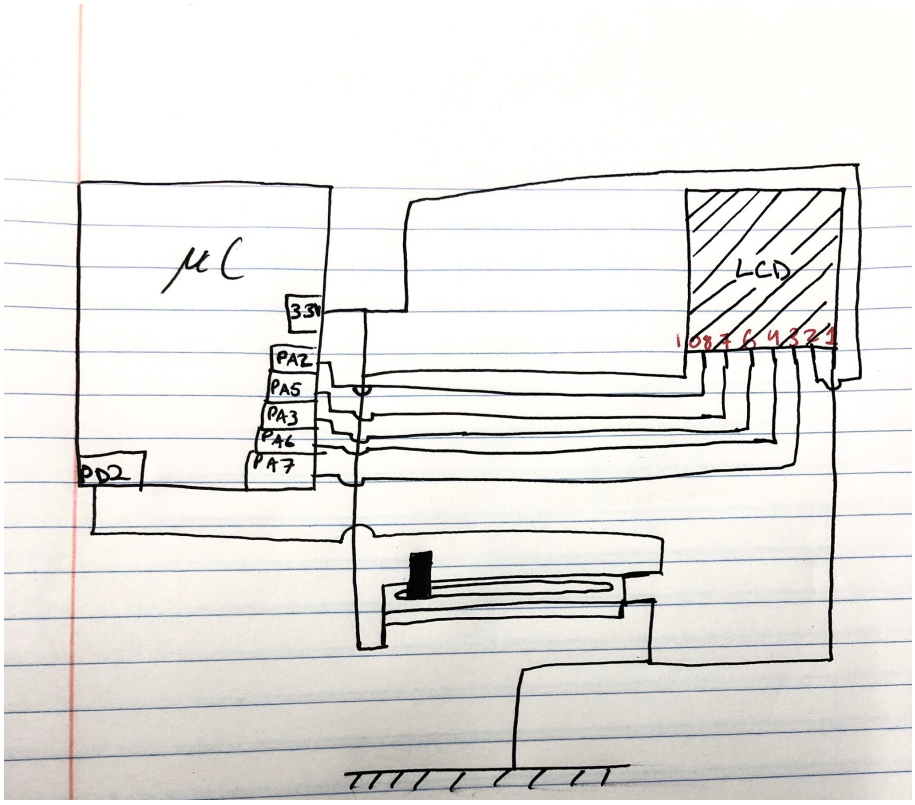
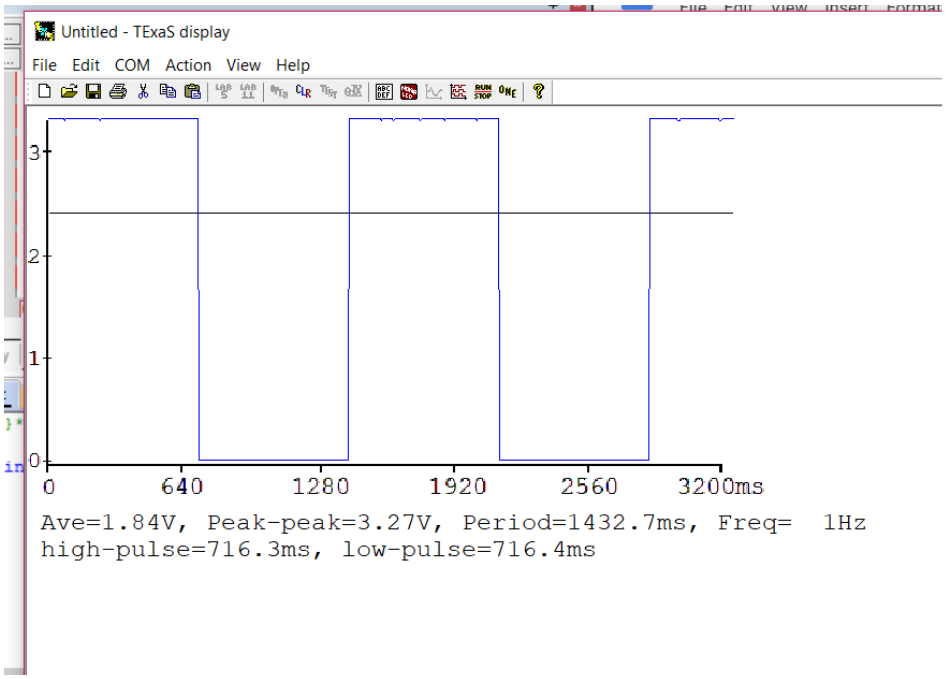


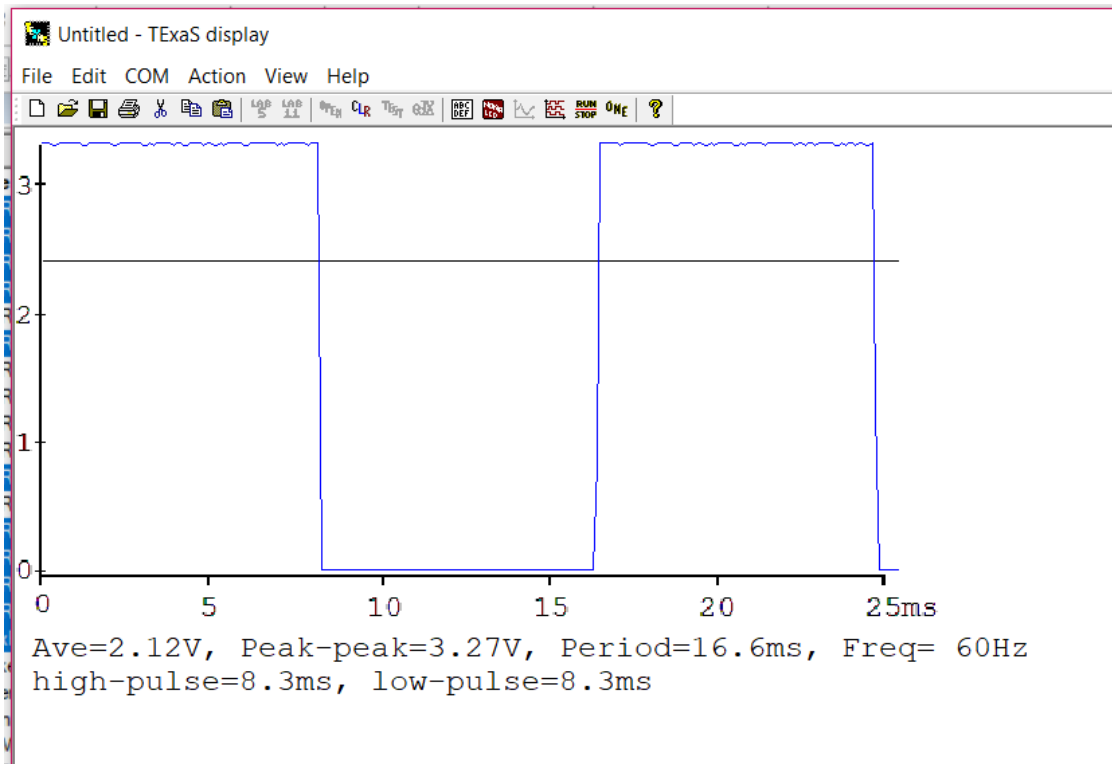


Time to execute 100 times is 1432.7ms
Time to execute once 14.32ms

Calibration Data

-

.10	3760	100
.40	3746	400
.80	3839	800
1.20	3997	1200
1.40	4061	1400



True position x_{ti}	Measured Position x_{mi}	Error $x_{ti} - x_{mi}$
0	0	0
.5	.473	.024

1	1.15	-.05
1.5	1.58	-.08
2	2	2

Average Error: .051

```c
// ADC.c
// Runs on LM4F120/TM4C123
// Provide functions that initialize ADC0
// Last Modified: 11/6/2018
// Student names: change this to your names or look very silly
// Last modification date: change this to the last modification date or look very silly

#include <stdint.h>
#include "../inc/tm4c123gh6pm.h"

// ADC initialization function
// Input: none
// Output: none
// measures from PD2, analog channel 5
void ADC_Init(void){
  SYSCTL_RCGCGPIO_R |= 0x08;     //turn on clock for Port D
  uint8_t k;                     //wait for clock to start
  k = 42;
  GPIO_PORTD_DIR_R &= ~0x04;     //make PD2 input
  GPIO_PORTD_AFSEL_R |= 0x04;    //enable alternate function on PD2
  GPIO_PORTD_DEN_R &= ~0x04;     //disable digital I/O on PD2
  GPIO_PORTD_AMSEL_R |= 0x04;    //enable analog functionality on PD2
  SYSCTL_RCGCADC_R |= 0x0001;    //activate ADC0
  k = 42;
  k = 42;
  k = 42;
  k = 42;
  k = 42;
  k = 42;
  k = 42;
  k = 42;
  k = 42;
  k = 42;
  k = 42;
  k = 42;
  ADC0_PC_R = 0x01;              //configure for 125K
  ADC0_SSPRI_R = 0x0123;         //Seq 3 is highest priority
  ADC0_ACTSS_R &= ~0x0008;       //disable sample sequencer 3
  ADC0_EMUX_R &= ~0xF000;      //seq3 is software trigger
  ADC0_SSMUX3_R = (ADC0_SSMUX3_R&0xFFFFFFF0)+5;  //Ain5 (PD2)
  ADC0_SSCTL3_R = 0x0006;        //no TS0 D0, yes IE0 END0
  ADC0_IM_R &= ~0x0008;          //disable SS3 interrupts
  ADC0_ACTSS_R |= 0x0008;        //enable sample sequencer 3
//  ADC0_SAC_R = 0x0A;

}

//------------ADC_In------------
// Busy-wait Analog to digital conversion
// Input: none
// Output: 12-bit result of ADC conversion
// measures from PD2, analog channel 5
uint32_t ADC_In(void){
  uint32_t data;
  ADC0_PSSI_R = 0x0008;
  while((ADC0_RIS_R&0x08)==0){};
  data = ADC0_SSFIFO3_R&0xFFF;
  ADC0_ISC_R = 0x0008;
  return data;
}
```

```c
// Lab8.c
// Runs on LM4F120 or TM4C123
// Student names: change this to your names or look very silly
// Last modification date: change this to the last modification date or look very silly
// Last Modified: 11/6/2018

// Specifications:
// Measure distance using slide pot, sample at 60 Hz
// maximum distance can be any value from 1.5 to 2cm
// minimum distance is 0 cm
// Calculate distance in fixed point, 0.001cm
// Analog Input connected to PD2=ADC5
// displays distance on Sitronox ST7735
// PF3, PF2, PF1 are heartbeats (use them in creative ways)
//

#include <stdint.h>

#include "ST7735.h"
#include "TExaS.h"
#include "ADC.h"
#include "print.h"
#include "../inc/tm4c123gh6pm.h"

//*****the first three main programs are for debugging *****
// main1 tests just the ADC and slide pot, use debugger to see data
// main2 adds the LCD to the ADC and slide pot, ADC data is on ST7735
// main3 adds your convert function, position data is no ST7735

void DisableInterrupts(void); // Disable interrupts
void EnableInterrupts(void);  // Enable interrupts

#define PF1       (*((volatile uint32_t *)0x40025008))
#define PF2       (*((volatile uint32_t *)0x40025010))
#define PF3       (*((volatile uint32_t *)0x40025020))
  uint32_t ADCMail;
  uint8_t ADCMail_Flag = 0;
// Initialize Port F so PF1, PF2 and PF3 are heartbeats
void PortF_Init(void){
  SYSCTL_RCGCGPIO_R |= 0x20;        //turn on clock for Port F
  uint8_t k = 0;                    //wait for clock to start
  k = 42;
  GPIO_PORTF_DIR_R |= 0x0E;         //PF1,2,3 are outputs
  GPIO_PORTF_DEN_R |= 0x0E;         //enable digital I/O for PF1,2,3
}

// Initialize SysTick with busy wait running at bus clock.
void SysTick_Init(void){
  NVIC_ST_CTRL_R = 0;
  NVIC_ST_RELOAD_R = 0xA2038;
  NVIC_ST_CURRENT_R = 0;
  NVIC_SYS_PRI3_R = (NVIC_SYS_PRI3_R & 0x00FFFFFF) | 0x40000000; //priority 2
  NVIC_ST_CTRL_R = 0x07;
}

void SysTick_Handler(void){
  PF1 ^= 0x02;
  PF2 ^= 0x04;
  ADCMail = ADC_In();
  ADCMail_Flag = 1;
  //PF3 ^= 0x08;
}
uint32_t Data;        // 12-bit ADC
uint32_t Position;    // 32-bit fixed-point 0.001 cm
/*int main(void){       // single step this program and look at Data
  TExaS_Init();       // Bus clock is 80 MHz
  ADC_Init();         // turn on ADC, set channel to 5
  while(1){
    Data = ADC_In();  // sample 12-bit channel 5
  }
}*/

```

```
 73  /*int main(void){
 74    uint32_t count = 0;
 75    TExaS_Init();       // Bus clock is 80 MHz
 76    ADC_Init();         // turn on ADC, set channel to 5
 77    ST7735_InitR(INITR_REDTAB);
 78    PortF_Init();
 79    while(1){           // use scope to measure execution time for ADC_In and LCD_OutDec
 80      if (count >= 100) {
 81      count = 0;
 82      PF2 ^= 0x04;       // Profile ADC
 83      }
 84      Data = ADC_In();  // sample 12-bit channel 5
 85      //PF2 = 0x00;       // end of ADC Profile
 86      ST7735_SetCursor(0,0);
 87      PF1 = 0x02;       // Profile LCD
 88      LCD_OutDec(Data);
 89      ST7735_OutString("    ");  // spaces cover up characters from last output
 90      PF1 = 0;          // end of LCD Profile
 91      count++;
 92    }
 93  }*/
 94
 95  // your function to convert ADC sample to distance (0.001cm)
 96  uint32_t Convert(uint32_t input){
 97    //input = ((input*5.479452055)-20438.35616);
 98    if (input == 4095) return 2000;
 99    input = ((input*4.471787401)-16554.70338);
100    if (input <= 230) return 0;
101    return input;
102  }
103  /*int main3(void){
104    TExaS_Init();         // Bus clock is 80 MHz
105    ST7735_InitR(INITR_REDTAB);
106    PortF_Init();
107    ADC_Init();         // turn on ADC, set channel to 5
108    while(1){
109      PF2 ^= 0x04;      // Heartbeat
110      Data = ADC_In();  // sample 12-bit channel 5
111      PF3 = 0x08;       // Profile Convert
112      Position = Convert(Data);
113      PF3 = 0;          // end of Convert Profile
114      PF1 = 0x02;       // Profile LCD
115      ST7735_SetCursor(0,0);
116      LCD_OutDec(Data); ST7735_OutString("    ");
117      ST7735_SetCursor(6,0);
118      LCD_OutFix(Position);
119      PF1 = 0;          // end of LCD Profile
120    }
121  }   */
122  int main(void){
123    TExaS_Init();
124    ST7735_InitR(INITR_REDTAB);
125    PortF_Init();
126    ADC_Init();
127    SysTick_Init();
128    EnableInterrupts();
129    while(1){
130      PF2 ^= 0x04;
131      //PF3 = 0x08;
132
133      if (ADCMail_Flag == 1){
134        ST7735_SetCursor(0,0);
135        PF1 = 0x02;
136        LCD_OutFix(Convert(ADCMail));
137        //PF3 = 0;
138        PF1 = 0x02;
139        ST7735_OutString("    ");
140        ADCMail_Flag = 0;
141        PF1 = 0;
142      }
143      else {
144        for(uint32_t k = 0; k < 1000000; k++){
```

```
145           PF1 = 0;
146           PF2 = 0;
147           //PF3 = 0;
148         }
149       }
150       }
151     }
152
153
```