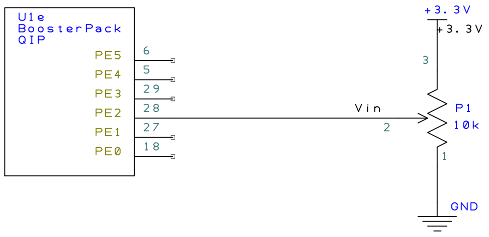


P1 Slide pot used to measure distance

Bourns PTA2043-2015CPB103

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 PTA2043-2015CPB103. Mouser 652-PTA20432015CPB10

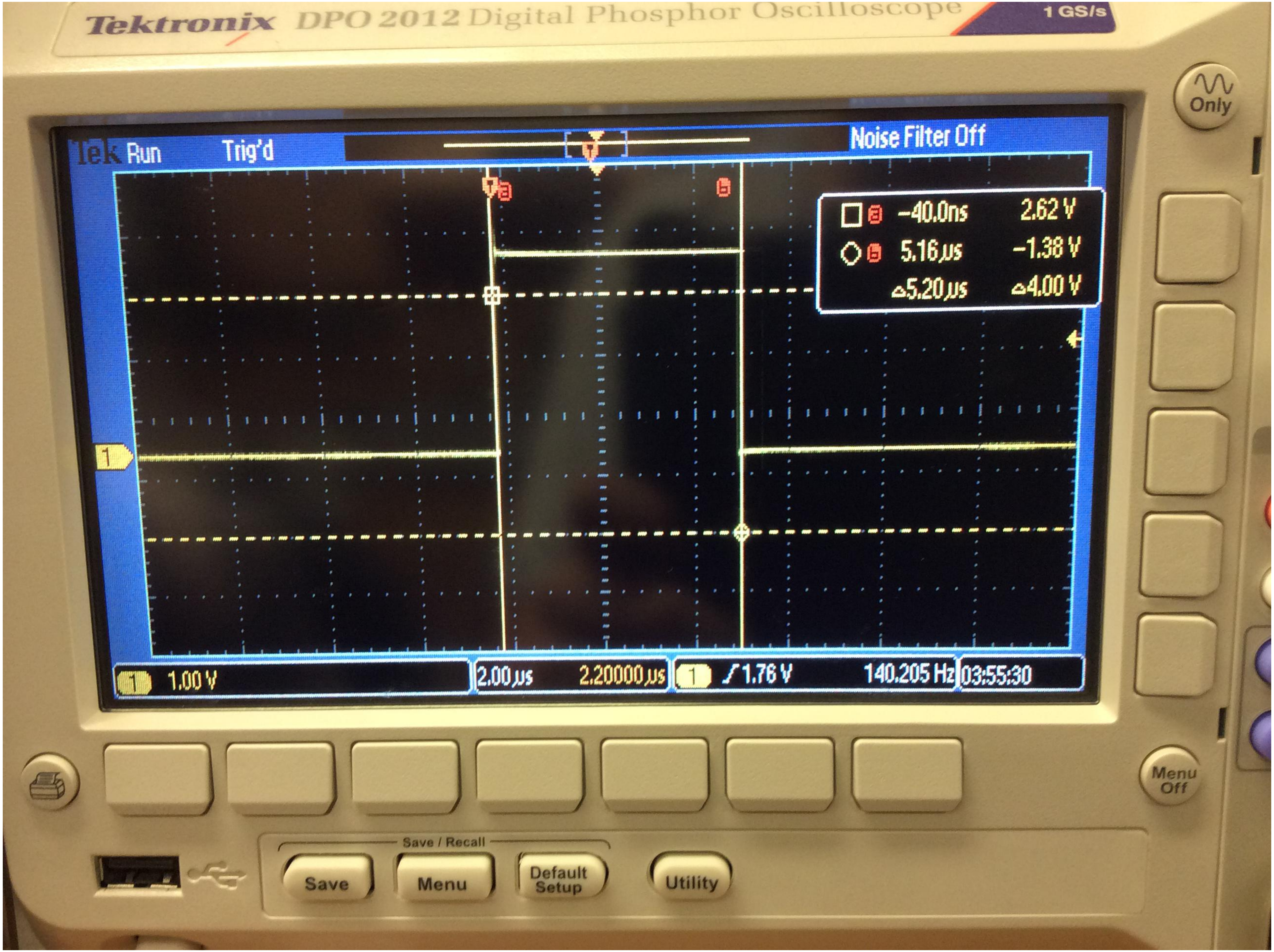
A1
frame

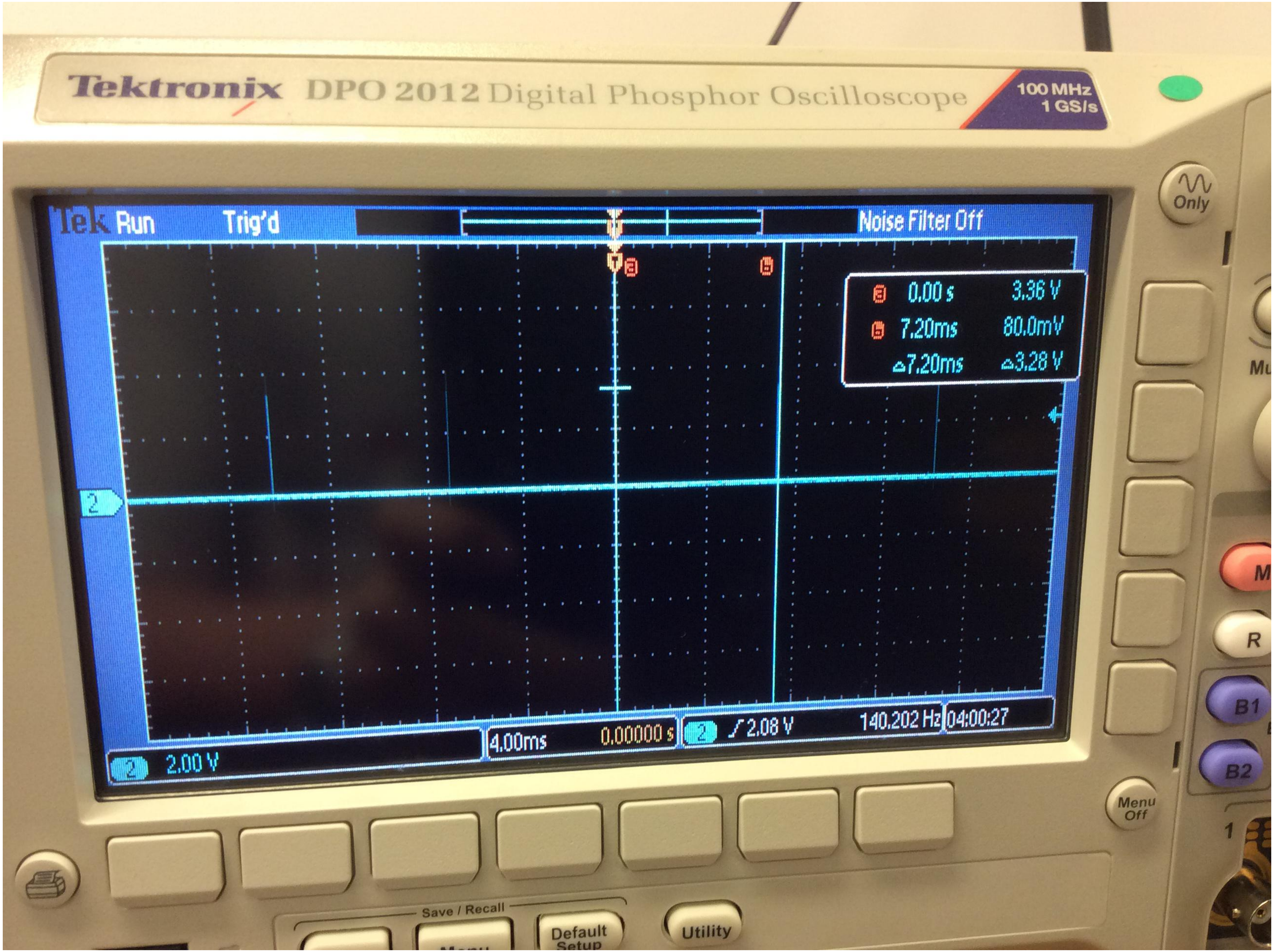
University Of Texas At Austin

Schematic Name: EK-LM4F120XL or EK-TM4C123GXL

Name(s): Aria Pahlavan, Khalid Qarryzada

Date: April 14, 2015 **Semester:** Spring 2015


Tektronix DPO 2012 Digital Phosphor Oscilloscope
1 GS/s
Tek Run
Trig'd
Noise Filter Off
a −40.0ns 2.62 V
b 5.16µs −1.38 V
△5.20µs △4.00 V
1
1 1.00 V
2.00µs
2.20000µs
1 ⟋1.76 V
140.205 Hz
03:55:30
Only
Menu Off
Save / Recall
Save
Menu
Default Setup
Utility


Tektronix DPO 2012 Digital Phosphor Oscilloscope
100 MHz
1 GS/s
Tek Run
Trig'd
Noise Filter Off
a 0.00 s 3.36 V
b 7.20ms 80.0mV
△7.20ms △3.28 V
2
4.00ms
0.00000 s
2 ⁄ 2.08 V
140.202 Hz
04:00:27
2 2.00 V
Only
Menu Off
R
B1
B2
Save / Recall
Default Setup
Utility

CALIBRATION DATA

Position	Analog Input	ADC Sample
0.1	0	3
0.5	0.55	683
0.8	1.2	1489
1	1.6	1970
1.2	1.99	2459
1.5	2.65	3288
1.6	2.88	3580
1.8	3.29	4092
2	3.3	4095

//SysTick

```
void SysTick_Init(void){
      NVIC_ST_CTRL_R = 0;                    // disable SysTick during setup
      NVIC_ST_CTRL_R = 0x07;      // enable SysTick with core clock &
enable interrupt
      NVIC_ST_RELOAD_R = 2000000;  // Set to this value to make interrupts
occur every 25 ms
      NVIC_ST_CURRENT_R = 0;       // any value written to CURRENT clears
}
uint32_t ADCMail, ADCStatus;

void SysTick_Handler(void){
      PF2 ^= 0x04;
      PF2 ^= 0x04;
      ADCMail = ADC_In();
      ADCStatus = 1;
      PF2 ^=0x04;
      return;
}
```

//ADC

```
void ADC_Init(void){//*********channel: PE2 = Ain1*********)
      SYSCTL_RCGCGPIO_R|= 0x10;          //Enable the port clock for the
ADC input pin
      while ((SYSCTL_PRGPIO_R & 0x10) ==0){};  //Delay, wait for clock to
stabilize
      GPIO_PORTE_DEN_R &= ~0x04;               //Disable the digital
function
      GPIO_PORTE_AFSEL_R |= 0x04;              //Enable the alternative
function
      GPIO_PORTE_AMSEL_R |=0x04;               //Enable the analog function
      GPIO_PORTE_DIR_R &= ~0x04;               //Make the pin an input
(clear the bit)
      SYSCTL_RCGCADC_R |= 0x01;                //Enable the ADC clock
      volatile uint32_t delay;                            //declare
variable
      delay = SYSCTL_RCGCADC_R; //after setting SYSCTL_RCGADC_R, wait for
stablization

      delay = SYSCTL_RCGCADC_R;
      delay = SYSCTL_RCGCADC_R;
      ADC0_PC_R =0x03;                   //Configure for 250 kHz sampling
rate
      ADC0_SSPRI_R = 0x0123;             //Set Sequencer 3 to highest
priority
      ADC0_ACTSS_R &= ~0x08;       //Disable sample sequencer 3 bifore
configuring it
      ADC0_EMUX_R &= ~0xF000;//Sequencer 3 software trigger(software start)
      ADC0_SSMUX3_R=(ADC1_SSMUX3_R & ~0xF)+1; //Clear SS3 field and set
channel Ain1 (PE2 = Ain1)
      ADC0_SSCTL3_R = 0x06;        //TS0:Temperature(N). IE0:INR3 flag(Y).
END0:One sample(Y). D0: Differential sampling(N)
      ADC0_IM_R &= ~0x08;                //Disable SS3 interrupts
      ADC0_ACTSS_R |= 0x08;              //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 result = 0;                           //create a result var
      ADC0_PSSI_R = 0x8;                       //Initiate SS3
      while((ADC0_RIS_R & 0x8)==0){};                //Wait for conversion
done
      result = ADC0_SSFIFO3_R & 0xFFF;         //Read 12-bit result
      ADC0_ISC_R = 0x8;                        // Acknowledge completion
      return result;
}
```

//Convert

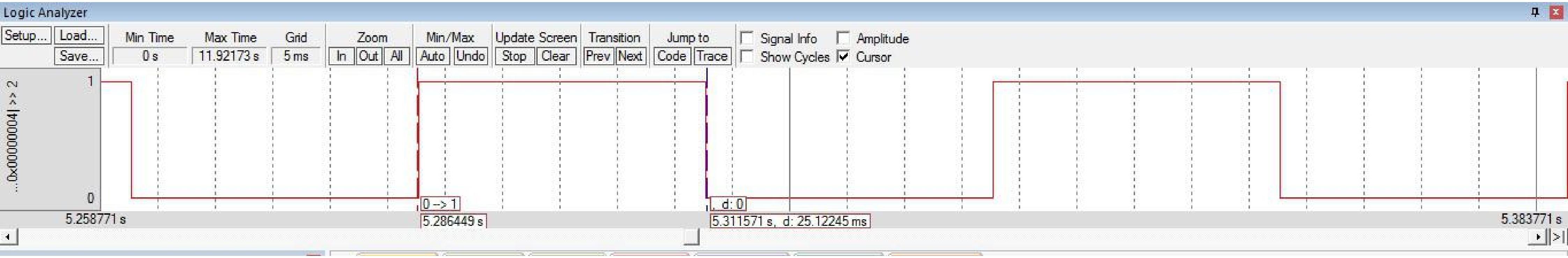
```
uint32_t Convert(uint32_t input){
      uint32_t previous=0;

      if(input <= 2) input=0; //if ADC inout less than 2 out 0.000 cm
            //else if(input < (previous+3) || input > (previous-3)) return
previous;
else if(input < 10) input=0.0386*input + 0.7857;          // for distance
less than 0.2 cm
      else if(input < (previous+5) || input > (previous-5)) return
previous; //if data ADC in is in the range of 5 return previous
```

```
            else if(input < 1400)  input = 0.381*input + 205.97;      //if
less than 0.7cm
                  else if(input < (previous+10) || input > (previous-10))
return previous;
                        else if(input <= 4030 && input ) input =
0.4198*input + 156.37; //if less than 1.7cm
                              else if(input <= 4094) input = 1.0274*input -
2207.2;          //if greater than 1.7cm
                                    else input = 2000;
                  //if greater than 2cm
            previous = input;
      return input;
}
```

//main

```
 int main(void){
 TExaS_Init();         // Bus clock is 80 MHz
  ST7735_InitR(INITR_REDTAB);
  PortF_Init();
  ADC_Init();         // turn on ADC, set channel to 1
  SysTick_Init();            //Initialize SysTick
  for(;;){
        while(ADCStatus == 0){}    //Poll ADCStatus flag
        uint32_t x = ADCMail;      //read ADCMail (input)
        ADCStatus = 0;             //clear flag
        x = Convert(x);            //convert the input
        ST7735_SetCursor(1,7);
    ST7735_OutString("D = ");      //print "D = "
    ST7735_SetCursor(5,7);
    LCD_OutFix(x);                       // print the fixed point value
            ST7735_SetCursor(10,7);
            ST7735_OutString(" cm");     // print " cm"
            /////////////////////
            ST7735_SetCursor(1,2);
                  ST7735_OutString("Lab 8:");
            ST7735_SetCursor(1,3);
            ST7735_OutString("Measurment of");
            ST7735_SetCursor(1,4);
            ST7735_OutString("Distance :)");
      }}
```



Logic Analyzer

Min Time: 0 s
Max Time: 11.92173 s
Grid: 5 ms

□ Signal Info
□ Amplitude
□ Show Cycles
☑ Cursor

ACCURACY CALCULATIONS

True position x_{ti}	Measured Position x_{mi}	Error $x_{ti} - x_{mi}$
0	0.000	0
0.3	0.303	-0.003
0.8	0.798	0.002
1.5	1.553	-0.053
1.9	1.990	-0.090