Name: Quang Huy Le

Student ID: 20210721

EE414 Embedded Systems KOLEX Lab 4 Report

I. Code explanation

- High-level code flow

  Step 1: Configuration and initialization for pull-up resistor, VDD, gesture sensor, IMU sensor, display, on-board LEDs, clock driver, RTC interrupt, GPIO interrupt(at pins `P0.19` for Proximity Interrupt Signal) as in lab 3

  - New: configure and initialize SoftDevice (BLE connections): App Scheduler, BLE Stack, GAP, GATT, Advertise packet, BLE Services, Peer Devices Manager. The system would initialize the handlers for event interrupts which are related to above BLE processes

  Step 2: Turn the proximity mode on, power on the display and start the timers, BLE Advertising process

  - To reduce power consumption and avoid collision, the interval between 2 advertising packet transmission is an random number between 0.5s and 1s

  Step 3: Put the system in while(true) loop. In this loop, the idle state handler will be processed, also collecting the data from IMU sensor (for our own-implemented function)

  - When there is an interrupt signal, the program jumps into respective handler described specificly below

1. BLE connection event:
   - If a subscribed notification is sent to the board, the system will update the clock on the display
   - If a device is connected, the connection handle will be assigned to the Queued Writes module
2. Current Time Service client events
   - If the Current Time Service is found on the peer device, the system will send a notification enable request to the peer device. Otherwise, the system will disconnect to the peer device
   - If the current time is received, the system will update the clock on the display
3. Immediate Alert Service client events
   - If the IAS is found on the peer device, the system will assert the IAS Present flag. From now on, any time when a fall/collision is detected, a High Alert is sent to the phone and respective action (e.g. Vibration, Alert sound) will be done
4. Peer manager event:
   - If adding a peer is successful, the Database Discovery interrupt signal will be triggered
   - If the peer device is deleted successful, the system starts advertising packet again
5. Database discovery event:

- The system starts to discover the available services on the peer devices. In this program, the desired services to be discovered is Current Time Service and Immediate Alert Service

6. The GPIO interrupt handler:

   Step 1: Power on the display if it was turned off previously and stop the sleep timer handler.
   Step 2: The program delays for some time for the sensor to recognize the gesture. Once the gesture data is ready to collect (GFLVL > GFIFOTH), continue to step 3. Otherwise jump back to step 2.
   Step 3: Data acquisition The system reads the data of pulse count of 4 direction-sensitive diodes U, D, L, R and store it to memory
   Step 4: Gesture data processing.
   Step 5: Turn on the LED and display the logo or clock, normal or inverse mode accordingly to the gesture decided previously.
   Step 6: If GMODE is reset to 0 (due to automatic GMODE exit mechanism on the sensor) the program exits the loop. Otherwise jump back to step 2
   Step 7: the sleep timer starts, the program sends command to clear the Proximity Interrupt and returns from the handler

7. The clock interrupt handler:

   Step 1: Update the clock text (HH:MM:SS) and the corresponding bitmap array
   Step 2: If the display is showing the clock, update the new bitmap array on display memory. Otherwise do nothing. Then the program returns from the handler

8. The sleep timer handler:

   Step 1: Turn off the display and the programs returns from the handler

- The key implementation of the code in this lab:

  The addition implementation to the original example ble_cts_c:

  - CCCD configuration to subscribe for notification from the Current Time Service (the detail code is in the source code):

```
static uint32_t cccd_configure(ble_cts_c_t const * const p_cts,
                               bool                       enable)

uint32_t ble_cts_c_notif_enable(ble_cts_c_t * p_cts)
```

  - Addition of event BLE_GATTC_EVT_HVX to handle the event of receiving new notification from CTS on peer device
  - Own implemented function: Fall/Collision detection on smartwatch
    This functionality is based on the Inertial Measurement Unit sensor: when the linear acceleration spiked (measured value exceeds pre-set threshold), the system would detect a collision and produce an alert. This alert is sent to the phone via BLE on Immediate Alert Service for 5 seconds and then turn off the Alert.
  - Implementation of Immediate Alert Service
    The IAS has UUID 0x1802. Its characteristics on the phone is only writable(no response) and Alert Level (No Alert, Mild and High alert). When a phone is bonded to the smartwatch, the system will discover for the IAS beside CTS
    ble_ias_c_send_alert_level(&m_ias_c, BLE_CHAR_ALERT_LEVEL_HIGH_ALERT);

static void on_ias_c_evt(ble_ias_c_t * p_ias_c, ble_ias_c_evt_t * p_evt)
ble_ias_c_init(&m_ias_c, &ias_c_init_obj)
ble_ias_c_on_db_disc_evt(&m_ias_c, p_evt);
All the implemented functions are referenced to ble_ias_c.h library

II. Discussion

1. The application of own-implemented function: to detect when the user is fall (passed out or accident). When the alert is sent to the phone, this function could be further developed by sending messages to trusted people or emergency service
2. Difficulties in implementation of the functionality

- The IMU sensor is used for the fall detection, hence the datasheet of this device needs to research carefully to operate the sensor properly
- The raw data of the IMU sensor needs to be processed thoroughly since it only measure the gravity exerted on the sensor. We needs the linear acceleration of the motion, so the raw data needs to be processed

III. References

- EE414 Lab 4 documentation
- Arduino Nano 33 BLE Schematic
- APDS-9960 sensor datasheet
- SSD1306 OLED display datasheet
- LSM9DS1 datasheet
- The Dot Factory Program for conversion from bitmap image and text to mapped array
- Nordic semiconductor devzone
- Libraries in SDK: `ble_hrs_c, ble_ias_c, ble_cts_c`