CPE403/ECG603/ECG710 – Advanced Embedded Systems/Real-Time Embedded Systems

Design Assignment 5

DO NOT REMOVE THIS PAGE DURING SUBMISSION:

Name: Joshua Martinez

PartnerS: Payson Story & Hashim Hameed

Email: marti87@unlv.nevada.edu

Github Repository link (root): https://github.com/JoshuaMa2003/submission_da_0011

Youtube Playlist link (root):
https://www.youtube.com/playlist?list=PLZ09MA_uFt61Vw2JCAtw-_8vAdmoCKXpZ

Follow the submission guideline to be awarded points for this Assignment.

Submit the following for all Assignments:

1. In the document, for each task submit the modified or included code (from the base code) with highlights and justifications of the modifications. Also include comments. If no base code is provided, submit the base code for the first task only.
2. Create a private Github repository with a random name (no CPE403/603/710, Lastname, Firstname). Place all labs under the root folder MSP432E4/CC1352, sub-folder named Assignment1, with one document and one video link file for each lab, place modified c files named as asng_taskxx.c.
3. If multiple c files or other libraries are used, create a folder asng1_t01 and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) with startup_ccs.c and other include files, c) text file with youtube video links (see template).
5. Submit the doc file in canvas before the due date. The root folder of the github assignment directory should have the documentation and the text file with youtube video links.

6. Organize your videos as a playlist under the name “ADVEMBSYS”. The playlist should have the video sequence arranged as submission or due dates.

7. Only submit the PDF. Upload this document in the github repository and in canvas.

1. **Code for tasks: For each task, submit the relevant modified/section or included code (from the base code) with highlights and justifications of the modifications. Also include the comments. If no base code is provided, submit the initialization and execution section of each task separately. Use a separate page for each task.**

ALL OTHER FILES LOCATED ON GITHUB

SMSGS.h - Modified File

```
/******************************************************************************

@file smsgs.h

@brief Data Structures for the sensor messages sent over the air.

Group: WCS LPC
Target Device: cc13xx_cc26xx

******************************************************************************

Copyright (c) 2016-2025, Texas Instruments Incorporated
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

*  Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.

*  Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

*  Neither the name of Texas Instruments Incorporated nor the names of
   its contributors may be used to endorse or promote products derived
   from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
```

```
 EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
 WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
 OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
 EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

 **************************************************************************



 **************************************************************************/
#ifndef SMGSS_H
#define SMGSS_H

/*********************************************************************
 Includes
 *********************************************************************/

#include <stdbool.h>
#include <stdint.h>
#include "ti_154stack_config.h"

#ifdef __cplusplus
extern "C"
{
#endif

/*!
 \defgroup Sensor Over-the-air Messages
 <BR>
 This header file defines the over-the-air messages between the collector
 and the sensor applications.
 <BR>
 Each field in these message are formatted low byte first, so a 16 bit field
 with a value of 0x1516 will be sent as 0x16, 0x15.
 <BR>
 The first byte of each message (data portion) contains the command ID (@ref
 Smsgs_cmdIds).
 <BR>
 The <b>Configuration Request Message</b> is defined as:
     - Command ID - [Smsgs_cmdIds_configReq](@ref Smsgs_cmdIds) (1 byte)
     - Frame Control field - Smsgs_dataFields (16 bits) - tells the sensor
     what to report in the Sensor Data Message.
     - Reporting Interval - in millseconds (32 bits) - how often to report, 0
     means to turn off automated reporting, but will force the sensor device
     to send the Sensor Data message once.
     - Polling Interval - in millseconds (32 bits) - If the sensor device is
     a sleep device, this tells the device how often to poll its parent for
     data.
 <BR>
 The <b>Configuration Response Message</b> is defined as:
```

```
    - Command ID - [Smsgs_cmdIds_configRsp](@ref Smsgs_cmdIds) (1 byte)
    - Status field - Smsgs_statusValues (16 bits) - status of the
    configuration request.
    - Frame Control field - Smsgs_dataFields (16 bits) - tells the collector
    what fields are supported by this device (this only includes the bits set
    in the request message).
    - Reporting Interval - in millseconds (32 bits) - how often to report, 0
    means to turn off reporting.
    - Polling Interval - in millseconds (32 bits) - If the sensor device is
    a sleep device, this tells how often this device will poll its parent.
    A value of 0 means that the device doesn't sleep.
<BR>
The <b>Sensor Ramp Data Message</b> is defined as:
    - Command ID - [Smsgs_cmdIds_rampdata](@ref Smsgs_cmdIds) (1 byte)
    - Data Fields - Variable length ramp data of size SENSOR_TEST_RAMP_DATA_SIZE
<BR>
The <b>Sensor Data Message</b> is defined as:
    - Command ID - [Smsgs_cmdIds_sensorData](@ref Smsgs_cmdIds) (1 byte)
    - Frame Control field - Smsgs_dataFields (16 bits) - tells the collector
    what fields are included in this message.
    - Data Fields - The length of this field is determined by what data fields
    are included.  The order of the data fields are determined by the bit
    position of the Frame Control field (low bit first).  For example, if the
    frame control field has Smsgs_dataFields_tempSensor and
    Smsgs_dataFields_lightSensor set, then the Temp Sensor field is first,
    followed by the light sensor field.
<BR>
The <b>Temp Sensor Field</b> is defined as:
   - Ambience Chip Temperature - (int16_t) - each value represents signed
     integer part of temperature in Deg C (-256 .. +255)
   - Object Temperature - (int16_t) -  each value represents signed
     integer part of temperature in Deg C (-256 .. +255)
<BR>
The <b>Light Sensor Field</b> is defined as:
   - Raw Sensor Data - (uint16_6) raw data read out of the OPT2001 light
   sensor.
<BR>
The <b>Humidity Sensor Field</b> is defined as:
   - Raw Temp Sensor Data - (uint16_t) - raw temperature data from the
   Texas Instruments HCD1000 humidity sensor.
   - Raw Humidity Sensor Data - (uint16_t) - raw humidity data from the
   Texas Instruments HCD1000 humidity sensor.
<BR>
The <b>Message Statistics Field</b> is defined as:
    - joinAttempts - uint16_t - total number of join attempts (associate
    request sent)
    - joinFails - uint16_t - total number of join attempts failed
    - msgsAttempted - uint16_t - total number of sensor data messages
    attempted.
    - msgsSent - uint16_t - total number of sensor data messages successfully
    sent (OTA).
```

```
    - trackingRequests - uint16_t - total number of tracking requests received.
    - trackingResponseAttempts - uint16_t - total number of tracking response
    attempted
    - trackingResponseSent - uint16_t - total number of tracking response
    success
    - configRequests - uint16_t - total number of config requests received.
    - configResponseAttempts - uint16_t - total number of config response
    attempted
    - configResponseSent - uint16_t - total number of config response
    success
    - channelAccessFailures - uint16_t - total number of Channel Access
    Failures.  These are indicated in MAC data confirms for MAC data requests.
    - macAckFailures - uint16_t - Total number of MAC ACK failures. These are
    indicated in MAC data confirms for MAC data requests.
    - otherDataRequestFailures - uint16_t - Total number of MAC data request
    failures, other than channel access failure or MAC ACK failures.
    - syncLossIndications - uint16_t - Total number of sync loss failures
    received for sleepy devices.
    - rxDecryptFailues - uint16_t - Total number of RX Decrypt failures.
    - txEncryptFailues - uint16_t - Total number of TX Encrypt failures.
    - resetCount - uint16_t - Total number of resets.
    - lastResetReason - uint16_t - 0 - no reason, 2 - HAL/ICALL,
    3 - MAC, 4 - TIRTOS
 <BR>
 The <b>Config Settings Field</b> is defined as:
    - Reporting Interval - in millseconds (32 bits) - how often to report, 0
    means reporting is off.
    - Polling Interval - in millseconds (32 bits) - If the sensor device is
    a sleep device, this states how often the device polls its parent for
    data. This field is 0 if the device doesn't sleep.
 */

/*********************************************************************
 Constants and definitions
 *********************************************************************/
// ...
// ...
/* Max BLE Data Length */
#define MAX_BLE_DATA_LEN 20

/*! Length of the genericSensor portion of the sensor data message */
#define SMSGS_SENSOR_GENERIC_LEN 2

/*! Generic Request message length (over-the-air length) */
#define SMSGS_GENERIC_REQUEST_MSG_LEN 1

/*! Generic Response message length (over-the-air length) */
#define SMSGS_GENERIC_RESPONSE_MSG_LEN 2

#define SMGS_SENSOR_EXTADDR_LEN 8
```

```
/*! Config Request message length (over-the-air length) */
#define SMSGS_CONFIG_REQUEST_MSG_LENGTH 11
/*! Config Response message length (over-the-air length) */
#define SMSGS_CONFIG_RESPONSE_MSG_LENGTH 13
/*! Tracking Request message length (over-the-air length) */
#define SMSGS_TRACKING_REQUEST_MSG_LENGTH 1
/*! Tracking Response message length (over-the-air length) */
#define SMSGS_TRACKING_RESPONSE_MSG_LENGTH 1
/*! Broadcast Command message length (over-the-air-length) */
#ifdef FH_LOW_LATENCY_BROADCAST
#define SMSGS_BROADCAST_CMD_LENGTH  4
#else  // FH_LOW_LATENCY_BROADCAST
#define SMSGS_BROADCAST_CMD_LENGTH  3
#endif // FH_LOW_LATENCY_BROADCAST

/*! Length of a sensor data message with no configured data fields */
#define SMSGS_BASIC_SENSOR_LEN (3 + SMGS_SENSOR_EXTADDR_LEN)
/*! Length of the tempSensor portion of the sensor data message */
#define SMSGS_SENSOR_TEMP_LEN 4
/*! Length of the lightSensor portion of the sensor data message */
#define SMSGS_SENSOR_LIGHT_LEN 2
/*! Length of the humiditySensor portion of the sensor data message */
#define SMSGS_SENSOR_HUMIDITY_LEN 4
/*! Length of the messageStatistics portion of the sensor data message */
#define SMSGS_SENSOR_MSG_STATS_LEN 44
/*! Length of the configSettings portion of the sensor data message */
#define SMSGS_SENSOR_CONFIG_SETTINGS_LEN 8
/*! Toggle Led Request message length (over-the-air length) */
#define SMSGS_TOGGLE_LED_REQUEST_MSG_LEN 1
/*! Toggle Led Request message length (over-the-air length) */
#ifdef FH_LOW_LATENCY_BROADCAST
#define SMSGS_TOGGLE_LED_RESPONSE_MSG_LEN 1
#else  // FH_LOW_LATENCY_BROADCAST
#define SMSGS_TOGGLE_LED_RESPONSE_MSG_LEN 2
#endif // FH_LOW_LATENCY_BROADCAST
/*! Identify Led Request message length (over-the-air length) */
#define SMSGS_INDENTIFY_LED_REQUEST_MSG_LEN 2
/*! Identify Led Response message length (over-the-air length) */
#define SMSGS_INDENTIFY_LED_RESPONSE_MSG_LEN 2
/*! Device type request message length (over-the-air length) */
#define SMSGS_DEVICE_TYPE_REQUEST_MSG_LEN 1
/*! Device type response message length (over-the-air length) */
#define SMSGS_DEVICE_TYPE_RESPONSE_MSG_LEN 3
/*! Length of a BLE Device Address */
#define B_ADDR_LEN 6
/*! Length of the ble sensor portion of the sensor data length not including variable data field */
#define SMSGS_SENSOR_BLE_LEN 5 + B_ADDR_LEN
/* Max BLE Data Length */
#define MAX_BLE_DATA_LEN 20

/*!
```

```
Message IDs for Sensor data messages.  When sent over-the-air in a message,
this field is one byte.
*/
typedef enum
{
   /*! Configuration message, sent from the collector to the sensor */
   Smsgs_cmdIds_configReq = 1,
   /*! Configuration Response message, sent from the sensor to the collector */
   Smsgs_cmdIds_configRsp = 2,
   /*! Tracking request message, sent from the the collector to the sensor */
   Smsgs_cmdIds_trackingReq = 3,
    /*! Tracking response message, sent from the sensor to the collector */
   Smsgs_cmdIds_trackingRsp = 4,
   /*! Sensor data message, sent from the sensor to the collector */
   Smsgs_cmdIds_sensorData = 5,
   /* Toggle LED message, sent from the collector to the sensor */
   Smsgs_cmdIds_toggleLedReq = 6,
   /* Toggle LED response msg, sent from the sensor to the collector */
   Smsgs_cmdIds_toggleLedRsp = 7,
   /* new data type for ramp data */
   Smsgs_cmdIds_rampdata = 8,
   /*! OAD mesages, sent/received from both collector and sensor */
   Smsgs_cmdIds_oad = 9,
   /* Broadcast control msg, sent from the collector to the sensor */
   Smgs_cmdIds_broadcastCtrlMsg = 10,
   /* KEY Exchange msg, between collector and the sensor */
   Smgs_cmdIds_KeyExchangeMsg = 11,
   /* Identify LED request msg */
   Smsgs_cmdIds_IdentifyLedReq = 12,
   /* Identify LED response msg */
   Smsgs_cmdIds_IdentifyLedRsp = 13,
   /*! SM Commissioning start command sent from collector to the sensor */
   Smgs_cmdIds_CommissionStart  = 14,
   /*! SM Commissioning message sent bi-directionally between the collector and sensor */
   Smgs_cmdIds_CommissionMsg  = 15,
   /* Device type request msg */
   Smsgs_cmdIds_DeviceTypeReq = 16,
   /* Device type response msg */
   Smsgs_cmdIds_DeviceTypeRsp = 17,
   /* Generic request msg */
   Smsgs_cmdIds_genericReq = 18,
   /* Generic response msg */
   Smsgs_cmdIds_genericRsp = 19

} Smsgs_cmdIds_t;

/*!
Frame Control field states what data fields are included in reported
sensor data, each value is a bit mask value so that they can be combined
(OR'd together) in a control field.
When sent over-the-air in a message this field is 2 bytes.
```

```
 */
typedef enum
{
    /*! Temperature Sensor */
    Smsgs_dataFields_tempSensor = 0x0001,
    /*! Light Sensor */
    Smsgs_dataFields_lightSensor = 0x0002,
    /*! Humidity Sensor */
    Smsgs_dataFields_humiditySensor = 0x0004,
    /*! Message Statistics */
    Smsgs_dataFields_msgStats = 0x0008,
    /*! Config Settings */
    Smsgs_dataFields_configSettings = 0x0010,
#ifdef LPSTK
    /*! Hall Effect Sensor */
    Smsgs_dataFields_hallEffectSensor = 0x0020,
    /*! Accelerometer Sensor */
    Smsgs_dataFields_accelSensor = 0x0040,
#endif /* LPSTK */
    Smsgs_dataFields_bleSensor = 0x0080,
    /*! Generic Sensor */
    Smsgs_dataFields_genericSensor = 0x0800,
} Smsgs_dataFields_t;

/*!
 Status values for the over-the-air messages
 */
typedef enum
{
    /*! Success */
    Smsgs_statusValues_success = 0,
    /*! Message was invalid and ignored */
    Smsgs_statusValues_invalid = 1,
    /*!
     Config message was received but only some frame control fields
     can be sent or the reportingInterval or pollingInterval fail
     range checks.
     */
    Smsgs_statusValues_partialSuccess = 2,
} Smsgs_statusValues_t;

/***************************************************************************
 Structures - Building blocks for the over-the-air sensor messages
 ***************************************************************************/

/*!
 Configuration Request message: sent from controller to the sensor.
 */
typedef struct _Smsgs_configreqmsg_t
{
    /*! Command ID - 1 byte */
```

```
    Smsgs_cmdIds_t cmdId;
    /*! Frame Control field - bit mask of Smsgs_dataFields */
    uint16_t frameControl;
    /*! Reporting Interval */
    uint32_t reportingInterval;
    /*! Polling Interval */
    uint32_t pollingInterval;
} Smsgs_configReqMsg_t;

/*!
 Configuration Response message: sent from the sensor to the collector
 in response to the Configuration Request message.
 */
typedef struct _Smsgs_configrspmsg_t
{
    /*! Command ID - 1 byte */
    Smsgs_cmdIds_t cmdId;
    /*! Response Status - 2 bytes */
    Smsgs_statusValues_t status;
    /*! Frame Control field - 2 bytes - bit mask of Smsgs_dataFields */
    uint16_t frameControl;
    /*! Reporting Interval - 4 bytes */
    uint32_t reportingInterval;
    /*! Polling Interval - 4 bytes */
    uint32_t pollingInterval;
} Smsgs_configRspMsg_t;

/*!
 Tracking Request message: sent from controller to the sensor.
 */
typedef struct _Smsgs_trackingreqmsg_t
{
    /*! Command ID - 1 byte */
    Smsgs_cmdIds_t cmdId;
} Smsgs_trackingReqMsg_t;

/*!
 Tracking Response message: sent from the sensor to the collector
 in response to the Tracking Request message.
 */
typedef struct _Smsgs_trackingrspmsg_t
{
    /*! Command ID - 1 byte */
    Smsgs_cmdIds_t cmdId;
} Smsgs_trackingRspMsg_t;

/*!
 Toggle LED Request message: sent from controller to the sensor.
 */
typedef struct _Smsgs_toggleledreqmsg_t
{
```

```
    /*! Command ID - 1 byte */
    Smsgs_cmdIds_t cmdId;
} Smsgs_toggleLedReqMsg_t;

/*!
 Toggle LED Response message: sent from the sensor to the collector
 in response to the Toggle LED Request message.
 */
typedef struct _Smsgs_toggleledrspmsg_t
{
    /*! Command ID - 1 byte */
    Smsgs_cmdIds_t cmdId;
    /*! LED State - 0 is off, 1 is on - 1 byte */
    uint8_t ledState;
} Smsgs_toggleLedRspMsg_t;

/*!
 Identify LED Request message: sent from controller to the sensor.
 */
typedef struct _Smsgs_identifyledreqmsg_t
{
    /*! Command ID - 1 byte */
    Smsgs_cmdIds_t cmdId;
    /*! time to identify in s */
    uint8_t identifyTime;
} Smsgs_identifyLedReqMsg_t;

typedef struct _Smsgs_identifyledrspmsg_t
{
    /*! Command ID - 1 byte */
    Smsgs_cmdIds_t cmdId;
    /* status */
    uint8_t status;
} Smsgs_identifyLedRspMsg_t;

/*!
 Temp Sensor Field
 */
typedef struct _Smsgs_tempsensorfield_t
{
    /*!
     Ambience Chip Temperature - each value represents a 0.01 C
     degree, so a value of 2475 represents 24.75 C.
     */
    int16_t ambienceTemp;
    /*!
     Object Temperature - each value represents a 0.01 C
     degree, so a value of 2475 represents 24.75 C.
     */
    int16_t objectTemp;
} Smsgs_tempSensorField_t;
```

```
/*!
 Light Sensor Field
 */
typedef struct _Smsgs_lightsensorfield_t
{
    /*! Raw Sensor Data read out of the OPT2001 light sensor */
    uint16_t rawData;
} Smsgs_lightSensorField_t;

/*!
 Humidity Sensor Field
 */
typedef struct _Smsgs_humiditysensorfield_t
{
    /*! Raw Temp Sensor Data from the TI HCD1000 humidity sensor. */
    uint16_t temp;
    /*! Raw Humidity Sensor Data from the TI HCD1000 humidity sensor. */
    uint16_t humidity;
} Smsgs_humiditySensorField_t;

/*!
 Hall Effect Sensor Field
 */
typedef struct _Smsgs_halleffectsensorfield_t
{
    /*! Magnetic Flux Switch. */
    uint8_t fluxLevel;
} Smsgs_hallEffectSensorField_t;

/*!
 Accelerometer Sensor Field
 */
typedef struct _Smsgs_accelsensorfield_t
{
    /*! X axis accelerometer value. */
    int16_t xAxis;
    /*! Y axis accelerometer value. */
    int16_t yAxis;
    /*! Z axis accelerometer value. */
    int16_t zAxis;
    /*! Device tilted in the X axis. */
    uint8_t xTiltDet;
    /*! Device tilted in the Y axis. */
    uint8_t yTiltDet;
} Smsgs_accelSensorField_t;

typedef struct _Smsgs_blesensorfield_t
{
    /*! BLE Sensor Address */
    uint8_t  bleAddr[B_ADDR_LEN];
```

```
    /*! Manufacturer ID */
    uint16_t manFacID;
    /*! UUID */
    uint16_t uuid;
    /*! Length of BLE Char */
    uint8_t dataLength;
    /*! Pointer to BLE Characteristic Value */
    uint8_t data[MAX_BLE_DATA_LEN];

} Smsgs_bleSensorField_t;

/*!
 Generic Sensor Field
 */
typedef struct _Smsgs_genericsensorfield_t
{
    /*! Raw Sensor Data read out of the generic sensor */
    uint16_t genericRawData;
} Smsgs_genericSensorField_t;

/*!
 Message Statistics Field
 */
typedef struct _Smsgs_msgstatsfield_t
{
    /*! total number of join attempts (associate request sent) */
    uint16_t joinAttempts;
    /*! total number of join attempts failed */
    uint16_t joinFails;
    /*! total number of sensor data messages attempted. */
    uint16_t msgsAttempted;
    /*! total number of sensor data messages sent. */
    uint16_t msgsSent;
    /*! total number of tracking requests received */
    uint16_t trackingRequests;
    /*! total number of tracking response attempted */
    uint16_t trackingResponseAttempts;
    /*! total number of tracking response success */
    uint16_t trackingResponseSent;
    /*! total number of config requests received */
    uint16_t configRequests;
    /*! total number of config response attempted */
    uint16_t configResponseAttempts;
    /*! total number of config response success */
    uint16_t configResponseSent;
    /*!
     Total number of Channel Access Failures.  These are indicated in MAC data
     confirms for MAC data requests.
     */
    uint16_t channelAccessFailures;
    /*!
```

```
     Total number of MAC ACK failures. These are indicated in MAC data
     confirms for MAC data requests.
     */
    uint16_t macAckFailures;
    /*!
     Total number of MAC data request failures, other than channel access
     failure or MAC ACK failures.
     */
    uint16_t otherDataRequestFailures;
    /*! Total number of sync loss failures received for sleepy devices. */
    uint16_t syncLossIndications;
    /*! Total number of RX Decrypt failures. */
    uint16_t rxDecryptFailures;
    /*! Total number of TX Encrypt failures. */
    uint16_t txEncryptFailures;
    /*! Total number of resets. */
    uint16_t resetCount;
    /*!
     Assert reason for the last reset -  0 - no reason, 2 - HAL/ICALL,
     3 - MAC, 4 - TIRTOS
     */
    uint16_t lastResetReason;
    /*! Amount of time taken for node to join */
    uint16_t joinTime;
    /*! Max re-join delay */
    uint16_t interimDelay;
    /*!
     Number of broadcast messages received from the collector
     */
    uint16_t numBroadcastMsgRcvd;
    /*!
    Number of broadcast messages missed from the collector
    */
    uint16_t numBroadcastMsglost;
    /*!
    Average end to end delay
    */
    uint16_t avgE2EDelay;
    /*!
    Worst Case end to end delay
    */
    uint16_t worstCaseE2EDelay;
} Smsgs_msgStatsField_t;

#ifdef POWER_MEAS
/*!
 Power Meas Statistics Field
 */
typedef struct _Smsgs_powerMeastatsField_t
{
    /*! total number of polls sent. Not applicable for beacon mode */
```

```
    uint16_t pollRequestsSent;
    /*! total number of collector ramp data received */
    uint16_t rampDataRcvd;
} Smsgs_powerMeastatsField_t;
#endif

/*!
 Message Statistics Field
 */
typedef struct _Smsgs_configsettingsfield_t
{
    /*!
     Reporting Interval - in millseconds, how often to report, 0
     means reporting is off
     */
    uint32_t reportingInterval;
    /*!
     Polling Interval - in millseconds (32 bits) - If the sensor device is
     a sleep device, this states how often the device polls its parent for
     data. This field is 0 if the device doesn't sleep.
     */
    uint32_t pollingInterval;
} Smsgs_configSettingsField_t;

/*!
 Sensor Data message: sent from the sensor to the collector
 */
typedef struct _Smsgs_sensormsg_t
{
    /*! Command ID */
    Smsgs_cmdIds_t cmdId;
    /*! Extended Address */
    uint8_t extAddress[SMGS_SENSOR_EXTADDR_LEN];
    /*! Frame Control field - bit mask of Smsgs_dataFields */
    uint16_t frameControl;
    /*!
     Temp Sensor field - valid only if Smsgs_dataFields_tempSensor
     is set in frameControl.
     */
    Smsgs_tempSensorField_t tempSensor;
    /*!
     Light Sensor field - valid only if Smsgs_dataFields_lightSensor
     is set in frameControl.
     */
    Smsgs_lightSensorField_t lightSensor;
    /*!
     Humidity Sensor field - valid only if Smsgs_dataFields_humiditySensor
     is set in frameControl.
     */
    Smsgs_humiditySensorField_t humiditySensor;
    /*!
```

```
     Message Statistics field - valid only if Smsgs_dataFields_msgStats
     is set in frameControl.
     */
    Smsgs_msgStatsField_t msgStats;
    /*!
     Configuration Settings field - valid only if
     Smsgs_dataFields_configSettings is set in frameControl.
     */
    Smsgs_configSettingsField_t configSettings;
#ifdef LPSTK
    /*!
     Hall Effect Sensor field - valid only if Smsgs_dataFields_hallEffectSensor
     is set in frameControl.
     */
    Smsgs_hallEffectSensorField_t hallEffectSensor;
    /*!
     Accelerometer Sensor field - valid only if Smsgs_dataFields_accelSensor
     is set in frameControl.
     */
    Smsgs_accelSensorField_t accelerometerSensor;
#endif /* LPSTK */
    /*!
     BLE Sensor field - valid only if Smsgs_dataFields_bleSensorField_t
     is set in frameControl.
     */
    Smsgs_bleSensorField_t bleSensor;
    /*!
     Generic Sensor field - valid only if Smsgs_dataFields_genericSensor
     is set in frameControl.
     */
    Smsgs_genericSensorField_t genericSensor;
} Smsgs_sensorMsg_t;

/*!
 Broadcast Cmd Request message: sent from controller to the sensor.
 */
typedef struct _Smsgs_broadcastcmdmsg_t
{
    /*! Command ID - 1 byte */
    Smsgs_cmdIds_t cmdId;
#ifdef FH_LOW_LATENCY_BROADCAST
    uint16_t destdevAddr; // destination device address
    uint8_t camCmd;       // camera command for the destination device
#else  // FH_LOW_LATENCY_BROADCAST
    uint16_t broadcastMsgId;
#endif
}Smsgs_broadcastcmdmsg_t;


#ifdef __cplusplus
}
```

```
#endif

#endif /* SMGSS_H */
```

COLLECTOR.H - Modified File

```
/*********************************************************************
***********

@file collector.h

@brief TIMAC 2.0 Collector Example Application Header

Group: WCS LPC
Target Device: cc13xx_cc26xx


 *********************************************************************
 **********

Copyright (c) 2016-2025, Texas Instruments Incorporated
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

*  Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.

*  Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in
  the
   documentation and/or other materials provided with the
  distribution.

*  Neither the name of Texas Instruments Incorporated nor the names
```

```
  of
   its contributors may be used to endorse or promote products
  derived
   from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
  "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
  PARTICULAR
PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
  PROFITS;
OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
  LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
  OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.


  **********************************************************************
  **********



  **********************************************************************
  *********/
#ifndef COLLECTOR_H
#define COLLECTOR_H

/**********************************************************************
  *********
```

```
 Includes

   *************************************************************************
   *********/

#include <stdbool.h>
#include <stdint.h>

#include "ti_154stack_config.h"
#include "api_mac.h"

#ifdef OSAL_PORT2TIRTOS
#ifndef FREERTOS_SUPPORT
#include <ti/sysbios/knl/Task.h>
#endif
#endif

#ifdef FEATURE_SECURE_COMMISSIONING
#include "sm_ti154.h"
#endif

#ifdef __cplusplus
extern "C"
{
#endif


/*************************************************************************
   *********
 Constants and definitions

   *************************************************************************
   *********/
#ifdef IEEE_COEX_TEST
#define APS_RETRY_INTERVAL       (50)
```

```
#define APS_INTERVAL             (10)
#define APS_MAX_FRAME_RETRIES    (2)
#endif // IEEE_COEX_TEST
/*! Event ID - Start the device in the network */
#define COLLECTOR_START_EVT              0x0001
/*! Event ID - Tracking Timeout Event */
#define COLLECTOR_TRACKING_TIMEOUT_EVT   0x0002
/*! Event ID - Generate Configs Event */
#define COLLECTOR_CONFIG_EVT             0x0004
#define COLLECTOR_BROADCAST_TIMEOUT_EVT  0x0008
#ifdef FH_LOW_LATENCY_BROADCAST
#define COLLECTOR_BROADCAST_CAMCMD_EVT   0x1000
#define COLLECTOR_BROADCAST_SUCCESS_EVT  0x2000
#define COLLECTOR_BROADCAST_CAMCMDACK_EVT 0x4000
#endif // FH_LOW_LATENCY_BROADCAST

/*! Event ID - Start Provisioning Event */
#define COLLECTOR_PROV_EVT               0x0010
/*! Event ID - Open Network Event */
#define COLLECTOR_OPEN_NWK_EVT           0x0040
/*! Event ID - Close Network Event */
#define COLLECTOR_CLOSE_NWK_EVT          0x0080

#ifdef DMM_OAD
/*! Event ID - Pause 154 Collector */
#define COLLECTOR_PAUSE_EVT              0x0100
/*! Event ID - Resume 154 Collector */
#define COLLECTOR_RESUME_EVT             0x0200
#endif

#ifdef LPSTK
/* LPSTK configuration frame control */
#define CONFIG_FRAME_CONTROL (Smsgs_dataFields_tempSensor | \
                              Smsgs_dataFields_lightSensor | \
                              Smsgs_dataFields_humiditySensor | \
```

```
                                Smsgs_dataFields_msgStats | \
                                Smsgs_dataFields_configSettings | \
                                Smsgs_dataFields_hallEffectSensor | \
                                Smsgs_dataFields_accelSensor)
#else
/* Default configuration frame control */
#define CONFIG_FRAME_CONTROL (Smsgs_dataFields_tempSensor | \
                                Smsgs_dataFields_lightSensor | \
                                Smsgs_dataFields_humiditySensor | \
                                Smsgs_dataFields_msgStats | \
                                Smsgs_dataFields_configSettings | \
                                Smsgs_dataFields_genericSensor)
#endif
/*! Collector Status Values */
typedef enum
{
    /*! Success */
    Collector_status_success = 0,
    /*! Device Not Found */
    Collector_status_deviceNotFound = 1,
    /*! Collector isn't in the correct state to send a message */
    Collector_status_invalid_state = 2
} Collector_status_t;

/* Beacon order for non beacon network */
#define NON_BEACON_ORDER      15

/* Number of superframe periods to hold a indirect packet at collector
   for
Sensor to poll and get the frame*/
#define BCN_MODE_INDIRECT_PERSISTENT_TIME 3

#define MIN_PERSISTENCE_TIME_USEC 2000000

#ifndef MAX
```

```
#define MAX(n,m)    (((n) < (m)) ? (m) : (n))
#endif

#ifndef MIN
#define MIN(n,m)    (((n) < (m)) ? (n) : (m))
#endif


#if (CONFIG_PHY_ID == APIMAC_CUSTOM_PHY_ID)
/* MAC Indirevt Persistent Timeout */
#define INDIRECT_PERSISTENT_TIME (MIN((MAX((5 * 1000 *
   CONFIG_POLLING_INTERVAL / 2), MIN_PERSISTENCE_TIME_USEC)/ \
                                    (BASE_SUPER_FRAME_DURATION * \
                                     SYMBOL_DURATION_CUSTOM)), 65535))

#elif  ((CONFIG_PHY_ID >= APIMAC_MRFSK_STD_PHY_ID_BEGIN) &&
   (CONFIG_PHY_ID <= APIMAC_MRFSK_GENERIC_PHY_ID_BEGIN))

/* MAC Indirect Persistent Timeout */
#define INDIRECT_PERSISTENT_TIME (MIN((MAX((5 * 1000 *
   CONFIG_POLLING_INTERVAL / 2), MIN_PERSISTENCE_TIME_USEC)/ \
                                    (BASE_SUPER_FRAME_DURATION * \
                                     SYMBOL_DURATION_50_kbps)), 65535))

#elif ((CONFIG_PHY_ID >= APIMAC_200KBPS_915MHZ_PHY_132) &&
   (CONFIG_PHY_ID <= APIMAC_200KBPS_868MHZ_PHY_133)) || \
       (CONFIG_PHY_ID == APIMAC_200KBPS_920MHZ_PHY_136)

/* MAC Indirect Persistent Timeout */
#define INDIRECT_PERSISTENT_TIME (MIN((MAX((5 * 1000 *
   CONFIG_POLLING_INTERVAL / 2),MIN_PERSISTENCE_TIME_USEC) / \
                                    (BASE_SUPER_FRAME_DURATION * \
                                     SYMBOL_DURATION_200_kbps)), 65535))

#elif (CONFIG_PHY_ID == APIMAC_250KBPS_IEEE_PHY_0)
```

```
/* MAC Indirect Persistent Timeout */
#ifdef FEATURE_SECURE_COMMISSIONING
/* Increase the persistence time by a factor of 10 */
#define INDIRECT_PERSISTENT_TIME (MIN((MAX((5* 10 * 1000 *
   CONFIG_POLLING_INTERVAL / 2), MIN_PERSISTENCE_TIME_USEC)/ \
                                    (BASE_SUPER_FRAME_DURATION * \
                                     SYMBOL_DURATION_250_kbps)), 65535))
#else
#define INDIRECT_PERSISTENT_TIME (MIN((MAX((5 * 1000 *
   CONFIG_POLLING_INTERVAL / 2), MIN_PERSISTENCE_TIME_USEC)/ \
                                    (BASE_SUPER_FRAME_DURATION * \
                                     SYMBOL_DURATION_250_kbps)), 65535))
#endif

#else
/* MAC Indirect Persistent Timeout */
#define INDIRECT_PERSISTENT_TIME (MIN((MAX((5 * 1000 *
   CONFIG_POLLING_INTERVAL / 2), MIN_PERSISTENCE_TIME_USEC)/ \
                                    (BASE_SUPER_FRAME_DURATION * \
                                      SYMBOL_DURATION_LRM)), 65535))
#endif

/*************************************************************************
   *********
 Structures

   *************************************************************************
   *********/

/*! Collector Statistics */
typedef struct
{
    /*!
     Total number of tracking request messages attempted
```

```
 */
uint32_t trackingRequestAttempts;
/*!
 Total number of tracking request messages sent
 */
uint32_t trackingReqRequestSent;
/*!
 Total number of tracking response messages received
 */
uint32_t trackingResponseReceived;
/*!
 Total number of config request messages attempted
 */
uint32_t configRequestAttempts;
/*!
 Total number of config request messages sent
 */
uint32_t configReqRequestSent;
/*!
 Total number of config response messages received
 */
uint32_t configResponseReceived;
/*!
 Total number of sensor messages received
 */
uint32_t sensorMessagesReceived;
/*!
 Total number of failed messages because of channel access failure
 */
uint32_t channelAccessFailures;
/*!
 Total number of failed messages because of ACKs not received
 */
uint32_t ackFailures;
/*!
```

```
     Total number of failed transmit messages that are not channel
   access
     failure and not ACK failures
     */
    uint32_t otherTxFailures;
    /*! Total number of RX Decrypt failures. */
    uint32_t rxDecryptFailures;
    /*! Total number of TX Encrypt failures. */
    uint32_t txEncryptFailures;
    /* Total Transaction Expired Count */
    uint32_t txTransactionExpired;
    /* Total transaction Overflow error */
    uint32_t txTransactionOverflow;
    /* Total broadcast messages sent */
    uint16_t broadcastMsgSentCnt;
} Collector_statistics_t;

#ifdef IEEE_COEX_METRICS
typedef struct
{
    uint32_t dbgCoexGrants;
    uint32_t dbgCoexRejects;
    uint16_t dbgCoexContRejects;
    uint16_t dbgCoexMaxContRejects;
} Collector_coexStatistics_t;

#endif

/*************************************************************************
   *********
 Global Variables

   *************************************************************************
   *********/
```

```
/*! Collector events flags */
extern uint16_t Collector_events;

/*! Collector statistics */
extern Collector_statistics_t Collector_statistics;

extern ApiMac_callbacks_t Collector_macCallbacks;

/***********************************************************************
  *********
 Function Prototypes

  ***********************************************************************
  *********/

/*!
 * @brief Initialize this application.
 */
#ifdef OSAL_PORT2TIRTOS
extern void Collector_init(uint8_t _macTaskId);
#else
extern void Collector_init(void);
#endif
/*!
 * @brief Application task processing.
 */
extern void Collector_process(void);

/*!
 * @brief Build and send the configuration message to a device.
 *
 * @param pDstAddr - destination address of the device to send the
  message
 * @param frameControl - configure what to the device is to report
  back.
```

```
 *                      Ref. Smsgs_dataFields_t.
 * @param reportingInterval - in milliseconds- how often to report, 0
 *                            means to turn off automated reporting,
  but will
 *                            force the sensor device to send the
  Sensor Data
 *                            message once.
 * @param pollingInterval - in milliseconds- how often to the device
  is to
 *                          poll its parent for data (for sleeping
  devices
 *                          only.
 *
 * @return Collector_status_success, Collector_status_invalid_state
 *         or Collector_status_deviceNotFound
 */
extern Collector_status_t Collector_sendConfigRequest(ApiMac_sAddr_t
  *pDstAddr,
                uint16_t frameControl,
                uint32_t reportingInterval,
                uint32_t pollingInterval);

/*!
 * @brief Update the collector statistics
 */
extern void Collector_updateStats( void );

/*!
 * @brief Build and send the toggle led message to a device.
 *
 * @param pDstAddr - destination address of the device to send the
  message
 *
 * @return Collector_status_success, Collector_status_invalid_state
 *         or Collector_status_deviceNotFound
```

```
 */
extern Collector_status_t Collector_sendToggleLedRequest(
                ApiMac_sAddr_t *pDstAddr);

#if defined(DEVICE_TYPE_MSG)
/*!
 * @brief Build and send the device type request message to a device.
 *
 * @param pDstAddr - destination address of the device to send the
   message
 *
 * @return Collector_status_success, Collector_status_invalid_state
 *         or Collector_status_deviceNotFound
 */
extern Collector_status_t Collector_sendDeviceTypeRequest(
        ApiMac_sAddr_t *pDstAddr);
#endif /* DEVICE_TYPE_MSG */

#ifdef POWER_MEAS
/*! Generate data of fixed size for power measurement testing */
extern void generateIndirectRampMsg(void);
#endif

#ifdef __cplusplus
}
#endif

#endif /* COLLECTOR_H */
```

SENSOR.H - Modified File

```
/*********************************************************************
******

 @file sensor.h
```

```
 @brief TIMAC 2.0 Sensor Example Application Header

 Group: WCS LPC
 Target Device: cc13xx_cc26xx


******************************************************************************

 Copyright (c) 2016-2025, Texas Instruments Incorporated
 All rights reserved.

 Redistribution and use in source and binary forms, with or without
 modification, are permitted provided that the following conditions
 are met:

 *  Redistributions of source code must retain the above copyright
    notice, this list of conditions and the following disclaimer.

 *  Redistributions in binary form must reproduce the above copyright
    notice, this list of conditions and the following disclaimer in the
    documentation and/or other materials provided with the distribution.

 *  Neither the name of Texas Instruments Incorporated nor the names of
    its contributors may be used to endorse or promote products derived
    from this software without specific prior written permission.

 THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS"
 AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
 THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
 CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
 EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
```

```
 PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS;
 OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
 WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
 OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
 EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.


**************************************************************************
*****



**************************************************************************
****/
#ifndef SENSOR_H
#define SENSOR_H

/**************************************************************************
******
 Includes

**************************************************************************
****/

#include "smsgs.h"

#ifdef OSAL_PORT2TIRTOS
#ifndef FREERTOS_SUPPORT
#include <ti/sysbios/knl/Task.h>
#endif
#endif

#ifdef FEATURE_SECURE_COMMISSIONING
#include "sm_ti154.h"
```

```
#endif

#ifdef __cplusplus
extern "C"
{
#endif

/*********************************************************************
******
 Constants and definitions

*********************************************************************
****/

/*! Event ID - Start the device in the network */
#define SENSOR_START_EVT 0x0001
/*! Event ID - Reading Timeout Event */
#define SENSOR_READING_TIMEOUT_EVT 0x0002

#ifdef FH_LOW_LATENCY_BROADCAST
#define SENSOR_BROADCAST_CAMCMD_EVT     0x1000
#define SENSOR_BROADCAST_HEARTBEAT_EVT  0x2000
#define SENSOR_TIMEOUT_EVT              0x4000
#endif // FH_LOW_LATENCY_BROADCAST

#ifdef FEATURE_NATIVE_OAD
/*! Event ID - OAD Timeout Event */
#define SENSOR_OAD_TIMEOUT_EVT 0x0004
#ifdef OAD_IMG_A
#define SENSOR_OAD_SEND_RESET_RSP_EVT 0x0008
#endif

#endif /* FEATURE_NATIVE_OAD */
#ifdef DISPLAY_PER_STATS
/*! Event ID - Update Sensor Stats Event */
```

```
#define SENSOR_UPDATE_STATS_EVT 0x0010
#endif /* DISPLAY_PER_STATS */

#if (USE_DMM) && !(DMM_CENTRAL)
/*! Event ID - start provisioning Event */
#define SENSOR_PROV_EVT 0x0020
#endif /* USE_DMM && !DMM_CENTRAL */
/*! Event ID - Disassociate Event */
#define SENSOR_DISASSOC_EVT 0x0040

#ifdef DMM_OAD
/*! Event ID - Pause 154 Sensor */
#define SENSOR_PAUSE_EVT 0x0080
/*! Event ID - Resume 154 Sensor */
#define SENSOR_RESUME_EVT 0x0100
#endif /* DMM_OAD */

#ifdef FEATURE_TOAD
/*! Event ID - Turbo OAD Decoding Event*/
#define SENSOR_TOAD_DECODE_EVT 0x0200
#endif

/* Beacon order for non beacon network */
#define NON_BEACON_ORDER      15

/*! Clock tick period  */
#define CLOCK_TICK_PERIOD     (10)
/*! tick number for one ms  */
#define TICKPERIOD_MS_US      (1000/(CLOCK_TICK_PERIOD))

/*! Sensor Status Values */
typedef enum
{
    /*! Success */
    Sensor_status_success = 0,
```

```
    /*! Sensor isn't in the correct state to send a message */
    Sensor_status_invalid_state = 1
} Sensor_status_t;

/*************************************************************************
******
 Structures

*************************************************************************
****/

/*************************************************************************
******
 Global Variables

*************************************************************************
****/

/*! Sensor Task ID */
extern uint8_t Sensor_TaskId;

/*! Sensor events flags */
extern uint16_t Sensor_events;

/*! Sensor statistics */
extern Smsgs_msgStatsField_t Sensor_msgStats;

#ifdef POWER_MEAS
/*! Power Measurement Statistics */
 extern Smsgs_powerMeastatsField_t Sensor_pwrMeasStats;
#endif

/*************************************************************************
******
 Function Prototypes
```

```
*****************************************************************************
****/

/*!
 * @brief Initialize this application.
 *
 * @param       macTaskHndl - The MAC Task ID to send messages to.
 */
#ifdef OSAL_PORT2TIRTOS
extern void Sensor_init(uint8_t _macTaskId);
#else
extern void Sensor_init(void);
#endif

#ifdef USE_DMM
/*!
 * @brief Initialize MAC level security for this application.
 *
 * @param frameCounter - The initial frame counter
 */
extern void Sensor_securityInit(uint32_t frameCounter);
#endif /* USE_DMM */
/*!
 * @brief Application task processing.
 */
extern void Sensor_process(void);

/*!
 * @brief   Send MAC data request
 *
 * @param   type - message type
 * @param   pDstAddr - destination address
 * @param   rxOnIdle - true if not a sleepy device
 * @param   len - length of payload
```

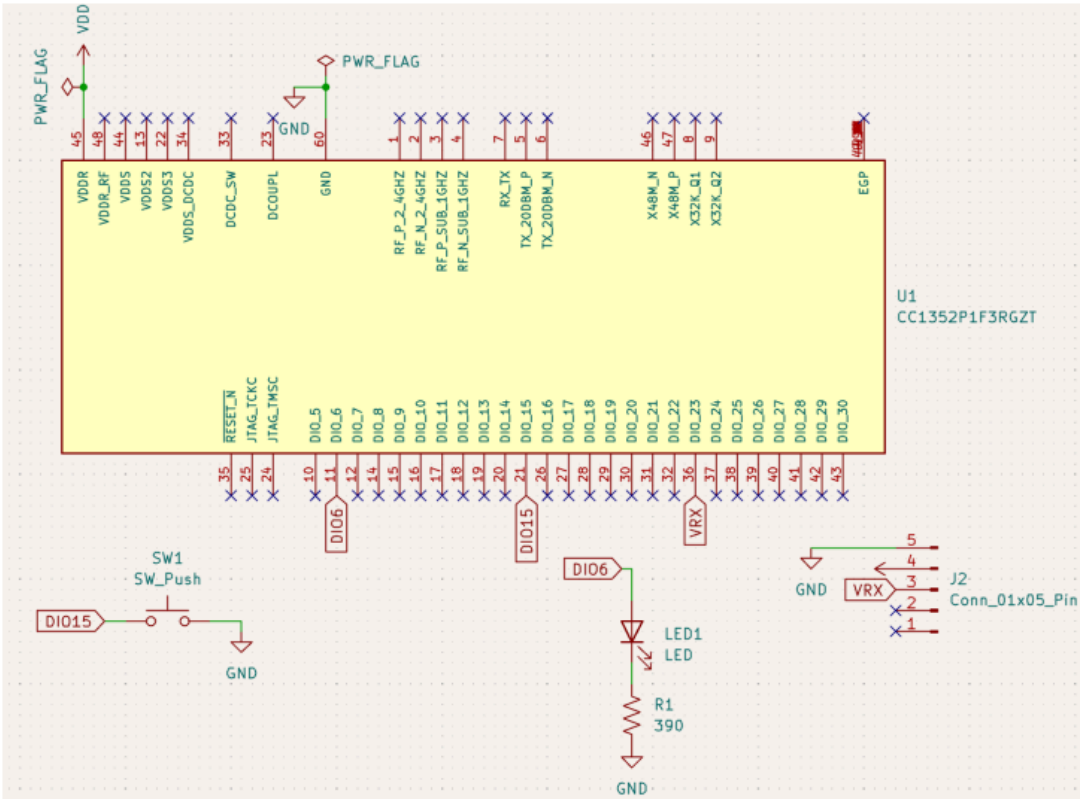
```
 * @param   pData - pointer to the buffer
 *
 * @return  true if sent, false if not
 */
extern bool Sensor_sendMsg(Smsgs_cmdIds_t type, ApiMac_sAddr_t *pDstAddr,
                    bool rxOnIdle, uint16_t len, uint8_t *pData);

/*!
 * @brief Send identify LED request to collector
 */
extern void Sensor_sendIdentifyLedRequest(void);


#ifdef FEATURE_SECURE_COMMISSIONING
/*!
 * @brief Sets the Security Authentication Mode
 *
 * @param   authMethod - Authentication Mode
 */
extern void Sensor_setSmAuthMethod(SMMsgs_authMethod_t authMethod);

/*!
 * @brief Gets the Security Authentication Mode
 *
 * @return   authMethod - Authentication Mode
 */
extern SMMsgs_authMethod_t Sensor_getSmAuthMethod(void);
#endif

#ifdef DMM_CENTRAL
/*!
 * @brief Updates BLE sensor data and forwards data to collector
 *
 * @param bleInfo - BLE sensor data
 */
```

```
extern void Sensor_forwardBleData(const Smsgs_bleSensorField_t *bleInfo);
#endif

#ifdef __cplusplus
}
#endif

#endif /* SENSOR_H */
```

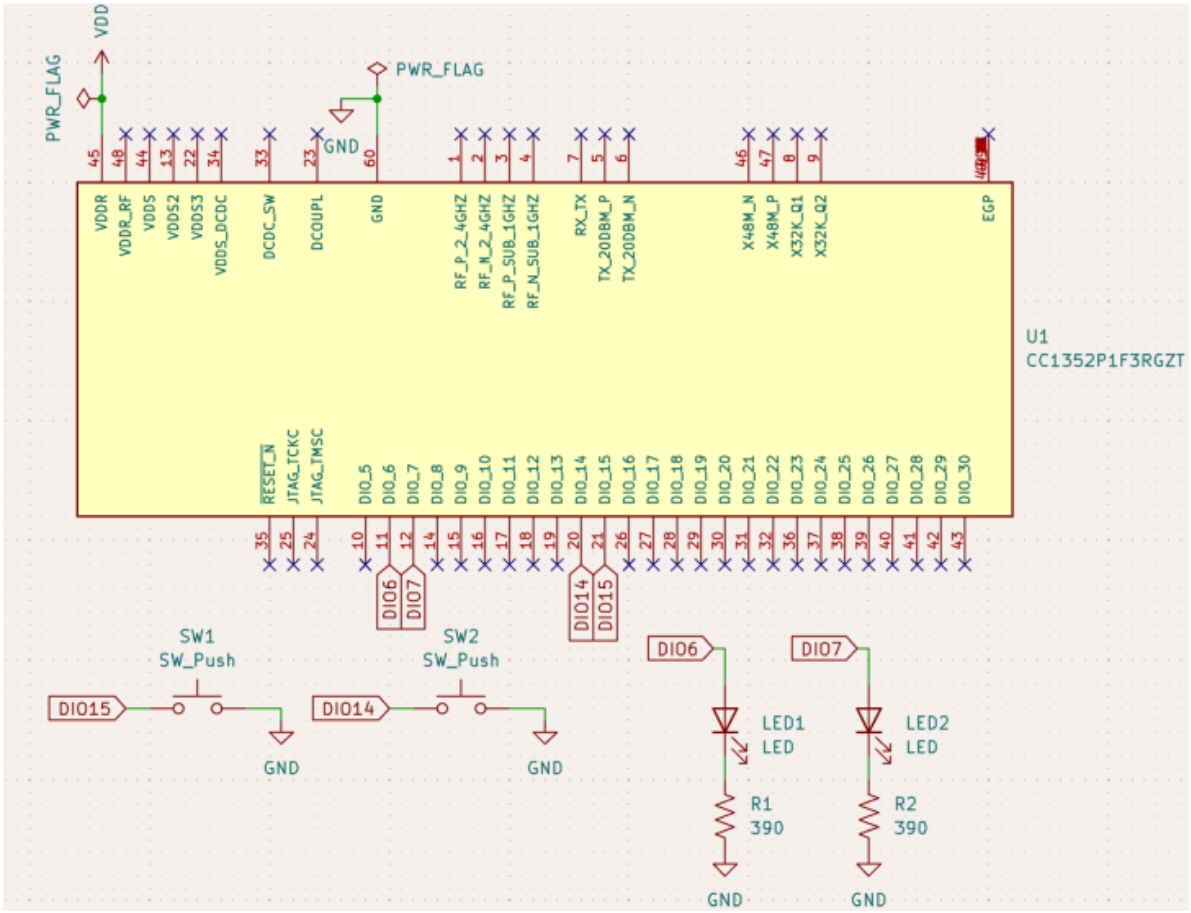
2. Block diagram and/or Schematics showing the components, pins used, and interface. You can use KiCAD/Eagle/Altium to get the schematics. KiCAD Symbol libraries for TI uCs are @ https://kicad.github.io/symbols/MCU_Texas.html and https://www.snapeda.com/parts/CC1352P1F3RGZT/Texas%20Instruments/view-part/

KiCad Design Schematics (ERC Compliant):

Sensor Board:



Collector Board:



3. Screenshots of the IDE, physical setup, debugging process – Provide screenshot of successful compilation, screenshots of registers, variables, graphs, etc. (Shown in Video)

CCS Output (Successful Compilation):

```
**** Build of configuration Release for project sensor_CC1352R1_LAUNCHXL_tirtos7_ticlang ****

"C:\\ti\\ccs1281\\ccs\\utils\\bin\\gmake" -k -j 12 all -O

gmake[1]: 'sensor_CC1352R1_LAUNCHXL_tirtos7_ticlang.out' is up to date.

**** Build Finished ****
```

```
**** Build of configuration Release for project collector_CC1352R1_LAUNCHXL_tirtos7_ticlang ****

"C:\\ti\\ccs1281\\ccs\\utils\\bin\\gmake" -k -j 12 all -O

gmake[1]: 'collector_CC1352R1_LAUNCHXL_tirtos7_ticlang.out' is up to date.

**** Build Finished ****
```

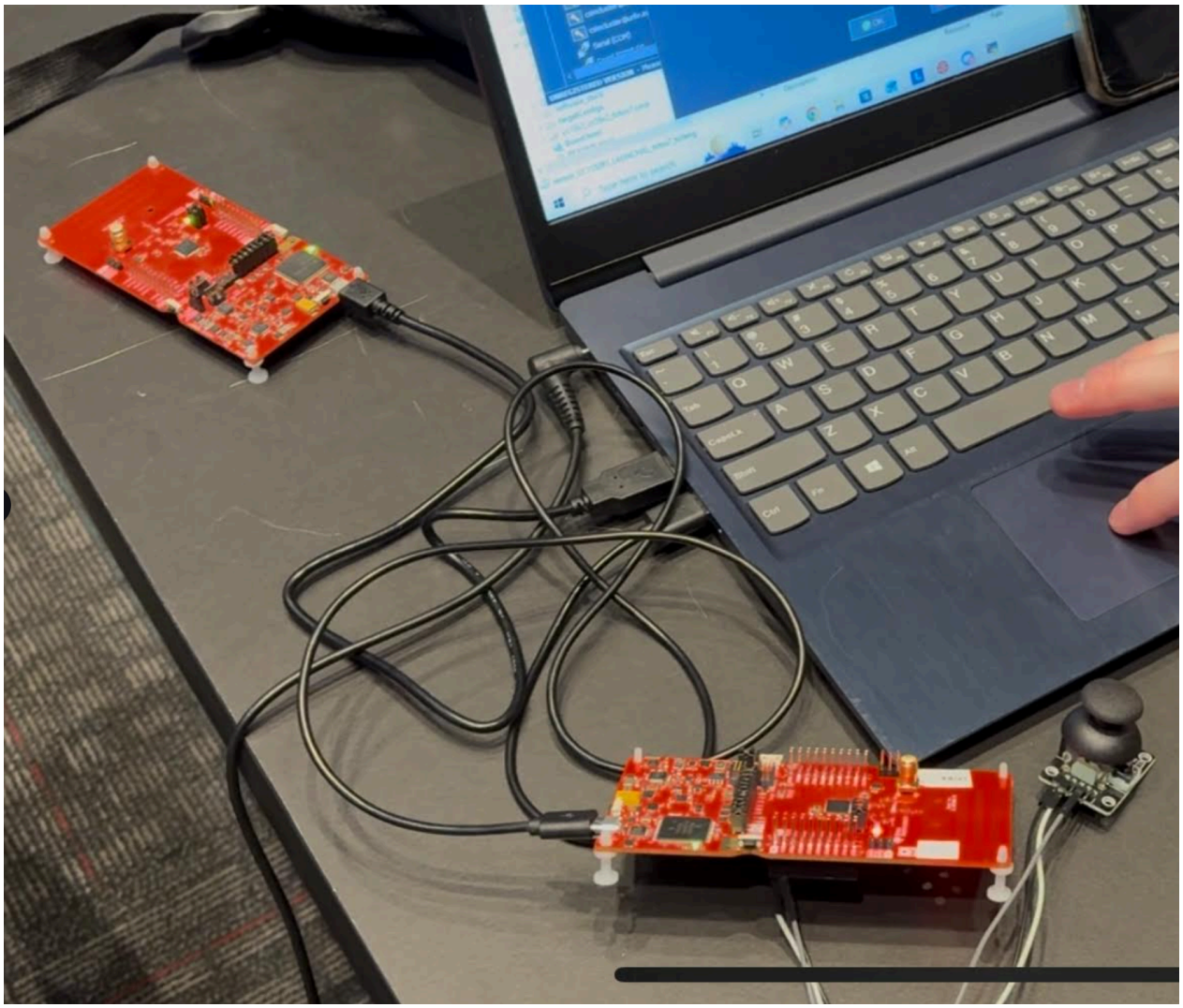
Task 1+2+4 (Sensor)

```
TI Sensor
Press Enter for Help
<      HELP      >

Status: Rejoined--Mode=NBCN, Addr=0x0006, PanId=0x0001, Ch=0
ADC Val: 1547
```

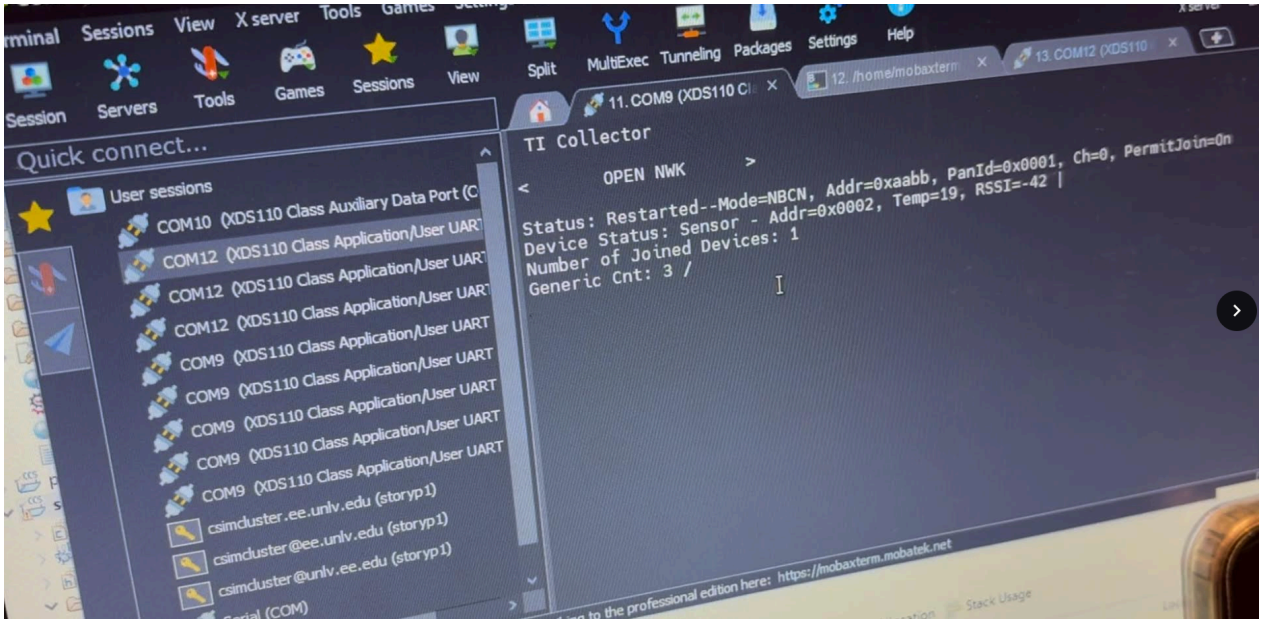
Task 1+3+4 (Collector)

```
TI Collector
Press Enter for Help
<      HELP      >

Status: Restarted--Mode=NBCN, Addr=0xaabb, PanId=0x0001, Ch=0, PermitJoin=On
Device Status: Sensor - Addr=0x0006, Temp=31, RSSI=-14 /
Number of Joined Devices: 1
ADC Val: 1545 -
```

Physical Setup:



Reading in Terminal:



4. Declaration

I understand the Student Academic Misconduct Policy - http://studentconduct.unlv.edu/misconduct/policy.html

"This assignment submission is my own, original work".
Joshua Martinez