Detailed Description, Specifications

Note : This page will be updated continuously to provide clarifications regarding specifications and requirements. Please keep checking this page once in a while.

********** Updates are in pruple colour **********

(0) About the report: it should be 10 pages including everything.

(1) COMMON QUESTIONS RECEIVED (KEEP UPDATED)

Q1: What if there are multiple warnings triggered and require different LED blinking frequencies?

Ans: Please choose the faster blinking speed if more than one sensors trigger the warnings.

Q2: What should happen if 2 of the sensors with the same blinking speed are triggered at different moments?

Ans: You can keep blinking with that frequency from the 1st warning generated by one sensor.

Q3: In Intensive Care Mode, does the LED blink to its respective frequency when a warning happens, or is it always on?

***** Ans: For all the possibilities never mentioned in the Assignment 2 descriptions, you can decide what to do. As mentioned on the description page: "For a project of this nature, it is impossible to have the specifications cover all possible scenarios. We leave it to your discretion to decide how the system should respond in scenarios not covered by the above specifications", so, think as a "product engineer" and choose your own way to serve your customers who are going to use your system, and design it appropriately 🙂**

(2) There is "HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_14);" in the EXTI15_10_IRQHandler() in the example program demo_exti_print.zip, which was used as an indicator to verify that interrupt handler is called automatically. If you are using demo_exti_print.zip to start your assignment 2, please remove it in stm32l4xx_it.c

(3) Note that Systick interrupt is compulsory to be used for ALL timing requirements. HAL_Delay() function or counting the regular routine is FORBIDDEN to be used for timing in the final program.

(4) In conclusion, "wait by doing nothing" is FORBIDDEN.

(5) printf() function is FORBIDDEN in the final program.

DEVICES USED

- ACCELEROMETER: 3D accelerometer LSM6DSL
- GYROSCOPE: 3D gyroscope LSM6DSL
- HUMIDITY_SENSOR: capacitive digital sensor HTS221
- TEMPERATURE_SENSOR: capacitive digital sensor HTS221
- LED: Green LED2
- MAGNETOMETER: high-performance 3-axis magnetometer LIS3MDL
- PRESSURE_SENSOR: 260-1260 hPa absolute digital output barometer LPS22HB
- MODE_TOGGLE & WARNING_CLEAR: USER Button (blue), read using interrupts
- CHIPACU: Terminal program (Tera Term) on a personal computer/laptop displaying Telemetry
- DEBUG_DEBUG_CONSOLE: The debug DEBUG_CONSOLE in STM32CubeIDE where printf() and other messages appear. These messages have to be disabled as soon as we start using CHIPACU
- *WIFI/BLUETOOTH: Telemetry messages to the cloud via WiFi/Bluetooth (Optional)*

FEATURES

Modes of Operation

- Two modes of operation of the monitoring system:
    - **Normal**
    - **Intensive Care**
- When the system is turned ON, the system is in **Normal** mode.
- The system switches from **Normal** mode to **Intensive Care** mode only when there is a warning generated from Respiratory Monitoring (refer to Respiratory Monitoring section below).
- To recover the **COPEMON** back from **Intensive Care** mode to **Normal** mode, press MODE_TOGGLE *twice* within 0.5 seconds (i.e., there is an additional press within the 0.5 seconds window following the first press). **COPEMON** enters the **Normal** mode after the second press of MODE_TOGGLE.
- Pressing MODE_TOGGLE once per second when the system is in **Intensive Care** mode has no effect.

Body Temperature Monitoring

- Essential to detect body temperature and trigger an emergency response.
- Use the TEMPERATURE_SENSOR to detect the body temperature and provide the following response when TEMPERATURE_SENSOR's reading is greater than TEMP_THRESHOLD is detected.
    - Show a warning message "***Fever is detected*\n**" ~~on the DEBUG_CONSOLE~~ and send through Telemetry.
    - Set the LED blinking at 5Hz.
- Body temperature monitoring should be done in **both Normal and Intensive Care modes**.
- In Normal mode, temperature readings need not be sent to **CHIPACU**, and temperature warning is enabled and sent through Telemetry every 10 seconds until the system's mode is changed.
- In Intensive Care mode, both temperature reading and warning (if any) need to be sent to **CHIPACU** every 10 seconds.

Fall Detection

- Essential to detect body fall and trigger emergency responses.
- Use the accelerometer to detect fall and provide the following responses when ACCELEROMETER's reading is lower than ACC_THRESHOLD is detected.
    - Show a warning message "***Fall detected*\n**" ~~on the DEBUG_CONSOLE~~ and send through Telemetry.
    - Set the LED blinking at 5Hz.
- Fall detection should be done in **both Normal and Intensive Care modes**, as the possibility of a fall exists in both modes.
- In Normal mode, ACCELEROMETER's readings need not be sent to **CHIPACU**, and the warning is permanently enabled and sent through Telemetry 10 seconds unless the system's mode is altered.
- In Intensive Care mode, both ACCELEROMETER's reading and warning (if any) need to be sent to **CHIPACU** 10 seconds.

Pain Detection

- Pain Detection is enabled when the system is used by hospitalized COVID-19 patients. A sudden movement can cause issues with the other monitoring / life-saving equipment such as ventilators.
- Use the GYROSCOPE to detect the patient's movement, i.e. sense the patient's sudden twisting/twitching which is an indication that the patient is in pain.
- provide the following responses when GYROSCOPE's reading (combinational effect in X, Y, and Z directions) greater than GYRO_THRESHOLD is detected.
    - Show a warning message "***Patient in pain! \n***" ~~on the DEBUG_CONSOLE~~ and send through Telemetry.
    - Set the LED blinking at 10Hz.
- Pain Detection is **only enabled in Intensive Care mode**. Both GYROSCOPE's reading and warning (if any) need to be sent to **CHIPACU** every 10 seconds.

Orientation Monitoring

- Orientation monitoring is used by hospitalized COVID-19 patients to detect the orientation of the patient lying on the bed.
- Use the MAGNETOMETER to sense the patient's abnormal orientation during hospitalization.
- Provide the following responses when MAGNETOMETER's readings hit the MAG_THRESHOLD.
    - Show a warning message "***Check patient's abnormal orientation! \n***" ~~on the DEBUG_CONSOLE~~ and send through Telemetry.
    - Set the LED blinking at 10Hz.
- Orientation Detection is **only enabled in Intensive Care mode**. Both MAGNETOMETER's readings and warnings (if any) need to be sent and updated to **CHIPACU** every 10 seconds.

Respiratory Monitoring

- Respiratory monitoring is used by hospitalized COVID-19 patients.
- Two sensors are used: (1) HUMIDITY_SENSOR is to detect the breath out exhale air from the patient, and (2) PRESSURE_SENSOR is used to simulate the measurement of air pressure in the patient's lungs.
- Provide the following responses when HUMIDITY_SENSOR's reading is below the HUMID_THRESHOLD, **or** when PRESSURE_SENSOR's reading exceeds the PRESSURE_THRESHOLD.
    - Show a warning message "***Check patient's breath! \n***" ~~on the DEBUG_CONSOLE~~ and send through Telemetry.
    - ~~Set the LED blinking at 10Hz.~~ **The system should enter the Intensive Care Mode.**

- Respiratory Monitoring is used in both **Normal** and **Intensive Care** modes. During Normal mode, the readings from HUMIDITY_SENSOR and PRESSURE_SENSOR need not be sent, but if a warning message is generated, the system should switch to **Intensive Care** mode after the warning message is sent.
- During **Intensive Care** mode, HUMIDITY_SENSOR's reading, PRESSURE_SENSOR's reading, and warning (if any) should be permanently enabled and sent through Telemetry every 10 seconds if the threshold is met.

Telemetry

- Various messages and data are sent to **CHIPACU**, to monitor the patient's vital conditions. The following information is sent.
    - Messages regarding entering Normal and Intensive Care modes.
    - Sensors' readings and warning messages (if any) whenever they occur are also sent. Some examples are below.
        - "***Fall detected.*** \r\n" - when a fall is detected.
        - "***Fever is detected***. \r\n" warning - when the body temperature exceeds the threshold.

MODES

Normal Mode

COPEMON must be on a horizontal surface when powered on for the first time. Upon powering on, **COPEMON** must have the following Normal mode behaviors:

- LED_2 is off
- The TEMPERATURE_SENSOR, ACCELEROMETER, HUMIDITY_SENSOR, and PRESSURE_SENSOR are enabled.
- The GYROSCOPE, MAGNETOMETER are idle.
- The following message is sent once to **CHIPACU** **each time** Normal mode is entered:

    Entering Normal Mode.\r\n
- Once a certain threshold is exceeded and the warning is raised, the system remains in a state of warning, and you need not detect threshold crossing again.

If a respiratory warning happens during the Normal Mode, COPEMON goes into the Intensive Care MODE, **and the warning messages and blinking of LED2 stop.**

Intensive Care Mode

Intensive Care Mode for **COPEMON** represents the situation where the patient is diagnosed with COVID 19 in need of detailed care. As soon as COPEMON enters the Intensive Care Mode, all the sensors are active. The following behavior occurs in Intensive Care Mode:

- The following message is sent once to ***CHIPACU*** **each time** Intensive Care mode is entered:

    Entering Intensive Care Mode.\r\n
- LED_2 should be always on.
- The TEMPERATURE_SENSOR, ACCELEROMETER, GYROSCOPE, MAGNETOMETER, HUMIDITY_SENSOR, and PRESSURE_SENSOR are **sampled** regularly ~~once every second~~.
- The format of the transmitted data to ***CHIPACU*** should be as follows:

    XXX TEMP_ttt.tt (degreeC) ACC_x.xx(g)_y.yy(g)_z.zz(g) \r\n

    XXX GYRO nnn.n() MAGNETO X.XX() Y.YY() Z.ZZ() \r\n

    XXX HUMIDITY h.hh() and PRESSURE p.pp() \r\n

    - XXX represents a 3-digit value that starts from 000 and increments by 001 after each transmission to ***CHIPACU***, from **COPEMON**.
    - XXX never resets itself to 000, unless **COPEMON** itself is powered on from a power-off state. It is assumed that 999 will never be reached.
    - ttt.tt stands for the temperature reading from the sensor up to 2 decimal places.
    - x.xx is the x-axis reading from the accelerometer, in 'g's, up to 2 decimal places. Similarly, y.yy and z.zz are the accelerometer's readings in 'g's on y and z axes. Note : 1 g = 9.8 m/s^2.
    - nnn.nn stands for the Gyroscope reading from the sensor up to 2 decimal places, it should be a combinational gyroscope reading by considering all 3 directions.
    - where X.XX, Y.YY and Z.ZZ are readings from Magnetometer sensor on x, y, and z axes to 2 decimal places.
    - h.hh stands for Humidity sensor reading and b.bb stands for Barometer reading to 2 decimal places.
    - You must indicate the unit in () for all the readings from different sensors.
- If high fever or fall is detected at the instant a scheduled transmission to **CHIPACU**, the following message must also be sent before the usual sensor values from the TEMPERATURE_SENSOR and ACCELEROMETER:

    Fever is detected.\r\n

    Fall detected.

- If the patient is in pain, in abnormal orientation, or has a respiratory problem at the instant a scheduled transmission to **CHIPACU** is happening, the following message must also be sent before the usual sensor values from the GYROSCOPE, MAGNETOMETER, HUMIDITY, and PRESSURE SENSOR:

  Patient in pain! \r\n

  Check patient's abnormal orientation! \r\n

  Check patient's breath! \r\n

- It is up to the students to determine which warning messages appear first if the events happen simultaneously.
- Once a certain threshold is exceeded and the warning is raised, the system remains in a state of warning, and you need not detect threshold crossing again.
- Transmission of the current readings from all the sensors as well as any triggered warning messages to **CHIPACU** occurs every 10 seconds.
- If MODE_TOGGLE is pressed twice within 0.5 seconds during Intensive Care Mode, **COPEMON** returns to Normal Mode, and the warning messages and blinking of LED2 (if any) should stop.

Specifications

- Students have the freedom to set appropriate thresholds for TEMP_THRESHOLD, ACC_THRESHOLD, GYRO_THRESHOLD, MAG_THRESHOLD, HUMID_THRESHOLD, PRESSURE_THRESHOLD.
- One more interrupt (in addition to SysTick and USER Button), preferably from a sensor, should be used. The interrupt to be used, as well as the use case of the interrupt, is left to the discretion of the student. Use of additional interrupts is optional (over and above the three interrupts mentioned above), not a basic requirement.
- Printf function (Printing) on Debug console should be **disabled** in the final program and information communication should be through Teraterm (Telemetry).

Note : Please do not damage the board by subjecting it to mechanical stress.

GENERAL CLARIFICATIONS & HINTS

- You are required to sketch flowcharts before starting assignment 2 coding. These flowcharts should be included in your report. Having a systematic plan through flowcharts/UML diagrams etc can make your coding easier and more systematic, and this systematic design methodology is practiced widely in the industry. Writing code without a proper design/plan takes a lot more time than it should. Basically, don't think and write code at the same time. Think, and then write code to express your thoughts/logic.
- The exact format displayed on Teraterm (Telemetry) isn't important. The format and messages are just suggestions. Feel free to make *reasonable* modifications.
- Make sure that you choose appropriate thresholds which can be triggered during demo/testing without subjecting the board to unnecessary mechanical stress. While you are free to set the various thresholds based on experimentation (i.e., empirically), you should be able to explain it in terms of the physical quantities involved.
- All display of information must be done through UART or other communication devices on the board (e.g., Bluetooth/WiFi). Use of printf() which makes use of the debugger (causing unpredictable delays) in the final program will be penalized, and it will cause timing issues.
- During debugging, make use of the debug features in the IDE effectively, such as running until a breakpoint, stepping over, stepping into, and inspecting various variable/array values without having to print them.
- For a project of this nature, it is impossible to have the specifications cover all possible scenarios. We leave it to your discretion to decide how the system should respond in scenarios not covered by the above specifications.