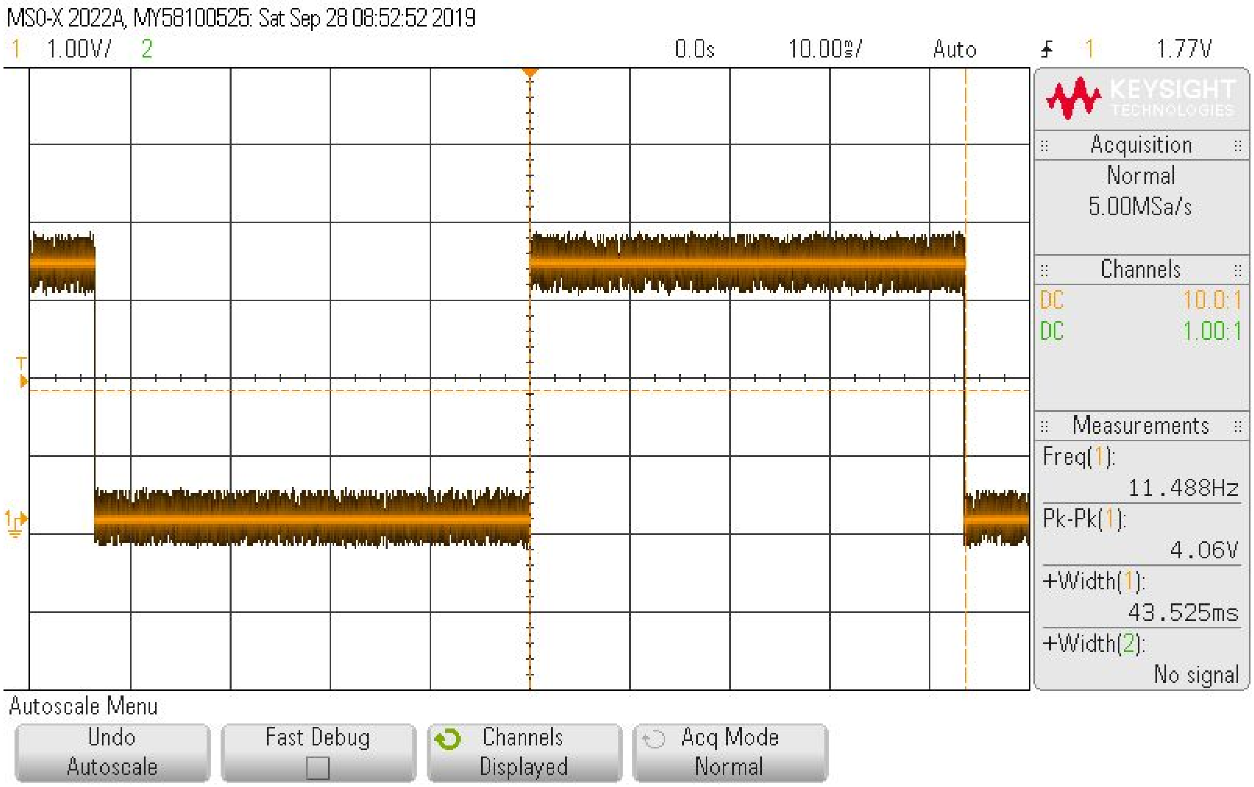
Ryan Myers
Mihir Deshmukh
Assignment 2: Blinking LED, Clock Control, and Software Delay

Deliverables

Part a: Scope captures of 40 ms pulse at each DCO frequency 1.5MHz, 6 MHz, 24 MHz, and 48 MHz



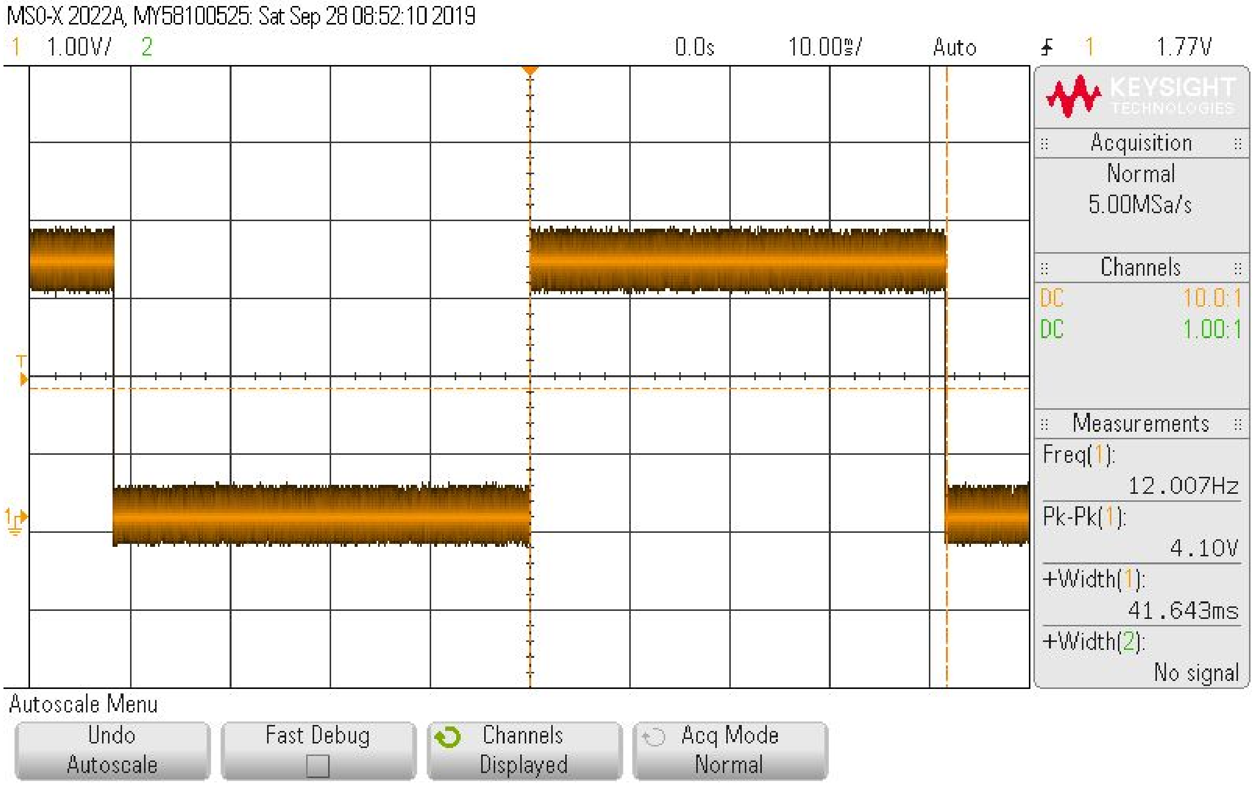
```
MSO-X 2022A, MY58100525: Sat Sep 28 08:52:52 2019
1  1.00V/  2          0.0s    10.00ms/    Auto    1    1.77V

Acquisition
Normal
5.00MSa/s

Channels
DC    10.0:1
DC    1.00:1

Measurements
Freq(1):
     11.488Hz
Pk-Pk(1):
     4.06V
+Width(1):
     43.525ms
+Width(2):
     No signal
```

Figure 1: 40ms delay (shown as logic high) with DCO clock frequency set to 1.5MHz



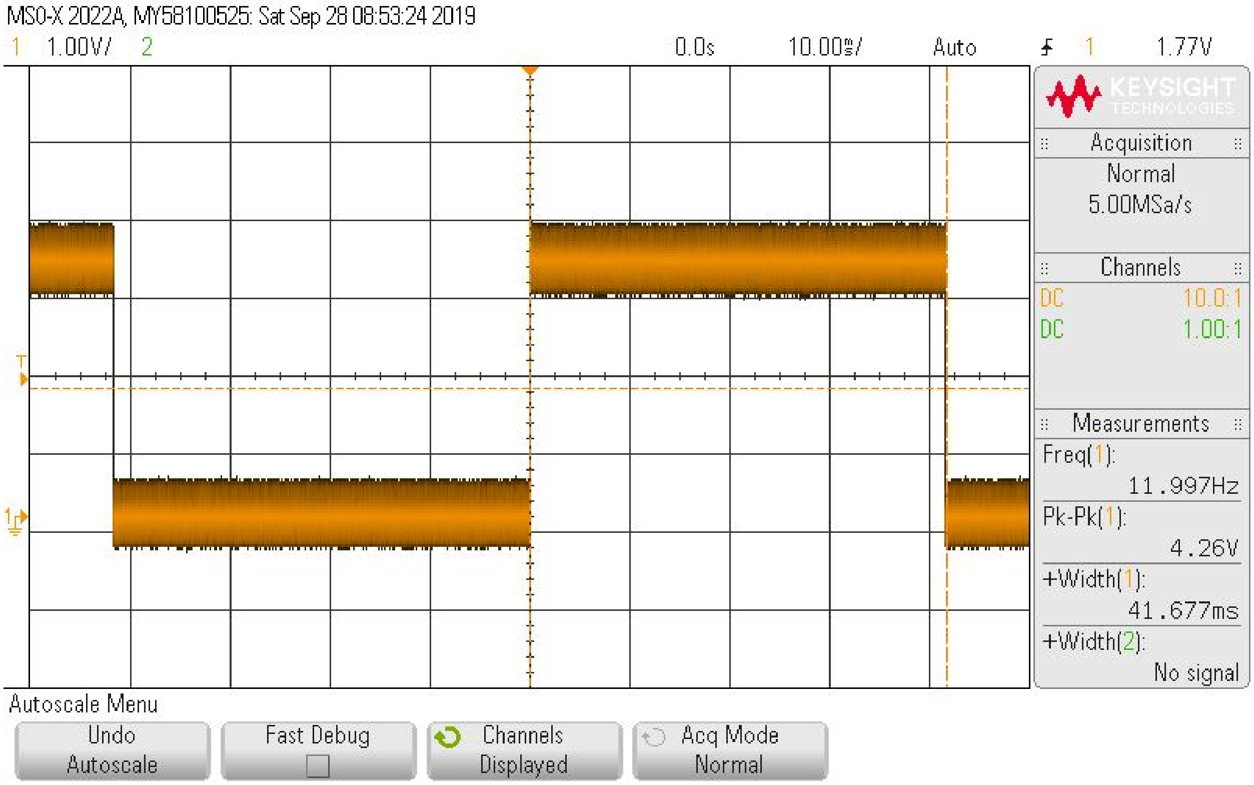
```
MSO-X 2022A, MY58100525: Sat Sep 28 08:52:10 2019
1  1.00V/  2                          0.0s    10.00ms/    Auto    1    1.77V

Acquisition
Normal
5.00MSa/s

Channels
DC          10.0:1
DC          1.00:1

Measurements
Freq(1):
        12.007Hz
Pk-Pk(1):
           4.10V
+Width(1):
        41.643ms
+Width(2):
       No signal
```

Figure 2: 40ms delay (shown as logic high) with DCO clock frequency set to 6MHz



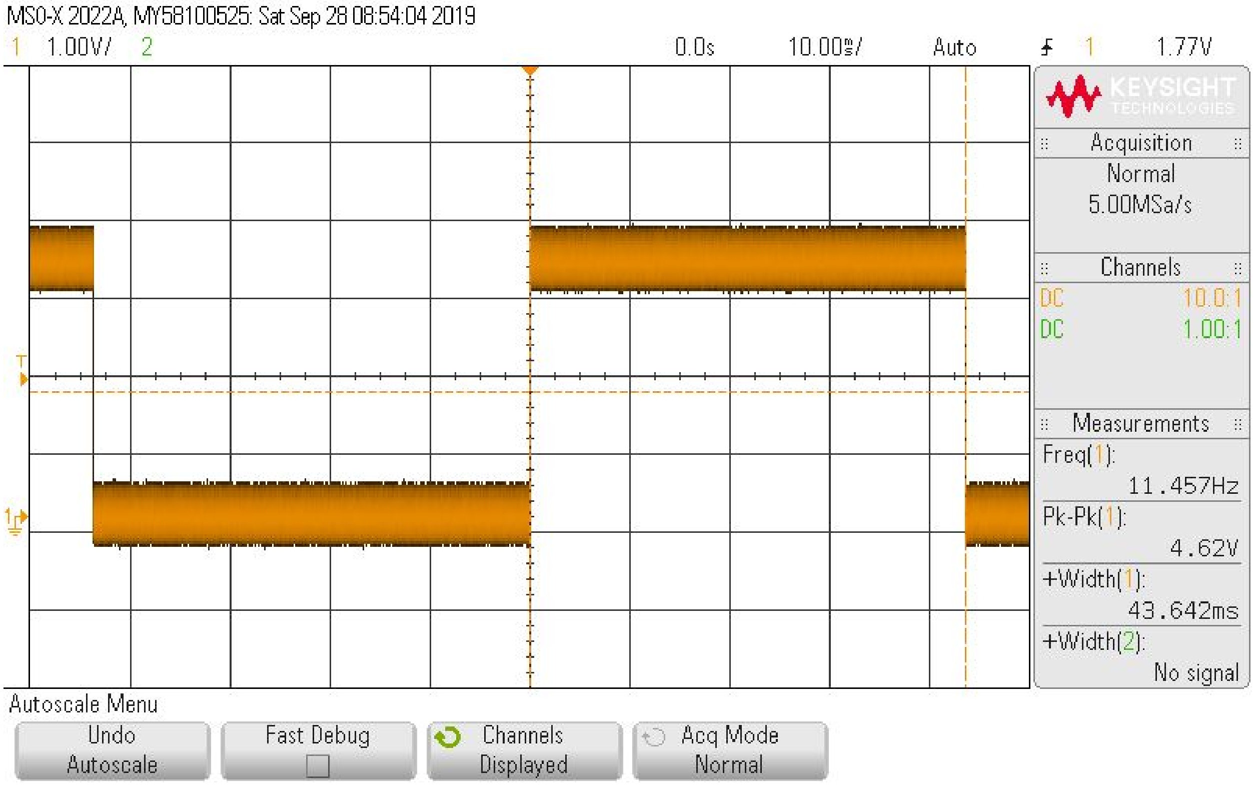
```
MSO-X 2022A, MY58100525: Sat Sep 28 08:53:24 2019
1  1.00V/  2                          0.0s    10.00ms/    Auto    1    1.77V

Acquisition
Normal
5.00MSa/s

Channels
DC          10.0:1
DC          1.00:1

Measurements
Freq(1):
        11.997Hz
Pk-Pk(1):
           4.26V
+Width(1):
        41.677ms
+Width(2):
       No signal
```

Figure 3: 40ms delay (shown as logic high) with DCO clock frequency set to 24MHz



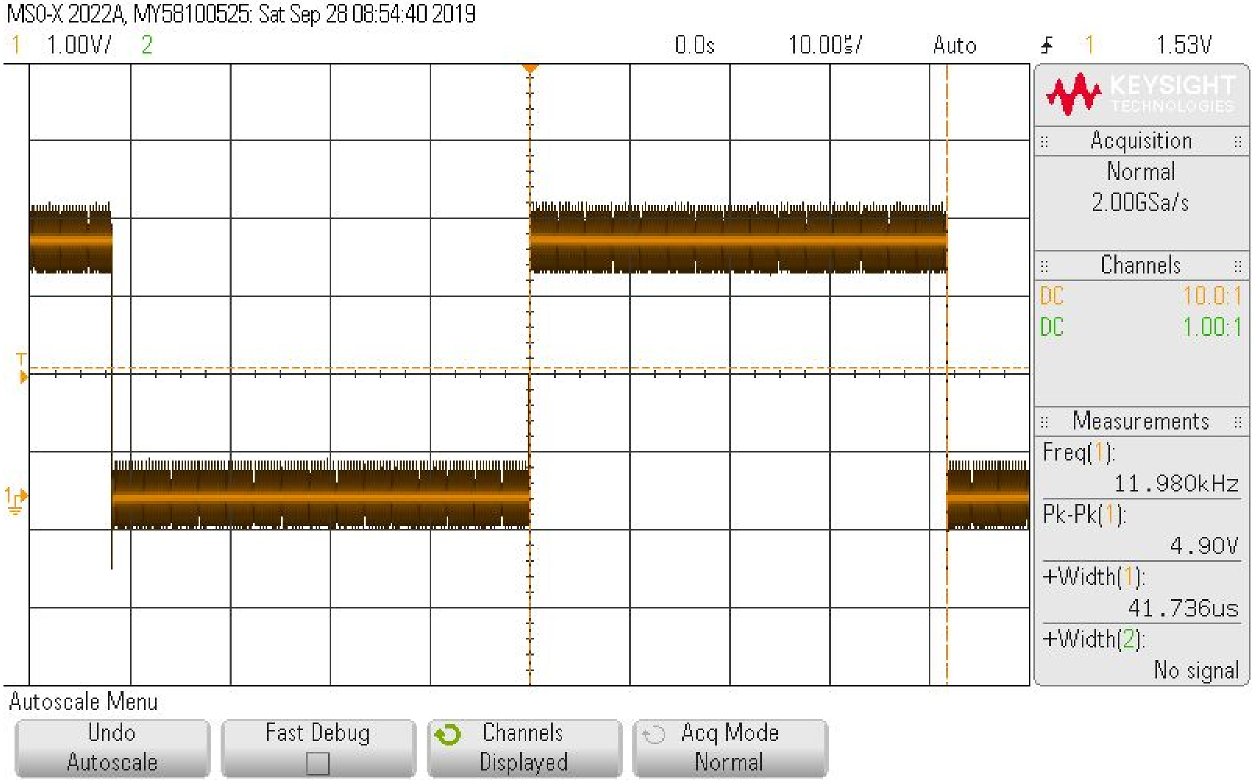
Acquisition
Normal
5.00MSa/s

Channels
DC 10.0:1
DC 1.00:1

Measurements
Freq(1): 11.457Hz
Pk-Pk(1): 4.62V
+Width(1): 43.642ms
+Width(2): No signal

Figure 4: 40ms delay (shown as logic high) with DCO clock frequency set to 48MHz

Part b: Scope captures of 40 us pulse at each DCO frequency 3 MHz, 12 MHz, and 48 MHz



MSO-X 2022A, MY58100525: Sat Sep 28 08:54:40 2019

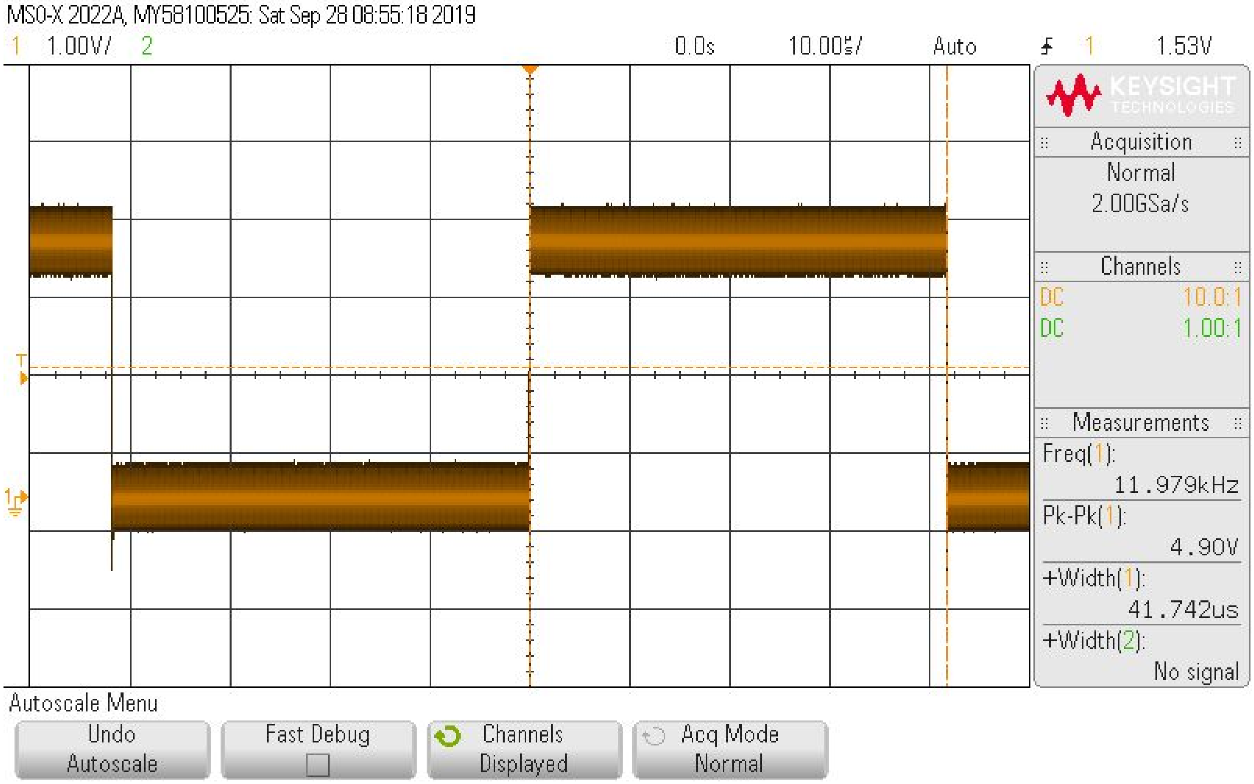
1 1.00V/ 2 0.0s 10.00µs/ Auto ƒ 1 1.53V

Acquisition
Normal
2.00GSa/s

Channels
DC 10.0:1
DC 1.00:1

Measurements
Freq(1): 11.980kHz
Pk-Pk(1): 4.90V
+Width(1): 41.736us
+Width(2): No signal

Figure 5: 40us delay (shown as logic high) with DCO clock frequency set to 3MHz



MSO-X 2022A, MY58100525: Sat Sep 28 08:55:18 2019

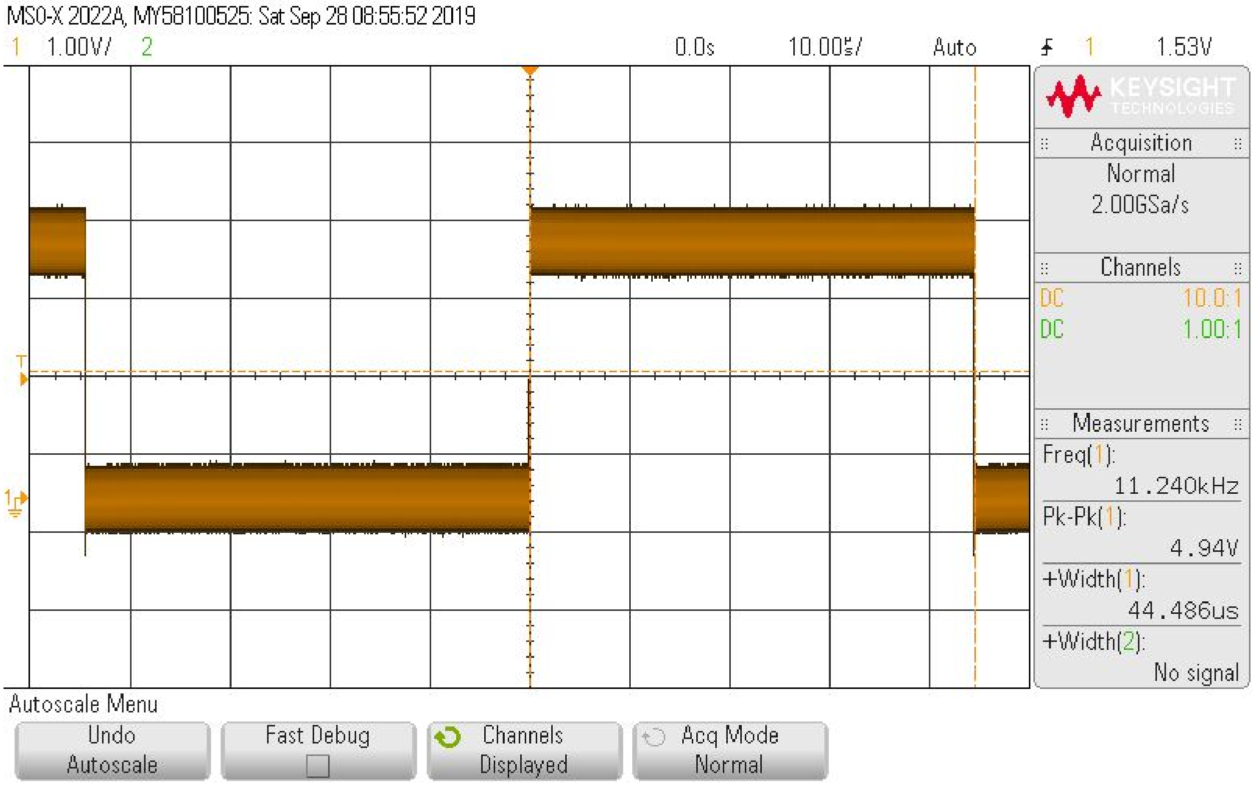
1 1.00V/ 2 0.0s 10.00µs/ Auto ƒ 1 1.53V

Acquisition
Normal
2.00GSa/s

Channels
DC 10.0:1
DC 1.00:1

Measurements
Freq(1): 11.979kHz
Pk-Pk(1): 4.90V
+Width(1): 41.742us
+Width(2): No signal

Figure 6: 40us delay (shown as logic high) with DCO clock frequency set to 12MHz



MSO-X 2022A, MY58100525: Sat Sep 28 08:55:52 2019

1 1.00V/ 2 0.0s 10.00µs/ Auto 1 1.53V

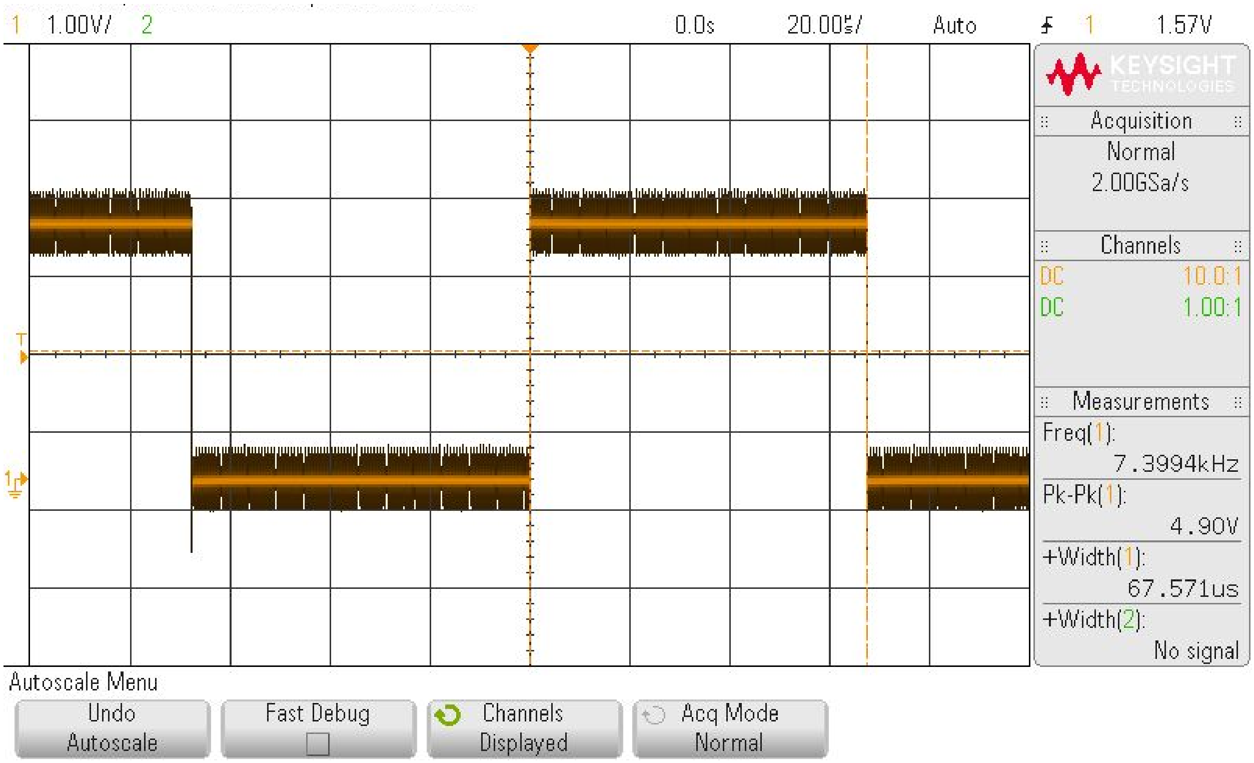
Acquisition
Normal
2.00GSa/s

Channels
DC 10.0:1
DC 1.00:1

Measurements
Freq(1): 11.240kHz
Pk-Pk(1): 4.94V
+Width(1): 44.486us
+Width(2): No signal

Figure 7: 40us delay (shown as logic high) with DCO clock frequency set to 48MHz

Part c. Scope captures of the shortest pulse generated at each DCO frequency 1.5 MHz, 3 MHz, 24 MHz, and 48 MHz



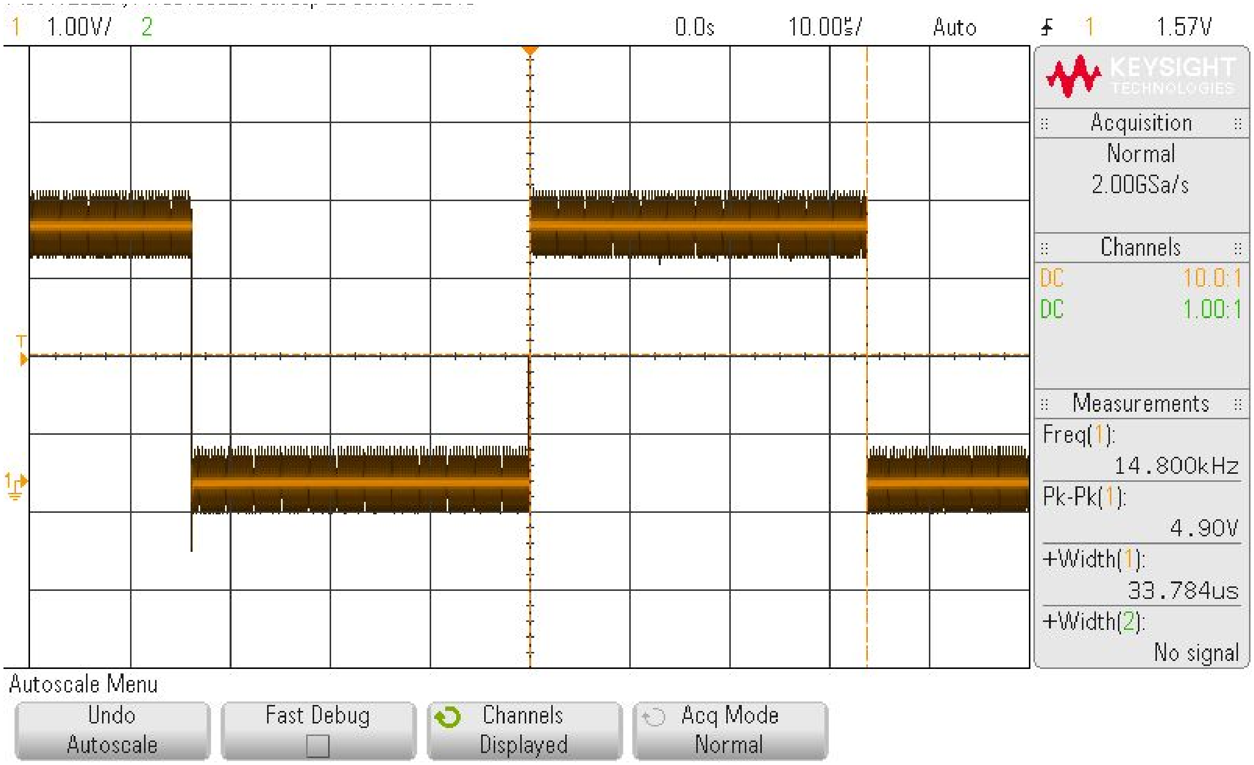
1 1.00V/ 2 0.0s 20.00us/ Auto ⨍ 1 1.57V

Acquisition
Normal
2.00GSa/s

Channels
DC 10.0:1
DC 1.00:1

Measurements
Freq(1): 7.3994kHz
Pk-Pk(1): 4.90V
+Width(1): 67.571us
+Width(2): No signal

Figure 8: Shortest pulse generated with DCO clock frequency set to 1.5MHz



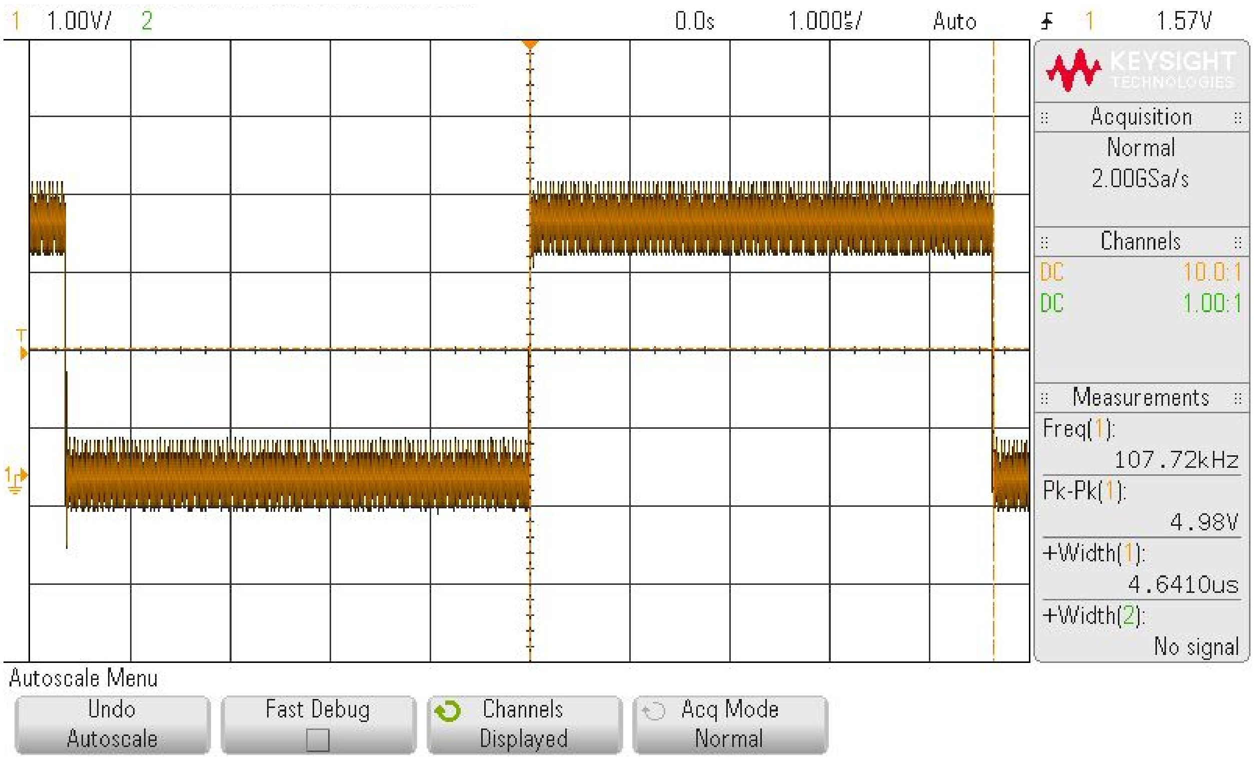
1 1.00V/ 2 0.0s 10.00us/ Auto ⨍ 1 1.57V

Acquisition
Normal
2.00GSa/s

Channels
DC 10.0:1
DC 1.00:1

Measurements
Freq(1): 14.800kHz
Pk-Pk(1): 4.90V
+Width(1): 33.784us
+Width(2): No signal

Figure 8: Shortest pulse generated with DCO clock frequency set to 3MHz



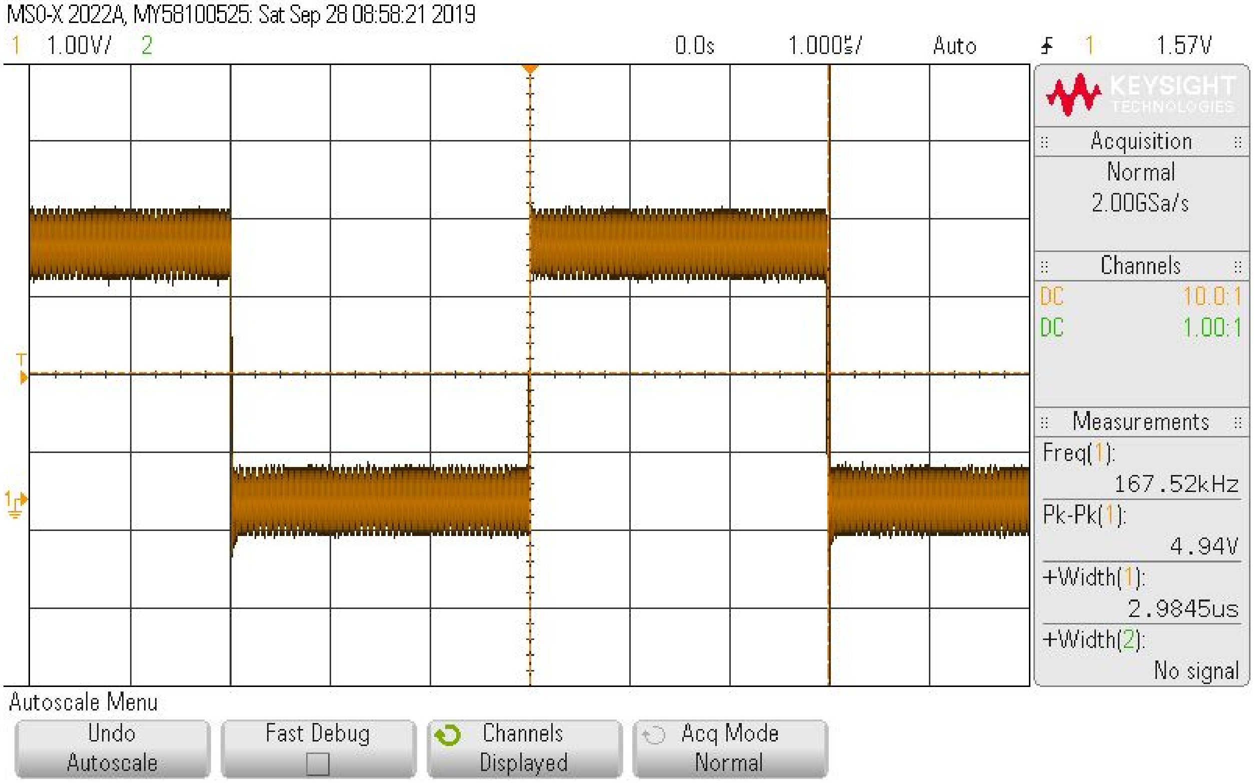
1 1.00V/ 2 0.0s 1.000µs/ Auto 1 1.57V

Acquisition
Normal
2.00GSa/s

Channels
DC 10.0:1
DC 1.00:1

Measurements
Freq(1): 107.72kHz
Pk-Pk(1): 4.98V
+Width(1): 4.6410us
+Width(2): No signal

Figure 10: Shortest pulse generated with DCO clock frequency set to 24MHz



MSO-X 2022A, MY58100525: Sat Sep 28 08:58:21 2019

1 1.00V/ 2 0.0s 1.000µs/ Auto 1 1.57V

Acquisition
Normal
2.00GSa/s

Channels
DC 10.0:1
DC 1.00:1

Measurements
Freq(1): 167.52kHz
Pk-Pk(1): 4.94V
+Width(1): 2.9845us
+Width(2): No signal

Figure 11: Shortest pulse generated with DCO clock frequency set to 48MHz

Part d. Project code formatted properly

Used https://emn178.github.io/online-tools/syntax_highlight.html for formatting

main.c

```
#include "msp.h"
#include "delay.h"

/*
 * main.c
 * Author: Mihir Deshmukh, Ryan Myers
 */
void main(void)
{
      WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD;      // stop watchdog
timer
      set_DCO(DCORSEL_12_MHz);   // Set DCO frequency to 1.5MHz
      P4->DIR |= BIT3;    // Set P4.3 to output
      P4->REN &= ~BIT3;   // Disable P4.3 pull up/down resistor
      P4->SEL0 |= BIT3;   // Select MCLK as source of output
      P4->SEL1 &= ~BIT3;
      P4->DIR |= BIT1;    // Set P4.1 to output
      P4->REN &= ~BIT1;   // Disable P4.1 pull up/down resistor
      P4->SEL0 &= ~BIT1;  // Setup P4.1 as GPIO
      P4->SEL1 &= ~BIT1;
      // loop infinitely to time delay function
      while (1) {
          P4->OUT ^= BIT1;    // Toggle bit to show beginning and end
of delay
          delay_us(40);
      }
}
```

delay.c

```
/*
 * delay.c
 *
 *  Created on: Sep 26, 2019
 *      Author: Ryan Myers, Mihir Deshmukh
 */

#include "delay.h"
#include "msp.h"

void set_DCO(uint32_t MHz_freq) {

    // Select the correct power mode if we are operating at 48MHz
    if (MHz_freq == DCORSEL_48_MHz) {
        /* Transition to VCORE Level 1: AM0_LDO --> AM1_LDO */
        while ((PCM->CTL1 & PCM_CTL1_PMR_BUSY));
            PCM->CTL0 = PCM_CTL0_KEY_VAL | PCM_CTL0_AMR_1;
        while ((PCM->CTL1 & PCM_CTL1_PMR_BUSY));

        /* Configure Flash wait-state to 1 for both banks 0 & 1 */
        FLCTL->BANK0_RDCTL = (FLCTL->BANK0_RDCTL &
                ~(FLCTL_BANK0_RDCTL_WAIT_MASK)) |
FLCTL_BANK0_RDCTL_WAIT_1;
        FLCTL->BANK1_RDCTL = (FLCTL->BANK0_RDCTL &
                ~(FLCTL_BANK1_RDCTL_WAIT_MASK)) |
FLCTL_BANK1_RDCTL_WAIT_1;
    }

    CS->KEY = CS_KEY_VAL;   // Unlock clock registers
    CS->CTL0 &= ~CS_CTL0_DCOTUNE_MASK;  // Clear DCO tune and select
registers
    CS->CTL0 &= ~CS_CTL0_DCORSEL_MASK;
    CS->CTL0 |= CS_CTL0_DCORSEL_MASK & MHz_freq;    // Set select to
given nominal frequency
    CS->KEY = 0;    // Lock clock registers
    return;
}
```

```
// Delay calculations are scaled based on function enter and exit
times as well as
// while loop iteration times for a 1.5MHz operating frequency
void delay_us(int us_delay) {
    uint32_t dco_freq;
    uint32_t scale;
    int i;
    // Determine currently configured DCO RSEL and TUNE values
    uint32_t dco_rsel = CS->CTL0 & CS_CTL0_DCORSEL_MASK;
    int dco_tune = (CS->CTL0 & CS_CTL0_DCOTUNE_MASK) >>
CS_CTL0_DCOTUNE_OFS;
    // if DCOTUNE is negative, expand it to a signed 32 bit number
    if (dco_tune & DCOTUNE_SIGN_MASK) {
        dco_tune = -1 * (~dco_tune + 1);
    }
    // Add DCOTUNE value to nominal frequency to find current DC0
    // operating frequency
    switch (dco_rsel) {
    case DCORSEL_1POINT5_MHz:
        dco_freq = FREQ_1POINT5_MHz + dco_tune;
        break;
    case DCORSEL_3_MHz:
        dco_freq = FREQ_3_MHz + dco_tune;
        break;
    case DCORSEL_6_MHz:
        dco_freq = FREQ_6_MHz + dco_tune;
        break;
    case DCORSEL_12_MHz:
        dco_freq = FREQ_12_MHz + dco_tune;
        break;
    case DCORSEL_24_MHz:
        dco_freq = FREQ_24_MHz + dco_tune;
        break;
    case DCORSEL_48_MHz:
        dco_freq = FREQ_48_MHz + dco_tune;
        break;
    }
// Calculate timing scaler based on 1.5MHz timing measurements
    scale = dco_freq/FREQ_1POINT5_MHz;
    i = FUNC_ENTER_EXIT_TIME / scale;
    // Timing values are scaled by 1000 to provide greater precision
    // without the use of floating point numbers
    while(i < (us_delay * 1000))
        i += LOOP_ITER_TIME / scale;
}
```

delay.h

```c
/*
 * delay.h
 *
 *  Created on: Sep 26, 2019
 *      Author: Ryan Myers, Mihir Deshmukh
 */

#ifndef DELAY_H_
#define DELAY_H_

#include <stdint.h>

// DCORSEL definitions used to pass into set_DCO function
#define DCORSEL_1POINT5_MHz CS_CTL0_DCORSEL_0
#define DCORSEL_3_MHz CS_CTL0_DCORSEL_1
#define DCORSEL_6_MHz CS_CTL0_DCORSEL_2
#define DCORSEL_12_MHz CS_CTL0_DCORSEL_3
#define DCORSEL_24_MHz CS_CTL0_DCORSEL_4
#define DCORSEL_48_MHz CS_CTL0_DCORSEL_5
// FREQ definitions used for delay calculations
#define FREQ_1POINT5_MHz 1500000u
#define FREQ_3_MHz 3000000u
#define FREQ_6_MHz 6000000u
#define FREQ_12_MHz 12000000u
#define FREQ_24_MHz 24000000u
#define FREQ_48_MHz 48000000u
// Used to check DCOTUNE sign
#define DCOTUNE_SIGN_MASK 0x200
// Measured while loop iteration time at 1.5MHz
#define LOOP_ITER_TIME 14000u
// Measured function enter/exit time at 1.5MHz
#define FUNC_ENTER_EXIT_TIME 72000u


void set_DCO(uint32_t MHz_freq);
void delay_us(int us_delay);

#endif /* DELAY_H_ */
```