ECEN 5823
BLE Health Temp Service Assignment
Fall 2017

Objective: To take the temperature measured by the Si7021 and communicate it via BLE to the Silicon Labs' BlueGecko iPhone or Android phone app.

Note: This assignment will begin with the completed I2C temp sensor assignment.

Due: Sunday, October 22nd, 2017 at 11:59pm

Instructions:

1. Make any changes required to the I2C temp sensor assignment.

2. Connect the STK6101C extension board to the main development kit board.

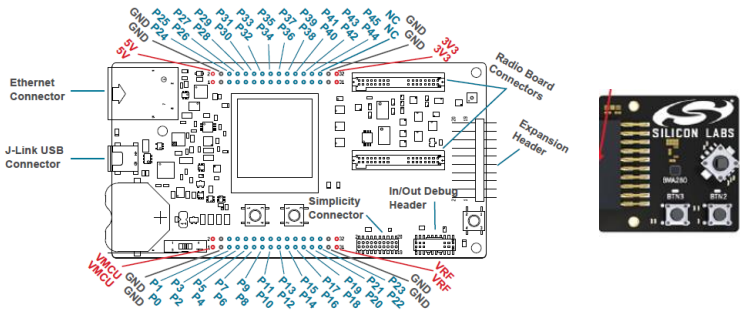

Figure 3.1. Mainboard Connector Layout

3. Program the ADC to sense / interrupt the Blue Gecko when an action occurs on the extension board's analog joystick.
    a. You must determine which pin the Joy Stick output is routed to the BGM121. One method is to trace the output of the Joy Stick to the extension board connector through the extension board of the main development kit board to the Radio Board connectors and finally to the BGM121 to determine which pin to use as the analog input.
    b. Program the designated analog input as an analog input pin

4. The ADC interrupt handler should be able to make the following distinctions when an event occurs:
    a. If Joy Stick is pressed away (north), enable the BMA280 (put in normal mode)
    b. If Joy Stick is pressed toward you (south), disable the BMA280 (suspend mode)
    c. If Joy Stick is pressed to the right (east), temperature set point is increased by 5 degrees C

d. If Joy Stick is pressed to the left (west), temperature set point is decreased by 5 degrees C
e. Note:
    i. Circuit should be software de-bounced so that only one increment or decrement of the on-time occurs per press / pulse of the Joy Stick.

5. BMA280 functionality:
    a. Upon power on reset or the Blue Gecko reset, the BMA280 should be in SUSPEND mode
    b. Single tap should enable the I2C sensor, Load Power Management ON, and no affect to LED1
    c. Double tap should disable the I2C temp sensor, Load Power Management OFF, and no effect on LED1

6. ADC's main parameters to enable grading are:
    a. 12-bit resolution
    b. No oversampling
    c. Continually sample the Joy Stick input 200 samples per second (200 Hz)
    d. 32 clocks for acquisition time
    e. ADC bias settings to lowest possible setting

7. BMA280 settings should be initialized to:
    a. Range +/- 4g
    b. Bandwidth 125Hz
    c. Tap quiet 30mS
    d. Tap samples 4
    e. Tap duration 200mS
    f. Tap shock 50mS
    g. Tap threshold 250mg

8. LETIMER0 should be set to the following conditions at startup / reset.
    a. Period = 1.75 seconds
    b. No period, so no need to have the LETIMER0 to interrupt twice per period
    c. During the LETIMER0 period interrupt, it will request, receive, and process the temperature reading from the Si7021

9. Si7021 I2C temp sensor
    a. It should be running at the lowest energy possible while enabled and while taking temperature measurements
    b. Temperature measurements should be 14-bit and calculated in degrees C
    c. There should be a define statement that sets the default temperature in degrees C to indicate whether the temperature is too low
        i. The default temperature should be 15C

d. If the temperature read from the Si7021 is below the set temperature, LED1 is latched on until cleared by pressing the joy stick button down
e. The temperature set point should increase by 5 degrees C every time the joy stick is pressed to the right
f. The temperature set point should decrease by 5 degrees C every time the joy stick is pressed to the left
g. If LED1 is turned off by pressing the joy stick button down and the temperature is still below the current set point, LED1 should turn on at the next temperature measurement
h. The temperature set point is not reset when the joy stick button is pressed down. The only function pressing the joy stick down should be to turn off LED1.

10. The Blue Gecko should be running at the lowest possible energy state while the system is waiting for an input from the Joy Stick.

11. LED0 is not used in this assignment

12. Delete the following lines of code from your project if you had not deleted them previously. You can find them in the main.c program.

```
      /* Events related to OTA upgrading
         ----------------------------------------------------------------------
--------- */

      /* Check if the user-type OTA Control Characteristic was written.
       * If ota_control was written, boot the device into Device Firmware
Upgrade (DFU) mode. */
//      case gecko_evt_gatt_server_user_write_request_id:

//        if (evt->data.evt_gatt_server_user_write_request.characteristic ==
gattdb_ota_control) {
          /* Set flag to enter to OTA mode */
//          boot_to_dfu = 1;
          /* Send response to Write Request */
//          gecko_cmd_gatt_server_send_user_write_response(
//            evt->data.evt_gatt_server_user_write_request.connection,
//            gattdb_ota_control,
//            bg_err_success);

          /* Close connection to enter to DFU OTA mode */
//          gecko_cmd_endpoint_close(evt-
>data.evt_gatt_server_user_write_request.connection);
//        }
//        break;

//      default:
//        break;
      //    }
```

13. Uncommit the following lines of code found in main. C

```
    /* Event pointer for handling events */
//    struct gecko_cmd_packet* evt;

    /* Check for stack event. */
//    evt = gecko_wait_event();

    /* Handle events */
//    switch (BGLIB_MSG_ID(evt->header)) {
      /* This boot event is generated when the system boots up after
reset.
       * Here the system is set to start advertising immediately after
boot procedure. */
//      case gecko_evt_system_boot_id:

        /* Set advertising parameters. 100ms advertisement interval. All
channels used.
         * The first two parameters are minimum and maximum advertising
interval, both in
         * units of (milliseconds * 1.6). The third parameter '7' sets
advertising on all channels. */
//        gecko_cmd_le_gap_set_adv_parameters(160, 160, 7);

        /* Start general advertising and enable connections. */
//       gecko_cmd_le_gap_set_mode(le_gap_general_discoverable,
le_gap_undirected_connectable);
//        break;

//      case gecko_evt_le_connection_closed_id:

        /* Check if need to boot to dfu mode */
//        if (boot_to_dfu) {
          /* Enter to DFU OTA mode */
//          gecko_cmd_system_reset(2);
//        } else {
          /* Restart advertising after client has disconnected */
//          gecko_cmd_le_gap_set_mode(le_gap_general_discoverable,
le_gap_undirected_connectable);
//        }
//        break;
```

14. The Bluetooth stack requires the HFXO, and it must be set to auto start. The default software should enable these if you had not deleted them. If that is the case, go and commit out disabling the HFXO and enabling the LFXO.

```
    // By default, HFRCO is enabled  cmuHFRCOFreq_19M0Hz
//    CMU_HFRCOBandSet(cmuHFRCOFreq_19M0Hz);                    // Set
HFRCO frequency
   // CMU_OscillatorEnable(cmuOsc_HFRCO, true, true);       // Enable HFRC0

    /* Set auto start and HF behaviour */
//    CMU_HFXOAutostartEnable(0, true, true);               // HFXO auto
start must be disabled before switching to HFRCO
//    CMU_ClockSelectSet(cmuClock_HF, cmuSelect_HFRCO);
```

```
//    CMU_OscillatorEnable(cmuOsc_HFXO, false, false);      // Disable HFXO
to save energy after HFRCO has been enabled

      // Enable the HFPERCLK for desired peripherals such as ADC
//    CMU_ClockSelectSet(cmuClock_HFPER, cmuSelect_HFRCO);
   // CMU_ClockEnable(cmuClock_HFPER, true);
```

15. The Bluetooth stack also requires the peripherals that you may have turned off to save energy and that we had not been using. Please re-enable or uncommit these peripheral clock enables depending how you disable them.

```
      // Disable peripherals that the default initialization routine enables
//    CMU_ClockEnable(cmuClock_GPCRC, false);
//    CMU_ClockEnable(cmuClock_LDMA, false);
//    CMU_ClockEnable(cmuClock_PRS, false);
   // CMU_ClockEnable(cmuClock_RTCC, false);
```

16. The Bluetooth stack needs to control the sleep and wake up cycle to get communicate when required via the BLE radio. For now, keep your sleep routine, but make it an empty routine by committing out the enter to sleep routines.

```
//    if (lowest_energy_mode[EM0] > 0){}
//    else if (lowest_energy_mode[EM1] > 0) EMU_EnterEM1();
//    else if (lowest_energy_mode[EM2] > 0) EMU_EnterEM2(true);
//    else if (lowest_energy_mode[EM3] > 0) EMU_EnterEM3(true);
      //    else EMU_EnterEM3(true);
```

17. Download and install the Silicon Labs' Blue Gecko app to your iPhone or Android phone

18. Now you are ready to begin putting your first Bluetooth Service together
    a. Add the Bluetooth Health Temperature Service to your application via the .isc file
    b. Your application should update the temperature and Bluetooth Health Temperature Service when the LETIMER0 period has expired

19. Now you will set up your second Bluetooth Service
    a. Add the Bluetooth TX_Power service to your application via the .isc file
    b. The TX_Power characteristic is automatically updated when you modify TX_Power

20. To maximize energy savings, the Bluetooth application should change its advertising, connection interval, and slave interval to what is appropriate to the application.
    a. Set the Advertising min and max to 500mS
    b. Set Connection Interval minimum and maximum to 75mS
    c. Set the Slave latency to enable it to be off the "air" up to 375mS

21. To further maximize energy savings, the application should automatically adjust its transmit power based on the proximity of the master / client, the phone. Please use the following settings:
    a. If rssi > -35db, set tx_power to BGM121 TX Min
    b. If -35db > rssi > -45db, set tx_power to -20db
    c. If -45db > rssi > -55db, set tx_power to -15db
    d. If -55db > rssi > -65 db, set tx_power to -5db
    e. If -65db > rssi > -75db, set tx_power to 0db
    f. If -75db > rssi > -85db, set tx_power to 5db
    g. If rssi < -85db, set tx_power to BGM121 TX Max
    h. Upon reset or a Bluetooth connection disconnect, TX Power should be set to 0db

Questions:

In a separate document to be placed in the drop box with the program code, please answer the following questions:

NOTE: All average currents should be taken at a time scale of 250uS/div.

1. Upon Power Up, after reset, what is the period between BLE advertisements of your application?

2. On your Blue Gecko phone application, open up the Bluetooth Browser. What is the TX_Power of your BLE device from the application? With it disconnected and advertising:
    a. What is the average current over a 10 second span of time?
    b. Reset the energy score and let run for 60 seconds. What is the energy score?

3. Set your phone approximately 1 foot from the Blue Gecko dev kit. Connect to your board via the Blue Gecko phone application through the Health Thermometer / other app. Enable the BMA280 and with a single tap turn on the Si7021. After you have confirmed that temperature values are going to the phone, put the BMA280 back into SUSPEND mode while the Si7021 is still taking measurements.

    a. After connecting and the connections has completed its set up, what is the maximum time between BLE radio transmissions?
    b. After connecting and the connections has completed its set up, what is the minimum time between BLE radio transmissions?
    c. From the two questions above, what is the effective slave latency?
    d. Measure current of the smallest BLE radio transmission wile connected
    e. What is the average current over a 10 second span of time?
    f. Reset the energy score and let run for 60 seconds. What is the energy score?

4. With the phone still connected, walk and set your phone approximately 10 feet from the Blue Gecko dev kit. Connect to your board via the Blue Gecko phone application through the Health Thermometer / other app.

    a. After connecting and the connections has completed its set up, what is the maximum time between BLE radio transmissions?
    b. After connecting and the connections has completed its set up, what is the minimum time between BLE radio transmissions?
    c. From the two questions above, what is the effective slave latency?
    d. Measure current of the smallest BLE radio transmission wile connected
    e. What is the average current over a 10 second span of time?
    f. Reset the energy score and let run for 60 seconds. What is the energy score?

5. With the phone still connected, walk and set your phone approximately 20 feet from the Blue Gecko dev kit. Connect to your board via the Blue Gecko phone application through the Health Thermometer / other app.

    a. After connecting and the connections has completed its set up, what is the maximum time between BLE radio transmissions?
    b. After connecting and the connections has completed its set up, what is the minimum time between BLE radio transmissions?
    c. From the two questions above, what is the effective slave latency?
    d. Measure current of the smallest BLE radio transmission wile connected
    e. What is the average current over a 10 second span of time?
    f. Reset the energy score and let run for 60 seconds. What is the energy score?

6. While staying at 20 feet away, disconnect from the phone the Health Temperature application and using the Bluetooth Browser, what is the TX_Power being displayed of your device? With it disconnected and advertising:
    a. What is the average current over a 10 second span of time?
    b. Reset the energy score and let run for 60 seconds. What is the energy score?

Deliverables:

1. One document that provides the answers to I2C temp sensor Assignment.

2. The completed program project or required files to enable the code to be ran on the instructing team's computer for grading.