Ronald Macmaster and Parth Adhia

EID: Rpm953 and Pda375

11/11/2016

EE 445L – Lab 9:

Temperature Data Acquisition System

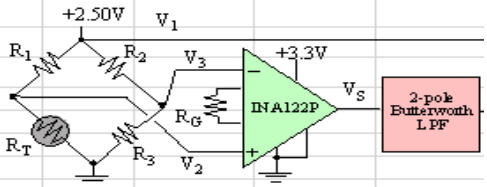
0.0: Objectives:

We will further study the concepts of ADC conversion and the Nyquist Theorem. Specifically, we plan to develop a temperature measurement system that uses a thermistor. We plan to analyze different performance measurements of the system including precision, resolution, and accuracy. To realize a working system, we will also have to understand some general concepts of analog circuit design. Our circuit will require the use of an operational amplifier and instrumentation amplifier. We need to calibrate these analog circuits in a way that will allow for the most accurate temperature measurements in the overall system. That means we also must gain a deep understanding of how these integrated circuits work and the parameters associated with them.

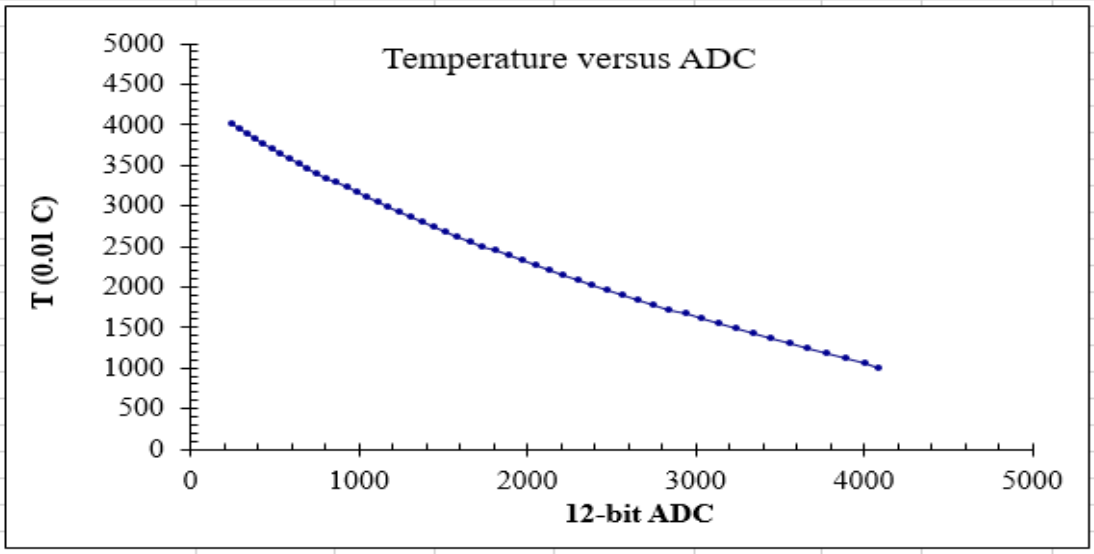
1.0: Sensor Calibration:

Here is a snapshot of our calibration Excel sheet.

1	1) Enter two point calibration			**Thermistor interface design process by Jonathan Valvano**					
2		R(kΩ)	T(°C)						
3	First Point=	95.8	25						
4	Second Point=	69	34		T=1/(H0+H1ln(R))-273.15			1/(T+273.15)=H0+H1ln(R)	
5	H1=	0.0002995			R=R0exp(b/(T+273.15))			ln(R)=ln(R0)+(b/(T+273.15))	
6	H0=	0.0019877			x=ln(R)	y=1/(T+273.15)		ln(R)/b-ln(R0)/b=1/(T+273.15)	
7	R0 = exp(-H0/H1)=	0.001311			4.5622627	0.003354		H0=-ln(R)/b	
8	β=1/H1=	3339.0538			4.2341065	0.003256		H1=1/b	
9	Embedded Systems. Real-Time Interfacing to ARM Cortex M Microcontrollers								
10	See Sections 10.2.7 and 10.6								
11						Dissipation Constant			
12	2) Define ADC precision, ADC max, fixed-point format					D(mW/°C)=	2.5		
13	ADC	12	bits			Self-heat error is ΔT=P/D			
14	ADCmax	4095	alternatives						
15	ADCmax	3.3	V						
16	fixed-point	0.01	C						
17									
18	3) Decide the values of Vref, R1, R2, Tmin, and Tmax								
19	V1 (V)	2.5							
20	R1(kΩ)=	470		R2 (kΩ)=	470				
21				R3 (kΩ)=	50				
22	V3(V)=	0.240							
23									
24	4) Build instrumentation amp using INA122 or equivalent (e.g., AD627)								
25	Desired Gain=	7.62							
26		INA122							
27			AD623						



28	desired Rg	76.48	15.12	kΩ					
29	Actual Rg	7.50E+01	15	kΩ					
30	Gain w/actual	7.67	7.67						
31		5+200k/Rg	1+100k/Rg						
32	choose gain			other					
33	Desired gain	AD627	AD623	20					
34	0	1	0	0	<= set 1 to select,		this gain is used=>	7.67	
35									



		T (°C)	RT (kΩ)	V2 (V)	V2-V3 (V)	Vs (V)	ADC	T (fixed)	Power (mW)
57									
58	0	10	173.391	0.674	0.433	3.322	4095	1000	0.0026

2.0: ADC Initializations:

There are many ways to trigger the ADC, but we implemented Processor, Timer, and Continuous trigger initialization methods.

1) Hardware Triggered (GP Timer 0A)

ST7735.c | main.c | startup.s | ADCHWTrigger.c | Timer.c | UART.c

```
74  void ADC0_InitHWTrigger(uint32_t period){
75    PortE_Init();
76    Timer0A_Init(period, 0);
77    volatile uint32_t delay;
78    SYSCTL_RCGCADC_R |= 0x01;      // activate ADC0
79    delay = SYSCTL_RCGCTIMER_R;    // allow time to finish activating
80    delay = SYSCTL_RCGCTIMER_R;    // allow time to finish activating
81
82    ADC0_PC_R = 0x01;          // configure for 125K samples/sec
83    ADC0_SSPRI_R = 0x3210;     // sequencer 0 is highest, sequencer 3 is lowest
84    ADC0_ACTSS_R &= ~0x08;     // disable sample sequencer 3
85    ADC0_EMUX_R = (ADC0_EMUX_R&0xFFFF0FFF)+0x5000; // timer trigger event
86    ADC0_SSMUX3_R = 9;
87    ADC0_SSCTL3_R = 0x06;          // set flag and end
88    ADC0_IM_R |= 0x08;             // enable SS3 interrupts
89    ADC0_ACTSS_R |= 0x08;          // enable sample sequencer 3
90    NVIC_PRI4_R = (NVIC_PRI4_R&0xFFFF00FF)|0x00004000; //priority 2
91    NVIC_EN0_R = 1<<17;              // enable interrupt 17 in NVIC
92    Timer0A_Start();
93    EnableInterrupts();
94  }
95
96  /** ADC0_In() **
97   * Read the ALREADY AVAILABLE ADC0 data.
98   * Input: none
99   * Output: 12-bit result of ADC conversion
100  */
101 static volatile uint32_t ADCvalue[100];
102 uint32_t ADC0_In(void){
103   static int count = 0;
104   if(count < 100){
105     ADCvalue[count] = ADC0_SSFIFO3_R;  // 12-bit result
106     count = count + 1;
107     return ADCvalue[count - 1];
108   } else{
109     return 0x0000; // finished sampling
```

2) Software Triggered (Processor)

ADCSWTrigger.c | ST7735.c | main.c | startup.s | ADCHWTrigger.c | Timer.c | UART.c

```
57  // Sequencer 0 priority: 1st (highest)
58  // Sequencer 1 priority: 2nd
59  // Sequencer 2 priority: 3rd
60  // Sequencer 3 priority: 4th (lowest)
61  // SS3 triggering event: software trigger
62  // SS3 1st sample source: Ain9 (PE4)
63  // SS3 interrupts: enabled but not promoted to controller
64  void ADC0_InitSWTrigger(void){
65    PortE_Init();
66    SYSCTL_RCGCADC_R |= 0x0001;     // 0) activate ADC0
67    while((SYSCTL_PRADC_R&0x0001) != 0x0001){}; // good code, but not yet implemented in simula
68    ADC0_PC_R &= ~0xF;              // 1) clear max sample rate field
69    ADC0_PC_R |= 0x1;               // 2) Sequencer 3 is highest priority
70    ADC0_ACTSS_R &= ~0x0008;        // 3) disable sample sequencer 3
71    ADC0_EMUX_R &= ~0xF000;         // 4) seq3 is software trigger
72    ADC0_SSMUX3_R &= ~0x000F;       // 5) clear SS3 field
73    ADC0_SSMUX3_R += 9;             //    set channel
74    ADC0_SSCTL3_R = 0x0006;         // 6) no TS0 D0, yes IE0 END0
75    ADC0_IM_R &= ~0x0008;           // 7) disable SS3 interrupts
76    ADC0_ACTSS_R |= 0x0008;         // 8) enable sample sequencer 3
77  }
78
79
80  //------------ADC0_In------------
81  // Busy-wait Analog to digital conversion
82  // Input: none
83  // Output: 12-bit result of ADC conversion
84  uint32_t ADC0_In(void){
85    uint32_t result;
86    ADC0_PSSI_R = 0x0008;            // 1) initiate SS3
87    while((ADC0_RIS_R&0x08)==0){};   // 2) wait for conversion done
88      // if you have an A0-A3 revision number, you need to add an 8 usec wait here
89    result = ADC0_SSFIFO3_R&0xFFF;   // 3) read result
90    ADC0_ISC_R = 0x0008;             // 4) acknowledge completion
91    return result;
92  }
```

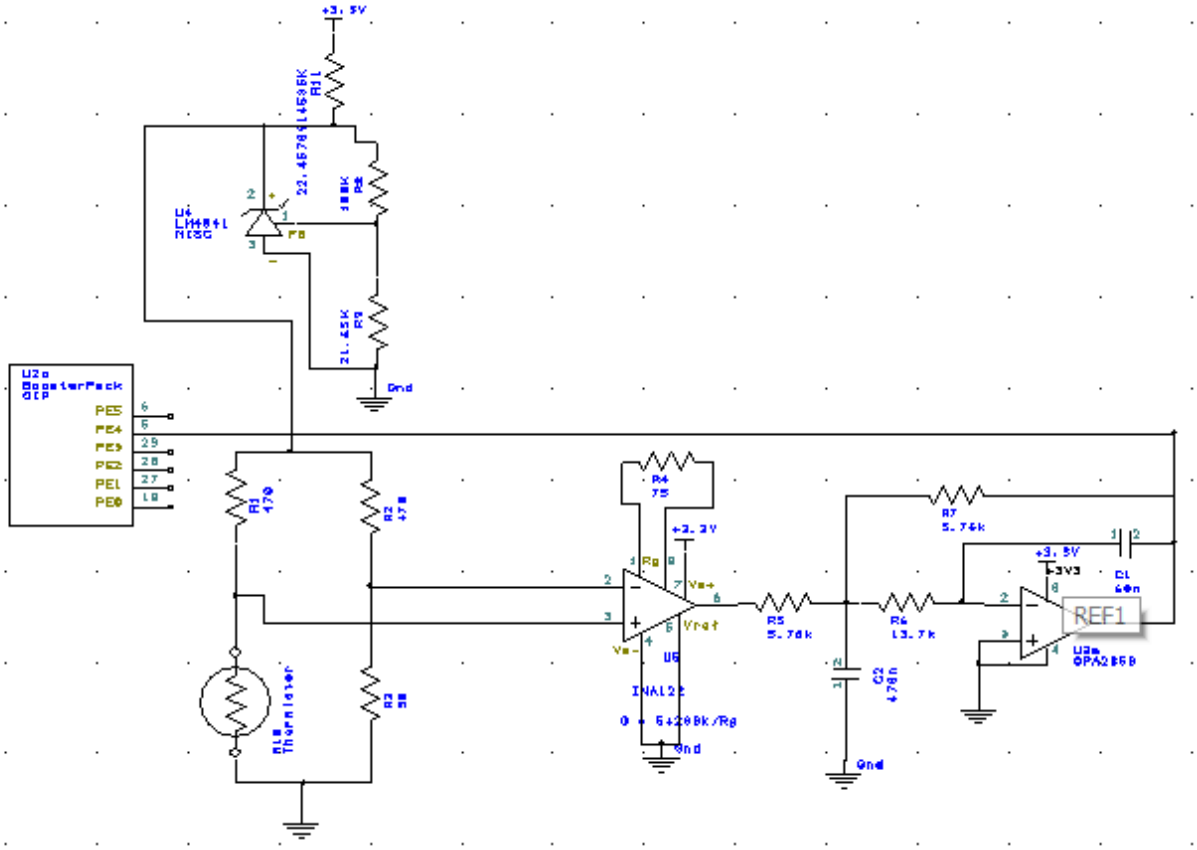
3) Continuous (Automatic and Endless)

ADCContinuous.c | ADCSWTrigger.c | ST7735.c | main.c | startup.s | ADCHWTrigger.c | Timer.c | UART.c

```
 */

#include <stdint.h>
#include "tm4c123gh6pm.h"
static void PortE_Init(void);

/** ADC0_InitContinous() **
 * Initialize ADC0 SEQ3 for continuous sampling
 * Output: none
 */
void ADC0_InitContinous(){
  PortE_Init();
  SYSCTL_RCGCADC_R |= 0x0001;     // 0) activate ADC0
  while((SYSCTL_PRADC_R&0x0001) != 0x0001){}; // good code, but not yet implemented in simulator
  ADC0_PC_R &= ~0xF;              // 1) clear max sample rate field
  ADC0_SSPRI_R = 0x3210;          // 2) sequencer 3 is lowest
  ADC0_ACTSS_R &= ~0x0008;        // 3) disable sample sequencer 3
  ADC0_EMUX_R |=  0xF000;         // 4) seq3 is continous sample
  ADC0_SSMUX3_R &= ~0x000F;       // 5) clear SS3 field
  ADC0_SSMUX3_R += 9;             //    set channel
  ADC0_SSCTL3_R = 0x0004;         // 6) no TS0 D0 END0, yes IE0
  ADC0_IM_R &= ~0x0008;           // 7) disable SS3 interrupts
  ADC0_ACTSS_R |= 0x0008;         // 8) enable sample sequencer 3
}

/** ADC0_In() **
 * Read the ALREADY AVAILABLE ADC0 data.
 * Input: none
 * Output: 12-bit result of ADC conversion
 */
uint32_t ADC0_In(void){
  uint32_t result;
  result = ADC0_SSFIFO3_R&0xFFF;   // 3) read result
  return result;
}

```

3.0: Hardware Design:

Below is the schematic design of our thermistor interface and Butterworth filter.



4.0: Nyquist Sampling Driver:

Below is a snapshot of our main sampling driver. It also includes the preparation questions.

ADCT0ATrigger.c | ST7735.c | main.c | startup.s | ADCHWTrigger.c | Timer.c | UART.c

```
29
30 /** Preperation Questions:
31  * 1) List three ways you could use to initiate the ADC conversion process.
32    - Software Triggered (Processor triggered by default)
33    - Hardware Triggered off a GP Timer (Timer 0A)
34    - Continuous Sampling (continuously samples the ADC)
35    - many more (external GPIO Port, analog comparator, etc..)
36  * 2) What is the way to know when the conversion process has been completed?
37    - Once the conversion process is complete,
38       the ADC_RIS_R (raw interrupt status register) has flags that are set.
39       (assuming the IE0 bit is also set).
40  */
41
42 #include <stdint.h>
43 #include <stdio.h>
44 #include "ADCHWTrigger.h"
45 #include "ADCT0ATrigger.h"
46 #include "uart.h"
47 #include "PLL.h"
48 #include "Timer.h"
49 #include "LCD.h"
50
51 int main(void){
52   uint32_t data;
53   PLL_Init(Bus80MHz);   // 80 MHz
54   LCD_Init(); // screen debugging
55   UART_Init(); // initialize UART device
56   ADC0_InitHWTrigger(TIMER_1000Hz);
57   //ADC0_InitTimer0ATriggerSeq3(9, TIMER_1000Hz);
58   //ADC0_InitSWTrigger();
59   for(int idx = 0; idx < 100; idx++){
60     data = ADC0_In();
61     UART_OutString("\n\rADC data ="); UART_OutUDec(data);
62     printf("sample %d: %x\n", idx, data);
63   }
64
```