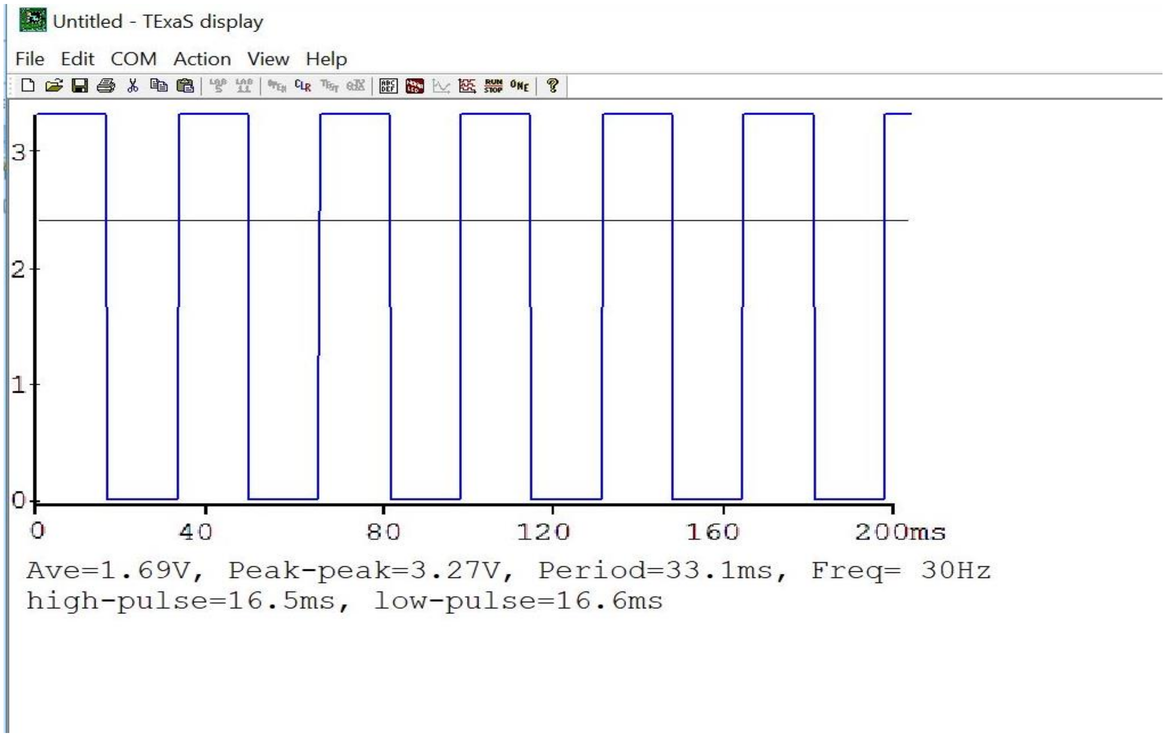


```
Ave=1.69V, Peak-peak=3.27V, Period=33.1ms, Freq= 30Hz
high-pulse=16.5ms, low-pulse=16.6ms
```

33.1ms in between interrupts

Average accuracy (with units in cm) = -0.0258cm

True position x_{ti}	Measured Position x_{mi}	Error $x_{ti} - x_{mi}$
0.1cm	0.132	-0.032
0.4cm	0.402	-0.002
0.8cm	0.795	0.005
1.2cm	1.271	-0.071
1.6cm	1.629	-0.029

	Analog	ADC
Distance	Input	Sample
100	0	18
400	0.44	750
800	1.42	1750
1200	2.36	3080
1400	2.84	3620
1700	3.27	4093

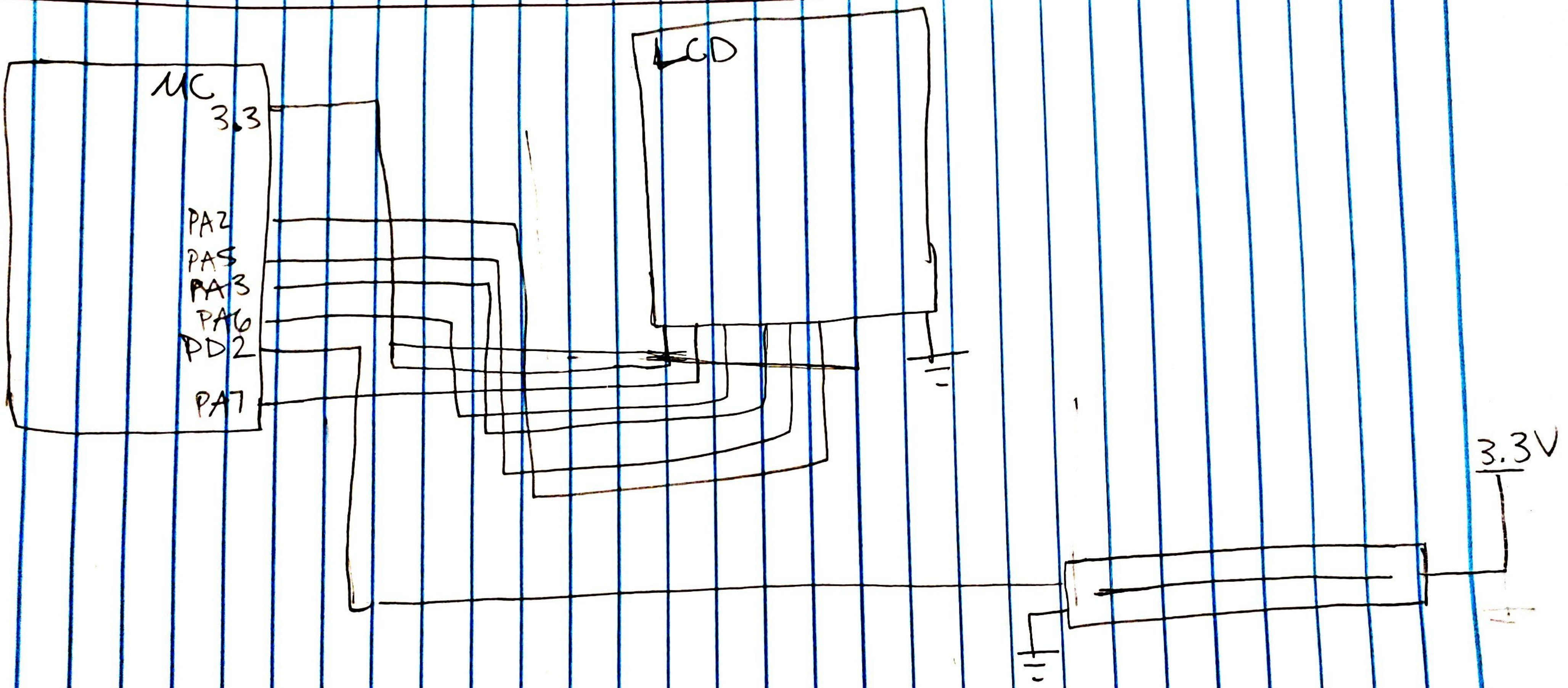
```
// ADC.c

// Runs on LM4F120/TM4C123
// Provide functions that initialize ADC0
// Last Modified: 11/6/2018
// Student names: Meha Halabe & Tejal Kulkarni
// 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){
    volatile int delay;
    SYSCTL_RCGCGPIO_R |= 0X08;
            //activate clock for port d

    while((SYSCTL_PRGPIO_R&0X08)==0){};
    GPIO_PORTD_DIR_R &= ~0X04;
            //make pd2 an input
    GPIO_PORTD_AFSEL_R |= 0X04;
            //enable alt func on pd2
    GPIO_PORTD_DEN_R &= ~0X04;
                //disable digital i/o
    GPIO_PORTD_AMSEL_R |= 0X04;
                //enable analog func
    SYSCTL_RCGCADC_R |=0X01;
                //activate ADC0
    delay=SYSCTL_RCGCADC_R;
                    //stablizie
    delay=SYSCTL_RCGCADC_R;
    delay=SYSCTL_RCGCADC_R;
    delay=SYSCTL_RCGCADC_R;
    ADC0_PC_R=0X01;
                                //configure for 125k sampling rate
```

```
      ADC0_SSPRI_R=0X123;
                                        // seq3 is highest priority
      ADC0_ACTSS_R &= ~0X0008;
                          //disable sample sequencer 3
      ADC0_EMUX_R &= ~0XF000;
                                 // sq3 is softare trigger
      ADC0_SSMUX3_R = (ADC0_SSMUX3_R&0XFFFFFFF0)+5;              //AIN5
      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


}

//------------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;
            GPIO_PORTF_DATA_R ^= 0X08;
      ADC0_PSSI_R = 0X0008;
                                        //start sample capture

      while((ADC0_RIS_R&0X08)==0){};
      data = ADC0_SSFIFO3_R&0XFFF;
                          //read sample and set data
      ADC0_SAC_R = 0x06;
      //2^{n-1} hardware averaging
      ADC0_ISC_R = 0X0008;
                                        //clears flag
      return data;
}


//Lab8.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))
// Initialize Port F so PF1, PF2 and PF3 are heartbeats
void PortF_Init(void){

        volatile int delay;
        SYSCTL_RCGCGPIO_R |=0X20;
        //turn on clock
        delay +=123456;

    GPIO_PORTF_DIR_R |=0X0E;
        //initialize pf4 as input, pf2 as output
    GPIO_PORTF_DEN_R |= 0X0E;
    GPIO_PORTF_LOCK_R = GPIO_LOCK_KEY;
    //unlock port f
    GPIO_PORTF_PUR_R |=0X10;
        //pull up resistor for negative logic
    GPIO_PORTF_CR_R |= 0XFFFFFFFF;

}
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
  }
}

int main(void){
  TExaS_Init();       // Bus clock is 80 MHz
  ADC_Init();         // turn on ADC, set channel to 5
  ST7735_InitR(INITR_REDTAB);
  PortF_Init();
  while(1){           // use scope to measure execution time for ADC_In and LCD_OutDec
    PF2 = 0x04;       // Profile ADC
    Data = ADC_In();  // sample 12-bit channel 5
    PF2 = 0x00;       // end of ADC Profile
    ST7735_SetCursor(0,0);
    PF1 = 0x02;       // Profile LCD
    LCD_OutDec(Data);
    ST7735_OutString("    ");  // spaces cover up characters from last output
    PF1 = 0;          // end of LCD Profile
  }
}*/

// your function to convert ADC sample to distance (0.001cm)
uint32_t Convert(uint32_t input){
  uint32_t pos;

              pos= (372*input)/1000;
              pos = pos +108;
              return pos;
}
/*int main (void){
  TExaS_Init();         // Bus clock is 80 MHz
  ST7735_InitR(INITR_REDTAB);
  PortF_Init();
  ADC_Init();         // turn on ADC, set channel to 5
  while(1){
    PF2 ^= 0x04;      // Heartbeat
    Data = ADC_In();  // sample 12-bit channel 5
    PF3 = 0x08;       // Profile Convert
    Position = Convert(Data);
    PF3 = 0;          // end of Convert Profile
    PF1 = 0x02;       // Profile LCD
    ST7735_SetCursor(0,0);
    LCD_OutDec(Data); ST7735_OutString("    ");
    ST7735_SetCursor(6,0);
```

```
   LCD_OutFix(Position);
   PF1 = 0;         // end of LCD Profile
  }
}  */
uint32_t ADCMail;
int32_t ADCStatus=0;

void SysTick_Init(void){
            // write this
            NVIC_ST_CTRL_R &=0;
                                                                              //Disable SysTick

            NVIC_ST_RELOAD_R|=0X00145855;
                                    //Set reload value to 60Hz (1/60)/12.5ns

            NVIC_ST_CURRENT_R&=0;


            NVIC_ST_CTRL_R|=0X00000007;
                                                //Enable interrupts and clock
}

void SysTick_Handler(){
            ADCMail=ADC_In();
                                                                                          //Set mail
            ADCStatus=-1;
                                                                                                      //Notify that status has changed
                        PF2^= 0x04;
                                                                                                                  //Toggle heartbeat
}

int main(void){
  TExaS_Init();
            EnableInterrupts();
            ST7735_InitR(INITR_REDTAB);
            PortF_Init();
  ADC_Init();
                                                                                                      // turn on ADC, set channel to 5

            SysTick_Init();

  while(1){
  // your Lab 8
                        while(ADCStatus<0)
                        {
                                    ST7735_SetCursor(0,0);
                                    LCD_OutFix(Convert(ADCMail));
                        //print out position
            ADCStatus=0;
                                                  //change status
            ST7735_OutString(" cm");
        }

 }

}
```



uC
3.3
PA2
PA5
PA3
PA6
PD2
PA7
LCD
3.3V