Cairo University
Faculty of Graduate Studies for Statistical Research
Department of Computer Science

The Smart Battery Monitor

A Project Presented for Fulfillment
For Diploma Project in Computer Science

Submitted by

Mohamed Sayed Hemed El-Sayed
Sherif Mostafa Samy Ahmed

Supervised by

Dr. Ahmed Hamza

Cairo, Egypt
July 2021

- **Abstract:**

In recent decades, smart devices have played the most significant role in human life, and power consumption has been one of the most trending issues. As a result, batteries and their lifespan have been one of the most sophisticated topics.

All people nowadays want to use smart devices for the longest time possible without recharge their batteries, that's because smart devices became one of the most important tools for all people regardless of their work positions or their business.

One of the solutions for dealing with this problem is trying to maintain battery lifespan as much as possible, so in this report, we introduce a smart solution to charge the battery with the standards that the major global companies advise, we support this hypothesis with many articles that specialist companies have been published.

This report discusses the design, implementation, and testing of a device that automates the charging process by protecting the human from the headache of monitoring the percentage of charging periodically.

- **<u>Declaration:</u>**

We here declare that the project entitled “The Smart Battery Charger” submitted for all training centers, and educational institutional, which serve the student affairs in smart and automated way.

No.	Name	Task	Signature
1	Mohamed Sayed Hemed	Hardware Implementation (Design Board & write low level codes)	
2	Sherif Mostafa Samy	Software Implementation (Design interfaces & coding)	

- **<u>Acknowledgments:</u>**

At the beginning and at the end we all thank Allah for helping us to achieve this work and ask Allah to benefit us with what he taught us and teach us what will benefit us.

On the behalf of the Faculty of Statistical Studies and Research, Cairo University, and on our own behalf, we would like to express our sincere gratitude to all those respectable Professors in capacity of Dr. Ahmed Hamza who guided us through the preparation of this project and for his continuous support, for his patience and motivation.

Our warm greetings for the computer science department management from the professors, doctors, teaching assistants and employees.

Finally, we thank our families for supporting us and helped us to reach so far and to have the tolerance to face difficulties and keep up.

- **Table of Contents**

- **Table of Figures**

- **Table of Tables**

- **Definitions, Acronyms, and abbreviations (Glossary)**

UML	Unified Modeling Language
UAT	User Application Test
MCU	Microcontroller
IDE	Integrated Development Environment
HW	Hardware
PC	Personal Computer
UI	User Interface
CPU	Central Processing Unit
GPU	Graphics processing unit
RAM	Random Access Memory
HD	Hard Disk
SQL	Structured Query Language
Li-ion	Lithium-Ion
IO	Input Output
FSM	Finite State Machine
IC	Integrated Circuit

Chapter 1: Introduction

1.1 INTRODUCTION:

In recent decades, smart devices have played the most significant role in human life, and power consumption has been one of the most trending issues. As a result, batteries and their lifespan have been one of the most sophisticated topics.

All people suffer from smart devices batteries that must be replaced periodically, and these battery's performance decreases almost daily.

This report discusses one of the smart solutions that depend on using the global standards of charging batteries taking into consideration the advice of specialist companies in this field like Apple and Samsung.

The overall goal of the project is to design, implement and produce (as a sale product) a tiny hardware device (tiny to be portable for any user) that can enable or disable charger functionality in real life.

We start to design a software that co-operate with this hardware in windows platform, designing of software is built to be able to develop to be multiplatform in future and able to contact with smartphones or tablets regardless their brands.

We design the software (that connects with the hardware device) to be compatible with Windows platforms.

Designing the software takes the concept of functional programming, parallel programming, and object-oriented programming in (C#9.0 & dotnet 5.0) to be much more flexible to has the ability to develop to be multiplatform in the future and able to contact with smartphones or tablets regardless of their brands.

1.2 Motivation:

The motivation for designing this application came from some reasons:
Firstly: As students in the computer science department, the laptop is one of the most important tools for us, so we want to save its battery’s life as much as we can.

This is the first serious contribution to overcome this issue by building an intelligent device from scratch that saves batterie’s life, which based on Embedded Systems by combining software with hardware.

On other hand: for each student in any faculty or institution, a laptop can be one of the most precious especially for those who suffer from poverty.

Replace the battery with a little bit of money for these students can literally kill their studies progress, so we are seeking for saving money for these students to save their brains.

1.3 Problem Definition:

Batteries do not have an infinite lifespan. Most battery manufacturers claim that their products have a 300-500 cycle rating [14]

After this, batteries would be unable to carry as much energy and will only be able to fuel the computer for limited periods of time [14]

The wrong way of charging Lithium-Ion batteries which included in laptops leads to a decrease in batterie's lifespan by 25% which in turn lead to replacing them frequently, as well as frequently and randomly power outages which lost user's work.

Leaving the battery connected to the charger when the battery is completely charged while you are using it may lower battery lifespan if you do it repeatedly [15]
"So, a good range to aim for when charging a Li-ion battery is from about 40% to 80% in one go. Try not to let the battery drop below 20%" [14]

As battery's cost has risen, it is necessary to save money and enhancing batteries' performance, so solving this problem now not later is the correct choice.

1.4 Background:

Three years of Experience in:

- Embedded Systems field by teaching Embedded systems and leading robotics teams in local and international competitions.

- Software engineering and do different useful applications like:

Project title	Language	Link
Interactive_Towers_of_Hanoi	C++	[Link]
Lexical_Analyzer	C++	[Link]
Mail_Manager	VB.NET	[Link]
WhatsApp_Automation	C#	[Link]
Silver badge in HackerRank platform Problem Solving section using C++	C++	[Link]
Easy_Query	VB.NET	[Link]
Smart-Servant-using-AVR-uC	Embedded-C	[Link]
Racing Team - Remotely Operated Vehicle 2018	Arduino-C	[Link]
Open-source Embedded Systems Foundation mini-diploma	Embedded-C	[Link]

Table 1.1: Previous projects

1.5 Related works:

There is a lot of software applications that give the user some statistics and information about the battery, but we did not find any application that monitor the charger by the hardware yet.

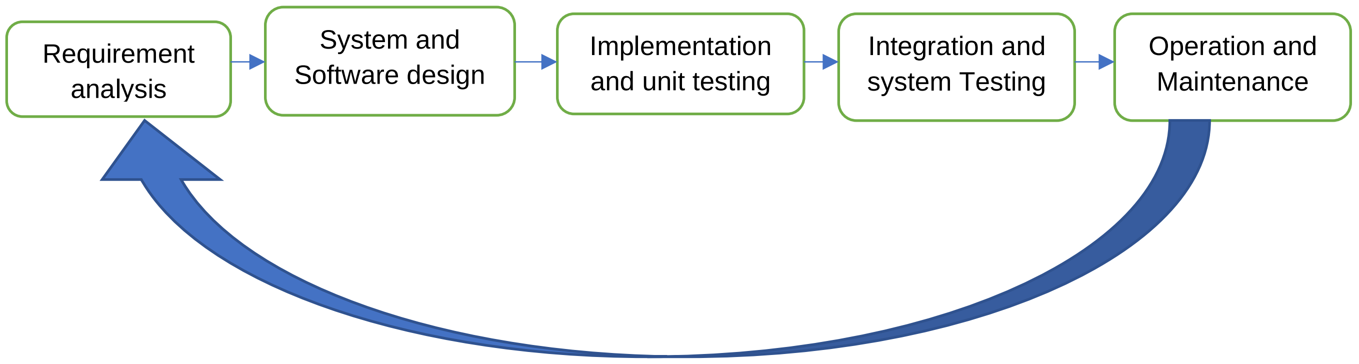
1.6 Scope:

This system allows users to monitor their smart devices' batteries from the face of performance, user can furthermore expected time that this battery MUST be replaced without surprises in some critical times like exams or during emergency works, this became available thanks to corporation between the system and database that generates detailed and aggregated reports explains batteries performance (decrease and increase) in the charge for each 1% and its relationship with machine recourses usages such as processor, ram and hard disk.

1.7 The system development life cycle is the Plan-driven methodology:

- How Plan-driven model works:

  “Plan-driven software development is a more formal specific approach to creating an application, Plan-driven methodologies all incorporate repeatability and predictability, a defined incremental process, extensive documentation, up-front system architecture, detailed plans, process monitoring, controlling and education, risk management, verification, and validation.” [3, 12]


Requirement analysis
System and Software design
Implementation and unit testing
Integration and system Testing
Operation and Maintenance

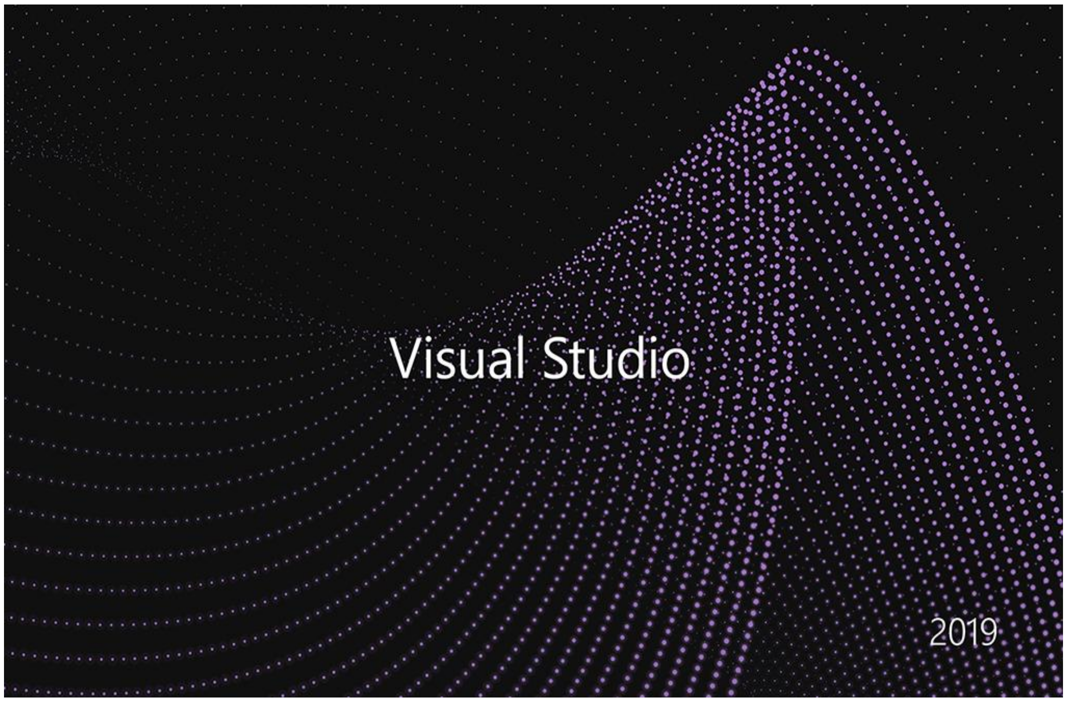
- There are separate identified phases in the waterfall model:
    - **Requirement analysis:** development team meets project stakeholders to identify needs and establish requirements to satisfy project goals. Requirement analysis initially is a feasibility study that identifies functional and non-functional requirements with its constraints. Documentation is generated [3, 12]
    - **System and Software design:** development team converts requirements into representation of architectural model. Prototypes are created with functional algorithms and data structures. Creating accurate design in this stage is vital because after completing design stage, development team does not go back to the design previous stage. Design is finalized and overall, of the project outcome depends on it [3, 12]
    - **Implementation and unit testing:** design is transformed into software [3, 12]
    - **Integration and system Testing:** software domain is tested for bugs. Different sets of tools are available for testers and developers to detect and fix defects in the software [3, 12]
    - **Operation and Maintenance:** stage where product is updated (patched), to meet changing needs and environments. Fixing undetected bugs during testing stage and defects that might arise during updates [3, 12]

Advantages	Disadvantages
The waterfall model is mostly used for sophisticated system engineering projects.	The main drawback of the waterfall model is the difficulty of accommodating change after the process is underway. In principle, a phase must be complete before moving onto the next phase.
	Inflexible partitioning of the project into distinct stages makes it difficult to respond to changing customer requirements.

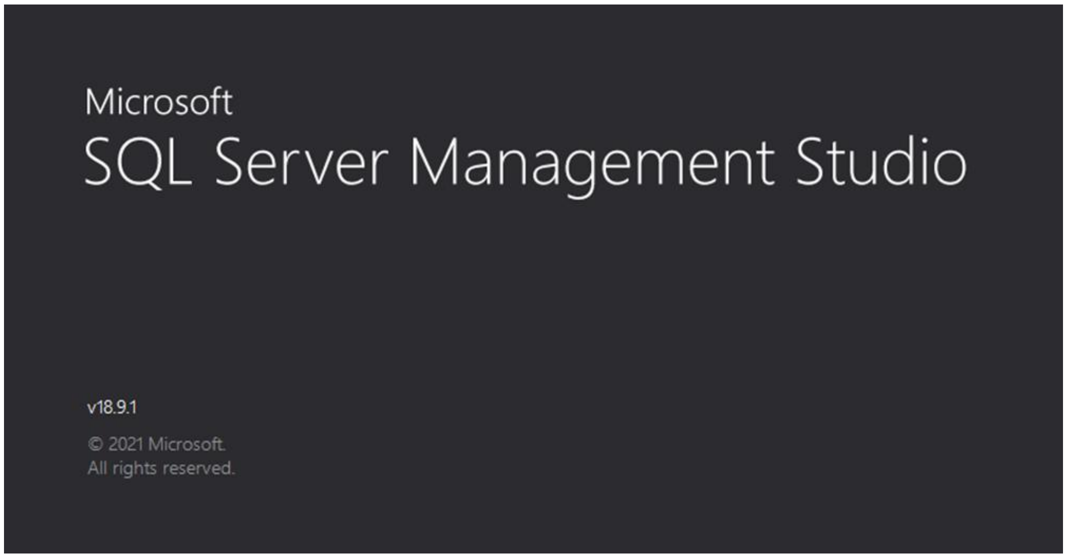
Table 1.2: Plan-driven model Advantages and Disadvantages [3, 12**]**

1.8 Development tools:

- Microsoft Visual Studio Enterprise 2019 Version 16.10.3



- Microsoft SQL Server Management Studio version 18.9.1



- JetBrains ReSharper 2021.1.4 211.0.20210713

- GitHub website

1.9 Frameworks and programming languages used:

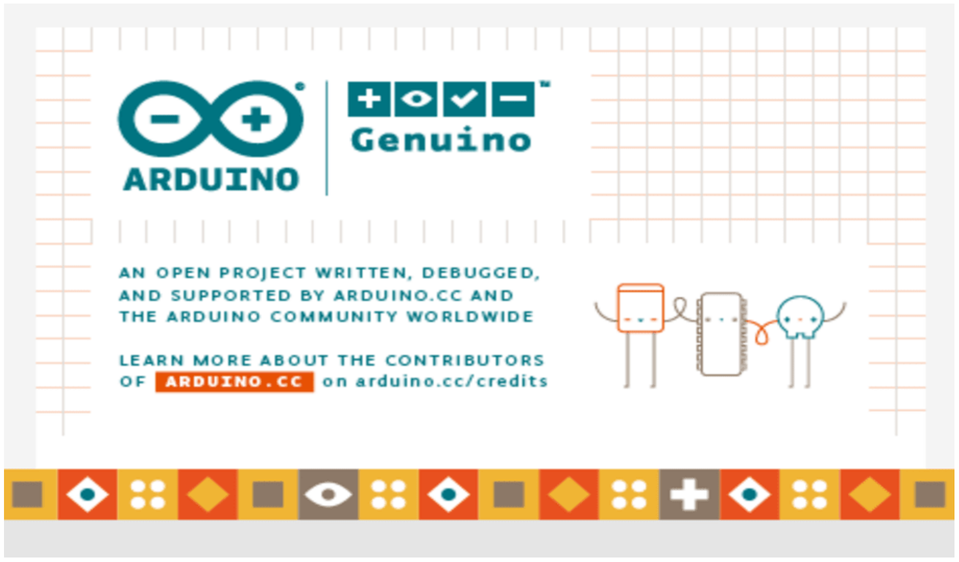
- Dot-Net core 5.0

- C-Sharp 9.0

- Microsoft SQL Server 2019

- git version 2.32.0. windows.2

- Arduino IDE 1.8.15



AN OPEN PROJECT WRITTEN, DEBUGGED, AND SUPPORTED BY ARDUINO.CC AND THE ARDUINO COMMUNITY WORLDWIDE

LEARN MORE ABOUT THE CONTRIBUTORS OF ARDUINO.CC on arduino.cc/credits

- Altium designer 21.0.8

ALTIUM DESIGNER

- Atmel studio 7

- MATLAB version R2020b

Chapter 2: Technicality Overview Part One

A closer look at

The Hardware Engineering of The Project

2.1 Attributes of Embedded system:

Embedded systems are designed so that the resulting device behaves in certain desirable ways.

- Embedded systems need to respond to events which occur in the environment, whether a user presses a button, or a motor overheats. A system which is not sufficiently responsive is not likely to be a successful product. For example, when we press a channel select button for the radio, we would like for it to respond within some reasonable time [1]

- For **real- time systems**, the timing of the responses is critical because late answers are wrong answers. Igniting the fuel in a cylinder is time- critical because bad timing can damage or destroy engine components (to say nothing of reducing power, or the efficiency and pollution concerns mentioned previously)

- Embedded systems typically require sophisticated fault handling and diagnostics to enable safe and reliable operation. Often the fault handling code is larger and more complex than the normal operation code. It is easy to design for the "Everything goes right and works fine" case. It is far more difficult to determine methods to handle the exceptional cases. What is likely to fail? Which failures can lead to dangerous conditions? How should the system handle failures? How will you test that the system handles the failures correctly?

- Embedded systems may be expected to **operate in dependently** for years without operator attention such as adjustment or resetting. The system is expected to operate robustly and always work. Given that it is very difficult and expensive to write perfect, bug- free software, developers build in mechanisms to detect faulty behavior and respond, perhaps by restarting the system.

2.2 CONSTRAINTS ON EMBEDDED SYSTEMS:

- Embedded systems often have constraints which limit the designer's options and can lead to creative and elegant solutions. These constraints are typically different from those for general-purpose computers.

- Cost is a common constraint. Many applications which use embedded systems are sold in very competitive markets, in which price is a major factor. Often a manufacturer will hire subcontractors to design and build individual sub- systems. This allows the manufacturer to pit potential subcontractors against each other, keeping prices down.

- There may be size and weight limits for portable and mobile applications. An embedded system for an automotive remote keyless entry transmitter must fit into a small fob on a key ring which fits easily into a pocket. Similarly, the receiver must not be too heavy. A heavier car will have worse acceleration, braking, cornering, and fuel efficiency. Aircraft and spacecraft are especially sensitive to weight since a heavier craft requires more fuel to achieve the same range.

- There may be limited power or energy available. For example, a battery has a limited amount of energy, while a solar cell generates a limited amount of power.
  High temperatures may limit the amount of cooling available, which will limit the power which can be used.

- The environment may be harsh. Automotive electronics under the hood of a car need to operate across a wide range of temperatures (_40°C to 125°C, or _40°F to 193°F), while withstanding vibrations, physical impact, and corroding salt spray. Spark plugs generate broadband radio frequency energy which can interfere with electronics [1]

2.3 SOFTWARE DEVELOPMENT STAGES:

Development Lifecycle Overview

- The embedded product lifecycle has multiple steps, as shown in Figure 1.1
- The software process consists of multiple steps within the embedded product lifecycle.

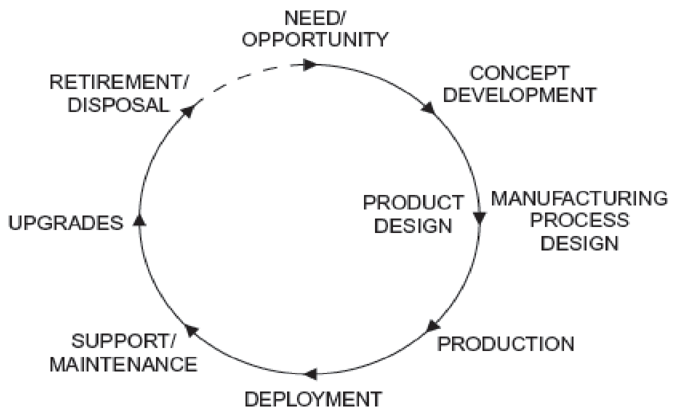


Figure 2.1: The embedded product lifecycle [1]

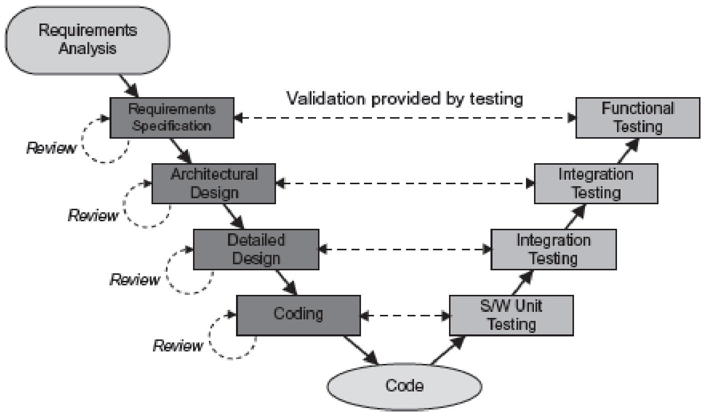


Figure 2.2: The "V" model of software development emphasizes testing at each level of design detail [1]

One example process model is the V model, shown in Figure 1.2. It consists of:
Defining system requirements.

- Creating an architectural or high- level design, deciding on the general approach to build the system, and then creating appropriate hardware and software architectures.
- Creating detailed designs.
- Implementing code and performing unit testing.
- Integrating the code components and performing integration testing.
- Changing the code after “completion” to fit custom deployment requirements, fix bugs, add features etc.
- The process breaks a problem into smaller easier problems through the process of top-down design or decomposition. Each small problem is solved, and the solutions are combined into a software solution.

There are many approaches possible:

- Do we design everything up front, and then code it? This is called a big up- front design.
- Do we have a prototype or a previous version which we can build upon?
- Do we design and build a little at a time?
- What do we work on first, the easy or the hard parts?
- Which model we should use depends on the types and severity of risks. Some industries may be required to follow a specific development process for the product to receive certification.

2.4 Finite State Machine for microcontroller code:

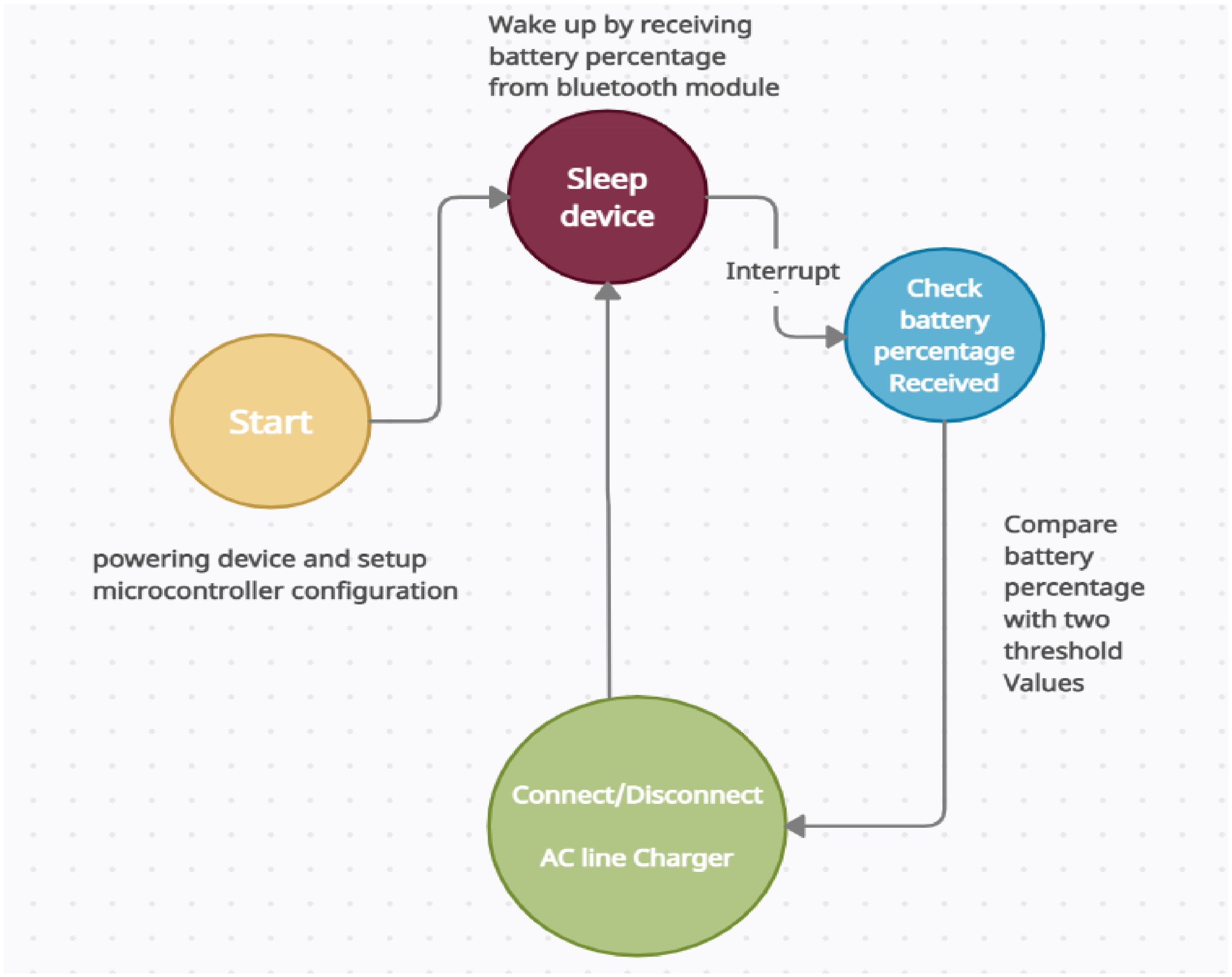

Wake up by receiving battery percentage from bluetooth module
Sleep device
Interrupt
Check battery percentage Received
Start
powering device and setup microcontroller configuration
Compare battery percentage with two threshold Values
Connect/Disconnect AC line Charger

Figure 2.3: Finite State Machine For microcontroller code

2.5 Flow chart for microcontroller code:

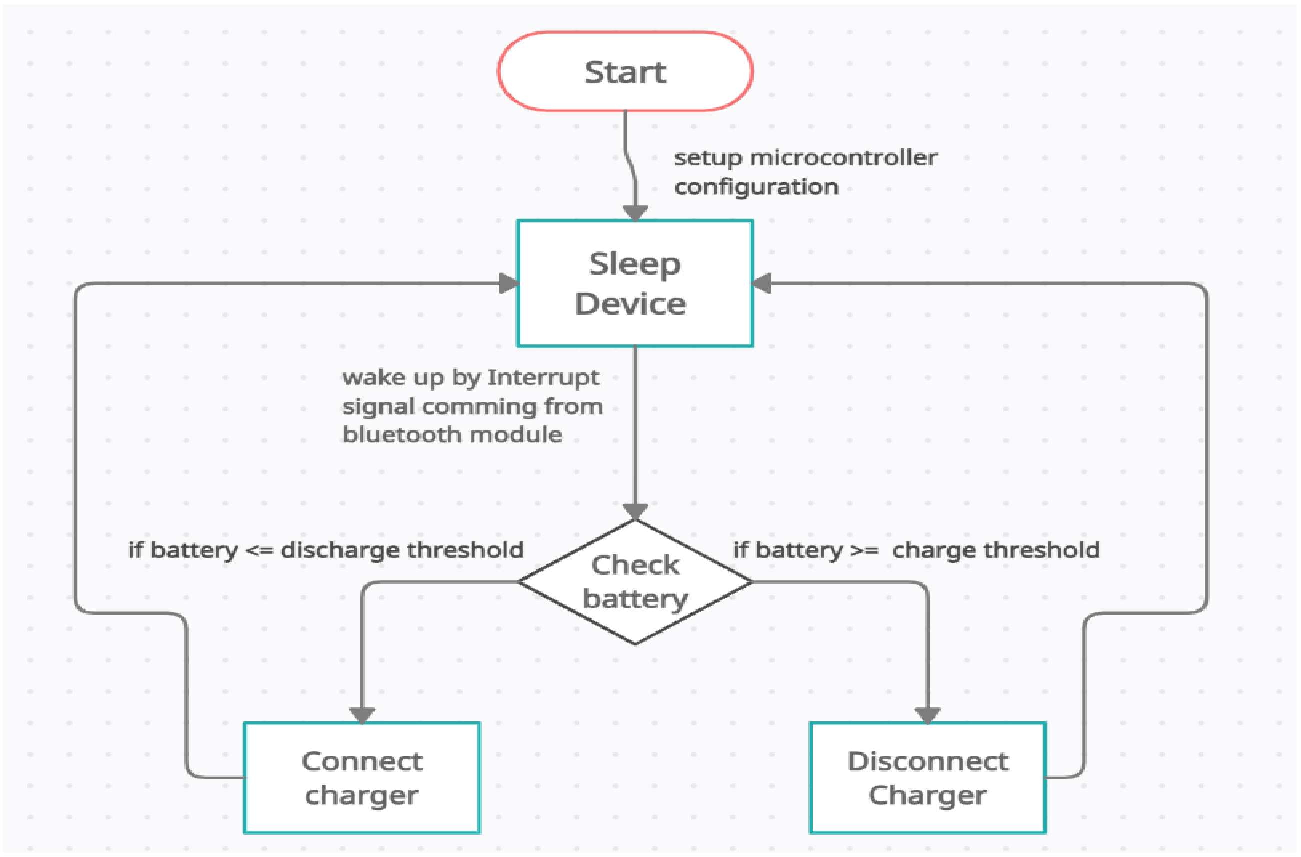

Start
setup microcontroller configuration
Sleep Device
wake up by Interrupt signal comming from bluetooth module
Check battery
if battery <= discharge threshold
if battery >= charge threshold
Connect charger
Disconnect Charger

Figure 2.4: Flow chart for microcontroller code

2.6 Stages of designing hardware:

Our project has passed several stages from raw material in our head until it become a real prototype which is ready to manufacture.

1. Raw Idea:
   - This stage has a lot of brainstorming, searching until the it became
   - Easy to be convertible into a prototype.

2. Prototype:
   - Every Embedded system device must be examined well after it come out to light, So, we make our device in a breadboard which is famous in this field, it's just an 2D- array of points to connect the circuit components with each other without needing to solder or use a lot of cost for simple examining to the initial trials of the device.

3. Hardware:

This is the most important factor that control the cost of any embedded system device, so we had to take a lot of consideration into account during the journey of designing, implementing, and testing the HW:

a. Size: The size is very crucial factor as depending on it the product will be portable or not, as well as every additional mm^2 in circuit material which made of copper will lead to more cost, so the size must be optimized well.

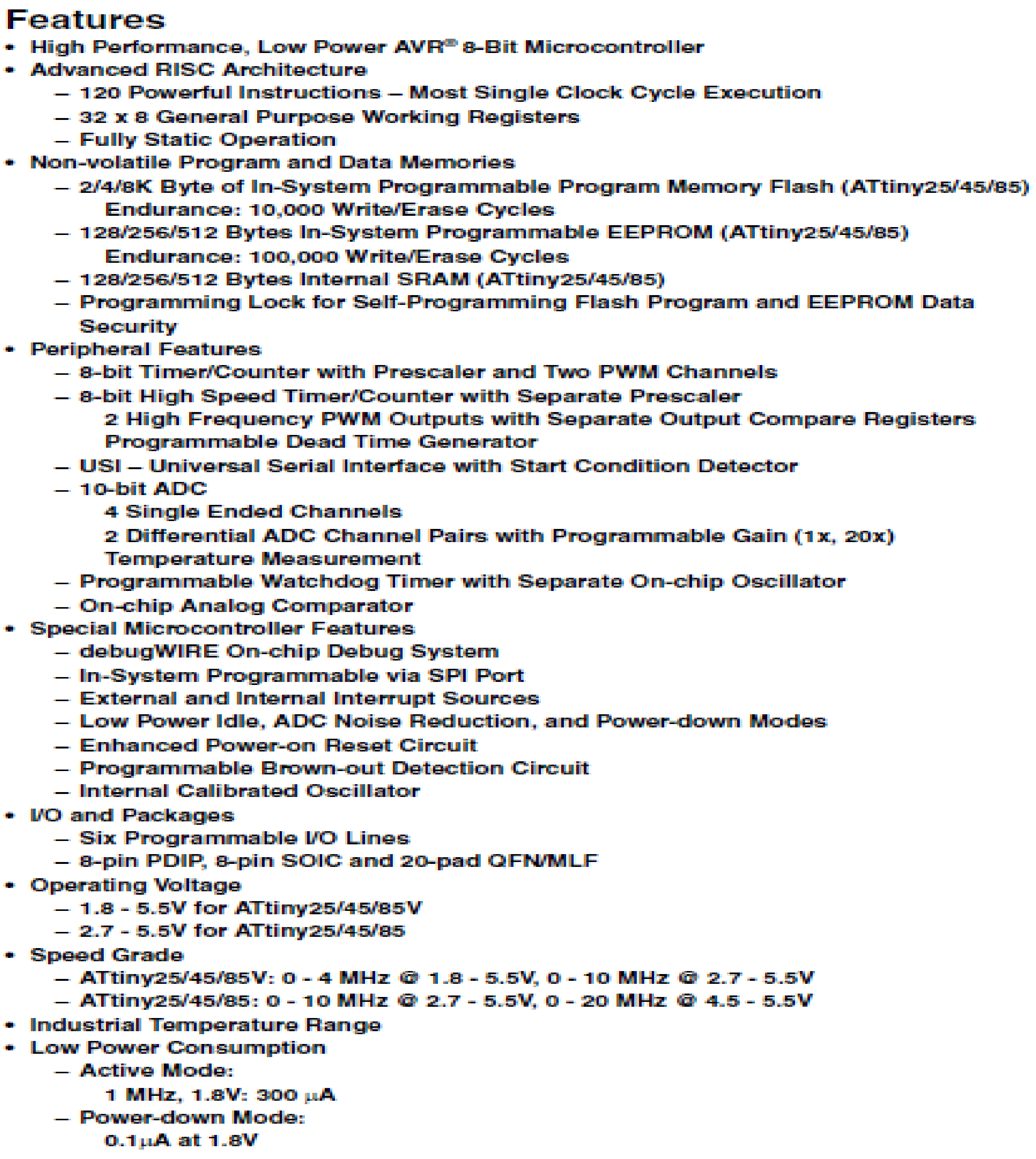
b. Choosing the suitable MCU to fit our application: in fact, stage is hard as the embedded market has a lot of vendors of MCUs and every vendor has several families to its architecture so after knowing what our product needs plus some experience in dealing with embedded systems we choose Attiny85 MCU which is dependent on AVR architecture.

Why we choose this specific type?

- It's the smallest size of MCUs that fit our application.
- Its power consumption is very efficient, as it has sleep modes that consume very tiny current in 0.31 mAmps.
- It has 6 GPIOs, we use one of them to control the AC line 220V charger, and we use two other GPIOs to send and receive data from Bluetooth Module, we use one of them to transmit data, and the other to receive we do that because of the small size of the MCU which make it doesn't support the appropriate communication protocol to talk to and listen from the Bluetooth module which in our case called USART so we made a beautiful trick by using Software Serial library that overcome this problem and give the Attiny85 MCU the ability to send data serially bit by bit with same baud rate of Bluetooth Module, without real USART hardware module embedded in the MCU, so we act in this step like a tailor when he try to convert the cloth into clothes.

2.7 Microcontroller datasheet:

First let us see microcontroller features:



Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 120 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
- Non-volatile Program and Data Memories
  - 2/4/8K Byte of In-System Programmable Program Memory Flash (ATtiny25/45/85)
    Endurance: 10,000 Write/Erase Cycles
  - 128/256/512 Bytes In-System Programmable EEPROM (ATtiny25/45/85)
    Endurance: 100,000 Write/Erase Cycles
  - 128/256/512 Bytes Internal SRAM (ATtiny25/45/85)
  - Programming Lock for Self-Programming Flash Program and EEPROM Data Security
- Peripheral Features
  - 8-bit Timer/Counter with Prescaler and Two PWM Channels
  - 8-bit High Speed Timer/Counter with Separate Prescaler
    2 High Frequency PWM Outputs with Separate Output Compare Registers
    Programmable Dead Time Generator
  - USI – Universal Serial Interface with Start Condition Detector
  - 10-bit ADC
    4 Single Ended Channels
    2 Differential ADC Channel Pairs with Programmable Gain (1x, 20x)
    Temperature Measurement
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - debugWIRE On-chip Debug System
  - In-System Programmable via SPI Port
  - External and Internal Interrupt Sources
  - Low Power Idle, ADC Noise Reduction, and Power-down Modes
  - Enhanced Power-on Reset Circuit
  - Programmable Brown-out Detection Circuit
  - Internal Calibrated Oscillator
- I/O and Packages
  - Six Programmable I/O Lines
  - 8-pin PDIP, 8-pin SOIC and 20-pad QFN/MLF
- Operating Voltage
  - 1.8 - 5.5V for ATtiny25/45/85V
  - 2.7 - 5.5V for ATtiny25/45/85
- Speed Grade
  - ATtiny25/45/85V: 0 - 4 MHz @ 1.8 - 5.5V, 0 - 10 MHz @ 2.7 - 5.5V
  - ATtiny25/45/85: 0 - 10 MHz @ 2.7 - 5.5V, 0 - 20 MHz @ 4.5 - 5.5V
- Industrial Temperature Range
- Low Power Consumption
  - Active Mode:
    1 MHz, 1.8V: 300 µA
  - Power-down Mode:
    0.1µA at 1.8V

Figure 2.5: Microcontroller features [4]

2.8 Microcontroller pin description on both types of ICs (SMD, DIP)

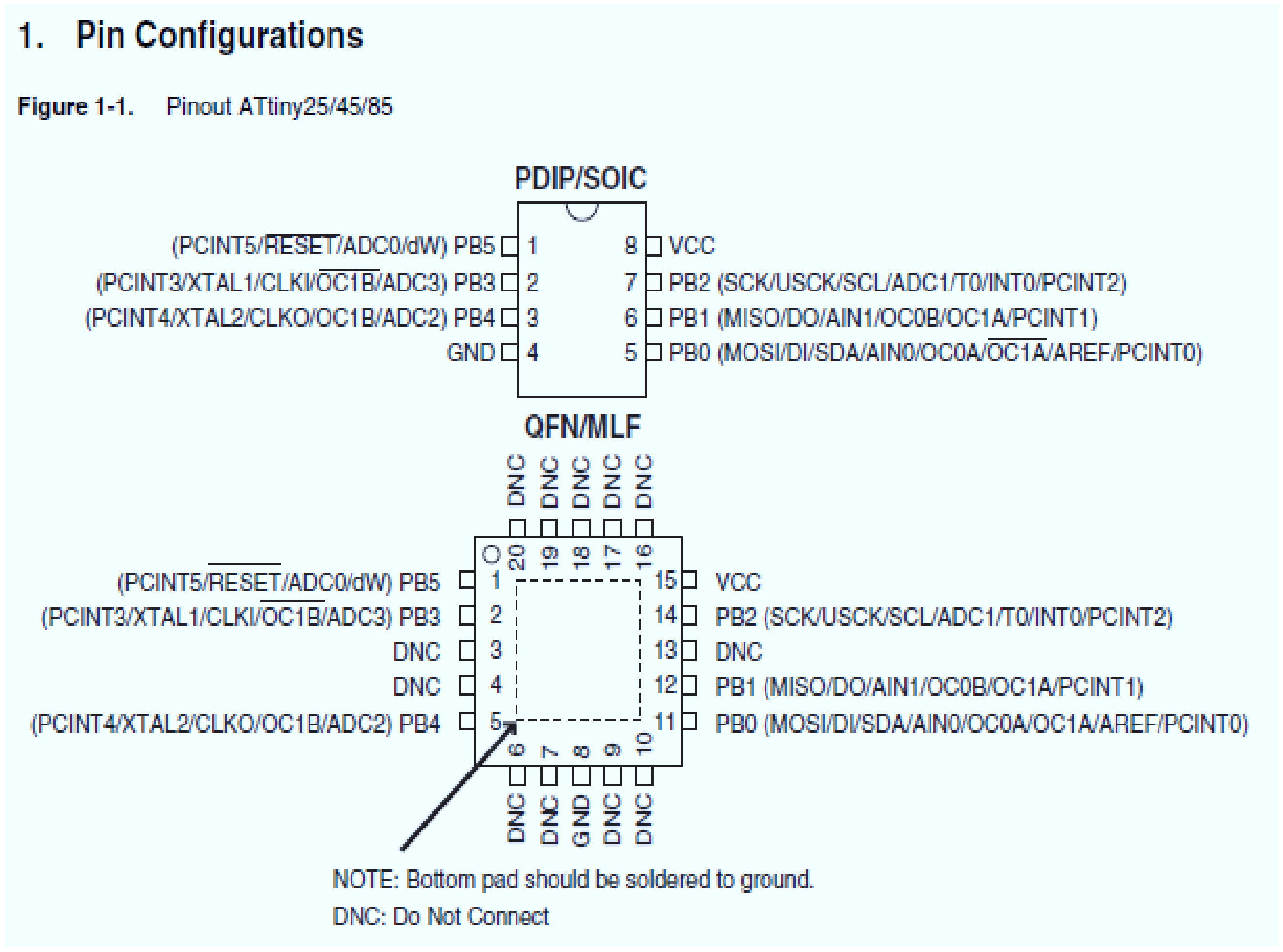

1. Pin Configurations
Figure 1-1. Pinout ATtiny25/45/85
PDIP/SOIC
(PCINT5/RESET/ADC0/dW) PB5 1
(PCINT3/XTAL1/CLKI/OC1B/ADC3) PB3 2
(PCINT4/XTAL2/CLKO/OC1B/ADC2) PB4 3
GND 4
8 VCC
7 PB2 (SCK/USCK/SCL/ADC1/T0/INT0/PCINT2)
6 PB1 (MISO/DO/AIN1/OC0B/OC1A/PCINT1)
5 PB0 (MOSI/DI/SDA/AIN0/OC0A/OC1A/AREF/PCINT0)
QFN/MLF
DNC DNC DNC DNC DNC
20 19 18 17 16
(PCINT5/RESET/ADC0/dW) PB5 1
(PCINT3/XTAL1/CLKI/OC1B/ADC3) PB3 2
DNC 3
DNC 4
(PCINT4/XTAL2/CLKO/OC1B/ADC2) PB4 5
15 VCC
14 PB2 (SCK/USCK/SCL/ADC1/T0/INT0/PCINT2)
13 DNC
12 PB1 (MISO/DO/AIN1/OC0B/OC1A/PCINT1)
11 PB0 (MOSI/DI/SDA/AIN0/OC0A/OC1A/AREF/PCINT0)
6 7 8 9 10
DNC DNC GND DNC DNC
NOTE: Bottom pad should be soldered to ground.
DNC: Do Not Connect

Figure 2.6: Microcontroller pin description [4]

From this datasheet we conclude that this microcontroller has a little number of GPIOs,

It has only I2C,SPI communication protocol we use SPI to burn the code on the microcontroller using the programmer hardware (Arduino as ISP) , we overcome the shortage of USART protocol which we need it as way to communicate with HC-06 Bluetooth module by using SoftwareSerial library which emulates the protocol and send data bit by bit by means of delays, and ensures that the data has arrived properly by means of software Boolean flags, it also ensure the baud rate by some calculations by dividing the Clock of the microcontroller by the appropriate pre-scalar to provide the common baud rates:

(9600,19200,38400,57600,115200) bits per second.

2.9 Hardware components:

When we talk about circuit that consist of some modules, we should illustrate all of them separately to show it’s features and together to show how this module interacting with each to make the device work properly as whole and this is the big picture.

a) **Power Supply:**

- Our MCU should be supplied with 5v, we have a lot of choices like buying AC/DC 220/5 Adapter like the one which all of us use it to supply the router of the internet, But the problem is the device won’t be portable, so we search a lot until we find small AC/DC 220/5v 1Amp converter, called Hi-link and it has the following features:

1- Ultra-thin, ultra-small, smallest volume.

2- Global universal input voltage（90~245Vac)

3- Low power consumption, green environmental protection, no-load loss<0.1W

4- Low ripple, low noise.

5- High output short circuit and over-current protection and self-recovery.

6- High efficiency, high power density.

7- Input and output isolation voltage 3000Vac

8- 100% full load aging and testing.

9- High reliability, long life design, continuous working time is greater than 100,000 hours

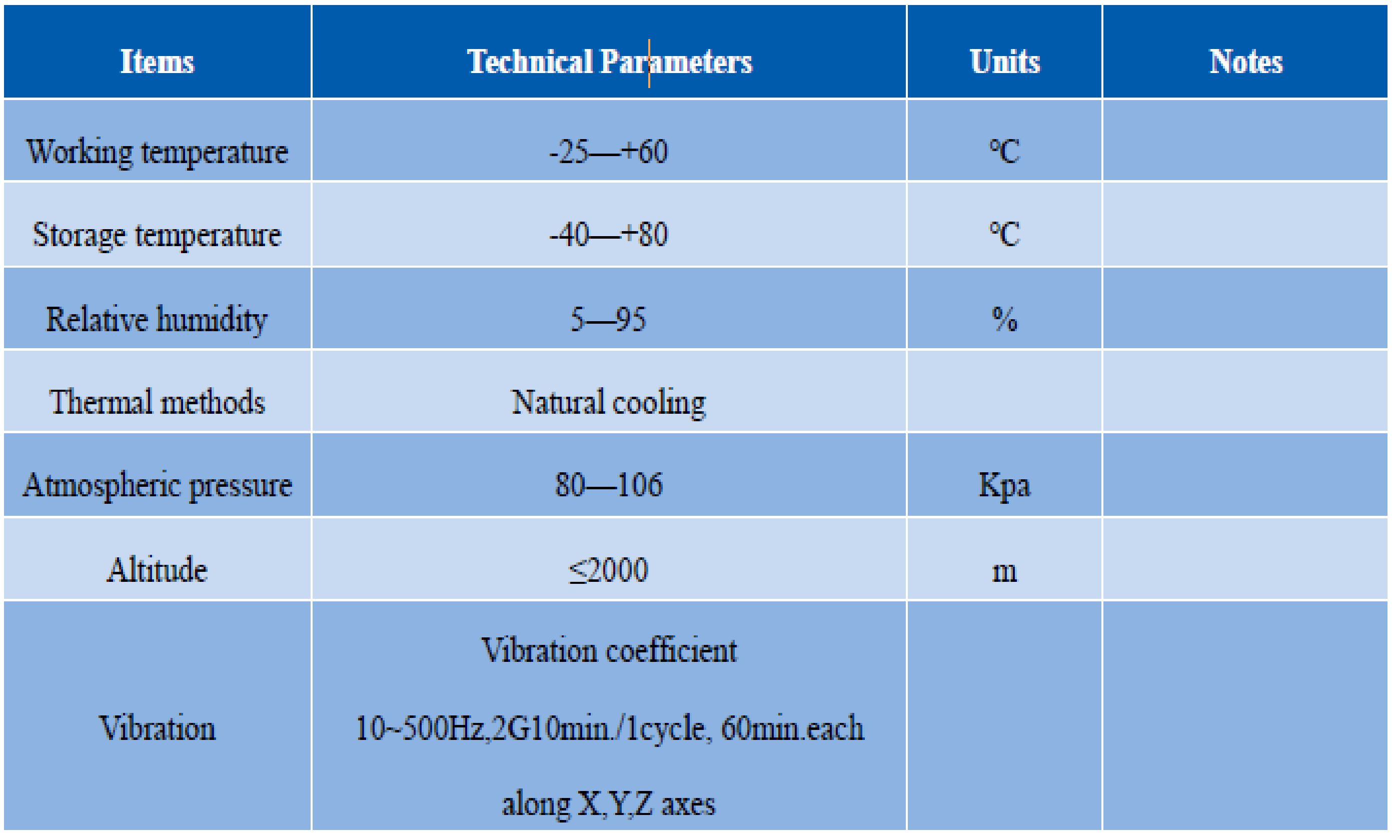
10- Meet UL, CE requirements; product design to meet EMC and safety testing requirement

11- Using high-quality environmentally friendly waterproof plastic potting, moisture, vibration, water, and dust to et IP65 standards

12- Economic solutions, cost-effective

13- Work without external circuit

14- 1 year quality guarantee period

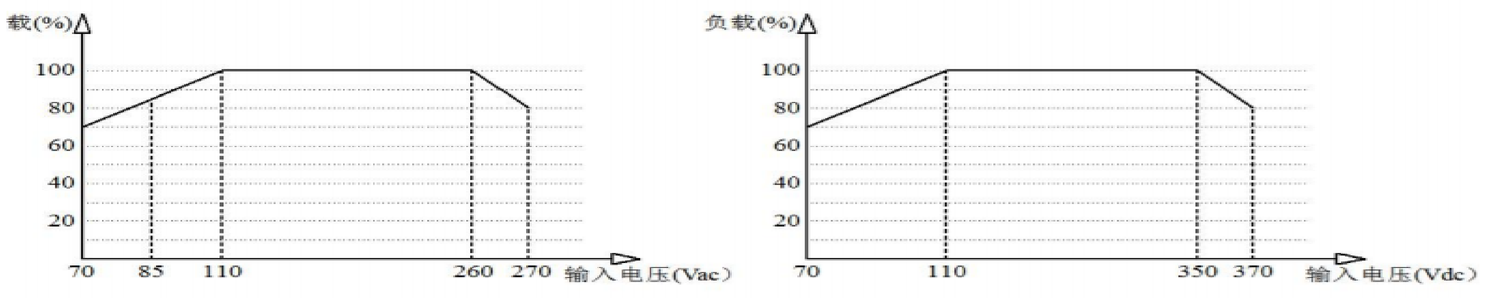


Items	Technical Parameters	Units	Notes
Working temperature	-25—+60	℃	
Storage temperature	-40—+80	℃	
Relative humidity	5—95	%	
Thermal methods	Natural cooling		
Atmospheric pressure	80—106	Kpa	
Altitude	≤2000	m	
Vibration	Vibration coefficient 10~500Hz,2G10min./1cycle, 60min.each along X,Y,Z axes		

Table 3.1: Environmental Conditions of HiLink [6]

Input Features of Hi-Link:

Input voltage and load characteristics



Input voltage and load characteristic curve

Working environment temperature and load characteristics

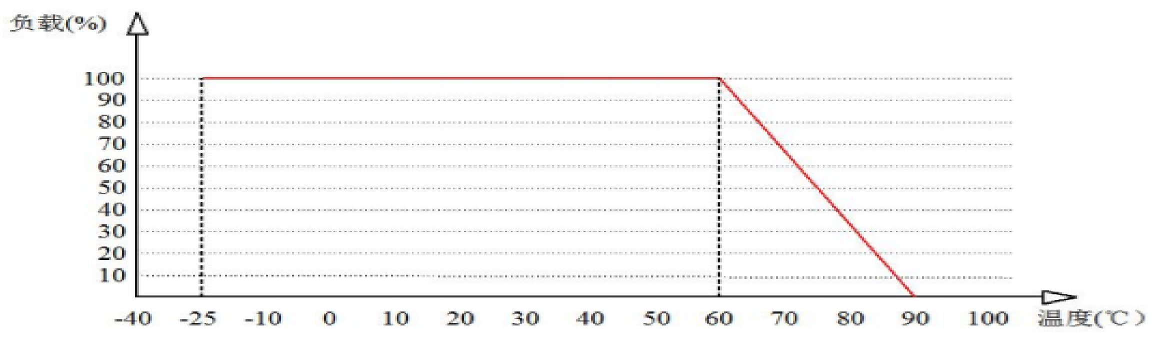
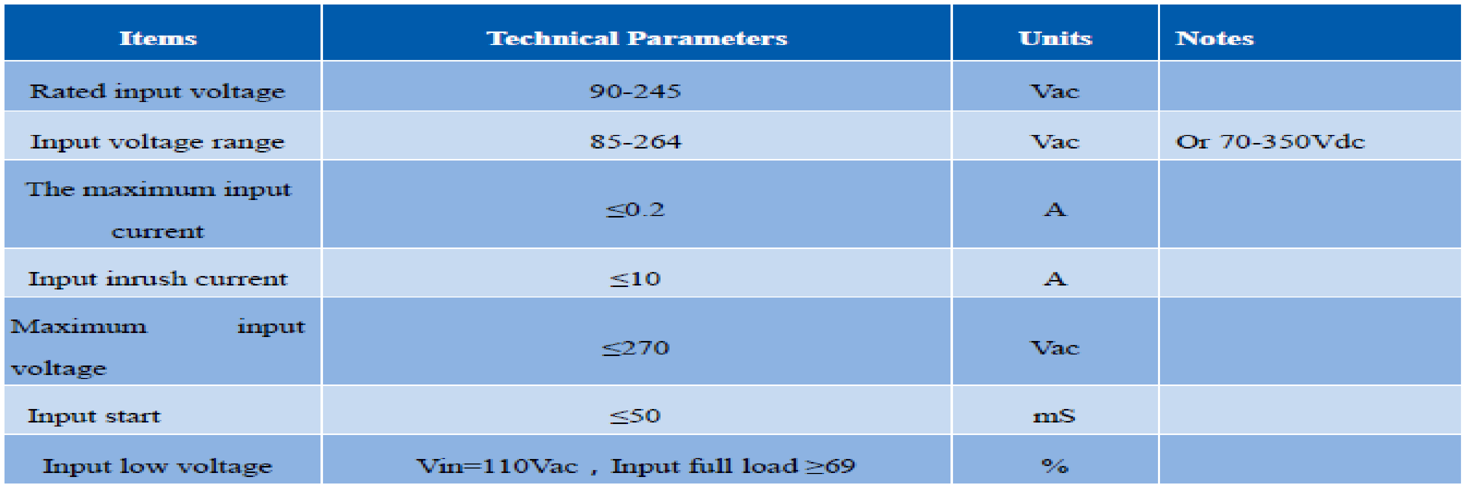


Figure 2.7: Hi-Link important graphs [6]



Items	Technical Parameters	Units	Notes
Rated input voltage	90-245	Vac	
Input voltage range	85-264	Vac	Or 70-350Vdc
The maximum input current	≤0.2	A	
Input inrush current	≤10	A	
Maximum input voltage	≤270	Vac	
Input start	≤50	mS	
Input low voltage	Vin=110Vac，Input full load ≥69	%	

Table 3.2: Dimensions and weight of Hi-Link [6]

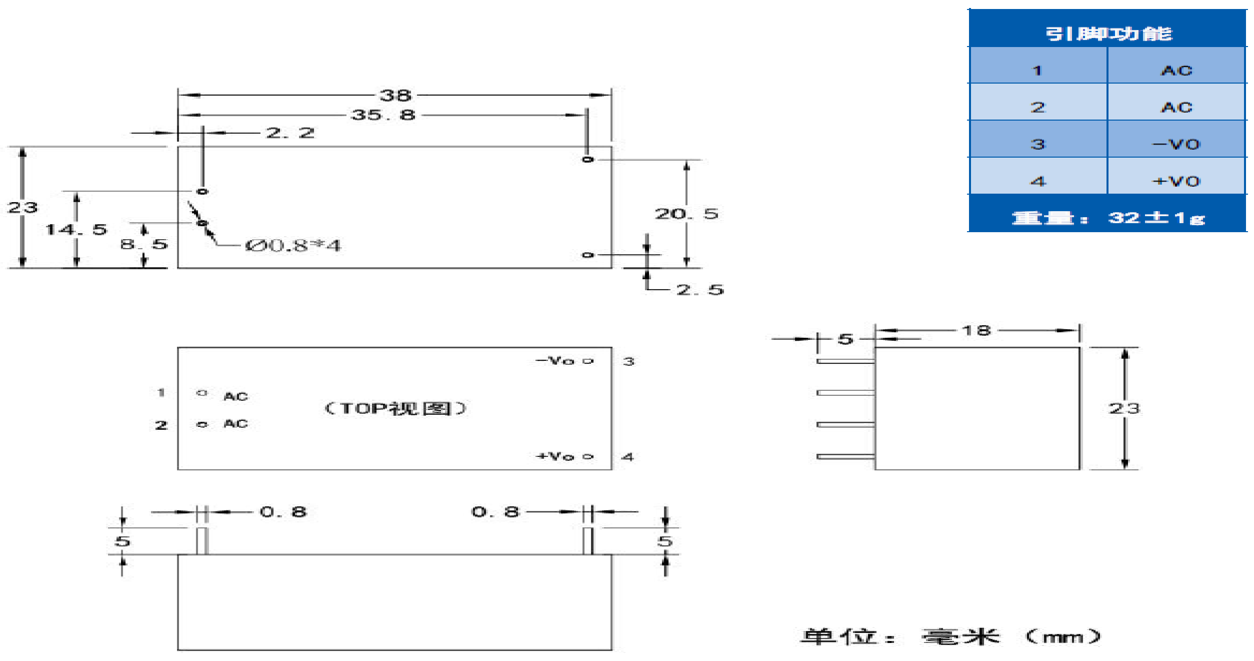
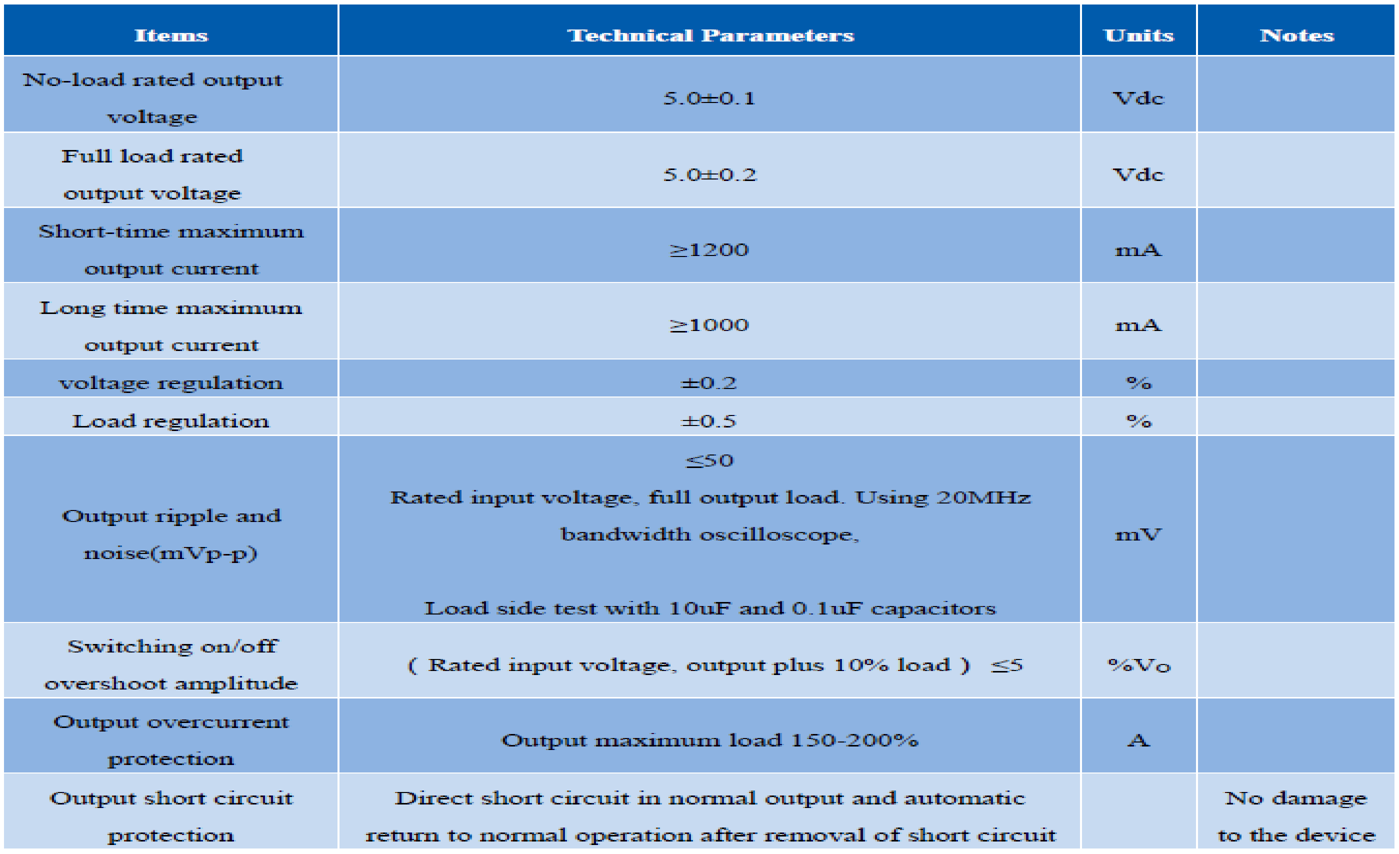


Figure 2.8: Output Features of Hi-link [6]



Items	Technical Parameters	Units	Notes
No-load rated output voltage	5.0±0.1	Vdc	
Full load rated output voltage	5.0±0.2	Vdc	
Short-time maximum output current	≥1200	mA	
Long time maximum output current	≥1000	mA	
voltage regulation	±0.2	%	
Load regulation	±0.5	%	
Output ripple and noise(mVp-p)	≤50 Rated input voltage, full output load. Using 20MHz bandwidth oscilloscope, Load side test with 10uF and 0.1uF capacitors	mV	
Switching on/off overshoot amplitude	（Rated input voltage, output plus 10% load） ≤5	%Vo	
Output overcurrent protection	Output maximum load 150-200%	A	
Output short circuit protection	Direct short circuit in normal output and automatic return to normal operation after removal of short circuit		No damage to the device

Table 3.3: Output Features of Hi-link [6]

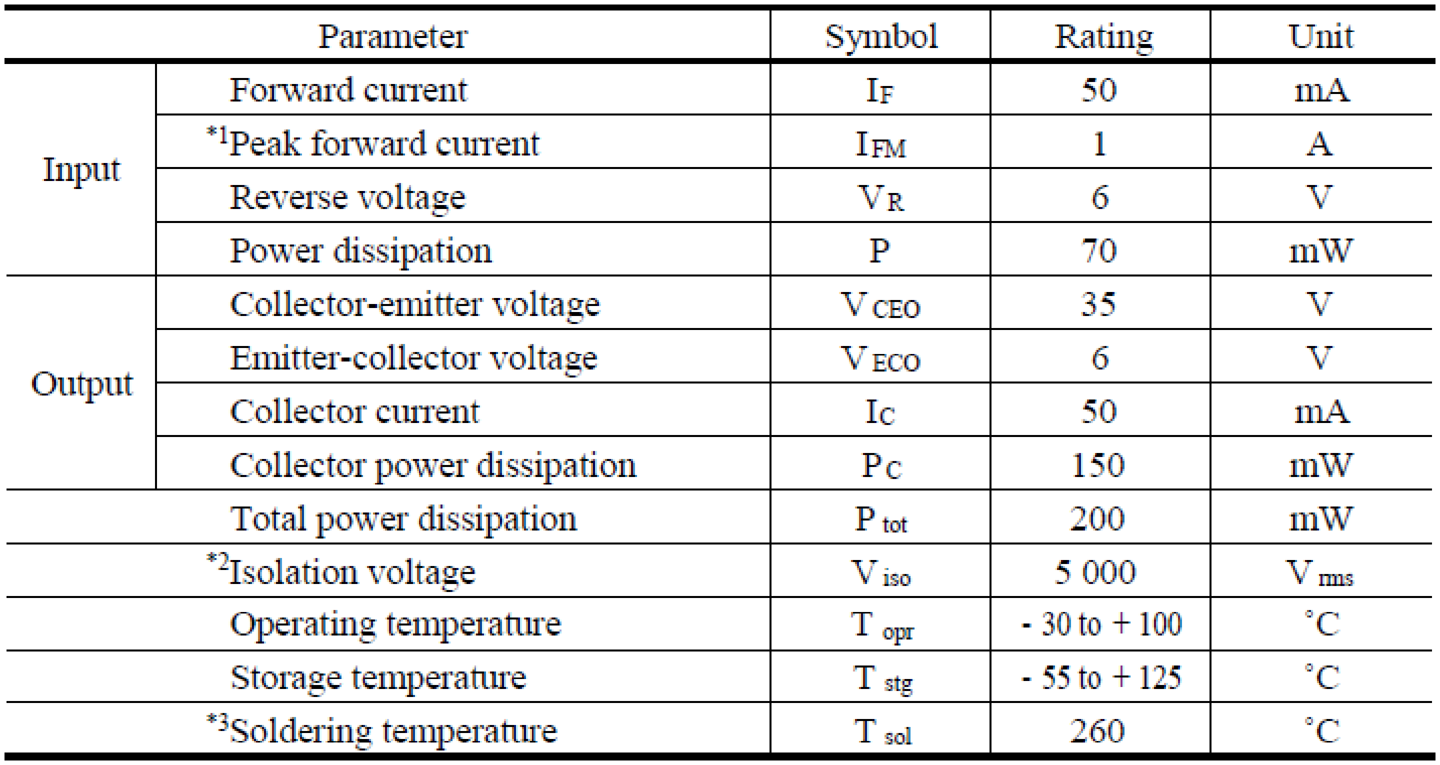
b) Optocoupler (PC817)

- By using this component, we ensure that the brain of our system is fully isolated from the High Voltage, as there is no physical connection between the output signal from the microcontroller and the relay which switch on/off the 220V charger.
- The technology used inside this component is different from the normal transistors as it has a led inside it and opto-transistor which its base supplied by turn this led on.
- Every Embedded system engineer must be aware what is called microcontroller logic level which maybe 5v or 3.3 according to microcontroller Architecture , in our case Attiny85 logic level = 5v , but our aim is to control High voltage load 220V AC line which control the charger so the best solution is to isolate our microcontroller from damage by adding optocoupler which is in our case (pc817), this is one way to protect the tiny brain of the circuit by isolating it from the relay that control the 220 AC line Voltage, the best advantage that no physical connection between the relay and MCU because the optocoupler consist of two parts:
    - Internal led that take signal form the MCU.
    - Opto-transistor which it’s base should be emitted by turning this internal led on

- So, we take Electro-Optical characteristic table from datasheet:

Parameter			Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Input	Forward voltage		V_F	$I_F = 20mA$	-	1.2	1.4	V
	Peak forward voltage		V_{FM}	$I_{FM} = 0.5A$	-	-	3.0	V
	Reverse current		I_R	$V_R = 4V$	-	-	10	μA
	Terminal capacitance		C_t	V = 0, f = 1kHz	-	30	250	pF
Output	Collector dark current		I_{CEO}	$V_{CE} = 20V$	-	-	10^{-7}	A
Transfer charac-teristics	*4Current transfer ratio		CTR	$I_F = 5mA, V_{CE} = 5V$	50	-	600	%
	Collector-emitter saturation voltage		$V_{CE(sat)}$	$I_F = 20mA, I_C = 1mA$	-	0.1	0.2	V
	Isolation resistance		R_{ISO}	DC500V, 40 to 60% RH	5×10^{10}	10^{11}	-	Ω
	Floating capacitance		C_f	V = 0, f = 1MHz	-	0.6	1.0	pF
	Cut-off frequency		f_c	$V_{CE} = 5V, I_C = 2mA, R_L = 100\Omega$, -3dB	-	80	-	kHz
	Response time	Rise time	t_r	$V_{CE} = 2V, I_C = 2mA, R_L = 100\Omega$	-	4	18	μs
		Fall time	t_f		-	3	18	μs

Table 3.4: Electro-Optical characteristic table from datasheet [5]



Parameter		Symbol	Rating	Unit
Input	Forward current	I_F	50	mA
	[*1]Peak forward current	I_{FM}	1	A
	Reverse voltage	V_R	6	V
	Power dissipation	P	70	mW
Output	Collector-emitter voltage	V_{CEO}	35	V
	Emitter-collector voltage	V_{ECO}	6	V
	Collector current	I_C	50	mA
	Collector power dissipation	P_C	150	mW
Total power dissipation		P_{tot}	200	mW
[*2]Isolation voltage		V_{iso}	5 000	V_{rms}
Operating temperature		T_{opr}	- 30 to + 100	°C
Storage temperature		T_{stg}	- 55 to + 125	°C
[*3]Soldering temperature		T_{sol}	260	°C

Table 3.5: Electro-Optical of I/O [5]

The absolute and maximum ratings:

c) Relay:

- To make this device work properly we used another layer of isolation which is a relay that is an electromechanical switch which is controlled electrically. It is used for isolation of two circuits having different operating voltages. For example, we want to isolate 220-volt AC power supply from 5-volt dc power source, in such case relay is used to separate them. This type of relay is called electromagnetic relay (EMR)

d) Flyback diode:

- A flyback diode is a diode connected across an inductor which in our case the relay, used to eliminate flyback, which is the sudden voltage spike seen across an inductive load when its supply current is suddenly reduced or interrupted.

e) Indicator led:

- To notify the user of charge/discharge process.

f) Bluetooth Module:

- This is our way to communicate with PC laptop embedded Bluetooth to know the most important two events in our system.

2.10 The reasons for choosing Bluetooth in communication:

- Cheap this module is inexpensive.

- The distance between Pc and charger won't exceed the Range of communication in meters which equals 9m

- To take the advantage of embedded Bluetooth which every PC nowadays has.

- To make the device neat and portable as no wired connection between laptop and the device, this lead to make the device as small as possible.

- Save energy and this is an important environmental factor

2.11 Manufacturing:

- After the initial success of the prototype circuit, we want to convert it to real product, so we design our circuit using program called Altium designer latest version, first we made a schematic for the circuit, we organize the schematic to some modules:
  - power supply and its protection (fuse)
  - microcontroller (brain of the system)
  - external socket for future development of the code.
  - Bluetooth module.
  - Relay module and isolation using opto-isolator (pc817)
  - Battery charger socket.
  - external on/off switch to the whole circuit.

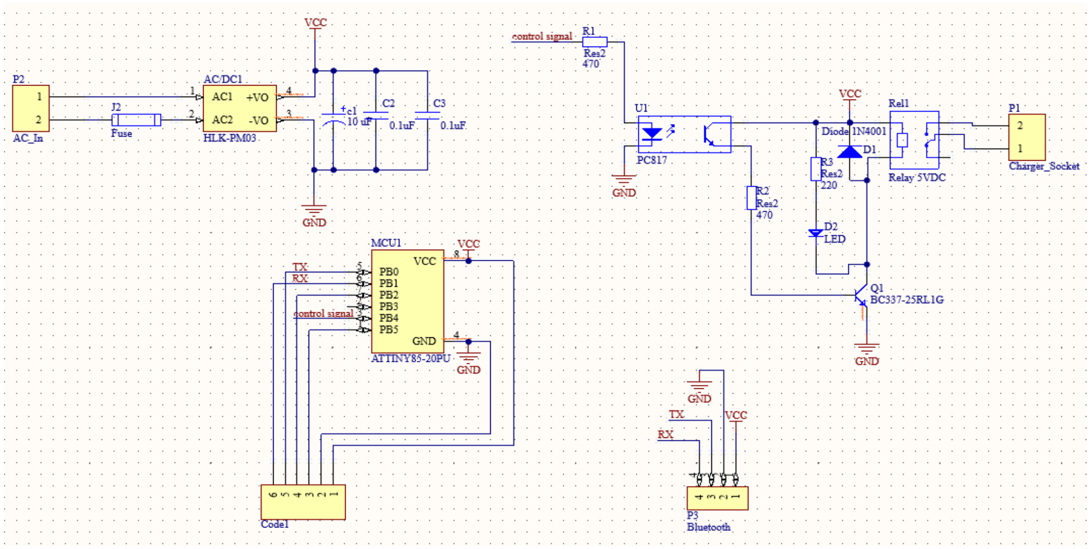


Figure 2.9: Schematic design

After schematic design stage we made a PCB design to this schematic:

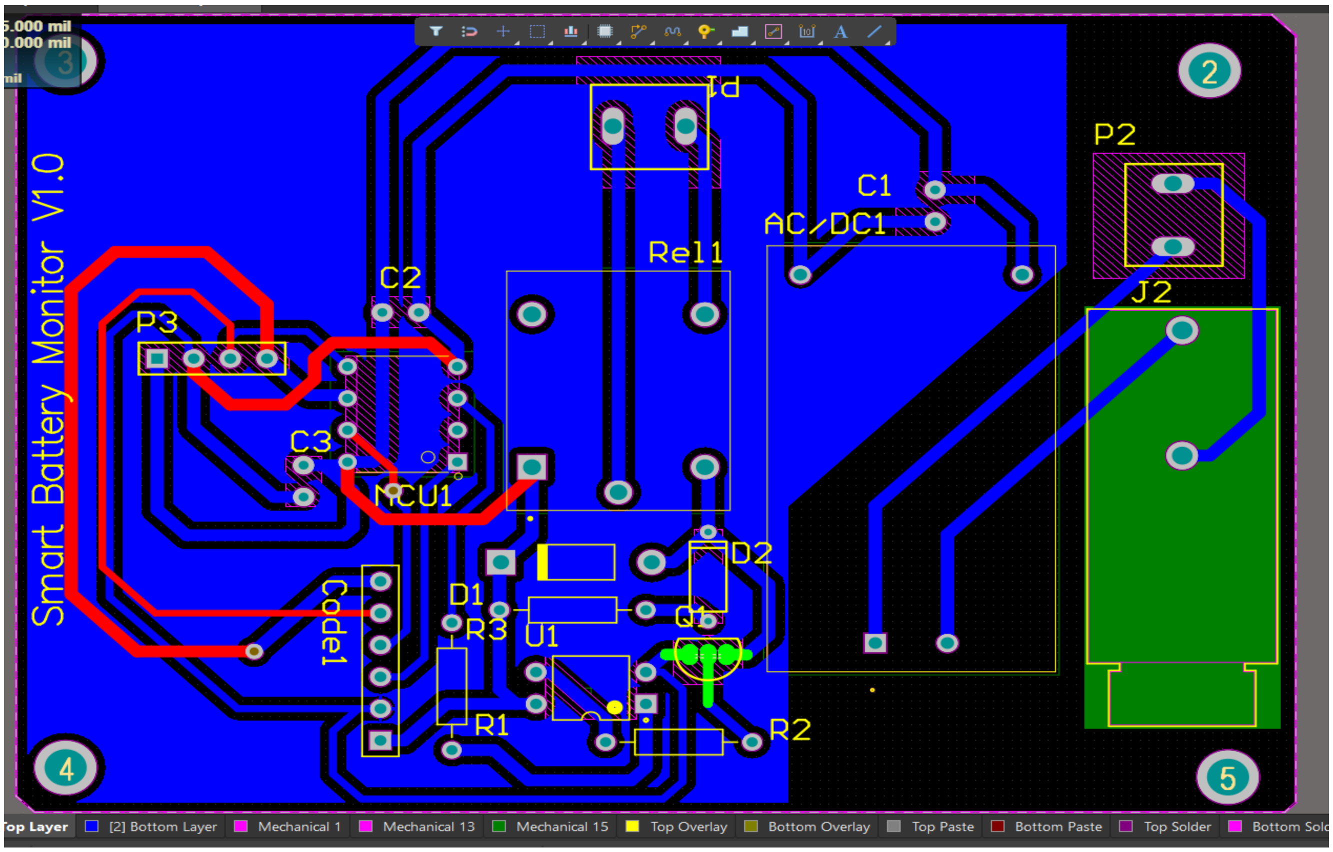


Figure 2.10: PCB 2D design

- This PCB is designed in two layers which means that the copper layers in the two layers of the board:
- The bottom layer which its color is blue, and the top layer which its color is red, this way of designing makes the board as small as possible which lead to less heat dissipation and then less power which lead to less cost and make the device portable which is the most important factor to a large scope of users.
- On the other hand making the board 2-layers lead us to hard manual design this board at home so we search for the cost for designing this board inside and outside Egypt and we find that the cost of designing is expensive as the companies won’t cut a large sheet of copper to make a small prototype, so there is a minimum number of boards to design which is ten boards , and this is a lot when we talk about one board prototype.

- This is the final 3D version of the board which we prepared to send to the manufacturer in China but according to high cost we will wait until we find a good source to fund us to convert this prototype into real product.

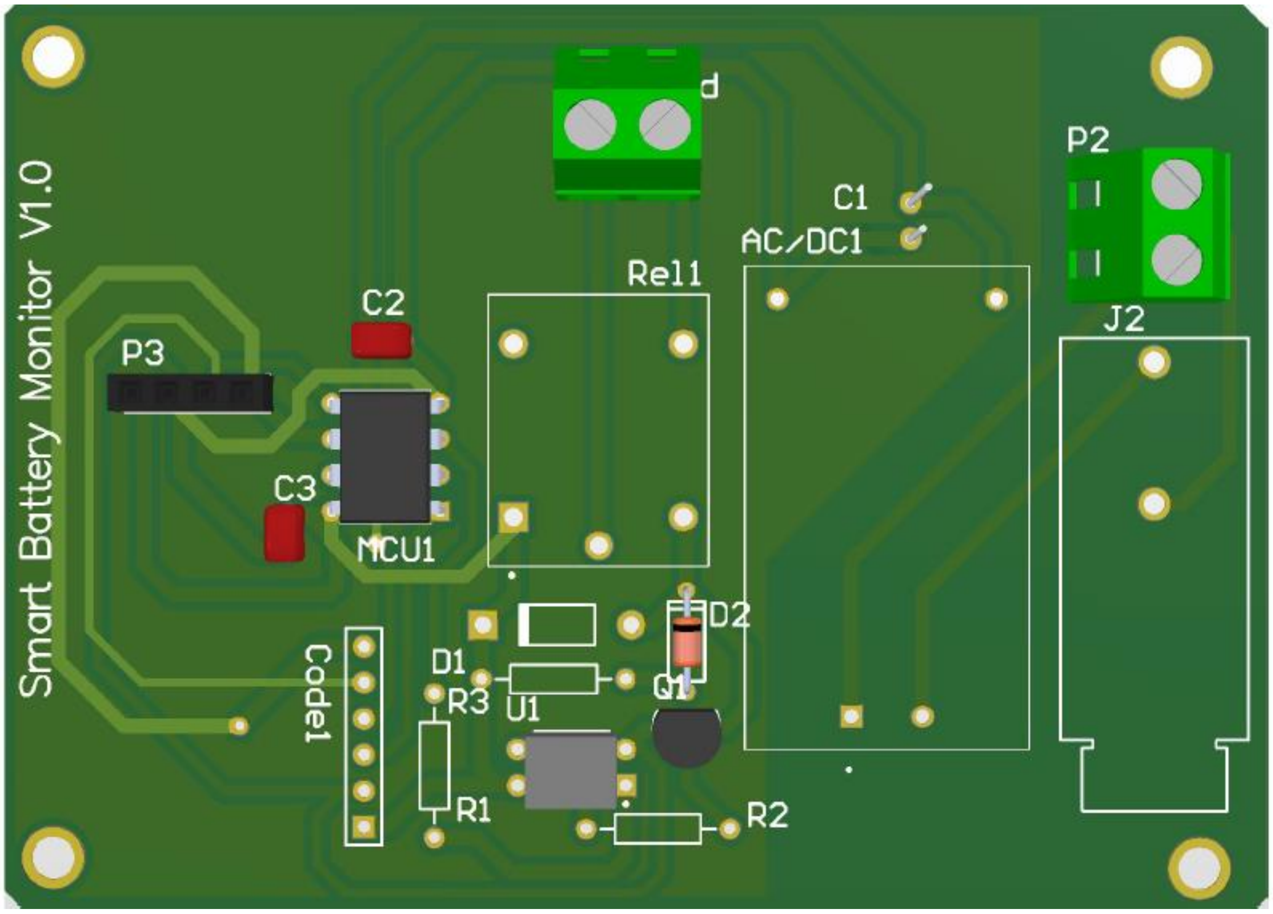


Figure 2.11: PCB 3D design

Chapter 3: Technicality Overview Part Two

A closer look at

The Software Engineering of The Project

3.1 INTRODUCTION:

- The system consists of two parts front end and back end, we combine between Object Oriented Programming and Functional Programming to make the program easy to maintain and readable as much as possible:
  This system contains eight entities which we converted them into eight classes when we design it with OOP.

- We made the UI just one form, this is for the seeking of the most important application purpose, which is saving energy, we testing this feature using Task Manager application in Windows platform.

- Because of the main environment of our application is to run in the background, the most important concept which our system depend on is multithreading, to enable the users to use their laptops freely without interrupting user's work.
  there are UI thread and a pool of threads, in some cases in one line or more in any method we need to deal with one of the "controls such as text-boxes or combo-boxes" in the UI which is impossible by using any other thread except the UI-thread, in this situation we use built in method in C# that check if we need to invoke UI thread or not.

- We use an important feature which is released on .NET Framework 3.5 in 2007 called LINQ (Language Integrated Query), this feature considered smart queries to manipulate data structures (which is called in C#: "Collections") without a lot of conditions statements and more loops.

- We summaries transactions in database into a logfile and calculate some statistical values depend on the time interval which user select.

- Thanks to version control (git) team members could work remotely without interference of each member's work and keep track of any update and belongs to whom.
  One of the main advantages of (git) is Commit History which describes stages of development and how we overcome bugs by giving the programmer the advantage of comparison between old and new version of development.

- We protect the database from inflation by prevent number of transactions from exceeding a predefined constant number via deleting the oldest records to ensure that the database always lite.

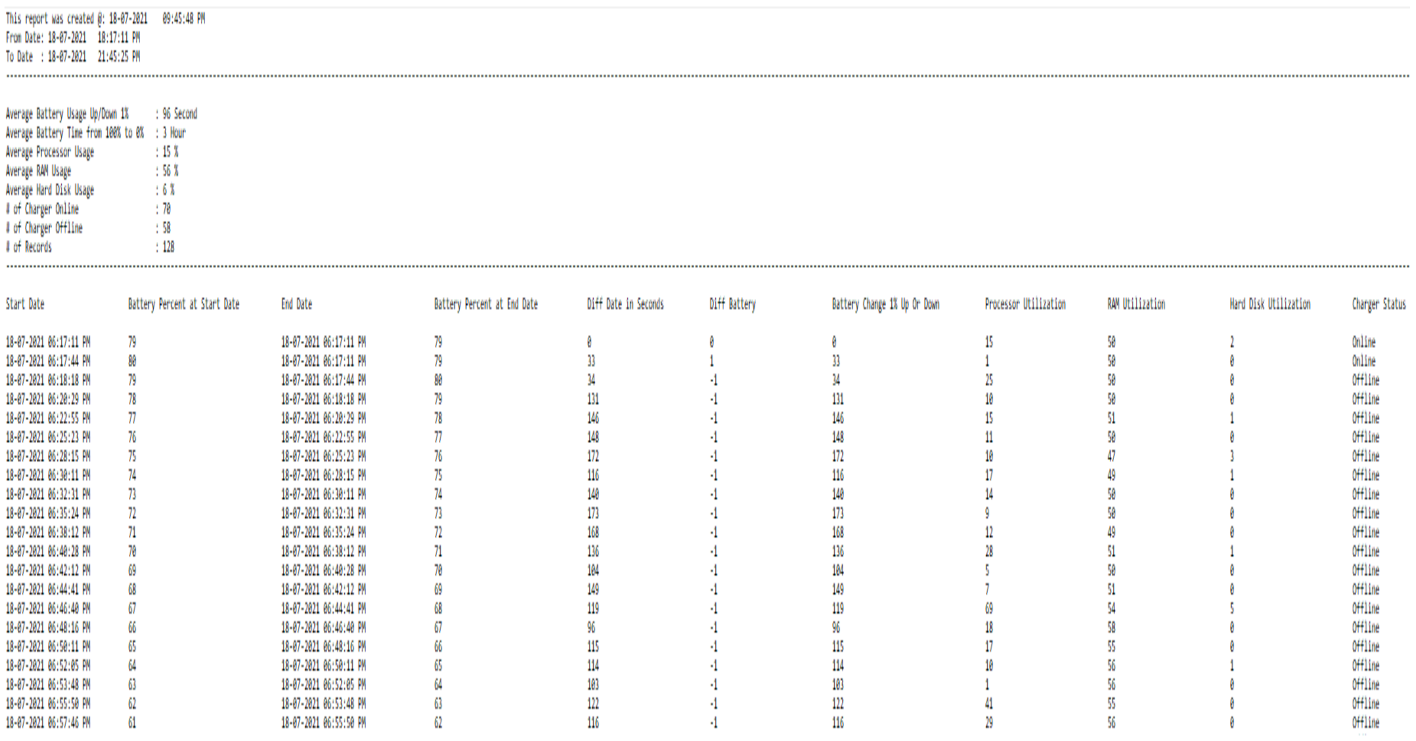
3.2 Functional Requirements

- Send a signal when battery percentage is less than or equal to 20% to disconnect the charger and send another signal when battery percentage is more than or equal to 80% to connect the charger.

- Recording every transaction that battery take it without user intervention, this transaction recorded when battery percentage increase or decrease by 1%

- Ensuring that all transactions in the database, not more than a specific number to minimize resources needed by the application.

- Enable the application's user to generate report contain some analysis about his/her battery's life span such:
    - Average Battery Usage Up/Down 1%
    - Average Battery Time from 100% to 0%
    - Average Processor Usage.
    - Average RAM Usage.
    - Average Hard Disk Usage.
    - # of Charger Online.
    - # of Charger Offline.
    - # of Records.



```
This report was created @: 18-07-2021   09:45:48 PM
From Date: 18-07-2021   18:17:11 PM
To Date  : 18-07-2021   21:45:25 PM
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Average Battery Usage Up/Down 1%      : 96 Second
Average Battery Time from 100% to 0%  : 3 Hour
Average Processor Usage               : 15 %
Average RAM Usage                     : 56 %
Average Hard Disk Usage               : 6 %
# of Charger Online                   : 70
# of Charger Offline                  : 58
# of Records                          : 128
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Start Date              Battery Percent at Start Date   End Date                Battery Percent at End Date   Diff Date in Seconds   Diff Battery   Battery Change 1% Up Or Down   Processor Utilization   RAM Utilization   Hard Disk Utilization   Charger Status

18-07-2021 06:17:11 PM  79                              18-07-2021 06:17:11 PM  79                            0                      0              0                              15                      50                2                       Online
18-07-2021 06:17:44 PM  80                              18-07-2021 06:17:11 PM  79                            33                     1              33                             1                       50                0                       Online
18-07-2021 06:18:18 PM  79                              18-07-2021 06:17:44 PM  80                            34                     -1             34                             25                      50                0                       Offline
18-07-2021 06:20:29 PM  78                              18-07-2021 06:18:18 PM  79                            131                    -1             131                            10                      50                0                       Offline
18-07-2021 06:22:55 PM  77                              18-07-2021 06:20:29 PM  78                            146                    -1             146                            15                      51                1                       Offline
18-07-2021 06:25:23 PM  76                              18-07-2021 06:22:55 PM  77                            148                    -1             148                            11                      50                0                       Offline
18-07-2021 06:28:15 PM  75                              18-07-2021 06:25:23 PM  76                            172                    -1             172                            10                      47                3                       Offline
18-07-2021 06:30:11 PM  74                              18-07-2021 06:28:15 PM  75                            116                    -1             116                            17                      49                1                       Offline
18-07-2021 06:32:31 PM  73                              18-07-2021 06:30:11 PM  74                            140                    -1             140                            14                      50                0                       Offline
18-07-2021 06:35:24 PM  72                              18-07-2021 06:32:31 PM  73                            173                    -1             173                            9                       50                0                       Offline
18-07-2021 06:38:12 PM  71                              18-07-2021 06:35:24 PM  72                            168                    -1             168                            12                      49                0                       Offline
18-07-2021 06:40:28 PM  70                              18-07-2021 06:38:12 PM  71                            136                    -1             136                            28                      51                1                       Offline
18-07-2021 06:42:12 PM  69                              18-07-2021 06:40:28 PM  70                            104                    -1             104                            5                       50                0                       Offline
18-07-2021 06:44:41 PM  68                              18-07-2021 06:42:12 PM  69                            149                    -1             149                            7                       51                0                       Offline
18-07-2021 06:46:40 PM  67                              18-07-2021 06:44:41 PM  68                            119                    -1             119                            69                      54                5                       Offline
18-07-2021 06:48:16 PM  66                              18-07-2021 06:46:40 PM  67                            96                     -1             96                             18                      58                0                       Offline
18-07-2021 06:50:11 PM  65                              18-07-2021 06:48:16 PM  66                            115                    -1             115                            17                      55                0                       Offline
18-07-2021 06:52:05 PM  64                              18-07-2021 06:50:11 PM  65                            114                    -1             114                            10                      56                1                       Offline
18-07-2021 06:53:48 PM  63                              18-07-2021 06:52:05 PM  64                            103                    -1             103                            1                       56                0                       Offline
18-07-2021 06:55:50 PM  62                              18-07-2021 06:53:48 PM  63                            122                    -1             122                            41                      55                0                       Offline
18-07-2021 06:57:46 PM  61                              18-07-2021 06:55:50 PM  62                            116                    -1             116                            29                      56                0                       Offline
```

Figure 3.1: Log File

3.3 Non - Functional Requirements

- Enable the application's user to hide the application, this feature MUST freeze every non-functional operation and make the application work in the background with the usage of minimum resources.

- Enable the application's user to see every transaction in real time without needing of restarting the application.

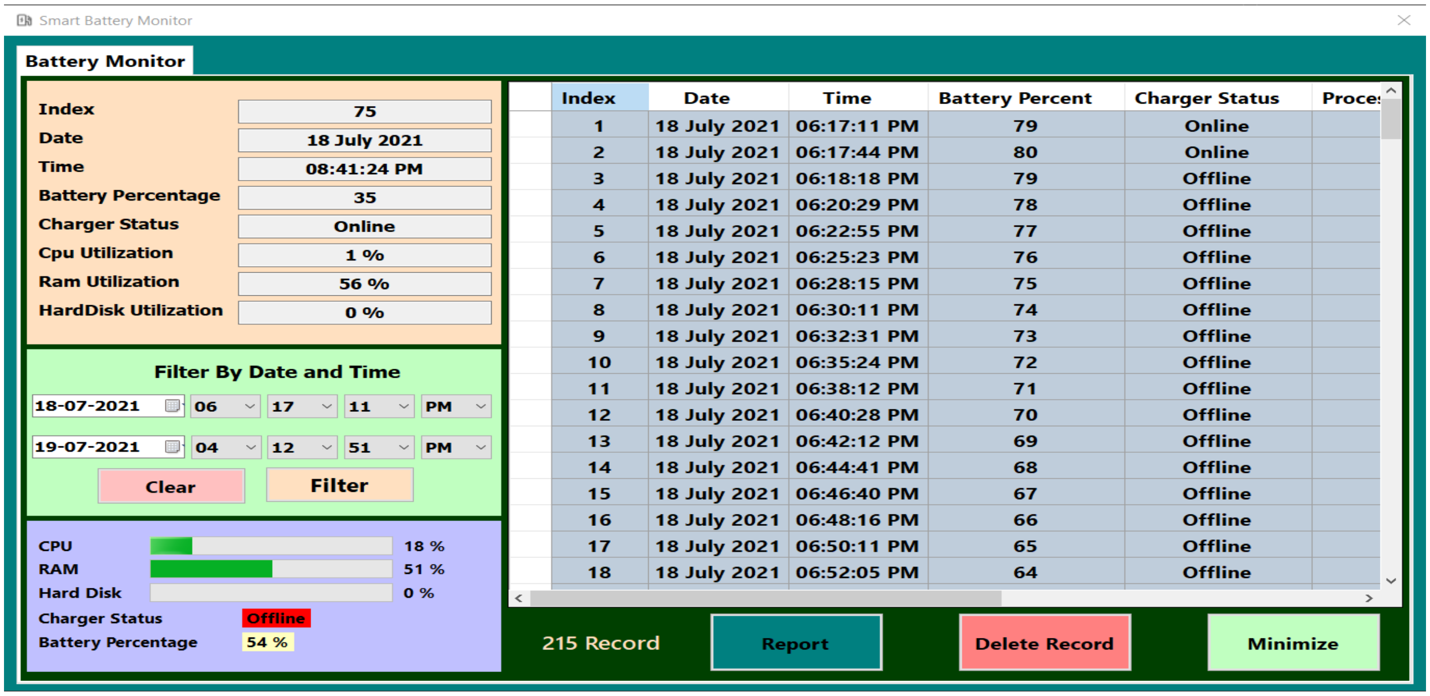
- Enable the application's user to filter transactions depend on date and time.

- Filter boxes MUST take their values from the earliest time and latest time from the database to facilitate the operation of filtration on the application's users and to protect users from filters depend on date or time out of range.

- Filter boxes take their values in two situations, the first one is when a new transaction is added to the database, and the second one is when the user clicks on the Clear button in the Filter section.

- Enable the application's user to make live monitoring for utilization of Processor, RAM, and Hard disk.

- Users can watch battery percentage, charger status, and number of records in data grid view in real time.



Index	75
Date	18 July 2021
Time	08:41:24 PM
Battery Percentage	35
Charger Status	Online
Cpu Utilization	1 %
Ram Utilization	56 %
HardDisk Utilization	0 %

18-07-2021 06 17 11 PM

19-07-2021 04 12 51 PM

CPU	18 %
RAM	51 %
Hard Disk	0 %
Charger Status	Offline
Battery Percentage	54 %

Index	Date	Time	Battery Percent	Charger Status	Proce
1	18 July 2021	06:17:11 PM	79	Online	
2	18 July 2021	06:17:44 PM	80	Online	
3	18 July 2021	06:18:18 PM	79	Offline	
4	18 July 2021	06:20:29 PM	78	Offline	
5	18 July 2021	06:22:55 PM	77	Offline	
6	18 July 2021	06:25:23 PM	76	Offline	
7	18 July 2021	06:28:15 PM	75	Offline	
8	18 July 2021	06:30:11 PM	74	Offline	
9	18 July 2021	06:32:31 PM	73	Offline	
10	18 July 2021	06:35:24 PM	72	Offline	
11	18 July 2021	06:38:12 PM	71	Offline	
12	18 July 2021	06:40:28 PM	70	Offline	
13	18 July 2021	06:42:12 PM	69	Offline	
14	18 July 2021	06:44:41 PM	68	Offline	
15	18 July 2021	06:46:40 PM	67	Offline	
16	18 July 2021	06:48:16 PM	66	Offline	
17	18 July 2021	06:50:11 PM	65	Offline	
18	18 July 2021	06:52:05 PM	64	Offline	

215 Record

Figure 3.2: User Interface

3.4.1 Finite State Machine

- In this section we will discuss FSM for the main purpose of the code, then we will discuss the code map for each important class, that is because the whole code was built using multithreading programming concepts and there are no sequential operations [10]
- the following finite state machine discusses the main objective of code which is checking battery percentage at every change in battery percentage up or down by 1%.
- If battery percentage reaches 20% or less, the Hardware device MUST connect the charger to the AC line, and if battery percentage reaches 80% or more, the Hardware device MUST disconnect the charger from the AC line.
- this process of checking battery percentage is eternal while the application is running in the foreground or in the background.
- There is no effect on the user's work because this operation is done by a thread that created in run time.

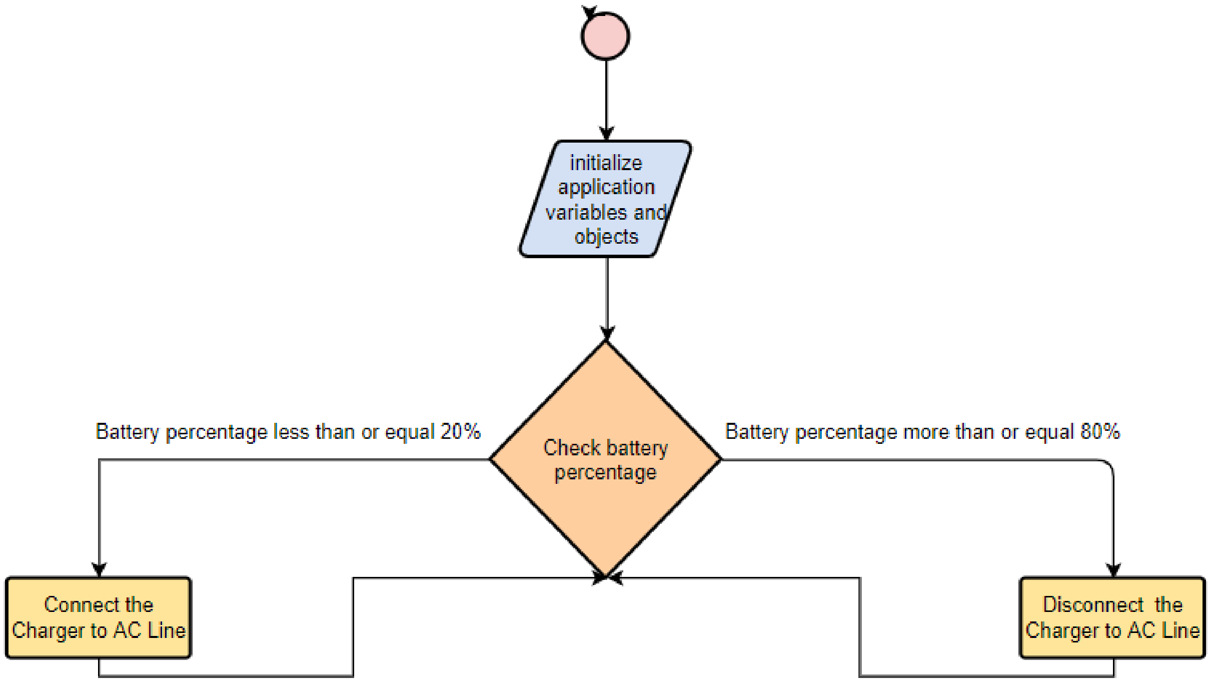

initialize application variables and objects
Check battery percentage
Battery percentage less than or equal 20%
Battery percentage more than or equal 80%
Connect the Charger to AC Line
Disconnect the Charger to AC Line

Figure 3.3: Finite State Machine

- **Let’s take an overview of the project architecture.**

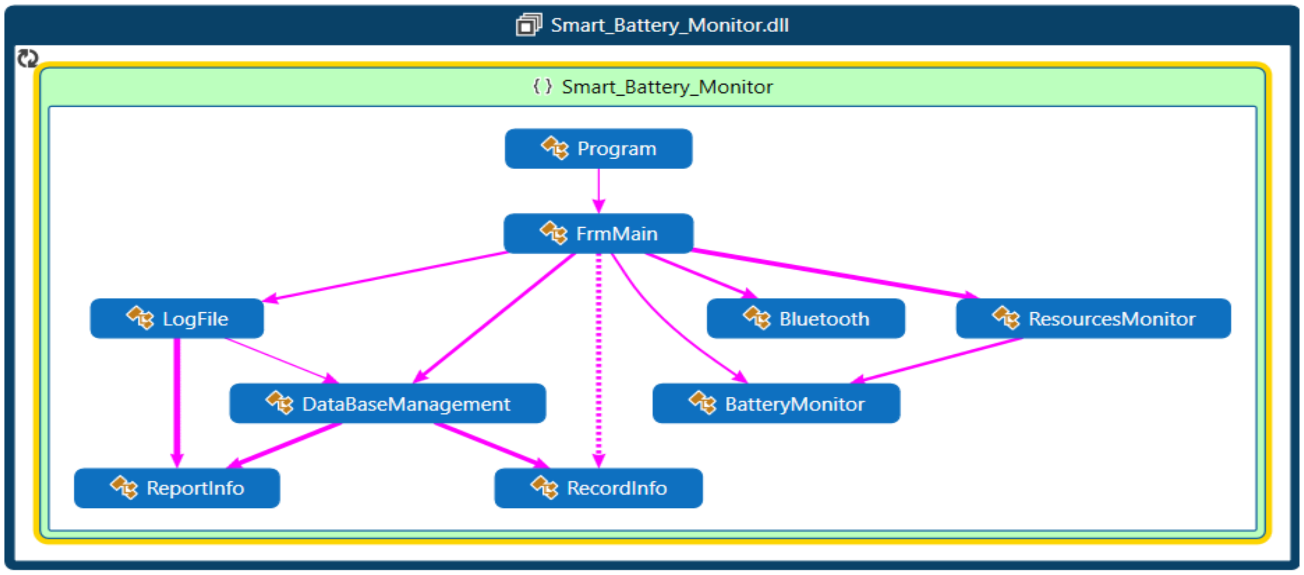


Figure 3.4: Project Architecture

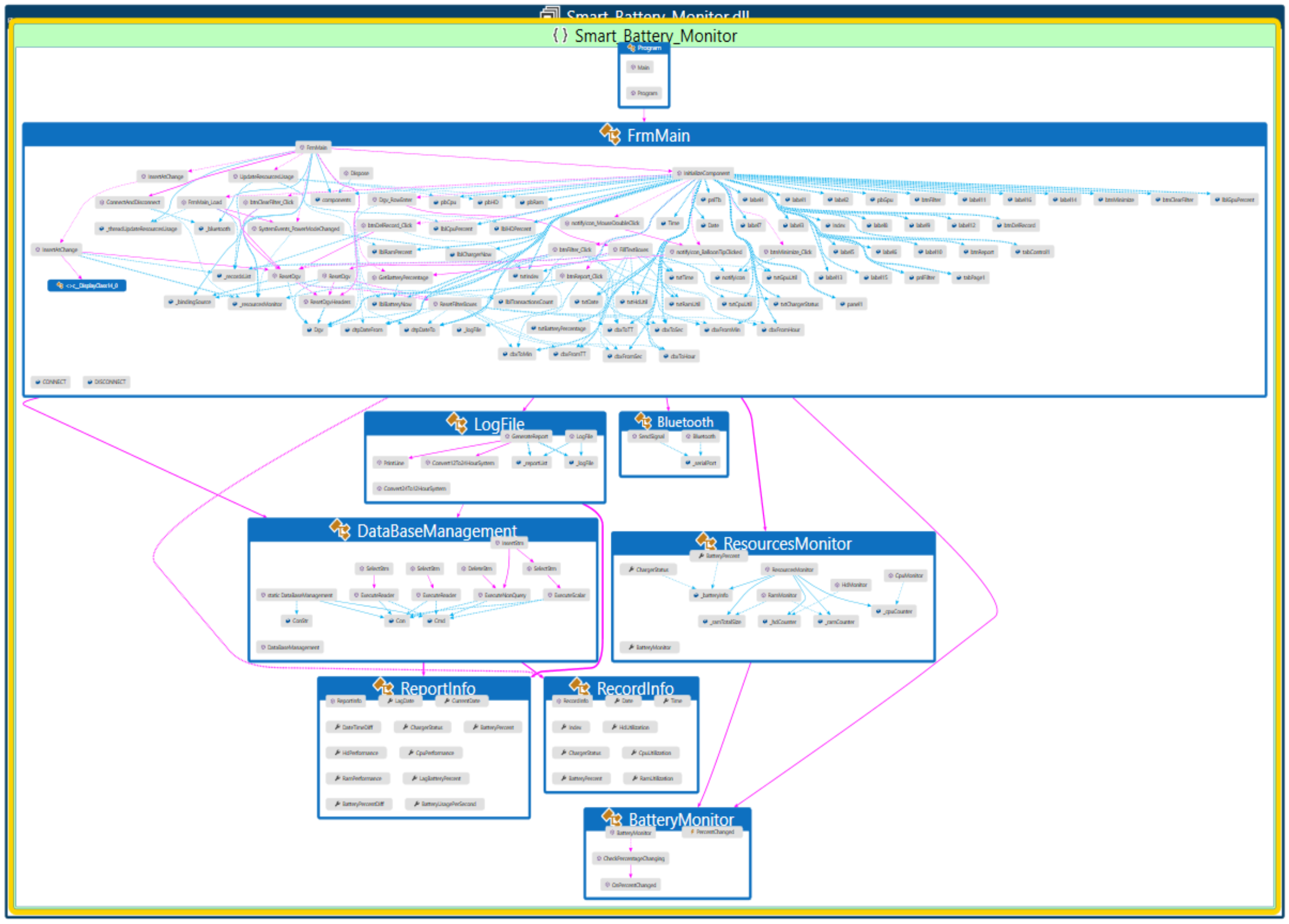


Figure 3.5: Project Architecture in detail

3.4.2 The architecture of DataBaseManagement Class

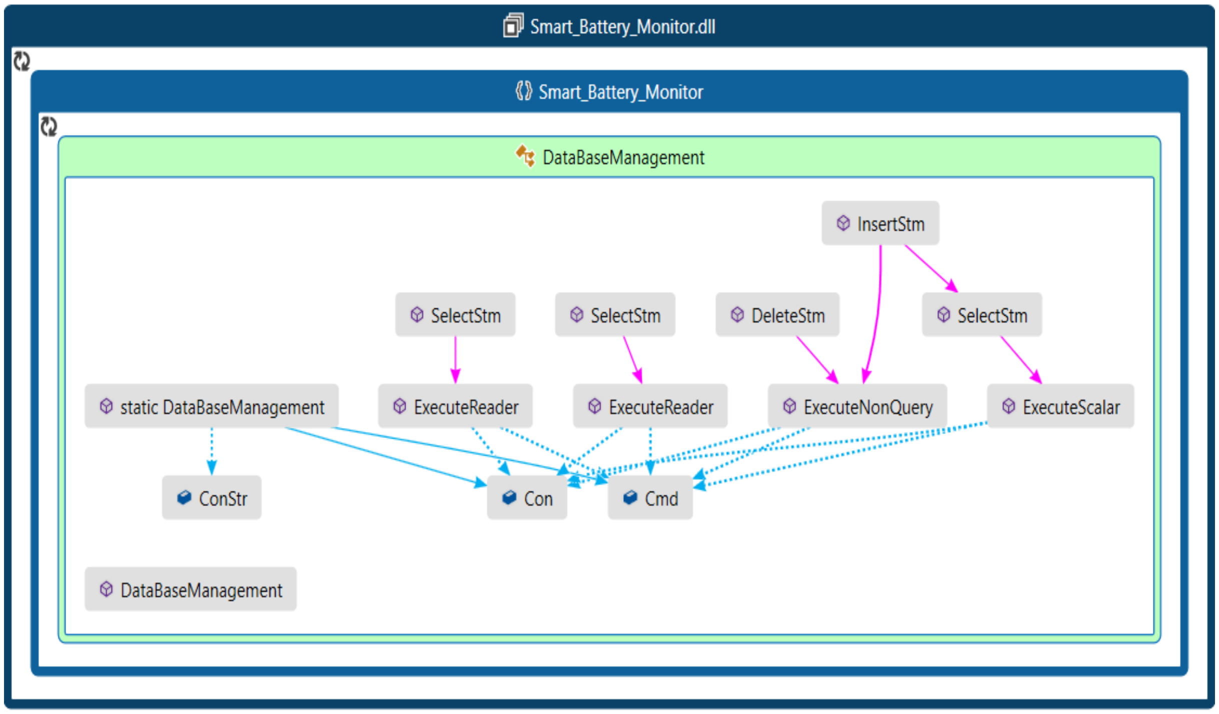


Figure 3.6.1: DataBaseManagement Class Architecture in detail

- This class is consisting of two main parts, firstly: Methods that interact directly with the database such as (ExecuteReader, ExecuteNonQuery, ExecuteScalarand)
- Other sections are methods that contain SQL statements and pass these statements to methods that are in section one.
- In this section, we use the concept of functional programming which “functions are treated as first-class citizens, meaning that they can be bound to names (including local identifiers), passed as arguments, and returned from other functions, just as any other data type can. This allows programs to be written in a declarative and composable style, where small functions are combined in a modular manner” [13]
- Methods like SelectStm, DeleteStm, and InsertStm were written using the concept of functional programming, to facilitate the operation of developing from any other developer that wants to add a new feature and wants to make some new processes on the database.

3.4.3 The architecture of ResourcesMonitor Class

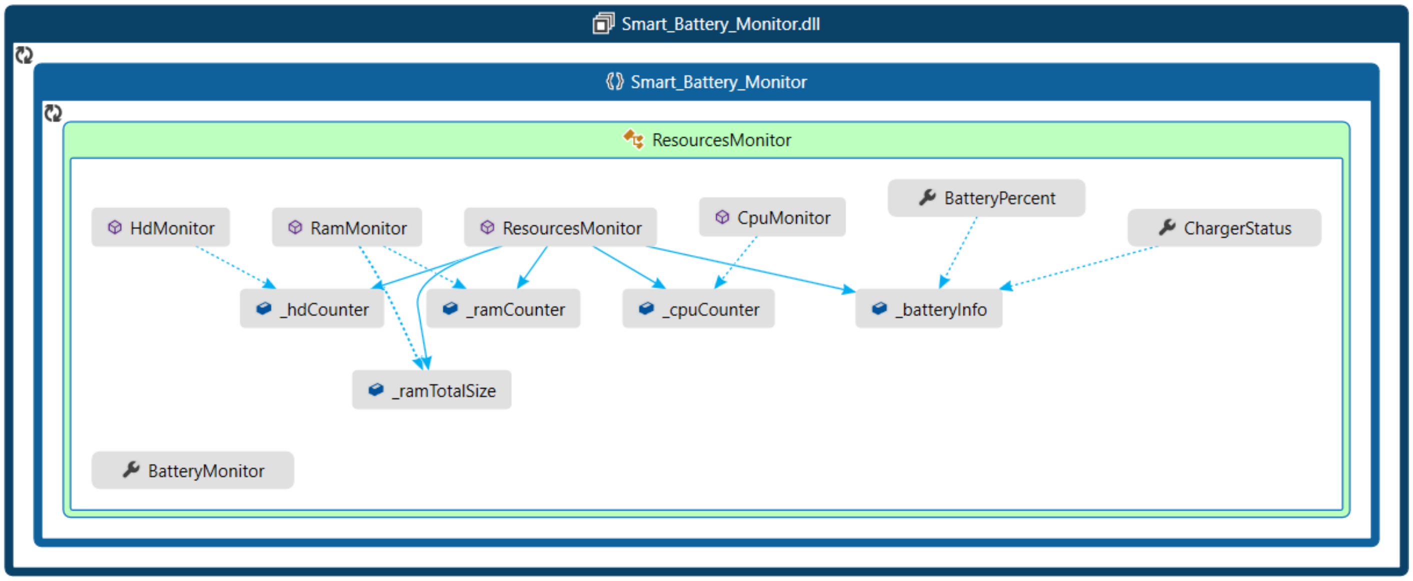


Figure 3.6.2: ResourcesMonitor Class Architecture in detail

- This class is responsible for monitoring performance of all resources (Processor, RAM, and Hard Disk) [11]
- In the future development cycle, we will try to monitor the Graphics processing unit also.
- This class is used with some other classes like FrmMain class to bring values of resources utilization that used in database recording system and make live streaming that feeds progress bar in the main form.

3.4.4 The architecture of LogFile Class

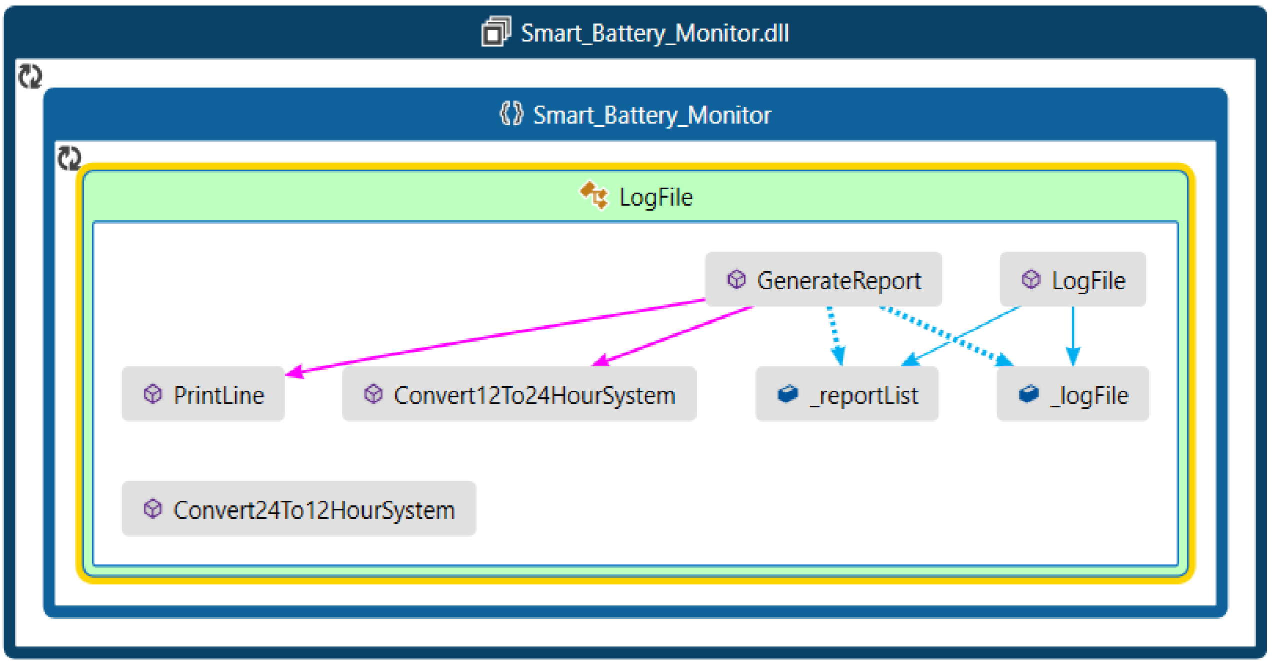


Figure 3.6.3: LogFile Class Architecture in detail

- This class is responsible for generate log file that contains all transactions that user selected using Filter feature.
- Furthermore, class can calculate some statistical values that expects Average Battery Time from 100% to 0% depend on the Average Battery Usage Up/Down 1%
  These values calculated depend on "VIEW" in database called "vWReport"
- We used some helper methods like Convert12To24HourSystem and PrintLine that facilitate our works.

3.4.5 The architecture of BatteryMonitor Class

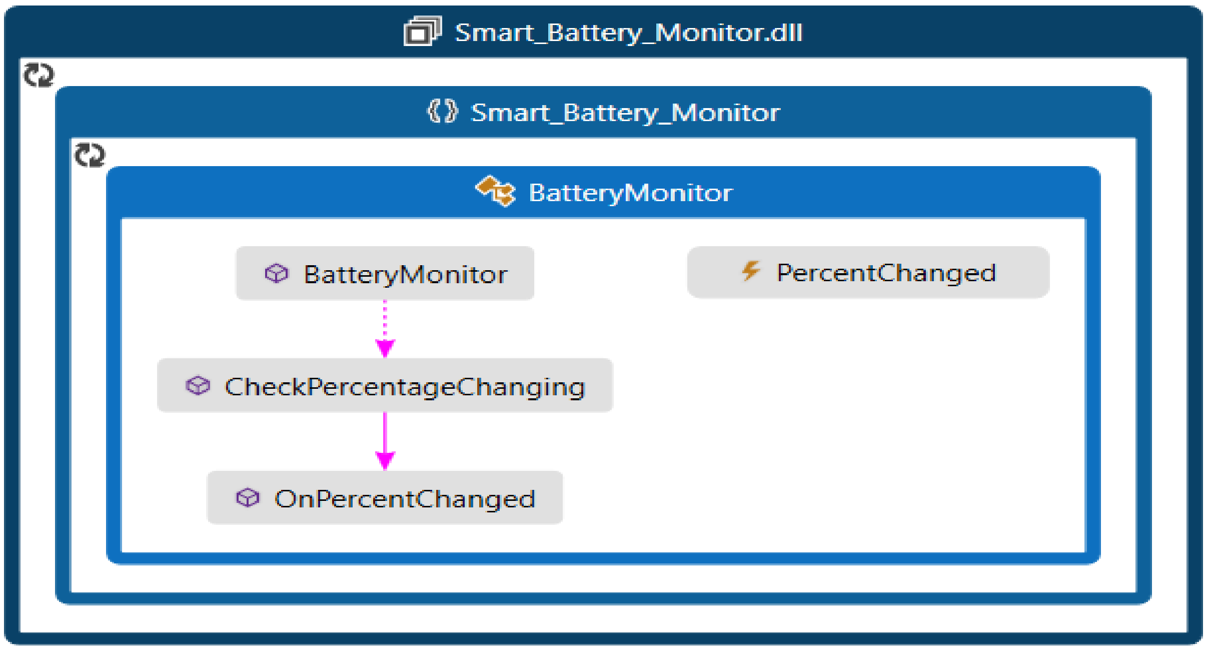


Figure 3.6.4: BatteryMonitor Class Architecture in detail

- This class has an important event that more than one method depends on, this event is called PercentChange, it's raised when battery percentage changes by 1% up or down [7]

- There is a thread of type Background called threadCheckPercentageChanging, this thread invokes a method that looping till battery percent changes by 1%, this method is an event handler for the PercentChange event [8]

- The method that's responsible for raising PercentChange event, it's called OnPercentChanged and it's from type virtual method.

3.4.6 The architecture of Bluetooth Class

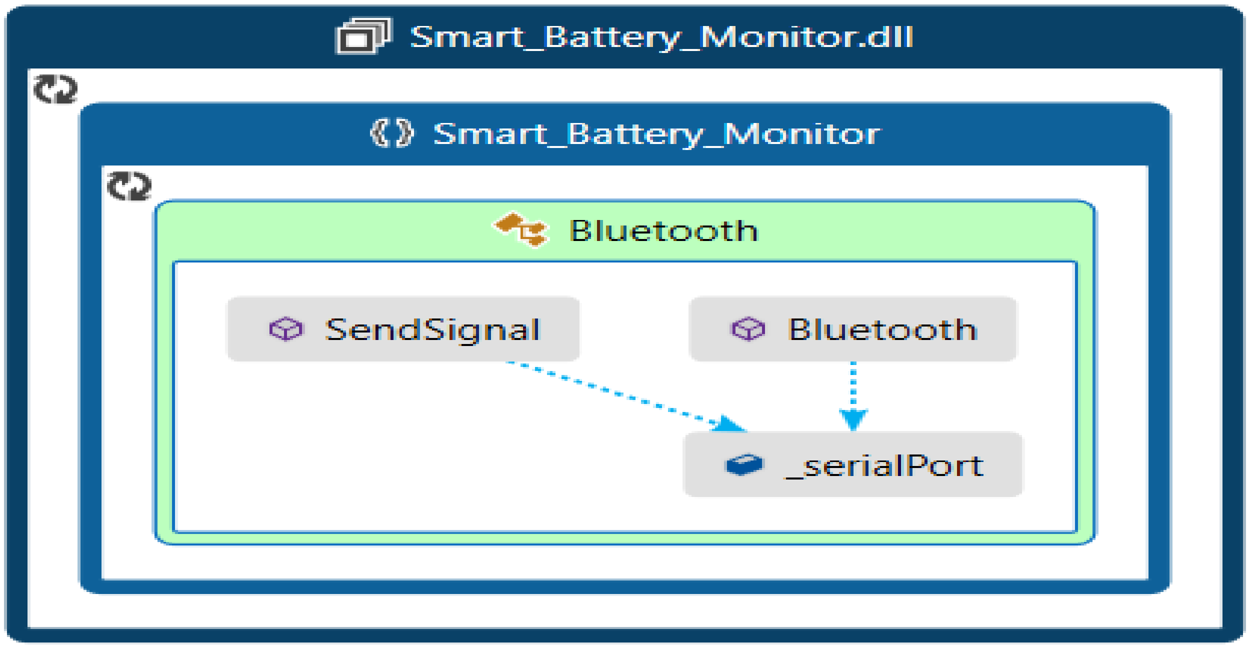


Figure 3.6.5: Bluetooth Class Architecture in detail

- This class is responsible for communication with Bluetooth module via Bluetooth protocol, it does the following functions:
    - Establish the connection:
        - First: we automate choosing the right port process by putting all the ports in an array, then we examine each element in the array if the communication is established well.
        - We break the loop; else we continue examination process till the end of the array.
        - If we reach that port in the array and the connection has not established yet, we display an error message to warn the user to check the hardware.
        - Note that if we don't do this step the code won't work well, and if that happened it will be by luck.
    - send a string via Bluetooth:
        - We send a string to the Bluetooth module within try catch block to ensure that all packets of message are send well to the target port, then after every communication process we close the port.
        - We send either "1" to turn on charger and "0" to turn off charger.

3.4.7 The architecture of FrmMain Class

- This class is a partial class, that is mean that class is divided in two files, one for designing the UI graphics and other for the logical code that interact with controls in this UI.

- The logical code is divided into 8 parts, but in this section, we will discuss the most important regions.

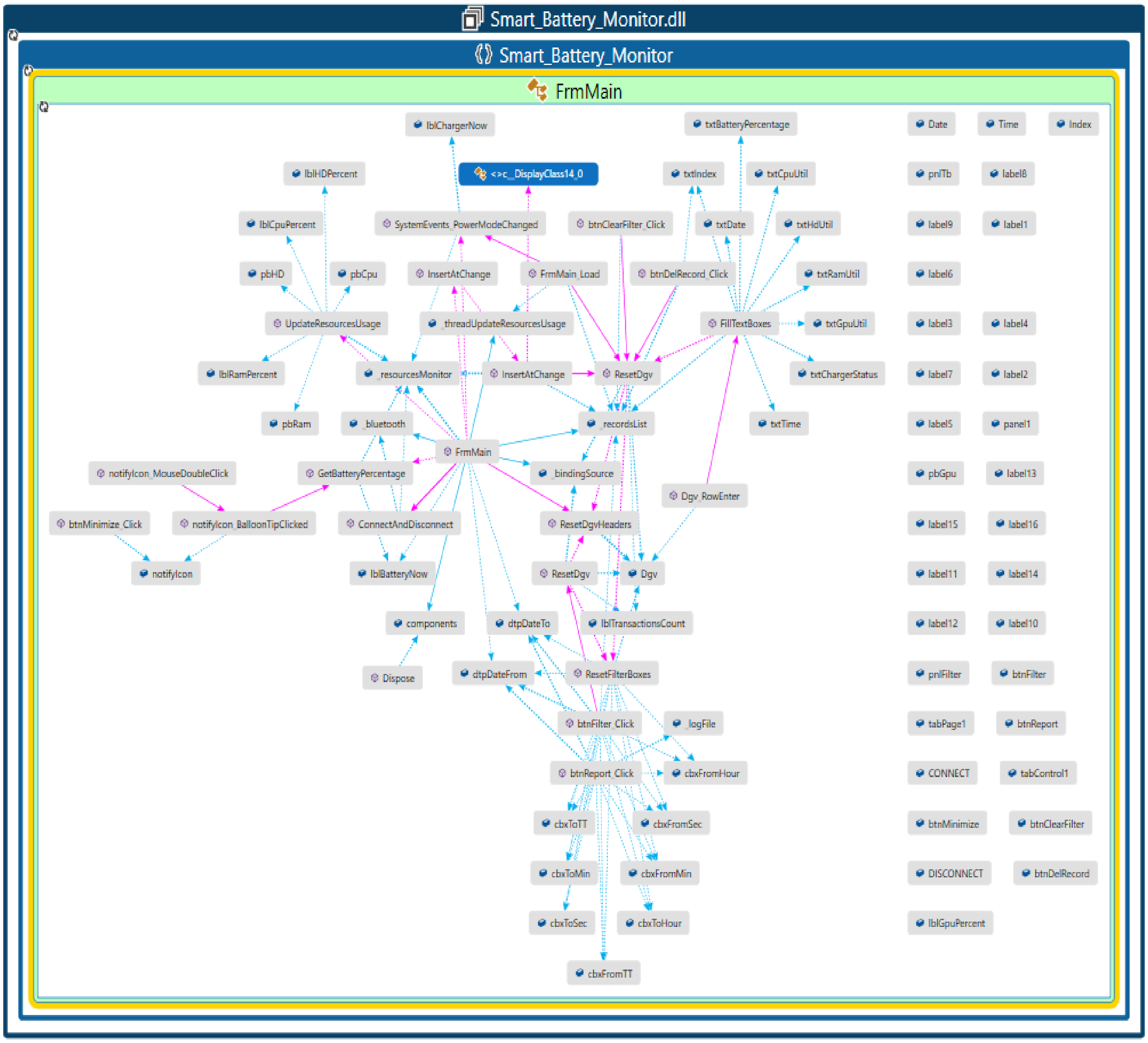


Figure 3.6.6: FrmMain Class Architecture in detail

- **notifyIcon region:**
  - This region is responsible for minimizing and maximizing the application.
  - This region contains the Minimize button that hides the main form and stops all non-functional operations like monitoring resources performance in real-time and gets battery percentage and charger status.
- **databaseHandling region:**
  - This section has some methods that directly pass needed info to another method in DataBaseManagement Class that insert new transaction in database if battery percentage up or down by 1%
- **batteryPercentage region:**
  - In this region, there is one of the most important Events in all code called "SystemEvents_PowerModeChanged"
    this event is responsible for observate battery's percentage, this event raised if this percentage change by 1% up or down.
  - This event is a base for many methods that depend on it to take its action and execute its logic code like InsertAtChange method that insert new transaction depend on raising from this event.
- **buttonsEvents region:**
  - this region handles the user's action with buttons and data grid view, in this section user can click on many buttons like:
    - Filter: that apply filter on records in data grid view.
    - Clear: clear filter that applied.
    - Delete Record: enable user to delete any record he/she selected from data grid view.
    - Report: generate a report to the user that contains some calculated values that expect the battery time from 100% to 0% and the average time that battery needed to change its percentage by 1%, furthermore, the report contains all transactions that happened from starting time to the ending time that the user selected before generating this report.
      in this button action, we passed some important values to LogFile Class that contains some LINQ statement that helps in generating report and make this operation fast and easy to maintain in the future [2]

- **resetControls region:**
    - This region contains one of the most important codes that reset the filter combo-boxes and refreshes the data grid view after each instruction of insertion or deletion.

    - Importunacy is not in the operation of resetting controls but in the way that these controls can be resetting.

    - To make any processes on any UI controls, it MUST use the UI thread, but in this case, there are many threads (pool of threads) in the application, and there is no way to verify that the thread that trying to process the UI controls is the UI thread [9]

    - So, we are checking before any operation of processing on controls that the thread that is used is the UI thread or not, and for this, we use the InvokeRequired built-in method that returns a Boolean type.

    - If this method returned true, that is mean that the UI thread MUST be invoked and if the method returned false, that is mean that there is no need to invoke the UI thread because it is already invoked.

    - Because we are using the InvokeRequired method, we use the concept of local methods to reduce the amount of code written and to prevent code redundancy.

Chapter 4: Conclusion and Future Work

4.1 CONCLUSION:

- The battery drain problem is still a huge problem that face the world nowadays, not only on the Earth but also on the outer space, so we aim to organize the charging process to get most of the battery and increase its lifetime.

- As computer Science students, our main job is solving real problem and facilitate lives of people, so after a lot of brainstorming we conclude that the highest priority went to the researchers who use their laptop almost time of their daily repeated routine, and because of bad habits the life time of the battery decreased and the work of researchers lost when this is happened many times the laptop depend on the AC line not on the battery, this is dangerous, especially when they do a research and suddenly the get interrupted by power failure without saving their work.

- So we decide to work hard to overcome this problem by making an embedded handheld device that automate charging process, to turn ON charger on specific charging threshold < 20%, and turn OFF charger on specific discharging threshold >80 , by doing this the life time of the battery will double about two times.

- Our device is the output of integration of Software and Hardware:

- We made a C# application that connect the Embedded Bluetooth module in the laptop to send the battery percentage to our device, also the user could see his battery status at the run time within an interval according to his desire, he could also analyze his battery performance with numbers and statistics which he could find in a log file, this led to expecting when this battery will drain.

- The hardware consist of a microcontroller that communicate with Bluetooth module using UART protocol and this Bluetooth module communicate with laptop via Bluetooth protocol and depend on the battery percentage we turn on the AC line 220v by sending the signal to a Relay through opto-transistor to isolate the brain of the device from damage, we also protect our device from overcurrent by putting a fuse, and from EMF spikes by putting a parallel diode with the Relay, we make this circuit in a breadboard as a prototype.

- To convert this prototype into a real product we make a schematic design and then make a Printed Circuit Board designs and make these files ready to send to any manufacturer.

4.2 Future work:

- We aim to convert this prototype into a real product, this lead to minimize the cost for each one device.
- Also, our goal is to integrate machine learning algorithms with the statistics that we obtain from our application to learn, analyze, and then predict battery performance within future interval as soon as the expected date of battery drain that lead us to replace it.
- There is another important point that we want to generalize this product by making this device available for Android and Apple smart phones, tablets, iPad, and IOS laptops by making one general page for each user and the user could register any device he owns.
- He also has a dashboard that contain information about every device we do this step by converting this native C# application into a cross platform version which generate an output available for any platform.
- The last point we seek to make a network between users to know battery types, their lifetime, and vendors to share the experience and get the most of social network to reach the best result and all users will be satisfied.

References

[1] Conrad, James & Dean, Alexander. (2011). Embedded Systems: An Introduction Using the Renesas RX62N Microcontroller.

[2] Troelsen A., Japikse P. (2021) LINQ to Objects. In: Pro C# 9 with .NET 5. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-6939-8_13

[3] S. (2021). Software Engineering (9th ed.). PEARSON EDUCATION.

[4] Fezari, Mohamed. (2018). Introduction to ATtiny85

[5] PC817 Datasheet.

[6] Hi-link Datasheet.

[7] http://www.blackwasp.co.uk/DetectPowerEvents.aspx

[8] https://www.onlinebuff.com/article_understand-monitor-vs-mutex-vs-semaphore-vs-semaphoreslim-onlinebuff_60.html

[9] https://stackoverflow.com/questions/142003/cross-thread-operation-not-valid-control-accessed-from-a-thread-other-than-the UI_Thread/

[10] https://www.youtube.com/watch?v=UI6lqHOVHic

[11] https://www.c-sharpcorner.com/UploadFile/29d7e0/get-the-processor-details-of-your-system-in-windows-form/

[12] https://en.wikiversity.org/wiki/Plan-driven_software_development#cite_note-1

[13] https://en.wikipedia.org/wiki/Functional_programming

[14] https://www.quora.com/How-should-we-charge-mobile-phone-battery

[15] https://www.samsung.com/nz/support/mobile-devices/tips-for-battery-charging-and-how-to-make-your-battery-last-longer/