National University of Singapore

College of Design and Engineering

EE2028: Microcontroller Programming and Interfacing

Academic Year 2023/2024 Semester 1

Assignment 2

Group/Team: B03/74

A/P: Qingqing Ni

Student Name	Matriculation Number
NUR SARAH BINTE ASMADI	A0254664X
VAN DER HORST NIGEL SEBESTIAAN	A0253455A

Introduction and objectives

Introduction

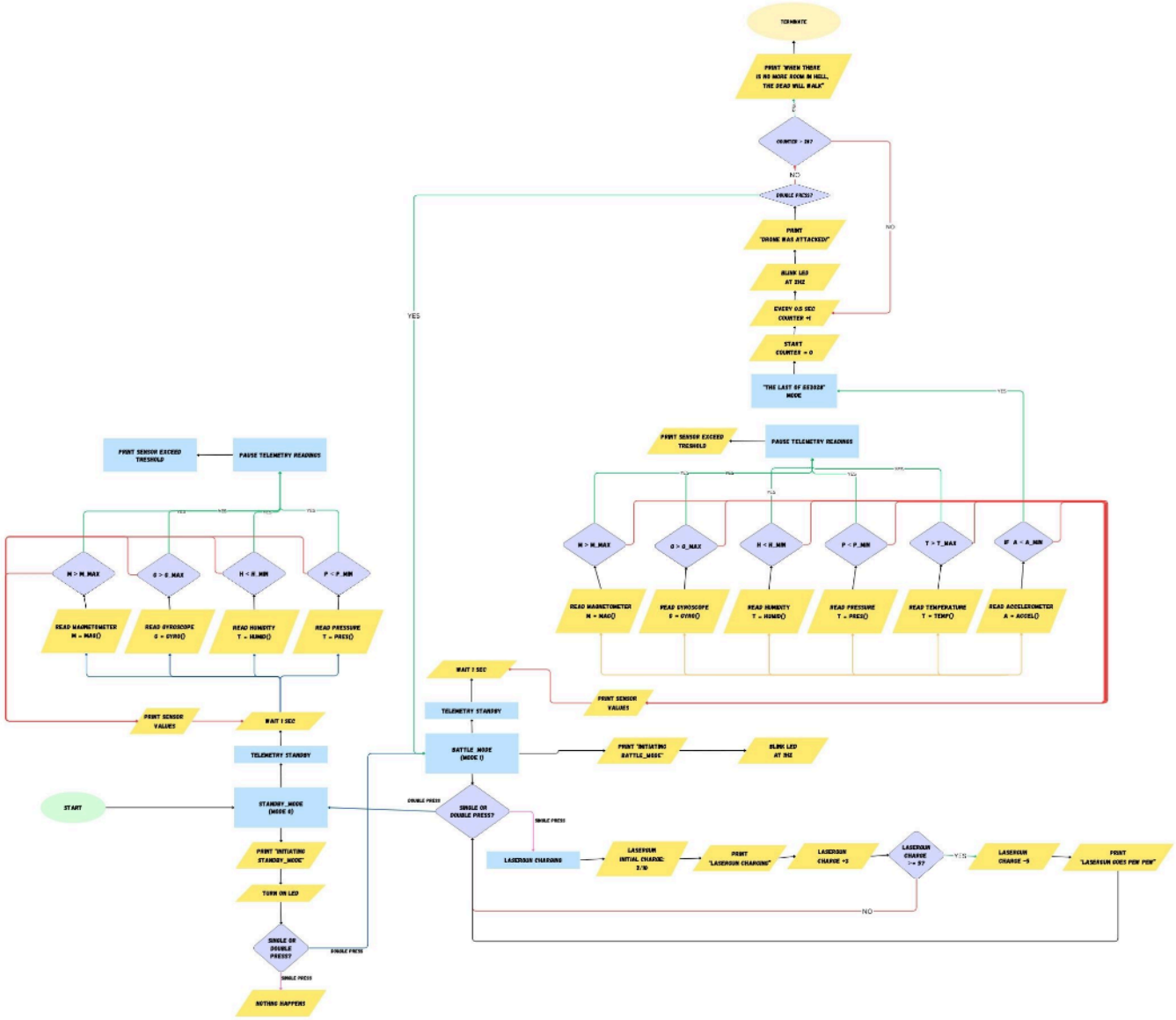
Using 6 sensors and 2 enhancements, we have created an effective drone to counter zombie infection. The sensors allow us to explore the environment and determine if it is safe for humans to live in. This drone has been coded for battle and scouting, using features such as push button interrupts for shooting lasers and systick interrupts for telemetry reading. With the enhancements to our drone, it can provide quick data for field engineers and easy maintenance.

Objectives

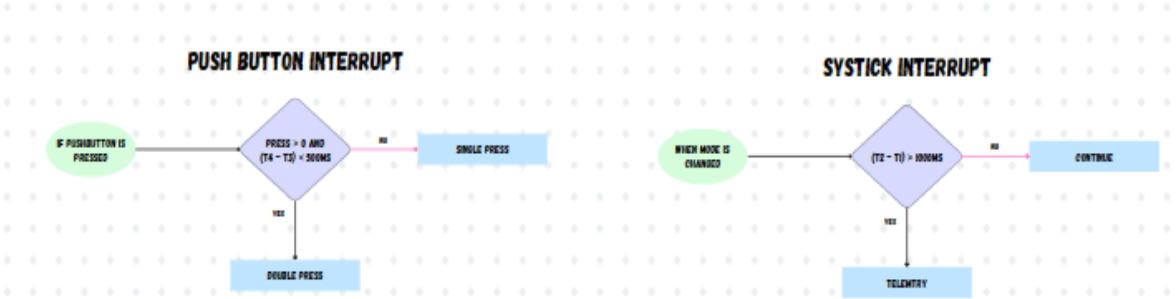
Accelerometer	Used to measure the z acceleration and detect if the drone is upside down and therefore “attacked” (< -6 m/s^2 indicates drone is upside down) and trigger The Last of EE2028. Measured in m/s^2.
Gyroscope	Used to measure the angular velocity (rotation) in all the 3 dimensions. Take the overall magnitude of all dimensions to find if the drone is suffering bad turbulence (> 5 rad/s indicates a chance of crashing). Measured in rad/s
Magnetometer	Used to detect magnetic fields and can therefore detect dangers for humans (eg: bombs) (> 500 uT indicates a chance of a bomb present). Measured in uT.
Temperature Sensor	Used to measure surrounding temperatures. Check if the temperature of the surrounding is less than 40 deg C (> 40 deg C indicates too hot for humans to live) . Measured in deg celsius
Pressure Sensor	Used to measure the atmospheric pressure. The atmospheric pressure can indicate the height of the drone or the current weather. If the measured pressure is less than 1000 mb (millibar), it indicates that the drone is too high and may lose connection to the base or indicates a chance of a storm coming. Measured in mb.
Humidity Sensor	Used to measure the relative humidity. Check if relative humidity is more than 50%rH.(< 50%rH means not enough humidity for humans, allowing for virus to spread more easily). Measured in %rH
OLED Display	The OLED Display allows for minimal level information display on the drone without a serial connection. It is able to display the 3 operating modes of the drone, Lasergun energy level, Lasergun firing, Last of ee2028 countdown and final message. This allows us to do low level troubleshooting without connecting to the drone.
Buzzer	The buzzer gives the drone sound feedback for warnings. The buzzer is activated when the lasergun is fired and during the countdown of the last of ee2028.

Flowcharts describing the system design and processes

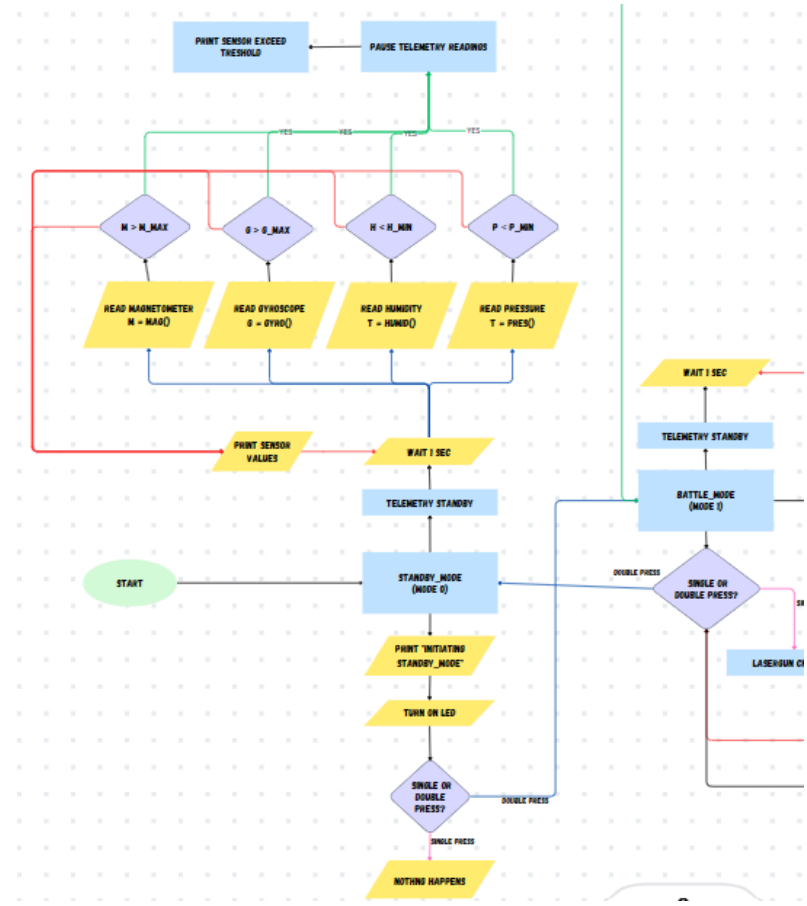
Overall view of flowchart:



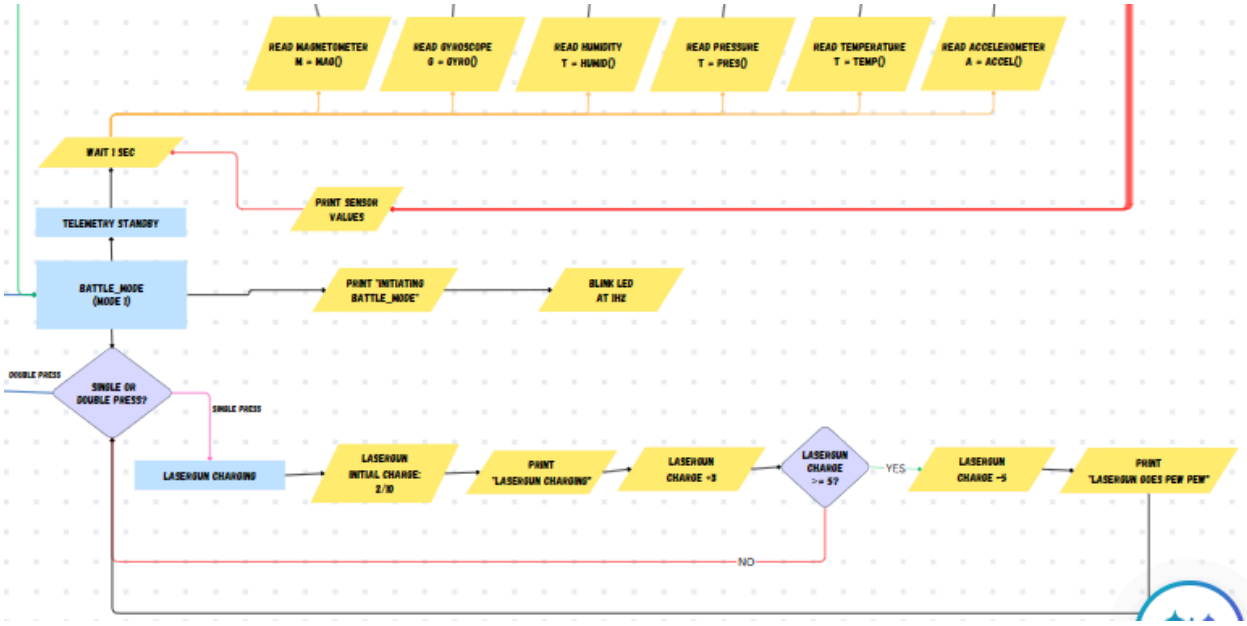
Pushbutton and Systick interrupt:



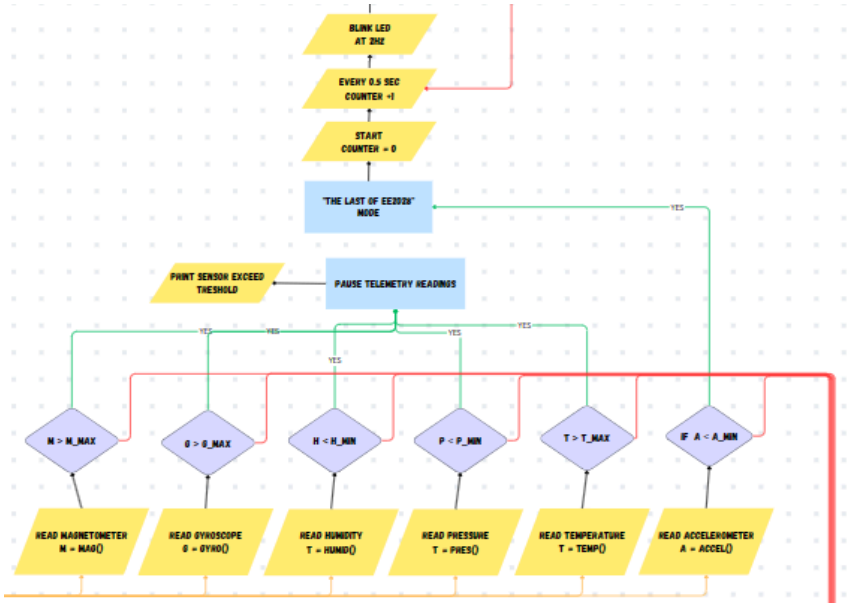
Standby mode flowchart:



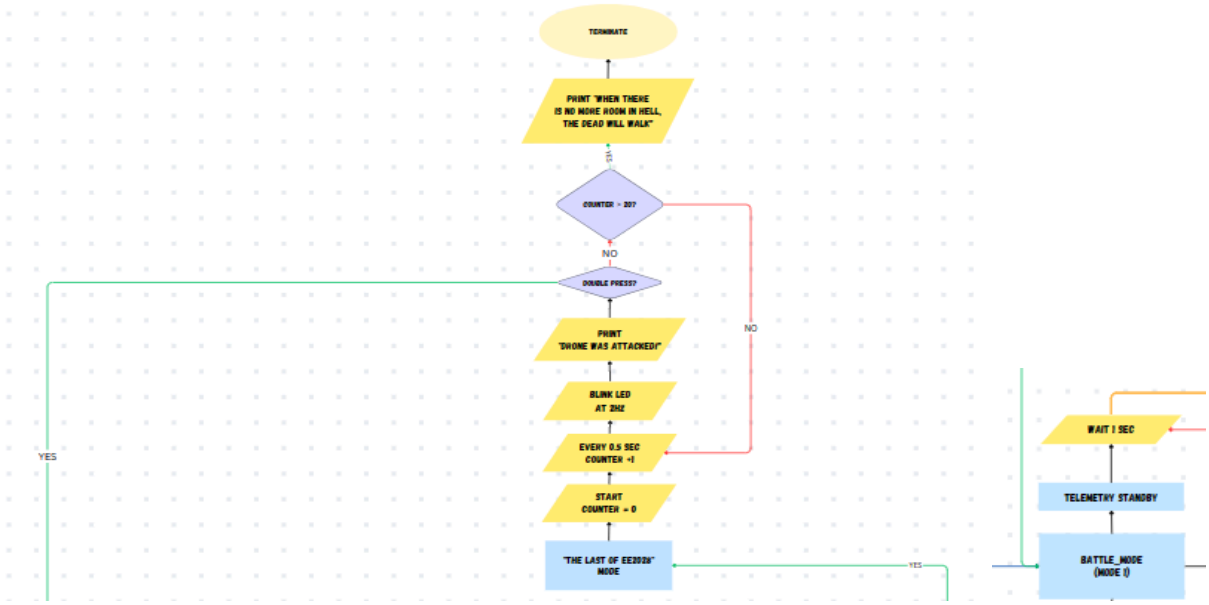
Battle Mode, Lasergun charging:



Battle Mode, Telemetry readings:



The Last of EE2028:



Detailed implementation: Describe the detailed implementation steps, especially indicate and explain your essential steps

For the sensors, we can use the BSP drivers to get the value and format them accordingly.

Accelerometer

The value obtained from the BSP function is in mg. We would need to convert it to m/s^2 by taking mg * (9.8/1000). We then only return the Z acceleration to detect if the drone is upside down and therefore attacked.

Gyroscope

The value obtained from the BSP function is in mdeg. We would need to convert it to rad/s by taking mdeg * 1/1000 * pi/180. We then return the overall magnitude. This allows us to detect if the drone is undergoing turbulence.

Magnetometer

The value obtained from the BSP function is in mgauss. We would need to convert it to uT by taking gauss * 1/1000. We then return the overall magnitude. This allows us to detect magnetic fields near the drone. This can indicate a potential bomb nearby thus allowing the drone to scout for landmines.

Temperature Sensor

The value obtained from the BSP function is in degC. We can use this reading to determine if the surrounding temperature is too hot for humans.

Pressure Sensor

The value obtained from the BSP function is in hPa. We would need to covert it to mb, it is a 1 to 1 conversion ratio.We can use this reading to determine the drone height or if a storm is impending.

Humidity Sensor

The value obtained from the BSP function is in %rH. We can use this reading to determine the relative humidity is sufficient for humans to live in.

Systick Interrupt

We take two ticks and measure the difference between them, if the difference is more than the

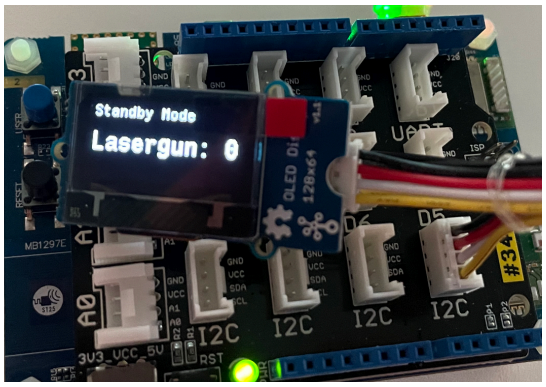
“delay”, we can execute the code. This prevents the board from idling and doing nothing during the interrupt which adds latency.

Pushbutton Interrupt

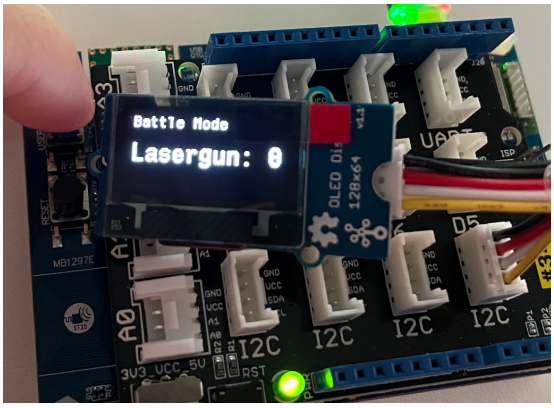
We take two ticks and measure the difference between them, if the difference is less than the “delay”, the button is single press and we set the flag to 1. If the difference in ticks is more than the “delay” and the flag is more than 0, the button is double press and the flag is set back to 0.

Enhancement: If you have implemented any enhancement, give a detailed description. You might consider including several photos of your working board at some special steps. This will help to distinguish your system and report from others

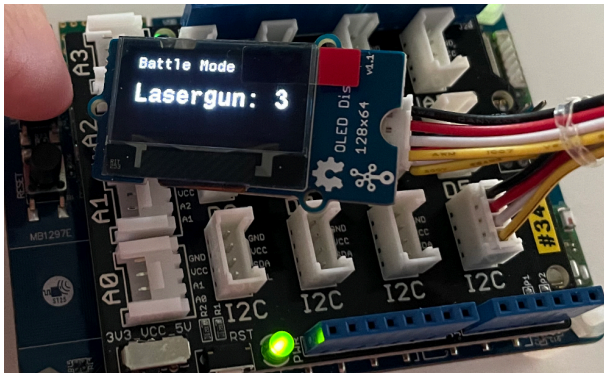
OLED Display



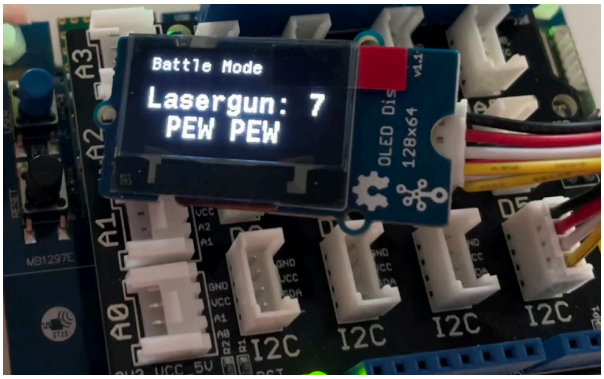
Standby Mode



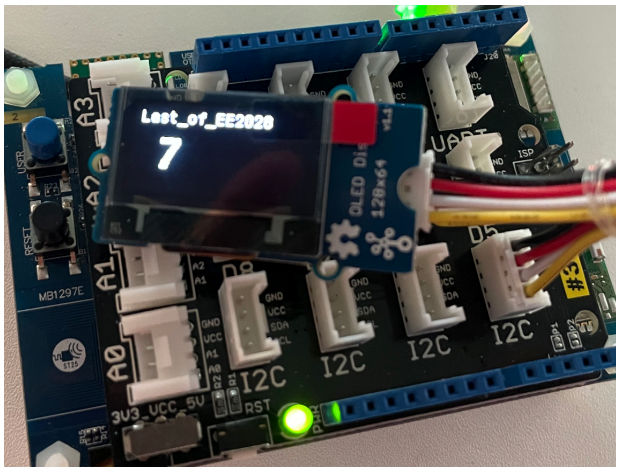
Battle Mode

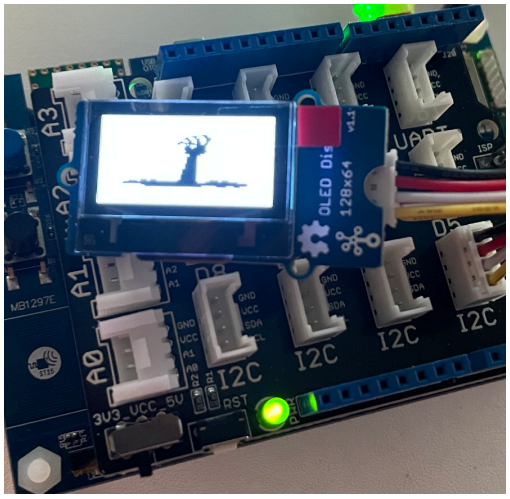


Laser gun charging



Laser gun firing





Last of EE2028 Countdown Final Message

The OLED is using I2C1 to communicate with the board, we used GPIOB8 & GPIOB9 in AF4 mode for I2C1 SDA and SCL. Then using an external library we can call a function to fill the display as black to clear it, set the cursor, print out a formatted line and update the screen.

Buzzer

Buzzes during Last of EE2028 countdown and during Lasergun Firing.
The buzzer GPIOB1 for arduino pin D6 and buzzes when a high is written to the pin.

Significant problems encountered and solutions proposed: What did you learn? What are the significant problems you encountered and how did you solve them in this assignment? If your code did not work in the lab, explain why

Hard fault during sprintf for float values

Reading more information online. Some people suggested that the heap and stack size might be too small for sprintf to handle float values. Increasing them might solve the issue.

```
31 /* Highest address of the user mode stack */
32 _estack = ORIGIN(RAM) + LENGTH(RAM);    /* end of "RAM" Ram type memory */
33
34 _Min_Heap_Size = 0x200 ;     /* required amount of heap  */
35 _Min_Stack_Size = 0x400 ;    /* required amount of stack */
36
37 /* Memories definition */
```

However, during testing even increasing them to 0x1000 and 0x2000 still resulted in a hard fault.
Another solution was to divide the float into 2 ints.

```c
float G = gyro();
int G_int = (int)G;
int G_frac = abs((int)((G - G_int) * 100));
float M = mag();
int M_int = (int)M;
int M_frac = abs((int)((M - M_int) * 100));
float P = pres();
int P_int = (int)P;
int P_frac = abs((int)((P - P_int) * 100));
float H = humid();
int H_int = (int)H;
int H_frac = abs((int)((H - H_int) * 100));
int max = 0;

if (G > G_Max){
    max = 1;
    char message_print[60];
    sprintf(message_print, "G:%d.%02d(rad/s), exceeds threshold of %d.%02d(rad/s).\r\n",
    HAL_UART_Transmit(&huart1, (uint8_t*)message_print, strlen(message_print),0xFFFF);
}
```

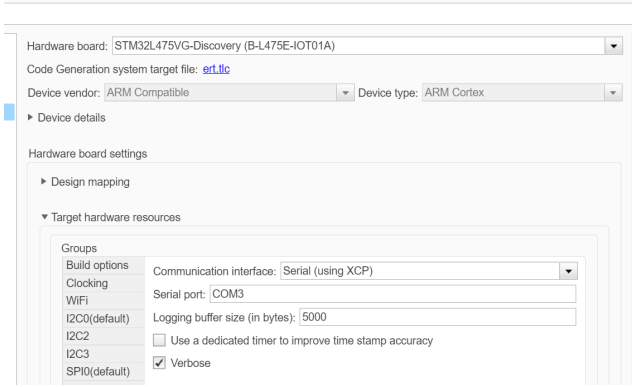
This is a temporary solution to the float issue.

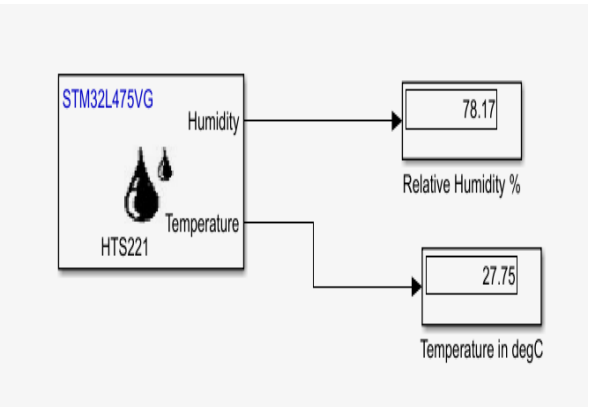
BSP values convert to SI units

Another issue was reading the values from the BSP function and converting them to the appropriate units. To this we can follow the suggestion of reading the datasheet to see the sensitivity and then reading the BSP function to see how the return value is calculated. However, I was still unsure of how accurate this reading was. The solution was to use matlab. For example, we can use stm32l475vg_humidity_temperature_sensor command to generate a model to measure humidity and temperature.

```
Initiating Battle Mode
T:27.24(C), A:9.84(m/s^2), G:0.19(rad/s), M:179.19(mT), P:1004.98(mb), H:77.96(%
RH)
T:27.24(C), A:9.92(m/s^2), G:0.04(rad/s), M:179.18(mT), P:1005.01(mb), H:77.92(%
RH)
T:27.28(C), A:9.92(m/s^2), G:0.04(rad/s), M:179.39(mT), P:1004.94(mb), H:78.37(%
RH)
T:27.24(C), A:9.92(m/s^2), G:0.04(rad/s), M:178.48(mT), P:1005.05(mb), H:78.33(%
RH)
```

Reading from serial: T: 27.24C and H:78.33%





Connect to board with this settings

T: 27.75C and H: 78.17%

(This shows are reading from serial is accurate)
We can use stm32l475vg_magnetometer_sensor for M, stm32l475vg_accel_gyro_sensor for A&G and stm32l475vg_pressure_sensor for P. However, some of their sensors are not configured correctly and their measurements are using the wrong units. However, it is still a good way to double check our readings. Also Matlab simulink uses com3 to communicate with the board, therefore we cannot use teraterm for UART at the same time.

Issues or suggestions: Please feel free to give us some feedback to make this course a better one . Your feedback, whether positive or negative, will not affect your marks.

This course can consider using arduino for micro controllers as STM32 does not seem frequently used outside.
The projects in this course could be done solo as sharing the board is difficult between teammates and can reduce productivity.

Conclusion

This assignment taught us how to use the STM32CubeIDE for debugging as well as an IDE for us to write code in. It also taught us how to research on our own using the device manual and datasheet instead of googling everything or asking teachers. This skill will definitely be used in the future where we would need to research the solution on our own instead of depending on supervisors. No help was given also for enhancements, forcing us to self learn and depend on our own interpretation of the datasheet. Leading us to research more about a deeper understanding for the STM32 board. We also learnt using various tools such as teraterm for serial communication and matlab simulink. We also learnt how to implement external drivers for the OLED boards.