**Date Submitted: December 10, 2019**

A CC1352R1 will be set up as a collector module and a CC1350 will be setup as a sensor module.

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

**Task 01 :**

Youtube Link: Task 3’s video showing both collector and sensor running https://youtu.be/Fz9JTOFhRGI

**Modified Code:**

**Code is not modified as the function shown below already displays a temperature value received from the sensor module.**

**void** Csf\_deviceSensorDataUpdate(ApiMac\_sAddr\_t \*pSrcAddr, int8\_t rssi,

Smsgs\_sensorMsg\_t \*pMsg)

{

CUI\_ledToggle(csfCuiHndl, CONFIG\_LED\_GREEN);

**#ifndef** POWER\_MEAS

CUI\_statusLinePrintf(csfCuiHndl, deviceStatusLine, "Sensor - Addr=0x%04x, Temp=%d, RSSI=%d",

pSrcAddr->addr.shortAddr, pMsg->tempSensor.ambienceTemp, rssi);

CUI\_statusLinePrintf(csfCuiHndl, numJoinDevStatusLine, "%x", getNumActiveDevices());

**#endif** /\* endif for POWER\_MEAS \*/

**#if** defined(MT\_CSF)

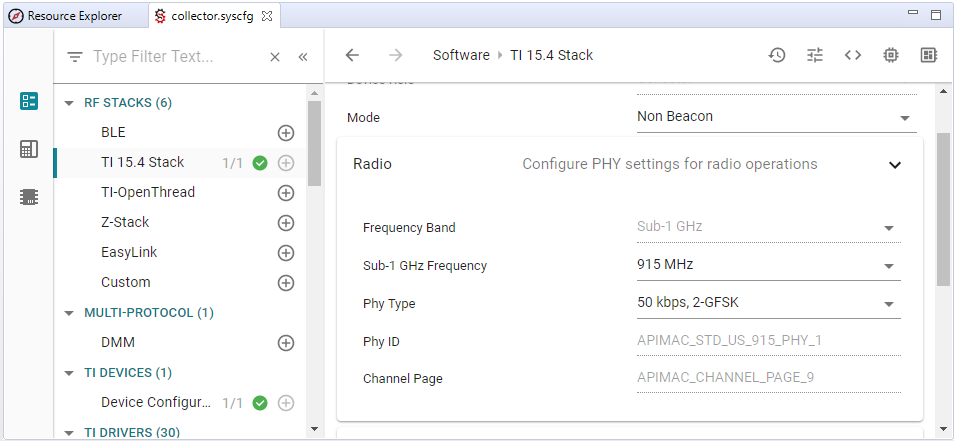
MTCSF\_sensorUpdateIndCB(pSrcAddr, rssi, pMsg);

**#endif** /\* endif for MT\_CSF \*/

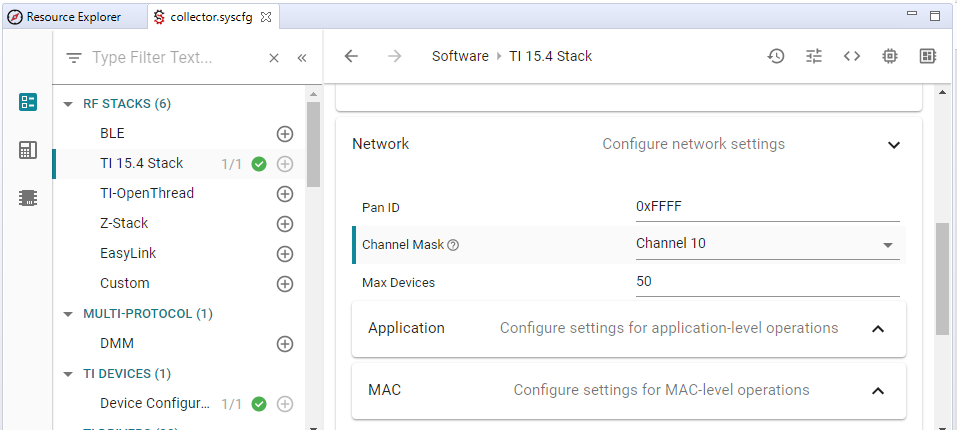
}

**Screenshots**

**Collector is set for 50kbps and US-compatible PHY**

****

**Only Channel 10 is enabled.**

****

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

**Task 02 :**

Youtube Link: Task 3’s video showing both collector and sensor running https://youtu.be/Fz9JTOFhRGI

**Modified Code:**

**Channel mask set to channel 10 using**

**#define** CONFIG\_CHANNEL\_MASK { 0x00, 0x04, 0x00, 0x00, 0x00, 0x00, \

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, \

0x00, 0x00, 0x00, 0x00, 0x00 }

**Screenshots:**

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

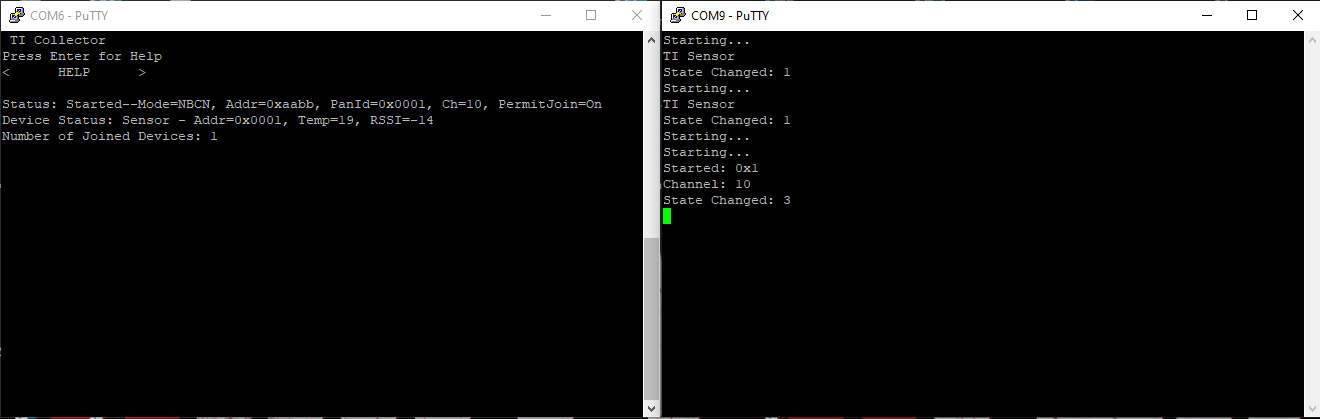
**Task 03 :**

Youtube Link: https://youtu.be/Fz9JTOFhRGI

**Modified Code:**

**Screenshots:**

**UART Screen capture**

****

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

**Task 04:**

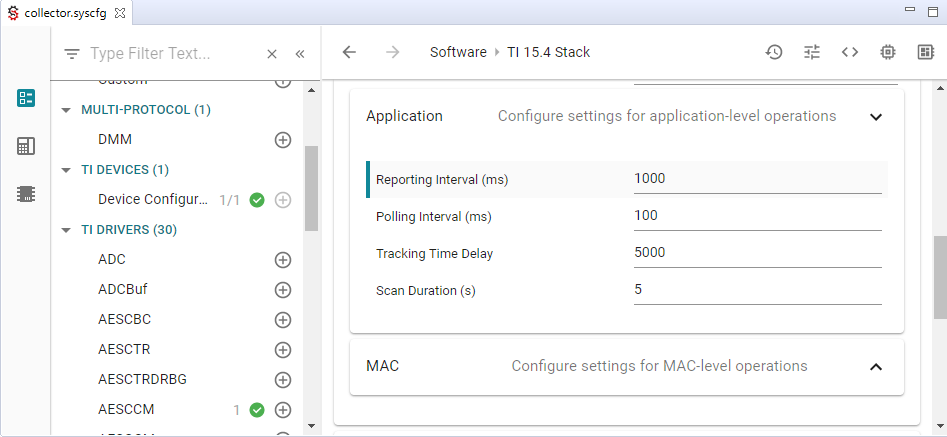
Youtube Link: https://youtu.be/b\_lpkOnEAV4

**Sensor Modification:**

**#define** CONFIG\_REPORTING\_INTERVAL 500

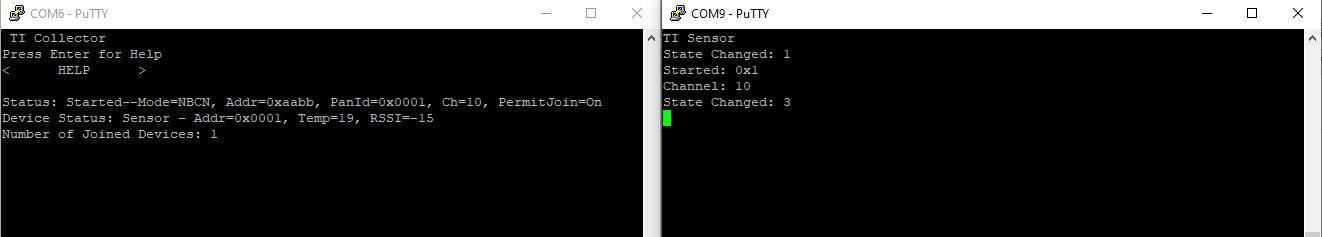
**Collector Modification:**

**Changed so that collector reports every second and polls ever 100ms.**

****

**Screenshots:**

**UART Screen capture**

****

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

**Task 05 (Remote Sensor 1):**

Youtube Link: https://youtu.be/QXWPbpAIJTk

Temperature.c Modifications:

Added global definitions

/\* ======== Si7021 Registers ======== \*/

**#define** Si7021\_TMP\_REG 0xE3

**#define** Si7021\_HUM\_REG 0xE5

**#define** Si7021\_ADDR 0x40;

Searches for Si7021 sensore

/\* 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.");

}

Changed how temperature is gathered from Si7021

**while** (1) {

**if** (**I2C\_transfer**(i2c, &i2cTransaction)) {

/\*

\* Extract degrees C from the received data; see sensor datasheet.

\* Make sure we are updating the global temperature variables

\* in a thread-safe manner.

\*/

**pthread\_mutex\_lock**(&temperatureMutex);

/\*

\* Extract degrees C from the received data;

\* see Si7021 datasheet

\*/

temperatureC = (rxBuffer[0] << 8) | (rxBuffer[1]);

temperatureC = (((175.72 \* temperatureC)/ 65536) - 46.85);

temperatureF = (temperatureC\*5)/9+32;

**pthread\_mutex\_unlock**(&temperatureMutex);

/\* Send an alert if the temperature is too high!! \*/

**if** ((**int**)temperatureC >= HIGH\_TEMP) {

sendAlert(temperatureC);

}

**else** {

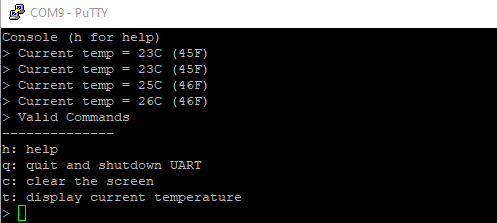
clearAlert(temperatureC);

}

}

**Screenshots:**

**UART**

****

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

**Task 06:**

Youtube Link: https://youtu.be/lJHNSgNNXW8

Modified Code:

Added following code to bottom of app.cfg

/\* Include POSIX Support \*/

**var** Settings = xdc.useModule('ti.sysbios.posix.Settings');

In main\_tirtos.c

Renamed main() to main\_app() and commented out calls to Board\_init(), GPIO\_init() and BIOS\_start().

In main.c

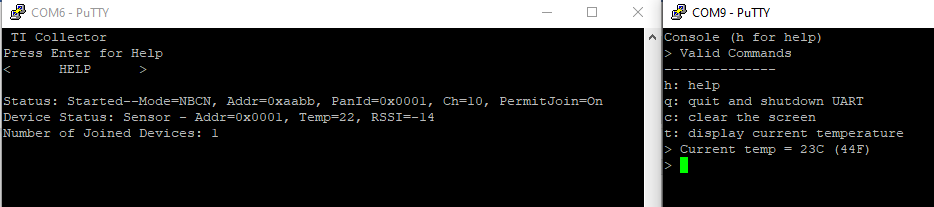
Added main\_app(); towards the end and added the following line before the start of main()

Extern int main\_app(void);

In sensor.opts, removed DBOAD\_DISPLAY\_USE\_UART and DxBOARD\_DISPLAY\_USE\_LCD

Screenshots:

UART Output



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

**Task 07:**

Youtube Link: https://youtu.be/HG6ytxO1sbE

Modified Code:

In sensor.h, added #define EXT\_SENSOR\_READING\_TIMEOUT\_EVT 0x0004 to global definitions

In sensor.c, added the following coded inside the Sensor\_process() function

**if**(Sensor\_events & EXT\_SENSOR\_READING\_TIMEOUT\_EVT)

{

/\* Process Sensor Reading Message Event \*/

processSensorMsgEvt();

/\* Clear the event \*/

Util\_clearEvent(&Sensor\_events, EXT\_SENSOR\_READING\_TIMEOUT\_EVT);

}

Also removed ‘STATIC’ from the line STATIC Smsgs\_tempSensorField\_t tempSensor =

In console.c, added following lines under global context

**#include** "smsgs.h"

**#include** "mac\_util.h"

**#include** "api\_mac.h"

**#include** "sensor.h"

**extern** Smsgs\_tempSensorField\_t tempSensor;

Also added the following lines in simpleConsole(), after the line “case t”

tempSensor.objectTemp = localTemperatureC;

tempSensor.ambienceTemp = localTemperatureC;

Util\_setEvent(&Sensor\_events, EXT\_SENSOR\_READING\_TIMEOUT\_EVT);

Screenshots:

UART, t needs to be held down has the collector update the temperature frequently

