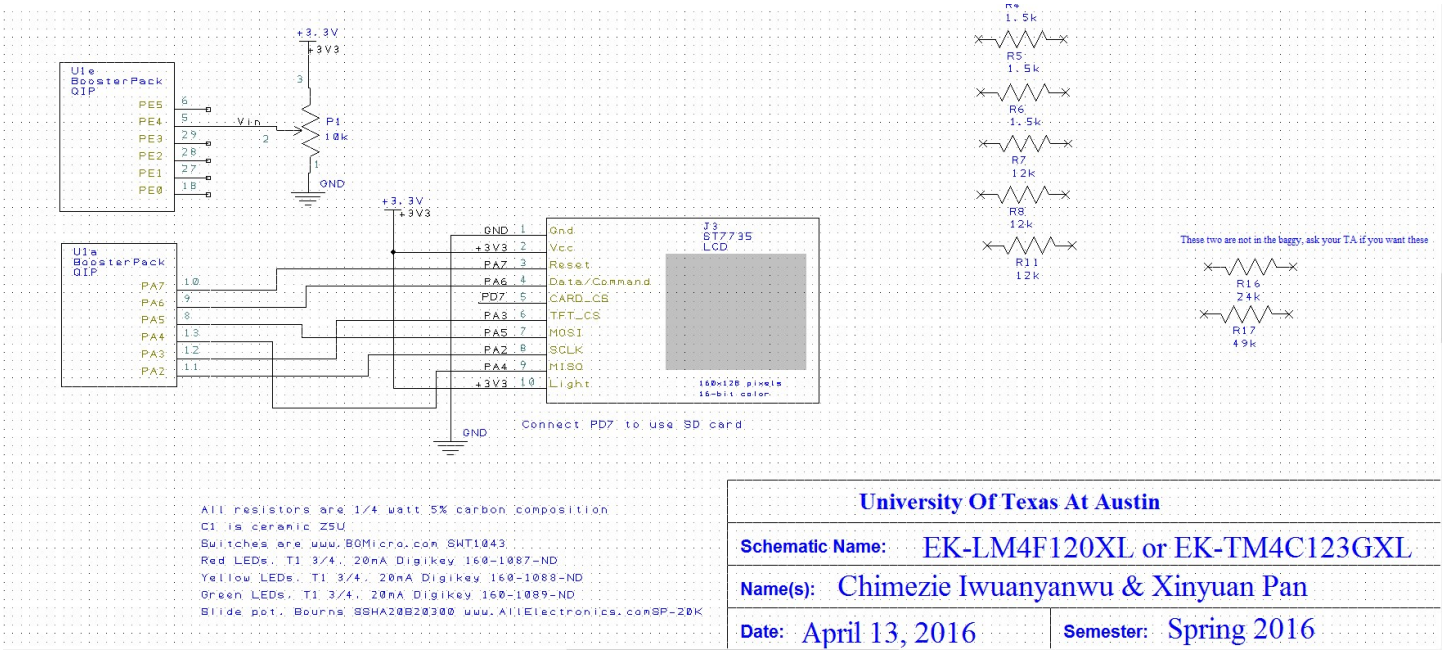
Chimezie Iwuanyanwu cci273
Xinyuan Pan xp572

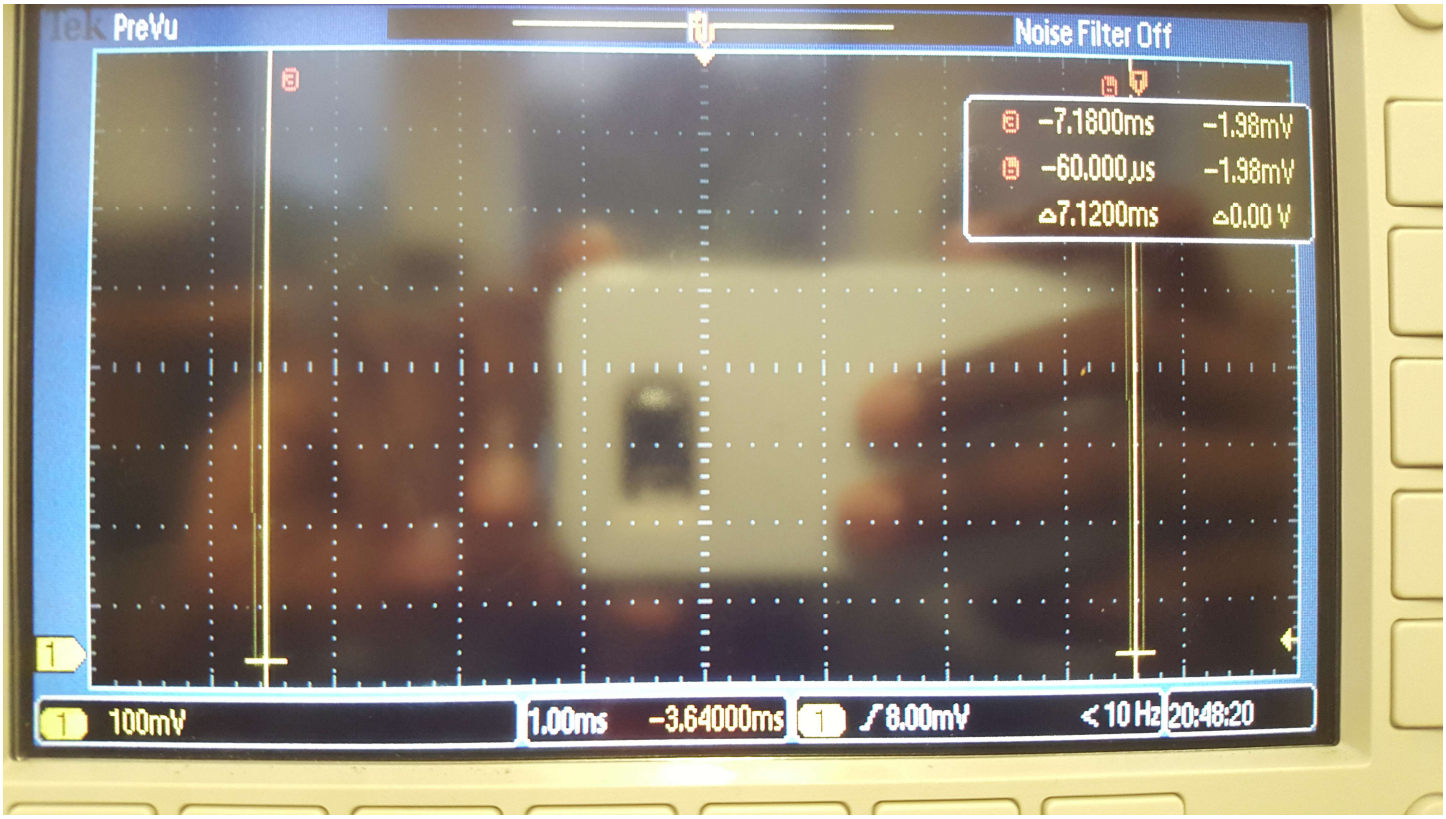
Circuit Diagram



Execution Time

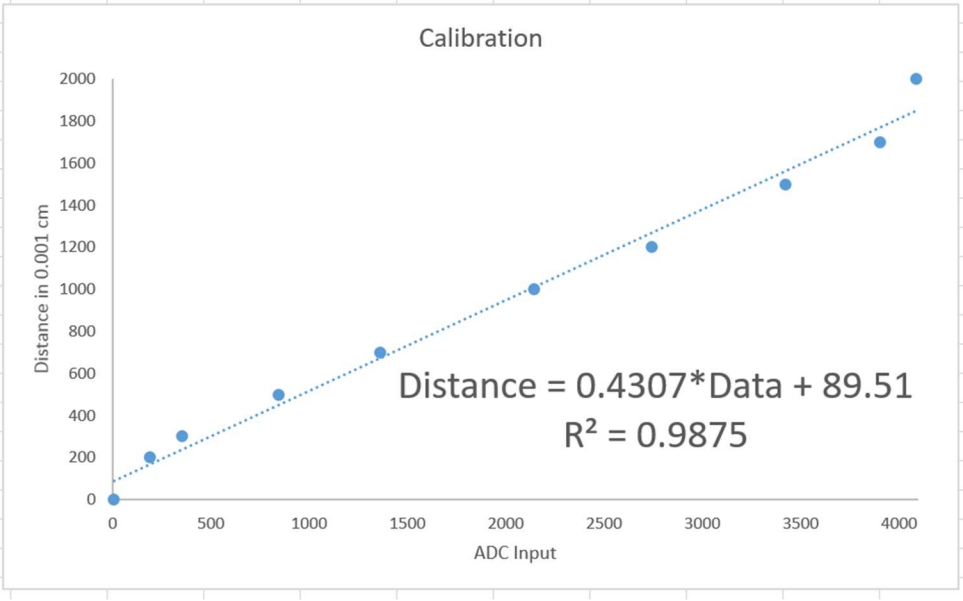
Time for ADC to sample

Time to output



Calibration Data

ADC Input	Distance	Analog Input
5	0	0.004
190	200	0.153
354	300	0.285
845	500	0.681
1360	700	1.096
2140	1000	1.725
2742	1200	2.210
3423	1500	2.758
3903	1700	3.145
4087	2000	3.294



Code

Main.c
// Lab8.c
// Runs on LM4F120 or TM4C123

```
// Student names: Chimezie Iwuanyanwu & Xinyuan Pan
// Last Modified: 4/12/2016

// Analog Input connected to PE2=ADC1
// displays on Sitronox ST7735
// PF3, PF2, PF1 are heartbeats

#include <stdint.h>

#include "ST7735.h"
#include "TExaS.h"
#include "ADC.h"
#include "print.h"
#include "tm4c123gh6pm.h"
#include "SysTickInts.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 Nokia
// main3 adds your convert function, position data is no Nokia

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){

        GPIO_PORTF_DEN_R |= 0x0E;                                   // Initialiation of PortF
        GPIO_PORTF_DIR_R |= 0x0E;
        GPIO_PORTF_AMSEL_R &= ~0x0E;
        GPIO_PORTF_AFSEL_R &= ~0x0E;
        GPIO_PORTF_PCTL_R &= ~0x0E;
        GPIO_PORTF_DATA_R |= 0x0E;
}
uint32_t Data;        // 12-bit ADC
uint32_t Position;    // 32-bit fixed-point 0.001 cm
int main1(void){      // single step this program and look at Data
  TExaS_Init();       // Bus clock is 80 MHz
  ADC_Init();         // turn on ADC, set channel to 1
  while(1){
    Data = ADC_In();  // sample 12-bit channel 1
  }
}

int main2(void){
        SYSCTL_RCGC2_R |= 0x31;
                                                                                                        //
Turns on clock for Port A, E, F
        int32_t delay = SYSCTL_RCGC2_R;
```

```
  TExaS_Init();       // Bus clock is 80 MHz
  ADC_Init();         // turn on ADC, set channel to 1
  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 1
    PF2 = 0x00;       // end of ADC Profile
    ST7735_SetCursor(0,0);
    PF1 = 0x02;       // Profile LCD
    LCD_OutDec(Data);
    ST7735_OutString("    ");  // these spaces are used to coverup characters from last output
    PF1 = 0;          // end of LCD Profile
  }
}

uint32_t Convert(uint32_t input){
  return (4307*input)/10000 + 90; // Formula found from taking a linear representation of ADC input
}
int main3(void){
          SYSCTL_RCGC2_R |= 0x31;
                                                                                                  //
Turns on clock for Port A, E, F
          int32_t delay = SYSCTL_RCGC2_R;
  TExaS_Init();         // Bus clock is 80 MHz
  ST7735_InitR(INITR_REDTAB);
  PortF_Init();
  ADC_Init();         // turn on ADC, set channel to 1
  while(1){
                      PF2 ^= 0x04;      // Heartbeat
    Data = ADC_In();  // sample 12-bit channel 1
    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("    "); // Prints raw ADC input
    ST7735_SetCursor(6,0);
    LCD_OutFix(Position); // Prints convereted input
    PF1 = 0;          // end of LCD Profile

  }
}
uint32_t ADCMail, ADCStatus;
int main(void){
          SYSCTL_RCGC2_R |= 0x31;
                                                                                                  //
Turns on clock for Port A, E, F
          int32_t delay = SYSCTL_RCGC2_R;
          ADCStatus = 0; ADCMail = 0; // Initializes ADC status and input
  TExaS_Init();         // Bus clock is 80 MHz
```

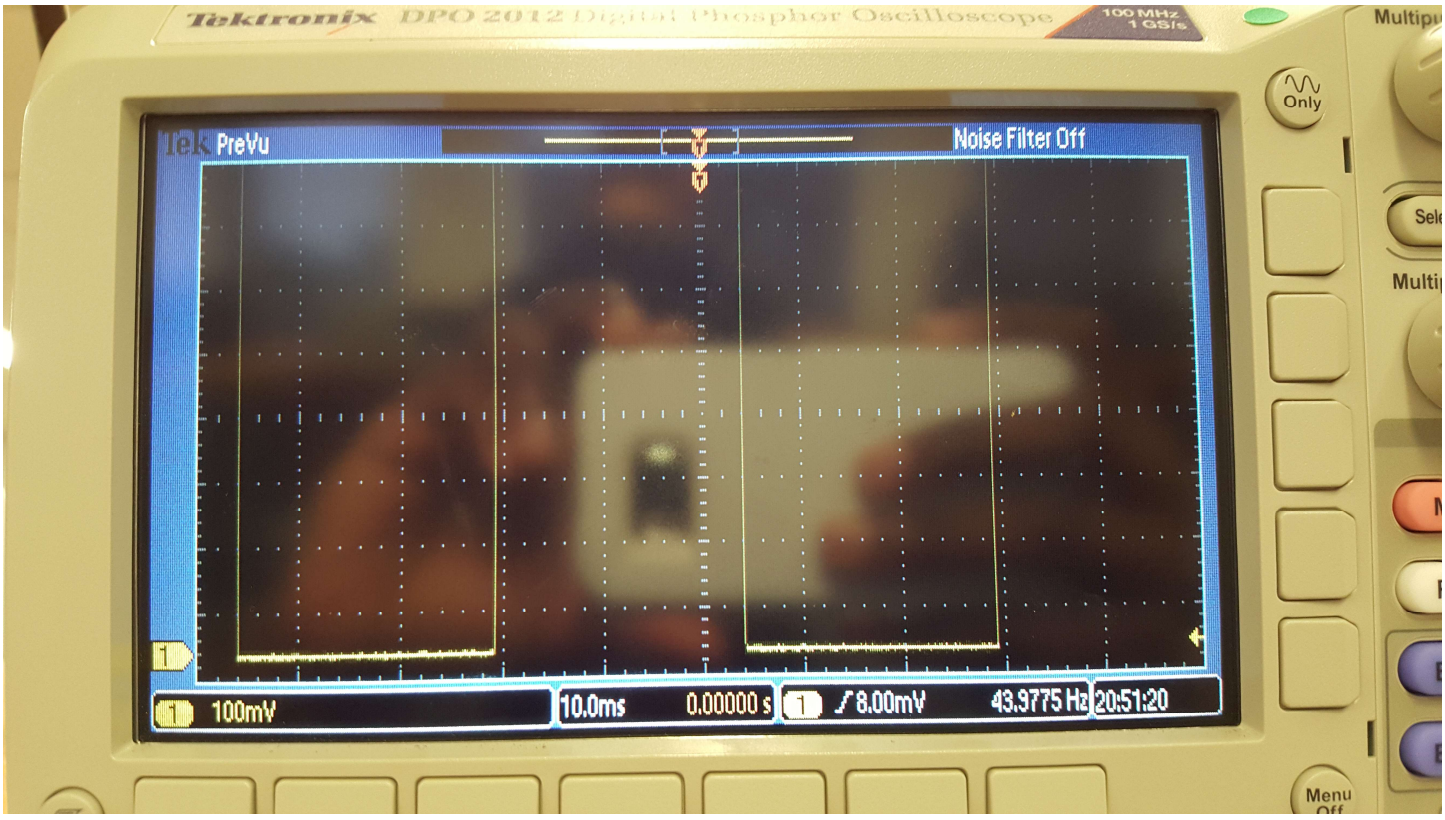
```
  PortF_Init();
  ADC_Init();        // turn on ADC, set channel to 1
        SysTick_Init(80000000/80);  // Sets frequency to 40 Hz sampling
        ST7735_InitR(INITR_REDTAB);
  while(1){
                while(ADCStatus == 0){} // Waits till status is set to print
                ADCStatus = 0;                                  // Sets status to wait for print
                Data = ADCMail;
                PF3 = 0x08;       // Profile Convert
                Position = Convert(Data);  // Converts ADC input to position
                PF3 = 0;          // end of Convert Profile

                PF1 = 0x02;       // Profile LCD
                ST7735_SetCursor(0,0);
                LCD_OutFix(Position); // Prints convereted input
                ST7735_OutString(" cm");
                PF1 = 0;          // end of LCD Profile
        }
}
void SysTick_Handler(void){
        PF2 ^= 0x04;      // Heartbeat
        PF2 ^= 0x04;      // Heartbeat
        ADCMail = ADC_In();         // Fetches what ADC input is
        ADCStatus = 1;                          // Sets status to print
        PF2 ^= 0x04;      // Heartbeat
}
```

ADC.c

```
// ADC.c
// Runs on LM4F120/TM4C123
// Provide functions that initialize ADC0
// Last Modified: 4/12/2016
// Student names: Chimezie Iwuanyanwu & Xinyuan Pan
// Last modification date: change this to the last modification date or look very silly

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

// ADC initialization function
// Input: none
// Output: none
void ADC_Init(void){
        int32_t delay = SYSCTL_RCGC2_R;
  GPIO_PORTE_DIR_R &= ~0x10;      // 2) make PE4 input
  GPIO_PORTE_AFSEL_R |= 0x10;     // 3) enable alternate fun on PE4
  GPIO_PORTE_DEN_R &= ~0x10;      // 4) disable digital I/O on PE4
  GPIO_PORTE_AMSEL_R |= 0x10;     // 5) enable analog fun on PE4
  SYSCTL_RCGCADC_R |= 0x01;       // 6) activate ADC0
  delay = SYSCTL_RCGCADC_R;       // extra time to stabilize
  delay = SYSCTL_RCGCADC_R;       // extra time to stabilize
  delay = SYSCTL_RCGCADC_R;       // extra time to stabilize
  delay = SYSCTL_RCGCADC_R;
```

```
        ADC0_SAC_R = 0x04;
  ADC0_PC_R = 0x01;             // 7) configure for 125K
  ADC0_SSPRI_R = 0x0123;        // 8) Seq 3 is highest priority
  ADC0_ACTSS_R &= ~0x0008;      // 9) disable sample sequencer 3
  ADC0_EMUX_R &= ~0xF000;       // 10) seq3 is software trigger
  ADC0_SSMUX3_R = (ADC0_SSMUX3_R&0xFFFFFFF0)+9;  // 11) Ain9 (PE4)
  ADC0_SSCTL3_R = 0x0006;       // 12) no TS0 D0, yes IE0 END0
  ADC0_IM_R &= ~0x0008;         // 13) disable SS3 interrupts
  ADC0_ACTSS_R |= 0x0008;       // 14) enable sample sequencer 3
}

//------------ADC_In------------
// Busy-wait Analog to digital conversion
// Input: none
// Output: 12-bit result of ADC conversion
uint32_t ADC_In(void){
        uint32_t data;
  ADC0_PSSI_R = 0x0008; // Turns on ADC get value
  while((ADC0_RIS_R&0x08)==0){};   // Waits till ADC gets value from input
  data = ADC0_SSFIFO3_R&0xFFF;  // Reads data from ADC
  ADC0_ISC_R = 0x0008; // Clears flag to signal data has been read
  return data;
}
```

Sample Rate



Accuracy Calculation

True position x_{ti}	Measured Position x_{mi}	Error $x_{ti} - x_{mi}$
0.3	0.254	0.046
0.6	0.583	0.017
0.9	0.927	-0.027
1.2	1.283	-0.083
1.5	1.574	-0.074

Maximum Error: 0.083cm Average Error: 0.049cm