Nathan Hanuscin

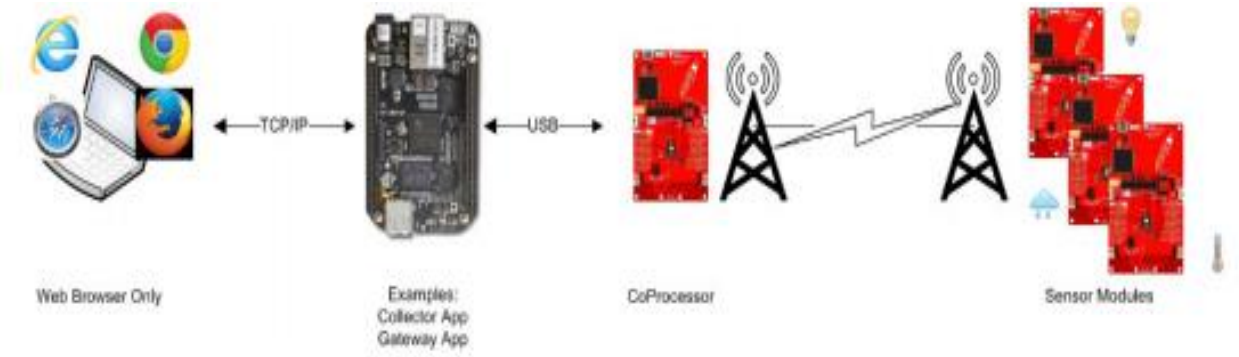
Jose Mendoza

Dr. Muthukumar

CPE403

December 13, 2019

TI 15.4-STACK LINUX SDK WITH CC1350 AND BBB

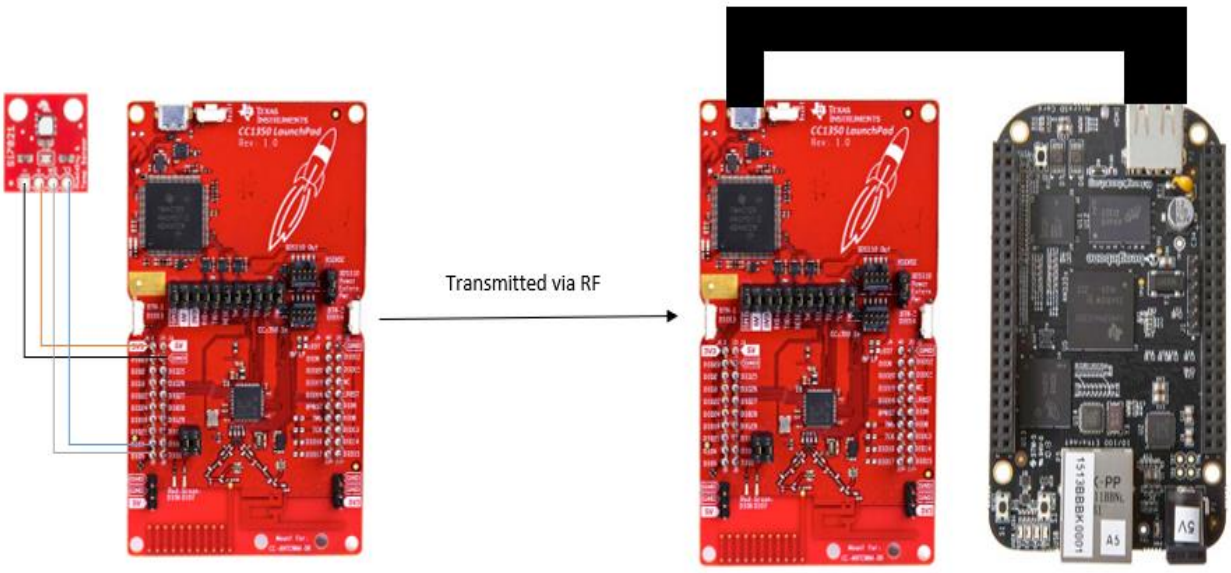


PROBLEM STATEMENT:

Our goal was to use the TI 15.4-Stack Linux SDK to create a star topology network with the BBB and two CC1350 launchpads. We needed to use the two CC1350s to have one act as a sensor launchpad and the other as a co-processor for the BBB. The BBB would act as the embedded host. After setting this up we would need to connect multiple sensors the sensor launchpad to transmit data to the BBB and display it on the web application.

The sensor we used was the Si7021 temperature and humidity sensor. This would be connected to the sensor launchpad via I2C. In order for data to be transmitted we had to modify the sensor.c file. After adding some header files and modifying the readSensors() function we were able to successfully send the temperature value to the BBB and display it on the web application. However, due to time constraints we were not able to get another sensor to and connect it to the sensor launchpad to send another set of data.

Diagram of our project:



PRE-REQUISITES:

Components used:

- 2 CC1350 – Used as sensor and co-processor
- BeagleBone Black – Used as embedded host to run web application to display sensor data
- Si7021 Temperature and Humidity Sensor – Connected to the sensor launchpad via IC2

Software used:

- UniFlash- used to flash the CC1350s to the correct configurations
- Code Composer Studio – The Sensor launchpad was programed in CCS
- Putty – Used to verify BBB was booted properly and to check value being sent
- Ubuntu VM – Used to set up the BBB

IMPLEMENTATION DETAILS:

Implementing Si7021 to sensor node with I2C:

1. The first step was to add the appropriate library header files into sensor.c:

```
#include <ti/drivers/GPIO.h>
#include <ti/display/Display.h>
// 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
```

2. The next step was to add address definitions for I2C:

```
//Si7021 address definitions
#define TASKSTACKSIZE       640
/*
 *  ======== TMP Registers ========
 */
#define Si7021_TMP_REG          0xE3
#define Si7021_HUM_REG          0xE5
#define Si7021_ADDR         0x40;
```

3. After that we modified the readSensors() function with variable declarations and initialized the Si7021:

```
static void readSensors(void)
{
#if defined(TEMP_SENSOR)

    //Variable definitions
    uint16_t        temperature,temperaturef;
    uint8_t         txBuffer[1];
    uint8_t         rxBuffer[2];
    I2C_Handle      i2c;
    I2C_Params      i2cParams;
    I2C_Transaction i2cTransaction;

    //Call driver init functions
    Display_init();
    GPIO_init();
    I2C_init();
    Display_printf(display, 0, 0, "Starting the i2ctmp example.");

    /* Create I2C for usage */
    I2C_Params_init(&i2cParams);
    i2cParams.bitRate = I2C_400kHz;
    i2c = I2C_open(Board_I2C_TMP, &i2cParams);
    if (i2c == NULL)
    {
        Display_printf(display, 0, 0, "Error Initializing I2C\n");
        while (1);
    }
    else
    {
        Display_printf(display, 0, 0, "I2C Initialized!\n");
    }

        /* Common I2C transaction setup */
    i2cTransaction.writeBuf   = txBuffer;
    i2cTransaction.writeCount = 1;
    i2cTransaction.readBuf    = rxBuffer;
    i2cTransaction.readCount  = 2;
```

4. Then we extracted the data from the sensor and sent it to the co-processor:

```
        /* Try Si7021 */
    txBuffer[0] = Si7021_TMP_REG;
    i2cTransaction.slaveAddress = Si7021_ADDR;
    if (!I2C_transfer(i2c, &i2cTransaction))
    {
        /* Could not resolve a sensor, error */
        Display_printf(display, 0, 0, "Error. No TMP sensor found!");
        while(1);
    }
    else
    {
        Display_printf(display, 0, 0, "Detected Si7021 sensor.");
    }

    if (I2C_transfer(i2c, &i2cTransaction))
    {
        /*
         * Extract degrees C from the received data;
         * see Si7021 datasheet
         */
        temperature = (rxBuffer[0] << 8) | (rxBuffer[1]);
        temperaturef = (((175.72 * temperature)/ 65536) - 46.85);
        Display_printf(display, 0, 0, "Temperature is: %d (C)", temperaturef);
    }
    else
    {
        Display_printf(display, 0, 0, "I2C Bus fault.");
    }

    I2C_close(i2c);
    Display_printf(display, 0, 0, "I2C closed!");
    /* Read the temp sensor values */
    tempSensor.ambienceTemp = (uint16_t) temperaturef;
    tempSensor.objectTemp =  tempSensor.ambienceTemp;
#endif
}
```

OUTCOMES, RESULTS AND CONCLUSIONS:

We were able to successfully send data to the embedded host and display it in the web application. The photos below show the temperature before and after applying an ice pack to the Si7021. However, we did not set up a second sensor to send data due to time constraints.

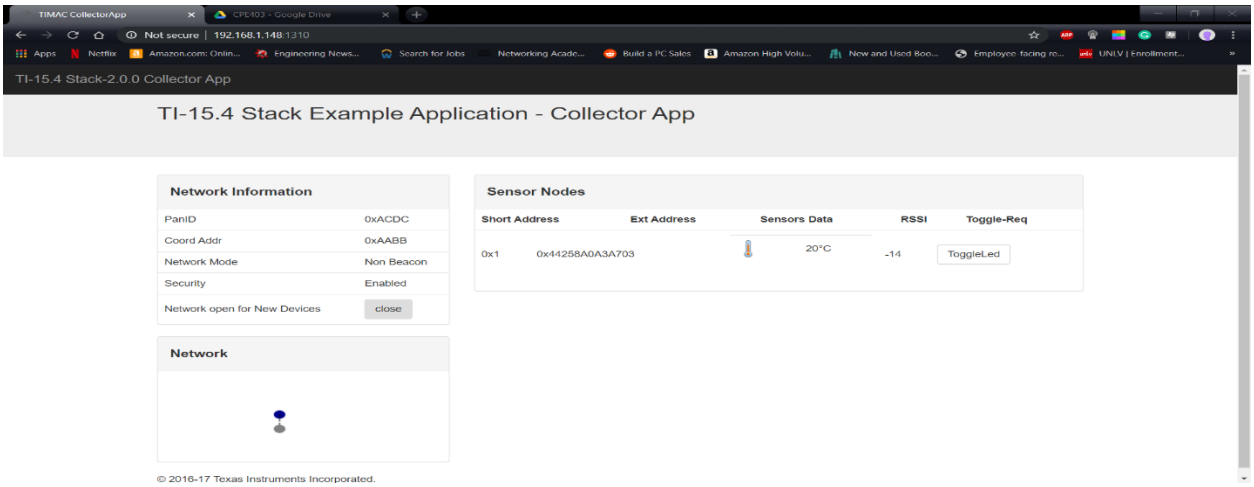


Fig 1 [Shows the initial temperature at 20 Celsius]

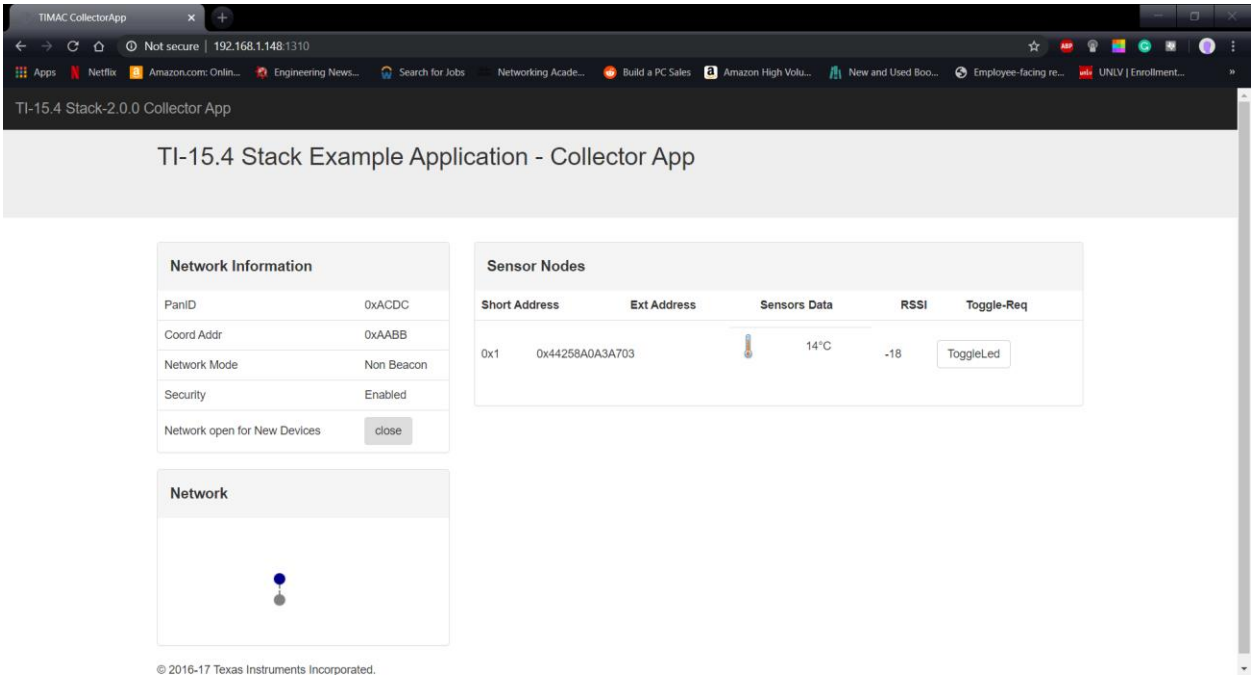


Fig 2 [Shows the temperature drop to 14 Celsius]

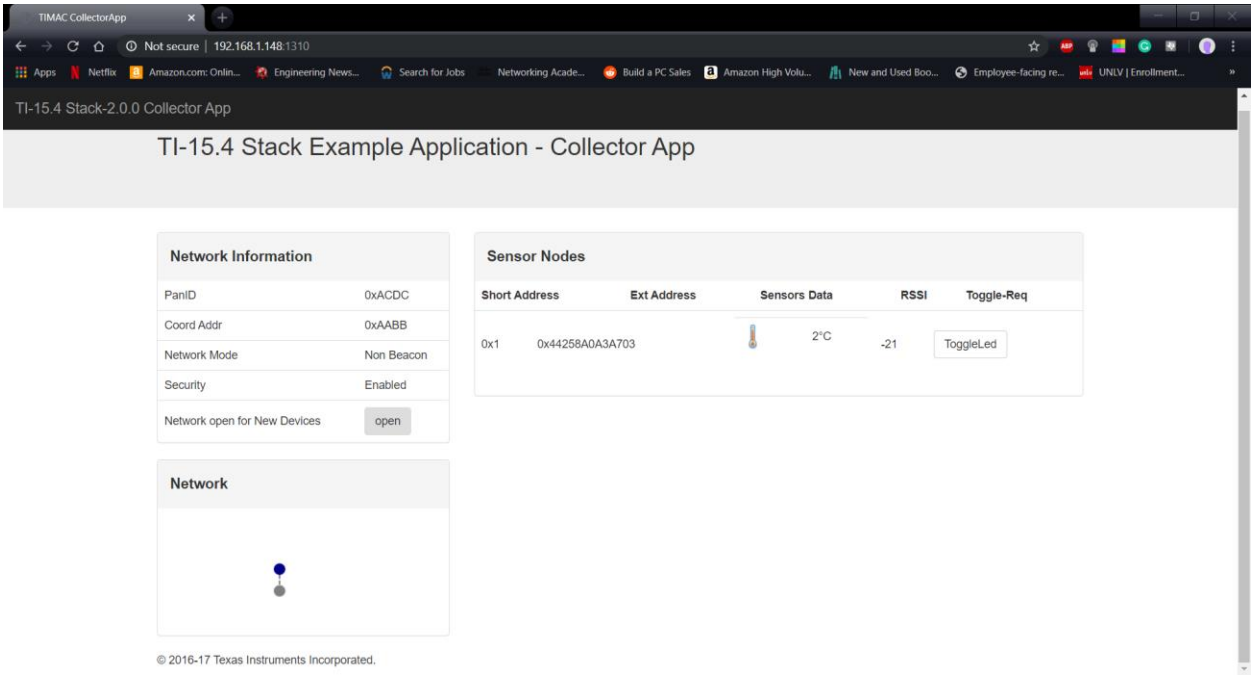


Fig 3 [Shows the temperature drop to 2 Celsius]

Video Demo:

https://youtu.be/INwq8uz3WdU