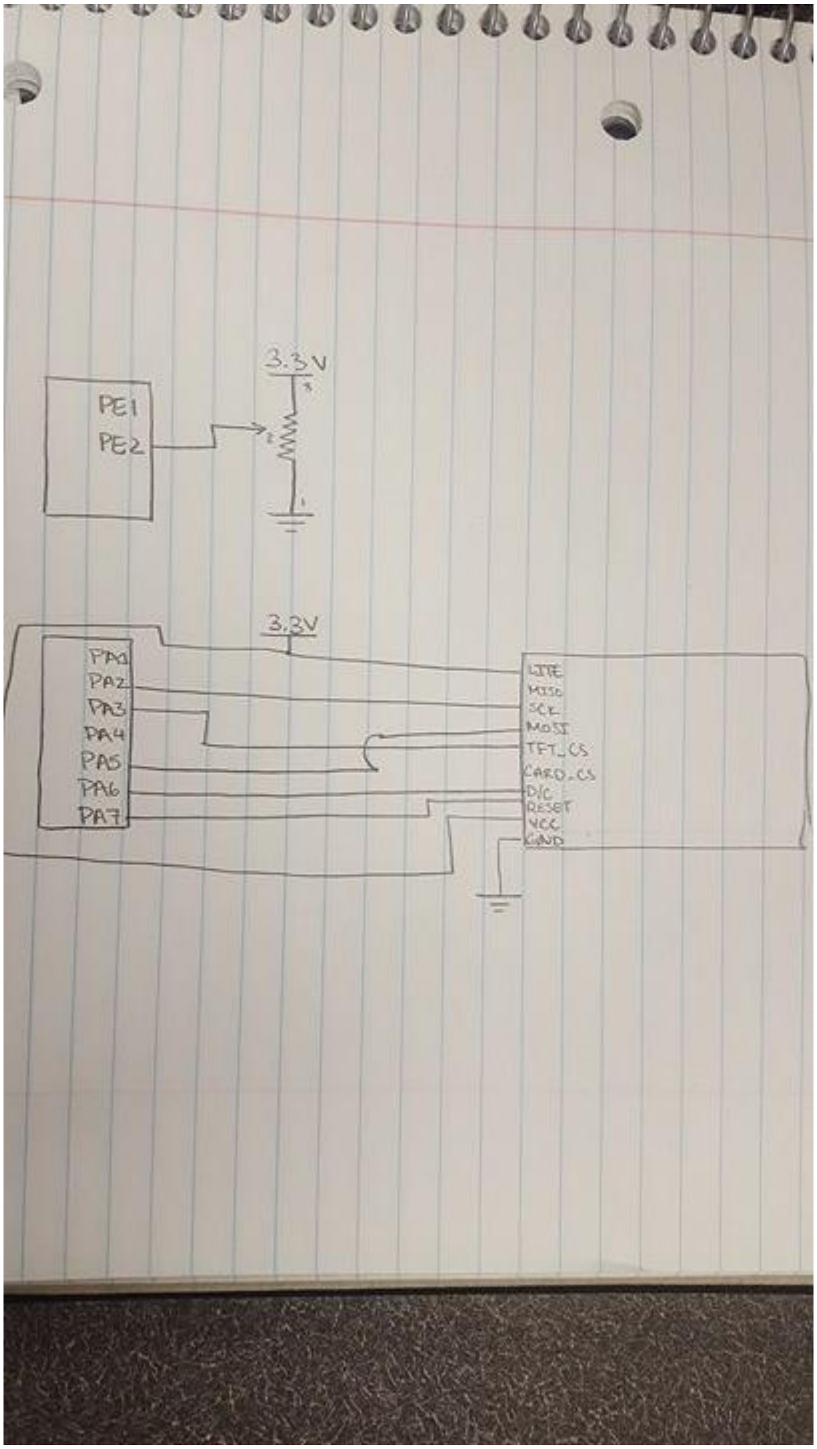
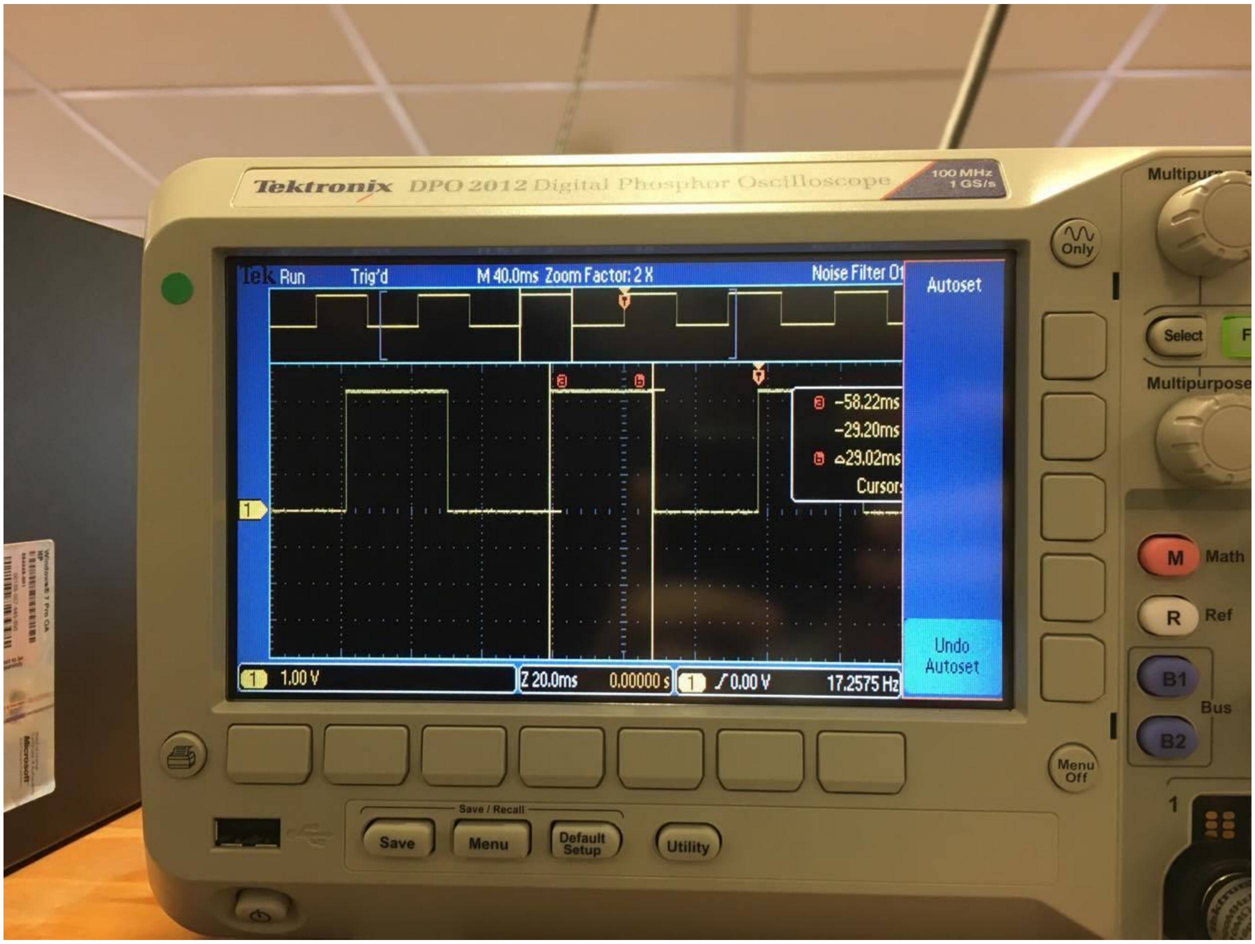
Lab 8 Deliverables

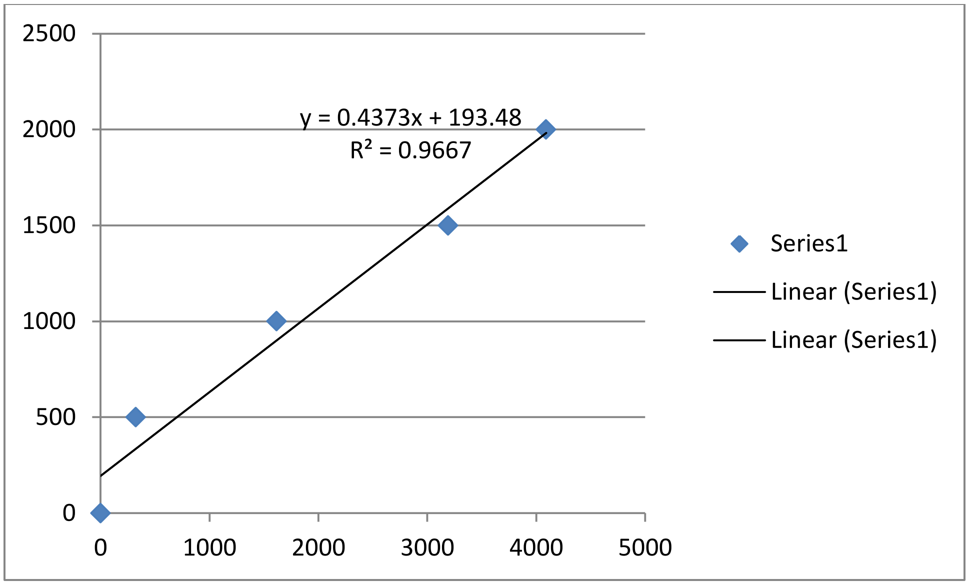
Daniel Canterino djc3323

Pranav Padmanabha pp9638





Analog Input	reading	point in .001cm
0.24	0	0
1.12	322	500
2.05	1616	1000
2.92	3192	1500
3.28	4092	2000



```
// Lab8.c

// Runs on LM4F120 or TM4C123

// Student names: Pranav Padmanabha and Daniel Canterino

// Last modification date: 4/8/2017

// Last Modified: 4/5/2016


// 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"

//*****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
#define PD0                                           (*((volatile uint32_t *)0x40007004))
volatile uint32_t Counts;
int ADCmail;
int ADCstatus;


void PortF_Init(void){
        SYSCTL_RCGCGPIO_R|=0x20;
        GPIO_PORTF_LOCK_R=GPIO_LOCK_KEY;
        GPIO_PORTF_CR_R|=0x0E;
        GPIO_PORTF_DIR_R|=0x0E;
```

```
	GPIO_PORTF_DEN_R|=0x0E;
	GPIO_PORTF_AFSEL_R&=~(0x0E);//turn on pf123 for heartbeats, no inputs, only outputs, disable af and am
	GPIO_PORTF_AMSEL_R&=~(0x0E);
	GPIO_PORTF_PCTL_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){
  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){
	input=((4373*input)+1984300)/10000;							///NEED TO STABILIZE DATA AND GET DATA FOR LINE, RIGHT NOW IT IS NOT LINEAR
  return input;
}
int main3(void){
  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("    ");
  ST7735_SetCursor(6,0);
  LCD_OutFix(Position);
  PF1 = 0;          // end of LCD Profile
 }
}



        void SysTick_Init(uint32_t period){
        NVIC_ST_CTRL_R = 0; //disable the interrupt during setup
        NVIC_ST_RELOAD_R = period - 1; //reload value (2*10^6)
        NVIC_ST_CURRENT_R = 0; //clears data in current
        NVIC_SYS_PRI3_R = (NVIC_SYS_PRI3_R&0x00FFFFFF)|0x40000000; //priority 2
        NVIC_ST_CTRL_R = 0x00000007; //Enable the interrupts
}


void SysTick_Handler(void){
        GPIO_PORTF_DATA_R^=0x02;//toggles pf1
        GPIO_PORTF_DATA_R^=0x04;//toggles pf2

        Data= ADC_In();//sets data equal to the adc input

        ADCmail=Data;//sets global variable adcmail to adc input

        ADCstatus=1;//sets global variable adcstatus to 1 to indicate there is new data available

        GPIO_PORTF_DATA_R^=0x08;//toggles pf3

}


int main(void){

  TExaS_Init();

  // your Lab 8

        ST7735_InitR(INITR_REDTAB);

  PortF_Init();

  ADC_Init();      // turn on ADC, set channel to 1

        EnableInterrupts();      //enables interrupts after all initialization is complete

        int output;

  while(1){

                SysTick_Init(1160000);   //calls systick init with 2*10^6 ;;;         80MHz * 40Hz (1/40)

                while(ADCstatus==0){//waits for systick handler to set adc status to 1 indicating fresh
data

                        }

                output=Convert(ADCmail);// calls convert on the adc data stored at the global variable --
> converts to cm stores to output

                ST7735_SetCursor(0,0);

   LCD_OutDec(ADCmail);                  //assembly function out dec

                ST7735_OutString("    ");

   ST7735_SetCursor(6,0);
   LCD_OutFix(output);                           //assembly function outfix
                        ST7735_OutString(" cm");
                        ADCstatus=0;                                                     //resets adc status to 0
  }
}
// ADC.c
// Runs on LM4F120/TM4C123
// Provide functions that initialize ADC0
// Last Modified: 3/6/2015
// 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 "tm4c123gh6pm.h"

// ADC initialization function
// Input: none
// Output: none
void ADC_Init(void){

            SYSCTL_RCGCGPIO_R|=0x10;//port e initialiation, pe2 is input, no outputs, enable alternate function and analong function on pe2, disable i/o function
            while ((SYSCTL_PRGPIO_R&0x10) ==0){};//2 cycles
            GPIO_PORTE_DIR_R&=~(0x04);
            GPIO_PORTE_AFSEL_R|=(0x04);
            GPIO_PORTE_DEN_R&=~(0x04);
```

```
	GPIO_PORTE_AMSEL_R|=(0x04);


	SYSCTL_RCGCADC_R|=0x01;//activate adc0
	while ((SYSCTL_RCGCADC_R&0x01)==0){};		//4 cycles
	ADC0_PC_R=0x01;//configure for 125K
	ADC0_SSPRI_R=0x0123;//sequencer 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) +1;//clear ss3 field and set channel ain0 (pe2)
	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=0x04;//AVERAGES 16 SAMPLES NOW


}


//------------ADC_In------------
// Busy-wait Analog to digital conversion
// Input: none
// Output: 12-bit result of ADC conversion
uint32_t ADC_In(void){
	uint32_t result;
	ADC0_PSSI_R = 0x0008;//initiate ss3
	while ((ADC0_RIS_R & 0x08) == 0){//wait for conversion done
```

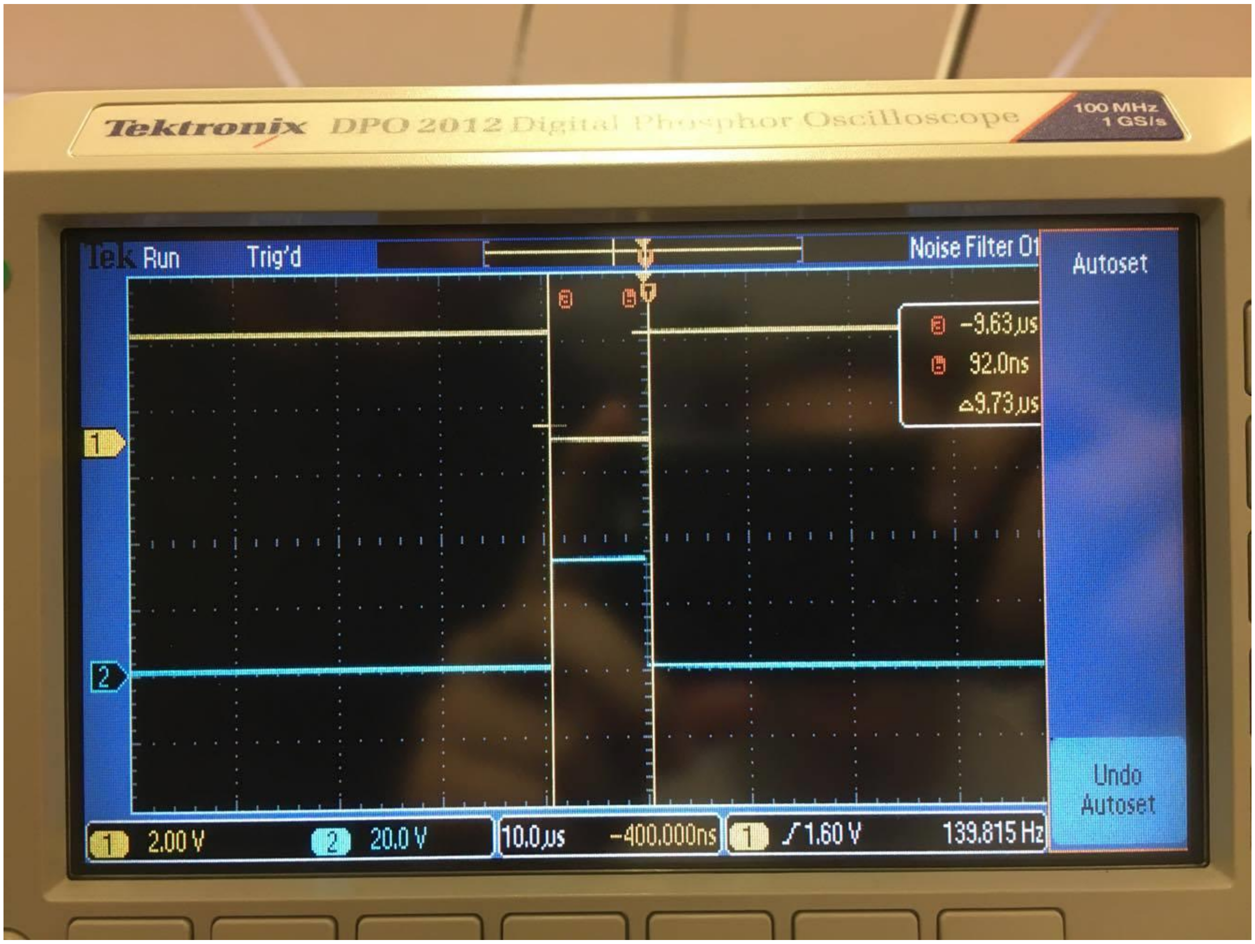
```
          }

          result=(ADC0_SSFIFO3_R & 0xFFF);//read 12-bit result

          ADC0_ISC_R = 0x0008;//acknowledge completion

          return (result);

}
```



True Position x_{ti}	Measured Position x_{mi}	Error x_{ti}-x_{mi}
1.17cm	1.155cm	.015cm
.52cm	.489cm	.031cm
1.68	1.709	.011cm
.41cm	.319	.091cm
.73cm	.738cm	.008cm