Lab 2 – Use of sensors on B-L475E-IOT01A

The purpose of this Lab is to explain how to measure with sensors available in the STM32L4 Discovery kit.

The main sensors available in the **STM32L4 Discovery kit IoT node (B-L475E-IOT01A)** are **:**

- Capacitive digital sensor for relative humidity and temperature (HTS221)
- 260-1260 hPa absolute digital output barometer (LPS22HB)
- 3D accelerometer and 3D gyroscope (LSM6DSL)
- High-performance 3-axis magnetometer (LIS3MDL)

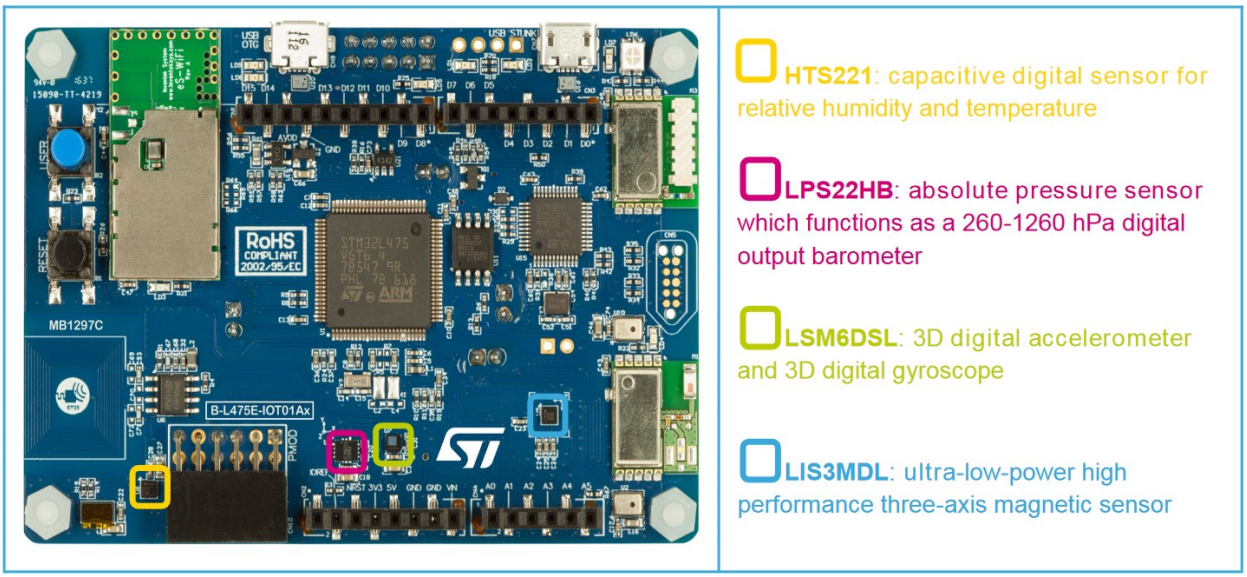

Figure 1. B-L475E-IOT01A on-board sensors

The following sections provide a step-by-step guide of the way we can interact with the HTS221 sensor to get temperature values and display them on a terminal. This logic can extended for all sensors included in the MCU.

1. IoT node - Project Setup

Firstly, create a new project, under the name Lab 2. You can reuse one of the setups you created in previous lab exercises. In this lab you will transfer the environmental temperature, as recorded from the node, to the hub. Optionally, you can use the cryptographic mechanism you implemented at the last lab.

2. IoT node – Import drivers to your project

The BSP (Board Support Package) drivers are available in the STM32CubeL4 package. It provides APIs corresponding to the hardware components of a board.

Here are the steps to follow in order to copy the BSP drivers into your project:

1. **Download** the drivers from the following website (the file is also available in the Handout folder):
   https://os.mbed.com/teams/ST/code/BSP_B-L475E-IOT01//archive/bfe8272ced90.zip

2. Unzip and drag-and-drop the BSP_B-L475E-IOT01-bfe8272ced90 folder in your project.

3. Define Node's Behaviour

Fill the question marks accordingly in order to:

1. Include the header file form the BSP package that concerns the temperature sensor
2. Initialize the sensor
3. Read temperature
4. Add the value to the buffer elements

The temperature value returned from the sensor is in float form. In order to use the buffer structure you could:

- Split that value to the whole part and the decimal value
- The whole part will go to the first element of the buffer, in uint8 form
- The decimal part will go to the second element of the buffer, in uint8 form

4. Setup Matlab interface

Using the setup from the previous labs:

1. Receive the data
2. Apply decryption, if you have send encrypted data
3. Reconstruct the information from the buffer's bytes
4. Create a dynamic display:
   a. Add the temperature information to an array – y axis
   b. Add the current time information to an array – x axis
   c. Plot the two arrays. Update them on every data received.

Appendix

main.cpp (STM32 Code)

```
#include "aes.h"
#include "mbed.h"

// Include temperature driver from BSP
????

// Maximum number of element the application buffer can contain
#define MAXIMUM_BUFFER_SIZE 16

// Create a DigitalOutput object to toggle an LED whenever data is received.
static DigitalOut led(LED1);

// Create a BufferedSerial object with a default baud rate.
static BufferedSerial serial_port(USBTX, USBRX);

int main(void) {

  //Initialize temperature sensor
  ????

  // Set desired properties (9600-8-N-1).
  serial_port.set_baud(9600);
  serial_port.set_format(
     /* bits */ 8,
     /* parity */ BufferedSerial::None,
     /* stop bit */ 1);

  // Application buffer to receive the data
  uint8_t buf[MAXIMUM_BUFFER_SIZE] = {0};
  uint8_t trigger[MAXIMUM_BUFFER_SIZE] = {0};

  while (1) {

    float sensor_value = 0;

     // Get temperature value from sensor
    sensor_value = ????;

    // Create a variable to extract the integer part
    uint8_t tmpInt1 = ????
```

```
    // Create a variable to extract the fractional part
    float tmpFrac = ????;

    uint8_t tmpInt2 = trunc(tmpFrac * 100);

    // Add the data to buffer structure
    ???? = tmpInt1;
    ???? = tmpInt2;

    // serial_port.write(buf, sizeof(buf));

    //uint8_t ciphertext[16];

    if (uint32_t num = serial_port.read(trigger, sizeof(trigger))) {

      // Toggle the LED.
      led = !led;

      serial_port.write(buf, sizeof(buf));
    }
  }
}
```

serial_comm.m (MATLAB code)

```
%% Script to read constantly the UART sychronously
clear all
close all
delete(instrfind)

% Init UART
COM_PORT_NUMBER = "????"; %
baud_rate = 9600;
data_bits = 8;
myComPort = serial(COM_PORT_NUMBER,'BaudRate',baud_rate,'DataBits',data_bits);

% Init variables
datain = '';
dataout = '1';
count = 1;

% Open communication channel
fopen(myComPort);

figure
title('Environmental Temperature');

% Name x and y-axis
????
????
% Keep plot
```

```
% Keep plot
????

% Infinite While-loop
while (1)
   % Send one or more characters to the MCU to trigger it
   fprintf(myComPort,dataout);
   % Pause for a while to slow down the exchange
   pause(1);
   % Read the available UART data as characters and store the result
   datain = fread(myComPort,????,'uint8');

    % append temperature to temps - remember to concatenate the bytes
   temps(count)= ????;

   % append the current time to times
   times(count) = ????;

   % plot temps vs times
   plot(????);

   count = count + 1;

end
```

Αναφορά Εργαστηρίου 3

Για τη βαθμολόγησή σας είναι υποχρεωτική η παράδοση αναφοράς, η οποία θα έχει αυστηρή προθεσμία την ημέρα του επόμενου εργαστηρίου.

Η αναφορά θα πρέπει να περιέχει αναλυτικά όλα τα βήματα που ακολουθήσατε στο εργαστήριο και τη δικαιολόγησή τους.

Επιπλέον ερωτήσεις για την αναφορά του εργαστηρίου 3

1. **Ποια είναι η ακρίβεια του αισθητήρα HTS221;**
2. **Δοκιμάστε πειραματικά και μετρήστε τη μέγιστη δυνατή συχνότητα δειγματοληψίας και απεικόνισης της θερμοκρασίας.**
3. **Διαμορφώστε την εφαρμογή με τέτοιο τρόπο ώστε να λαμβάνονται υπόψη αρνητικές θερμοκρασίες.**
4. **Υλοποιήστε την μέτρηση της υγρασίας και απεικονίστε τη στο MATLAB.**
5. **Υλοποιήστε στο MATLAB μια εφαρμογή η οποία να υπολογίζει την περιβαλλοντική άνεση (thermal comfort or human comfort) δεδομένης της υγρασίας και της θερμοκρασίας. Γενικές πληροφορίες: https://en.wikipedia.org/wiki/Thermal_comfort**
   Χρησιμοποιήστε το "Relative Humidity vs Air Temperature" από την σελίδα https://comfort.cbe.berkeley.edu/