

Embedded platforms and communications for IoT

Implementation of the embedded platform for plant monitoring IoT system using the B-L072Z-LRWAN1 ARM mbed-based platform

Final project: Report template

v 1.0

Daniel Rodríguez Moya
Álvaro Rodríguez Piñeiro

Table of contents

1	OVERVIEW AND INTRODUCTION.....	4
2	SUMMARY OF THE SPECIFICATIONS IMPLEMENTED VERSUS SPECIFICATIONS REQUIRED	9
3	HARDWARE BLOCK DIAGRAM OF THE SOLUTION IMPLEMENTED	12
3.1	HARDWARE RESOURCES USED IN THE SENSORS.....	12
3.1.1	HARDWARE RESOURCES USED IN THE ANALOGIC SENSORS	12
3.1.2	HARDWARE RESOURCES USED IN THE GPS.....	13
3.1.3	HARDWARE RESOURCES USED IN THE I2C SENSORS.....	15
4	SOFTWARE ORGANIZATION	18
4.1.1	Description of the implementation	18
4.1.2	Code size	18
4.1.3	Problems detected and implemented solutions	19
5	RESULTS.....	20
6	ADVANCED SPECIFICATIONS IMPLEMENTED	23
7	REFERENCES.....	24

Table of figures

Figure 1: Demo of the project	4
Figure 2: B-L072Z-LRWAN1 LoRa/Sigfox Discovery Kit	5
Figure 3: Block diagram of the project	12
Figure 4: Simplified view of a capacitive humidity sensor	13
Figure 5: Phototransistor	13
Figure 6: Simplified ADC working principle	13
Figure 7: GPS working principle	14
Figure 8: Serial communication overview	14
Figure 9: Si7021 block diagram	15
Figure 10: Capacitive accelerometer working principle	15
Figure 11: MMA8451Q state machine	16
Figure 12: TCS34725 functional block diagram	16
Figure 13: TCS34725 state machine	17
Figure 14: I2C message overview	17
Figure 15: Mbed program flowchart of the project	18
Figure 16: Application code size	19
Figure 17: Test being carried out	20
Figure 18: Test Mode information block	20
Figure 19: Google Maps providing the coordinates where the experiment was being carried out	21
Figure 20: Normal Mode information blocks	21
Figure 21: Advanced Mode information block	21
Figure 22: Advanced Mode code implementation	23

Table of tables

Table 1: Variables of interest and units _____	4
Table 2: Input devices main features _____	7
Table 3: System requirements of the project _____	9
Table 4: Project requirements _____	11
Table 5: Available interruptions to be enabled in the MMA8451Q, CTRL_REG4 I2C register ____	23
Table 6: Interruptions routing to each pin in the MMA8451Q, CTRL_REG5 I2C register _____	23

1 OVERVIEW AND INTRODUCTION

This document focuses on the description of the objectives and requirements established for the development of the final project of the course “Embedded platforms and communications for IoT”, which consists of an IoT system capable of locally monitoring the crucial environmental conditions and health of a plant throughout its life. Figure 1 illustrates how the device performs in practice.

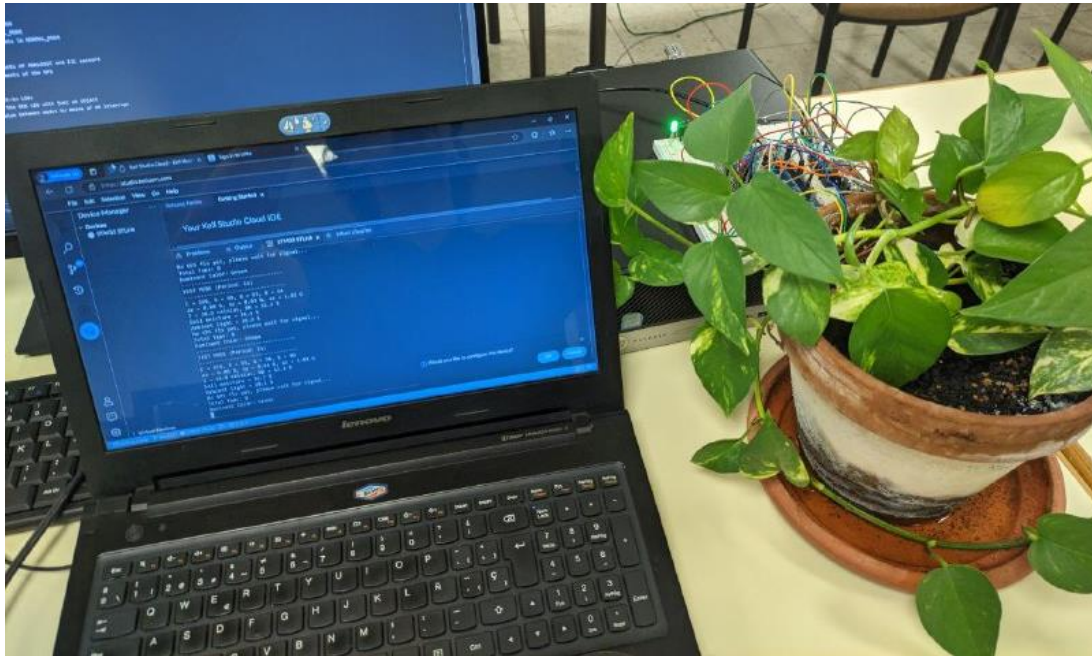


Figure 1: Demo of the project

The variables of interest to be measured are listed in Table 1.

Table 1: Variables of interest and units

Field	Magnitude	Units
Ambient	Relative humidity	%
	Temperature	° C
	Light	%
Soil	Humidity	%
Geolocation	Altitude	m
	Latitude	° ‘ “
	Longitude	° ‘ “
	Current time	HH:mm:ss.ss
Accelerations	Gravity	m/s ²
Colour	Intensity	cd

This way, the election of the hardware to be used has been proposed by the professors, bearing in mind a justified dimensioning related to the input/output peripherals and number of computations per execution cycle required for the application.

To begin with, the microcontroller unit (MCU) is the STM32 based B-L072Z-LRWAN1 LoRa/Sigfox Discovery Kit [1], Figure 2. This development board is powered by an STM32L072ZCZ, based on ARM Cortex M0+ core, whose specifications are 192 kb of flash memory, 20 kb of RAM and 20 kb of EEPROM. The module also includes a SX1276 LoRa radio chip to provide the board with long-range and low power consumption data transmission/reception. Moreover, the MCU can access peripherals via 16-bit ADC, LP-UART, SPI, I2C and USB. To complete the pack, the board includes an ST-LINK embedded debug tool interface, built-in LEDs, built-in buttons, antenna for LoRa and Arduino UNO shields connectors, among other features.

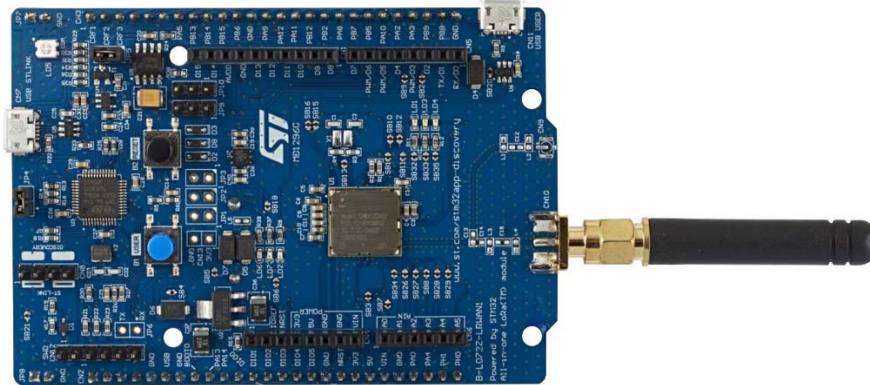








Figure 2: B-L072Z-LRWAN1 LoRa/Sigfox Discovery Kit

On the input devices side,

Table 2 describes the sensors used to measure each variable.

Table 2: Input devices main features

Name	Variable	Range and accuracy	Connection method	Picture
Si7021 [2]	Relative humidity	0 – 80 % (± 3 %)	I2C	
	Temperature	-10 – 50 °C (± 0.4 °C)		
HW5P-1 [3]	Ambient light	0 – 100 %	ADC	
SparkFun Soil Moisture Sensor [4]	Soil moisture	0 – 100 %	ADC	
TCS34725 [5]	Leaf colour	ADC range (bits)	I2C	
MMA8451Q [6]	Accelerations	-8 – 8 g	I2C	
Adafruit Ultimate GPS [7]	Longitude	-180 ° – 180 °	Serial line	
	Latitude	-90 ° – 90 °		
	Altitude	-		
	Current time	-		

The built-in USER button is also used to allow users to interact with the system.

On the output devices side, an external common anode RGB LED is connected to three digital pins on the board to adjust each channel intensity. Common anode results in an inverse logic programming. Limiting current resistors are required to protect each channel as well. Built-in LEDs number one, three and four are used to indicate working modes to the user.

To be able to implement all the required logic, Mbed OS [8] has been used as it provides an accessible API to develop C++ applications for ARM based microcontrollers and includes a wide collection of handbooks with code examples.

Keil Studio [9] has been chosen to compile and flash the programs into the board. This online IDE is part of the Mbed tool kit and allows developers to easily debug code, WebUSB flash and do version control.

2 SUMMARY OF THE SPECIFICATIONS IMPLEMENTED VERSUS SPECIFICATIONS REQUIRED

This section must begin with the statement that every specification required in the course has been finally implemented so, rather than presenting a versus, this section will summarize the functionalities.

To begin with, the system must fulfil the system requirements listed in Table 3.

Table 3: System requirements of the project

Functionality	Reference code
Temperature in range -10 to 50 °C. Accuracy of a tenth	SR1
Relative humidity in range 25 to 75 %. Accuracy of a tenth	SR2
Ambient light in range 0 to 100 %. Accuracy of a tenth	SR3
Soil moisture in range 0 to 100 %. Accuracy of a tenth	SR4
Colour of a leaf. RGB and clear channels	SR5
Global location and current time	SR6
Accelerations at least on the three axes. Accuracy of a hundredth	SR7
System must be robust and stable	GR1
Task partitioning and threads established according to requirements	GR2

The working mode of the system is based on a cyclical and sequential state machine in which the user can navigate just by pressing the USER button. Those modes and their corresponding functionalities are listed in

Table 4.

Table 4: Project requirements

Mode	Functionalities	Reference code
Test	Check connections and sensor management	TM1
	Required variables monitored every 2 seconds	TM2
	Variables are sent to the computer via USB every 2 seconds	TM3
	RGB LED is coloured in the dominant detected one	TM4
	LED1 on the board must be ON	TM5
	ADVANCED: the system should count the number of taps detected by the accelerometer every 2 seconds	-
Normal	Required variables monitored every 30 seconds	NM1
	Variables are sent to the computer via USB every 30 seconds	NM2
	Calculation of max, min and average values of temperature, relative humidity, ambient light, and soil moisture every hour. These are sent to the computer	NM3
	Calculation of the dominant colour of the leave every hour. It is sent to the computer	NM4
	Calculation of max and min values of the three axes of the accelerometer every hour. These are sent to the computer	NM5
	Global location of the plant and local time every 30 seconds	NM6
	Limits for every measured variable must be reflected using the RGB LED when values are not in the valid range	NM7
	LED2 on the board must be ON	NM8
Advanced	ADVANCED: the system should count the number of taps detected by the accelerometer every 2 seconds	-
	The system must detect if a freefall has taken place. When it happens, the system must shut down. That is disable the USER button functionalities, sensor readings and the LED3 must blink to indicate the freefall detection	-
	LED3 on the board must be ON	-

3 HARDWARE BLOCK DIAGRAM OF THE SOLUTION IMPLEMENTED

As mentioned in the previous section, the proposed advanced mode includes implementing two different interruptions from the accelerometer. Moreover, it has been opted to use the LED pin on the colour sensor to only enable the LED when strictly necessary, so from the proposed block diagram in the course guide document, some additions have been made.

Figure 3 depicts the final block diagram for the project. It is worth mentioning that the LED pin of the colour sensor has been connected to a digital pin of the board to low the power consumption and both interruption pins of the accelerometer have been connected to the board as well for the advanced mode that will be described in detail in following sections.

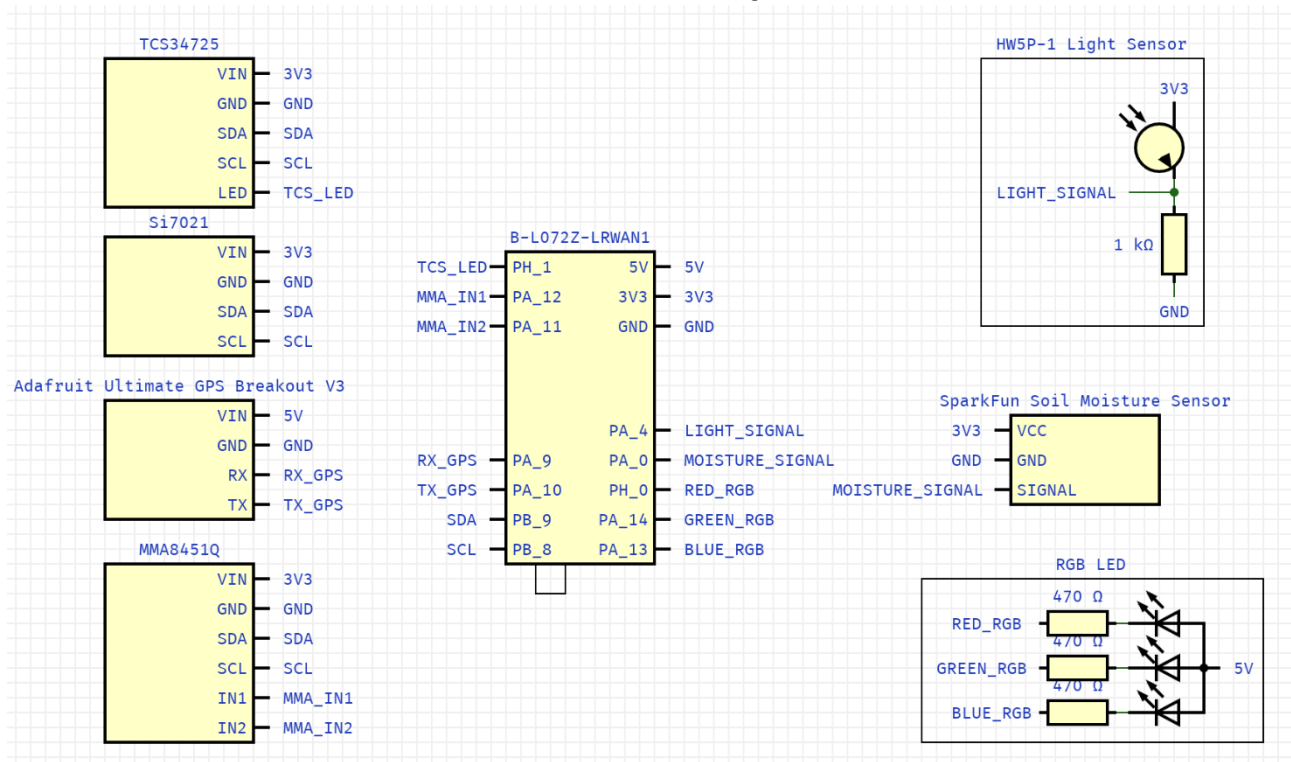


Figure 3: Block diagram of the project

3.1 HARDWARE RESOURCES USED IN THE SENSORS

This subsection focuses on describing the hardware resources that allow the sensors to convert real world variables into virtual information and the communication protocols involved in connecting each device with the MCU.

3.1.1 HARDWARE RESOURCES USED IN THE ANALOGIC SENSORS

Listing the hardware interfaces used, the first one is the 12-bit ADC to read the soil moisture and ambient light sensors, each connected to a channel.

Briefly explained, soil moisture sensors are typically based on the capacitive humidity phenomenon, consisting of a dielectric material and how that material changes as a function of relative humidity, Figure 4 [10].

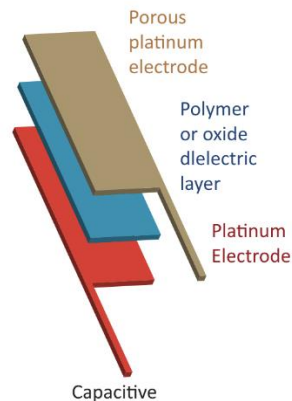


Figure 4: Simplified view of a capacitive humidity sensor

For the ambient light sensor, it is based on a phototransistor, which operates by converting incoming photons to electrons in the base of a bipolar transistor, Figure 5 [11]. As for any such transistor, the base current causes a larger collector-emitter current flow, which is detected by the ADC of the MCU.

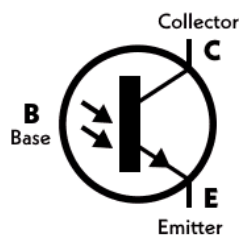


Figure 5: Phototransistor

Using the Mbed function *analogIn.read()* returns a normalized float value of the analogic signal read, corresponding 0.0 to 0 bit and 1.0 to 4095 bit. Figure 6 [12] displays a simplified view of the process.



Figure 6: Simplified ADC working principle

3.1.2 HARDWARE RESOURCES USED IN THE GPS

The working principle of a GPS module like the Adafruit Ultimate GPS Breakout v3 is based on the trilateration to calculate the user's exact location by receiving the signals provided by the satellites orbiting the Earth. Satellites circle the globe twice a day in a precise orbit, transmitting unique signals and orbital parameters that allow for the mentioned trilateration, in addition to information like current time and date, Figure 7 [13].

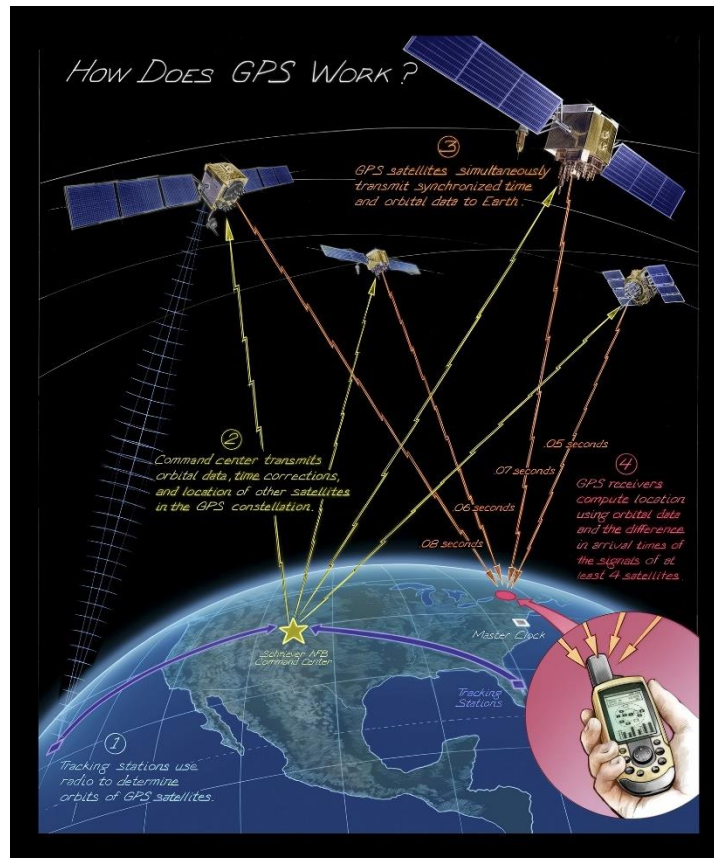


Figure 7: GPS working principle

To receive those signals, the implemented GPS includes a internal patch antenna and a u.FL connector for external active antenna, which has finally been connected to the SMA connector to improve its effectiveness.

To communicate with the GPS, UART or *Serial1* in the board has been used. To enable it, *BufferedSerial* class is set up to transmit and receive bytes of data in a sequence, being a full duplex communication. The baud rate has been set at 9600 bauds.

To communicate with the GPS, both *gps.read()* and *gps.write()* functions are used. Writing focuses on specifying the desired working mode of the GPS, while readings return information from the GPS from which it is possible to obtain the desired data.

Figure 8 [14] illustrates how the protocol works for one line as the working principle is the same for both RX and TX lines.

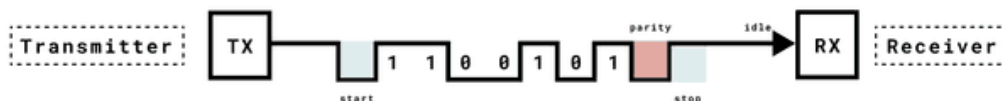


Figure 8: Serial communication overview

3.1.3 HARDWARE RESOURCES USED IN THE I2C SENSORS

To begin with I2C sensors, the first to be described is the Si7021. This relative humidity and temperature sensor works by converting measurements read by independent analogic sensors, a capacitive humidity sensor and a thermistor, to precise values to be sent to the I2C bus. This is done by an integrated ADC, a control logic module which calibrates the measurement to improve their quality and a I2C interface which organizes all the data into I2C registers, Figure 9 [2].

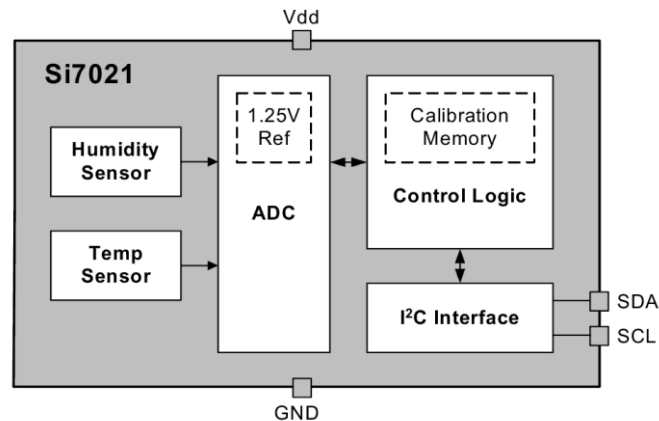


Figure 9: Si7021 block diagram

Continuing with the accelerometer, among the few technologies, the MMA8451Q uses capacitive technology, relying on a change in electrical capacitance in response to acceleration, Figure 10 [15]. They utilize the properties of an opposed plate capacitor for which the distance between the plates varies proportionally to applied acceleration, thus altering the capacitance. Again, those values are sent to integrated circuits that provide calibration and specific registers for the data to be used by the MCU.

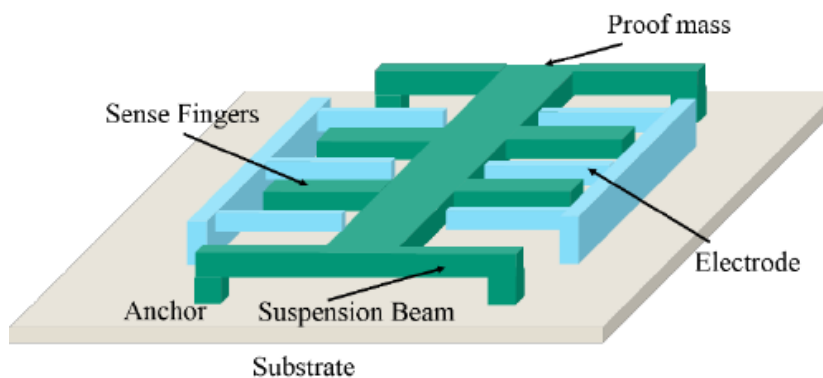


Figure 10: Capacitive accelerometer working principle

To implement it, the state machine in Figure 11 [6] must be considered.

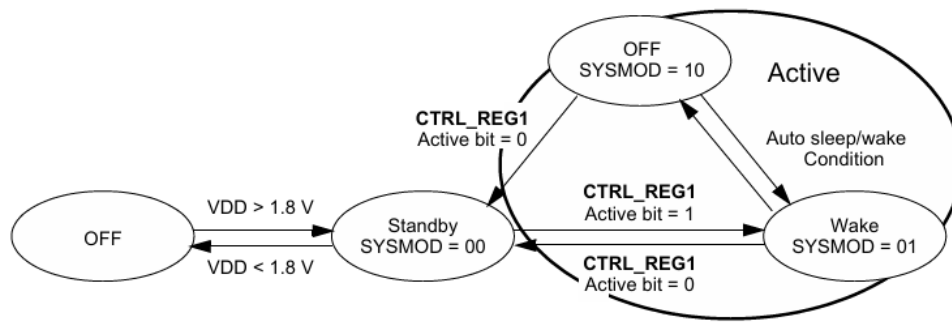


Figure 11: MMA8451Q state machine

The last to be mentioned is TCS34725 colour sensor that operates utilizing an RGBC engine that incorporates gain control (AGAIN) and four integrating ADCs dedicated to the RGBC photodiodes, Figure 12 [2]. The sensor's integration time (ATIME) directly affects the resolution and sensitivity of the RGBC readings, allowing for customization based on the application requirements.

During operation, all four channels (Red, Green, Blue, and Clear) are integrated simultaneously. Once the integration cycle completes, the sensor transfers the collected data to dedicated colour data registers. These transferred values, referred to as channel counts, represent the measured light intensity for each channel.

To ensure reliable data handling, the sensor employs a double-buffering mechanism. This design prevents the reading of invalid data during the transfer process, ensuring the integrity of the measurements. After the transfer is complete, the TCS34725 automatically advances to the next operational state based on its preconfigured state machine, streamlining the measurement process.

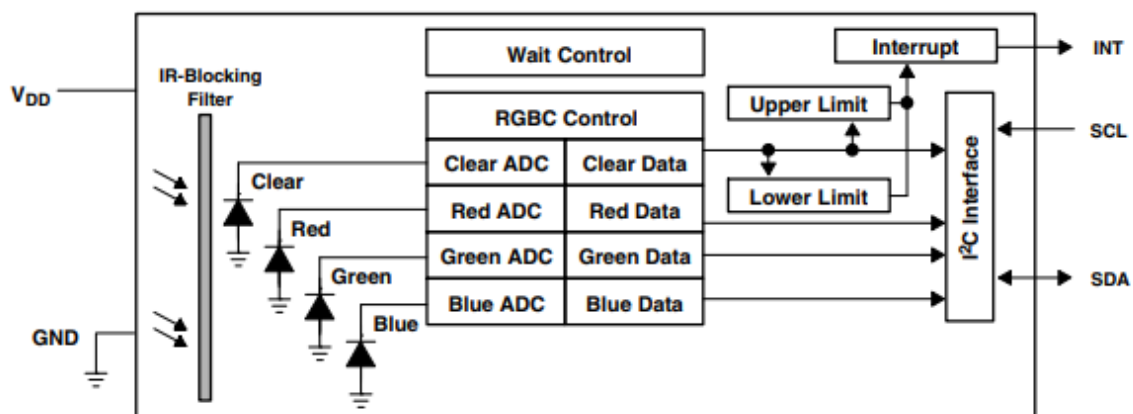


Figure 12: TCS34725 functional block diagram

In addition, the state machine of the colour detection sensor is composed of 4 main states Sleep, Idle, Wait and RGBC (measurement acquisition), Figure 13 [2].

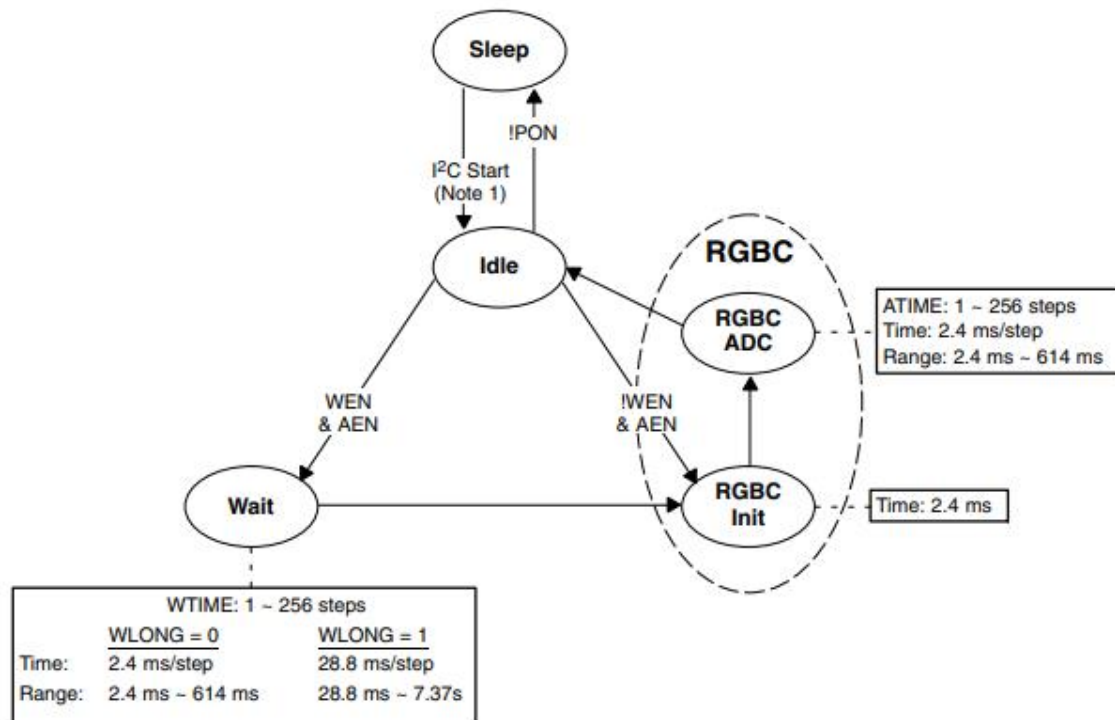


Figure 13: TCS34725 state machine

Finally, to communicate with devices with a built-in I2C interface, the *I2C1* bus in the board is shared for each of the three sensors in the system. This protocol is based on a master-slave communication with two lines, data, and clock, at a frequency of 100 KHz. The key of this protocol is that, as the bus is shared among all compatible sensors, each of them has to first identify itself by revealing its unique address.

The main two Mbed functions used to communicate with I2C peripherals are *i2c.write()* and *i2c.read()*. While writing is a two-parameter function in which a value is written into a register from a device, reading only involves accessing the value stored in a register. Figure 14 [16] illustrates how I2C operations work.

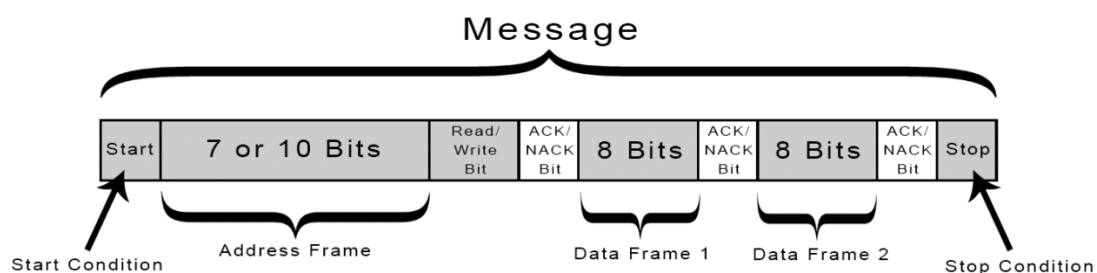


Figure 14: I2C message overview

4 SOFTWARE ORGANIZATION

4.1.1 Description of the implementation

To illustrate the implementation of the developed program, the flowchart in Figure 15 will be used to go into further details.

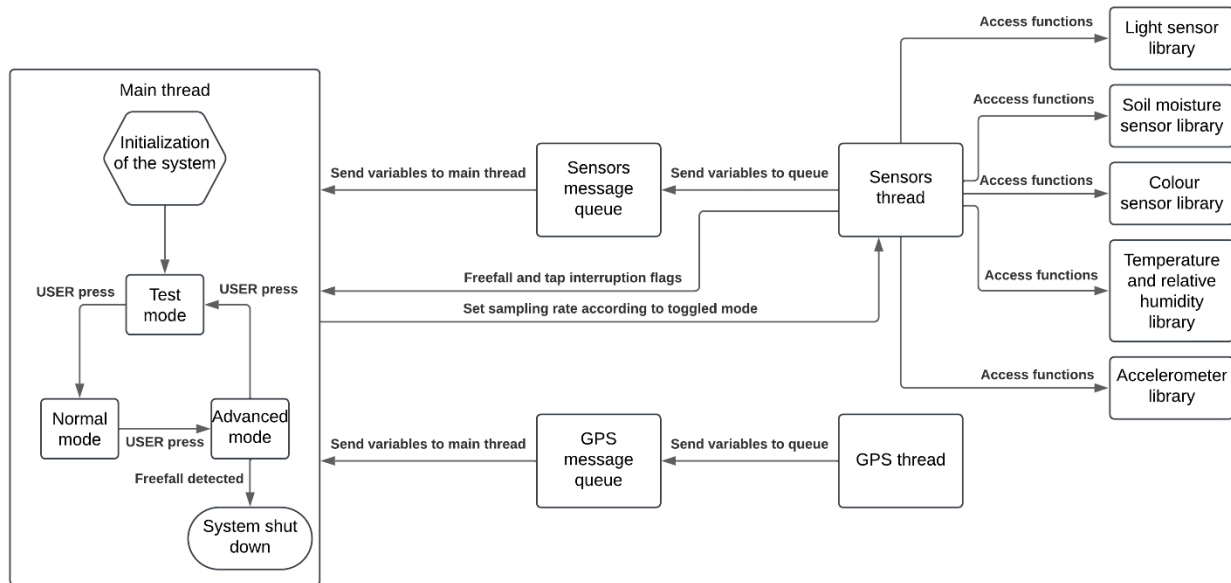


Figure 15: Mbed program flowchart of the project

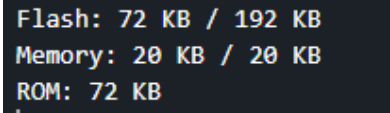
If more detailed information is required for the comprehension of the implementation, it is strongly recommended to check the delivered code, as it includes many comments to explain key concepts.

4.1.2 Code size

The application uses **72 KB** of flash memory. On the other hand, we have configured the thread sizes with the following memory stack space:

1. The main thread occupies **2048 bytes**.
2. The GPS thread occupies **1024 bytes**.
3. The thread managing the other sensors occupies **512 bytes**.

Figure 16 that has been extracted from the compilation process showcases the main information regarding application's memory:



```
Flash: 72 KB / 192 KB
Memory: 20 KB / 20 KB
ROM: 72 KB
```

Figure 16: Application code size

4.1.3 Problems detected and implemented solutions

First, we had to increase the number of bytes allocated to the GPS thread in the memory stack because an exception was triggered indicating that this thread did not have enough memory. We also had to increase the stack size for the main thread because the same exception as with the GPS module was triggered.

On the other hand, we encountered issues when trying to receive signals from satellites on the GPS module. To address this, we implemented the following hardware configurations:

1. We used an external antenna to improve signal reception from the GPS satellites.
2. We dedicated a 5V power pin from the microcontroller exclusively to power the GPS receiver module. This helped prevent current leaks and ensured that the supply voltage remained as stable as possible.
3. We isolated the antenna from other system cables to avoid any electromagnetic interference.

As for software configurations:

1. We set the internal frequency of the GPS module and the sampling rate to 1Hz.
2. During program initialization, we sent the command \$PGCMD,33,1*6C to the GPS module to check the antenna status and ensure it was functioning correctly.

Regarding the temperature and humidity sensor, we decided to use the **hold master mode** instead of the **no hold master mode**. This allowed us to sequentially organize all measurements taken via the I2C bus with this sensor in a straightforward manner, thus avoiding potential errors when reading the sensor's registers.

Finally, regarding the colour detection sensor, we encountered issues obtaining stable readings when measuring all four output channels (R, G, B, and C). To address this, before taking each measurement from any of the channels, we turn on the sensor's LED 30 milliseconds before the acquisition of each measurement to achieve more stable readings regardless of the ambient light level.

5 RESULTS

To demonstrate the specifications have been met, each of the working modes were implemented with a set of information blocks to be printed in Keil Studio's serial output. To have the GPS in coverage range, the test was carried out outside a window, Figure 17.



Figure 17: Test being carried out

Figure 18 corresponds to Test Mode, where its requirements are printed every 2 seconds.

```
-----  
TEST MODE (Period: 2s)  
-----  
C = 8122, R = 3906, G = 3330, B = 3576  
ax = -1.23 m/s2, ay = 1.44 m/s2, az = 9.80 m/s2  
T = 20.1 celsius, RH = 55.9 %  
Soil moisture = 20.7 %  
Ambient light = 100.0 %  
Fix Status = 1, Time (UTC + 1): 12:31:31.0, Alt = 683.90 m, Lat = 40.391472 deg, Lon = -3.635018 deg  
Total Taps: 0  
Dominant Color: Red
```

Figure 18: Test Mode information block

As it can be seen, GPS is giving a correct geolocation if being compared to Google Maps, Figure 19.

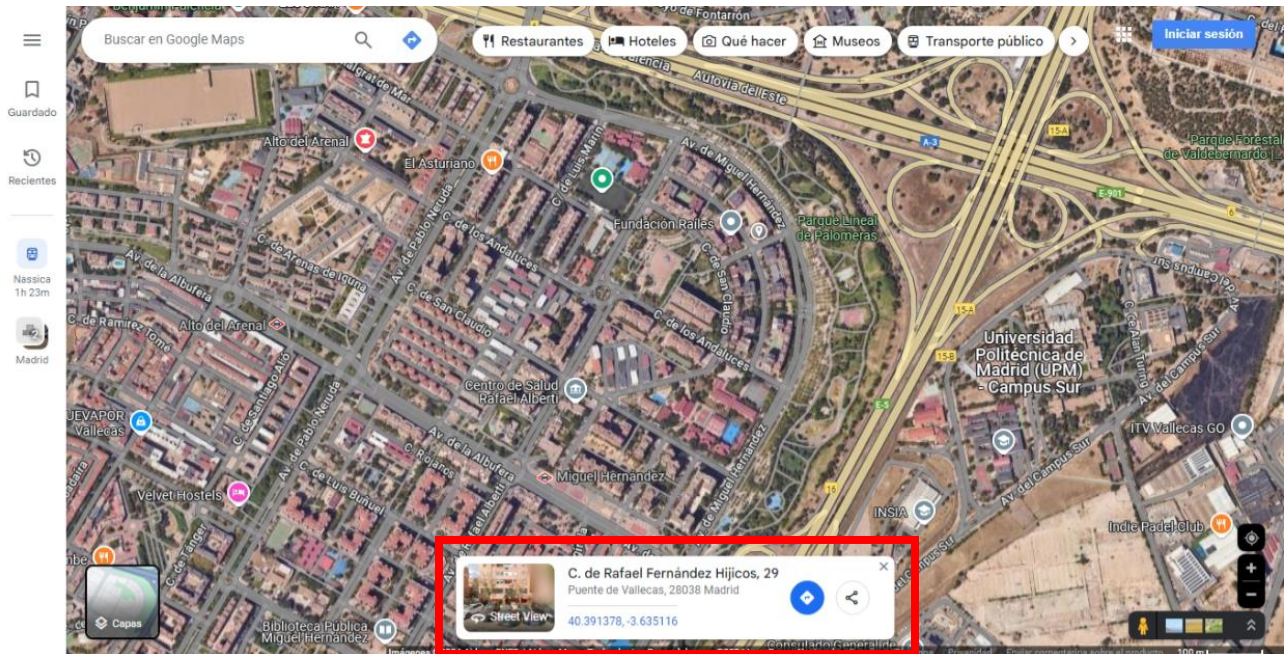


Figure 19: Google Maps providing the coordinates where the experiment was being carried out

Next, Normal Mode follows a similar implementation, where the requirements are printed every 30 seconds but, in addition, every hour, there is an extra block with the required statistics, Figure 20.

```

-----
NORMAL MODE (Period: 30s)
-----
C = 7816, R = 3773, G = 3233, B = 3455
ax = -1.25 m/s2, ay = 1.44 m/s2, az = 9.81 m/s2
Temperature out of valid range! RH = 59.3 %
Soil moisture = 21.6 %
Ambient light = 100.0 %
Fix Status = 1, Time (UTC + 1): 12:56:43.0, Alt = 694.20 m, Lat = 40.391235 deg, Lon = -3.634748 deg
Total Taps: 0
-----
ONE HOUR STATS:
-----
RHmin = 59.3 %, RHmax = 59.7 %, RHavg = 59.5 %
Tmin = 18.6 celsius, Tmax = 18.6 celsius, Tavg = 18.6 celsius
SMmin = 21.3 %, SMmax = 22.3 %, SMAvg = 21.8 %
ALmin = 100.0 %, ALmax = 100.0 %, ALavg = 100.0 %
axmin = -1.27 m/s2, axmax = -1.24 m/s2
aymin = 1.39 m/s2, aymax = 1.44 m/s2
azmin = 9.80 m/s2, azmax = 9.84 m/s2
Dominant Color: Red
-----

```

Figure 20: Normal Mode information blocks

Last, Advanced Mode, which will be described later on, only prints a statement when it does detect a freefall, Figure 21.

```

-----
ADVANCED MODE (FREEFALL DETECTION)
-----
Freefall detected on Z-axis. SYSTEM SHUT DOWN!
=====

```

Figure 21: Advanced Mode information block

The results obtained in the test have been the product of consulting a series of sources to craft the final version of the code. Those sources have been:

- API Classes document from the course Moodle [17]
- The specific datasheet of each device implemented
- Mbed OS 6.15 API list [18]
- Generative AI for small snippets of code [19]

6 ADVANCED SPECIFICATIONS IMPLEMENTED

The Advanced Mode to be implemented consisted in two new functionalities:

- A tap counter [20] for Test and Normal modes where the accelerometer must detect how many times per sampling interval the plant was moved
- A free fall detector [21] as an isolated mode in which the system is just waiting for the event to take place and, if so, LED3 must start to blink and interruptions are disabled together with sensor readings until the user presses the RESET button

The code implementation consisted in:

1. Setting up the accelerometer as the datasheet suggests
2. Enabling the ability to have tap and free fall interruptions generated by the accelerometer, Table 5 [6]

Table 5: Available interruptions to be enabled in the MMA8451Q, CTRL_REG4 I2C register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INT_EN_ASLP	INT_EN_FIFO	INT_EN_TRANS	INT_EN_LNDPRT	INT_EN_PULSE	INT_EN_FF_MT	—	INT_EN_DRDY

3. Setting up each of the interruption modes characteristics [20] [21]
4. Routing each of the two interruptions independently to the two available interruption pins of the device, Table 6 [6]

Table 6: Interruptions routing to each pin in the MMA8451Q, CTRL_REG5 I2C register

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INT_CFG_ASLP	INT_CFG_FIFO	INT_CFG_TRANS	INT_CFG_LNDPRT	INT_CFG_PULSE	INT_CFG_FF_MT	—	INT_CFG_DRDY

5. Connecting each interruption pin to a digital pin in the microcontroller
6. Implementing an interruption service routine for each case which triggers to true a flag when the events take place

Further details on how to setup this mode can be read in the code in Figure 22.

```
// FUNCTION TO INITIALIZE THE ACCELEROMETER WITH FREEFALL DETECTION =====
void init_mma8451_pulse_ff() {
    write_register_mma8451(CTRL_REG1, 0x08); // bit 3 = 1, Standby Mode, DRO = 400 Hz

    // FREEFALL INTERRUPT COMMAND -----
    write_register_mma8451(FF_MT_CFG, 0xB8); // Enable motion detection on Z-axis with event latch enabled
    write_register_mma8451(FF_MT_THS, 0x03); // Set threshold to ~0.18g (0x03 * 0.063g/LSB)
    write_register_mma8451(FF_MT_COUNT, 0x06); // Set debounce count to filter transient events, 400 Hz (2.5 ms) timer 120

    // TAP INTERRUPT COMMANDS -----
    write_register_mma8451(PULSE_CFG, 0x15); // Configure PULSE_CFG to enable single tap on X, Y, Z with latch enabled
    write_register_mma8451(PULSE_THSX, 0x19); // Set X threshold for 1.575g
    write_register_mma8451(PULSE_THSY, 0x19); // Set Y threshold for 1.575g
    write_register_mma8451(PULSE_THSZ, 0x2A); // Set Z threshold for 2.65g
    write_register_mma8451(PULSE_TMLT, 0x50); // Set time limit for 50 ms
    write_register_mma8451(PULSE_LTCY, 0xF0); // Set latency for 300 ms

    // SHARED INTERRUPT COMMANDS -----
    write_register_mma8451(CTRL_REG4, 0x0C); // Enable pulse (bit 3) and FF (bit 2) interrupt - 0000 1100
    write_register_mma8451(CTRL_REG5, 0x08); // Route INT_CFG_PULSE (bit 3 = 1) to IN1 and INT_CFG_FF_MT (bit 2 = 0) to IN2
    char data = read_register_mma8451(CTRL_REG1); // Initialize the MMA8451Q accelerometer by setting it to active mode, read
    write_register_mma8451(CTRL_REG1, data | 0x01);
}
```

Figure 22: Advanced Mode code implementation

7 REFERENCES

- [1] ST, "ST," 6 June 2019. [Online]. Available: <https://www.st.com/en/evaluation-tools/b-l072z-lrwan1.html>. [Accessed 11 October 2024].
- [2] S. Labs, "silabs," 2022. [Online]. Available: <https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf>. [Accessed 10 October 2024].
- [3] Arduino, "arduino," [Online]. Available: <https://wiki-content.arduino.cc/documents/datasheets/HW5P-1.pdf>. [Accessed 11 October 2024].
- [4] SparkFun, "sparkfun," [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Biometric/SparkFun_Soil_Moisture_Sensor.pdf. [Accessed 20 October 2024].
- [5] TAOS, "adafruit," August 2012. [Online]. Available: <https://cdn-shop.adafruit.com/datasheets/TCS34725.pdf>. [Accessed 22 October 2024].
- [6] N. Semiconductors, "nxp," February 2017. [Online]. Available: <https://www.nxp.com/docs/en/data-sheet/MMA8451Q.pdf>. [Accessed 14 October 2024].
- [7] Adafruit, "adafruit," 23 August 2012. [Online]. Available: <https://cdn-learn.adafruit.com/downloads/pdf/adafruit-ultimate-gps.pdf>. [Accessed 15 October 2024].
- [8] a. MBED, "MBED," 2024. [Online]. Available: <https://os.mbed.com/mbed-os/>. [Accessed 18 September 2024].
- [9] arm, "arm KEIL Studio," 2024.
- [10] "PST," [Online]. Available: <https://www.processsensing.com/en-us/blog/capacitive-sensor-technology.htm>. [Accessed 2024].
- [11] "Electrical Technology," August 2022. [Online]. Available: <https://www.electricaltechnology.org/2022/08/phototransistor.html>. [Accessed 2024].
- [12] G. M. Smith, "DEWESoft," 13 February 2024. [Online]. Available: <https://dewesoft.com/blog/what-is-adc-converter>. [Accessed 2024].
- [13] B. Morser, "Time and Navigation," 2012. [Online]. Available: <https://timeandnavigation.si.edu/multimedia-asset/how-does-gps-work>. [Accessed 2024].
- [14] H. Siebeneicher, "Arduino DOCS," 4 September 2024. [Online]. Available: <https://docs.arduino.cc/learn/communication/uart/>. [Accessed 2024].
- [15] S. Sinha, "ResearchGate," July 2014. [Online]. Available: https://www.researchgate.net/figure/Capacitive-accelerometer-with-lateral-sensing_fig2_264833557. [Accessed 2024].
- [16] S. Campbell, "Circuit Basics," 2016. [Online]. Available: <https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/>. [Accessed 2024].

- [17] V. H. Díaz, M. C. Lapastora, E. J. Martínez and G. A. De Pablo, "Moodle UPM," 2024. [Online]. Available:
https://moodle.upm.es/titulaciones/oficiales/pluginfile.php/12148693/mod_resource/content/10/EmbeddedSystems_W3B6_v6.0.pdf. [Accessed 15 October 2024].
- [18] a. MBED, "mbed," 2024. [Online]. Available: <https://os.mbed.com/docs/mbed-os/v6.15/apis/index.html>. [Accessed 22 September 2024].
- [19] OpenAI, "ChatGPT," OpenAI, 2024. [Online]. Available: <https://chatgpt.com/>. [Accessed 2024].
- [20] K. Tuck, "NXP Semiconductors," September 2010. [Online]. Available:
<https://www.nxp.com/docs/en/application-note/AN4072.pdf>. [Accessed 2 October 2024].
- [21] K. Tuck, "NXP Semiconductor," October 2011. [Online]. Available:
<https://www.nxp.com/docs/en/application-note/AN4070.pdf>. [Accessed 7 October 2024].