

DESIGN (E) 314
TECHNICAL REPORT

High-Altitude Balloon Data Logger


Author:
Noah Foroma

Student Number:
20687184

April 06, 2019

Plagiaatverklaring / Plagiarism Declaration

1. Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is.
Plagiarism is the use of ideas, material and other intellectual property of another's work and to present is as my own.
2. Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is.
I agree that plagiarism is a punishable offence because it constitutes theft.
3. Ek verstaan ook dat direkte vertalings plagiaat is.
I also understand that direct translations are plagiarism.
4. Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelike aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is.
Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism.
5. Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie.
I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.


Handtekening / Signature

N. FOROMA
Voorletters en van / Initials and surname

20687184
Studentenommer / Student number

17 MAY 2019
Datum / Date

Abstract

The report documents the design and development of a data logger for a High Altitude Balloon launch. The hardware design, decisions and justifications made for the implementation of the hardware are also discussed. In addition to this, the software structure and design are made mention of in the report, as well as measurement methods and their results. The report concludes with the assertion that complete functionality of the system is in order, and suggestions for future design are included.

Contents

1	Introduction	7
2	System description	8
2.1	System block diagram	8
2.2	Interface description	8
2.2.1	Power supplies	8
2.2.2	STM32F334R8 Board	9
2.2.3	Serial communication (UART)	9
2.2.4	ADC & Current and Voltage sensing	9
2.2.5	GPIO & LCD	9
2.2.6	I ² C, BME280 sensor & LIS2DH12 accelerometer	10
3	Hardware design and implementation	10
3.1	Power supply	10
3.1.1	5 V Power Supply	10
3.1.2	3.3 V Power Supply	11
3.2	UART communications	12
3.3	Current and Voltage sensing	13
3.3.1	Current sensing	13
3.3.2	Voltage sensing	15
3.4	LCD	16
3.5	BME280 Temperature, humidity and pressure sensor	17
3.6	LIS2DH12 3-axis MEMS accelerometer	18
3.7	Release mechanism signal	18
4	Software design and implementation	18
4.1	Layout of Project	19
4.2	Description of Functions	20
4.3	Peripheral Setup	22
4.3.1	UART	22
4.3.2	GPIO	22
4.3.3	ADC	22
4.3.4	I ² C	23
4.3.5	Timers	23
4.4	Release signal logic	23
4.5	Flow diagram	23
5	Measurements and Results	25
5.1	Power Supply	25
5.1.1	5 V Power Supply	25
5.1.2	3.3 V Power Supply	25
5.2	UART Communication	26
5.3	Current and Voltage Sensing	27

5.3.1	Current Sense Circuit.....	27
5.3.2	Voltage Measurement Circuit.....	28
5.4	LCD	29
5.5	Temperature and humidity measurement.....	30
5.6	Pressure measurement.....	30
5.7	Acceleration measurement.....	30
5.8	Release signal	30
6	Conclusions	31
6.1	Strengths.....	31
6.2	Weaknesses	31
6.3	Improvements and suggestions.....	31
7	References	32
8	Appendices	33
8.1	Appendix A – Circuit Diagrams	33
8.2	Appendix B – STM32 pinout table.....	35

List of Figures

1	Figure 1: System block diagram	8
2	Figure 2: UART connection	9
3	Figure 3: 5 V power supply schematic	10
4	Figure 4: 3.3 V power supply schematic	11
5	Figure 5: USB to UART circuit schematic	12
6	Figure 6: Current sensor break-out board schematic.....	13
7	Figure 7: Current sense circuit.....	14
8	Figure 8: Non-inverting operational amplifier	15
9	Figure 9: Voltage divider circuit.....	16
10	Figure 10: LCD circuit diagram	16
11	Figure 11: BME280 I2C connection diagram.....	17
12	Figure 12: LIS2DH12 accelerometer connection diagram.....	17
13	Figure 13: Flow diagram of system.....	24
14	Figure 14: UART oscilloscope results	27
15	Figure 15: LCD test result	29
16	Figure 16: BME280 schematics (incorrect and corrected).....	33
16	Figure 17: LIS2DH12 schematics	34

List of Tables

1	Table 1: System Components.....	8
2	Table 2: Project Files	19
3	Table 3: main.c and stm32f3xx_it.c files	20
4	Table 4: project.c file	20
5	Table 5: lcd.c file.....	21
6	Table 6: bme_sensor.c file	21
7	Table 7: accelerometer.c file	22
8	Table 8: 5 V power supply results.....	22
9	Table 9: 3.3 V power supply results.....	26
10	Table 10: Current sense circuit results	28
11	Table 11: Voltage divider circuit results.....	28
12	Table 12: STM32 pinouts	35

List of Abbreviations

AC	Alternating Current
ADC	Analog to Digital Converter
GPIO	General Purpose Input/Output
GPS	Global Positioning System
HAB	High Altitude Balloon
HAL	Hardware Abstraction Layer
ISR	Interrupt Service Routine
I²C	Inter Integrated Circuit
LED	Light Emitting Diode
LCD	Liquid Crystal Display
PCB	Printed Circuit Board
PDD	Project Definition Document
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

1 Introduction

Whether used to observe current weather conditions, or as a niche project to improve on design skills or for research, a High Altitude Balloon satisfies the abovementioned criterion. To obtain the relevant data, development of a data logging system is required, after which it is mounted onto the balloon and launched into the stratosphere. This data logging system requires that measurements of altitude, latitude, longitude, temperature, relative humidity, pressure and acceleration are taken. In addition to this, should the HAB stray from the bounds within which it is confined, a release mechanism should be in place such that the data logger can find its way back onto the ground.

The developed data logger is driven by an STM32F334R8 MCU, which is responsible for communicating with the various components that are used for the previously mentioned measurements. The altitude, latitude and longitude are recorded by a GPS module (however for this project the GPS coordinates were simulated via PC). Measurements of the temperature, relative humidity and pressure were taken with a Bosch BME280 environmental sensor, and the acceleration was recorded with an STM LIS2DH12 accelerometer. The data logger is powered by a 9 V external source, and this is regulated down to 5 V and 3.3 V to power the components of the logger that have limited operating voltage ranges.

2 System description

The following table lists the components used in the system and their operating voltages.

Table 1: System Components

Component	Operating Voltage
STM32 module	2.0 V - 5.5 V
FT230X UART	3.3 V
PC1601 – LCD Module	5.0 V
BME280 sensor	1.71 V - 3.6 V
LIS2DH12 accelerometer	1.71 V - 3.6V
Current sense module	2.5 V - 20 V
STM L7805CV regulator	7.5 V – 18 V
MCP1700 regulator	2.3 V – 6.0 V

2.1 System block diagram

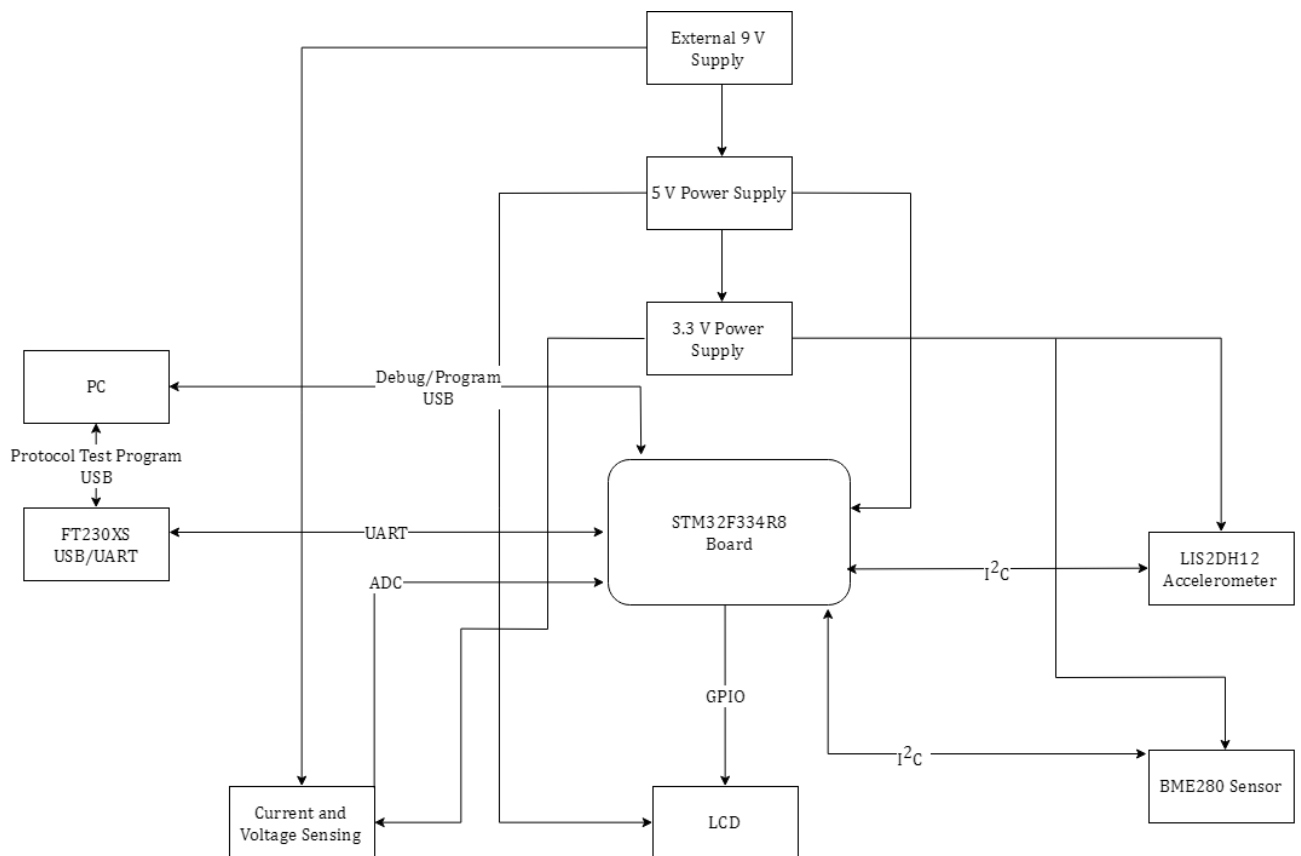


Figure 1: System block diagram

2.2 Interface description

2.2.1 Power supplies

As shown in the block diagram, the project board (PCB) was powered by an external 9 V power supply. The 9 V input was then regulated down to 5 V using an L7805CV regulator, which was then further regulated down to 3.3 V. The regulated power supplies were then used to power the components used as shown in the block diagram.

2.2.2 STM32F334R8 Board

The STM32F334R8 was the primary/main component of the system. The module makes use of an ARM microcontroller, and various peripheral interfaces (such as UART, ADC, I²C, GPIO) for communication with external devices. This module was programmed to communicate with the various components that were incorporated in the system, and debugging of the system's communication happened through this board.

2.2.3 Serial communication (UART)

An FT230X USB-UART module was used to communicate with the board using the UART interface. The module was connected to a PC via USB and received its power from the USB connection. The module allowed for the communication between the board and the PC, with both receive and transmit functionality included. The UART communication was specified as 8 bit asynchronous, with 1 start and stop bit, and no parity bit. The baud rate was also specified as 115200 bits/s [1].

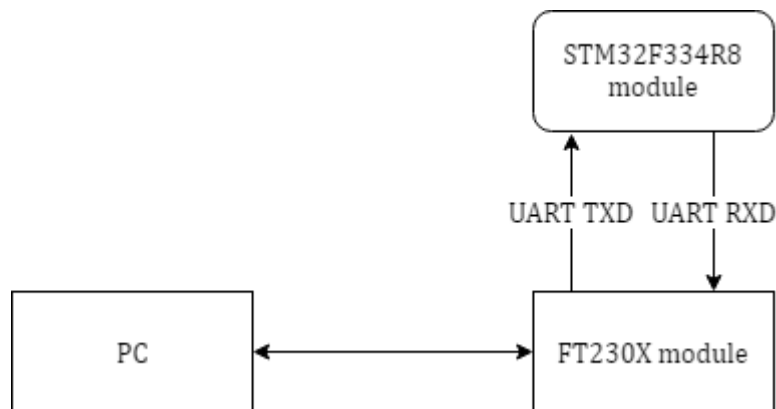


Figure 2: UART connection

2.2.4 ADC & Current and Voltage sensing

The analog to digital converter of the STM32F334R8 module was used to sample the voltage and current supplied to the project board. This occurred through use of a voltage divider circuit for the voltage, and a current sensing circuit for the current, which are discussed later. The ADC converts a sampled voltage/current to an equivalent number in the range 0 – 4095, and then software must be developed to obtain the actual value that was sampled from this number. The ADC is only capable of sampling voltages up to 3.3 V, hence the need for the voltage divider and current sense circuit.

2.2.5 GPIO & LCD

The LCD was used to display data obtained from the BME280 sensor (specifically the measured temperature) and the altitude obtained from the GGA string sent to the system. In addition to this, if the release signal conditions were satisfied, the LCD would display a “B” for ten seconds. The LCD was programmed by the STM32 module through use of GPIO pins which were set as output, hence the

required data was only sent to the LCD, with the LCD not sending anything back. The GPIO pins were connected to LCD pins DB4-DB7, RS (Register Select), RNW (Read-Not-Write), and E (Enable).

2.2.6 I²C, BME280 sensor & LIS2DH12 accelerometer

The BME280 sensor was used to measure the environmental conditions, namely temperature, relative humidity and pressure, whereas the LIS2DH12 accelerometer was used to measure the acceleration of the board in the x, y and z directions. The components interacted with the STM32 module through use of the I²C communication interface. Since both components used the same I²C bus, and were the slaves of the communication, with the STM32 module being the master, they both shared the same SDA and SCL connections to the module.

3 Hardware design and implementation

The following section provides a detailed discussion on the decisions, justifications, and calculations of each hardware element that was incorporated in the design of the data logger.

3.1 Power supply

The project PCB was to be supplied with a nominal 9 V battery source, and this power was to be regulated down to two separate supplies, a 5 V supply and a 3.3 V supply. Each supply was meant to power various components of the board.

3.1.1 5 V Power Supply

The 5 V supply was predesigned and is shown in the figure below, taken from the PDD [1].

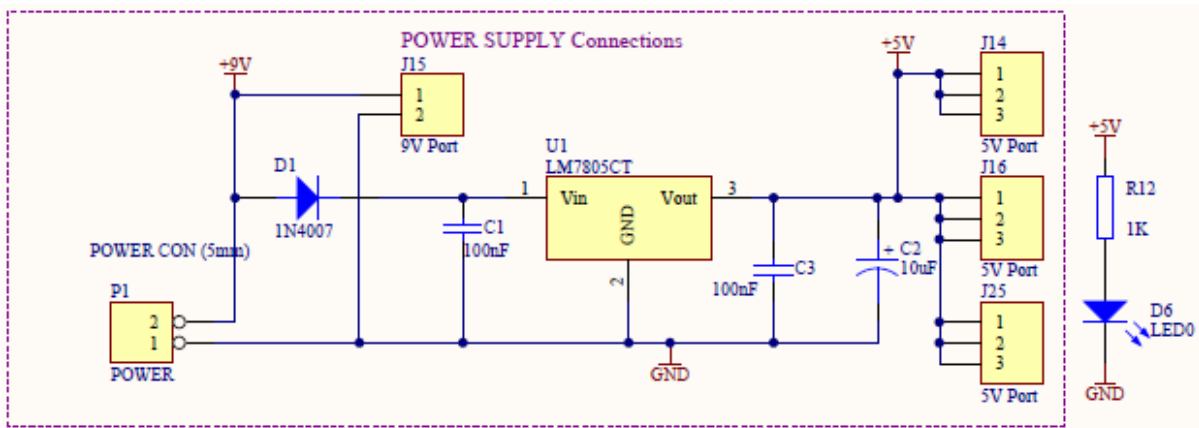


Figure 3: 5 V power supply schematic

The 5 V supply makes use of a 7805 regulator (U1) to regulate the 9 V input power down to 5 V. Due to damage of the initially provided regulator, a replacement one was obtained which was not the same as

the one initially provided. The replacement regulator was an STM L7805CV in a TO-220 package and operates similarly to the initially provided regulator. Therefore, it was possible to incorporate the new regulator into the designed sub-circuit with no changes.

The diode D1 in the above figure was included in the design to prevent damage to the regulator and other connected components if the input voltage polarity was reversed. For stable operation under all load conditions, the input bypass capacitor C1 was included in the design of the 5V power supply [2]. The capacitors C2 and C3 improve stability and transient response. Also, the LED D6 signifies that a voltage is present on the 5V line.

The values chosen for the capacitors are well within the range suggested by the datasheet [2] for the regulator, and since no AC source is being used, do not have to completely conform to the suggested values.

3.1.2 3.3 V Power Supply

A hand drawn schematic of the 3.3 V supply is provided in the figure below:

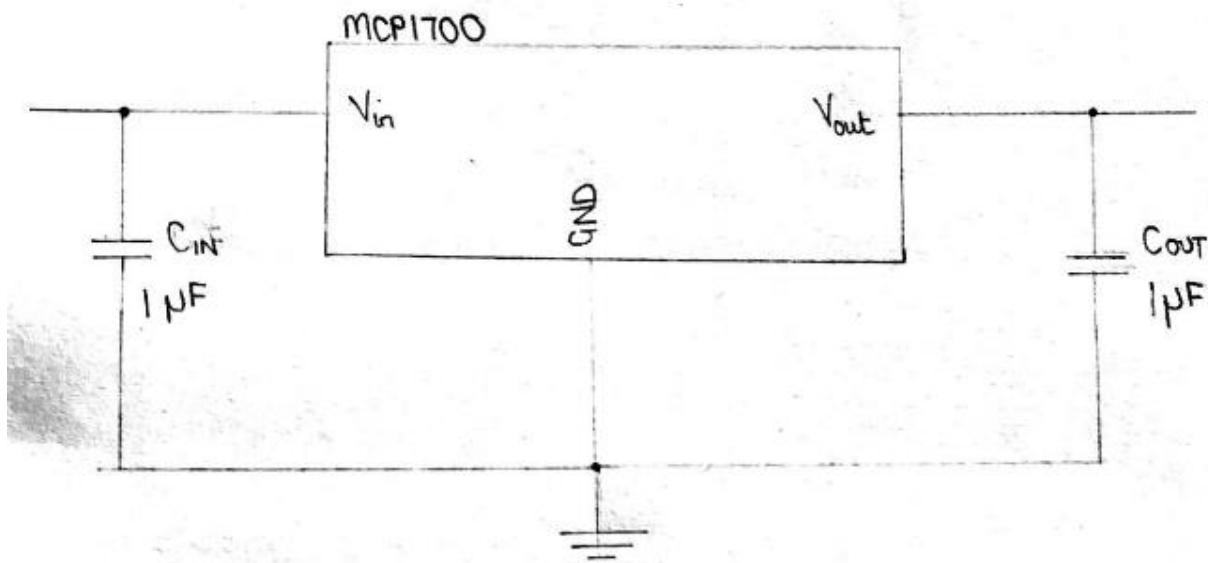


Figure 4: 3.3 V power supply schematic

The 3.3 V power supply makes use of an MCP1700 regulator that regulates the 5 V supply further down to 3.3 V. The input capacitor was included to ensure circuit stability and was the recommended value from the datasheet. A larger value could have been used, but this is only to improve AC performance, and since an AC source is not used to supply power to the board, the 1µF ceramic capacitor proved sufficient for the design. The output capacitor was also chosen as the recommended minimum value (1µF) and was included as it is required for small signal stability [3].

The following calculation was carried out to determine whether a heat sink would be required for the sub-circuit:

From the datasheet [3]:

$$T_{J(MAX)} = P_{TOTAL} \times R\theta_{JA} + T_{A(MAX)}$$

$$T_{J(MAX)} = ((3.3 \times 250 \times 10^{-3}) \times 74) + 25 = 86.05 \text{ } ^\circ\text{C}.$$

The calculated maximum junction temperature is less than the absolute maximum rated junction temperature, hence no heat sink was required.

3.2 UART communications

The USB to UART connection was predesigned [1] and is shown in the following schematic:

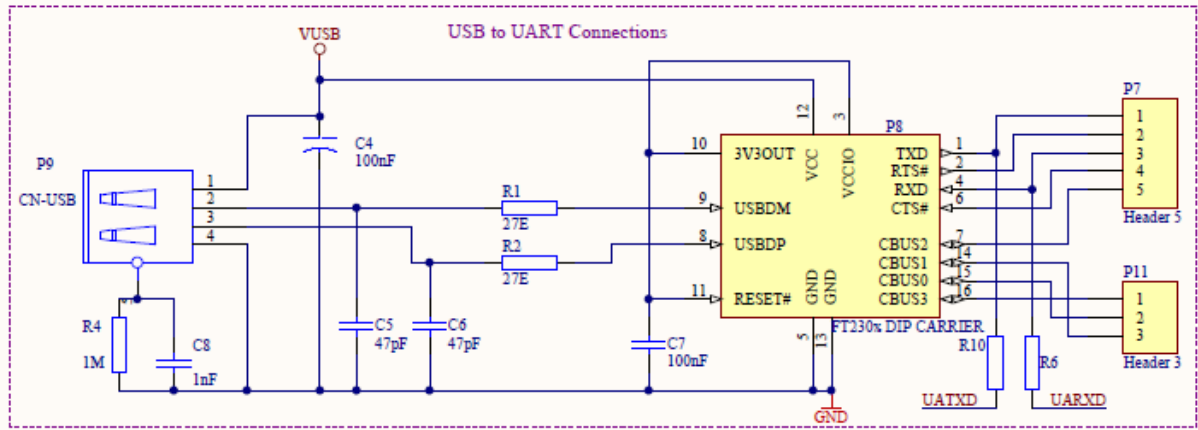


Figure 5: USB to UART circuit schematic

The components of the circuit shown above that had to be designed for were the resistors R_6 and R_{10} . These resistors were included to prevent the flow of parasitic current between the UART module (which is powered by a 3.3 V USB), and the STM32F334R8 board (which is powered by the 5V supply). The parasitic current, if larger than the maximum current supplied to a pin (25 mA from the datasheet [4]), would cause damage to the pin. The value of the resistors was thus calculated as:

$$R = \frac{V}{I} = \frac{5}{0.025} = 200 \text{ } \Omega$$

The resistors R_6 and R_{10} were thus chosen to be 1 k Ω in value, to ensure any parasitic current that flows would be very small. The datasheet [5] recommended 10 k Ω resistors, but this was deemed unnecessarily large for the design.

The baud rate used for communication over UART was 115200. Hence, for 1 bit to be transferred over UART, the necessary time for this to occur is:

$$period = \frac{1}{Baudrate} = \frac{1}{115200} = 8.86 \text{ } \mu\text{s}$$

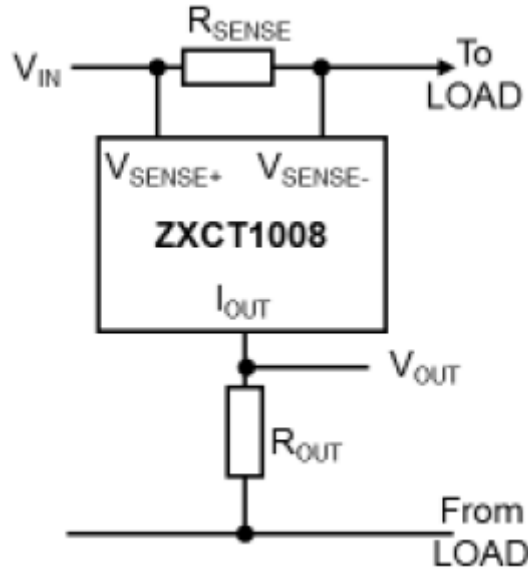


Figure 7: Current sense circuit

An output current that is proportional to the voltage drop over the sense resistor is provided by the current sensor. In this regard, a $1\ \Omega$, $1\ \text{W}$ resistor was chosen, such that the output current and the voltage drop across the sense resistor were the same in magnitude. The sense resistor was placed in series between the diode D1 (from the $5\ \text{V}$ power supply) and the input pin of the $5\ \text{V}$ regulator (U1), and as shown in the diagram above, $V_{\text{SENSE}+}$ and $V_{\text{SENSE}-}$ were connected on either side of the sense resistor, with the $V_{\text{SENSE}+}$ side corresponding to the connection to D1, and $V_{\text{SENSE}-}$ side corresponding to the connection to U1. To obtain a voltage equivalent to the output current from the sense resistor, a resistor R_{OUT} was connected between I_{OUT} and ground. The value of R_{OUT} was calculated as follows:

$$R_{\text{OUT}} = \frac{V_{\text{OUT}}}{V_{\text{SENSE}} \times 0.01}$$

$$R_{\text{OUT}} = \frac{(500 \times 10^{-3})}{((500 \times 10^{-3}) \times 0.01)}$$

$$= 100\ \Omega$$

As the current measured from the current sense circuit was to be sampled over the ADC of the STM32F334R8 module, the relatively low output voltage of the current sensor needed to be amplified to obtain a suitable voltage to be sampled by the ADC. As mentioned previously, a non-inverting operational amplifier was used to amplify the output voltage of the current sense circuit. A schematic of the amplifier, taken from its datasheet [7] is shown below:

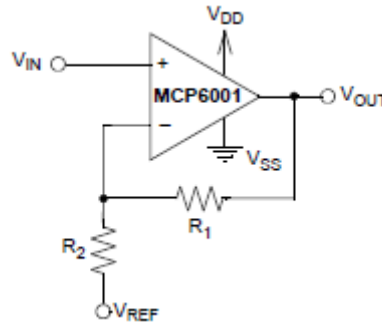


Figure 8: Non-inverting operational amplifier

The gain of the amplifier was calculated using the following formula:

$$Gain = 1 + \frac{R_1}{R_2}$$

Since the magnitude of the input to the amplifier was in the mV range, a gain of 11 was chosen, such that the highest output would be in the range $0.1 \text{ V} < V_{OUT(AMP)} < 2 \text{ V}$. With this gain, the values of R_1 and R_2 were calculated as follows:

$$R_1 = 10R_2$$

$$R_2 = 1 \text{ k}\Omega \text{ (chosen)}$$

$$R_1 = 10 \text{ k}\Omega \text{ (from above formula)}$$

3.3.2 Voltage sensing

The supply voltage to the board (voltage measured after the protection diode) was to be measured using the ADC of the STM32F334R8 module. Since the maximum voltage that can be sampled over the ADC is 3.3 V, the supply voltage (which was greater than 3.3 V), had to be lowered to a more appreciable value to prevent damage to the ADC pin (which is 3.3 V tolerant). In order to do this, a voltage divider circuit was used. The schematic of this circuit is shown below:

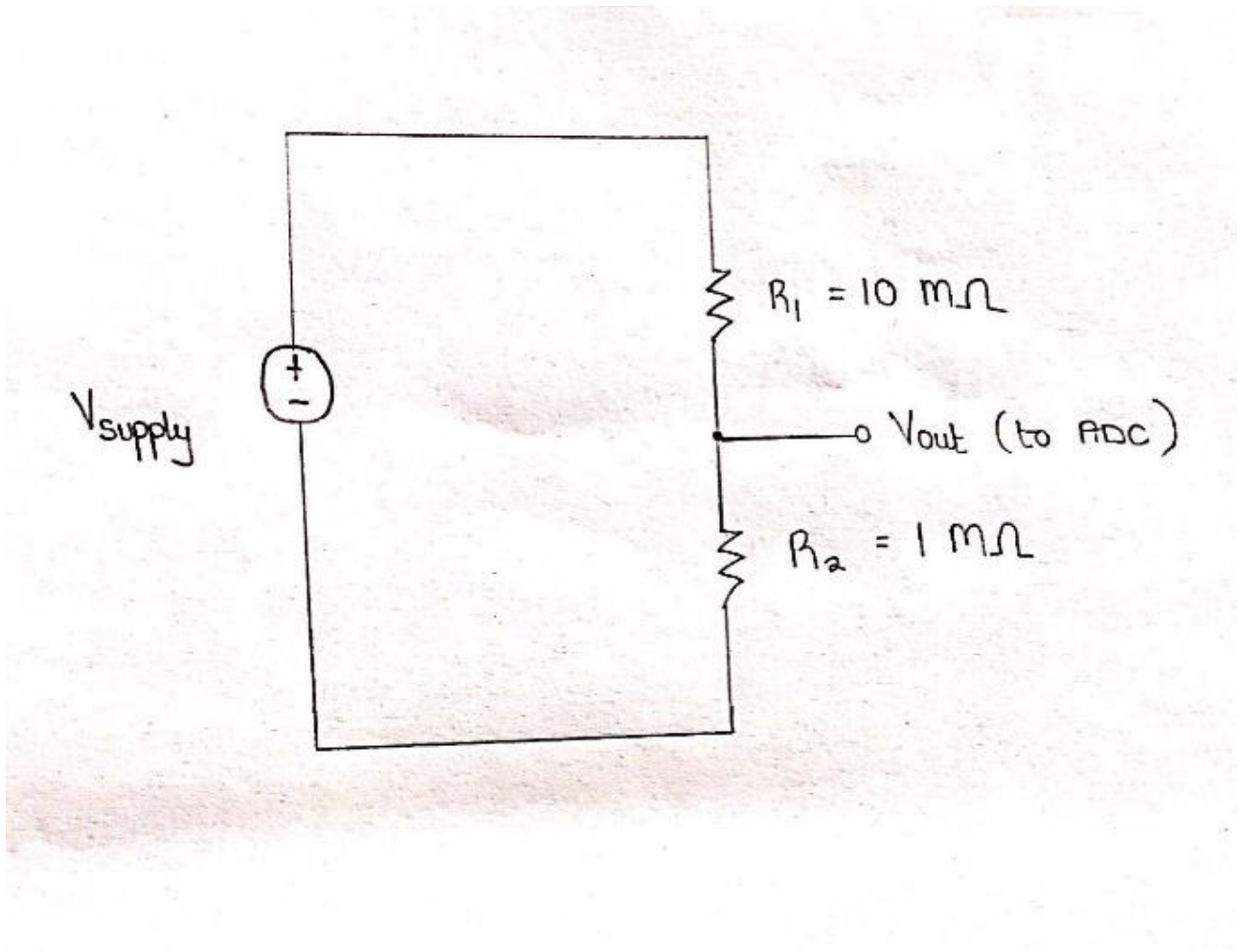


Figure 9: Voltage divider circuit

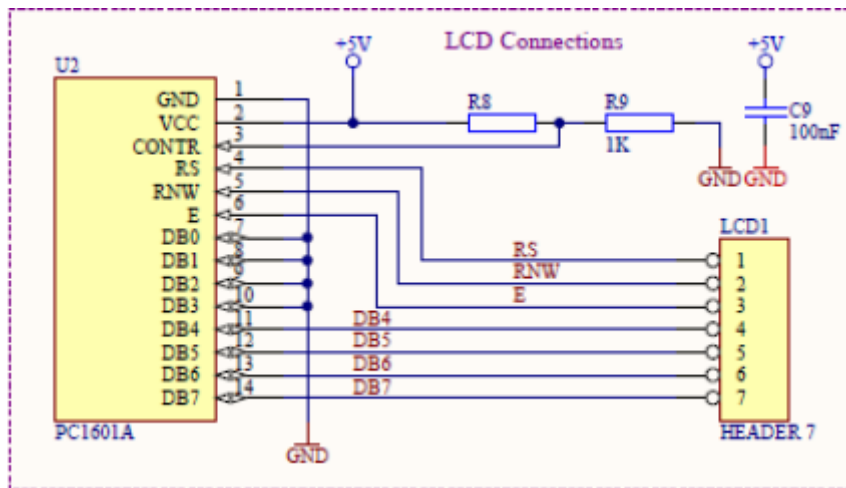
The above resistor values were chosen such that a negligible amount of current would be drawn by them, to not affect current measurements by the ADC, as well as to have a voltage gain of 11 (preselected). The output voltage of the voltage divider was calculated as follows:

$$V_{out} = \frac{R_2}{R_1 + R_2} \times V_{supply}$$

For a maximum assumed supply voltage of 9.9 V (9.2 V after the diode drop), the output voltage of the voltage divider circuit, from the above formula, would be 0.9 V, which is within the acceptable range of input voltages to the ADC pin.

3.4 LCD

The LCD used in this design was a PowerTip 1601 LCD, and was used in nibble mode, therefore only seven connections in total were required to communicate with the LCD. The schematic of the LCD [1], is shown below:



The pins that were to be used to interface with the LCD were DB4 to DB7, as well as RS, E (Enable) and RNW (Read/Write), with DB0 to DB3 connected to ground. The contrast of the LCD was the only aspect of the component to be designed for. It was observed that the contrast was dependent on the ratio between the resistors R8 and R9, and after testing different combinations, values of 18k Ω and 1.2k Ω for R8 and R9 respectively were found to give a contrast which made for easy viewing. The range of values for the resistors to be used was obtained from [8].

3.5 BME280 Temperature, humidity and pressure sensor

A schematic of the BME280 module, taken from its datasheet [9] is shown below:

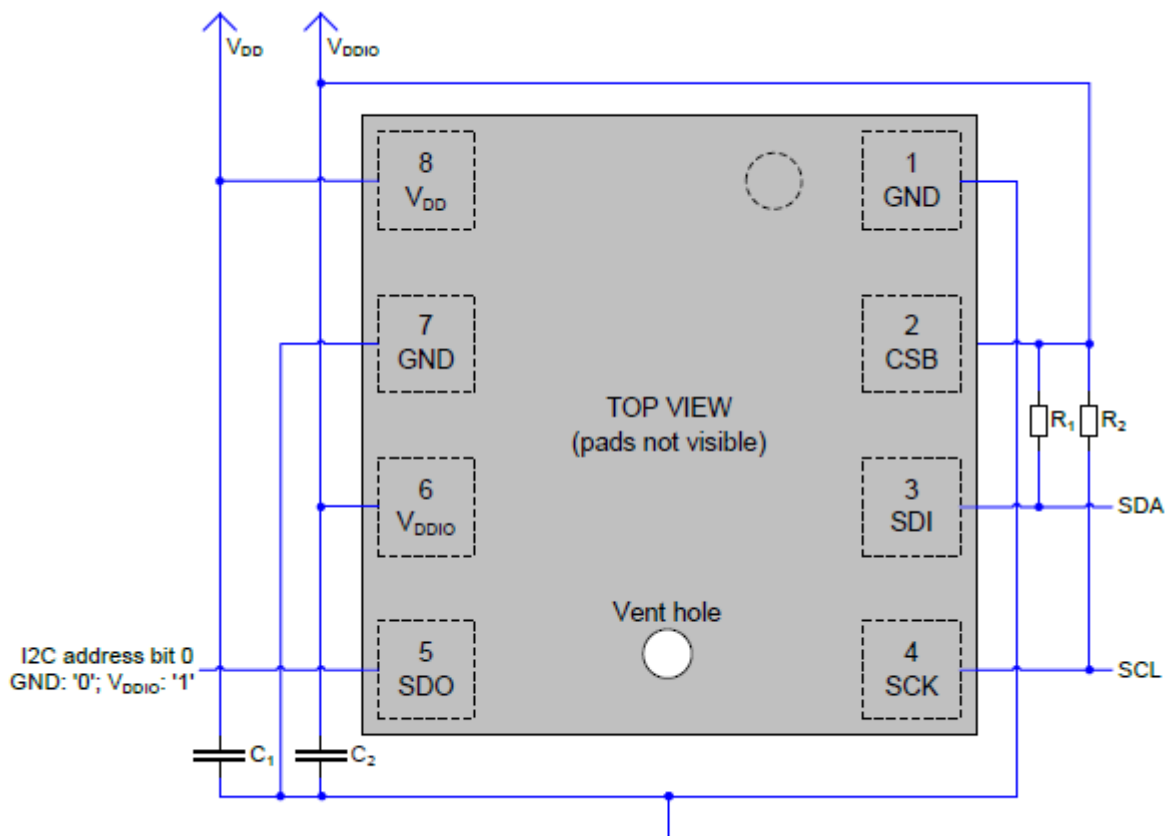


Figure 11: BME280 I²C connection diagram

The BME280 sensor was to be used in I²C mode, due to a flaw in the design of the manufactured PCB module on which it was placed (see Appendix A). The capacitor C_1 was removed from the module, and C_2 (value 100 nF, as recommended in datasheet [9]) was already included on the PCB module, hence, the only components to be designed for were the resistors R_1 and R_2 . From the datasheet [9], the recommended value for both resistors was 4.7 k Ω , and this value is dependent on the bus load and interface timing. It should also be noted that V_{DD} and V_{DDIO} were both connected to the same 3.3 V supply.

3.6 LIS2DH12 3-axis MEMS accelerometer

The following is a schematic of the LIS2DH12 module [10]:

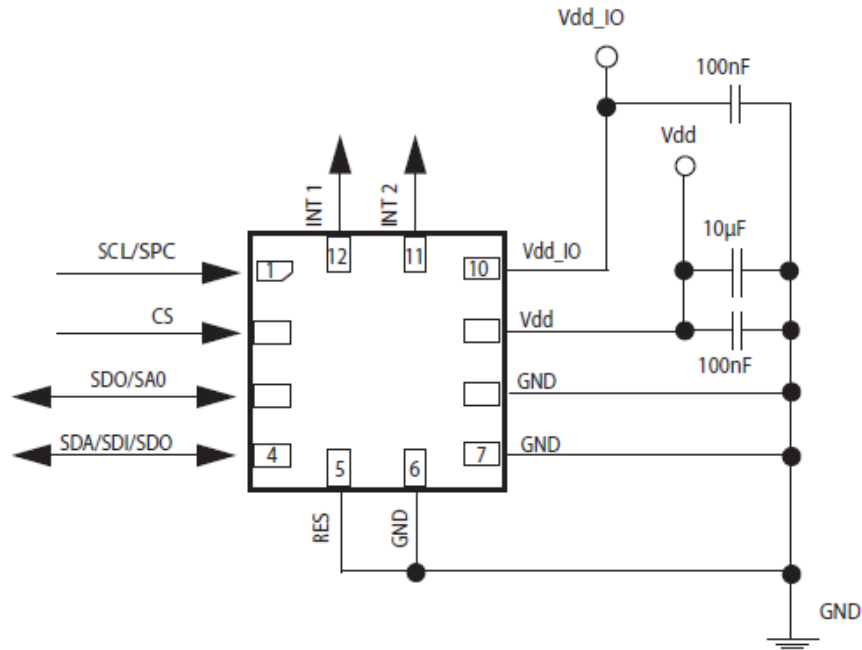


Figure 12: LIS2DH12 accelerometer connection diagram

The accelerometer was chosen to be used in I²C mode, and due to this decision, the SDA and SCL pins had to be connected to the same node as R_1 and R_2 (from the BME280 module sub-circuit) respectively, since it shares the same I²C bus. V_{DDIO} and V_{DD} were connected to the same 3.3 V supply, and decoupling capacitors were added as recommended in the datasheet [10].

3.7 Release mechanism signal

The release mechanism signal (also known as the burn signal), was mostly software driven. A free pin of the STM32F334R8 was configured as GPIO output, and after certain conditions were satisfied (altitude above 10km and longitude outside range 17.976343 deg to 18.9354 deg), the pin would be set to 3.3V for ten seconds, after which it would reset.

4 Software design and implementation

The software development tools used were Atollic TrueStudio and STM32CubeMX. This section provides a discussion on the software design and implementation.

4.1 Layout of Project

The following table consists of the files that were added and edited for use in development of the program, and the description of the files.

Table 2: Project Files

Filename	Description
main.c	Necessary for operation, contains all the peripheral initializations (as generated by STM32CubeMX) and calls to functions
stm32f3xx_it.c	Contains ISR code for UART and ADC
project.h, project.c	Contains respectively: <ul style="list-style-type: none"> • Main project function declarations • Complete functions as declared in header file
variables.h, variables.c	Contains respectively: <ul style="list-style-type: none"> • Global variables and macro declarations • Global variables initialization
lcd.h, lcd.c	Contains respectively: <ul style="list-style-type: none"> • Function declaration for use in communicating with the LCD module • Complete functions as declared in header file
bme280_defs.h, bme280.h, bme280.c, bme_sensor.h, bme_sensor.c	Contains respectively: <ul style="list-style-type: none"> • Constants, macros and datatype declarations • Declarations of sensor driver APIs • Definitions of sensor driver APIs • Declarations of functions to interface with sensor • Definitions of functions to interface with sensor.
lis2dh12_reg.h, lis2dh12_reg.c, accelerometer.h, accelerometer.c	Contains respectively: <ul style="list-style-type: none"> • Declarations of accelerometer driver APIs, and of constants, macros and datatypes • Definitions of accelerometer driver APIs • Declarations of functions to interface with accelerometer • Definitions of functions to interface with accelerometer

4.2 Description of Functions

The functions that were written for use in the project are described in the following table:

Table 3: main.c and stm32f3xx_it.c files

Function	int main(void)
Description	Contains main loop in which all the program functions are called and executed.
Function	void USART1_IRQHandler(void)
Description	Contains ISR responsible for UART communication. Executed when UART interrupt is triggered.
Function	void ADC1_2_IRQHandler(void)
Description	Contains ISR responsible for ADC sampling. Executed when ADC interrupt is triggered.

Table 4: project.c file

Function	void initialise(void)
Description	The main function of the source file. Contains calls to all the functions defined for operation of the data logger. This function is then called in the main loop (minimizes the lines of code in the main loop).
Function	void get_string(void)
Description	Copies received UART message from temporary buffer to a more permanent buffer for processing.
Function	void checksum(void)
Description	Calculates checksum of GPS string and verifies that it is identical to the one contained in the GPS string.
Function	void process_gps(void)
Description	If checksum of string valid and GPS message is of type GGA, this function extracts the time, latitude, longitude and altitude from the GPS string. Release mechanism signal conditions are also evaluated here.
Function	void get_time(void)
Description	Makes use of SysTick(). Evaluates the runtime of the board since power on.
Function	void burn_signal(void)
Description	If release mechanism conditions satisfied, function toggles the state of a GPIO output pin high for ten seconds, and then resets the pin after.
Function	void sample_current(void)
Description	Samples the current every 10 ms by generating ADC interrupt. Makes use of one ADC channel.
Function	void get_current(void)
Description	When a suitable number of samples for the current has been taken, this function calculates the mean of the samples to be reported in the report string.
Function	void sample_voltage(void)
Description	Samples the voltage every 10 ms by generating ADC interrupt. Makes use of one ADC channel.

Function	void get_voltage(void)
Description	When a suitable number of samples for the voltage has been taken, this function calculates the mean of the samples to be reported in the report string.
Function	void process_string(void)
Description	Appends required data to report string to be sent over UART.
Function	void print_string(void)
Description	Transmits report string via UART every second.

Table 5: lcd.c file

Function	void lcd_init(void)
Description	Initializes the LCD so communication with the LCD can occur.
Function	void next_line(void)
Description	Scrolls the LCD to its next line so that data can be shown on entire screen.
Function	void clear_display(void)
Description	Clears the display of the LCD.
Function	void delay(int milliseconds)
Description	Used as a delay for the specified time in milliseconds so that the LCD can perform its required functions during that time.
Function	void e_cycle(int time)
Description	The enable cycle for the LCD. Used when writing data to the LCD, with a delay equivalent to the time passed as an argument.
Function	void write_lcd(void)
Description	Sets the RS pin high and the RNW pin low (required when writing data to the LCD).
Function	void upper_nibble(int a, int b, int c, int d)
Description	Sends upper nibble of data to the LCD.
Function	void lower_nibble (int a, int b, int c, int d)
Description	Sends lower nibble of data to the LCD
Function	void send_char(char c)
Description	Sends a character to the LCD.
Function	void send_str(char str[])
Description	Sends a string to the LCD.
Function	void get_lcd_data(void)
Description	Obtains, formats, and populates string with data required to be displayed on the LCD.

Table 6: bme_sensor.c file

Function	void bme_init(void)
Description	Initializes the BME280 sensor so communication with the sensor can occur.
Function	int8_t stream_sensor_data_normal_mode(struct bme280_dev *dev)
Description	Streams the sensor's data in normal mode.
Function	void user_delay(uint32_t period)

Description	For a specified period in milliseconds, this function implements a delay.
Function	int8_t user_i2c_read(uint8_t dev_id, uint8_t reg_addr, uint8_t *reg_data, uint16_t len)
Description	This function reads data from the BME280 sensor over the I ² C bus.
Function	int8_t user_i2c_write(uint8_t dev_id, uint8_t reg_addr, uint8_t *reg_data, uint16_t len)
Description	This function writes data to the BME280 sensor over the I ² C bus.
Function	void get_bme_data(void)
Description	Obtains the temperature (in °C), relative humidity and pressure (in kPa) from the BME280 sensor and populates the respective variables to be used to fill the report string.

Table 7: accelerometer.c file

Function	void accelerometer_init(void)
Description	Initializes the LIS2DH12 accelerometer so that communication with the accelerometer can occur.
Function	int32_t platform_read(void *handle, uint8_t reg, uint8_t *bufp, uint16_t len)
Description	Reads data from the accelerometer over the I ² C bus.
Function	int32_t platform_write(void *handle, uint8_t reg, uint8_t *bufp, uint16_t len)
Description	Writes data to the accelerometer over the I ² C bus.
Function	void get_acceleration_data(void)
Description	Obtains the x, y, and z accelerations (according to axis specified in PDD) from accelerometer and fills respective variables to be used in populating the report string.

4.3 Peripheral Setup

The following peripherals were setup using STM32CubeMX in order to interface with the microcontroller and for specified functionality to be satisfied. To communicate with the peripherals, the code made use of the already defined functions from the HAL library [11].

4.3.1 UART

Channel 1 of the UART peripheral was configured to transmit and receive data using interrupts. The baud rate was set to 115200 bits/s, and the format of the data was 8N1 (8 data bits, 1 start bit, 1 stop bit and no parity bit).

4.3.2 GPIO

The GPIO pins used in the project were for the release signal and to interface with the LCD. All the GPIO pins were configured as output and no alterations were made to their default settings.

4.3.3 ADC

The ADC was used to measure the voltage and current supplied to the board. To achieve this, the ADC was configured to sample the abovementioned using interrupts, with both ADC channels being used. The ADC channels used were configured as single ended input, and the rest of the settings were kept as default. Two formulas used for the sampling are provided below:

$$\text{Sampled voltage} = \frac{R_2}{R_1 + R_2} \times V_{\text{supplied}} \times \frac{(2^{12} - 1)}{3.3}$$

$$\text{Sampled current} = \frac{\text{Amplifier gain} \times V_{\text{in(amplifier)}}}{3.3} \times (2^{12} - 1)$$

4.3.4 I²C

The I²C peripheral was set up to communicate with the BME280 sensor and the accelerometer, and all the default settings were left unaltered.

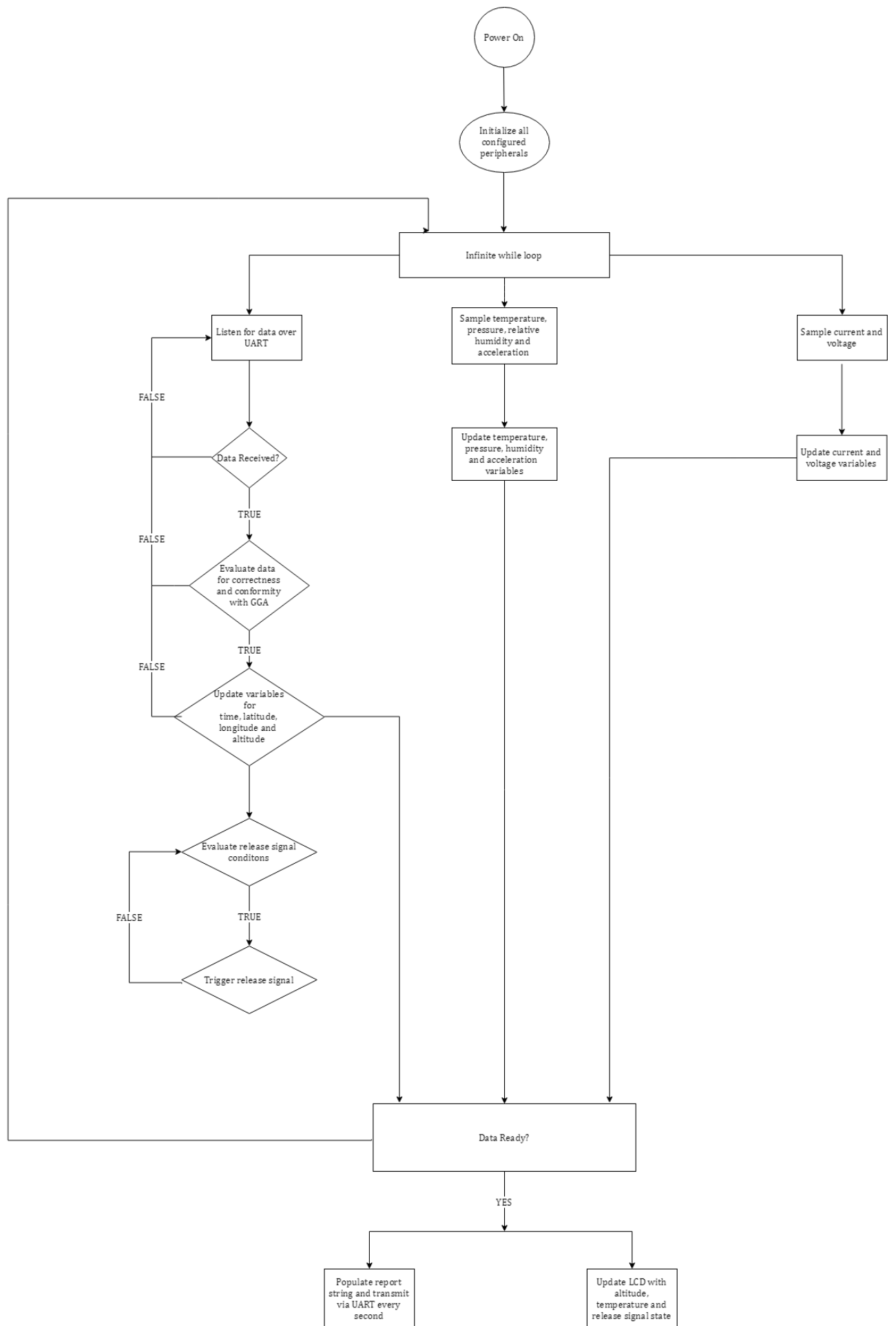
4.3.5 Timers

No timers were used in this project.

4.4 Release signal logic

The release signal conditions, as specified in the PDD, were evaluated every time a valid GGA message was sent via UART. When the conditions were satisfied, the relevant GPIO pin was set high, and then low, using already defined HAL functions. The release signal would trigger again if the conditions were satisfied, but not immediately after it had already happened.

4.5 Flow diagram



5 Measurements and Results

5.1 Power Supply

5.1.1 5 V Power Supply

Two tests were carried out to evaluate the performance of the 5V regulator: a no-load test and a load test over a range of loads. The following procedure was used to determine the performance of the 5V supply:

Requirement(s):

- The 5V power supply voltage shall be $5V \pm 0.1V$ under loads ranging from a minimum 100mA, to a maximum of 500mA.

Procedure:

A range of loads (10Ω to 46Ω) were connected to power supply output. Voltage and current measurements were then obtained using a multimeter.

Results:

Table 8: 5 V power supply results

Resistor / Ω	Output Current (I_o) / mA	Output Voltage (V_o) / V
0 (No-load)	0	5.02
10	530	5.00
15	340	5.01
33	160	5.01
46	100	5.02

The above results confirm that the regulator provided an output within the 5% tolerance range as required.

5.1.2 3.3 V Power Supply

Two tests were carried out to evaluate the performance of the 3.3V regulator: a no-load test and a load test over a range of loads. The following procedure was used to determine the performance of the 3.3V supply:

Requirement(s):

- The 3.3V power supply voltage shall be $3.3V \pm 0.1V$ under loads ranging from a minimum 100mA, to a maximum of 250mA.

Procedure:

A range of loads (10Ω to 22Ω) were connected to power supply output. Voltage and current measurements were then obtained using a multimeter.

Results:

Table 9: 3.3 V power supply results

Resistor / Ω	Output Current (I_o) / mA	Output Voltage (V_o) / V
0 (No-load)	0	3.33
10	310	3.33
15	210	3.31
22	150	3.33

Due to unavailability of a load that would give a measured output current of 250mA, which would be a 13.2Ω resistor, a 10Ω resistor was chosen, with the expectation that the maximum current of 250mA that could be provided by the regulator as stated in the datasheet would be drawn by the load. The test however gave a measured result as shown in the table above, and this was far above the expected measured value. The datasheet [3] makes mention of the peak output current being internally limited, without stating an expected value. Since this was a stress test, the difference in the expected value versus the obtained value can be attributed to the reason mentioned prior. The above results do confirm that the regulator provided an output within the 5% tolerance range as expected.

5.2 UART Communication

Requirement(s):

- The measured baud rate shall be approximately 115200.

Procedure:

An oscilloscope probe was connected to one of the UART pins of the microcontroller used for communication with the USB/UART module, and a character was sent from the microcontroller. The period was then evaluated using the oscilloscope.

Results:

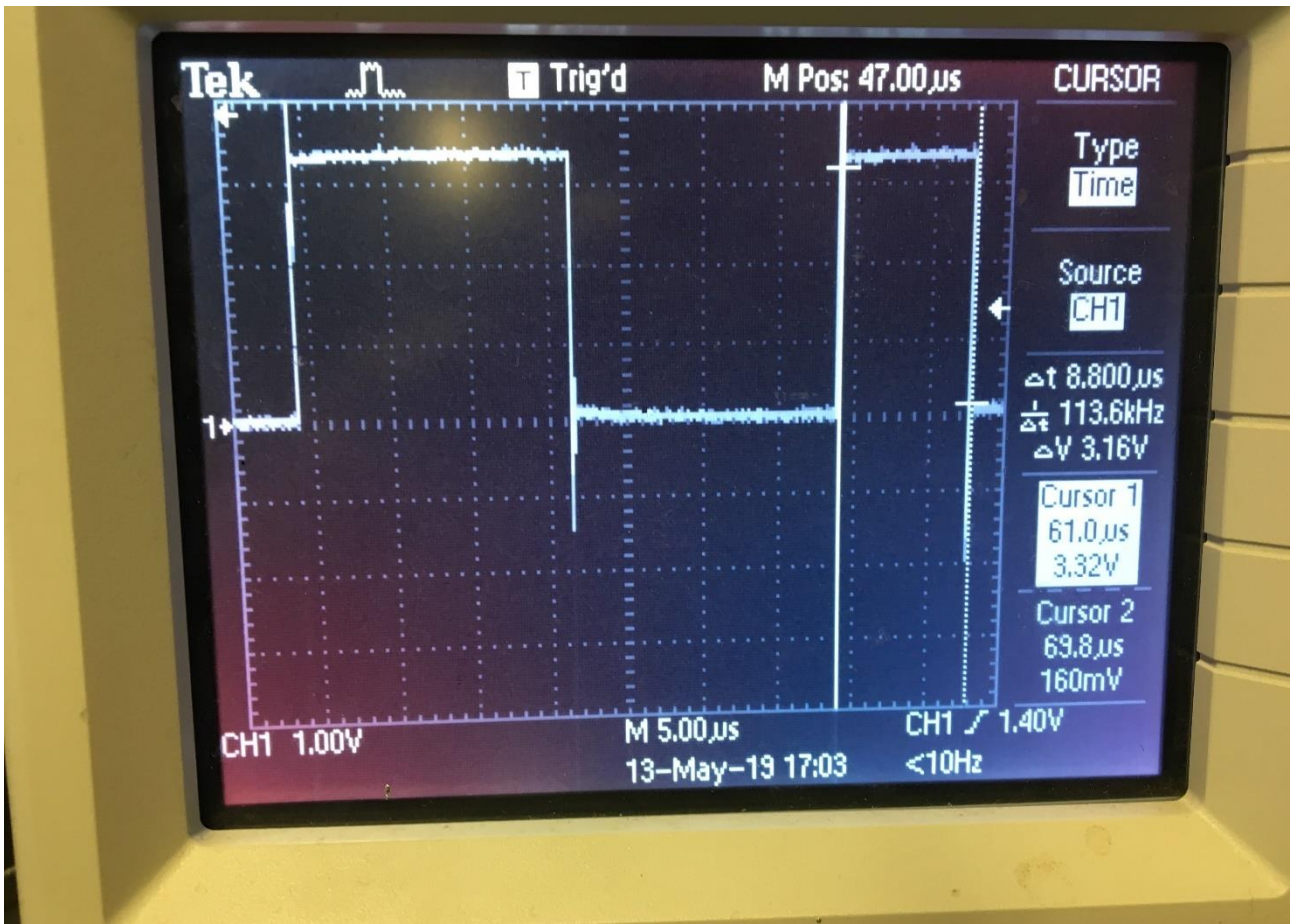


Figure 14: UART oscilloscope results

From the above graphic, the period is 8.8 μs. The baud rate is then calculated as:

$$\text{Baudrate} = \frac{1}{\text{period}} = \frac{1}{8.8 \times 10^{-6}} = 113636$$

The measured baud rate is approximately equal to calculated baud rate, with an error of approximately 1.36%, hence it was assumed that the USB/UART module was functioning as expected (since the error is very small).

5.3 Current and Voltage Sensing

5.3.1 Current Sense Circuit

Due to imperfections in the resistors used (i.e. resistor values not entirely corresponding to actual measured resistances), and the dependence of the ADC measurements on the circuit, to be able to obtain measurements as accurate as possible, the following test was carried out.

Requirement(s):

- Voltage drop across the sense resistor, R_{sense} , should correspond to output voltage across R_{out} , with a difference of $\pm 5\%$, which corresponds further to the output current from the current sense circuit.
- Measured operational amplifier gain (output voltage of amplifier / input voltage to amplifier) should correspond to calculated gain, with a difference of $\pm 5\%$.

Procedure:

The voltage drop across R_{sense} was measured using a multimeter, whereas the voltage drop across R_{out} was measured using crocodile clips connected to an oscilloscope, since the current was fluctuating, and the oscilloscope gives a mean of the sampled values (which is more accurate than a multimeter).

Results:

Table 10: Current sense circuit results

$V_{\text{source}} / \text{V}$	$V_{\text{sense}} / \text{mV}$	$V_{\text{out(sense)}} / \text{mV}$	$V_{\text{out(amp)}} / \text{mV}$	Gain
9.0	83.8	87.1	952	10.93
9.2	83.7	86.7	960	11.07
9.4	84.6	86.9	964	11.09
9.6	85.1	87.4	975	11.15
9.8	84.1	87.8	967	11.01
Average				11.05

The measured gain was then calculated as shown above and is within the $\pm 5\%$ range of the calculated value, as expected. Also, the output voltage of the sense circuit was also within the $\pm 5\%$ range of the voltage measured across the R_{sense} resistor, as expected.

5.3.2 Voltage Measurement Circuit

Due to imperfections in the resistors used (i.e. resistor values not entirely corresponding to actual measured resistances), and the dependence of the ADC measurements on the circuit, to be able to obtain measurements as accurate as possible, the following test was carried out.

Requirement(s):

- The measured gain of the voltage divider circuit should be within $\pm 5\%$ of the calculated gain

Procedure:

The voltage supplied to the source was varied from 9.0 V (minimum supply voltage) to 9.9 V (maximum supply voltage), and the voltage at the input and output of the circuit was measured using a multimeter. The gain was then calculated as the output voltage divided by the input voltage.

Results:

Table 11: Voltage divider circuit results

$V_{\text{source}} / \text{V}$	V_{in} / V	$V_{\text{out(expected)}} / \text{V}$	$V_{\text{out(actual)}} / \text{V}$	Gain
9.0	8.36	0.76	0.71	11.77
9.2	8.48	0.77	0.72	11.78
9.4	8.70	0.79	0.74	11.76
9.6	8.90	0.81	0.75	11.87

9.8	9.11	0.83	0.77	11.83
9.9	9.21	0.84	0.78	11.81
Average				11.80

As can be seen from the above table, the measured gain exceeded the $\pm 5\%$ tolerance range within which it was supposed to fall, and this can be attributed to the reason mentioned earlier. Due to this discrepancy, adjustments had to be made in the written code to compensate for the inaccuracy that would occur if the raw value of the gain was used.

5.4 LCD

Requirement(s):

- The LCD shall display the data sent to it from the SMT32 microcontroller.

Procedure:

A string was sent to the LCD and the data sent in the string was to appear on the screen.

Results:

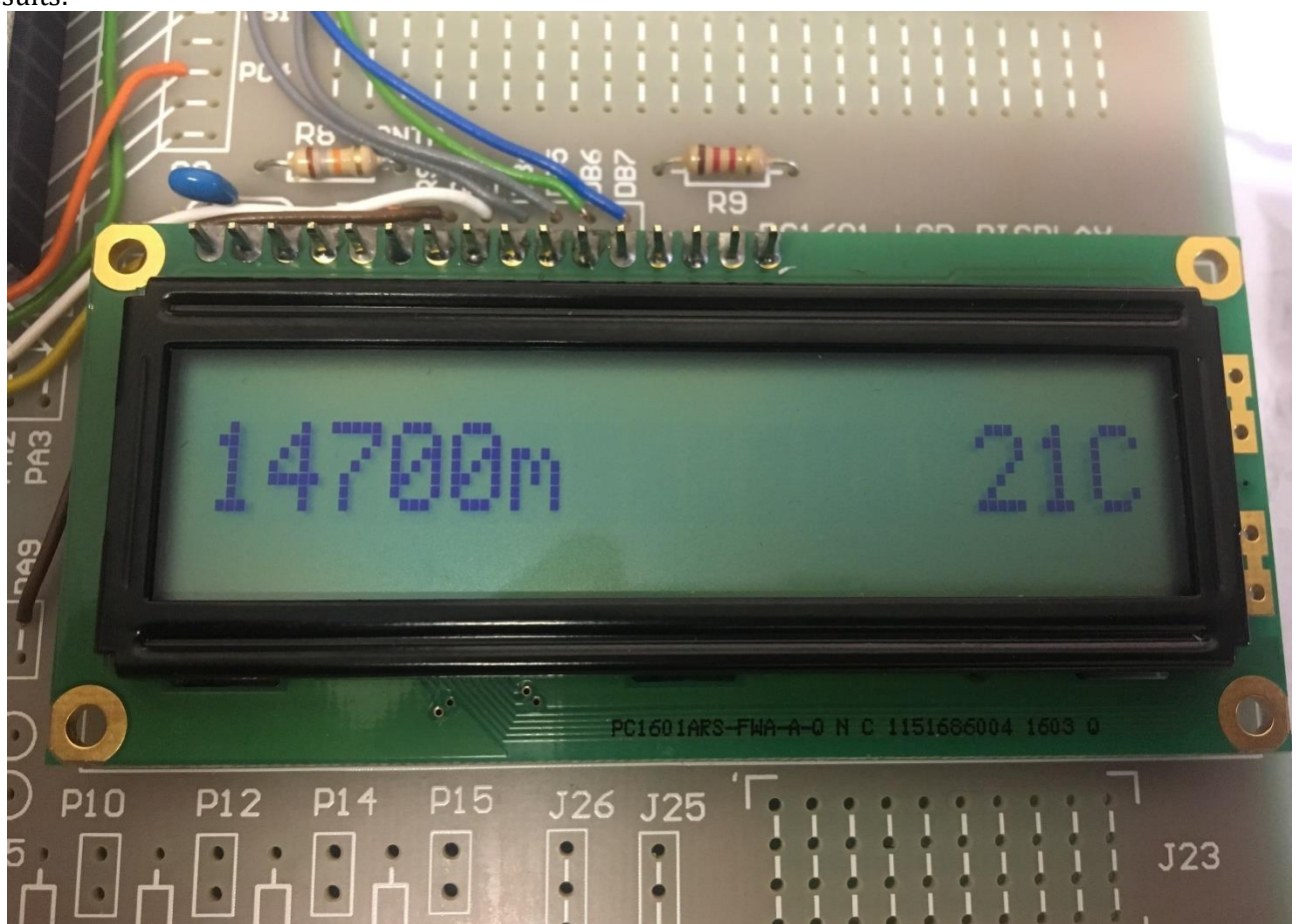


Figure 15: LCD test result

The LCD module behaves as expected.

5.5 Temperature and humidity measurement

Requirement(s):

- The temperature reported from the BME280 sensor and the measured temperature of the surroundings shall have a difference of ± 5 °C.
- The relative humidity reported from the BME280 sensor and the equivalent humidity of the surroundings shall have a difference of $\pm 10\%$.

Procedure:

Due to unavailability of a thermometer or a hygrometer, the temperature and humidity values reported by the BME280 sensor were compared to the temperature and humidity reported on weather.sun.ac.za and were found to fall within the expected ranges. In addition to this, breathing out over the sensor caused a rise in both temperature and humidity readings, hence it was assumed the device was operating as required.

5.6 Pressure measurement

Requirement(s):

- The pressure reported from the BME280 sensor and the pressure of the surroundings shall have a difference of $\pm 5\%$.

Procedure:

Due to unavailability of a barometer, the reported pressure was compared to that reported on weather.sun.ac.za and was found to fall within the range, hence it was assumed that the device was operating as required.

5.7 Acceleration measurement

Requirement(s):

- The acceleration reported by the accelerometer module shall conform to that specified in the PDD

Procedure:

The project board was positioned on a straight table and the values observed. It was then tilted in the direction of the defined axes and the values were observed again. It was found that the reported values conformed to the requirements.

5.8 Release signal

Requirement(s):

- When the release signal conditions have been satisfied, a GPIO pin shall register a reading of 3.3 V for 10 seconds, after which the voltage should drop to 0.

Procedure:

Using a multimeter and stopwatch, the voltage output of a cable soldered into the board at the relevant pin was measured, and the duration timed, when the release signal conditions were satisfied. It was found that the release signal conformed to the requirements.

6 Conclusions

After testing the system to establish whether the required functionality was satisfied, it was concluded that the system was fully compliant. The following observations were also made:

6.1 Strengths

- The programming approach was modular, easing the debugging process and allowing for extra functionality to be added without affecting already written code.
- Measurements made by the system are accurate.
- The system runs smoothly.

6.2 Weaknesses

- Certain applications (LCD code in particular) were only developed specifically for the project use. Good programming practice would entail developing code to satisfy as many solutions as possible, such that the code is reusable.
- Timers could have been used to develop delays that were as accurate as the functionality required them to be.
- The project board had to be taken in for maintenance due to damage caused by soldering.
- Unavailability of certain measuring equipment made it difficult to ascertain that system performs as accurately as could be expected.

6.3 Improvements and suggestions

- Use higher quality PCBs to avoid as much damage to the tracks caused by soldering.
- Design code to satisfy as many applications as possible for code reusability.
- Make use of all resources available i.e. datasheets, online forums, fellow developers.
- Use of measuring equipment tailored to measure specific values (such as humidity).

7 References

- [1] A. Barnard, "Project Definition Document," Feb. 2019 [Revised May. 2019]. [Online]. Available:
https://learn.sun.ac.za/pluginfile.php/1670833/mod_resource/content/1/OntwerpE314-Projek%20v0.8.pdf.
- [2] STMicroelectronics NV, "L7805CV Regulator," L78 Datasheet, Jun. 2004 [Revised Sept. 2018].
- [3] Microchip Technology Incorporated, "MCP1700-Low-Quiescent-Current-LDO-20001826E," MCP1700 Datasheet, Nov. 2005.
- [4] STMicroelectronics NV, "STM32F334x4 STM32F334x6 Datasheet," 2014 December. [Online]. Available:
https://learn.sun.ac.za/pluginfile.php/1505261/mod_resource/content/1/en.DM00097745.pdf.
- [5] Future Technology Devices International Limited, "FT230X USB TO BASIC UART IC," FT230X Datasheet, Dec. 2011 [Revised May. 2016].
- [6] Diodes Incorporated, "INCREASED AMBIENT TEMPERATURE HIGH-SIDE CURRENT MONITOR," ZXCT1008 Datasheet, Dec. 2017.
- [7] Microchip Technology Incorporated, "1 MHz, Low-Power Op Amp," MCP6001/1R/1U/2/4 Datasheet, Jun. 2002 [Revised Nov. 2009].
- [8] A. Barnard, "Lecture_7 LCD," Feb. 2017.
- [9] Bosch Sensortech, "Combined humidity and pressure sensor," BME280 Datasheet, Nov. 2012 [Revised Sept. 2018].
- [10] STMicroelectronics NV, "MEMS digital output motion sensor: ultra-low-power high-performance 3-axis "femto" accelerometer," LIS2DH12 Datasheet, Aug. 2013 [Revised May. 2017].
- [11] STMicroelectronics NV, "Description of STM32F3 HAL and low-layer drivers," December 2014. [Online]. Available:
https://learn.sun.ac.za/pluginfile.php/1505265/mod_resource/content/1/en.DM00122016.pdf.

8 Appendices

8.1 Appendix A – Circuit Diagrams

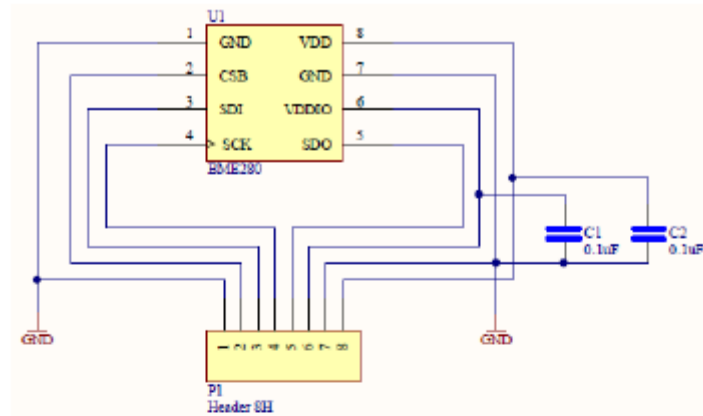


Figure 8: Original BME280 module schematic

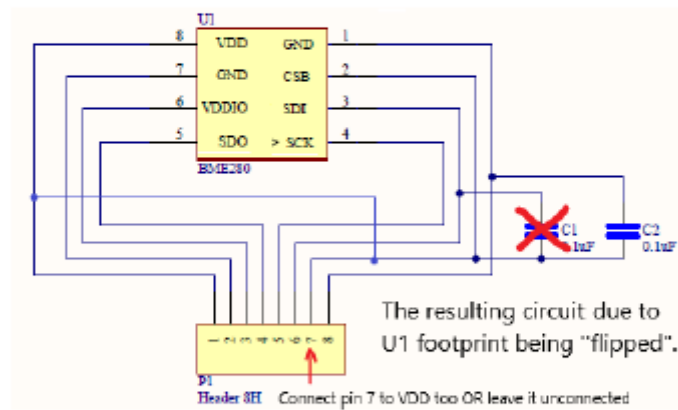


Figure 9: Fixed BME280 schematic to match manufactured module

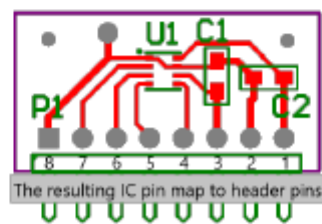


Figure 10: New BME280 module pinouts

Figure 16: BME280 schematics (incorrect and corrected)
Source: Adapted from [1]

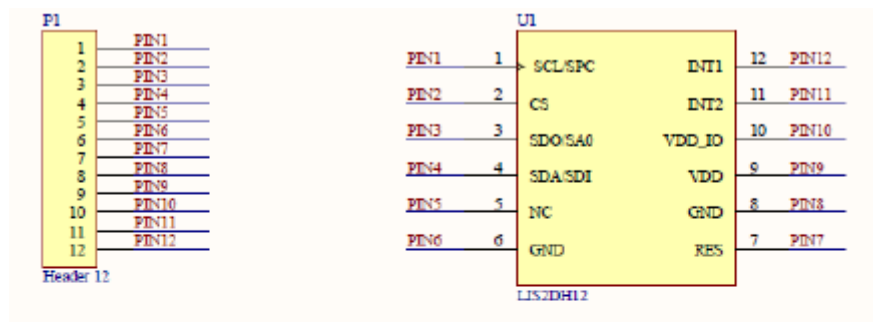


Figure 11: LIS2DH12 module schematic

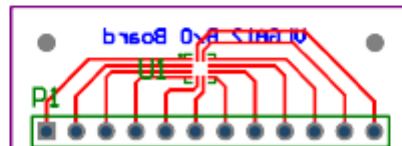


Figure 12: LIS2DH12 module pinouts

Figure 17: LIS2DH12 schematics

Source: Adapted from [1]

8.2 Appendix B – STM32 module pinout table

Table 12: STM32 module pinouts

Pin	Configuration	Connection to hardware element
PA0	GPIO_Analog ADC1_IN1	Output of voltage divider circuit
PA8	GPIO_Output	
PA9	GPIO_Output	RNW pin of LCD
PA10	GPIO_Output	E pin of LCD
PA11	GPIO_Output	DB4 pin of LCD
PA12	GPIO_Output	DB5 pin of LCD
PA15	I2C1_SCL	SCL pin of BME280 sensor SCL pin of LIS2DH12 module
PB5	GPIO_Output	RS pin of LCD
PB7	I2C1_SDA	SDI pin of BME280 sensor SDI pin of LIS2DH12 module
PC0	GPIO_Analog ADC2_IN6	Output pin of amplifier from current sense circuit
PC4	USART1_TX	J13 pin 6 (FT230X RX)
PC5	USART2_RX	J13 pin 4 (FT230X TX)
PC6	GPIO_Output	DB6 pin of LCD
PC8	GPIO_Output	DB7 pin of LCD