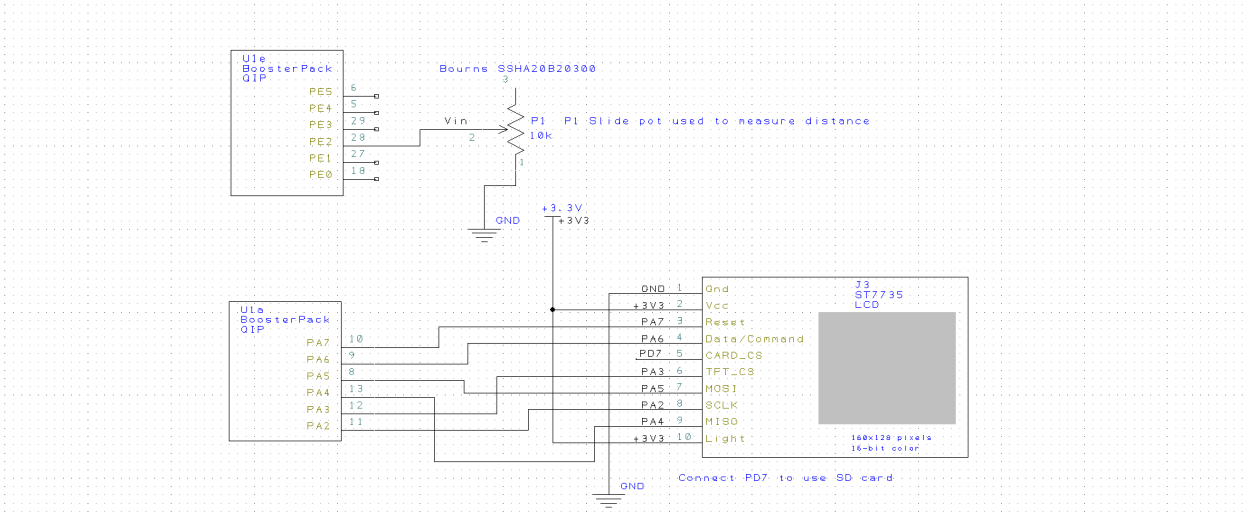


All resistors are 1/4 watt 5% carbon composition
C1 is ceramic Z5U
Switches are www.BGMicro.com SWT1043
Red LEDs. T1 3/4. 20mA Digikey 160-1087-ND
Yellow LEDs. T1 3/4. 20mA Digikey 160-1088-ND
Green LEDs. T1 3/4. 20mA Digikey 160-1089-ND
Slide pot. Bourns SSHA20B20300 www.AllElectronics.comSP-20K

University Of Texas At Austin

Schematic Name: EK-LM4F120XL or EK-TM4C123GXL

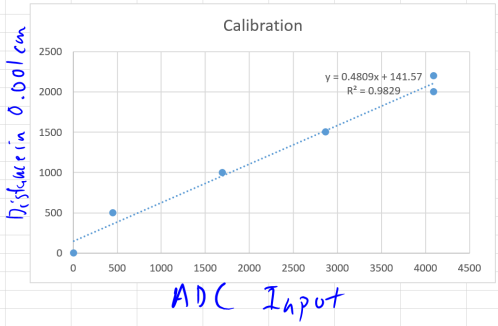
Name(s): Ali Ziyaan Momin and Zain Modi

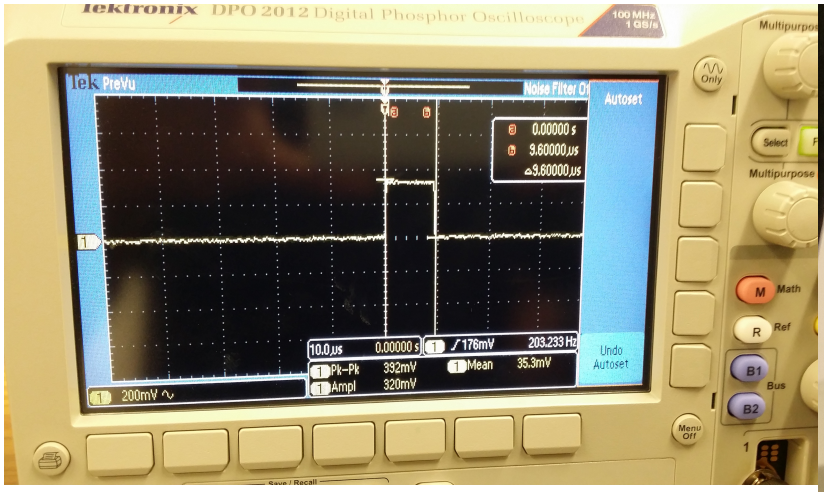
Date: November 11, 2015 **Semester:** Fall 2015

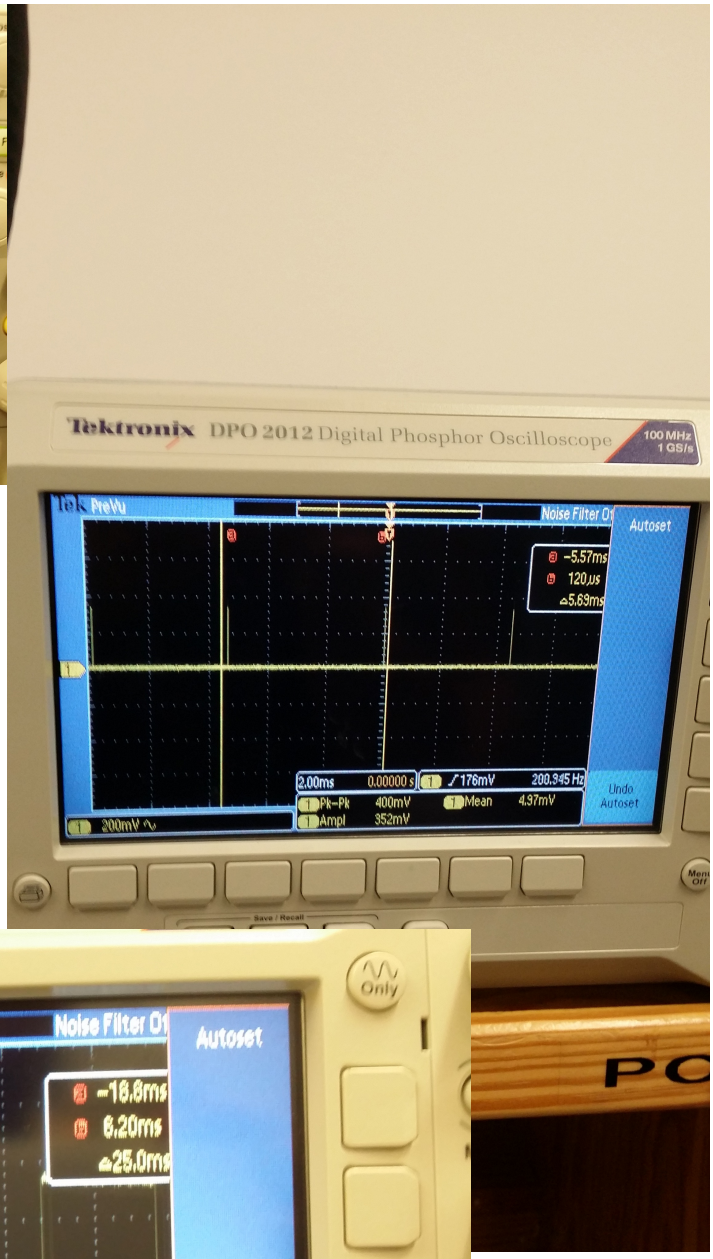
H3

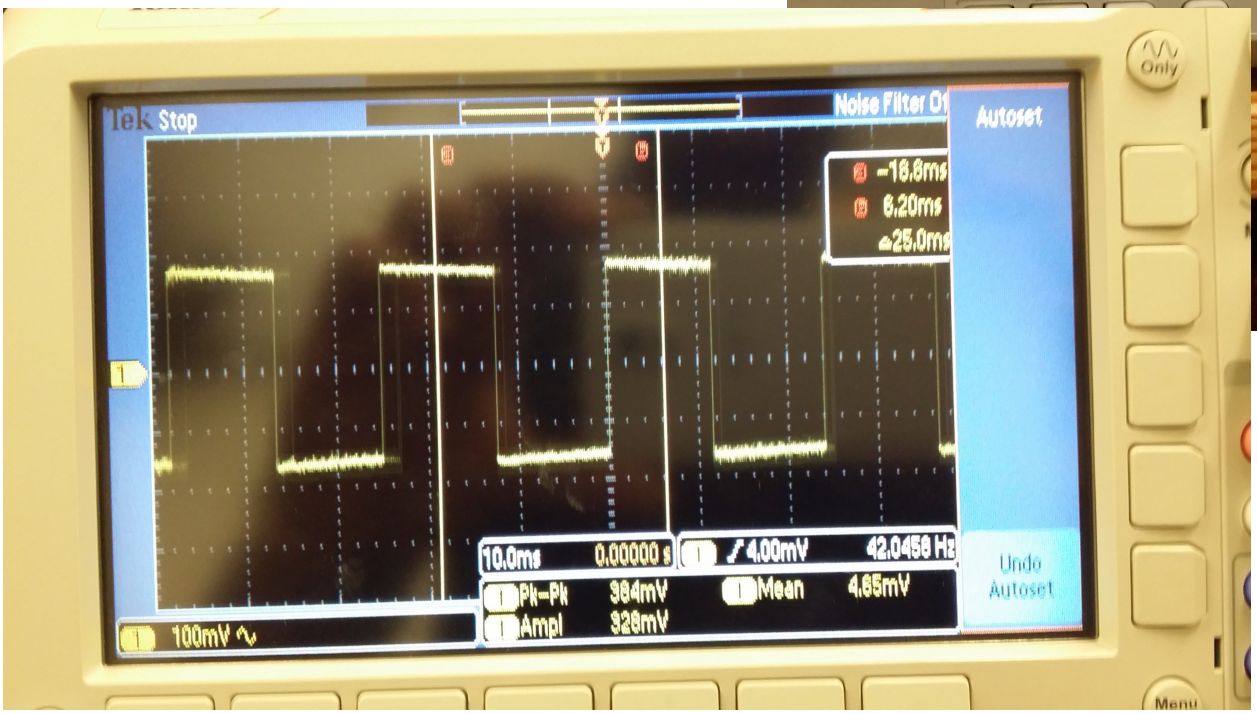
	A	B	C	D	E	F	G	H	I
1	True Position		Measured Position		Error		Avg Error		
2	0		0.142		14.20%		7.34%		
3	0.5		0.457		4.30%				
4	1		1.022		2.20%				
5	1.5		1.562		6.20%				
6	2		2.098		9.80%				
7	2.2		2.11		9%				

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	8	0		0		0.142		11							
2	453	500		0.5		0.487		495							
3	1693	1000		1		0.986		994							
4	2865	1500		1.5		1.605		1489							
5	4092	2000		2		2.108		1996							
6	4095	2200		2.2		2.111		2183							
7	ADC sample			Position (cm)		Analog input		Measured Fixed Pt.							
8		Corrected fixed pt													




Tektronix DPO 2012 Digital Phosphor Oscilloscope
100 MHz 1 GS/s
Tek PreVu
Noise Filter Of
Autoset
0.00000 s
9.60000 µs
Δ9.60000 µs
10.0 µs 0.00000 s
176mV 203.233 Hz
Pk-Pk 392mV
Mean 35.3mV
Ampl 320mV
200mV
Undo Autoset
Multipurpose
Select
Math
Ref
Bus
B1
B2
Menu Off


Tektronix DPO 2012 Digital Phosphor Oscilloscope
100 MHz 1 GS/s
Tek PreVu
Noise Filter Of
Autoset
-5.57ms
120 µs
Δ5.69ms
2.00ms 0.00000 s
176mV 200.945 Hz
Pk-Pk 400mV
Mean 4.37mV
Ampl 352mV
200mV
Undo Autoset


Tek Stop
Noise Filter Of
Autoset
-18.8ms
6.20ms
Δ25.0ms
10.0ms 0.00000 s
4.00mV 42.0458 Hz
Pk-Pk 384mV
Mean 4.65mV
Ampl 328mV
100mV
Undo Autoset
Only
Menu

```
// Lab8.c
// Runs on LM4F120 or TM4C123
// Student names: Zain Modi and Ali Ziyaan Momin
// Last modification date: change this to the last modification date or look
very silly
// Last Modified: 3/6/2015

// 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

uint32_t data = 0;
uint32_t ADCStatus = 0;
uint32_t ADCMail = 0;

#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){
        SYSCTL_RCGCGPIO_R |= 0x20;
        //NOP
        int i;
        extern int idx;
```

```
        for(i=0; i<5; i ++){
        }
        //Setting I/0
        GPIO_PORTF_DIR_R |= 0x04;
        //AFSEL off
        GPIO_PORTF_AFSEL_R = 0x00;
        //DEN on
        GPIO_PORTF_DEN_R |= 0x04;
}
void SysTick_Init(void){
        NVIC_ST_CTRL_R = 0;
        NVIC_ST_RELOAD_R = 800000000;
        NVIC_ST_CURRENT_R = 0;
        NVIC_ST_CTRL_R = 0x00000007;
        ADC_Init();
        PortF_Init();
}

uint32_t AVeraG(void){
        uint32_t avg;
        int z;
        uint32_t dat = 0;
        for(z=0; z<256; z ++){
                dat = dat + ADCMail;
        }
        avg = dat / z;

        return(avg);
}
void SysTick_Handler(void){
        GPIO_PORTF_DATA_R ^= 0x04;
        GPIO_PORTF_DATA_R ^= 0x04;
        ADC0_ISC_R = 0x0008;
        ADCMail = ADC_In();
        ADCStatus = 1;
        GPIO_PORTF_DATA_R ^= 0x04;
}
uint32_t Data;        // 12-bit ADC
uint32_t Position;    // 32-bit fixed-point 0.001 cm
int main8(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 main6(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
  }
}
//y=0.4809x + 141.57
uint32_t Convert(uint32_t input){
  uint32_t y;
        y = 481*input + 141570;
        y = y / 1000;
        return (y);
}
int main9(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
  }
}
int main(void){
  TExaS_Init();
        ST7735_InitR(INITR_REDTAB);
        SysTick_Init();
  // your Lab 8
  while(1){
                uint32_t con;
                if(ADCStatus == 1){
                        data = AVeraG();
                        con = Convert(data);
                        ST7735_SetCursor(0,0);
                        LCD_OutFix(con);
                        ADCStatus = 0;
                        PF1 = 0;          // end of LCD Profile

                }
  }
}


// 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;
                                                        // 1)  Activate clock for Port
E
  while((SYSCTL_PRGPIO_R&0x10) == 0){};
        int i;
        for(i=0; i<5; i ++){
        }
        //Setting I/0
        GPIO_PORTE_DIR_R &= 0xFB;
        //AFSEL off
        GPIO_PORTE_AFSEL_R |= 0x04;
        //DEN on
        GPIO_PORTE_DEN_R &= 0xFB;
        GPIO_PORTE_AMSEL_R |= 0x04;
        SYSCTL_RCGCADC_R |= 0x01;
        for(i=0; i<5; i ++){
        }
        ADC0_PC_R = 0x01;
        ADC0_SSPRI_R |= 0x123;
        ADC0_ACTSS_R &= 0xF7;
        ADC0_EMUX_R |= 0x0FFF; //Hold Johnny Responsible
        ADC0_SSMUX3_R = (ADC0_SSMUX3_R & 0xFFFFFFF0) + 1;
        ADC0_SSCTL3_R = 0x06;
        ADC0_IM_R &= 0xF7;
        ADC0_ACTSS_R |= 0x08;

}

//------------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;
        while((ADC0_RIS_R & 0x08) == 0){};
        data = ADC0_SSFIFO3_R & 0xFFF;
        ADC0_ISC_R = 0x0008;
  return data;
}
```