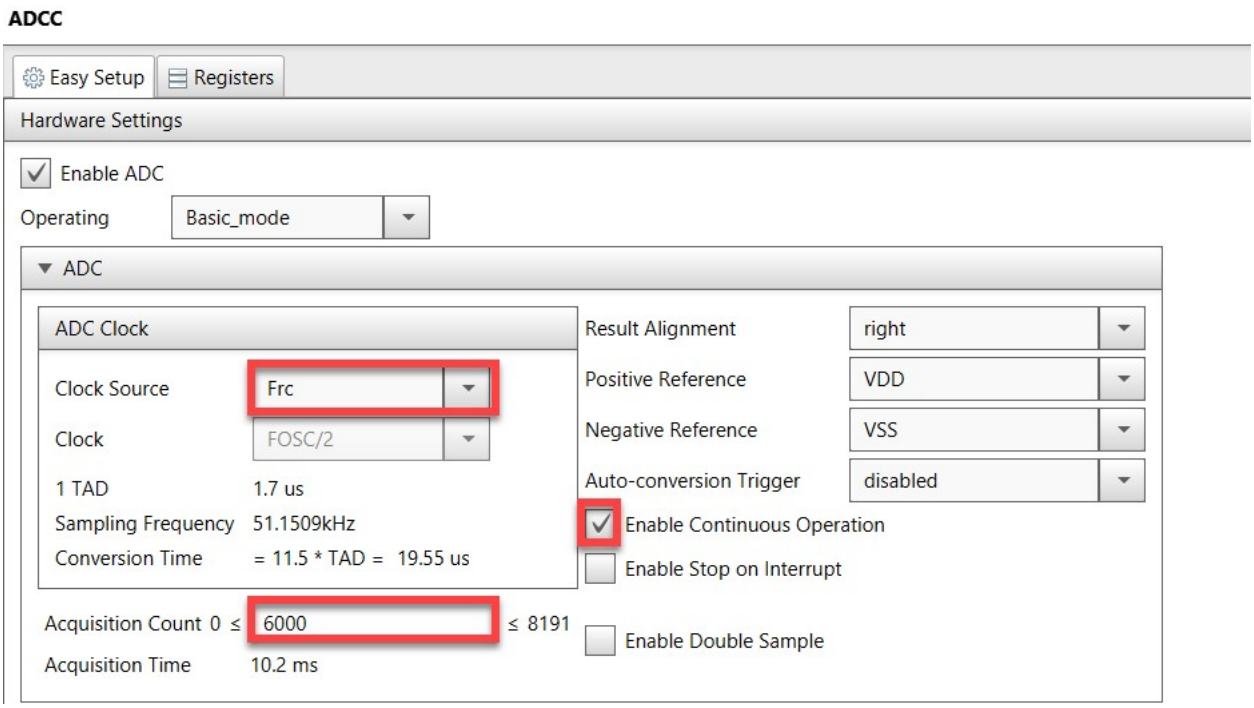
Lab 4: Wake Up For Interrupt

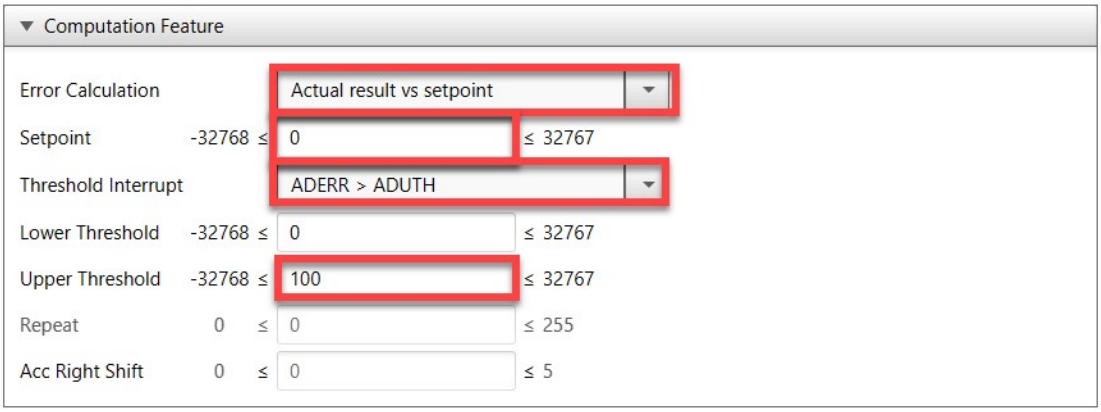
This section will show you how to run the ADCC in continuous operation mode. Through this your ADCC will continuously sample once initially started in software. Once it is started, you can put the device to sleep. We will also set the ADCC up to generate an interrupt whenever the noise level exceeds a certain threshold. The device will then be able to wake itself from sleep and print out the noise level that caused it to wake up.

1. **Configure ADCC:**
    a. Set Clock Source to FRC
    b. Enable Continuous Operation
    c. Set Acquisition Count to 6000 *(You should now see the field **Acquisition Time** reflecting around 10 ms. This provides a 10 ms delay in-between each ADCC sample reading. This will help minimize the ADCC interrupting multiple times when responding to the same noise outburst.)*



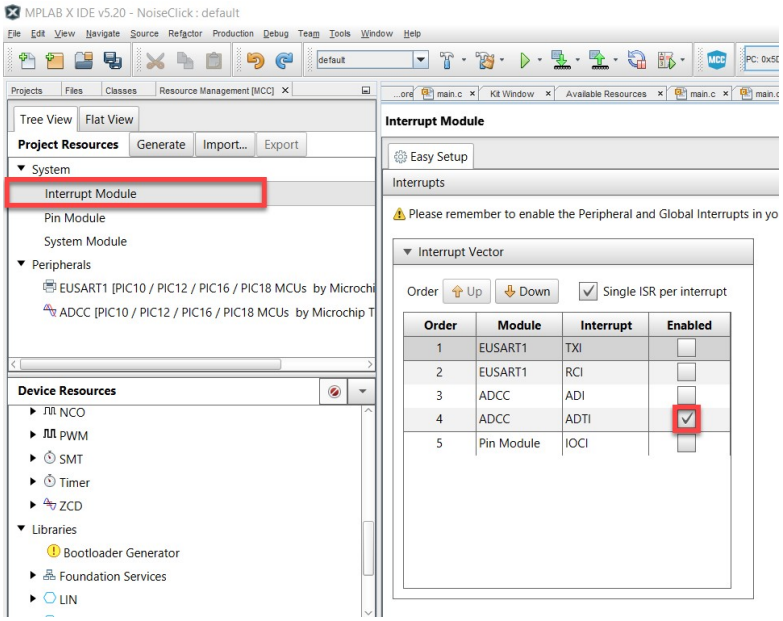
2. **Enable Thresholding**
    a. Set Error Calculation to **Actual result vs setpoint**
    b. Change Setpoint to 0
    c. Change Threshold Interrupt to **ADERR > ADUTH**
    d. Change Upper Threshold to 100

*The **ADERR** register will be the result of the **ADC Actual Result – the Setpoint**. Since we set the setpoint to 0, this value will simply be the Actual Result of the ADCC reading. The ADERR register will then be compared to our upper threshold which we set to **100**. If our ADC result exceeds the value of 100 then we need to generate an interrupt. The correct configuration should look like the image below.*

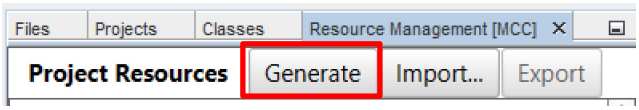


3. **Enable Interrupts:**

4. Ensure that Interrupts are enabled in the Interrupt Module



5. **Generate Code:** Click on generate code from the top of the MCC window



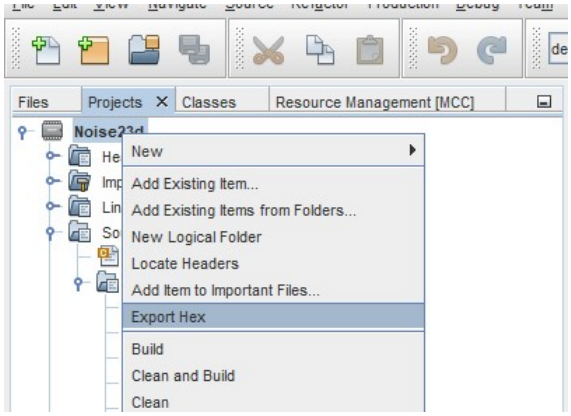
6. **Edit main.c:** Now go back to Main.c. Make the following changes:

```
50  void main(void)
51  {
52      // Initialize the device
53      SYSTEM_Initialize();
54
55      // If using interrupts in PIC18 High/Low Priority Mode you need to enable the Global High and Low Interrupts
56      // If using interrupts in PIC Mid-Range Compatibility Mode you need to enable the Global and Peripheral Interrupts
57      // Use the following macros to:
58
59      // Enable the Global Interrupts
60      INTERRUPT_GlobalInterruptEnable();
61
62      // Disable the Global Interrupts
63      INTERRUPT_GlobalInterruptDisable();
64
65      // Enable the Peripheral Interrupts
66      //INTERRUPT_PeripheralInterruptEnable();
67
68      // Disable the Peripheral Interrupts
69      //INTERRUPT_PeripheralInterruptDisable();
70      ADCC_StartConversion(sensor);
71      while (1)
72      {
73          SLEEP();
74          printf("\n %i", ADCC_GetConversionResult());
75          __delay_ms(100);
76      }
77  }
```

As mentioned previously, in this part of the lab we only want to start the ADCC conversion once. After it is started in software, the ADCC will run continuously. Therefore the device can be put into sleep mode in the while(1) loop. However, we need to enable global and peripheral interrupts so that our ADCC can wake the device once the threshold is exceeded.

Once the device is woken up from the threshold being exceeded, we want to print out the ADCC conversion result. We will add another delay to ensure that everything is successfully printed out to the serial terminal. The device will then go to sleep until the next interrupt is generated.

7. **Compile and Program:** Go to the top level of your project and right click. In the dropdown menu you will see **Export Hex**. It will then ask you to provide a file name. Do this and click save.



8. **Connect and See Output:** Navigate back to CoolTerm and click connect if the settings have not changed. Now you should only see values being printed out when you tap on the noise click or make a loud enough noise into the microphone.