

John Duriman

CPE 403

Lab 5

5002373995

Task 00: Execute the provided code, no submission is required.

Task 01: Change the ADC Sequencer to SS2. Turn on the LED at PF2 if the temperature is greater than 75 degF, else PF1 is ON. Use internal temperature sensor for all SS2 sequence

Youtube Link: <https://youtu.be/HhIpT49eDQI>

Modified Code:

```
main.c main.c
1 #include <stdint.h>
2 #include <stdbool.h>
3 #include "inc/hw_memmap.h"
4 #include "inc/hw_types.h"
5 #include "driverlib/debug.h"
6 #include "driverlib/sysctl.h"
7 #include "driverlib/adc.h"
8 #include "inc/hw_gpio.h"
9 #include "inc/tm4c123gh6pm.h"
10 #include "driverlib/interrupt.h"
11 #include "driverlib/gpio.h"
12 #include "driverlib/timer.h"
13 #include "driverlib/pin_map.h"
14 #include "driverlib/rom_map.h"
15
16 #ifdef DEBUG
17 void __error__(char *pcFilename, uint32_t ui32Line){}
18 #endif
19
20
21 int main(void)
22 {
23     uint32_t ui32ADCValue[4];           //Array for storing ADC FIFO data
24     volatile uint32_t ui32TempAvg;      //Temp sensor data
25     volatile uint32_t ui32TempValueC;   //Celsius
26     volatile uint32_t ui32TempValueF;   //Fahrenheit
27
28     //Setup clock
29     SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);
30     SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0); //Enable ADC0
31
32     //Enable GPIO and configure pins as outputs
33     SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
34     GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);
35
36     //Configure ADC sequencer, sample sequencer 2, trigger the sequence at highest priority
37     ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0);
38
39     //Configure all four steps in the ADC sequencer to sequencer 2
40     ADCSequenceStepConfigure(ADC0_BASE, 2, 0, ADC_CTL_TS);
41     ADCSequenceStepConfigure(ADC0_BASE, 2, 1, ADC_CTL_TS);
42     ADCSequenceStepConfigure(ADC0_BASE, 2, 2, ADC_CTL_TS);
43     //Final sequencer step
44     ADCSequenceStepConfigure(ADC0_BASE, 2, 3, ADC_CTL_TS|ADC_CTL_IE|ADC_CTL_END);
45 }
```

```

36 //Configure ADC sequencer, sample sequencer 2, trigger the sequence at highest priority
37 ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0);
38
39 //Configure all four steps in the ADC sequencer to sequencer 2
40 ADCSequenceStepConfigure(ADC0_BASE, 2, 0, ADC_CTL_TS);
41 ADCSequenceStepConfigure(ADC0_BASE, 2, 1, ADC_CTL_TS);
42 ADCSequenceStepConfigure(ADC0_BASE, 2, 2, ADC_CTL_TS);
43 //Final sequencer step
44 ADCSequenceStepConfigure(ADC0_BASE, 2, 3, ADC_CTL_TS|ADC_CTL_IE|ADC_CTL_END);
45
46 //Enable ADC Sequencer 2
47 ADCSequenceEnable(ADC0_BASE, 2);
48
49 while(1)
50 {
51     //Clear ADC interrupt status flag
52     ADCIntClear(ADC0_BASE, 2);
53     //Trigger ADC conversion
54     ADCProcessorTrigger(ADC0_BASE, 2);
55
56     while(!ADCIntStatus(ADC0_BASE, 2, false)){
57
58         //Read ADC value
59         ADCSequenceDataGet(ADC0_BASE, 2, ui32ADC0Value);
60
61         //Calculate average of the temperature sensor data
62         ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value[3] + 2)/4;
63         ui32TempValueC = (1475 - ((2475 * ui32TempAvg) / 4096))/10;
64         ui32TempValueF = ((ui32TempValueC * 9) + 160) / 5;
65
66         //Make all pins low
67         GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
68
69         //If temperature is >75, set PF2 to HIGH. Else set PF1 to HIGH
70         if(ui32TempValueF > 75)
71         {
72             GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);
73
74         }
75         else
76         {
77             GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1, 2);
78
79         }
80     }
81 }
82

```

Task 02: Introduce hardware averaging to 32. Using the timer TIMER1A conduct an ADC conversion on overflow every 0.5 sec. Use the Timer1A interrupt.

Youtube Link: <https://youtu.be/HhIpT49eDQI>

Modified Code:

```
main.c tm4c123gh6pm_startup_ccs.c
1 #include <stdint.h>
2 #include <stdbool.h>
3 #include "inc/hw_memmap.h"
4 #include "inc/hw_types.h"
5 #include "driverlib/debug.h"
6 #include "driverlib/sysctl.h"
7 #include "driverlib/adc.h"
8 #include "inc/hw_gpio.h"
9 #include "inc/tm4c123gh6pm.h"
10 #include "driverlib/interrupt.h"
11 #include "driverlib/gpio.h"
12 #include "driverlib/timer.h"
13 #include "driverlib/pin_map.h"
14 #include "driverlib/rom_map.h"
15
16 #ifdef DEBUG
17 void __error__(char *pcFilename, uint32_t ui32Line){}
18 #endif
19
20
21 int main(void)
22 {
23     //Setup clock and ADC
24     SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);
25     SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0); //Enable ADC0
26     ADCHardwareOversampleConfigure(ADC0_BASE, 32);
27
28     //Enable GPIO and configure pins as outputs
29     SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
30     GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);
31
32     //Configure ADC sequencer, sample sequencer 2, trigger the sequence at highest priority
33     ADCSequenceConfigure(ADC0_BASE, 2, ADC_TRIGGER_PROCESSOR, 0);
34
35     //Configure all four steps in the ADC sequencer to sequencer 2
36     ADCSequenceStepConfigure(ADC0_BASE, 2, 0, ADC_CTL_TS);
37     ADCSequenceStepConfigure(ADC0_BASE, 2, 1, ADC_CTL_TS);
38     ADCSequenceStepConfigure(ADC0_BASE, 2, 2, ADC_CTL_TS);
39     //Final sequencer step
40     ADCSequenceStepConfigure(ADC0_BASE, 2, 3, ADC_CTL_TS|ADC_CTL_IE|ADC_CTL_END);
41
42     //Enable ADC Sequencer 2
43     ADCSequenceEnable(ADC0_BASE, 2);
44
45     //Setup Timer 1
46     SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER1);
47     TimerConfigure(TIMER1_BASE, TIMER_CFG_PERIODIC);
48     TimerLoadSet(TIMER1_BASE, TIMER_A, (SysCtlClockGet()/2)-1);
49     TimerEnable(TIMER1_BASE, TIMER_A);
50
51     //Enable interrupts
52     IntEnable(INT_TIMER1A);
53     TimerIntEnable(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
54     IntMasterEnable();
55
56     while(1){}
57 }
58
```

```

59
60 void Timer1IntHandler(void)
61 {
62     uint32_t ui32ADC0Value[4];           //Array for storing ADC FIFO data
63     volatile uint32_t ui32TempAvg;       //Temp sensor data
64     volatile uint32_t ui32TempValueC;    //Celsius
65     volatile uint32_t ui32TempValueF;    //Fahrenheit
66
67     //Clear interrupt
68     TimerIntClear(TIMER1_BASE, TIMER_TIMA_TIMEOUT);
69
70     //Clear ADC interrupt status flag
71     ADCIntClear(ADC0_BASE, 2);
72     //Trigger ADC conversion
73     ADCProcessorTrigger(ADC0_BASE, 2);
74
75     while(!ADCIntStatus(ADC0_BASE, 2, false)){
76
77         //Read ADC value
78         ADCSequenceDataGet(ADC0_BASE, 2, ui32ADC0Value);
79
80         //Calculate average of the temperature sensor data
81         ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value[3] + 2)/4;
82         ui32TempValueC = (1475 - ((2475 * ui32TempAvg) / 4096))/10;
83         ui32TempValueF = ((ui32TempValueC * 9) + 160) / 5;
84
85         //Make all pins low
86         GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
87
88         //If temperature is >75, set PF2 to HIGH. Else set PF1 to HIGH
89         if(ui32TempValueF > 75)
90         {
91             GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);
92         }
93         else
94         {
95             GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1, 2);
96         }
97     }
98

```