**Date Submitted: 10.12.2018**

**Task 00: *No submission***

**------------------------------------------------------------------------------------**

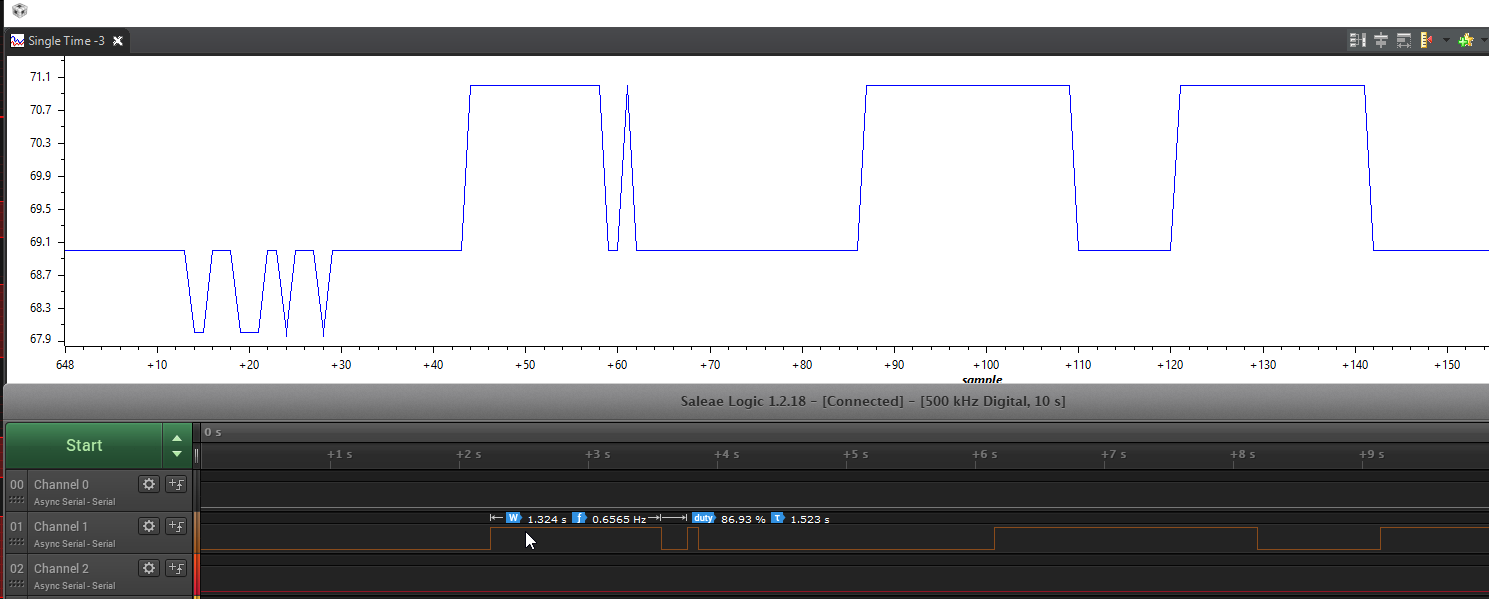
**Task 01:** Change the ADC Sequencer to SS3. Turn on the LED at PF2 if the temperature is greater

that 72°F. Use internal temperature sensor for all SS2 sequence. Display the temperature in

the built-in graph tools.

**Youtube Link**: <https://youtu.be/DIaz1m-nf4c> for T1&2

For task 1 I had to lower the variable threshold because of my room temperature. While it said to use SS3 after 10 hours of going through datasheets I was not able to understand why SS3 would not work, so I used SS2. The top graph is the CCS graphing tool displaying temp the lower graph is the LED coming on after the threshold is reached.



*//ADC Sequencer configure switched to another SS2 per instructions worked on 1 as well*

ROM\_ADCSequenceConfigure(ADC0\_BASE, 2, ADC\_TRIGGER\_PROCESSOR, 0);

ROM\_ADCSequenceStepConfigure(ADC0\_BASE, 2, 0, ADC\_CTL\_TS);

ROM\_ADCSequenceStepConfigure(ADC0\_BASE, 2, 1, ADC\_CTL\_TS);

ROM\_ADCSequenceStepConfigure(ADC0\_BASE, 2, 2, ADC\_CTL\_TS);

//Red if(t>67)

**if**(ui32TempValueF > 67)

ROM\_GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 4);

**else**

ROM\_GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 0);

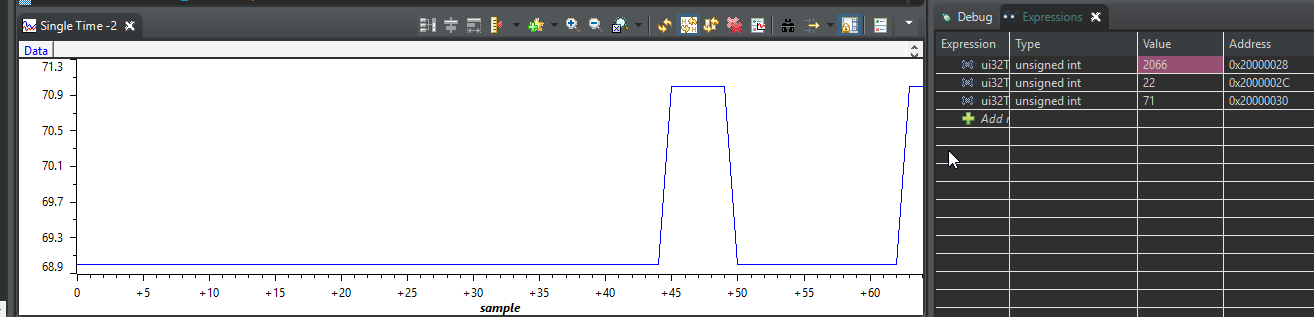
**Task 02:** Introduce hardware averaging to 32. Using the timer TIMER1A conduct an ADC conversion

on overflow every 0.5 sec. Use the Timer1A interrupt. Display the temperature in the

built-in graph tool.

**Youtube Link:**

**For task 2 the code that updated the variables was moved to an Interrupt handler which so that the variable would only be updated at the time of the interrupt every 0.5 seconds.**



**Added handler with code moved from main**

void Timer1AIntHandler(void)

{

*//moved everything inside while loop from main*

uint32\_t ui32ADC0Value[4];

**volatile** uint32\_t ui32TempAvg;

**volatile** uint32\_t ui32TempValueC;

**volatile** uint32\_t ui32TempValueF;

TimerIntClear(TIMER1\_BASE, TIMER\_TIMA\_TIMEOUT);

ROM\_ADCIntClear(ADC0\_BASE, 1);

ROM\_ADCProcessorTrigger(ADC0\_BASE, 1);

**while**(!ROM\_ADCIntStatus(ADC0\_BASE, 1, false))

{

}

ROM\_ADCSequenceDataGet(ADC0\_BASE, 1, ui32ADC0Value);

ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value[3] + 2)/4;

ui32TempValueC = (1475 - ((2475 \* ui32TempAvg)) / 4096)/10;

ui32TempValueF = ((ui32TempValueC \* 9) + 160) / 5;

*// turn on blue led if temperature > 72 degrees, off otherwise*

**if**(ui32TempValueF > 72)

ROM\_GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 4);

**else**

ROM\_GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 0);

}