Brian Lopez
Phillip Sortomme
Venkatesan Muthukumar
CPE403-1001
12 December 2018

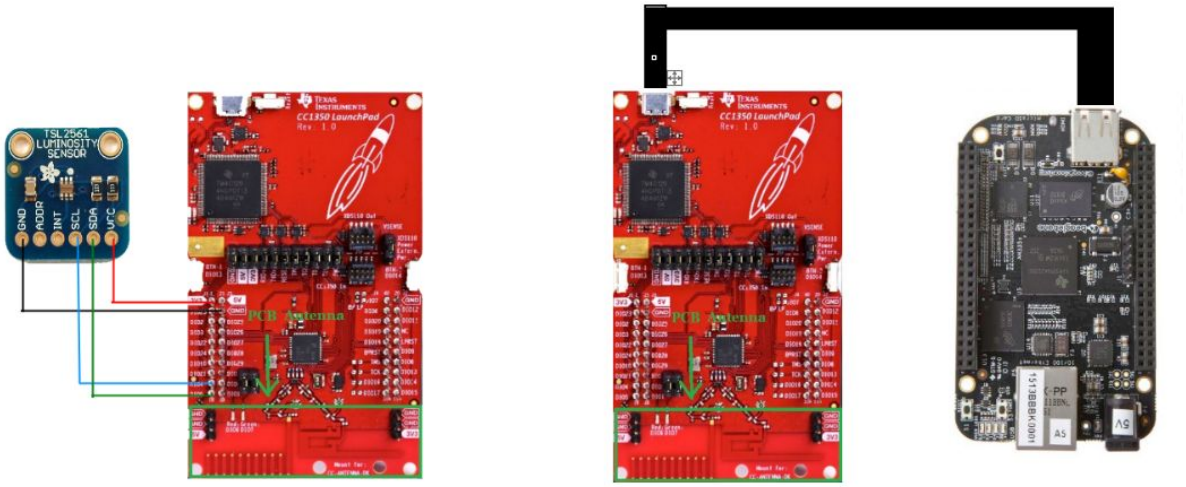
CC1350 Communication with Beaglebone black

PROBLEM STATEMENT:

We needed to establish a communication between two CC1350 microcontrollers, where one of the modules is connected to a Beaglebone Black to display the data on a webpage. The TI 15.4-Stack project was used for interfacing between both the CC1350s and the collector node with the Beaglebone Black. The end goal is to modify the Linux Gateway project to show the values of any external sensor instead of the internal temperature.

We were able to configure the sensor node to a TSL2591 lux sensor via I2C communication to display the lux value on both the sensor and collector nodes. After that, the BeagleBone Black was successfully configured to run the TI-Stack application. Once we connected the collector node to the BBB, we started the web page with the data and were able to see the internal temperature of the sensor node displayed on the page. **However**, we were not able to combine the two parts and display the lux values onto the page. When we combined the two and tried sending over the lux value, the only value obtained on the page is 20 no matter what we send or how we send it.

Diagram of our layout:



PRE-REQUISITES:

Components:

- 2x CC1350: Microcontrollers from Texas Instrument that are capable of RF communication. One of them will interface with the sensor and then send the data wirelessly to the other CC1350, which is sending the data to the BBB
- BeagleBone Black: The BeagleBone Black contains a 1GHz ARM processor that is able to run a Linux environment. From the Linux environment, it is able to collect that data from the collector CC1350 and create a webpage in the network to display the information. The data can then be accessed on any computer that is connected to the same network.
- TSL2591: Lux sensor that can be interfaced via I2C communication. This measures how much light is hitting the sensor and outputs them between two different register values. The raw value is given to the microcontroller that interfaces with it.

Software:

- Code Composer Studio: This is used to be able to program both of the CC1350s with our modified code. Only the sensor node was needed to be changed with Code Composer Studio
- Uniflash: Uniflash is used to load prebuild files into the CC1350, more specifically the collector node.
- Putty: Used to establish a serial communication between the CC1350s and the BeagleBone.

IMPLEMENTATION DETAILS:

Implementing I2C to the sensor node:

1. The first thing that is needed is to add the I2C library files into the sensor.c file:

```
// added to add I2C communication
#include <ti/drivers/I2C.h>
#include <ti/drivers/i2c/i2CCC26XX.h>
#include "board.h"
// Not needed for I2C, only for debugging
// Used to display text/values through UART
#include "board_lcd.h"
```

2. After that, everything else was done inside the readSensors function. The next step is to declare some variables and enable I2C through pins 4 and 5 of the CC1350.

```
uint8_t         txBuffer[5]; // holds the commands being sent
uint8_t         rxBuffer[5]; // holds anything sent from sensor
I2C_Handle      i2c;
I2C_Params      i2cParams;
I2C_Transaction i2cTransaction;

I2C_init();          // Configure I2C on pins 4 and 5
I2C_Params_init(&i2cParams); // set up the parameters
i2c = I2C_open(Board_I2C_TMP, &i2cParams); // establish I2C
```

We also need to assign the I2C class parameters to match the declared variables.

```
i2cTransaction.slaveAddress = 0x29;
i2cTransaction.readBuf = rxBuffer;
i2cTransaction.writeBuf = txBuffer;
```

3. After this the TSL2591 needs to be configured to keep its values. Here we need to add values into the txBuffer to send everything. The writecount variable is updated to 2 to indicate that we are sending two bytes of data. We then send everything to configure the lux sensor in two different times:

```
i2cTransaction.writeCount = 2;
txBuffer[0] = 0xA1; // Register control | Command bit
txBuffer[1] = 0x10;
i2cTransaction.readCount = 0;
if (I2C_transfer(i2c, &i2cTransaction)) { // I2C_transfer sends the data
    // sends the two registers to lux sensor
}


txBuffer[0] = 0xA0; // Register Enable | Command bit
txBuffer[1] = 0x8B; //enable poweron, aen, aien, npien
if (I2C_transfer(i2c, &i2cTransaction)) { // I2C_transfer sends the data
    // sends next two registers to finish configuration
}
```

4. Now we can finally obtain values of the lux sensor. The next two bytes of data we send are to obtain the high and low of the raw values of the sensor. We need to indicate that we are reading in two values as well to populate the rxBuffer variable. Once that's done, we obtain both values from the buffer and create one lux value from it. After we get that, we close the I2C channel and assign the obtained value to be sent to the collector node:

```
uint32_t x1;
i2cTransaction.writeCount = 2;
txBuffer[0] = 0xB4; // Command bit | C0DataH
txBuffer[1] = 0xB5; // Command bit | C0DataL
i2cTransaction.readCount = 2; // indicate that we are reading 2 values
if (I2C_transfer(i2c, &i2cTransaction)) {
    x1 = rxBuffer[0];  // get C0DataH
    x1 <<= 16;
    x1 |= rxBuffer[1]; // get C0DataL

    x1 /= 2500;    // no lux calculation, just divide the raw value
}

/* Deinitialized I2C */
I2C_close(i2c);
// Display the lux value through UART, only for debugging
LCD_WRITE_STRING_VALUE("Lux is: ", (uint16_t) x1, 10, 5);
tempSensor.objectTemp = (uint16_t) x1;
```

5. For the Collector node, that only thing needed was to flash the build given with Uniflash. This is the same thing as running the default project from the collector project from the CC1350 SDK.

6. Finally, the BeagleBone Black was flashed with a version of Linux, Arago, and then the TI Stack gateway was installed in it. This just meant sending a zipped file into the BBB and extracting it inside. After that, the project is ran from the terminal which checks to see if the collector node is connected to it. This was done just from following the directions of the instructions, so the details are omitted from this report.

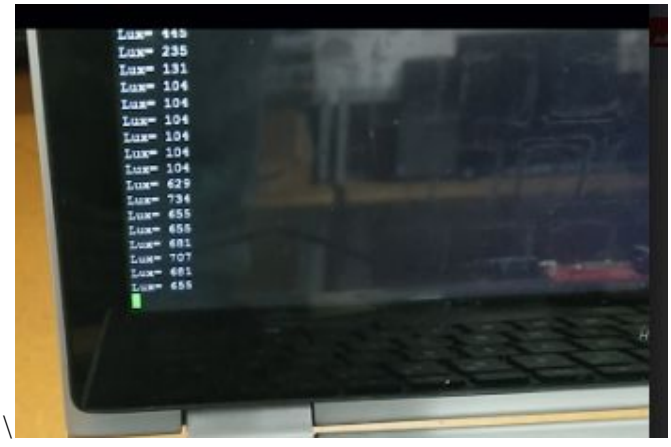
OUTCOMES, RESULTS AND CONCLUSIONS:

Results from I2C Transmission:

Video link: https://youtu.be/NHI1rWbbn-k

We were able to send the lux sensor's values from the sensor node to the collector node. We did not take good screenshots from here, but here are stills from the video that shows the serial communication of both the sensor and collector nodes:

Serial port from the sensor node

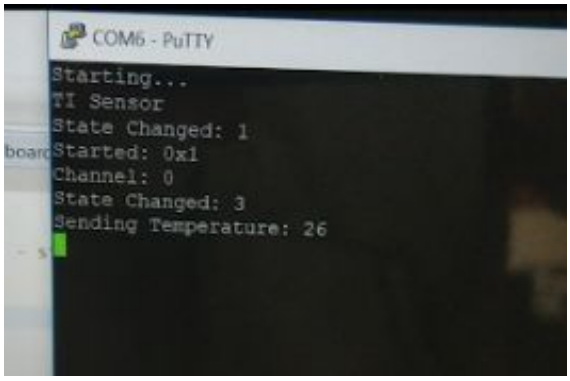


\
Serial port from the collector node at almost the same time

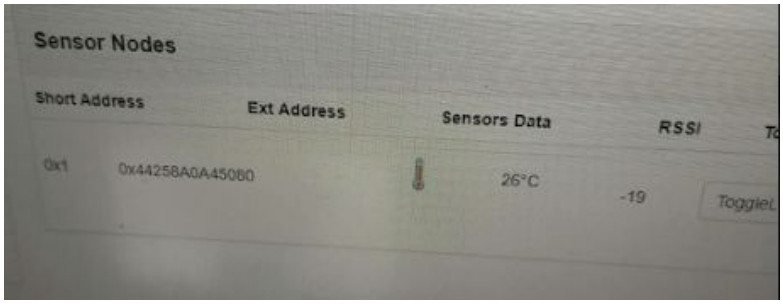
Results from Linux Gateway:

Video link: https://youtu.be/3QfJmNJaPts

The BeagleBone Black was also able to be flashed with the correct version of Linux. Once we connected the BBB to the network and connected the collector node to it, we were able to send the internal temperature of the sensor node to the collector, which will then display the data on a web page.



- Serial port of sensor, modified slightly to show what was sent to collector



- View of the page that is created by the BBB. Shows the same value sent by sensor

As stated earlier, we were not able to combine the two to send the lux values to the BBB to display. Only a value of 20 was being sent, which we aren't sure where it was coming from.