This project aims to design code for interface box MCB200 and develop a corresponding computer software for engineers. This will help engineers to easily write and calibrate data for the chips from different companies, which is an important part of batteries production. The project will involve testing based on V-model to ensure that the system is reliable and robust.





Microchip Programming and Calibration

Software and Program Design for BMS Production

Zhiyuan Yu

**Author:** Zhiyuan Yu

**Student number:** 00091654

**Contact e-mail:** yu0021@hz.nl

**Company address:** Informaticastraat 20, 4538 BT Terneuzen

**In-company Mentor:** Rens van Bolderik

**1st examiner**: Willem Haak

**2nd examiner**: Marjoleine van Dootingh

# Glossary

|  |  |
| --- | --- |
| **Term** | **Full name** |
| App | Application |
| BMS | Battery Management System |
| CAN | Controller Area Network |
| GUI | Graphic User Interface |
| IDE | Integrated Development Environment |
| PC | Personal Computer |
| TI | Texas Instruments |
| UART | Universal Asynchronous Receiver/Transmitter |
| UI | User Interface |
| USB | Universal Serial Bus |
| IC | Integrated Circuit |

Table of Contents

[Glossary 1](#_Toc133317496)

[1. Introduction 3](#_Toc133317497)

[1.1 Background 3](#_Toc133317498)

[1.1.1 Company Background 3](#_Toc133317499)

[1.1.2 Research Background 3](#_Toc133317500)

[1.2 Problem Analysis 4](#_Toc133317501)

[1.2.1 Present situation 4](#_Toc133317502)

[1.2.2 Desired situation 4](#_Toc133317503)

[1.3 Research Goals 7](#_Toc133317504)

[1.4 Research Objectives 7](#_Toc133317505)

[1.4.1 Three critical criteria 8](#_Toc133317506)

[1.5 Problem statement 9](#_Toc133317507)

[1.5.1 Main question 9](#_Toc133317508)

[1.5.2 Sub questions 9](#_Toc133317509)

[2. Theoretical Framework 10](#_Toc133317510)

[2.1 Search Plan 10](#_Toc133317511)

[2.1.1 Key words 10](#_Toc133317512)

[2.1.2 Process of information gathering 10](#_Toc133317513)

[2.2 Configuration of BMS programming 11](#_Toc133317514)

[2.2.1 Battery management system 11](#_Toc133317515)

[2.2.2 Microchip suppliers 12](#_Toc133317516)

[2.2.3 User Interface concept design 13](#_Toc133317517)

[2.3 Hardware 15](#_Toc133317518)

[2.3.1 PIC Kit4 debugger/programmer 15](#_Toc133317519)

[2.3.2 MCB200 interface box 16](#_Toc133317520)

[2.3.3 PIC18F27Q84 16](#_Toc133317521)

[2.4 Software environment 16](#_Toc133317522)

[2.4.1 Battery management studio 17](#_Toc133317523)

[2.4.2 MPLAB 17](#_Toc133317524)

[2.4.3 Visual Studio 18](#_Toc133317525)

[2.5 Method for mass production of batteries 19](#_Toc133317526)

[2.6 Data sheet 20](#_Toc133317527)

[2.7 S-rec document 21](#_Toc133317528)

[2.8 Hex dump 21](#_Toc133317529)

[2.9 Bootloader 22](#_Toc133317530)

[2.10 Basement of code design 23](#_Toc133317531)

[2.11 Protocols 23](#_Toc133317532)

[2.11.1 UART 23](#_Toc133317533)

[2.11.2 I2C 24](#_Toc133317534)

[3. Methodology 27](#_Toc133317535)

[3.1 Choice of methods 27](#_Toc133317536)

[3.1.1 V-model 27](#_Toc133317537)

[3.1.2 Oskam Method 28](#_Toc133317538)

[3.1.3 Choice between them 28](#_Toc133317539)

[4. Schedule and preconditions 31](#_Toc133317540)

[5. Reference 32](#_Toc133317541)

[6. Appendix 33](#_Toc133317542)

[6.1 Test Plan for software and program 33](#_Toc133317543)

# Introduction

This chapter explains the current situation and expected results of this project.

## Background

This project is under the supervision of company EmergoStar and HZ University of applied Sciences. Sustainable solution of problem under real situation is expected.

### Company Background

EmergoStar is a developer and manufacturer of Lithium batteries which offers sustainable solution for projects of different scales. The company locates in Terneuzen, Netherlands while also have industry in Hangzhou, China. The company focuses on utilization of renewable energy and other relevant fields.

Figure EmergoStar in Terneuzen (EmergoStar, 2023)

### Research Background

EmergoStar develops its own Battery Management System (BMS) boards to manage Lithium batteries. Those BMS boards are developed with microchips from different companies. Presently, the batteries are programed with software from microchips manufacturers which are not user-friendly enough for mass production. Design for a new set of software is scheduled by the company. Also, program for interface boxes is necessary in the process which is another topic.

Presently-used microchips are from Texas Instruments (TI) and Maxim Integrated. At the same time, there are also batches of BMS boards based on previous ICs from Top bands. The principles of programming are similar, while access to memories varies from brands.

For sustainability of problem solution, software should be written in C code for the PIC microcontroller in the environment of MPLAB. C# is selected for UI design phase.

## Problem Analysis

This chapter discuss about the content of this research project.

The end results of the project will be packaged in two parts: software code designed in Visual Studio and a set of program designed in MPlab.

### Present situation

EmergoStar is developing a self-designed software for programming Battery Management Systems, aiming to replace the inconvenient software provided by hardware suppliers and improve production performance. The company hopes to benefit from the convenience independence of this new software platform. Code for interface box program is also needed to work together with the software for data transmission.

Multiple series of code will be designed separately depending on the differences of chips. A graphical user interface is required for user-friendly operation, and an instruction booklet will be provided in the appendix for guidance.

The assignment is divided into three parts:

1. Software design for TI and Maxim microchips.
2. Software design for Top Band microchips.
3. Bootloader design for microchip PIC18.

Currently, battery programming is done through interface boxes and software from TI. A new software design based on the interface box MCB200 is required. Once completed, similar functions should be realized for other battery batches with boards from Maxim and Top Band.

In addition, a bootloader for PIC18 is needed to initialize the C# environment with I2C or CAN connections, as data loss issues have been observed during independent PC operation. A reliable bootloader is expected to solve this problem.

### Desired situation

This part describes the desired situation with specific criteria.

#### Process of data transmission

The figure below describes the basic principle of interface among battery, interface box and PC. The connection between PC and MCB200 is RS485/USB.

Program for MCB200 will be designed to transmit data through I2C. Additionally, software needs to be developed for the PC that has a clear Human Machine Interface, which includes a block for file input regarding programming and verification, as well as a block for entering Name/Version/Serial Number & Password.

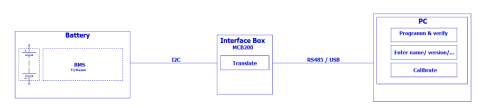


Figure Interface between hardware

After writing and testing the C program, it can be compiled and loaded onto the MCB200 interface box using the MPLAB IDE's built-in programming tools. Once the program is running on the MCB200, it will enable the interface box to communicate with the BQ78350-R1A chip and write data to it via the I2C protocol.

To further define the “data” in this process, the data consist of information of flash memory and instruction memory. According to instruction books from company TI, the registration addresses of value settings are already known. For the BQ78350-R1A we are using this time, the flash memory can be mapped into the data memory.

To this extent, we can say that the programming of flash memory has the equal effect with programming the data memory.

While for the instruction memory, commonly, it can be a commercial confidence. Without the instruction from hardware supplier, it is almost impossible to transfer the accordingly data. The communication between the client and the suppliers are continuously proceeded.

#### End results

This project focuses on three brands of microchips, software designed for PC and program designed for interface box are needed. As explained before, the assignment consists of three parts. To classify the end results in a more understandable way, the end results will mainly have two parts. The code will be packaged in two parts. One is program designed for inter box which will be executed in the environment of MPlab. The other part is software designed in Visual Studio.

As mentioned by in-school examiners, the success criteria for this project needs to be defined beforehand. During the discussion with the client, the expected results are as follow. Some results rely on the attitude of hardware suppliers.

Below is a table where end results are briefly listed. More details are in the attachment “Portfolio”.

|  |  |
| --- | --- |
| Results | Description |
| Code for Software (on PC) | A set of code designed in Visual Studio and written in C#. It should contain UI interface with at least 4 functions for document selection, programming, change setting and calibration. Default value for password can be stored.  Except the UI interface, the software will also subtract the data from the selected document and transmit it to MCB200. |
| Code for program (on MCB200) | A set of code which can do programming and calibration for microchips from TI/Maxim and Top band. At least it should be able to deal with the flash memory. |
| Code for bootloader | A set of code for bootloader to initialize the basic settings of C language and CAN protocols for updating the batteries firmware. |
| Packaged files with information | Further information may be needed to help engineers from the client company to use and alter the code. Data sheets of address, instruction booklets will be delivered. Extra contents may be added if necessary. |
| A thesis report | A report which describes the whole process of the research. From orientation to execution and then the results and recommendation. |
| Portfolio | A document which would explain and prove the competencies of the author. |
| Presentation | A presentation which provides all deliverables to be assessed. |

Table Brief description of end results

#### Potential risks

This research project proposal is written to ensure a structured and organized process, while the final results may not be fully the same as listed in this proposal. The change of results may happen on purpose or due to objective reasons.

It is common that the demands of clients may change with the current situation. On March 31, 2023, the product requirements of this project changed for the first time, which leads to some changes in the orientation of this project. Reading from the S-rec document is no longer one of the cores of the design. Instead, data will be stored in other types of documents in arrays.

Furthermore, it can’t be denied that there is a risk that some of the tasks may not be finished at the end.

There are several reasons:

1. During the preparation phase, the chips from Top band wasn’t discussed by the supervisor and the company. The extra work load and technical difficulties may cause a postpone of some tasks.
2. Some technical issues are commercial secrets, which may not be shared by hardware suppliers. i.e., the access to Instruction memory.

These potential risks have been discussed with the client, and they admit that those can impede the content of final products. The change of final production should follow the principles that:

1. If there is any change in expression of tasks, it should be recorded. The substitution of deliverable should happen in the same level. The examiner will be announced at the first time.

2. If there is any difficulty which is out of control, it should be precisely recorded for later assessment among the mentor and examiners. The difficulties encountered and associated conclusions are also part of the research.

On the other hand, the three critical criteria for final products should always be fulfilled. Those criteria will be explained in the later chapter. (Chapter 1.4.1)

## Research Goals

The goal of this project is described from two aspects:

1.Develop an effective solution for the research problem.

2.Prove that the author is capable of Engineering Bachelor competency.

## Research Objectives

The objectives of this research project are：

1.To develop a software which can handle the programming/calibration of Microchips from several suppliers.

2. To test the software mainly for verification of user-friendliness.

3. To develop a set of program for MCB200 to program Microchips from suppliers.

4. To test some present cases provided by the company to verify the functions of program for MCB200.

### Three critical criteria

To ensure the quality of this project. Three critical criteria are derived from the demand of client.

1. **The accuracy of the data transmission.**
2. **User-friendliness.**
3. **The stability of the program on MCB200**

To further explain the three criteria, the following are reasons and explanation:

1. Data transmission accuracy refers to the ability of a software system to ensure that data is transferred reliably and without errors between different components of the system. (*Tanenbaum* & Wether*all ,2011*) Inaccurate or incomplete data transmission can lead to various problems. In the case of the MCB200 interface box, which is used to program chips from TI and even more companies, the accuracy of data transmission is essential to ensure that the program is loaded correctly onto the chip. To ensure the accuracy of data transmission, the software must be designed with robust error handling and data validation that can detect and report errors in real-time. After the programming/calibration process ends, hardware provided by manufacturers of chips will be used to compare data read from the chips and the raw data from documents.
2. User-friendliness refers to the ability of a software system to provide a positive and easy-to-use experience for its users. This includes various aspects such as the design, layout, functionality, and accessibility of the user interface. UI interface will be included with functions appointed by the client. User-friendliness also helps reduce errors and misunderstandings, as users are less likely to make mistakes leading to malfunctions of outputs or inputs. Inquiry of advice from the client will prevent the deviation of design.
3. One of the significant reasons for the client company to develop software on their own is the frequent errors in the accompanying software provided by suppliers, which has caused inconvenience for the engineers who have to repeatedly restart the software, enter passwords, etc. Developing software with high stability can improve product quality and enhance the experience of engineers by saving time and increasing efficiency.

The project has three parts as defined by the client while the results will be packaged in two group of code, one designed in MPlab for program in MCB200 and one designed in Visual Studio for software on PC.

The specific sub-systems, components and test plan design for them will be listed in the appendix.

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Standard** | **Proof** |
| Accuracy of data | The data should be 100% completely and correctly stored in the right address. | The test results, based on comparison between calibrated value and the required data. |
| User-friendliness | The software can provide users with appointed functions in a clear view. | The comments and marks from software engineers in the client company. Their approvement will be a clear proof of quality. |
| The stability | The MCB200 can correctly write data for 20 consecutive times. | The test results, based on the work performance without error for 20 tasks in one time. |

Table The standards of 3 critical criteria

## Problem statement

The research questions are listed below.

### Main question

What is the code for program and software in the system for mass production of Battery Management System?

### Sub questions

The sub questions for this project are：

1. What is the description of the software on PC for BMS mass production?
2. What is the test plan and results for the software on PC for BMS mass production?
3. What is the integration of code for the software on PC for BMS software?
4. What is the description of the program for programming and calibration for TI/ Maxim chips?
5. What is the test plan and results of the program for MCB200 to program/calibrate TI/Maxim chips?
6. What is the integration of code for program for MCB200 to program/calibrate TI/ Maxim chips?
7. What is the description of the program for MCB200 to program/calibrate Top band chips?
8. What is the test plan and results of the program for MCB200 to program/calibrate Top band chips?
9. What is the integration of code for program for MCB200 to program/calibrate Top band chips?
10. What is the description of bootloader for PIC18F27Q84?
11. What is the test plan and results of bootloader for PIC18F27Q84?
12. What is the integration of code for bootloader for PIC18F27Q84?

# Theoretical Framework

This chapter focus on relevant academic literature.

## Search Plan

### Key words

**PC software design, data transmission, microchips, MCB200, protocols**

### Process of information gathering

Throughout the search, a variety of information will be utilized. Including academic papers, formal instruction books, industry forums, manufacturer websites and in-company tutorial. All sources will be precisely documented in the appendix.

Below is a brief description of what and how information will be described in this research. Because the needed information is supposed to support the answer of main and sub-questions, the steps are:

1. Understand Battery Management System (BMS) mass production programming requirements:
   1. Research the key components of programming for BMS mass production, including hardware, software, and programming tools.
   2. Define the software requirements, including programming languages. Research on user interface (UI) design, particularly for applications dealing with document selection, data import, data modification, and calibration.
   3. Study existing solution based on I2C protocol, describe the common principles.
2. Clarify the customized feature of microchips from different suppliers.

**A.** Study protocols involved in this project, including CAN, UART and I2C.

**B.** Deliver a corrected data sheet for TI, Maxim and Top band. Describe the method to program Top band chips.

1. Research on bootloader:
   1. Research bootloader requirements, i.e., functionality, compatibility, etc.
   2. Investigate existing methods for developing a custom bootloader.

|  |  |
| --- | --- |
| **List of topics** | **Source type** |
| Microchip programming | in-company guidance, forum of suppliers or technicians, websites of suppliers, research papers, books |
| BMS description | In-company guidance, research paper |
| Tools involved in research | forum of suppliers, tutorial from suppliers, websites of suppliers |
| PC Software Design | In-company tutorial, forum of suppliers, books, existing examples |
| Bootloader | forum of suppliers, websites of suppliers, books, instruction from suppliers |

Table Resource type for field of information

|  |  |
| --- | --- |
| **Quality Criteria** | **Corresponding Proof** |
| Completeness | Include all aspects of the problem statement. |
| Provide sufficient evidence to support the concept made in the proposal. |
| Clear and concise research objectives and research questions. |
| Deepness | Analyze the problem statement from multiple perspectives. |
| Apply relevant theories and frameworks to support the analysis. |
| Describe the process on level of binary signal, not only on capsuled block of code. |
| Discuss potential limitations and challenges to the research. |
| Variation of Information | More than 5 types of sources are included in the final report. i.e., Books, experts, forums, official websites, etc. |
| Use a variety of search terms and databases to ensure comprehensive coverage. |

Table Quality criteria of search plan

## Configuration of BMS programming

This part will mainly discuss about technical components for product.

* + 1. Battery management system

The Battery management system (BMS) is the core of a battery pack which includes PCB board and electronic components. The purpose of the BMS board is mainly to monitor and manage the performance of the battery. (EmergoStar, 2020)

As explained by the client company, the BMS act as the master of the battery performance,

This research project focus on the software part of the BMS production. After the consolidation of the PCB board, the programming and calibration of data will be processed, where the software and program are needed.

Figure BMS from EmergoStar (own figure)

### Microchip suppliers

Microchips from TI, Maxim and Top band are separately embedded on different batches of BMS. Those chips are designed for the same tasks and with similar characteristics. The proof is that three types of chips are all applied on X30 series of lithium battery pack in EmergoStar. (EmergoStar, 2023) The main difference among them is the data sheet with the address. The calibration of data sheet is part of pre-work before programming. Some characteristics of them vary, such as the series of battery packs they work with and power consumption modes. While they all support I2C communication interface, so that the majority of code design for them will be similar.

The chip from TI is BQ78350-R1A. It is an advanced battery management companion chip, designed to work with TI's BQ769x0 family of battery monitors. (TI, 2019) As this type of chip is now most commonly applied by the client company, the code design will firstly base on TI chips. After the solution for TI chips has been testified, the modification will be proceeded according to data sheet of chips from other suppliers.

The chip from Maxim is MAX17320, a high-performance fuel gauge and protector chip for Li-ion batteries, designed by Maxim Integrated. (Maxim, 2019)

The chip from Top band names SH79F829AU, which also has communication via I2C. There was no attached software for chips from Top band before.

* + 1. User Interface concept design

The functions are supposed to be realized through user interface windows. The basic functions are as following (EmergoStar, 2023):

**1. File selection**

This function allows users to select the proper document for data import by dragging it to the appointed area in the software.

**2. Programming**

This function is to import the data from PC to BMS through interface box. The data are from the previously chosen document.

**3. Change settings**

For each batch of battery production, the production date, serial number, manufacturer name need to be stored separately. This function allows users to enter information to be stored in the proper memory.

**4. Calibration**

This function is to check the present situation of the BMS. Data measured by meters will be transmitted to PC and compared with raw data. Errors will be reported through UI.

The figure below is the UI window of bqStudio from TI, the choices are complete while too complex for the clients. Some of the most-used functions will be chosen for the software design and the stability of the functions need to be improved as mentioned in chapter 1.4.1.

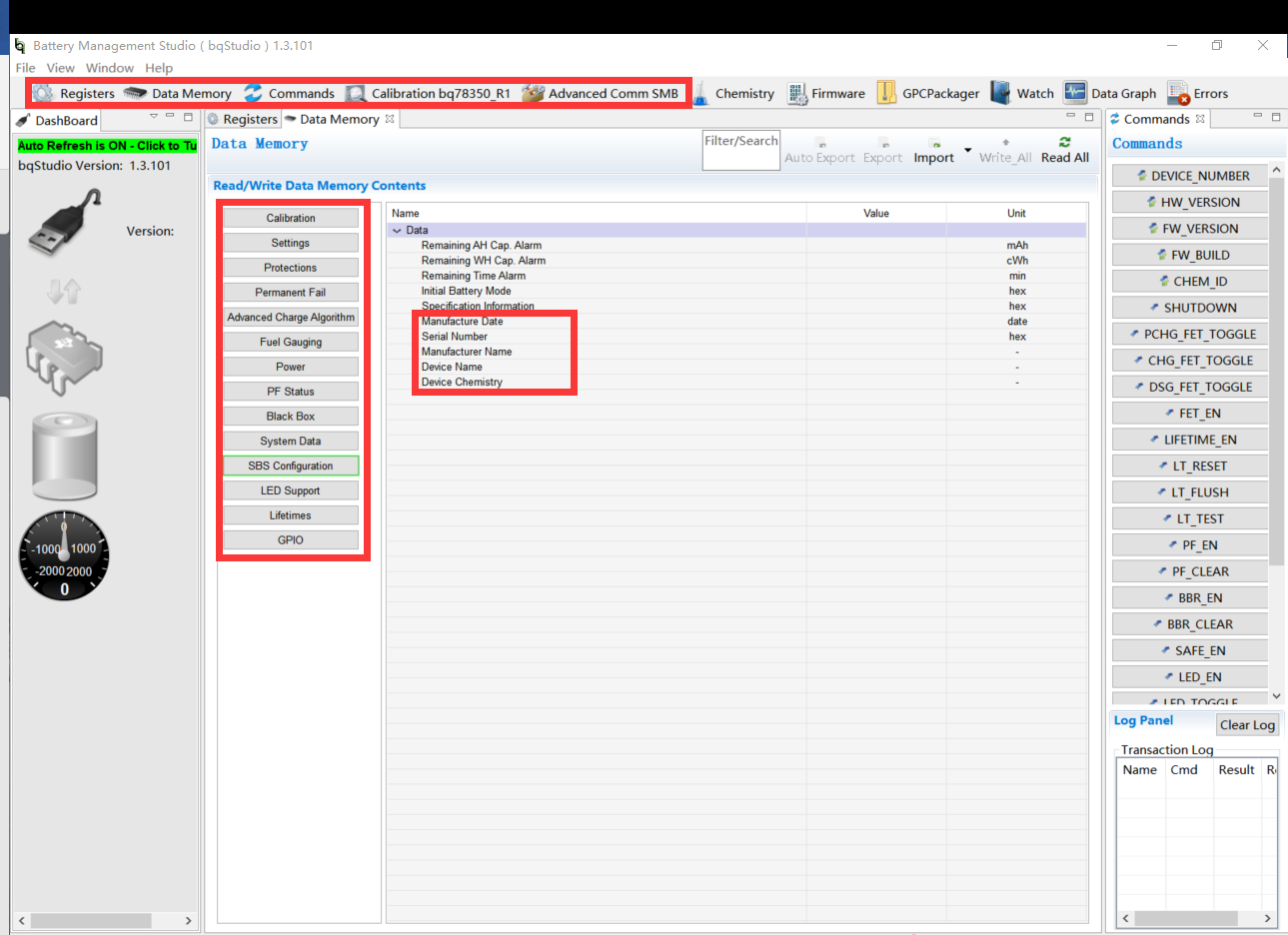


Figure A screen shot of UI window of bqStudio (own figure)

Below is the UI window of software for UART protocol, designed by engineers from the client company. It can be clearly recognized that the functions are plainly demonstrated. Also, on the bottom there is a brief description.

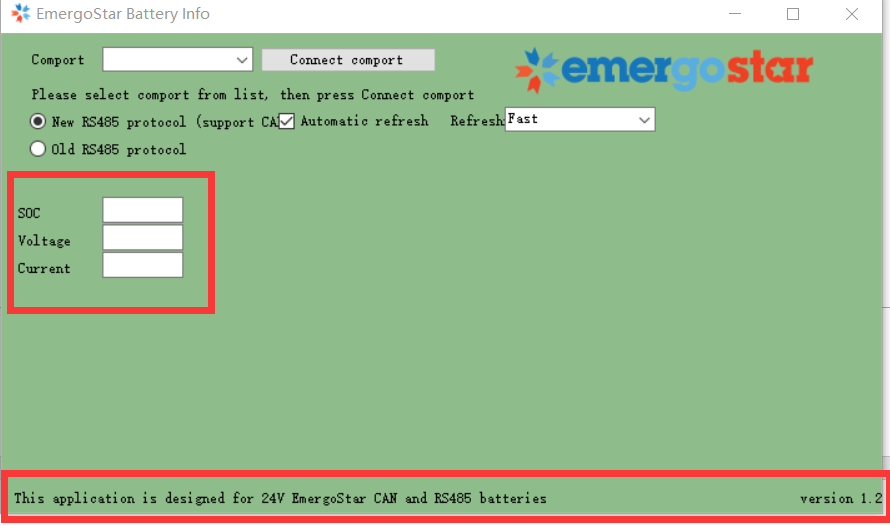


Figure UART software designed by the client company (EmergoStar, 2023)

As explained by the client company, the UI window doesn’t need to be attractive. It will not be released to the public, instead, it is only used by professionals within the company. Thus, the user-friendliness and stability are selected as critical criteria as explained before. The design of UART software can be an example to compare with the final product.

Based on the concept above, a draft of software has been designed which will demonstrate the content in a .txt document and allow users to transmit it by clicking a button. As shown in the figure below, the data in output.txt are show in the text box after it has been dragged in it.

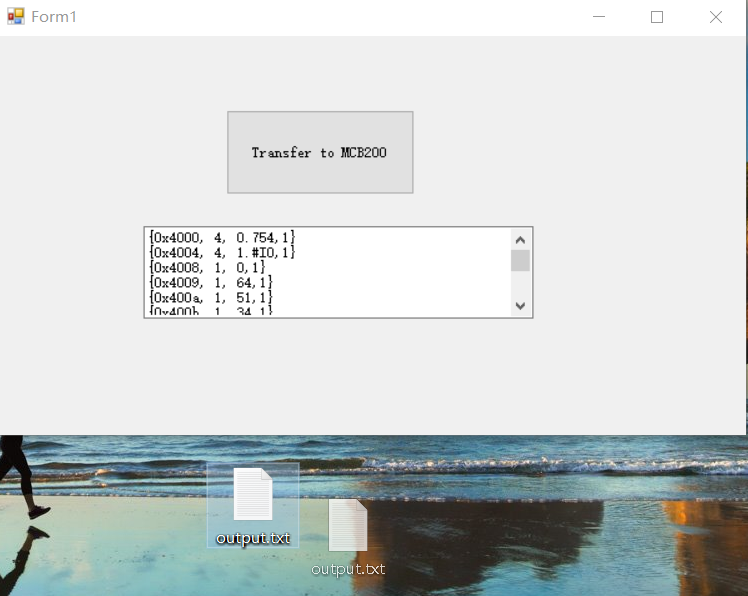


Figure A screen shot of draft design of UI (own figure)

## Hardware

There is a batch of hardware included in this project. Below is introduction of these devices.

* + 1. PIC Kit4 debugger/programmer

PIC Kit4 is a programmer and debugger device for Microchip's PIC microcontrollers. It supports a wide range of PIC MCUs, providing programming and debugging capabilities through a USB interface. (Microchip Technology, 2020)PIC Kit4 is designed for use in development, testing, and production environments, providing users with a low-cost and efficient solution for programming and debugging PIC microcontrollers. Its compact size and easy-to-use interface make it ideal for both experienced and novice users in the field of embedded systems. While this device will only be applied in design phase by the engineers (including the author). When the programming process is finished, the debugger will be removed to keep the interface box alone for a specific batch. Furthermore, the bootloader

Figure PIC Kit4 (own figure)

will play the role of this device to help with the update of the firmware at the end because the customers of the client company don’t have this device.

* + 1. MCB200 interface box

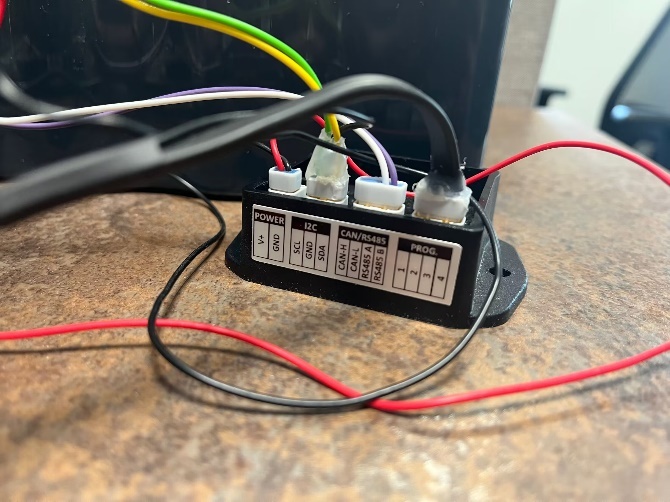
The MCB200 interface box is a device used for communication between a computer and various types of measuring instruments, including oscilloscopes, function generators, and power supplies. It provides a USB interface for connecting to a computer and multiple ports for connecting to the measuring instruments. In this scenario, the MCB200 is interfacing with the BQ78350-R1A chip, which is an advanced battery management system designed by Texas Instruments for monitoring and controlling battery packs in electric vehicles, consumer electronics, and other applications. (Texas Instruments, 2019) The MCB200 interface box supports various communication protocols and

Figure -MCB200 (own figure)

UART is applied in this project. It also provides advanced features such as signal triggering, data acquisition, and waveform analysis, which enable users to perform complex measurement tasks with high accuracy and precision.

### PIC18F27Q84

The microcontroller embedded in MCB200 is PIC18F27Q84, which means the code design should be customized based on the settings of memory and pins of this chip. (EmergoStar, 2023)

The figure on the left shows the pin settings of PIC18F27Q84. RB1 and RB2 are used for I2C protocol while RC7 and RC6 are used for the UART protocol.

Figure A screen shot of pin settings of PIC18F27Q84 (own figure)

## Software environment

A group of professional software are involved in code design and test in this project. Some of them are included in the future plan of EmergoStar while others are expected to be substituted. Basically, all employees use Windows operating system as it allows a wide range of software.

### Battery management studio

Battery management studio (bqStudio) offers a group of functions for assessment, design, testing for batteries from TI (Texas Instruments), including some real-time functions. (TI, 2019) The interface box MCB200 assisted with debugger is needed for the programming process. The company is going to use MCB200 instead in the future.

The present problem is that the operation of this software is complex, at the same time, there are unknown bugs existing which can lead to a restart of the process.

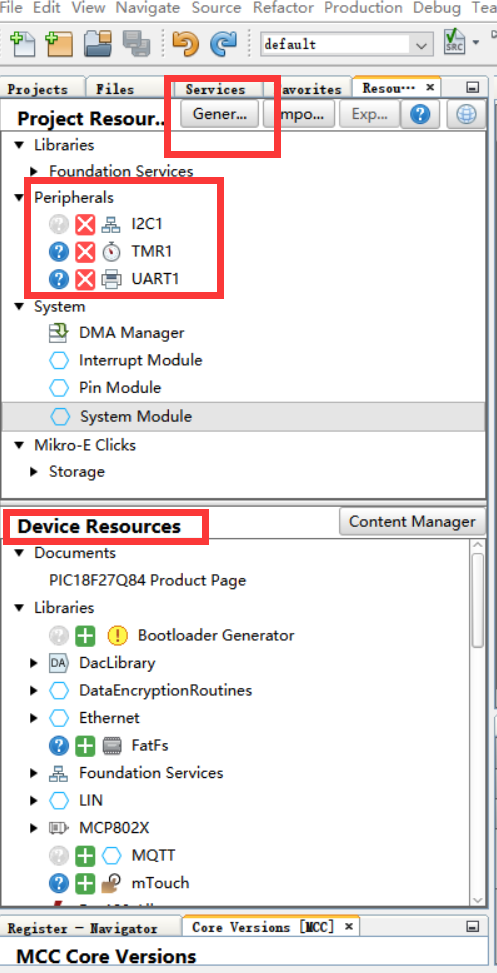
The EmergoStar is looking forward to a self-designed software which is a simplified version of Battery management studio. Critical functions should remain, while the bugs caused by non-commonly-used modules are blocked.

Another reason to substitute the Battery management studio is that it cannot store default keys from users once it is shut down. Re-entering of keys would be a waste of time.

While bqstudio will still be used to as a tool of tests for this project.

### MPLAB

The MPLAB Integrated Development Environment (IDE) is a software suite developed by Microchip Technology for developing, debugging, and simulating embedded systems using microcontrollers and digital signal controllers. (Microchip Technology, 2020) It offers a lot of default samples for basic functions. The users don’t need to finish every setting independently. Instead, the function “generator” offers a lot of solution in the header files. Visualized interface also helps with settings of pins.



MPLAB will mainly be used for the design, compilation and debugging of code. C language is chosen for this project. A reliable approach to solve unknown errors or bugs is to dig into smaller block of code.

As the client company suggests, it is wide to use the sample code as MPlab generated for the users after settings are considered.

MPlab provides code solution which make users able to not dig into the binary level of code for every section.

On the left is part of the UI window in MPlab. Protocols and instances are easy to be activated under the label “Device Resources”. “Generate” button will generate sample code in header files which can be found in “MCC generated files” under “Projects” (as shown in the figure below). Even the code cannot be used directly, it can give the code designer an overview of the components.

Figure MPlab embedded project resources (own figure)

In the “MCC Generated Files“, there are header files with declaration and definition of functions, the localization of those functions are necessary as different chips perform in different ways.

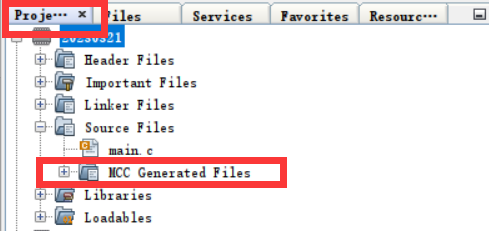


Figure MCC Generated Files (own figure)

### Visual Studio

Visual Studio (VS) is appointed by the client as the IDE of software design, which *is an integrated development environment (IDE) developed by Microsoft.* VS supports multiple programming languages and can be used to design, develop, test, and deploy various types of applications, including desktop, web, and mobile applications. *(Anandmeg. ,n.d.)* C# will be the coding language, also appointed by the client. The draft design of UI window is already designed from Visual Studio. Also, Visual Studio is not suitable for execution of script because user can only establish “project” with a lot of properties, rather than a single file. It is specialized for design of the entirety of code.

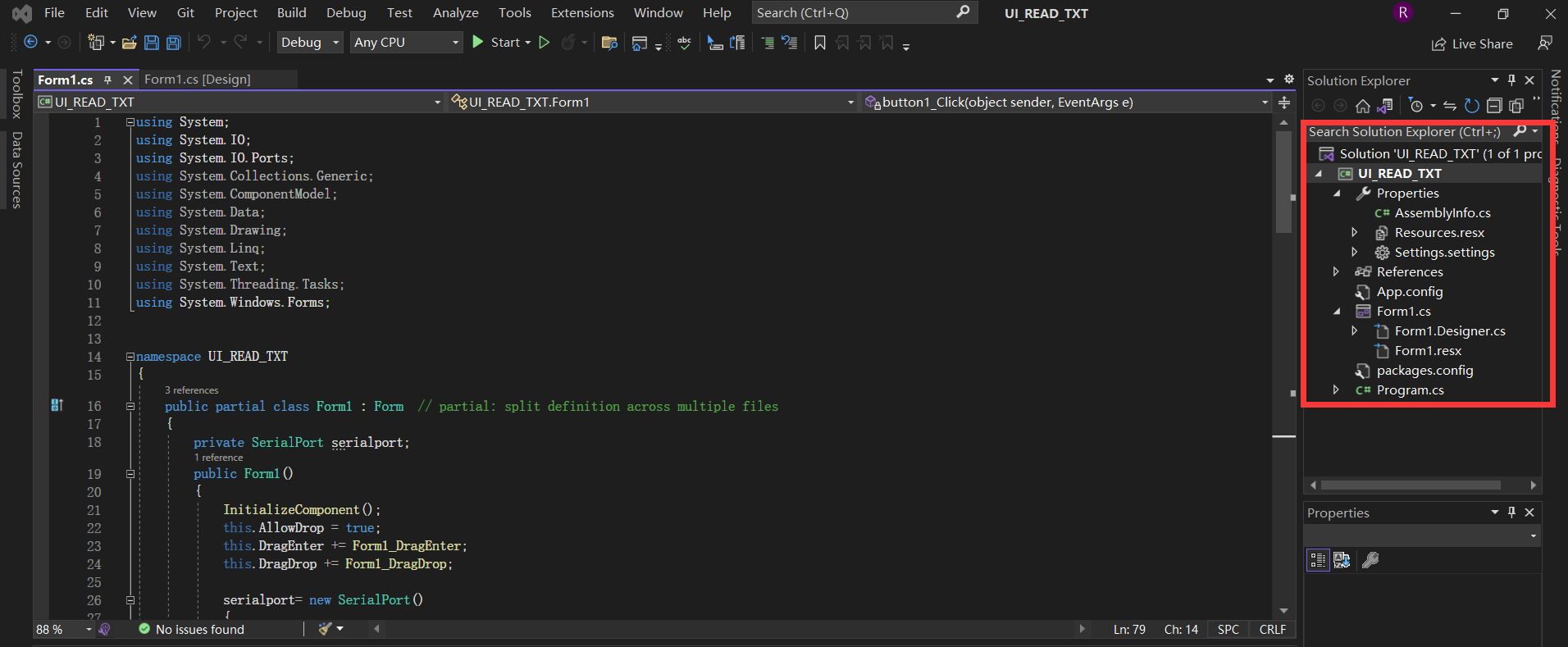


Figure A Project in Visual Studio with properties in the red block (own figure)

## Method for mass production of batteries

The code design for mass production is different from those for production of smaller scales. More user-friendly functions such as automatic storage of default value should be considered.

Presently in the bqStudio, the programming is realized through block read or write command 0x44. This command can also be recognized by the chip through signals by I2C. (Williams, 2015)

Below the figure shows how data are ordered inside the UI window of bqStudio.

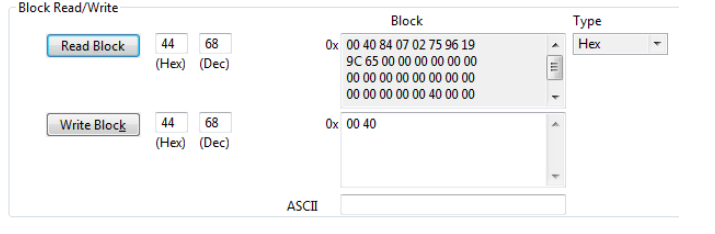


Figure Reading command in bqStudio (TI, 2015)

To load the data memory in each pack with the desired configuration pattern for the project.

A simple loop is enough for reading data memory from batteries because the data always start at 0x4000. The end of data can be determined by users through hex dump figure. Some examples will be provided in the coming chapters.

## Data sheet

The main difference among different chips is the data sheet. (EmergoStar, 2023) Below is the data sheet provided by company TI in 2019. Even its’ from an official document, the data are outdated which is an obstacle of data programming. Calibrated data sheet with address for TI, Maxim and Top band are expected to be delivered in the end.

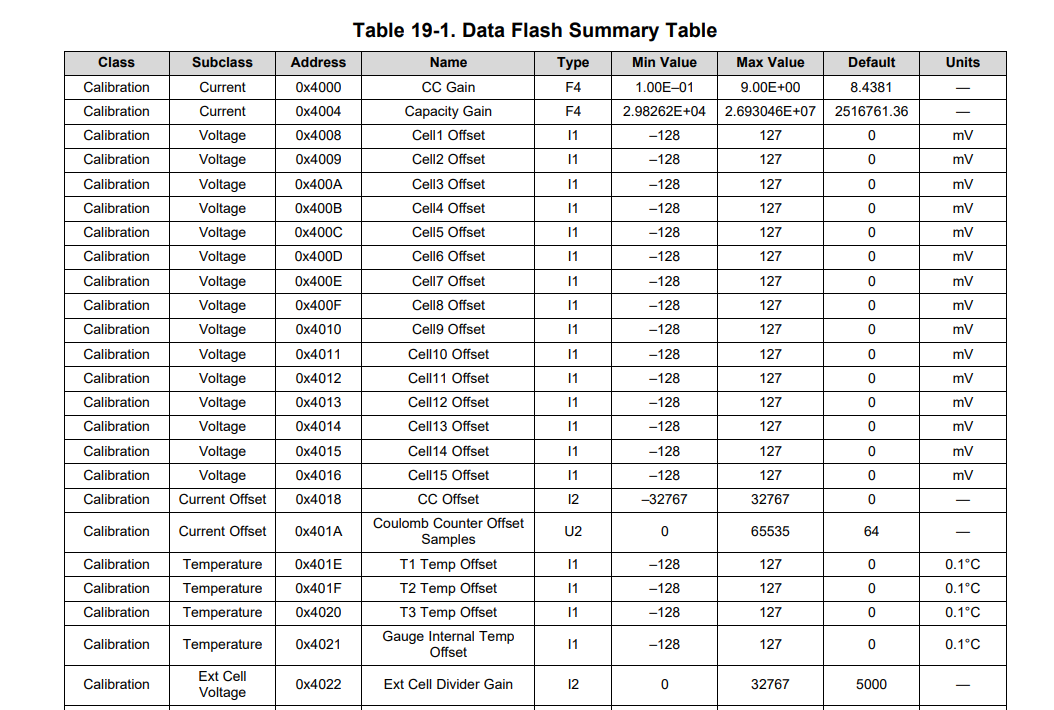


Figure Part of “outdated” data sheet from TI (TI, 2019)

Here is an example of data sheet in Excel. It was not exported from the software because when opening the .csv file by excel from the device, there can be loss of data. A proper way of data storage should also be determined and tested for the final product. **The criteria here is to be able to store the address, data length and value precisely with no loss.**

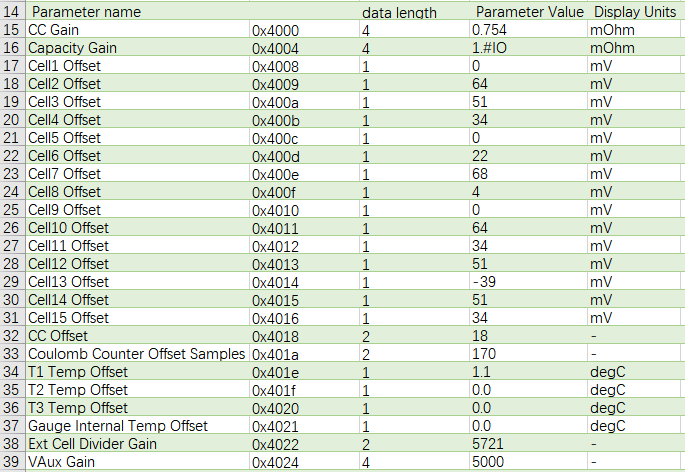


Figure Example of data sheet in Excel (own figure)

## S-rec document

The full name is Motorola S-record. It is a kind of document format that can store binary information in ASCII text format for later purpose such as firmware upload. (Canonical Ltd., 2019) It can be one method to store data from the BMS.

S-Record files consist of a series of records, each of which contains an address, a record type, and the data associated with that record. The record type specifies the type of record being used, such as data record, address record, or end-of-file record.

The S-Record format is designed to be platform-independent, meaning that S-Record files can be generated and read by devices and systems of different architectures and operating systems. In this project, an S-rec document is selected, detected and then transmitted through the software designed in Visual Studio.

To analyze S-rec documents, “hex dump” is applied which will be explained in the coming chapter.

## Hex dump

Hex dump is a hexadecimal view of computer data. It is used for designer to have a clear sight into the sequences of data. In hex dump, every byte is represented by 2 hexadecimal numbers and data are ranged in columns separated by space. (Stevens, R., Rago, S., & Fenner, B. ,2013)

When designers are reading the S-rec documents, hex dump gathers those bytes with relative information together in columns. Similar process is applied in code design to read specific bytes.

Notepad++ is recommended for analysis for data stored in S-rec document. When using Windows Notepad to open S-rec document, even lines with same digits can have different length shown on the screen. This can cause confusion and inconvenience.

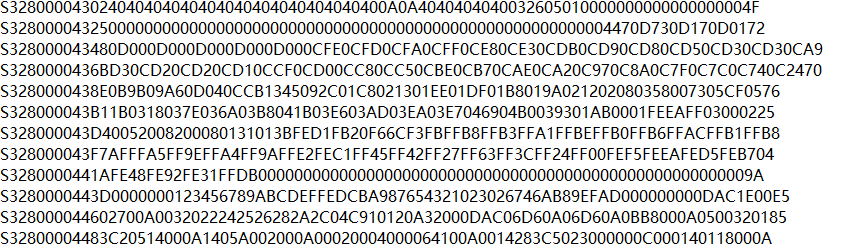


Figure -S-rec opened in Windows Notepad (own figure)

Below the data in an S-rec document has been automatically marked by Notepad++. The data type, length of data, address and crc code are highlighted in different colors. This helps engineers to dig into the data.

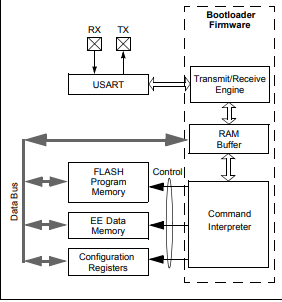


Figure S-rec opened in Notepad++ (own figure)

## Bootloader

The purpose of using Bootloader is to download and execute applications on devices after they have been in work duty. It shares the available memory with apps. (Martin, T. ,2016)A bootloader is needed by the customers of the client company to program the BMS without the help of PIC Kit4.

The term “Bootloader” can actually be divided into two parts: “Boot” and “Loader”. Boot helps with initializing the environment for C language (or other high-level language) development. After the device is powered, a series of settings will be established.



The purpose of Loader is to execute the logic of apps. E.g., the programming of a hardware need the initialization of pins. When a specific board want to execute loader, the boot should be finished at first. (Fosier, 2002)

In the context of this project, the bootloader will mainly focus on settings for communication on CAN bus. Presently, there is only example code for UART bootloader on PIC18F series chips. On the left there is an example.

Figure Bootloader functional diagram (UART)

In this project, the bootloader is the code embedded in the flash memory which will be firstly executed when the chip is activated. It arranges the use of memory to help with the updating of firmware. (Richey, 2002) As Richey explained in booklet “A FLASH Bootloader for PIC16 and PIC18 Devices”, there is only one example for code of bootloader design. It can be considered as the basic structure as the client company explained.

## Basement of code design

EmergoStar has already offered a document with code for sealing and unsealing of batteries, attached with a software as example The file is now regarded as the “Golden file”, which means it contains the basic settings for this project. Sources from the company will be as attachment for this project.

Among all the information provided in the file, the setting of pins on PIC18F27Q84 embedded in MCB200 have the most significant impact on code design. The figure below is the table for pin settings which act as the reference for data transmission.

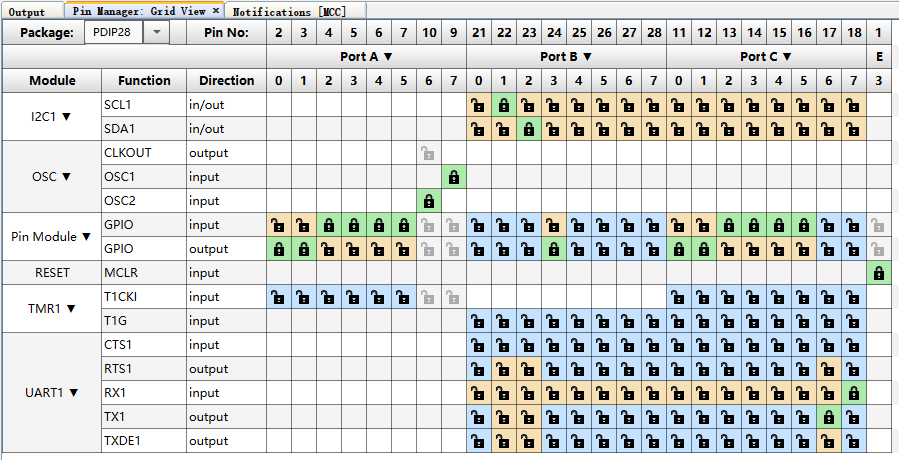


Figure Pin settings for PIC18F27Q84 (own figure)

## Protocols

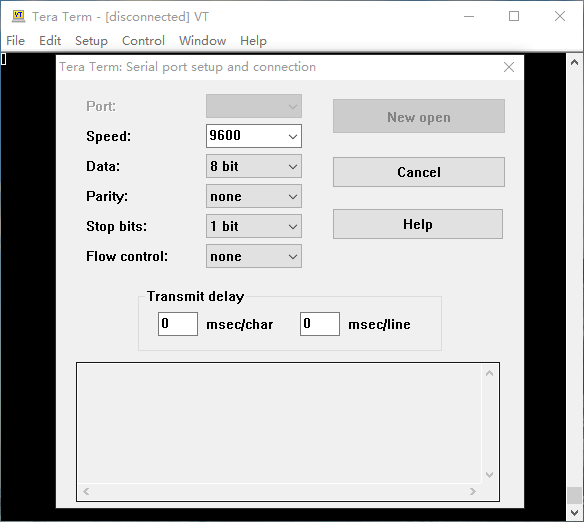
Data are transmitted through specific protocols. To activate these protocols, MCC manager in MPlab offers “Content Manager” to easily add them into the project.

### UART

UART stands for Universal Asynchronous Receiver/Transmitter, and it's a type of serial communication interface that's commonly used in microcontroller-based systems. (Dawoud, D. S., & Dawoud, P., 2020) The UART protocol is designed to allow two devices to communicate with each other, with one device acting as the transmitter and the other device acting as the receiver. In this project, UART is selected for the communication between the PC and MCB200 to make the data visible for PC software users, it’s a common use of it. (Dawoud, D. S., & Dawoud, P., 2020)

As mentioned before, UART1 will be initialized at the beginning of the programming process as a specific instance. It is often used as the primary serial communication interface in a microcontroller-based system, and it typically has a set of registers that control its behavior.

#### Tera Term

To receive data from serial port, Tera Term is used so that the information printed by function printf() in C language can be visible. It is an open-source software Tera Term is the terminal emulator for Microsoft Windows, that supports serial port, telnet and SSH connections. (Tera Term, 2007)

The settings of the serial port should be identical for the transmitter and the receiver. *(Microchip Technology, Inc., 2021)* The default settings of the serial port are shown on the left. During the test, the baud rate (Speed) may be modified for a shorter loading time.

Figure UI window of Tera Term (own figure)

### I2C

I2C (Inter-Integrated Circuit) is a serial communication protocol for connecting microcontrollers, sensors, and other integrated circuits. (Muller, W. ,2017) I2C uses two bidirectional lines, a clock line and a data line, for communication and supports multiple devices on the same bus. The figure below shows a general structure of data on I2C. The data are expected to be transmitted in format that is “address+data+crc code” which should be considered in code design.

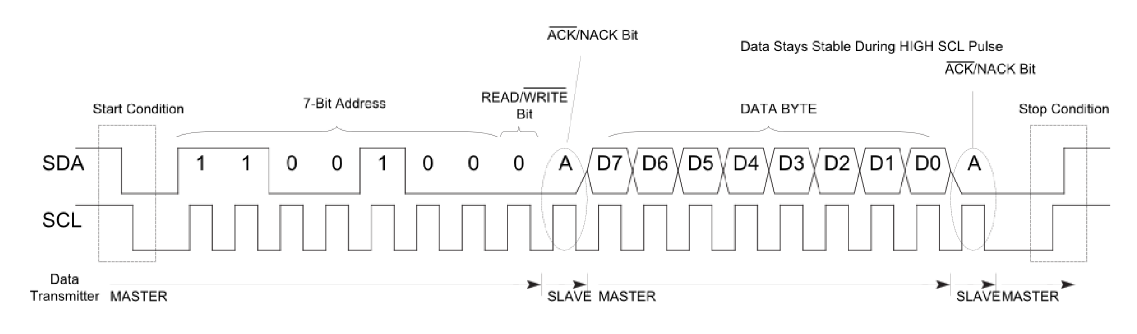


Figure Structure of data transmission *(Muller, W. ,2017)*

As mentioned before, for PIC18F27Q84, pin RB2 will be the data line and pin RB1 will be the clock line because of the characteristics itself.

#### I2C Signal detection

To detect data transmitted on I2C, the software Pulse View is applied with the device Logic Analyzer. The Logic Analyzer captures and records the signals, while PulseView provides a user-friendly interface to visualize and analyze the captured data, (SparkFun Electronics., n.d.) enabling users to debug and troubleshoot digital circuits and communication protocols quickly and efficiently.Together they make the analysis of data transmission more visible as the figure below shows. Thepink “A” stands for “Acknowledge” which means the data transmission

Figure Logic Analyzer (own figure)

is successful. When there is no pink “A” after a byte of data, the process needs to be checked with value of data and connection of hardware.

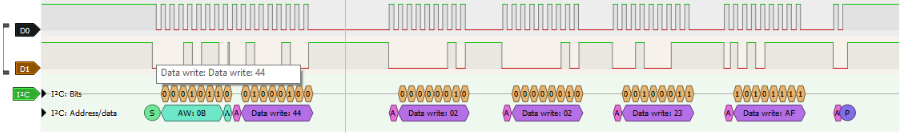
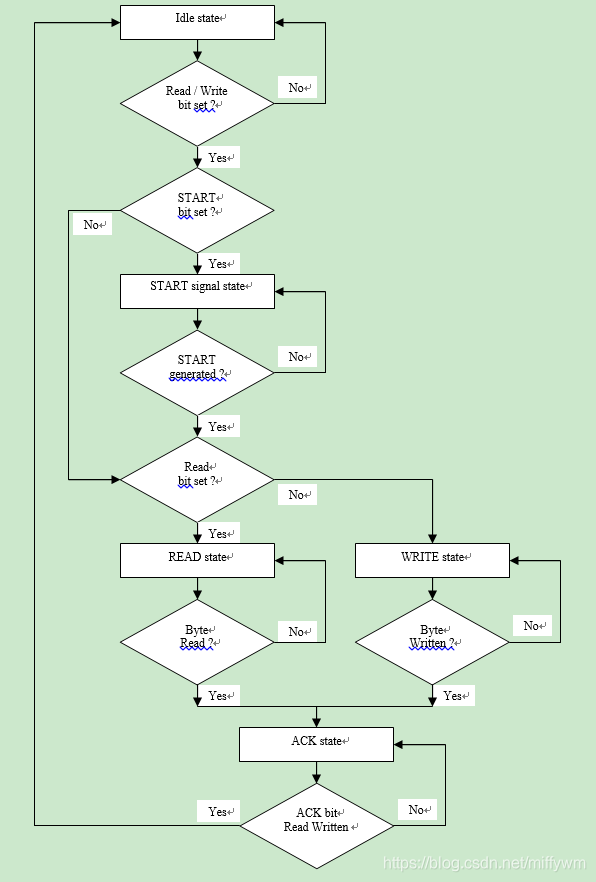


Figure I2C signal in Pulse View (own figure)

#### Data Transmission process

As the figure below shows, the I2C protocol can be illustrated by the form of a flow chart. The code to realize each section is the component of this research project. While they won’t be singularly tested because of the time limit. Also, a solution for a small function may not be suitable for the sub-system (code for one brand of chip). That’s why component test plan will not be involved in this project.

I2C uses a master-slave architecture, where one device (the master) initiates and controls the communication with one or more slave devices. In this context, the MCB200 interface box acts as the master, and the BQ78350-R1A chip serves as the slave device.

Once the I2C module is properly configured, the C program should define functions to perform various I2C operations, such as starting a communication session, sending the BQ78350-R1A chip's I2C address, writing data to the chip, and ending the communication session. Error handling mechanisms should also be included to manage any issues that may arise during the I2C communication process.

Figure - The flow chart of I2C (CSDN, 2017)

#### Code design for I2C

In this process, we are designing a program in C language using the MPLAB software to enable the MCB200 interface box to write data to the BQ78350-R1A chip manufactured by Texas Instruments (TI) via the I2C protocol.

Figure 25(below) code for Unseal section on I2C (EmergoStar, 2022)

To develop the C program, first, the necessary header files and libraries for the MPLAB IDE, the MCB200 interface box, and the I2C communication protocol need to be included. These header files and libraries contain pre-written code and function declarations that make it easier to implement the required functionality.

Next, the MCB200's I2C module needs to be initialized and configured according to the specifications of the BQ78350-R1A chip. This may involve setting the I2C clock frequency, selecting the appropriate I2C mode (standard, fast, or high-speed), and configuring the I2C pins.

In summary, the project involves developing a C program using the MPLAB IDE to enable the MCB200 interface box to write data to the microchips via the I2C protocol. This is achieved by configuring the MCB200's I2C module, implementing functions for I2C operations, and handling potential errors during communication.

The principles of programming for three brands of chips are similar, and the code above for unseal of I2C has been tested on three brands of chips.

To make it more readable, the code above is now expressed in the figure. Code will be added to the appendix in plain text after the integration phase.

# Methodology

Two typical methods are introduced in this chapter.

## Choice of methods

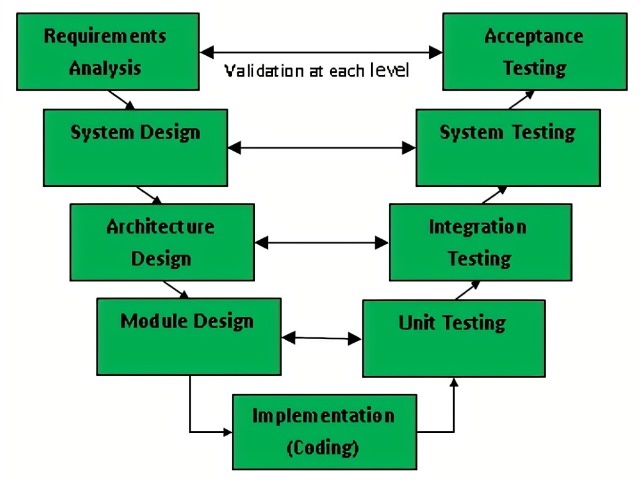
A choice between them is delivered according to the feature and requirement of this project.

### V-model

The V-model is a software development model that emphasizes the importance of testing and verification throughout the development process. It is called the V-model because the process is represented by a V-shaped diagram, with testing and verification starting at the top of the V and progressing down to the bottom. (Kwekkeboom, G. K. ,2019)

While in the real situation, the two phase of V-model may not be symmetrical. The integration phase may actually cost more time to be completed.

The basic process of V-model consists of “Breaking down the system” and “Building up the system” as the following figure shows.



Fige 3- V-model overview (Marjerison,2013)

### Oskam Method

Oskam method distinguish for its innovative performance in design. It is useful when designers are trying to establish a creative thinking.(Kendall, 2006)

For those products which focusing on abstract user experience, exp. For which need a qualitative survey to alert, Oskam method is suitable.



Figure - Life cycle of Oskam Method *(Rich, n.d.)*

### Choice between them

Between the two classic methods, V-model is selected for this project. The reasons are as follows:

1. V-model is specialized for tests. The core of this project is to design code with reliable performance. The V-model is specialized in the field of test plan design to ensure the quality of products.
2. The requirement of this project has been clearly decided by the client, which means the process can be broken down step by step.
3. This project focus on code design for battery production. A software for PC and a set of program for MCB200 will be delivered.
4. This project deals with large volumes of data that need to be processed accurately and efficiently. The V-model emphasizes testing at every stage of the development process, ensuring that the quality of the data and the software are validated before moving on to the next stage.

To prove that the final product is qualified to be used by professional engineers in the company, a refined test plan is necessary. The test plan should include different phases for all required functions under various situation.

The instruction book “The Engineer's Guide To The V-model (4th edition)” will be the main reference of this chapter.

In this project, the V-model has two layers. On the level of whole system, the software and program together work for the programming of microchips. On the sub-system levels, the system is divided according to different brands of chips.

Here comes a customized “V-model” that truly describe the proportion of two parts of V-model.

**System Integration**

System Test Plan

System Test Plan

Sub-System Test Plan

**Sub-System Integration**

Sub-System Test Plan

**Coding**

Figure -Customized figure for V-model (own figure)

To guarantee functionality of the design, system descriptions on all levels of design will be delivered for preparation of Integration phase. The integration edge of V-model determines the final product. Coding at the bottom is the bridge between tests and products.

During the test phase of this project. The demand and comments of designers will be the priority. The table below shows the details of deliverables. The test plan will not be delivered on levels below sub-systems.

|  |  |  |
| --- | --- | --- |
| Phase | Deliverables | Actions |
| System Design | List of demands | Have the meeting with clients, summarize requirements and mention potential risks |
| Function overview | Draw the function tree for the software on PC and program on MCB200. |
| System description | Find out the input and output of the programming process based on demands. |
| Analysis of current situation | Describe the current situation and describe the ideal solution with measurable standards of proof. |
| System test plan | Write the test plan for the general test of the system for collaboration between the software and the program according to the template derived from V-model. |
| System division | Divide the whole system into sub-systems according to the supplier |
| Subsystem design | Formula overview | Describe the format of data stored and transferred in this data in a figure and provide the address sheet. |
| Subsystem description | Write down the description of each subsystem and draw the relationship between each subsystem. |
| Subsystem test plan | Create test plans of every sub-system for judging each function in the program can be worked well. |
| List of components | A table which describes the input/output and function of functions |
| Subsystem Integration | Subsystem assembly plan | Write down the flowchart of how to connect each function to other functions. |
| Subsystem assembly | Combine the functions into an entirety which can deal with microchip from a specific supplier. |
| Subsystem test | All sub-systems should be tested individually by following the test plan. |
| System Integration | System assembly plan | Detail the connections between and positioning of each subassembly for reference during the assembly process. |
| System assembly | Assemble the system into an entirety which can deal with the main research question. |
| System test | Test the performance mainly based on three criteria with the participation of clients. |

Table  *Deliverables, (Kwekkeboom, 2019)*

# Schedule and preconditions

This project starts from Feb.6th and is expected to end in the end of June (about Jun.24th).

Below is the schedule designed for results of every phase. For the ability relevant with the results, they will be discussed in the portfolio.

|  |  |  |
| --- | --- | --- |
| **Orientation Phase: about 6 weeks** | | |
| *Content* | *Start date* | *Duration* |
| Background description | Feb.6th | 1 week |
| Problem analysis | Feb.6th | 1 week |
| Meeting with client | Feb.6th | 1 day |
| Theoretical framework | Feb.6th | 6 weeks |
| Desk research | Feb.6th | 3 weeks |
| Research question definition | Feb.13th | 1 week |
| Meeting with in-school supervisor | Mar.9th | 2 hours |
| Partial Portfolio | Mar.10th | 2 days |
| Self-evaluation | Mar.20th | 1 day |
| **Execution Phase: 8 weeks** | | |
| *Content* | *Start date* | *Duration* |
| System design | Mar.20th | 7 days |
| Sub-system design | Mar.27th | 7 days |
| Sub-system test plan design | Mar.27th | 2 weeks |
| System test plan design | Apr.5th | 3 days |
| Sub-system tests | Apr.5th | 3 days |
| Sub-system integration | Apr.8th | 6 weeks |
| System tests | Apr.15th | 1 week |
| System integration | Apr.17th | 6 weeks |
| Self-evaluation | May.10th | 1 day |
| **Completion Phase: 4 weeks** | | |
| *Content* | *Start date* | *Duration* |
| Results | May.23th | 1 week |
| Discussion, reflection and recommendation | May.30th | 1 week |
| Portfolio | Jun.1st | 2 weeks |
| Conclusion | May.29th | 1 week |
| Collection and verification of deliverables | Jun.15th | 10 days |
| Package and upload deliverables | Jun.22nd | 1 day |

Table Timetable

# Reference

Kwekkeboom, G. K. (2019). The Engineer's Guide To The V-model (4th edition).

*Williams, D. W. (2015). Bq78350 Data Memory Programming for Mass Production.*

*MicrochipWebSminars.(2012).*[*https://www.microchip.com/stellent/groups/SiteComm\_sg/documents/DeviceDoc/en558478.pdf*](https://www.microchip.com/stellent/groups/SiteComm_sg/documents/DeviceDoc/en558478.pdf)

*Marjerison, B. (2013). The Software Bug Book: Thoughts on Software Quality. Createspace Independent Pub.*

*Tanenbaum, A. S., & Wetherall, D. (2011). Computer networks. Pearson Education.*

*Muller, W. (2017). I2C bus specification and user manual (version 6.0). NXP Semiconductors.*

*Texas Instruments. (2019).*BQ78350-R1A Technical Reference*.*

*Microchip Technology.(2020)* *MPLAB® PICkit™ 4 In-Circuit Debugger User's Guide*

Kendall, K. E., & Kendall, J. E. (2006). Systems analysis and design. Prentice Hall.

*Rader, T.R. (2013). Methods for documenting systematic review searches: a discussion of common issues*

*Martin, T. (2016). The Designer*’s Guide to t*he Cortex-M Processor Family. Newnes.*

*Anandmeg. (n.d.). Introduction to Visual Studio - Training. Microsoft L*earn. <https://learn.microsoft.com/en-us/training/modules/visual-studio-intro/>

*Canonical Ltd. (2019). Ubuntu Manpage:        srec - Motorola S-record record and file format. https://manpages.ubuntu.com/manpages/bionic/man5/srec.5.html*

*Stevens, R., Rago, S., & Fenner, B. (2013). UNIX Network Programming, Volume 1: The Sockets Networking API (3rd ed.). Addison-Wesley Professional.*

*Dawoud, D. S., & Dawoud, P. (2020). Serial Communication Protocols and Standards: Rs232/485, Uart/Usart, Spi, Usb, Insteon, Wi-Fi and Wimax.*

*Microchip Technology, Inc. (2021). 8-Bit USART Serial Communication - Developer Help. (n.d.).*[*https://microchipdeveloper.com/projects:mcu1101-project-12*](https://microchipdeveloper.com/projects:mcu1101-project-12)

*Tera Term developer Team. (2007) Terminal Emulator for Windows.* [*http://www.teraterm.org/*](http://www.teraterm.org/)

*SparkFun Electronics. (n.d.) Using the USB Logic Analyzer with sigrok PulseView.*

*https://learn.sparkfun.com/tutorials/using-the-usb-logic-analyzer-with-sigrok-pulseview/all*

*Fosier, M.F.,& Richey, R.R.(2002) A Flash Bootloader for PIC16 and PIC18 Devices*

# Appendix

## Test Plan for software and program

The test plan for this project will do performance testing on both the system and subsystems, according to the structure of the V-model. This is to ensure that they meet the customer's requirements. The V-model is a structured approach that involves testing at each stage of development which has been mentioned before.

Based on V-model, the following aspects will be elaborated specifically for the system and each sub-system:

1.Aim & Hypothesis

2.Variables

3.Tools

4.Methods

5.Project boundaries and conditions

6.Involved people

7.Expected results