IMPLEMENTATION OF ALGORITHM TO DETERMINE ESTIMATED STATE OF CHARGE,TRAVELLING TIME AND DISTANCE TO COVER FOR ELECTRICALLY POWERED VEHICLE

A Thesis Submitted to the Department of Electrical and Electronic Engineering of BRAC University

By

Aparna Saha- ID 13121079

Murshida Anjum- ID 12221087

Dewan Abu Md. Arefin- ID 12221019

Mir RayhanuzzamanTaif- ID 12221032

Supervised by

Dr. A. K. M. Abdul Malek Azad
Professor
Department of Electrical and Electronic Engineering
BRAC University, Dhaka.

In partial fulfillment of the requirements for the degree of Bachelor of Science in Electrical and Electronic Engineering

Summer 2016
BRAC University, Dhaka

DECLARATION

We hereby declare that our research titled “Implementation of algorithm to determine estimated state of charge, travelling time and distance to cover for electrically powered vehicle”, a thesis submitted to the Department of Electrical and Electronics Engineering of BRAC University in partial fulfillment of the Bachelors of Science in Electrical and Electronics Engineering, is our own work has not been presented elsewherefor assessment. The materials collected from other sources have also been acknowledged here.

Signature of Supervisor

Signature of Authors

....

Aparna Saha

....

Dr. A. K. M. Abdul Malek Azad

Professor

Department of

Electrical & Electronic Engineering

BRAC University

....

Murshida Anjum

....

Mir RayhanuzzamanTaif

....

Dewan Abu Md. Arefin

ACKNOWLEDGEMENT

We are thankful to our thesis supervisor Dr. A.K.M. Abdul Malek Azad, Professor, Department of Electrical and Electronic Engineering (EEE), BRAC University, for giving us direction and providing valuable feedbacks throughout our thesis work to complete our thesis. Regards to our Project Engineer, Sheri Jahan Chowdhury and Research Engineer Jaber Al Rashid and Ataur Rahman for their support throughout the thesis time span. We are thankful to BRAC University for providing us the necessary equipment for the completion of this project and funding this project undertaken by Control and Application Research Centre (CARC).

ABSTRACT

Electrically assisted with PV panel supported vehicle is developing a market for the automobile manufacturers as it is a promising innovation in case of reducing CO_2 emissions and fossil fuel consumption. The number of electric vehicles on the road is not up to the expectation due to some constraints which includes limited driving range that depends on many factors such as load type, driving style and road condition. The paper describes the development and implementation of an algorithm to produce and install a useful electrical device on the electric vehicle which will show State of Charge (SOC), remaining discharging time of battery and available travel distance correspond to SOC on the LCD display by using microcontroller along with necessary external electrical connections with required components. This electrical device provides the features for the puller to take the decision about charging the batteries used in the vehicle in time which helps to avoid unwanted risky occurrence in the middle of the road due to limited driving range. This project include a brief description of the method used to develop the algorithm and also describes the data obtained from the field test which assures performance, efficiency and feasibility of the electrical device.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

1.1 Introduction to Development of Electric Vehicle

In the world of automotive industry, Electric Vehicle (EV) is a relatively new concept which is achieving more popularity day by day because of its high level efficiency, eco-friendly behavior, low level of noise pollution and regenerative multiple energy sources [1]. Conventional fossil fuel vehicles inject a lot of carbon emissions into our natural atmosphere that creates pollution and greenhouse gases [2]. In order to mitigate the environmental degradation, an electric vehicle is a great step forward such as the use of EVs is viewed as an attractive option to the transportation sector which can reduce CO_2 emission, consumption of fossil fuel by using a clean and renewable energy source like solar energy [3]. Electric vehicle can harvest energy as it has a regenerative braking system, otherwise energy can be lost when heat is produced during the braking situation. Thus, EVs are more energy efficientcompared to all conventional fossil fuel vehicles.However, the number of electric vehicles on the road is not up to expectations than other conventional fossil fuel vehicles due to its driving range, long charging time, low battery life time and lack of charging infrastructures [4]. Thus, the development of electric vehicles provide a potential partial solution to world-wide concerns related to environmental pollution and energy security. In our project, we are developing an electrical device for the electric vehicle which will display SOC (State of Charge) of the batteriesand the whole system will be developed by the microcontroller. Besides, we will also display the time and covered distance where time will provide us how much time is remaining to discharge batteries completely and in this time how much distance we can travel as this electrical device gives the information of the charging status of the batteries.

1.2 Background and Motivation

Such a mass gathered country like Bangladesh, manually driven vehicle like rickshaw or van is one of the most chosen option for earning the livelihood for the majority of poor people. In the history of Bangladesh, these types of vehicles will show their existence even in the oldest page of history. The first cycle rickshaw in Bangladesh appeared in the 1930's [5]. After the invention of first cycle rickshaw, its being popular that the total number of rickshaws in the city is at least two and a half times that of the registered ones and accordingly, the city had at least 280,000 rickshaws in 2000 [6]. Recently, motor driven vehicles are getting large in number and earning positive response from the people who used to choose these vehicles for livelihood or who like to ride on these. Rickshaws and vans are becoming motorized for the purpose of its effectiveness and provides a non-polluting silent transport system which is also a very cost effective vehicle. In addition, these motor control vehicle need driver but driver does not need any qualification to become the rickshaw puller. Anyone can choose to be a driver of these vehicles that does not need any road permission or even license. Since the motor controlled one does not require any physical labor, it is the best option for the person who choose to become rickshaw puller whereas the existing traditional rickshaws are very poorly designed so that it takes a heavy toll on the health of a rickshaw puller. Furthermore, the rickshaw puller has to work very hard while climbing even a slight slope as the gearing and mechanical advantage of the pedal is poor. Hence a common sight is of rickshaw puller getting down and pulling on foot the rickshaw with passengers and the speed of motor controlled rickshaws is also better than the manually driven rickshaws. For this reason, the motorized vehicles are loudly showing its presence in the rural areas and most of people are getting interested on this vehicle day by day.Such as Figure 1.1 shows a comparison between traditional rickshaw and CARC prototype of electrically assisted rickshaw vans.

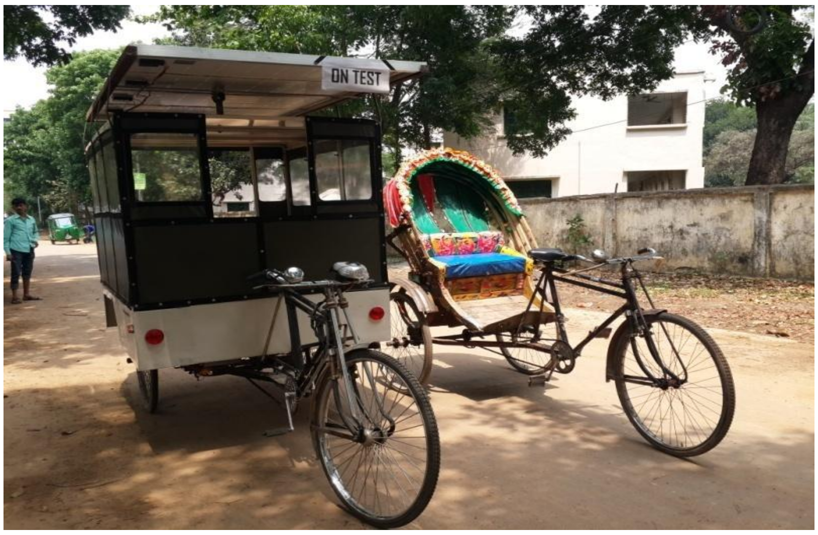


Figure 1.1: Motor Controlled Rickshaw Vans VS Traditional Rickshaw

In this new era of technology intellectual works have given more priority than physical works as physical labor are replacing by the intelligence of human brain. In order to keep pace with this fastest and latest world, our country is also facing the replacement of physical works. Smart technology has entered in almost every sector, hence the transportation system could not escape from the technological advancement. In the last few decades technology has touched our transport system in such a dramatic way that our old transport system like rickshaws, vans, etc. are also getting furnished by the technology. These types of vehicles are getting modernized to cope up with the upgrading world. Among all types of vehicle, people use rickshaws mostly to travel short distances that has the highest amount of popularity. Therefore, modification with the update of this vehicle is highly required to reduce the effect of physical work, hence this project developing a display for electric vehicle. To meet with the success of this project, we four people join together to develop an algorithm for the most popular vehicle in our country to measure some of its important information such as SOC, travelling time and coverable distance. Basically we become motivated due to the problem of not having this device in the existing electric vehicle like drivers cannot know when the battery charge will be finished. In the middle of the road, the vehicle may not move further because the SOC is 0%. Hence, both passengers and drivers face huge problems for not knowing these valuable information like SOC, time remaining to discharge batteries and preferable distance to cover. We potentially believe that our project will be helpful for both passengers and drivers as they will come to know that how much charge is remaining to discharge the batteries. Thus, the driver can charge up the batteries in time which will bring positive impact to people of the society who runs and ride these vehicles; it will also motivate other individual to come up with the further improvement of this project.

1.3 Objective of our project

The purpose of our project is to produce and install a useful electrical device in the electric vehicle to help drivers with required necessary battery information that will help to take precautions regarding how long they can drive the vehicle, and when his vehicle needs to be charged which is driven by the financially needy van pullers. Using microcontroller with other required components we make a display for the electric vehicle which will help the rickshaw van pullers by showing remaining charge, time and in this time how long distance they can travel, so this make the life of the van pullers easier and smooth than before. The following informationwill be displayed in the LCD Display of the electrical device that will be installed in the electrically assisted vehicle.

- **The SOC (State of Charge) of the electric vehicle:** It helps the van pullers to know about the charging state of the van and thus they can charge the battery of the vehicle when there shows scarcity of charge in the display.
- **The time can be travelled with the unused stored charge of the battery:** It is very essential information for the puller that after the particular time which shows in the display he needs to stop the vehicle. If he does not have the information the vehicle might stop at the midway of his destination which is a financial loss for his daily earning.
- **The distance can be travelled with the remaining charge of the battery:** This information allows the puller to take the decision about how much further the vehicle can be driven before discharging the batteries completely which is viewed by the remaining charge in the display and stop the vehicle in the time as well as charge up the batteries.

In a nutshell our main purpose is to provide the very crucial information about the charging state of the battery of the vehicle, how much time and distance can be driven by the pullers. Lastly, our project is affordable for the van pullers who live below the poverty line and bring immense benefits to both passengers riding on the vehicle as well as pullers driving the vehicle.

1.4 Thesis Overview

We will explain our whole thesis work in eight chapters where chapter 1 contains the introduction which includes mainly motivation and objective of our project. In chapter 2, we discusses about the overview of our whole project where we are giving system block diagram and circuit diagram. The chapter 3 contains burning process with hardware and software components with all working process and existing features. The chapter 4 gives the field test data for developing the algorithm with necessary curves and equation. The chapter 5 is about the current sensing device with its working principle, features and necessary curves correspond to practical data. The chapter 6 mainly includes the algorithm of our project which will show SOC, remaining travelling time and distance on the LCD display. The chapter 7 contains final output on the LCD display and chapter 8 is about the conclusion where mentions drawbacks of our project, future plan for improvement of our project.

Chapter2
Overview of the project

2.1 Scope and application area of the project

For the application of our project, we basically focused on IEEE Infrastructure Development Company Limited (IDCOL) funded projects on developed solar assisted electric vehicles like human hauler, Ambulance and Cargo hauler (C5). Therefore, we are going to use our electrical device on these three types of electric vehicles which will display some important information such as SOC, remaining travelling time and projected distance coverage of the battery. The solar electric ambulance van is a fully throttle controlled electric vehicle as well as incorporated with torque sensor pedal and light weighted steel body that can carry patient along with two attendants. As it is very cost effective and easy to construct, so the solar powered ambulance brings a great change for the people living in the rural areas those whom cannot afford the city ambulances [7]. Our project is to incorporate new technological innovative electrical device to assist drivers of the electric vehicle. The puller and the people of the rural areas will be benefitted from our project by knowing the coverable distance with the current state of charge of batteries and they can decide within the remaining charge they can travel how much further and find out whether they can reach destination or notwhen transporting a patient, otherwise discharge the battery completely in the middle of the road. If delay happens on the road can deteriorate health condition of patients further, who need immediate medical services where city ambulance cannot reach because of the narrow lanes. In addition, our project also benefits the puller of human hauler and cargo hauler (C5) where human hauler transport passengers and C5 transport goods from one place to another. If the electrical device is installed in all electric vehicle, they will be able to know current status of SOC, remaining time and coverable distance which helps the puller to charge up the batteries on time rather than discharging the battery completely in the middle of the road that will be harmful for the driver as well as for the passengers. The figure 2.1 denotes the three application area of our project.

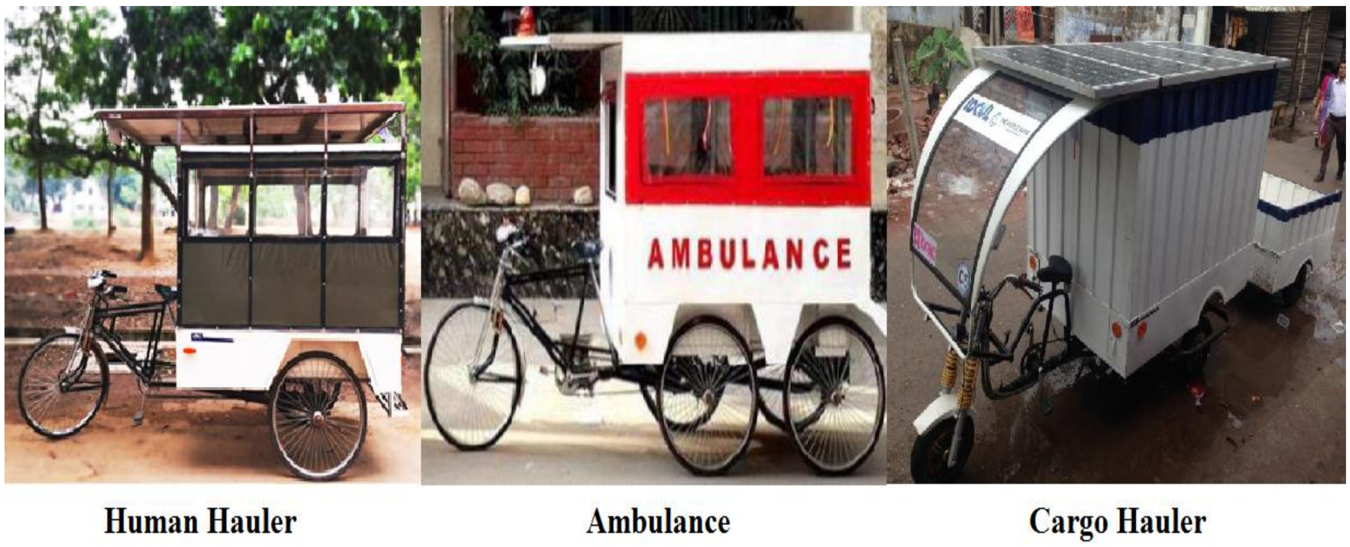


Figure 2.1: Three application area and scope of implementing the project

2.2 Block diagram of overall process

The figure 2 represent the block diagram of the overall process where one of the major parts is burning the algorithm into the microcontroller with externally connected hardware components are LCD display, potentiometer and USBASP AVR Programmer. First of all, write the algorithm in AVR Studio and then burn the algorithm using Extreme Burner software through physically connected device, USBASP AVR Programmer. After that, two input voltage will be fed into the microcontroller where one input voltage is come from the battery of the electric vehicle and the other input voltage is from the motor current of the electric vehicle through the current sensor (ACS758). Current sensor converts the current into voltage. Input voltage from the battery must need to step down the voltage within 5V as microcontroller cannot take more than 5V. Potentiometer is used to tune the brightness of LCD display. Thus, LCD display will show some current important information about the electric vehicle such as estimated state of charge (SOC), remaining battery discharging time as well as available travel distance. The figure 2.2 represent the whole block diagram of our system.

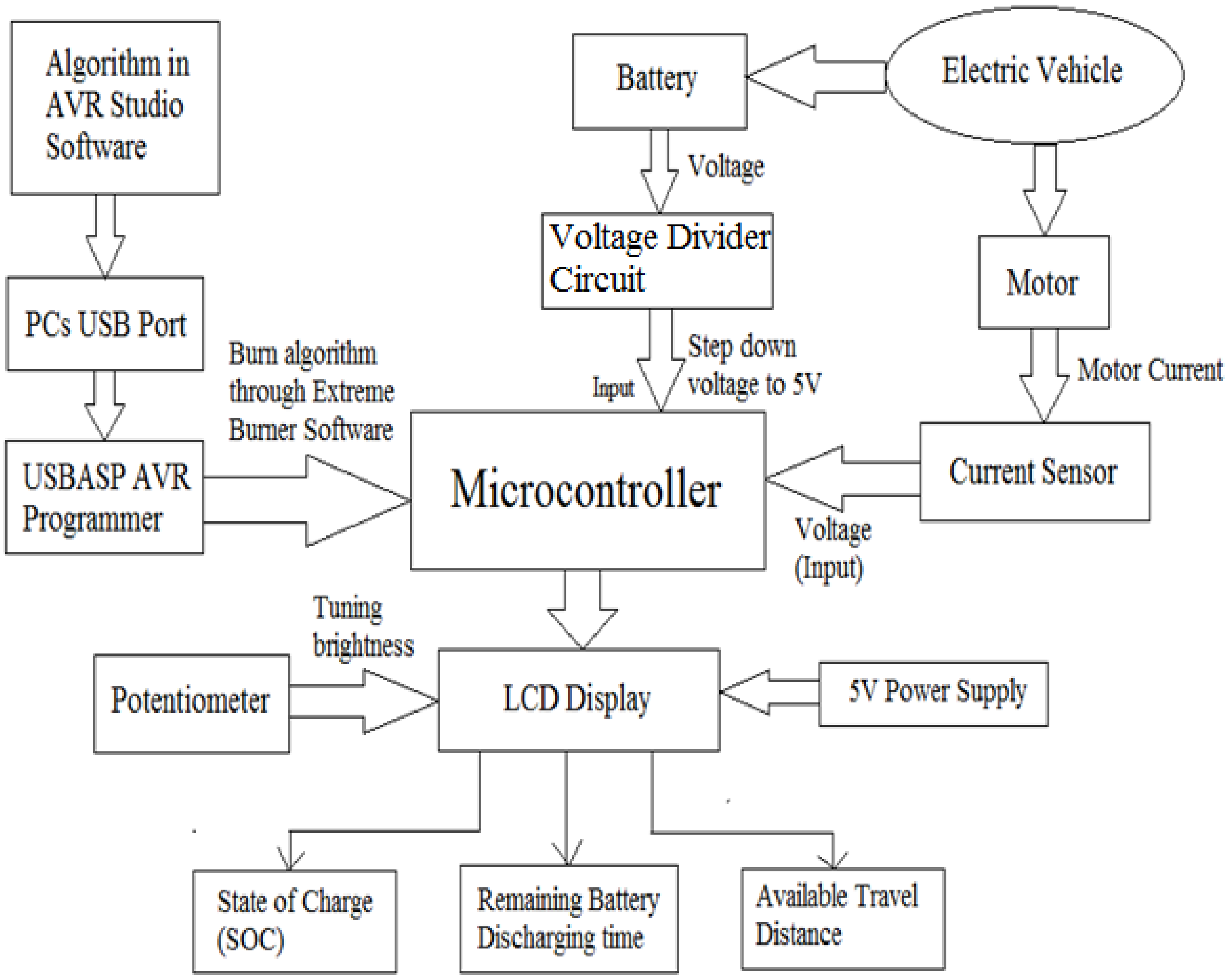


Figure 2.2: Block diagram of our project work

2.3 Circuit connection for whole system

In the block diagram, we already discussed about the whole system. The figure 2.3 represent the whole circuit connection what we applied in our project work to display SOC, remaining travelling time and distance.

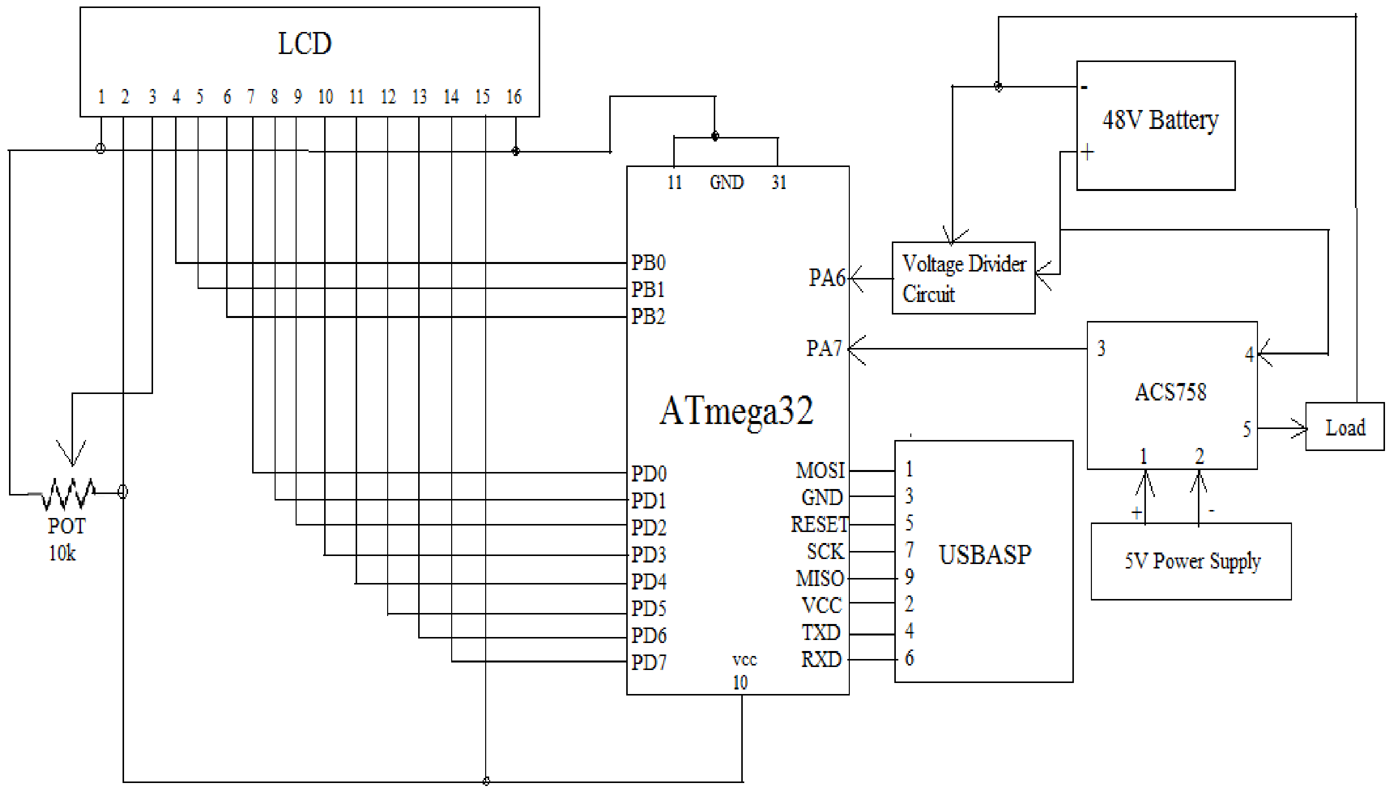


Figure 2.3: Circuit connection diagram for the whole system

Chapter 3

Burning algorithm process with hardware and software components

3.1 Background of AVR Programmer Software

The AVR is a modified 8-bit single chip microcontroller which was developed by Atmel in 1996. However, it was first introduce in 1997 and Atmel had shipped 500 million AVR flash microcontrollers within 2003. Atmel says that ‘AVR’ is not an acronym does not mean any particular thing. The inventors of AVR have not given any specific answer for what ‘AVR’ stands. Although it is usually use that AVR stands for Alf (EgilBogen) and Vegard (Wollan)’s RISC processor [8]. AVR programmer is an USB in circuit programmer for Atmel AVR microcontrollers. With this programmer, we can load hex files to AVR chips. Here, we use microcontrollers as AVR chips. The programmer uses an USB driver software. The AVR Programmers are directly controlled with port signals because it has no controller on the programmer. These programmers can also be worked with AVR Studio [9].

AVRs have been used in various automotive applications such as security, safety, display, powertrain and entertainment system. Besides, USB based AVRs have been used in the Microsoft Xbox hand controllers. Here, the USB connects the controllers and Xbox. Various companies induce AVR based microcontroller boards intended for use by hobbyists, robot builders experiments and small system developers including cubloc, gnusb, BasicX, Oak Micros,ZX Microcontrollers, and myAVR [8].

In our project we have used only AVR Studio 5.0 and Extreme Burner AVR. In this chapter, the software components will be described in sub-section 3.1.1 and 3.1.2.

3.1.1 AVR Studio 5.0

AVR Studio 5.0 is a software development environment produced by Atmel which is used with AVR microcontrollers. This software is a powerful tool designed for programming microcontrollers. Atmel AVR Studio 5.0 is the Integrated Development Environment (IDE) for developing and debugging embedded Atmel AVR applications. Indeed, it works with windows version of XP, Vista, 7, 8, and 10. The version of AVR Studio 5.1, 5.0 and 4.9 are usually downloaded to use by the programmers. AvrStudio.exe, avrstudio5.exe, avr32studio.exe, studio.exe or _294823.exe, these are the common filename of the program's installer. This gives us a seamless and easy-to-use environment for both beginners and experienced developers to write, build, and debug ourC/C++ and assembler code [10]. One of the great advantages of using this software was its large number of tutorials along with it which helps us to get familiar with this software.

Figure 3.1: Opening view of AVR Studio 5.0 software

It has got some very important features that are mentioned below-

- This software provides the environment for editor, simulator, programmer and so on.
- Having its own integrated C compiler the AVR GNU C Compiler (GCC), it does not need any third party C compiler [11].
- It permits chip simulation and in-circuit emulation.
- Same User Interface (UI) for simulation and emulation are used here.
- It can develop programs for both 8-bits and 32-bits AVR series of microcontrollers [11].
- Users can derive supports from this software by its easy access of datasheets, STK500, AVR Dragon etc. [10].
- It has 400 Example Projects [10].

3.1.1.1 Working with AVR Studio at a Glance:

After installation the AVR Studio software, we open the software to create a new C project. For creating a new C project, we need to select a file from the toolbar of the menu at the top corner. A window named new project will appear with the options of saving the location of our project with our desired file name. Hence, we will move on to select chip for our project. Here, we used ATmega32A for our project.Now, the main job has started for writing the code. Then, a new window comes with basic algorithm of our selected chip. The header of the algorithm“#include<avr32/io.h>” which provides with various input/output operations like DDRx, PINx, PORTx, etc. The port name of DDRx, PINx, PORTx refers to Port Direction Register, Port Input Register and Port Output Register respectively. Within the section of the while loop, we begin to write the required algorithm for displaying SOC (State Of Charge), remaining travelling time and distance travelled with the remaining SOC. The working process of AVR Studio 5.0 with required pictures have given stepwise in below-

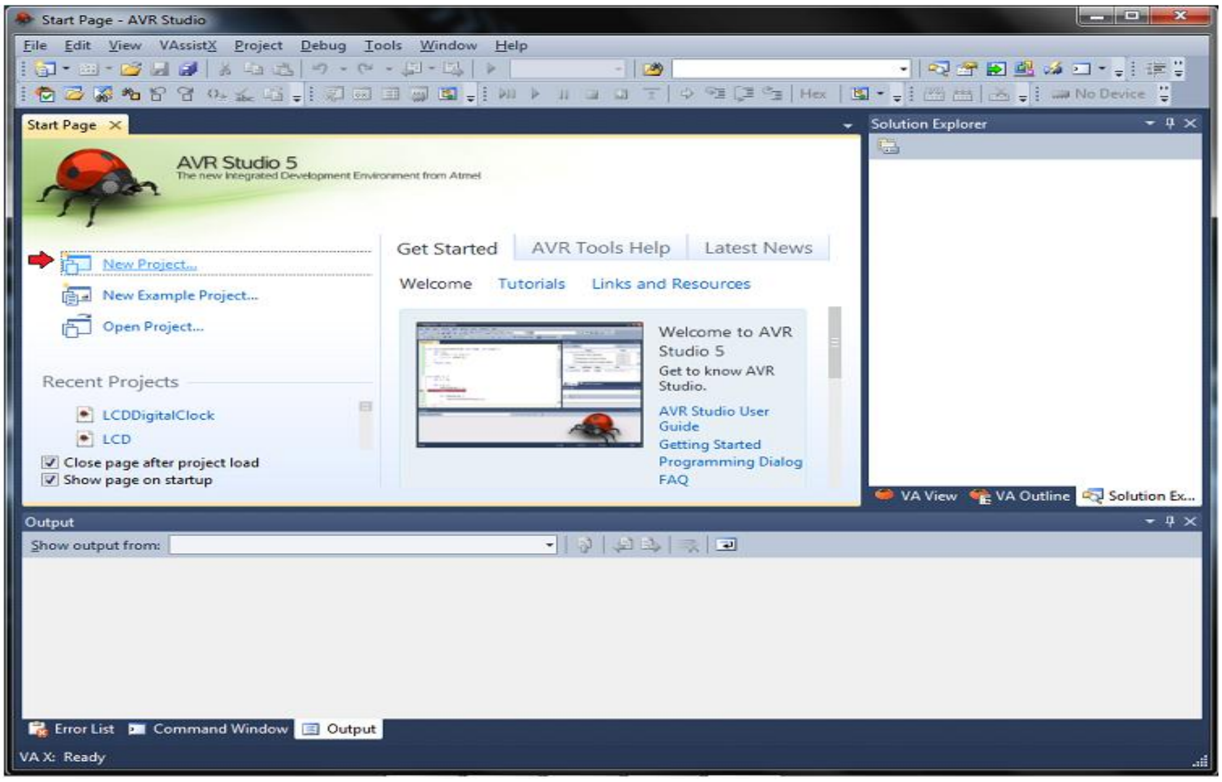
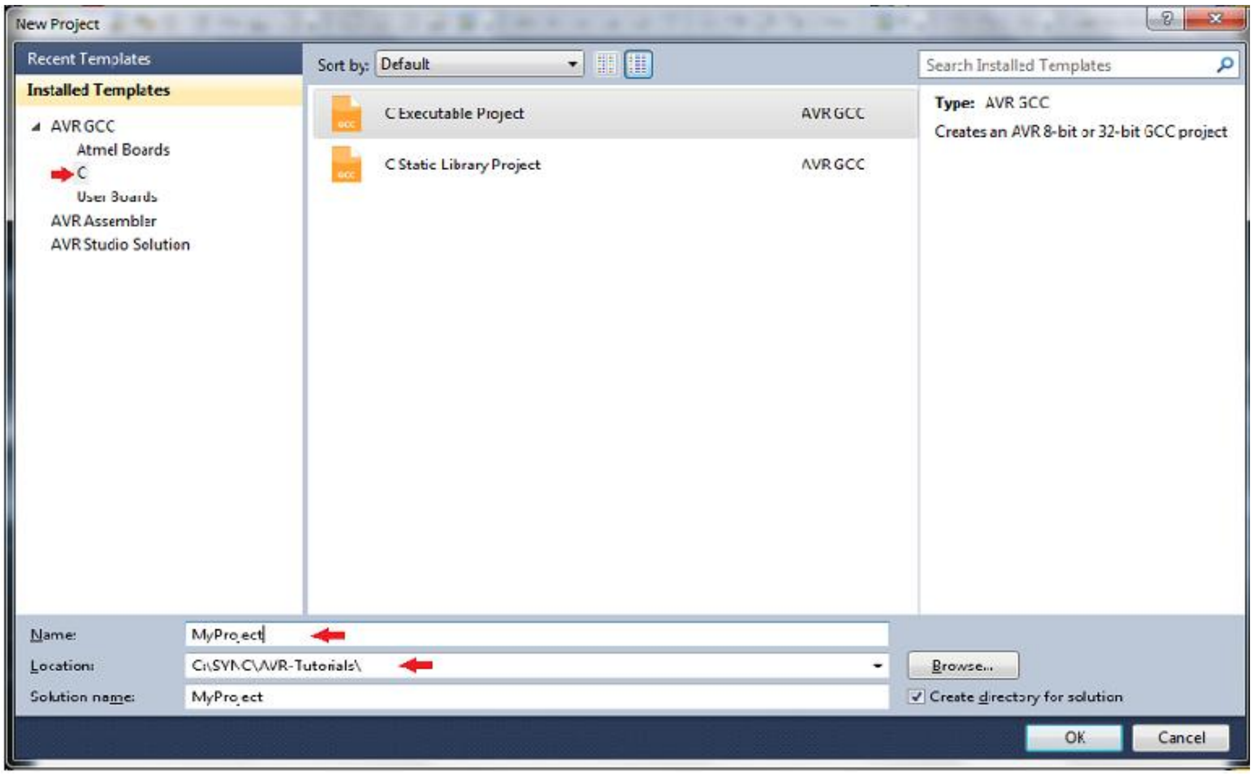


Figure 3.2: Starting page of AVR Studio 5.0



New Project

Recent Templates

Installed Templates

- AVR GCC
  - Atmel Boards
  - C
  - User Boards
- AVR Assembler
- AVR Studio Solution

Sort by: Default

Search Installed Templates

C Executable Project	AVR GCC
C Static Library Project	AVR GCC

Type: AVR GCC

Creates an AVR 8-bit or 32-bit GCC project

Name: MyProject

Location: C:\SYNC\AVR-Tutorials\

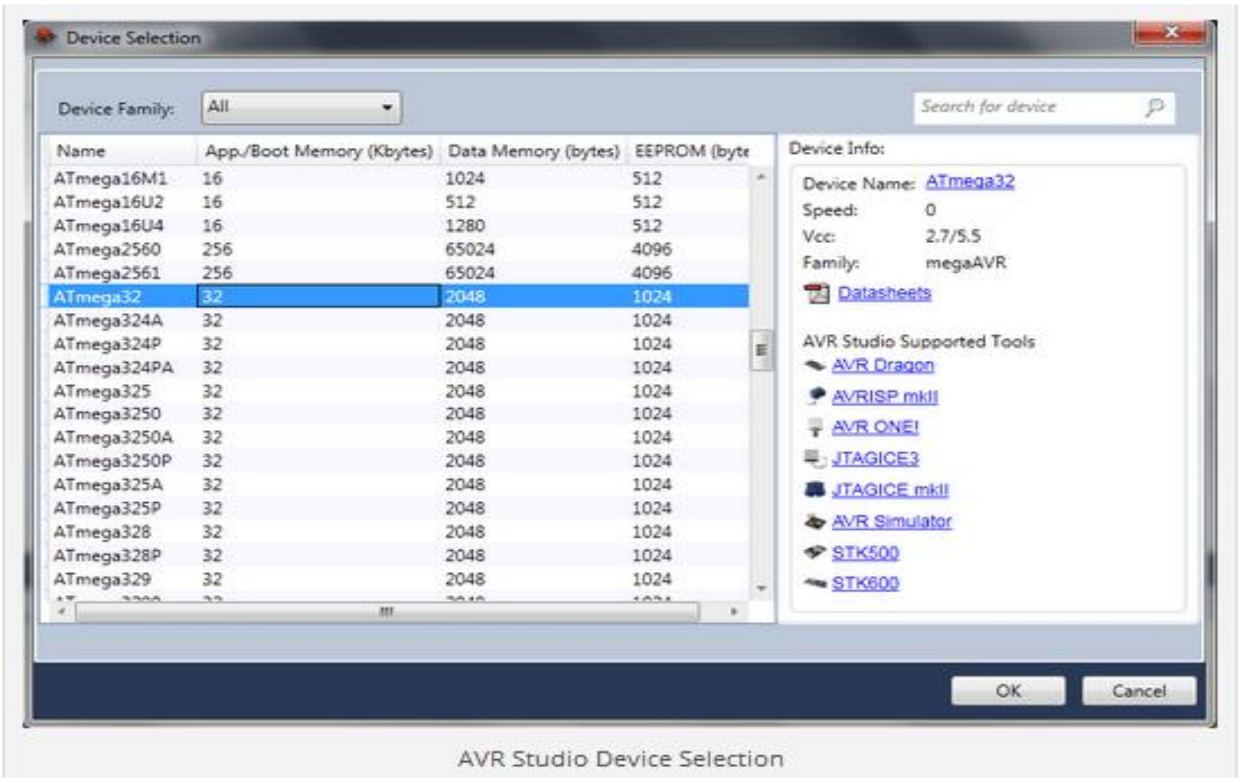
Solution name: MyProject

Browse...

✓ Create directory for solution

OK Cancel

Figure 3.3: New window for project information



Device Selection

Device Family: All

Search for device

Name	App./Boot Memory (Kbytes)	Data Memory (bytes)	EEPROM (byte
ATmega16M1	16	1024	512
ATmega16U2	16	512	512
ATmega16U4	16	1280	512
ATmega2560	256	65024	4096
ATmega2561	256	65024	4096
ATmega32	32	2048	1024
ATmega324A	32	2048	1024
ATmega324P	32	2048	1024
ATmega324PA	32	2048	1024
ATmega325	32	2048	1024
ATmega3250	32	2048	1024
ATmega3250A	32	2048	1024
ATmega3250P	32	2048	1024
ATmega325A	32	2048	1024
ATmega325P	32	2048	1024
ATmega328	32	2048	1024
ATmega328P	32	2048	1024
ATmega329	32	2048	1024

Device Info:

Device Name: ATmega32

Speed: 0

Vcc: 2.7/5.5

Family: megaAVR

Datasheets

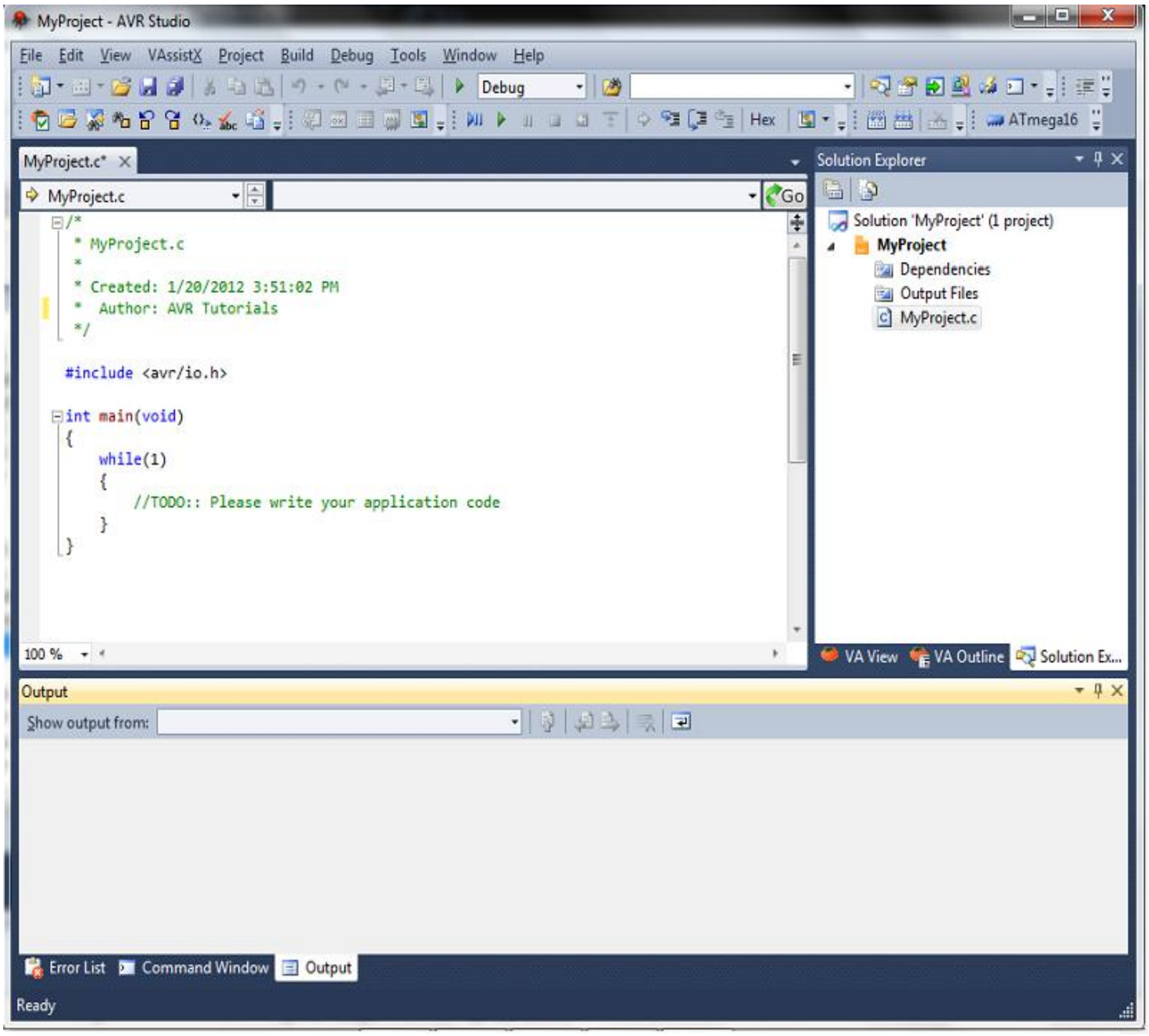
AVR Studio Supported Tools

- AVR Dragon
- AVRISP mkII
- AVR ONE!
- JTAGICE3
- JTAGICE mkII
- AVR Simulator
- STK500
- STK600

OK Cancel

AVR Studio Device Selection

Figure 3.4: Chip selection of AVR Studio



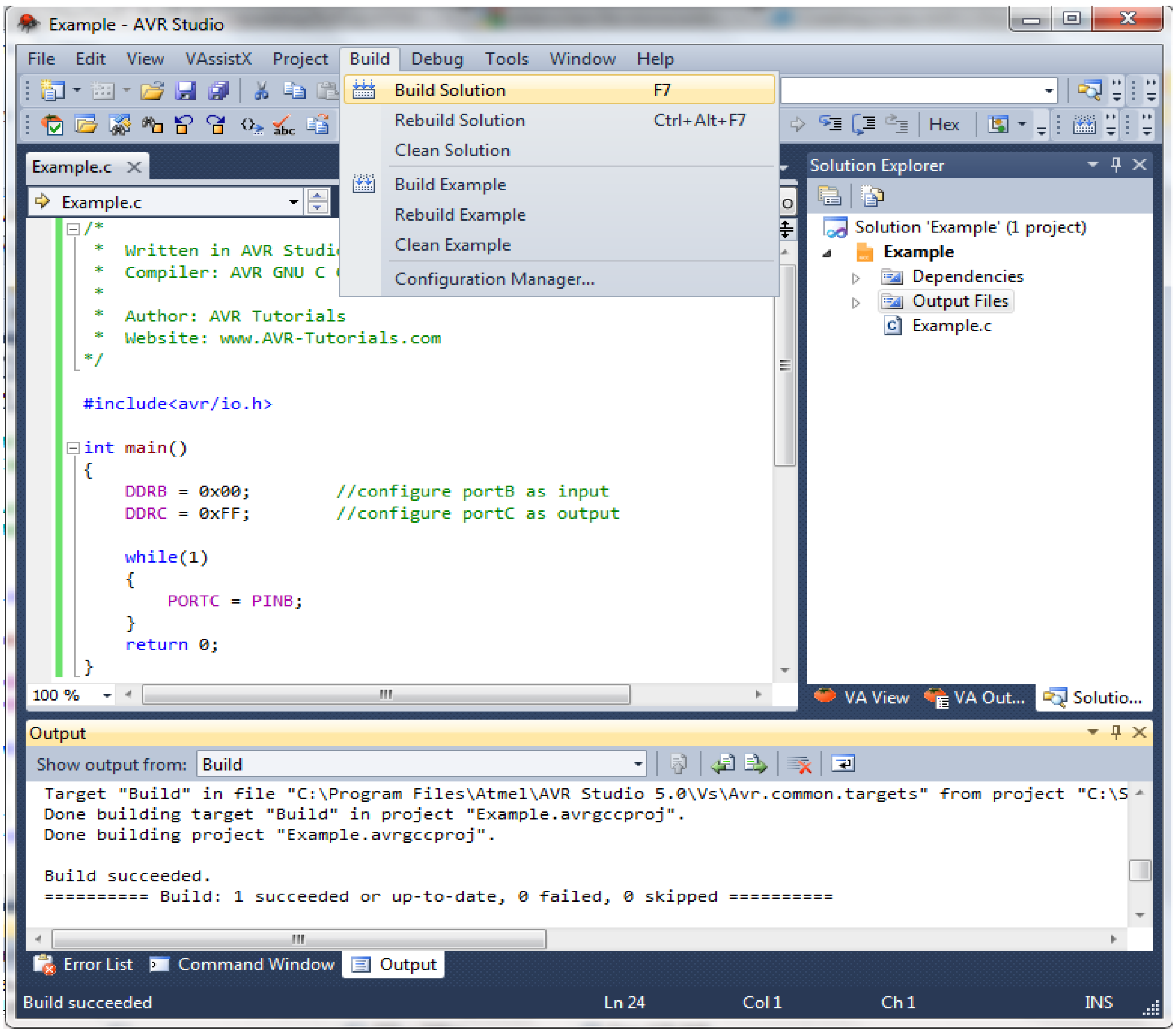
```
/*
 * MyProject.c
 *
 * Created: 1/20/2012 3:51:02 PM
 *  Author: AVR Tutorials
 */

#include <avr/io.h>

int main(void)
{
    while(1)
    {
        //TODO:: Please write your application code
    }
}
```

Figure 3.5: AVR Studio New Project Start Screen

After writing the algorithm, we instruct the Solution Explorer to compile and check whether there are any errors in the code. If the AVR C/C++ or Assembly program compiled and built successfully, a message will display at the bottom of the AVR Studio 5.0 Editor indicating that **Build succeeded** which leads us to move on further process to burn the algorithm into the microcontroller (ATmega32A) using our next software Extreme Burner AVR otherwise it shows error as a result in the output box which recommend us to check the algorithm again and amend all the errors we have done while writing the code. In this case we will have to recompile the algorithm to get the assurance to continue the remaining process.



```
/*
 *  Written in AVR Studi
 *  Compiler: AVR GNU C
 *
 *  Author: AVR Tutorials
 *  Website: www.AVR-Tutorials.com
*/

#include<avr/io.h>

int main()
{
    DDRB = 0x00;        //configure portB as input
    DDRC = 0xFF;        //configure portC as output

    while(1)
    {
        PORTC = PINB;
    }
    return 0;
}
```

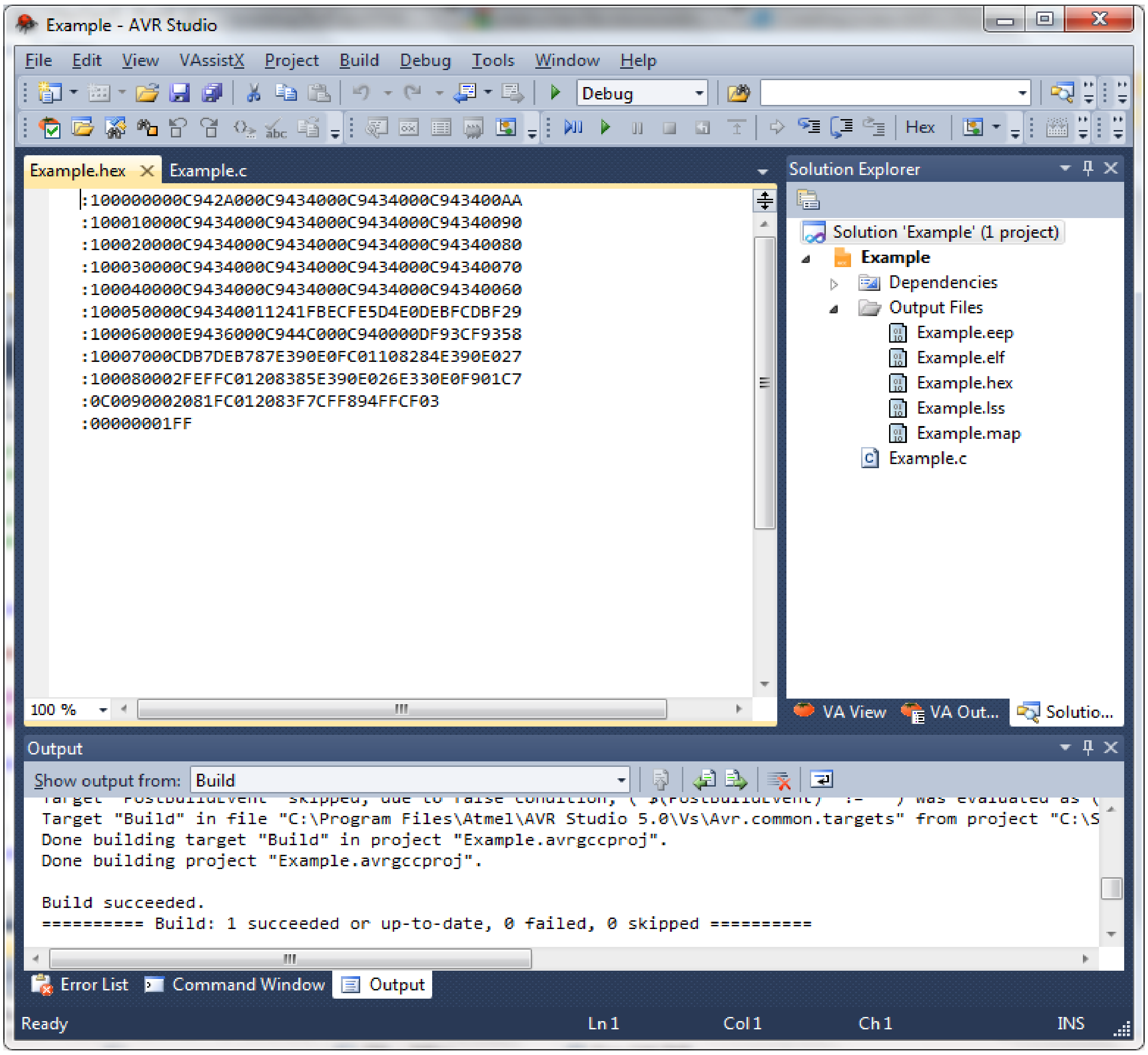
```
Target "Build" in file "C:\Program Files\Atmel\AVR Studio 5.0\Vs\Avr.common.targets" from project "C:\S
Done building target "Build" in project "Example.avrgccproj".
Done building project "Example.avrgccproj".

Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

Figure 3.6: Sample of Successful compilation an algorithm in AVR Studio 5.0 [12]

When the compilation is completed successfully, we go to the 'Output Files' folder in the Solution Explorer window. There are many output files named **.eep, .elf, .hex, .lss, .map** which are machine readable file but we need to save only the **.hex** file for burning the algorithm using the Extreme Burner software into the microcontroller.The content of the generated hex file can be viewed by double clicking on the file with the**.hex** extension in the **Solution Explorer** to the left of the AVR Studio 5.0. In addition to the output files, there is one **.c** file where we created our algorithm in human language. Note to be pointed that the complete code has been given in the appendix C section.

A sample hex file that generated is displayed in the AVR Studio 5.0 Editor shown in the figure 3.7 below-



```
:100000000C942A000C9434000C9434000C943400AA
:100010000C9434000C9434000C9434000C94340090
:100020000C9434000C9434000C9434000C94340080
:100030000C9434000C9434000C9434000C94340070
:100040000C9434000C9434000C9434000C94340060
:100050000C94340011241FBECFE5D4E0DEBFCDBF29
:100060000E9436000C944C000C940000DF93CF9358
:10007000CDB7DEB787E390E0FC01108284E390E027
:100080002FEFFC01208385E390E026E330E0F901C7
:0C0090002081FC012083F7CFF894FFCF03
:00000001FF
```

```
Target "PostBuildEvent" skipped, due to false condition; ( $(PostBuildEvent) != '' ) was evaluated as (
Target "Build" in file "C:\Program Files\Atmel\AVR Studio 5.0\Vs\Avr.common.targets" from project "C:\S
Done building target "Build" in project "Example.avrgccproj".
Done building project "Example.avrgccproj".

Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

Figure 3.7: A sample of generating hex file in AVR Studio 5.0 [12]

3.1.2 Extreme Burner AVR (Burner Software)

Figure 3.8: Opening view of Extreme Burner AVR software

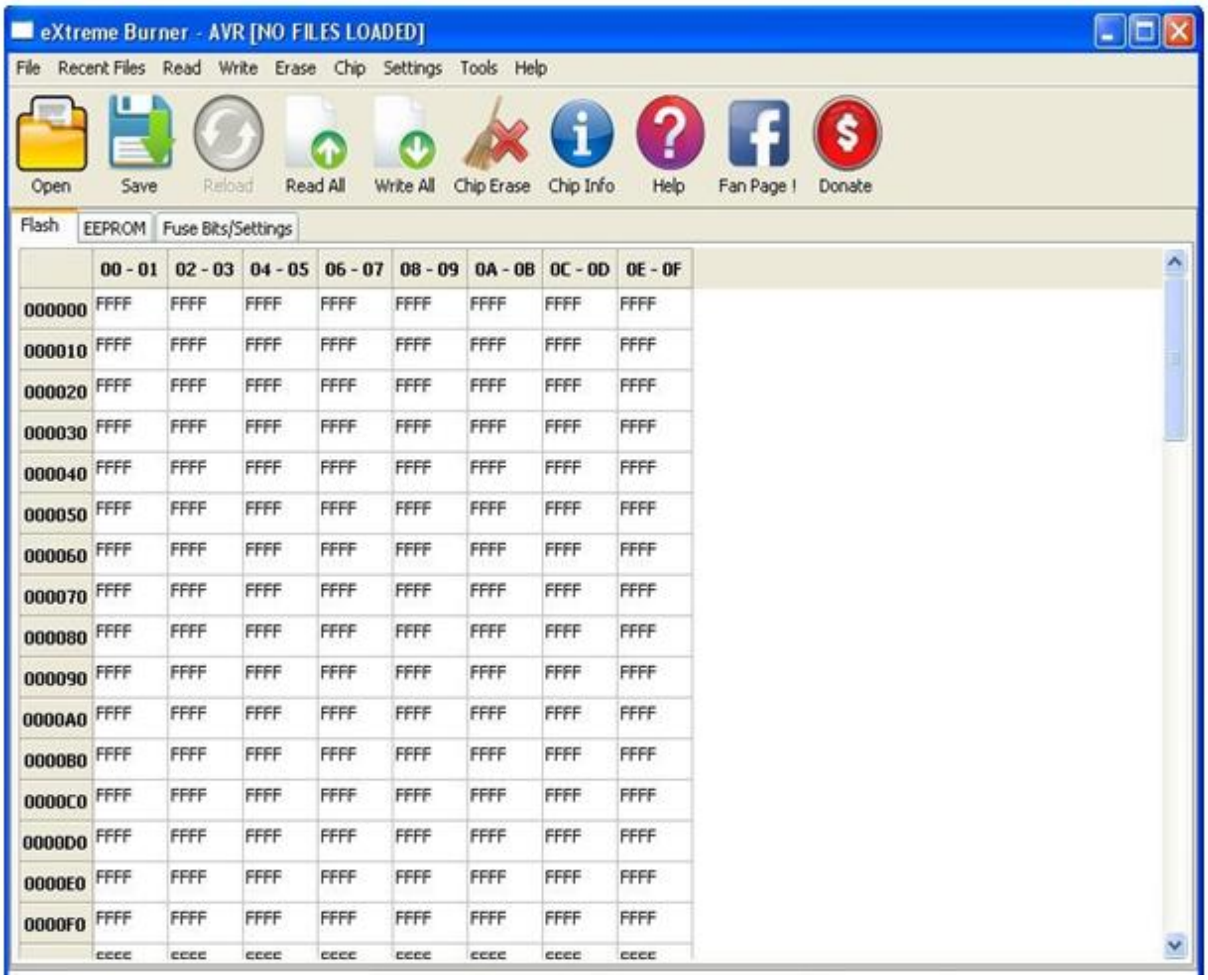
Extreme Burner is a low cost USB port that provides burning the hex file of the algorithm in the microcontroller. There are some basic features like In Circuit Serial Programming (ICSP) and High Voltage Programming supports by the Extreme Burner AVR for more than 45 devices [13]. Moreover, different kinds of clock sources for several applications are supported by the Extreme Burner AVR which is completely graphical user interface (GUI) AVR series of microcontroller. This software permits us to read and write a RC oscillator or a perfect high speed crystal oscillator. Clock sources can be selected from the following options:

-external clock
-calibrated Internal RC Oscillator
-external RC oscillator
-external low frequency crystal
-external crystal/ceramic resonator

There are versions of 1.4, 1.3, 1.3 beta, 1.2, 1.1, 1.0 [14]. Among these versions, we have used version 1.2 to execute our project.

3.1.2.1 Working Process of Extreme Burner AVR

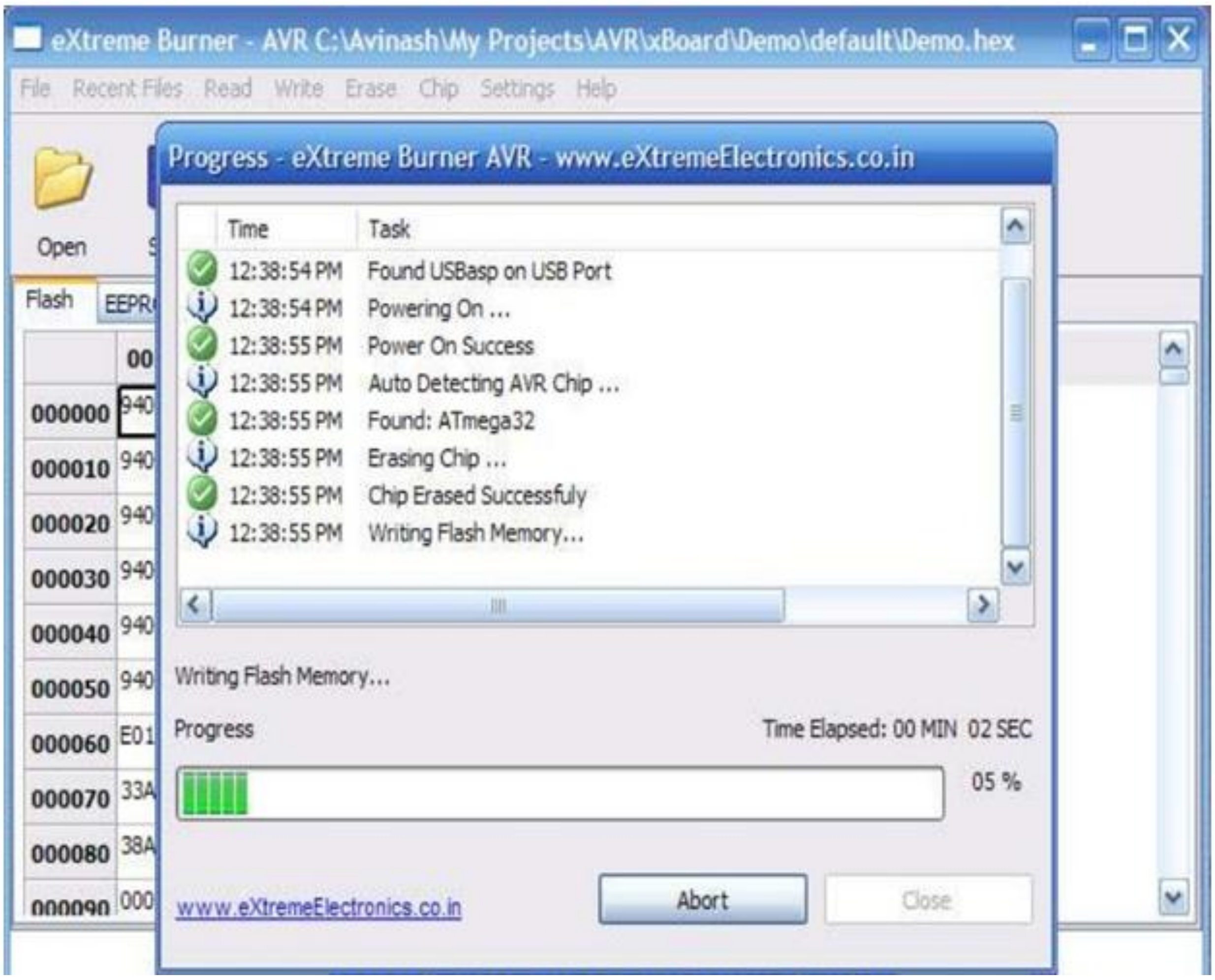
In this section, we will discuss how to burn the code into the microcontroller by the help of Extreme Burner AVR. We will have to make sure that during the installation of Extreme Burner Software, USB AVR programmer must be connected with the USB port. If the **USB device is not recognized**is shown in the window we need to check whether the USB cable is broken or not. If the command says "**New Hardware Found"**,it means we have connected the programmer to different port instead of using the port we have installed [15]. After the installation process, we launch Extreme Burner AVR from desktop icon or start menu. The software will open and the screen will be similar like the figure 3.9.



```
        00 - 01  02 - 03  04 - 05  06 - 07  08 - 09  0A - 0B  0C - 0D  0E - 0F
000000  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
000010  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
000020  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
000030  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
000040  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
000050  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
000060  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
000070  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
000080  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
000090  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
0000A0  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
0000B0  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
0000C0  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
0000D0  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
0000E0  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
0000F0  FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF     FFFF
```

Figure 3.9: Starting page of Extreme Burner AVR

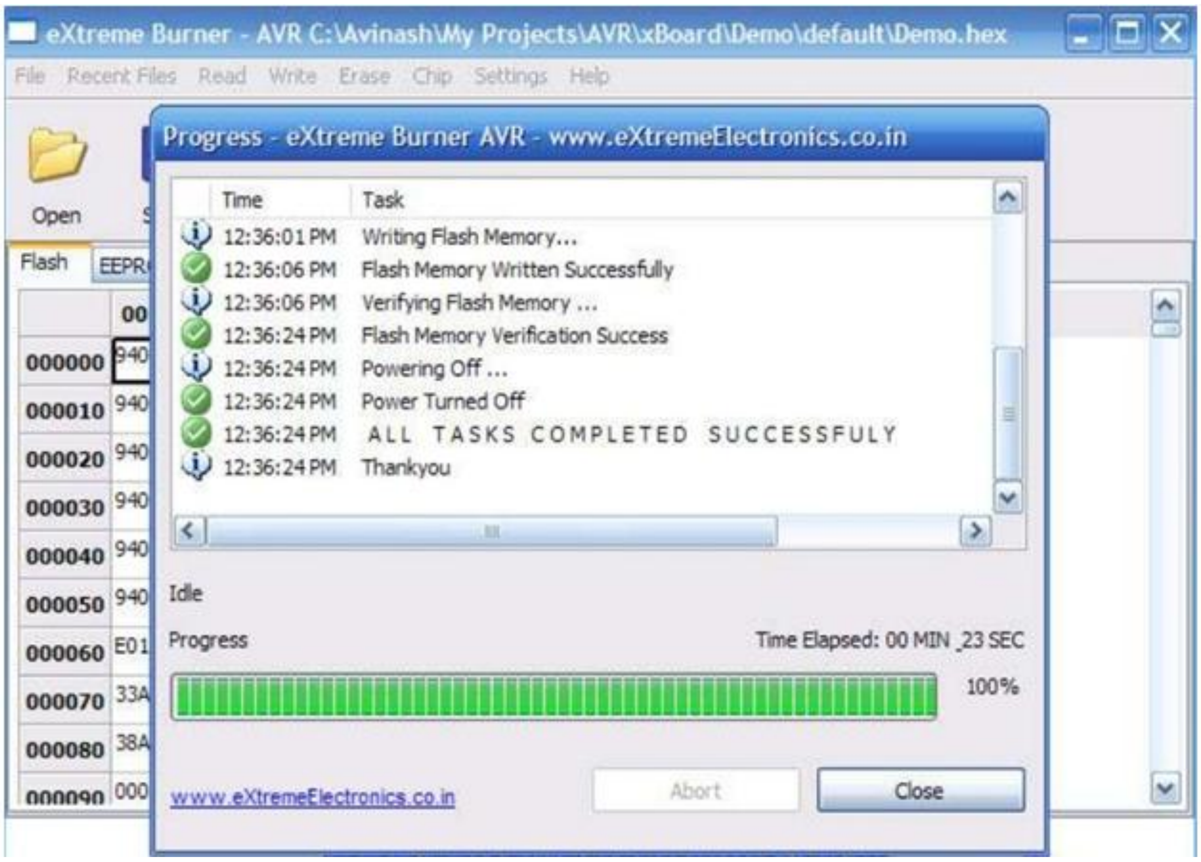
Before burning the algorithm, we connect the microcontroller (ATmega32A) with the AVR Programmer according to the pin configuration which will be discussed in section 3.2. At first, we open the hex file in the Extreme Burner AVR. Additionally, we select the chip (the microcontroller, ATmega32A). Then we select "**Write All"** icon to burn the program in the microcontroller that has been simulated using the previous software (AVR Studio 5.0). While burning the code, Red LED will glow which indicates **BUSY** State and the following messages will appear if everything is connected properly.


eXtreme Burner - AVR C:\Avinash\My Projects\AVR\xBoard\Demo\default\Demo.hex
File Recent Files Read Write Erase Chip Settings Help
Open
Flash
EEPR
Progress - eXtreme Burner AVR - www.eXtremeElectronics.co.in
Abort
Close
www.eXtremeElectronics.co.in

```
Time          Task
12:38:54 PM   Found USBasp on USB Port
12:38:54 PM   Powering On ...
12:38:55 PM   Power On Success
12:38:55 PM   Auto Detecting AVR Chip ...
12:38:55 PM   Found: ATmega32
12:38:55 PM   Erasing Chip ...
12:38:55 PM   Chip Erased Successfuly
12:38:55 PM   Writing Flash Memory...

Writing Flash Memory...
Progress                       Time Elapsed: 00 MIN 02 SEC
                               05 %
```

Figure 3.10: Extreme Burner AVR-Burning Process


eXtreme Burner - AVR C:\Avinash\My Projects\AVR\xBoard\Demo\default\Demo.hex
File Recent Files Read Write Erase Chip Settings Help
Open
Flash
EEPR
Progress - eXtreme Burner AVR - www.eXtremeElectronics.co.in
Idle
Progress
Time Elapsed: 00 MIN 23 SEC
100%
www.eXtremeElectronics.co.in
Abort
Close

```
Time         Task
12:36:01 PM  Writing Flash Memory...
12:36:06 PM  Flash Memory Written Successfully
12:36:06 PM  Verifying Flash Memory ...
12:36:24 PM  Flash Memory Verification Success
12:36:24 PM  Powering Off ...
12:36:24 PM  Power Turned Off
12:36:24 PM  ALL TASKS COMPLETED SUCCESSFULY
12:36:24 PM  Thankyou
```

```
       00
000000 940
000010 940
000020 940
000030 940
000040 940
000050 940
000060 E01
000070 33A
000080 38A
000090 000
```

Figure 3.11: Extreme Burner AVR-Task Complete

However, we have faced a problem at the time of burning the algorithm in the microcontroller. The problem that has occurred sometimes is the miscommunication between microcontroller and AVR programmer. If the chip is programmed properly, the command "**ALL TASKS COMPLETED SUCCESSFULLY"** appears in the window named progress as shown in the above figure 3.11. Now we will disconnect the programmer kit from the PC.

In this section 3.1, we discussed how these two softwares are dependent on each other to display our desired result in LCD display. As we mentioned before, AVR Studio 5.0 is mainly for writing and compiling the algorithm whereas Extreme Burner software is for burning those whole algorithm into the AVR microcontroller. Thus in section 3.2, we will discuss briefly about the hardware components those we used to accomplish the desired tasks for our project purpose.

3.2 Overview of our hardware components

Hardware components are considered as the structured combination of physical components which make up a system. These hardware components drawn interdependent relationship with software components which means these two types of components are built in such a way that one is controlled by another. For our project work, we used various types of hardware components which will be described in this section. In our project, we are displaying SOC (State Of Charge), travelling time and coverable distance with in LCD display using microcontroller (MCU). As microcontroller controls the overall system, we need to be more conscious about the input voltage in the microcontroller which should not be more than 5V, otherwise microcontroller will be burnt.

The list of hardware components we used for our project purpose is given below-

1. USBASP AVR Programmer (USBASP tiny programmer USB to ISP 10 Pin with case silver)
2. Microcontroller (ATmega32A)
3. LCD display (YJ204A LCD text display blue 20X4 line)
4. Potentiometer (10kΩ)
5. Resistors
6. Wire connections (Crocodile wires, Bread board, Multi-meter, Clamp-meter, Trainer-board, Jumper wire etc.)

3.2.1 USBASP AVR Programmer (Burning code into MCU)

USBASP is a Universal Serial Bus (USB) in-circuit programmer for Atmel AVR controllers which simply consists of an ATMega8 and a couple of passive components. As no special USB controller is needed so the programmer uses a firmwaredriver that makes this programmer attractive to many amateurs. USBASP is officially included and supported by WinAVR. Therefore, USBASPs are one of the least expensive options to programming AVRs as we are developing an electrical device for the electric vehicle which is used by the people of lower economic state.

It has got some very important features that are mentioned below:

1. Designed to read or write the microcontroller EEPROM, firmware, fuse bits and lock bits.
2. Supported by Windows, Mac OS X and Linux.
3. Speed of programmingis up to 5kBytes/sec.
4. To support targets with low clock speed(< 1.5MHz), it has software controlled SCK (Serial Clock) option.
5. Consists of 10 pin ISP (In-System Programming) interface with case silver [16].

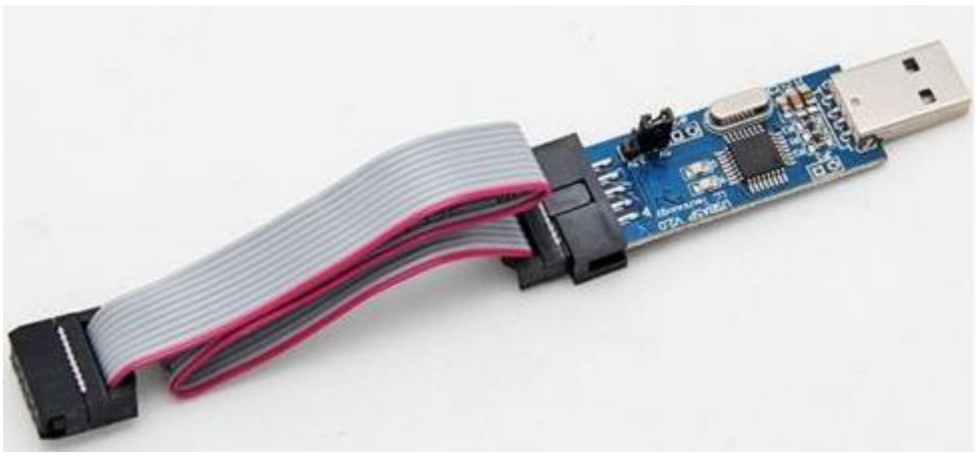
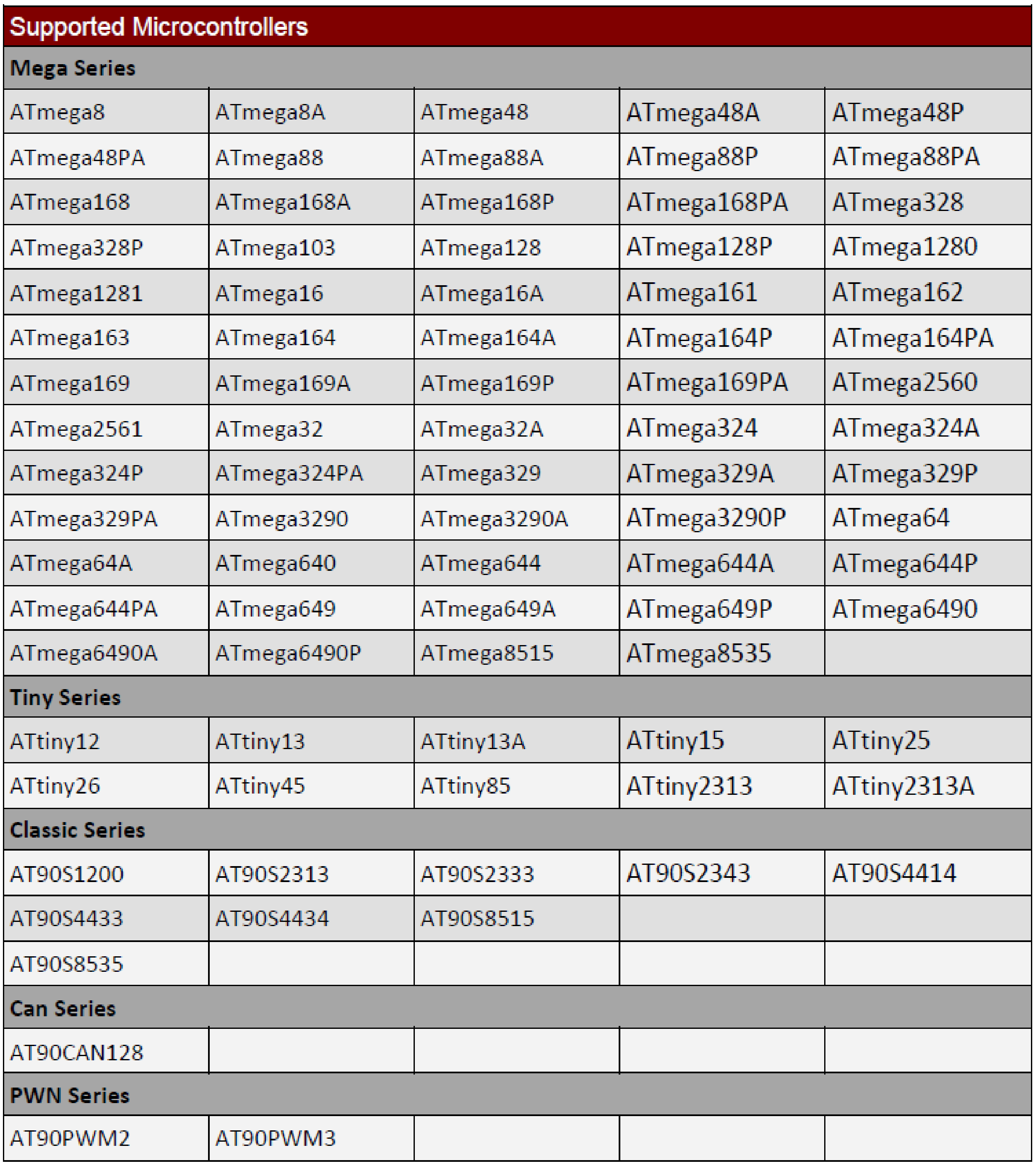


Figure 3.12: USBASP AVR Programmer (V2.0)

3.2.1.1 Supported Microcontrollers for USBASP Programmer:

From the list of microcontroller, we are using ATmega32A to control the whole display system according to the algorithm for our project purpose.



Supported Microcontrollers				
Mega Series				
ATmega8	ATmega8A	ATmega48	ATmega48A	ATmega48P
ATmega48PA	ATmega88	ATmega88A	ATmega88P	ATmega88PA
ATmega168	ATmega168A	ATmega168P	ATmega168PA	ATmega328
ATmega328P	ATmega103	ATmega128	ATmega128P	ATmega1280
ATmega1281	ATmega16	ATmega16A	ATmega161	ATmega162
ATmega163	ATmega164	ATmega164A	ATmega164P	ATmega164PA
ATmega169	ATmega169A	ATmega169P	ATmega169PA	ATmega2560
ATmega2561	ATmega32	ATmega32A	ATmega324	ATmega324A
ATmega324P	ATmega324PA	ATmega329	ATmega329A	ATmega329P
ATmega329PA	ATmega3290	ATmega3290A	ATmega3290P	ATmega64
ATmega64A	ATmega640	ATmega644	ATmega644A	ATmega644P
ATmega644PA	ATmega649	ATmega649A	ATmega649P	ATmega6490
ATmega6490A	ATmega6490P	ATmega8515	ATmega8535	
Tiny Series				
ATtiny12	ATtiny13	ATtiny13A	ATtiny15	ATtiny25
ATtiny26	ATtiny45	ATtiny85	ATtiny2313	ATtiny2313A
Classic Series				
AT90S1200	AT90S2313	AT90S2333	AT90S2343	AT90S4414
AT90S4433	AT90S4434	AT90S8515		
AT90S8535				
Can Series				
AT90CAN128				
PWN Series				
AT90PWM2	AT90PWM3			

Figure 3.13: List of supported microcontrollers for USBASP [17]

3.2.1.2 USBASP Device Outline

In USBASP device, there are several components like USB Type A, JP1-Supply Target, JP2-Self Program, JP3-Slow SCK, LEDs (Light Emitting Diode), ISP 10 Pin IDC (Insulation Displacement Connector). Introducing the silver case part which is the USB end of the programmer; it directly connects with the PCs USB port to write the whole algorithm in the microcontroller. Another part of this device is JP1-Supply Target which regulates the voltage of ISP VCC connector. If our desired particular device has its own power source then we exclude use of the jumper. Whenever no power source is there, we can enable multi-voltage programmer that will set voltage to +3.3V/+5V. The jumper JP2-Self Program is used to update the firmware of USBASP Programmer. Afterwards, JP3 jumper stands for slow SCK mode, this mode is enabled when JP3 is selected and set the clock again if it is lower than 1.5MHz. In this case, 8 KHz to about 375 KHz is the range for SCK. The outline of the USBASP Programmer is shown in Figure 3.14.

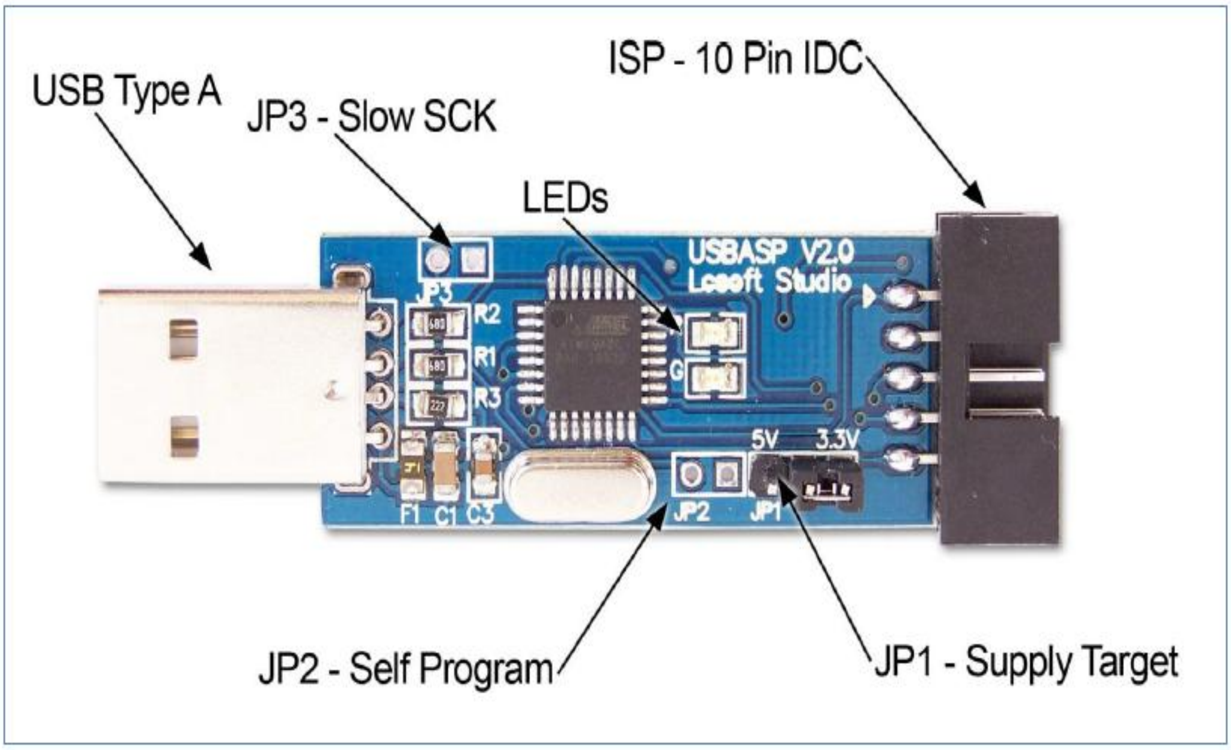


Figure 3.14: USBASP device outline (V2.0) [17]

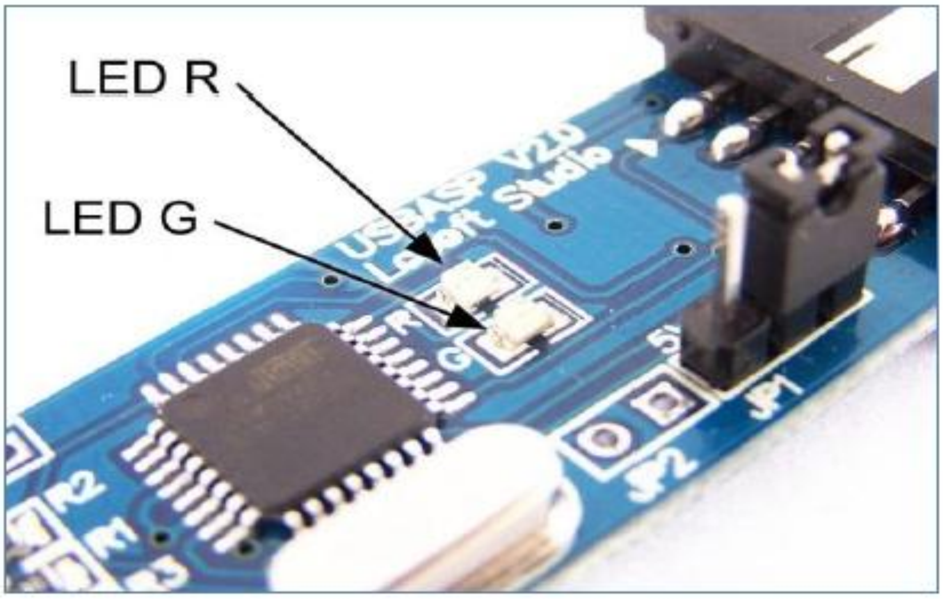


Figure 3.15: USBASP LEDs (V2.0) [17]

Two LEDs are also used in this device to represent some particular operation, where LED R refers to programmer which is communicating with target device and LED G is for power. When USBASP gets connected with PCs USB port then it is ready to write the algorithm from the Extreme Burner Software, LEDs glows and make a 'ding' sound if every connection is given properly with the microcontroller.

3.2.1.3 In System Programming (ISP)-10 Pin IDC

For interfacing with the microcontroller, 10 pin ISP connections are MOSI, VCC+5V, Ground, TXD, Reset, RXD, SCK and MISO. Ground pins are 3, 8, 10 and they are internally shorted, so we only need to connect one ground with the microcontroller as they operate with the same reference voltage. In addition, three pins constitute the SPI (Serial Peripheral Interface) which are SCK, MISO (Master In-Slave Out) and MOSI (Master Out-Slave In). In system programmer (USBASP AVR Programmer) always lead as the Master and the target device (microcontroller) lead as the Slave when programming with the AVR. In order to control any system, master and slave device need efficient communication amongst each other. The master device can write data to the slave device's flash memory and can receive data from the slave's memory through MOSI and MISO pins. The master device can communicate on the SCK, MISO and MOSI lines when the Reset pin is set low. TXD and RXD are also stands for transmitter and receiver respectfully to communicate with the microcontroller. The pin configuration of USBASP AVR Programmer is given in figure 3.16.

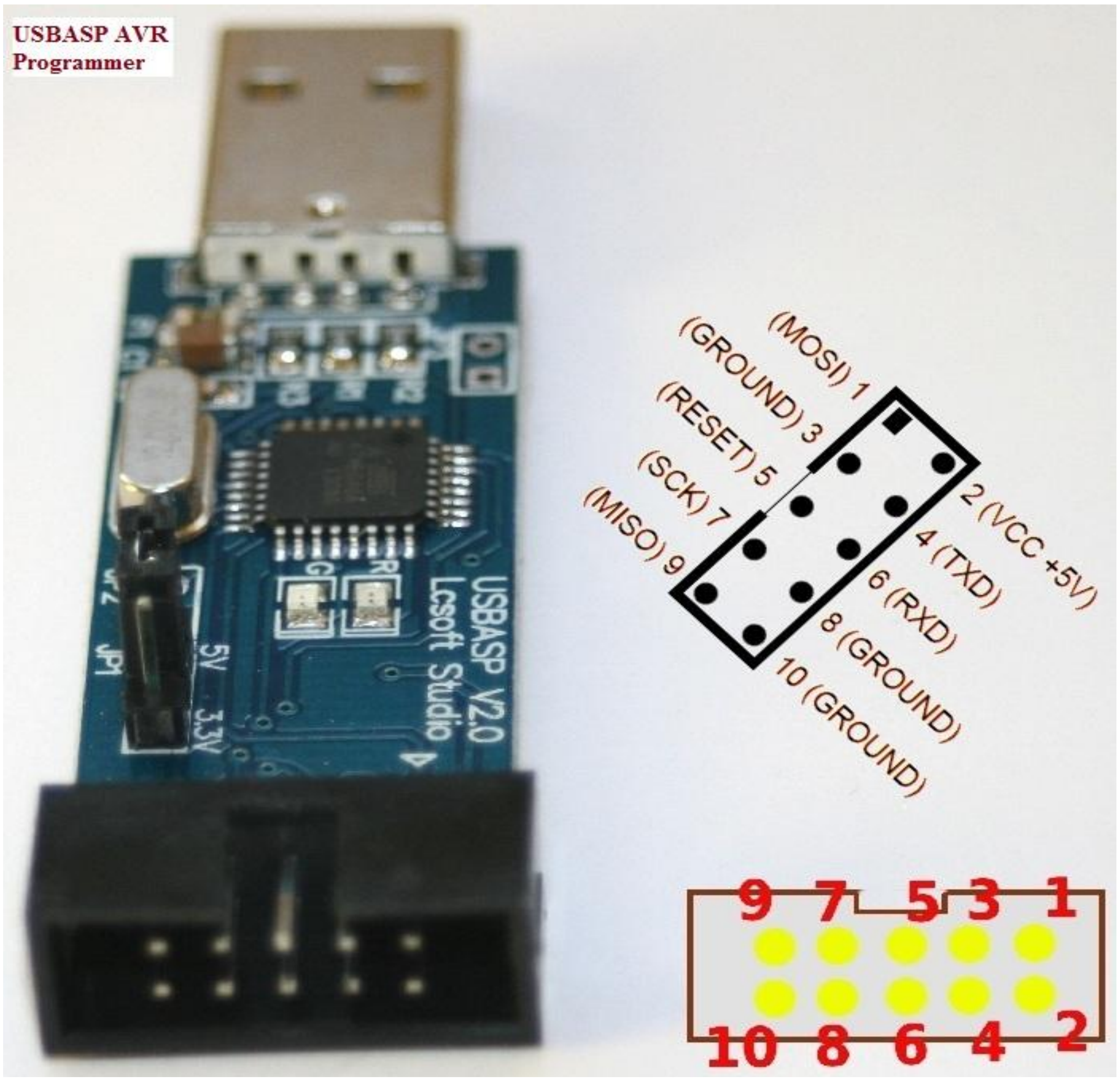


Figure 3.16: USBASP 10 Pin Configuration [17]

These ISP 10 pins are connected through IDC cable, male to male jumper wire and then connected with the breadboard according to the pin configuration of the microcontroller; connection diagram has given in chapter 2.

Communication details and the function of each pins between USBASP AVR Programmer and Microcontroller are mentioned below in Table 3.1 [18].

Pin	Comment
MOSI (Master Out-Slave In)	Communication line from USBASP In-System Programmer (Master) to desired AVR Microcontroller (Slave). This allows master device to send data to the slave device.
MISO (Master In-Slave Out)	Communication line from desired AVR Microcontroller (Slave) to USBASP In-System Programmer (Master). This allows the slave device to send information to master device.
SCK (Serial Clock)	Programming clock which is generated by the USBASP In-System Programmer (Master). For synchronized communication, this clock is shared between the master and slave device.
GND (Common Ground)	AVR Microcontroller and USBASP must be connected with the common ground for operation.
VCC (Target Power)	USBASP and AVR Microcontroller can operate with the highest value of voltage which is not more than 5V.
Reset (Target AVR Microcontroller Reset)	USBASP Programmer can control the target Microcontroller reset. This reset pin must be put in active low for the AVR microcontroller being programmed.

Table 3.1: Characteristics of USBASP ISP 10 Pin

3.2.2 LCD Display (20X4)

A 20X4 Liquid Crystal Display (LCD) has 4 lines and 20 characters with white characters on a vivid blue background and backlight. Based on the status of Register Select (RS) pin, LCD module can recognize two types of signals, one is data and another is control. LCD display takes 39-43µS time to execute a command or place any character. In addition, there are two types of RAMs such as DDRAM and CGRAM for LCD display where DDRAM refers to in which position which character in the ASCII chart would be displayed whereas CGRAM allows user to define their custom characters. LCD controller read the information from DDRAM and displays it on the LCD screen [19].

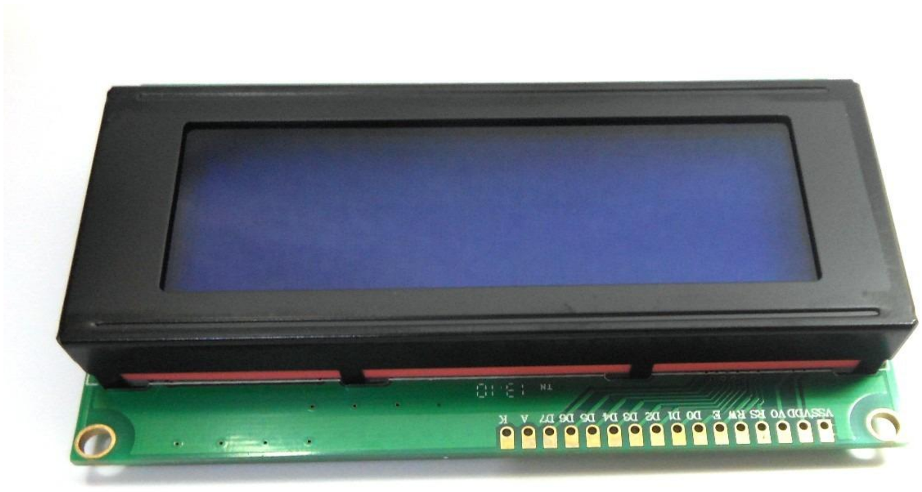


Figure 3.17: LCD Display (20X4)

LCD display has got some very important features that are mentioned below:

1. Wide viewing angle and high contrast is available.

2. Don't need any separate power supply for backlight.

3. Built in LCD controller equivalent to industry standard HD44780.

4. DC operating voltage is 5V.

5. Supported 4 or 8-bit parallel interface.

3.2.2.1 Pin configuration of LCD Display

The figure 3.18 is given below showing 16 pins in the LCD display which can be connected with the microcontroller using external circuits. Data is transferred from the microcontroller to the LCD display through these bidirectional data bus pin DB0-DB7 and display information corresponds to the algorithm. LCD contrast adjustment is possible by using potentiometer in Vo and LCD controlling pins are RS, R/W, E refers to register select, read/write signal and enable respectively. RS is used to make the selection between data and instruction register. When Rs=0 (Instruction register), RS=1 (Data register). Furthermore, R/W gives the choice between read and write operation; when R/W=1 (Read), R/W=0 (Write). Moreover, Enable pin is used by the LCD to latch information presented to its bi-directional data pins. When data is supplied to 8-bit data pins, a high to low pulse must be applied to this pin in order for the LCD to latch the data present at the data pins.

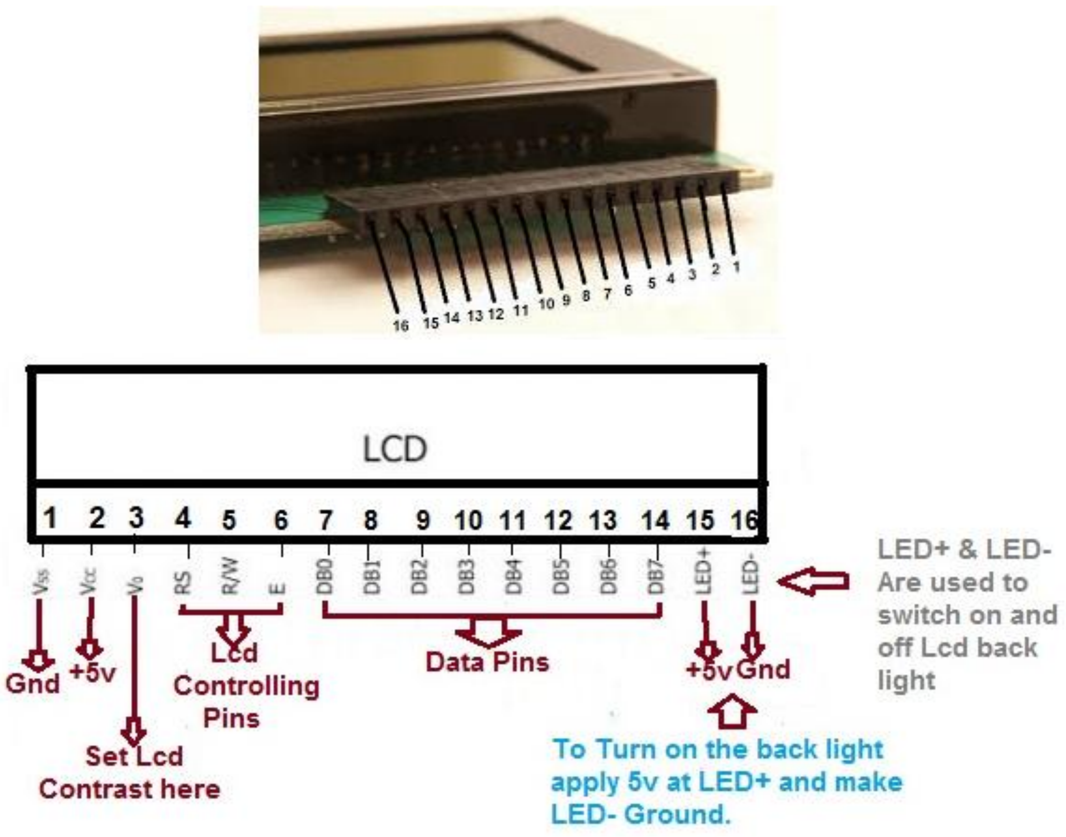


Figure 3.18: 16-pin configuration of LCD display [19]

3.2.3 Microcontroller (ATmega32A)

ATmega32A is a low power CMOS 8-bit high performance microcontroller of Atmel's mega series of AVR family and it is based on enhanced RISC (Reduced Instruction Set Computing) architecture with 131 powerful instructions. The AVR core combines 32 general purpose working registers with these powerful instruction set where general purpose registers directly connected to the Arithmetic Logic Unit (ALU). One machine cycle is needed to execute most of the instructions [20]. Nowadays, microcontroller production raised up to billions per year and the controllers are integrated to use it in many sectors such as household appliances (microwave, washing machine, coffee maker,....), telecommunication (mobile phones), automotive industry (fuel injection, ABC,....), aerospace industry, industrial automation and so on [21].

Figure 3.19: Microcontroller (ATmega32A)

The ATmega32A has got following basic features [20] that are mentioned below:

1. Operating voltage is 2.7-5.5V.

2. Maximum frequency of 16MHz.

3. Programmable flash memory of 32 KB, static RAM of 2 KB and EEPROM of 1 KB.

4. Power Consumption at 1MHz, 3V, 25°C and active in 0.6mA.

5. Capable of handling analog inputs because of the availability of on-chip analog comparator.

6. Up to 16 MIPS throughput at 16MHz.

7. Two cycle on-chip multiplication and 32 × 8 general purpose working registers.

8. ATmega32 has three data transfer modules which are two wire interface, USART, serial peripheral interface embedded in it.

9. JTAG boundary scan facilitates on chip debug.

10. RC oscillator is internally calibrated and 32-bit programmable I/O lines.

11. There are six sleep modes such as idle, ADC noise reduction, power-save, power-down, standby, and extended standby.

3.2.3.1 Pin Configuration of ATmega32A

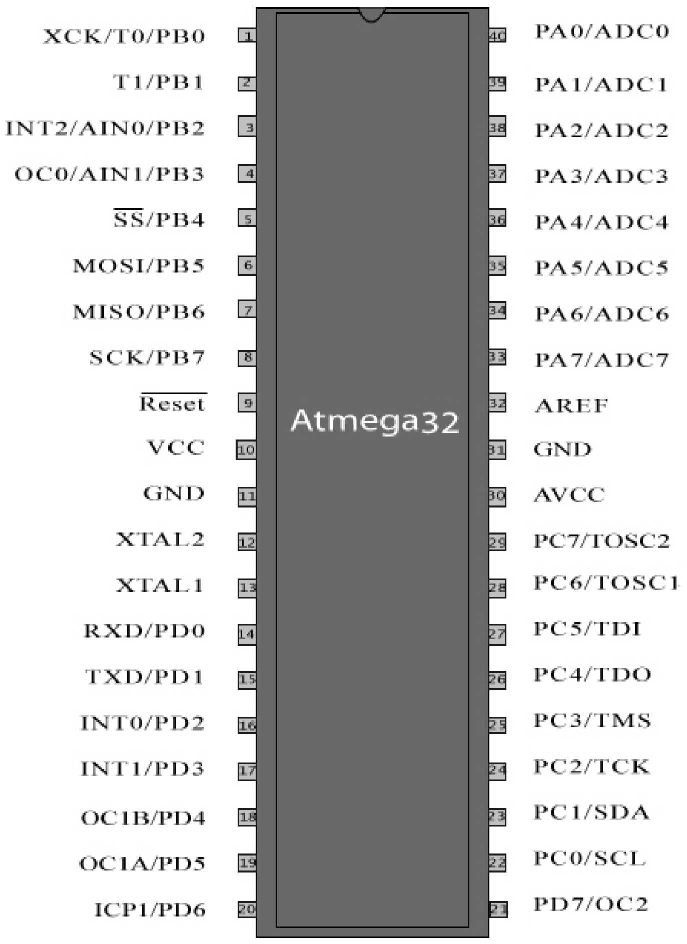


Figure 3.20: 40-Pinout of ATmega32A [22]

The figure 3.20 shows all the pin configurations of the microcontroller along with the functions of each pin where PORTA (PA7:PA0) serves the analog inputs to the A/D Converter. It can be also used as an 8-bit bi-directional I/O port, if A/D Converter is not used. In addition, there are also PORTB (PB7:PB0), PORTC (PC7:PC0), PORTD (PD7:PD0) which are also 8-bit bi-directional I/O port with internal pull-up resistors. XTAL1 pin refers the input to the inverting oscillation amplifier and input to the internal clock operating circuit whereas XTAL2 is the output from the inverting oscillation amplifier. The pin number 30, AVCC is the supply voltage for PORTA and A/D Converter. If the ADC is used, it should be connected to VCC through a low pass filter otherwise it should be externally connected to VCC. AREF is the analog reference pin for the A/D Converter.

Since ATmega32A microcontroller cannot take as input more than 5V, if we want to apply full charge of battery voltage which is 50.92V; we need to step down the voltage within the range 0-5V, otherwise microcontroller will be burnt. Hence, we can step down the voltage by using same range of 11 resistors of any value in series connection instead of transformer. Thus applying 50.92V in across to all registers and getting 5V from the last one register which will be fed into the PORTA ADC pin. Microcontroller converts the analog input to a digital one for executing any instruction will be discussed briefly in chapter 7.

3.2.4 Potentiometer

A potentiometer (POT) is an adjustable resistance within its range which can be changed when turn the shaft or wiper of the pot. Connect VCC to an outermost pin, Ground (GND) to the other, and the center pin will have a voltage depending on the rotation of pot that varies voltage from 0 to VCC [23]. In our project, we need to adjust the brightness of the LCD, so connect the LCD pin Vo to the center of the pot. Moreover, if we want to get fixed resistance then set the both outermost pin to the VCC and GND, hence there will be no connection in the center.

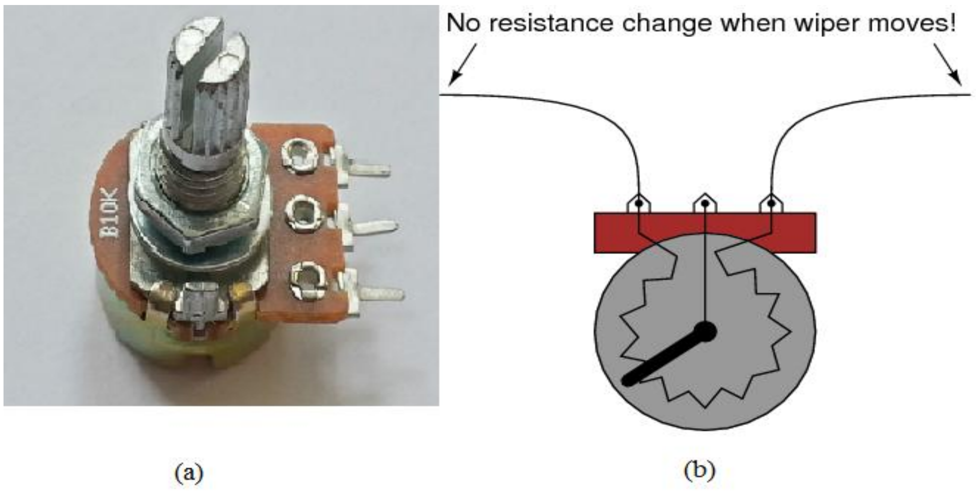


Figure 3.21: (a) 10K Potentiometer, (b) Potentiometer working as a fixed resistance

In this chapter, we discussed about how to burn algorithm in the microcontroller with all required external components. Shortly, all these hardware components needs to be connected in the breadboard according to their pin configuration. Thus, USBASP AVR Programmer is a device which is connected with the PCs USB port to burn the code into the microcontroller using Extreme Burner software; hence we can see our desired output such as SOC, remaining travelling time and coverable distance in the LCD display for our project purpose.

Chapter 4

Measuring field test data for developing the algorithm

4.1 Measuring motor current correspond to battery voltage

The cargo hauler (C5) was tested during discharging process, battery and current readings were taken by capturing the video during the entire field test period. Initially the battery was fully charged, 53V and the total running time of C5 was 74 minutes and 40 seconds with full load condition which was 466kg in any driving style and road condition. Battery voltage readings were measured using multimeter and clamp meter was used to measure current drawn by motor. After 20s interval, data was retrieved from the video. The retrieved data of field test for developing the algorithm which shows motor current correspond to battery voltage is given in appendix A.

4.2 Grouping data correspond to a constant range of motor current

From the field test data provided in appendix A, we grouped the motor current and battery voltage readings for a constant range of current. The varying motor current at different time intervals is divided in different motor current range for developing the algorithm based on the obtained characteristic equations from the graph plotted using MATLAB software. The field test data are grouped for different range of motor current is given in table 4.1, table 4.2, table 4.3, table 4.4, table 4.5, and table 4.6.

For 0-10A current:

Time (s)	Motor Current (A)	Battery Voltage (V)
160	0.2	51.3
280	0.1	51.9
300	0.1	52.0
320	0.3	52.0
500	0.5	51.2
620	0.5	50.6

640	1.7	50.0
800	2.3	50.4
900	5.5	50.6
920	1.1	51.5
1100	1.3	50.6
1160	8.6	50.1
1180	1.3	51.1
1200	1.3	50.8
1220	1.3	51.0
1240	1.3	50.7
1260	1.2	51.0
1340	2.1	50.2
1400	3.2	50.4
1420	1.5	50.7
1560	1.8	50.6
1580	1.8	51.0
1600	1.9	51.4
1640	6.8	49.5
1780	1.9	50.1
1820	1.5	50.0
1840	1.7	49.9
1880	3.5	50.2
1960	2.2	49.9
2020	2.6	49.8
2040	2.2	49.6
2140	2.0	50.0
2160	2.0	50.2
2240	2.0	49.6
2260	2.6	49.6
2300	2.0	49.6
2420	1.9	49.3
2440	0.1	49.8

2460	0.2	49.8
2640	3.7	48.9
2700	5.2	48.5
2740	0.3	49.0
2760	2.8	48.7
2800	0.3	48.8
2960	0.1	48.6
3000	0.1	48.4
3020	6.1	46.5
3220	0.4	47.8
3320	0.1	47.5
3520	0.2	47.5
3560	5.5	47.1
3760	0.4	47.3
3780	0.4	47.4
3800	0.3	47.5
3840	5.0	47.5
3880	2.5	48.2
3940	2.5	47.5
3980	2.4	47.6
4000	2.3	47.9
4020	2.4	47.9
4220	2.4	46.6
4240	1.9	47.2
4380	1.8	44.5
4480	1.4	44.8

Table 4.1: Field test data of cargo hauler for 0-10A motor current

For 10-20A current:

Time (s)	Motor Current (A)	Battery Voltage (V)
580	15.9	49.6
860	14.1	47.5
3160	16.6	46.7
3440	19.1	45.5
4420	13.6	43.0

Table 4.2: Field test data of cargo hauler for 10-20A motor current

For 20-30A current:

Time (s)	Motor Current (A)	Battery Voltage (V)
380	24.9	48.3
520	21.0	47.2
1140	29.8	47.8
1440	26.3	48.1
1460	21.5	48.0
1860	22.3	48.9
2080	29.2	46.4
2180	26.5	46.8
2200	20.8	47.7
2220	29.9	46.5
2360	23.5	47.3
3080	24.8	47.6
3260	26.2	44.1
3500	21.3	45.9
3760	26.0	43.5
3860	20.1	47.0
3960	27.8	45.9
4140	28.0	43.6
4160	27.7	43.6
4180	27.5	43.5

4200	27.4	43.3
4260	22.5	44.4
4280	29.4	43.3

Table 4.3: Field test data of cargo hauler for 20-30A motor current

For 30-40A current:

Time (s)	Motor Current (A)	Battery Voltage (V)
40	38.2	47.6
60	38.3	47.4
80	38.6	47.2
100	34.5	47.6
120	32.1	48.2
180	35.0	45.9
360	37.3	47.5
440	35.2	47.5
480	37.7	46.7
540	33.3	41.1
560	36.7	46.8
760	35.0	48.7
940	39.8	47.4
1360	39.3	46.3
1380	36.0	46.7
1760	36.5	46.1
2920	39.6	43.8
3580	33.6	43.4
3620	30.8	43.8
4080	35.8	43.0
4120	36.8	42.6
4360	37.9	39.4
4400	31.7	40.0
4440	36.4	39.3

Table 4.4: Field test data of cargo hauler for 30-40A motor current

For 40-50A current:

Time (s)	Motor Current (A)	Battery Voltage (V)
20	41.5	47.3
340	43.8	47.1
660	41.5	47.1
780	43.2	46.7
840	47.8	45.6
880	43.4	46.6
1280	41.4	46.5
1300	40.2	46.4
1320	40.2	46.7
1620	42.3	46.0
1800	47.0	45.2
2400	45.4	43.9
2480	41.7	46.5
2620	40.2	44.5
2840	43.3	44.0
3120	47.0	42.9
3360	46.8	42.1
3920	44.4	45.4
4340	42.8	38.9

Table 4.5: Field test data of cargo hauler for 40-50A motor current

For 50-60A current:

Time (s)	Motor Current (A)	Battery Voltage (V)
140	54.4	46.7
200	53.9	45.8
220	53.6	45.4
240	50.9	46.1
260	53.9	45.5
400	54.1	45.5
420	53.8	45.7

460	53.4	45.6
600	53.4	45.2
680	54.3	45.4
700	52.0	45.4
720	54.7	45.2
740	53.3	45.2
820	54.5	45.4
960	55.0	45.3
980	54.3	44.5
1000	54.8	45.2
1020	54.6	45.1
1040	53.4	45.5
1060	54.3	45.0
1080	54.2	45.0
1120	53.8	45.2
1480	55.3	44.8
1500	54.2	44.5
1520	51.6	45.3
1540	55.1	44.8
1660	54.3	44.6
1680	51.0	44.4
1700	55.4	44.5
1720	55.3	44.1
1740	55.4	44.2
1900	56.7	44.2
1920	55.0	44.5
1940	56.6	44.0
1980	50.3	44.7
2000	56.6	44.2
2060	55.0	44.0
2100	50.3	44.8
2120	55.5	43.6

2280	54.6	43.7
2320	55.4	43.4
2380	53.5	43.6
2500	52.0	46.6
2520	54.7	43.7
2540	54.9	43.3
2560	53.8	43.3
2580	54.1	43.2
2600	53.5	43.6
2660	54.3	43.2
2680	53.9	43.1
2720	54.1	43.0
2780	54.4	43.1
2820	54.0	42.9
2860	54.1	42.7
2880	53.5	42.4
2900	52.8	43.5
2940	53.8	42.2
2980	53.7	42.7
3040	54.5	42.7
3060	54.2	42.1
3100	54.4	42.0
3140	54.5	42.2
3180	53.9	41.7
3200	53.8	41.9
3240	53.0	41.8
3280	54.2	41.6
3300	53.0	41.7
3340	54.2	41.6
3380	53.5	41.2
3400	53.3	41.2
3420	53.4	40.6

3460	50.4	41.5
3480	53.6	40.8
3540	55.4	41.9
3600	55.0	40.7
3640	50.5	41.0
3660	54.1	40.4
3680	55.0	40.6
3700	54.6	40.3
3720	54.5	39.9
3820	58.1	42.9
3900	57.7	42.1
4060	56.5	40.6
4100	50.7	41.1
4300	56.2	39.6
4320	50.3	38.0

Table 4.6: Field test data of cargo hauler for 50-60A motor current

4.3 Field test curves and equations of grouping data

Based on the real-world data mentioned above in table 4.1, table 4.2, table 4.3, table 4.4, table 4.5 and table 4.6, the basic algorithm is written in MATLAB to plot the curves of the battery voltage at different time for the constant range of motor current. After simulating the algorithm, the graph will be plotted with the data for different range of motor current. From the tools icon, we select the basic fitting option to get the characteristic equation. A new window has come after selecting the basic fitting option; now we need to select the graph shape (linear) and "Show equations" options as well as significant digit number based on our requirement. Then close the basic fitting window to get our desired graphs with characteristic equationwhich are given in figure 4.1, figure 4.2, figure 4.3, figure 4.4, figure 4.5 and figure 4.6 accordingly. The algorithm is also provided for different range of motor current in appendix B.

For 0-10A current:

After plotting the data for 0-10A current, we get the blue line and take the best fit red line which is linear and the best fit line is occurred when we get the characteristic equation. The general linear equation formula is y=m*x+c where m=slope and c=constant. As we also get the linear characteristics for 0-10A current and the linear equation obtained is y=-0.0012824*x + 52.23; where x= Time, y= Battery Voltage, m= -0.0012824 and c= 52.23.

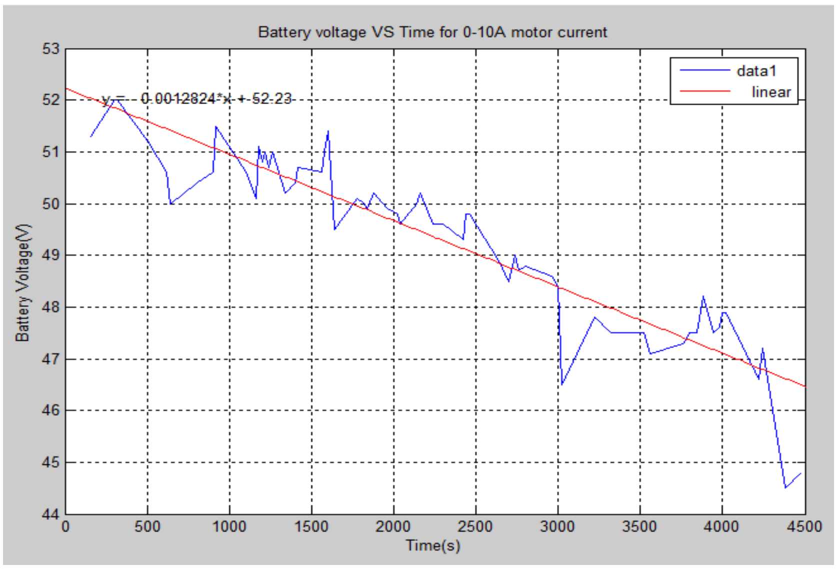


Figure 4.1: Battery voltage at different time for 0-10A motor current

For 10-20A current:

The linear equation obtained for 10-20A current is y= -0.0013228*x + 49.776; where x= Time, y= Battery Voltage, m= -0.0013228 and c= 49.776.

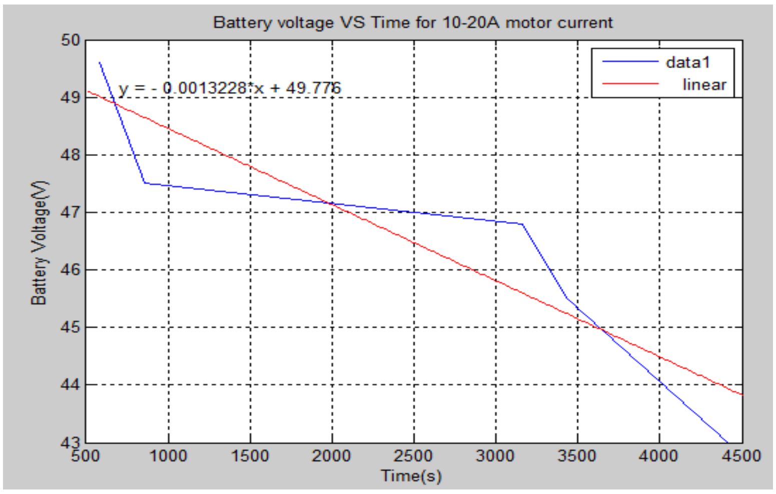


Figure 4.2: Battery voltage at different time for 10-20A motor current

For 20-30A current:

The linear equation obtained for 20-30A current is y= -0.0012236*x + 49.461; where x= Time, y= Battery Voltage, m= -0.0012236 and c= 49.461.

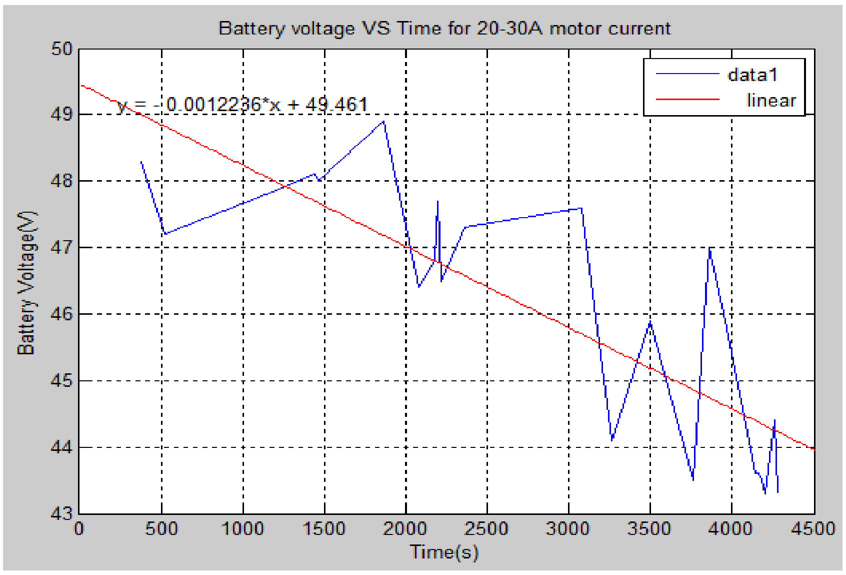


Figure 4.3: Battery voltage at different time for 20-30A motor current

For 30-40A current:

The linear equation obtained for 30-40A current is y= -0.0014394*x + 47.606; where x= Time, y= Battery Voltage, m= -0.0014394 and c= 47.606.

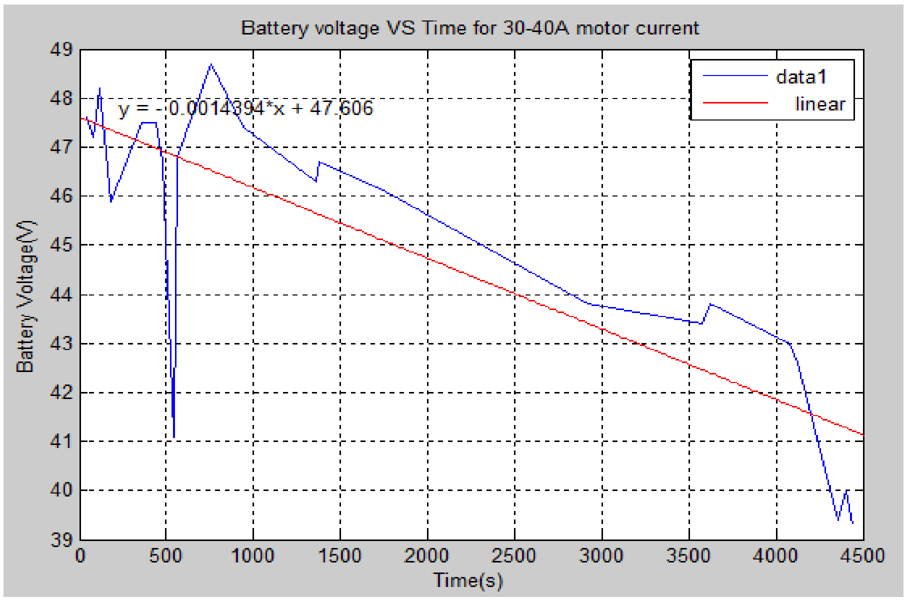


Figure 4.4: Battery voltage at different time for 30-40A motor current

For 40-50A current:

The linear equation obtained for 40-50A current is y= -0.001415*x + 47.907; where x= Time, y= Battery Voltage, m= -0.001415 and c= 47.907.

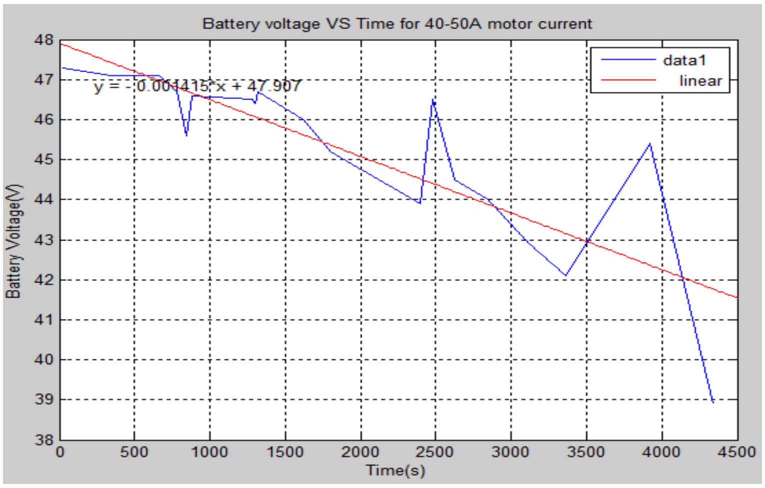


Figure 4.5: Battery voltage at different time for 40-50A motor current

For 50-60A current:

The linear equation obtained for 50-60A current is y= -0.0014653*x + 46.706; where x= Time, y= Battery Voltage, m= -0.0014653 and c= 46.706.

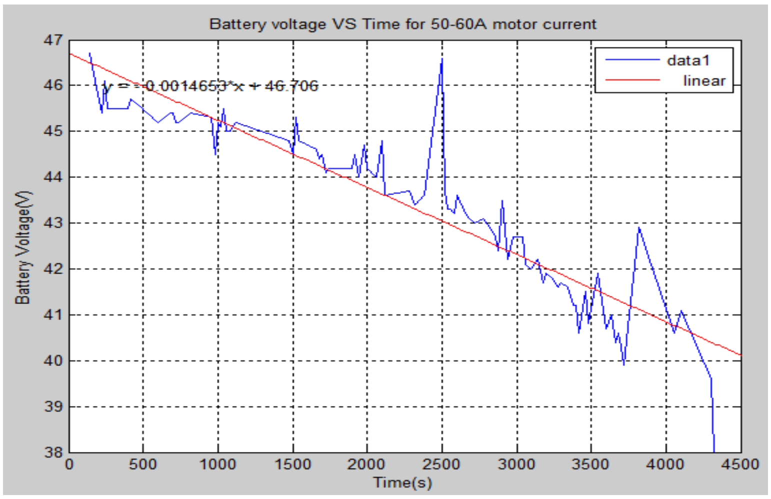


Figure 4.6: Battery voltage at different time for 50-60A motor current

For developing the desired algorithm for our project purpose, all these obtained equations are needed to find the available travelling time means how much time the electric vehicle can travel with the current battery voltage remaining.

Chapter 5

Data Acquisition Card

5.1 Introduction

Advantech USB-4716 has been popular for its user friendly behavior is also known as *DATA ACQUISITION CARD*or DAQ card. Mainly DAQ card converts the analog input into a digital value which can also be controlled by a computer with its required driver software. It can also be used to obtain analogue output signal after providing digital input value. As it is an alternative of traditional serial and parallel device, it does not need any external power source; only USB power is enough to communicate with the system. Advantech USB data acquisition module fits consistently into every industrial application because of its reliability, accuracy, multiple mounting options, plug and play system, screw-in fuses of the cable, and developed monitoring functions. The given figure 5.1 shows the Advantech USB-4716 which is a portable data acquisition module.

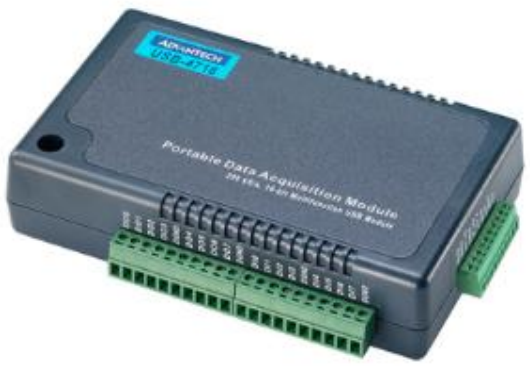


Figure 5.1: Advantech USB-4716

Advantech USB-4716 has some important features [24] related to its measurement and control functions as mentioned below-

1. USB-4716 has 16 analog input channels and 2 analog output channels.

2. 16-bit resolution of Analog to digital converter (A/D converter) where sampling rate is up to 200 KS/s.

3. It has 8 digital input channels.

4. It has 8 digital output channels.

5. It has programmable 16-bit counter/timer x 1 and programmable gain is required for each analog input channel.

6. Support USB 2.0.

7. LED indicators indicate about the device status.

8. FIFO buffer of 1K samples for Analog input channels.

9. It has auto calibration function and automatic channel/gain scanner.

10. Hot swappable.

11. One lockable USB cable is included for secure the connection.

12. Bus powered and portable.

13. There are detachable screw terminal on modules.

5.2 Pin configuration of DAQ card

The figure 5.2 gives the pin assignment for Advantech USB-4716 where the pin number AI0-AI15 is used for 16 analog input channels, AO0-AO1 is assigned for analog output channels, DI0-DI7 are used for 8 digital inputs and DO0-DO7 is dedicated for 8 digital output channels. For the use of analog input/output channels, we need to connect AGND port of the DAQ card for connecting analog input/output ground. In addition, the pin DGND has to be used for the digital ground which is for digital input/output channels. One EXT_TRG pin is also there which is for A/D external trigger. This pin is the external trigger signal input for A/D conversion, where A/D conversion starts when the signal goes from low to high edge triggered. GATE pin is for A/D external trigger gate; the EXT_TRG will be disabled if gate is connected with 5V. Furthermore, EVT_IN and P_OUT are for external events input channel and pulse output channel respectively. For interfacing between analog and digital system, we have to use separate ground DGND and AGND; otherwise DAQ card can be damaged.

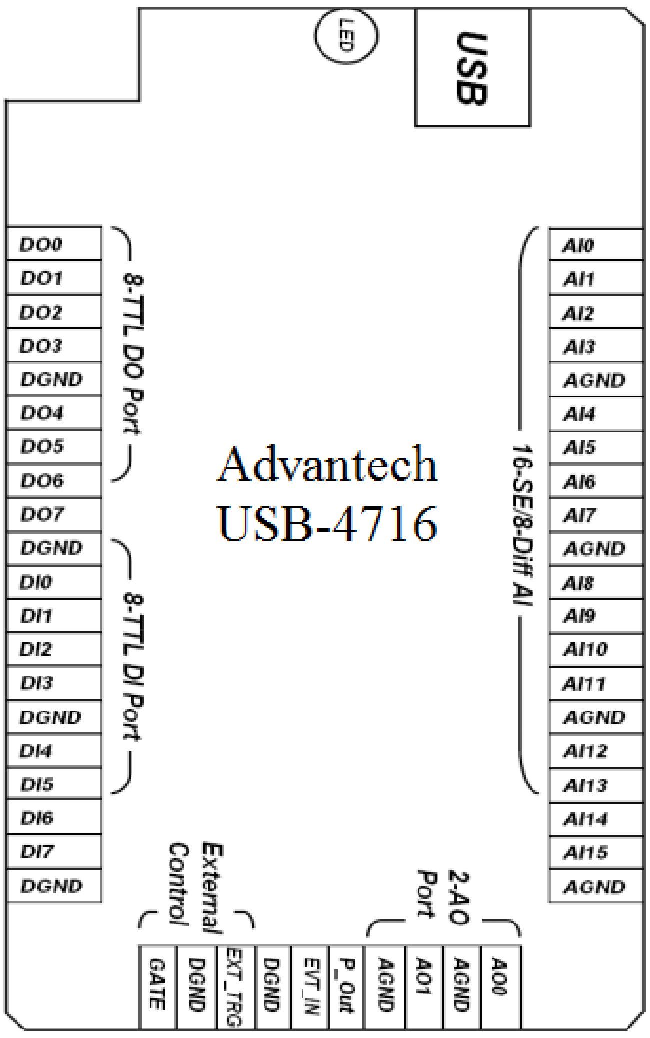


Figure 5.2: Pin assignment for Advantech USB-4716 [1]

5.3 Working principle of DAQ card

Three internal components are present in the DAQ card which mainly converts the real world analog input into a digital numeric value that can be shown in the computer. That internal elements of the DAQ card are transducer or sensor, signal conditioning and A/D converter. Firstly, the transducer or sensor sense and translate the variables of real world analog data into an electrical signal for an analog system and vice versa for digital system which signal it is translated. Then the signal conditioning makes a signal to be prepared to be converted into a numeric digital value. Mainly the signal conditioning block received the signal from the sensor and do amplification of the signal, filtering, multiplexing and isolation procedure based on the necessity of the DAQ card. Finally, A/D converter works like a built-in function in the DAQ card which is a combination on some control and logic registers as well as comparator that converts the analog input to a digital value [25]. The figure 5.3 shows how data acquisition card gives a result in digital value with a computer monitoring system.

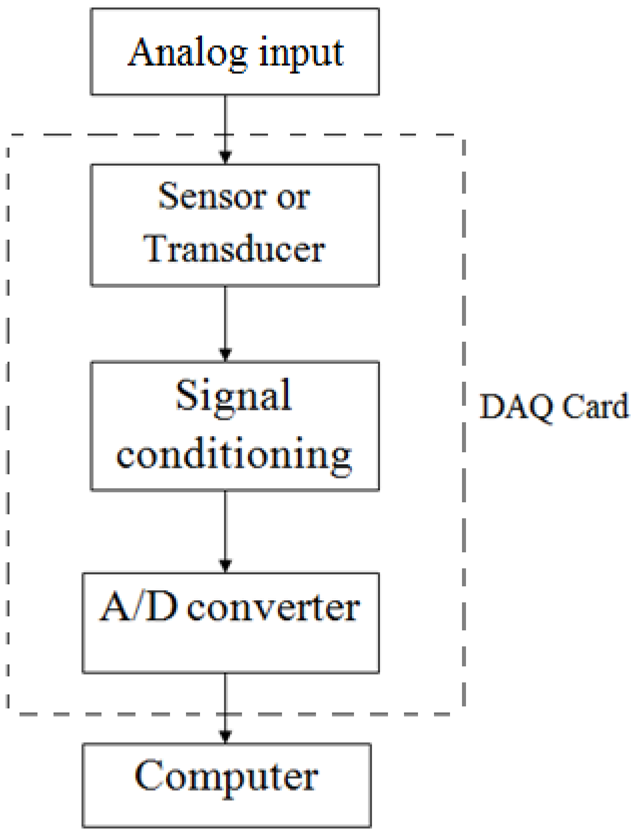


Figure 5.3: Internal working process of DAQ card

It should be mentioned that the DAQ card can only take readings up to 15V. If we need to apply more than 15V, a voltage divider circuit must be designed separately in order to step down the voltage before passing the voltage as input to the input port of the DAQ card.

5.4 DAQ device driver installation

DAQ device driver software must be installed in the PC to work with the DAQ card. Before the installation of USB-4716 module into the PC, we need to install first the device driver software. There are three setup files for the device driver software which are PCI1711L.exe, PCI_1747U_All_Examples_32bit.exe and USB4716.exe; all these setup files are required for 32-bit Windows 7. After the installation of the software, we need to install Advantech Device Manager_32bit.exe setup file to see our desired result in the PC. For this, we have to connect the DAQ card with the PC and then click on the TEST button from the Advantech Device

Manager. Hence, a different window will come up to show our desired output voltage of the current sensor.

5.5 The application of the DAQ card

The role of DAQ card in the project is to show output voltage of the current sensor up to at least 7 decimal places with precision and accuracy.DAQ card is connected with the current sensor whereone of the analog ports AI0 is connected with current sensor VOUT pin.Thus, the ground (GND) of the current sensor is also connected with the analog ground (AGND) of the DAQ card. The figure 5.4 gives the connection diagram of DAQ card with the current sensor.

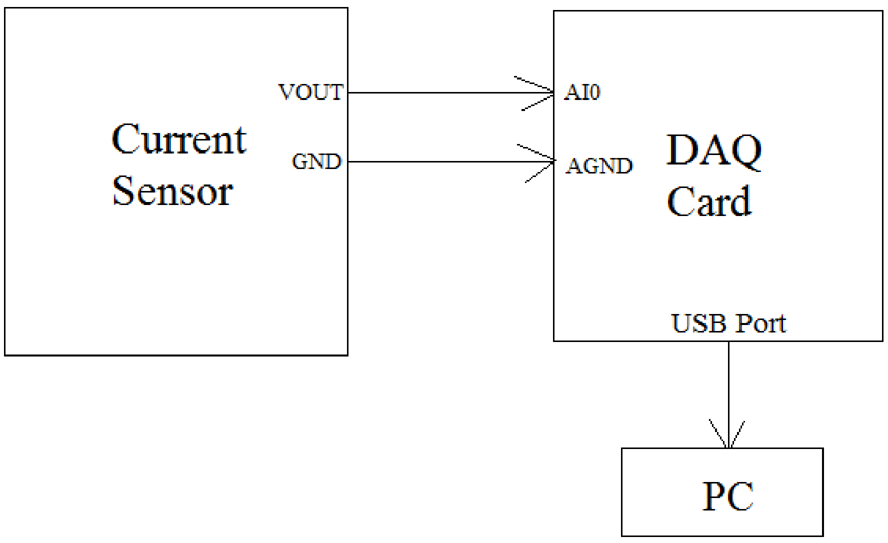


Figure 5.4: Connection diagram of DAQ card with the current sensor

Chapter 6

Working principle and application of current sensor to measure voltage from input current

6.1 Current sensor (ACS758)

This group of ACS758 current sensor ICs supports for AC or DC current (0 to 100A) which convert the dc input current readings into a corresponding output voltage. This device comprises of low offset linear hall circuit with a copper conduction path where the applied input current will flow through this copper conduction path and creates a magnetic field which changes over the magnitude of the current and resulting a corresponding voltage. At high over current condition, this thickness of the copper conduction allows the protection of the device. There are numerous utilizations of ACS758 current sensor such as motor control, DC to DC converter control, load discovery and management, power supply, inverter control, and overcurrent fault detection [26].

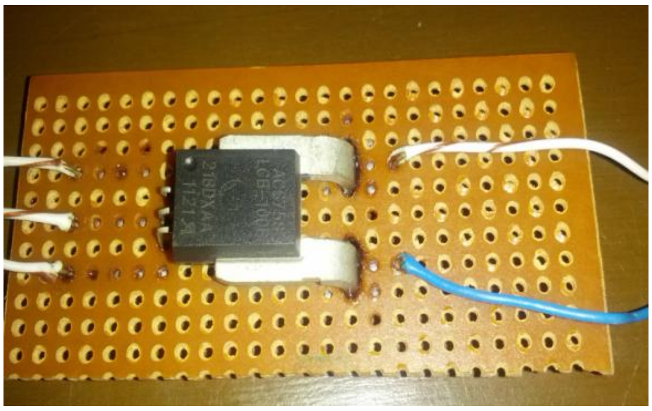


Figure 6.1: Current Sensor (ACS758)

In general current sensors have some important features [26] that are mentioned below-

1. High reliability assures Monolithic Hall IC and AEC Q-100 qualified.
2. 100 μΩ internal conductor resistance is for ultra-low power loss.
3. Galvanic isolation permits use in economical, high-side current detecting in high voltage frameworks.

4. Single supply operation voltage range is from 3.0 to 5.5V.
5. Stable output offset voltage and bandwidth is 120 KHz.
6. Magnetic hysteresis is close to zero.
7. Integrated shield significantly diminishes capacitive coupling from current conductor to pass on because of high dV/dt signals which prevents offset drift in high-side and high voltage applications.
8. 3 µs output rise time corresponds to step input current.
9. Package size is small with simple mounting ability.

There are 8 models of ACS758 such as X050B, X050U, X100B, X100U, X150B, X150U, X200B and X200U. We have used ACS758 (X100B) model of current sensor in out project. The features of this model is given below-

1. Input current range is -100 to 100A and nonlinearity is from -1.25 to +1.25%.
2. Typical noise scale is 6mV and output offset voltage is 2V.
3. Sensitivity is 10mV/A [26]. The change in voltage is 10mV for every 1A increase in current, and this change can only be measured using high precision digital multimeter/DAQ card.

6.1.1 Pin configuration of ACS758

There are 5 pins in ACS758 which are VCC, GND, VOUT, IP+ and IP-, where VCC refers to device power supply terminal, GND is signal ground terminal, VOUT is analog output signal and IP+, IP- is terminal for input current being sampled. Terminal 4 and 5 are used to take input current and converted voltage will get from terminal 3.

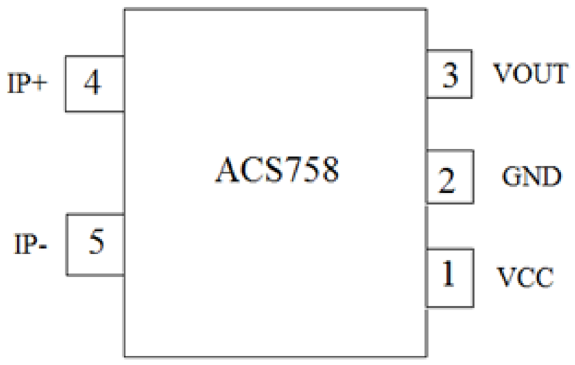


Figure 6.2: Pin configuration of current sensor, ACS758 [26]

6.1.2 Working principle of current sensor

Magnetic sensors are composed with an extensive variety of positive and negative magnetic fields whose output signal is a function of magnetic field density around it known as Hall Effect Sensor. The sensor detects a magnetic flux density if it surpasses a pre-set threshold voltage and gives an output voltage named Hall Voltage, V_H[27]. The input DC current is passes through a shunt which is extremely low resistance like a couple of milliohms. When the shunt is placed to get a current path, it creates a little insertion loss. A magnetic path is completed when ferrite core wrapped around the shunt path. Based on the strength of the magnetic field by the current flowing through the shunt, the current sensor produces a output voltage within 5V which can be connected with an A/D converter, microprocessor, or microcontroller [28]. For our project purpose, we gave the current sensor output voltage in a microcontroller. The Hall Effect is the generation of a voltage difference (the Hall voltage) over an electrical conductor where transverse an electric current which creates a magnetic field perpendicular to the current; this is the working principle of current sensor.

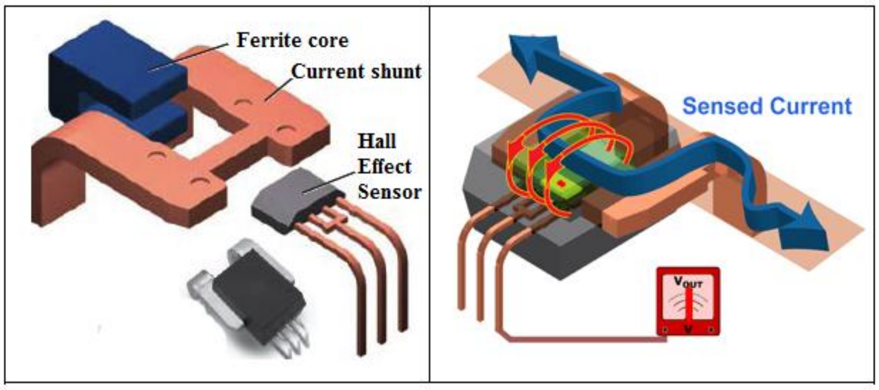


Figure 6.3: Hall Effect current sensor [28]

6.2 ACS758 connection diagram

The positive terminal of the battery is connected with the pin number 4 of the current sensor and the negative terminal is connected with the pin number 5 of ACS758 through the load. For the electric vehicle, motor is acting as a load. However, we used an electrical stove in the lab as a load and the output voltage from the current sensor is changing with the changeof the load resistance and in turn load input current changes occurred. Moreover, 5V power supply is needed to power up the ACS758, where VCC is connected with the positive terminal and ground is connected with the negative terminal.

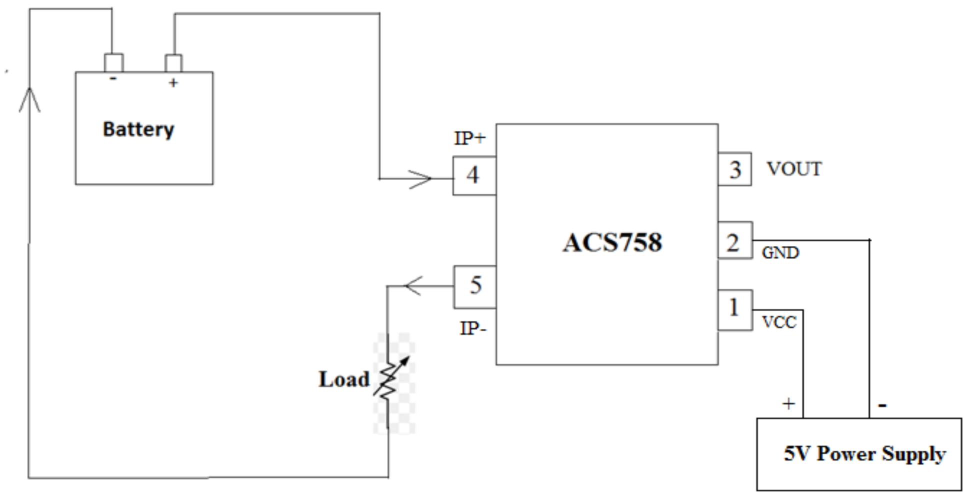


Figure 6.4: Connection diagram for current sensor, ACS758

6.2.1 Current sensor connection with electrical stove as a load

In the laboratory, the current sensor has been connected with two electrical stoves as a load to draw current from 0-20A. The current consumed by the load is varied by the help of heat controller. Changing the current has allowed the current sensor to give corresponding output voltage. This voltage is later used up for implementing required algorithm to show certain parameters using the micro-controller based device. Furthermore, we used 48V battery where four 12V, 20AH are connected in series. The figure 6.5 represent the series connection of battery to make 48V battery for our project.

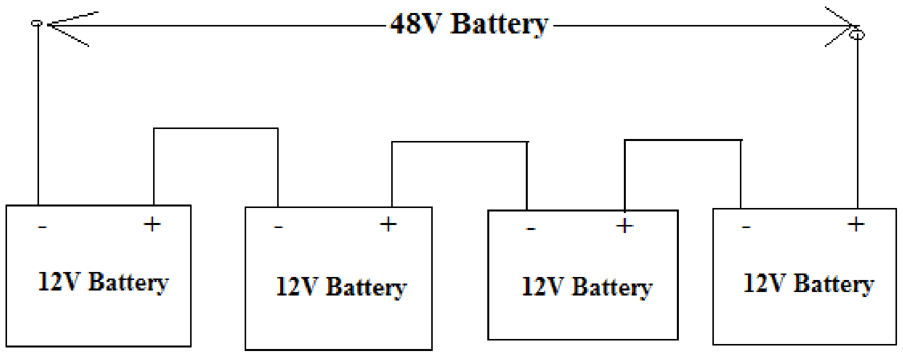


Figure 6.5: Four 12V battery connected in series

The positive terminal of 48V battery is connected with the pin number 4 of the current sensor and the pin number 5 of the current sensor is connected with the whole load. Again, the negative terminal of the battery is connected with the negative power of the heat controller.The positive and negative terminal of the load of the heat controller are connected with the heating coil of each stove. Hence, these two heat controllers are enabling to draw close to 20A of total current from both stoves, for each stove maximum 10.6A can be drawn. Moreover, 5V power supply is needed to power on the current sensor. The pin number 1 of the current sensor is VCC which is connected with the 5V and the pin number 2 is connected with the ground. Analog input channel from the data acquisition card is connected with the pin number 3 (VOUT) of the current sensor and analog ground (AGND) is connected with the common ground. The figure 6.6 shows the connection diagram of the current sensor with the electrical stove.

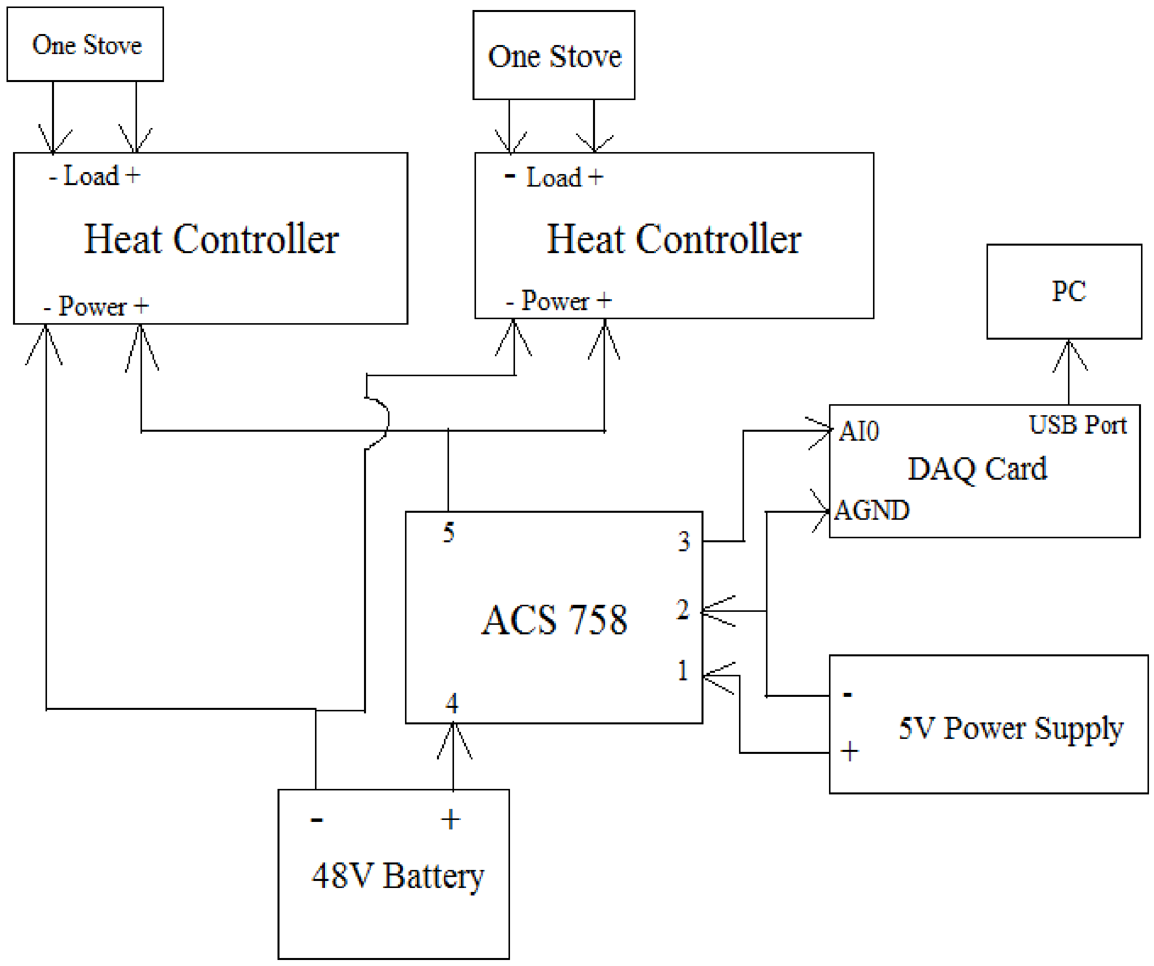


Figure 6.6: Connection diagram of the current sensor with electrical stove

Our given connection in the laboratory to test the current sensor that is following the given connection diagram in figure 6.6 which is given below in figure 6.7. The mentioned multimeter and clampmeter was used to measure battery voltage and input current respectively.

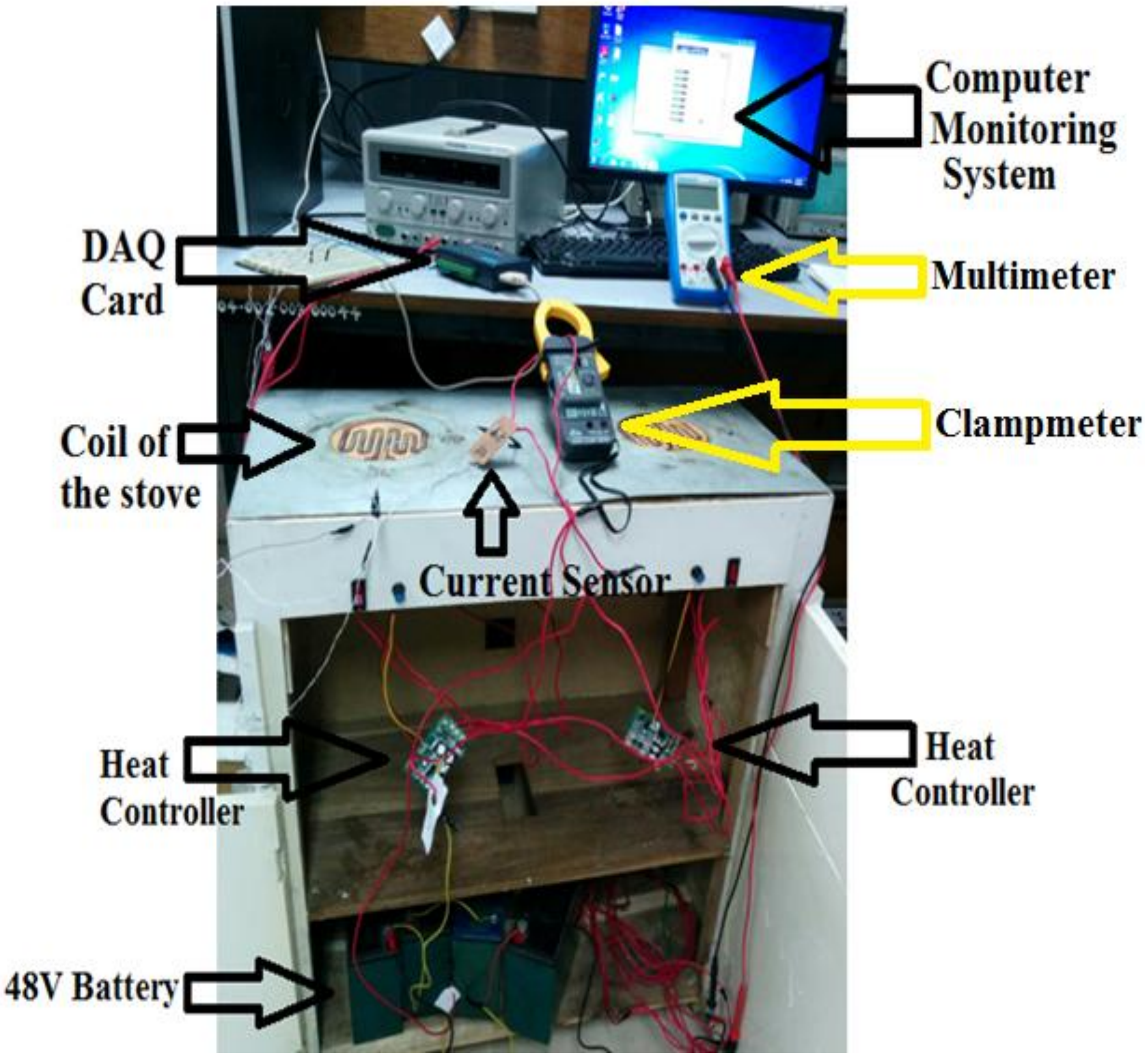


Figure 6.7: Practically given connection diagram of the current sensor with electrical stove

6.3 Simulation data analysis of current sensor (ACS758)

For every 1A current, the change in output voltage (VOUT) of the current sensor is 10mV which is 0.01V and note that the offset voltage is around 2V. Thus to see the change in the output voltage of the current sensor for the corresponding input current; we used DAQ card (Advantech USB-4716) because of its high precision and greater accuracy in measuring the voltage. The reason of choosing DAQ card over multimeter that is multimeter cannot show the change in voltage in very low mV value. As per datasheet we know that, maximum output voltage of the current sensor is 5V for 100A current so the tabular results showing such small changes in output voltage is acceptable. Initially, the battery voltage was 50.1V and after the test, the battery voltage was 49.8V which is close to 60% SOC, and the current obtained has also reached nearly 20A at this battery voltage. While conducting the practical test of the current sensor, we measured the VOUT at every 2A current intervals to observe noticeable variations of the output voltage from the PC using DAQ card. The simulation data observed in the test of the current sensor for 20A current is given below in table 6.1.

Current (A)	Current Sensor Output Voltage (V)
02	2.0236210
04	2.0460510
06	2.0573430
08	2.0687870
10	2.0906770
12	2.1131900
14	2.1426390
16	2.1598820
18	2.1748350
19.2	2.1815490

Table 6.1: Simulation data of the current sensor

The simulation data of the current sensor is used in MATLAB software to plot the graph of output voltage VS current, where x is the current and y is the output voltage of the current sensor. The code for the plotted graph of figure 6.6 is given in appendix B. From the basic fitting, we got the best fit line and equation for the linear characteristics. The equation of this graph is y= 0.0095486*x + 2.0016, here 0.0095486 is the slope and 2.0016 is the y intersect constant voltage at 0A current. Theoretically we know that 0.01V change for 1A current and from the practically tested data we got 0.0095486V for the change of 1A current.

The error calculation of the slope is given below-

$$\text{Error} = \frac{0.01-0.0095486}{0.01} * 100 = 4.5\%$$

The error calculation of the threshold voltage is also given below-

$$\text{Error} = \frac{2.0016-2.00}{2.00} * 100 = 0.08\%$$

The plotted graph with the data of the current sensor is given below in figure 6.8.

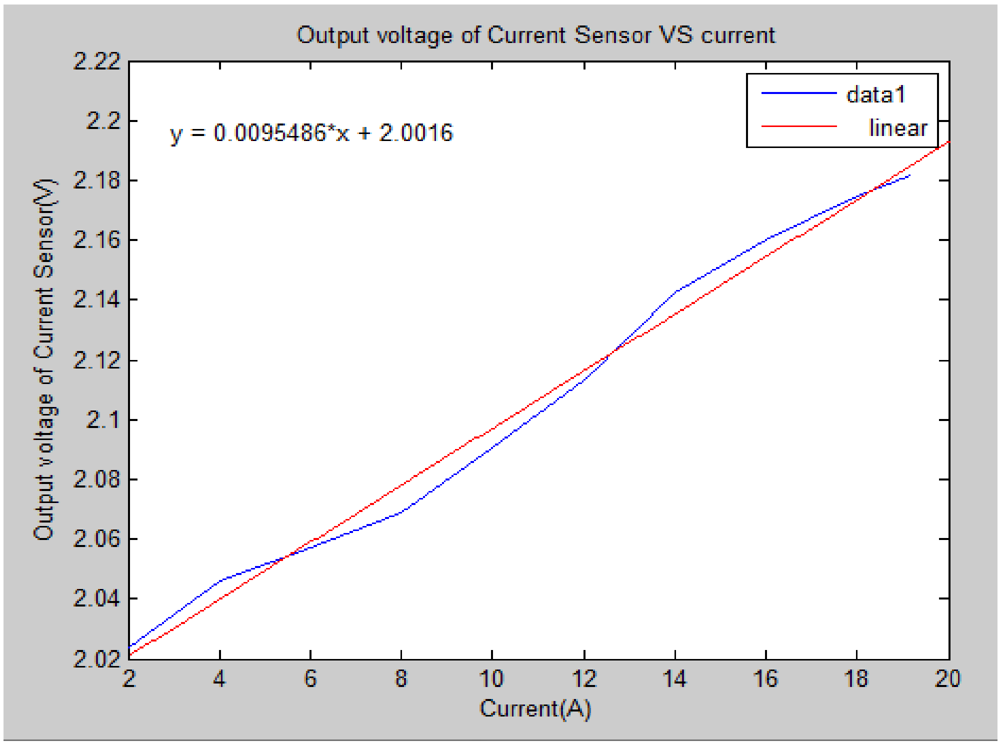


Figure 6.8: Output voltage of current sensor at different current interval

This chapter is all about how a current sensor response with a load and the battery as well as the characteristics of a current sensor.

Chapter 7

Algorithm development for SOC, remaining discharging time and distance on LCD display

7.1 State of Charge (SOC)

The term SOC is a critical parameter for the electric vehicle which refers to state of charge and it is a measurement of how much electrical energy stored in the battery. Measuring the estimated SOC of an electric vehicle is a troublesome and error prone procedure. The accuracy of determining SOC is relies on upon numerous factors as the battery is influenced by ecological temperature, charging time, discharging ratio, age of the cell and so on [29].

SOC Chart

The units of SOC are calculated in percentage where 0% is no charge and 100% means completely charged-up the batteries. We need the term SOC to know the current state of a battery while the battery is in use. SOC measurement is not accurate always, so SOC chart obtained from online source is using as a base guideline. The battery voltage is measured when the current has become approximately zero while charging the batteries. As we have taken data for cargo hauler; they used four 12V batteries which means total 48V. A fully charged 48V battery has 50.92V and half discharged battery showed 48.10V across its terminal. To increase the lifetime of battery, we need to charge up the batteries again when it discharging to 50% of charge. The SOC chart is given in table 7.1.

Charge (SOC)	12V battery	48V battery
100%	12.73	50.92
90%	12.62	50.48
80%	12.50	50.00
70%	12.37	49.48
60%	12.24	48.96
50%	12.10	48.40
40%	11.96	47.84
30%	11.81	47.24

20%	11.66	46.64
10%	11.50	46.04

Table 7.1: SOC chart of lead acid battery [30]

7.1.1 Analog to digital (A/D) converter of ATmega32A

The real world is full of analog signals like sound, light, temperature, humidity, all are continuously variable signals. However, computing system recognize the digital signal by converting the analog signal to a binary number. Thus for giving analog input into a digital or computing system, we need to convert the analog value to a digital value. This type of conversion is carried out by the analog to digital converter (ADC) and the process of converting is known as analog to digital conversion.

In our project, we are using AVR ATmega32A microcontroller which can convert the analog signal to a digital signal. The AVR ATmega32A has inbuilt 8 ADC pin in PORTA which is 10-bit analog to digital converter. There are three ADC registers such as ADC multiplexer selection register (ADMUX), ADC control and status register A (ADCSRA) and the ADC data register (ADCL and ADCH) where ADMUX is for selecting the reference voltage and the input channel, ADCSRA controls the status of ADC and ADCL, ADCH store the final output in this register. When 5V, 2.5V and 0V signal is converted, the output of the ADC is 0x3ff (1023), 512 and 0 respectively [31]. As microcontroller take input within 5V, so the output will be in-between 0 and 1023 for any other voltage levels.

7.1.2 Chart of step down and ADC voltage correspond to battery voltage

From the SOC chart for 48V battery, the highest voltage is 50.92; hence we have to use any 11 resistors (we used 560k) of same value to step it down to 5V. The input voltage will be given in first and last resistor terminal and correspond to last one resistor we will get the step down voltage. The equation of step down voltage for any input analog voltage is:

$$\frac{Resistor\ value \text{ x input analog voltage}}{Resistor\ value \text{ x number of resistor}} = Step\ down\ voltage$$

For example, the calculation of step down voltage for 50.92V is given below-

$$\frac{560k \text{ x } 50.92}{560k \text{ x } 11} = 4.63V$$

In this way, we can also calculate for other voltage levels and note that this is the theoretical value of step down voltage within 5V. We measured practically the step down voltage which is the input of the microcontroller and it convert the analog input to a digital number by the ADC built-in function of the microcontroller. The ADC value can be change because of the presence of noise, but the value will be close to the data mentioned below in table 7.2. The correspond ADC and step down voltage for the SOC chart of 48V battery is given below in table 7.2.

SOC (%)	Analog input voltage, V	Step down voltage, V (Theoretical)	Step down voltage, V (Practical)	ADC value (Practical)
100	50.92	4.63	4.58	956
90	50.48	4.59	4.54	947
80	50.00	4.55	4.50	938
70	49.48	4.50	4.45	929
60	48.96	4.45	4.41	918
50	48.40	4.40	4.35	908
40	47.84	4.35	4.29	897
30	47.24	4.29	4.25	885
20	46.64	4.24	4.19	873
10	46.04	4.19	4.14	863

Table 7.2: ADC and step down voltage for the SOC chart of 48V battery

7.2 Equation establishment for measuring discharging time of battery from the field test data

From the field test data of cargo hauler, we obtained a couple of equations for different range of current which is already mentioned in chapter 4. In this section, we are establishing the equations for developing the algorithm to measure discharging time of the battery. The variable x is for time and y is for battery voltage.

For 0-10A current:

y=-0.0012824*x + 52.23

$$\Longrightarrow x = \frac{y-52.23}{-0.0012824}$$

For 10-20A current:

y = -0.0013228*x + 49.776

$$\Longrightarrow x = \frac{y-49.776}{-0.0013228}$$

For 20-30A current:

y = -0.0012236*x + 49.461

$$\Longrightarrow x = \frac{y-49.461}{-0.0012236}$$

For 30-40A current:

y = -0.0014394*x + 47.606

$$\Longrightarrow x = \frac{y-47.606}{-0.0014394}$$

For 40-50A current:

y = -0.001415*x + 47.907

$$\Longrightarrow x = \frac{y-47.907}{-0.001415}$$

For 50-60A current:

y = -0.0014653*x + 46.706

$$\Longrightarrow x = \frac{y-46.706}{-0.0014653}$$

7.3 The idea and equation establishment for displaying the distance

The data we have taken from the field test of the cargo hauler was the battery voltage and the motor current which did not include any data from that we can relate distance with voltage or distance with current. Thus, no mathematical relation or formula could be estimated for calculating the remaining distance with the remaining battery supply and the current load of the electric vehicle. Therefore, the idea was to calculate theremaining distance using the average speed of the vehicle and the remaining time it could run. Itseemed logical as the average speed was constant and was going to be multiplied with the remainingtime. Hence, we can approximate our desired output in the display which will not be exactly accurate. It would be more accurate if the data we can take which indicate the relationship between distance and voltage or distance and current. Though it was notpossible because of time constraints and less scope to go for field test. The whole distance we travelled with the cargo hauler is 18km and it takes 4480 seconds. The average speed is determinedby dividing the total distance travelled with the total time taken to travel that distance during the field test. The equation of the average speed is given below-

$$\text{Average speed} = \text{Total distance travelled}/\text{Total time taken}$$

$$= 18000/4480 = 4.01786 \text{ m/s}$$

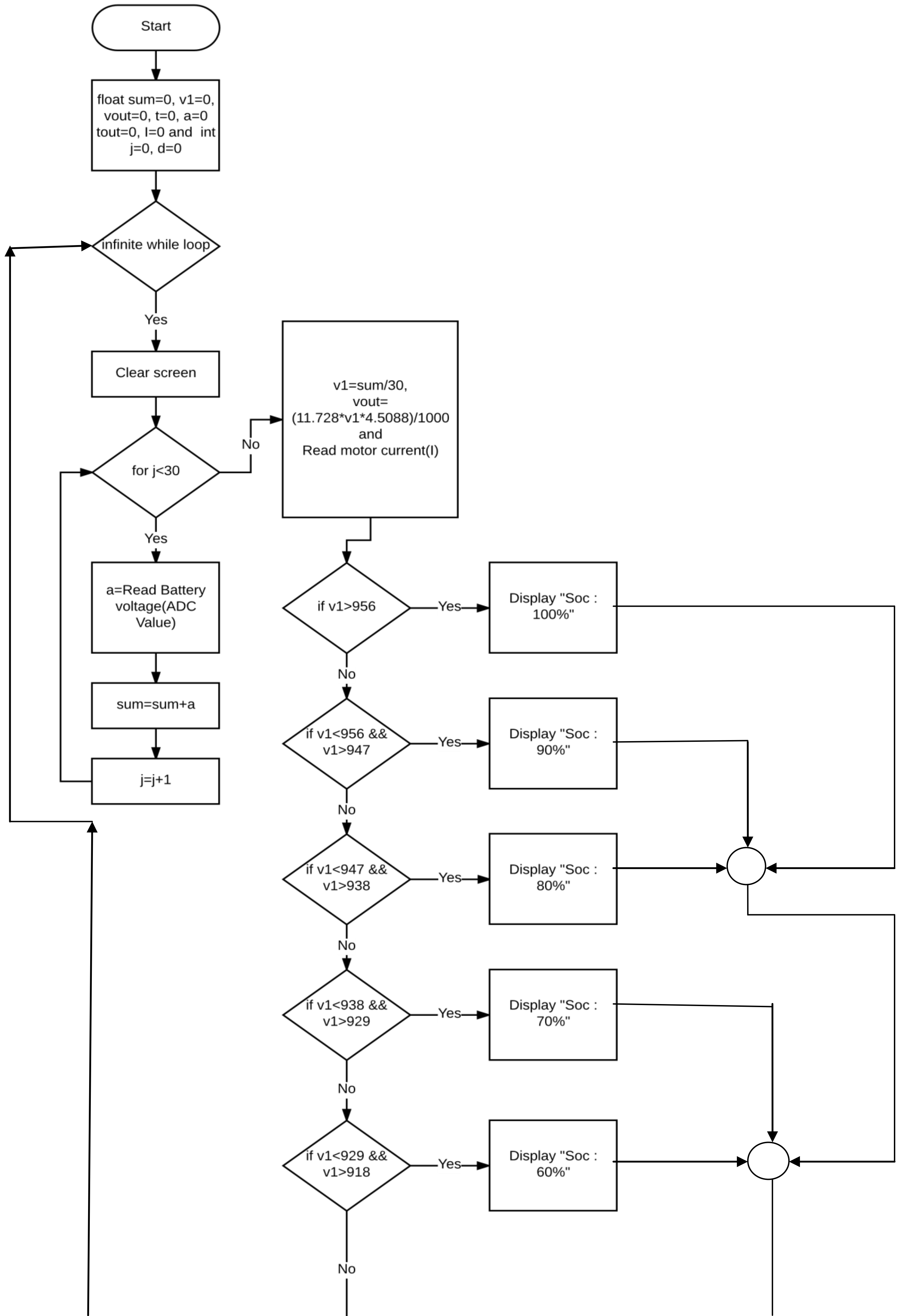
We found the average speed is 4.01786 m/s. The remaining time that the vehicle can run is calculated beforehand using different equationsdepending on current. It is briefly mentioned in the previous section how the equations are derivedfrom field test data and used to calculate the remaining time.Finally, remaining distance is calculated bymultiplying the average speed of the vehicle with the remaining time calculated earlier in the section 7.2. The equation for the remaining distance is given below-

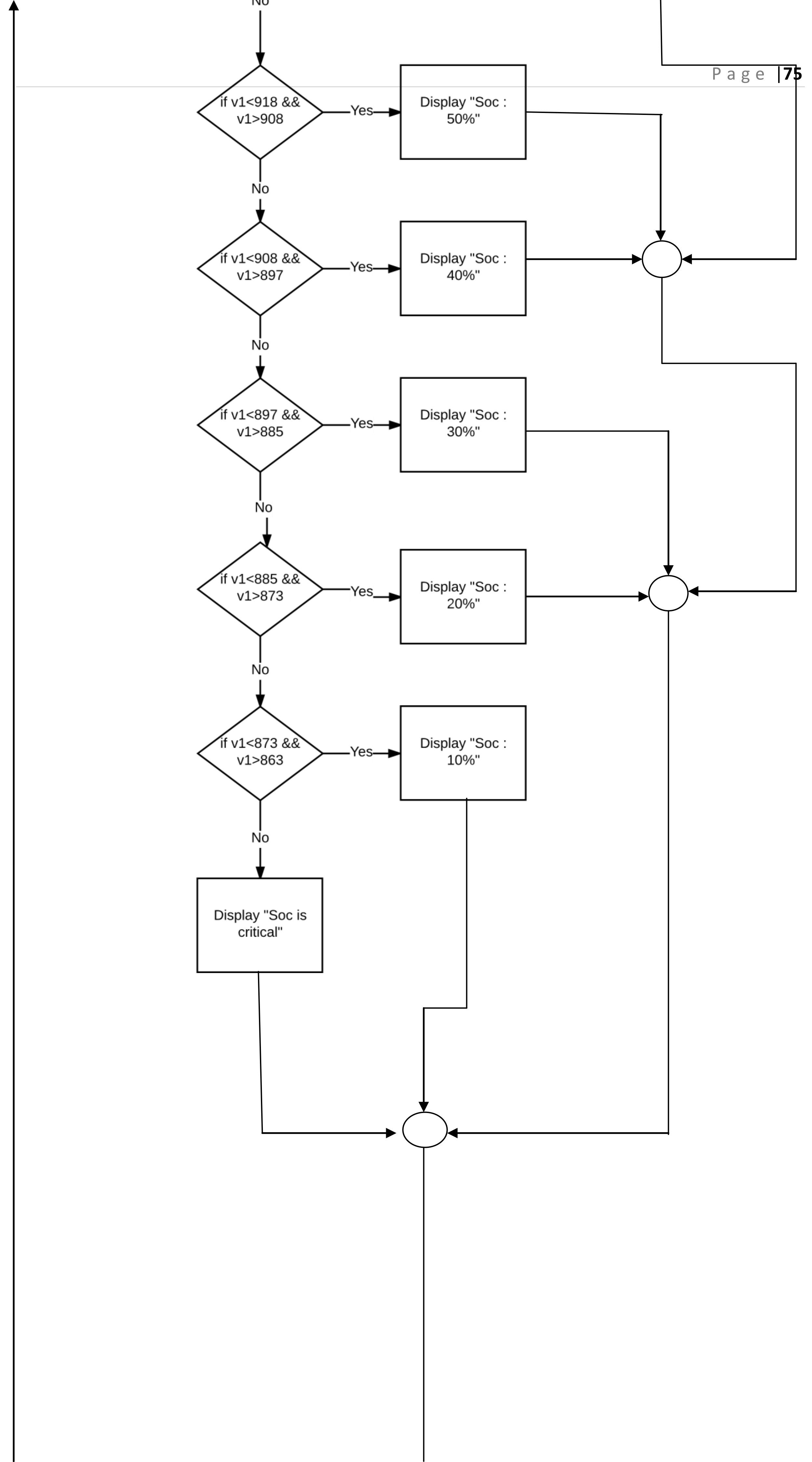
$$\text{Remaining distance} = \text{Average speed x Remaining time}$$

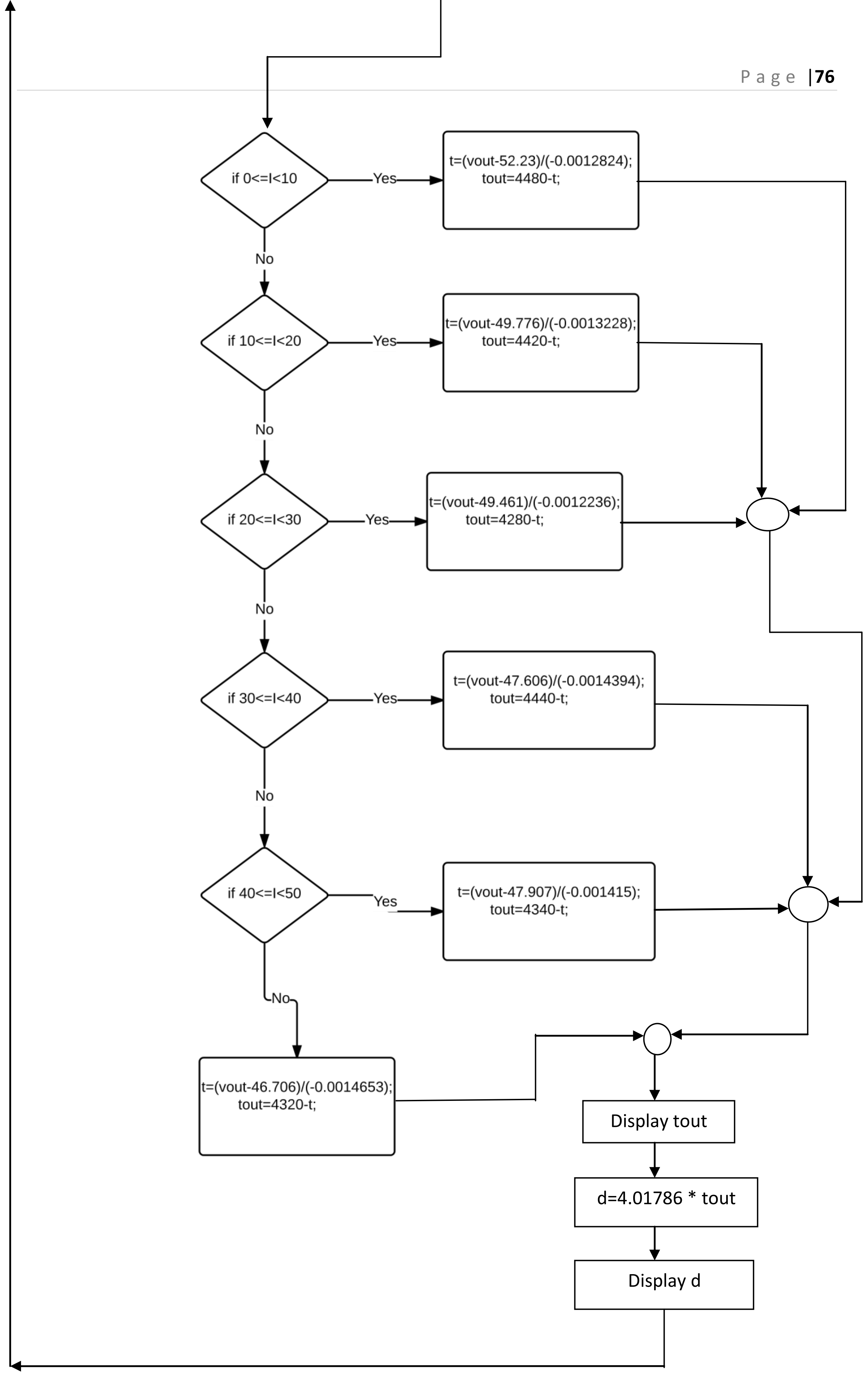
$$= 4.01786 \text{ x Remaining time}$$

We have written the following equations in our coding which is given in appendix C.

7.4 Whole algorithm of the project




No
if v1<918 && v1>908
Yes
Display "Soc : 50%"
No
if v1<908 && v1>897
Yes
Display "Soc : 40%"
No
if v1<897 && v1>885
Yes
Display "Soc : 30%"
No
if v1<885 && v1>873
Yes
Display "Soc : 20%"
No
if v1<873 && v1>863
Yes
Display "Soc : 10%"
No
Display "Soc is critical"



if 0<=I<10
Yes
t=(vout-52.23)/(-0.0012824); tout=4480-t;
No
if 10<=I<20
Yes
t=(vout-49.776)/(-0.0013228); tout=4420-t;
No
if 20<=I<30
Yes
t=(vout-49.461)/(-0.0012236); tout=4280-t;
No
if 30<=I<40
Yes
t=(vout-47.606)/(-0.0014394); tout=4440-t;
No
if 40<=I<50
Yes
t=(vout-47.907)/(-0.001415); tout=4340-t;
No
t=(vout-46.706)/(-0.0014653); tout=4320-t;
Display tout
d=4.01786 * tout
Display d

Figure 7.1: The whole algorithm of our project

Chapter 8

Final output on LCD display

8.1 Block diagram of the whole system with an electric vehicle

The block diagram mentioned in figure 8.1 where four 12V battery is connected in series and make it 48V battery. We installed our electrical device in the electrical vehicle to get our desired output on the LCD display. The positive terminal of the 12V battery is connected with the pin number 4 of the current sensor and pin number 5 of the current sensor is connected with the controller's pin number 4 of the electric vehicle. The motor was already connected with the controller properly. The negative terminal of the 48V battery is connected with the pin number 3 of the controller. We also connect a voltage regulator from the last 12V battery to get 5V which will give power to the current sensor, microcontroller and the LCD display. The 48V is feed into the microcontroller via a voltage divider circuit which converted the 48V into 5V. The output voltage of the current sensor is also feed into the microcontroller, and then the microcontroller is connected with the LCD display.

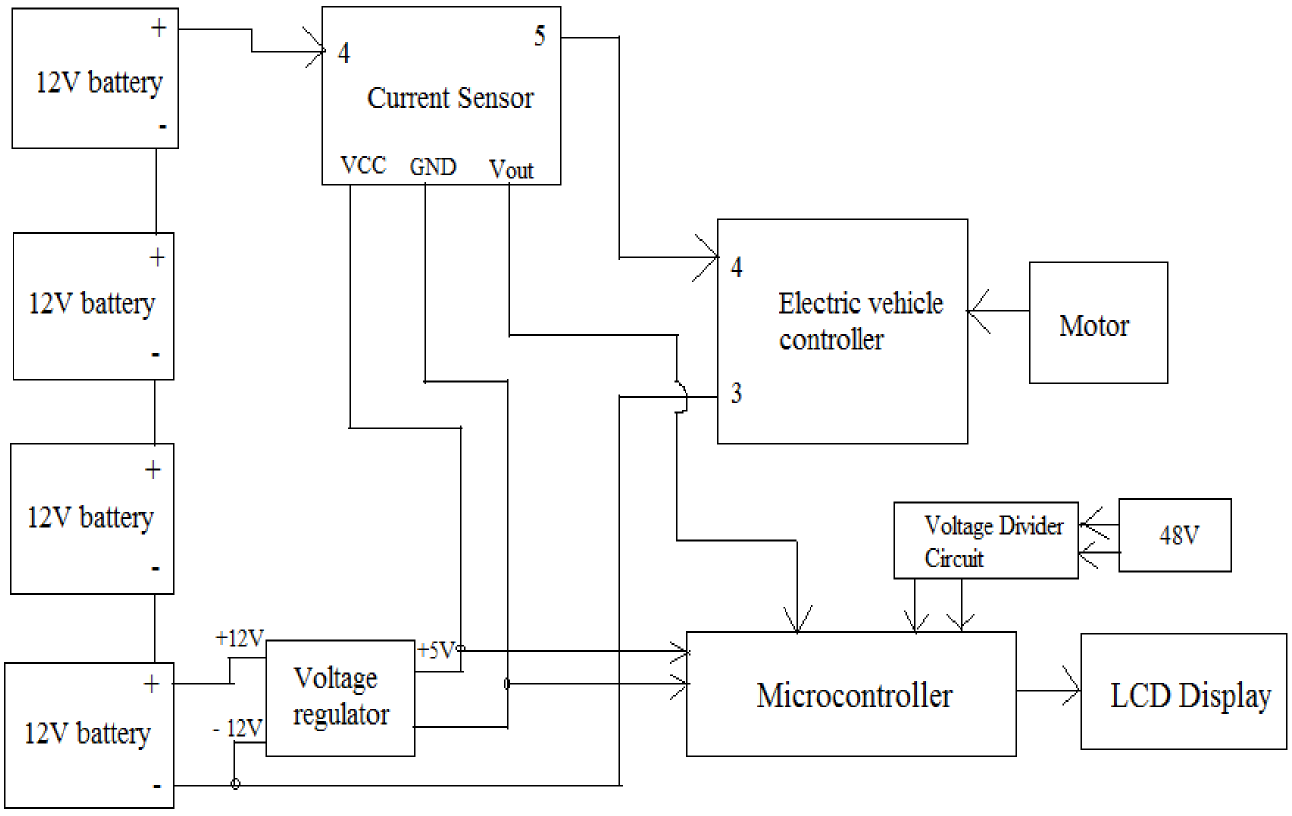


Figure 8.1: Block diagram of the whole system with an electric vehicle

8.2 Field test data of our electrical device

We installed our electrical device in an electric vehicle and have taken a field test using that electric vehicle. The field data also retrieved from the video and the data in given in the appendix D. The following figure 8.2, 8.3, 8.4 and 8.5 we have taken during the field test of our electrical device.

Figure 8.2: Electric vehicle used for our field test

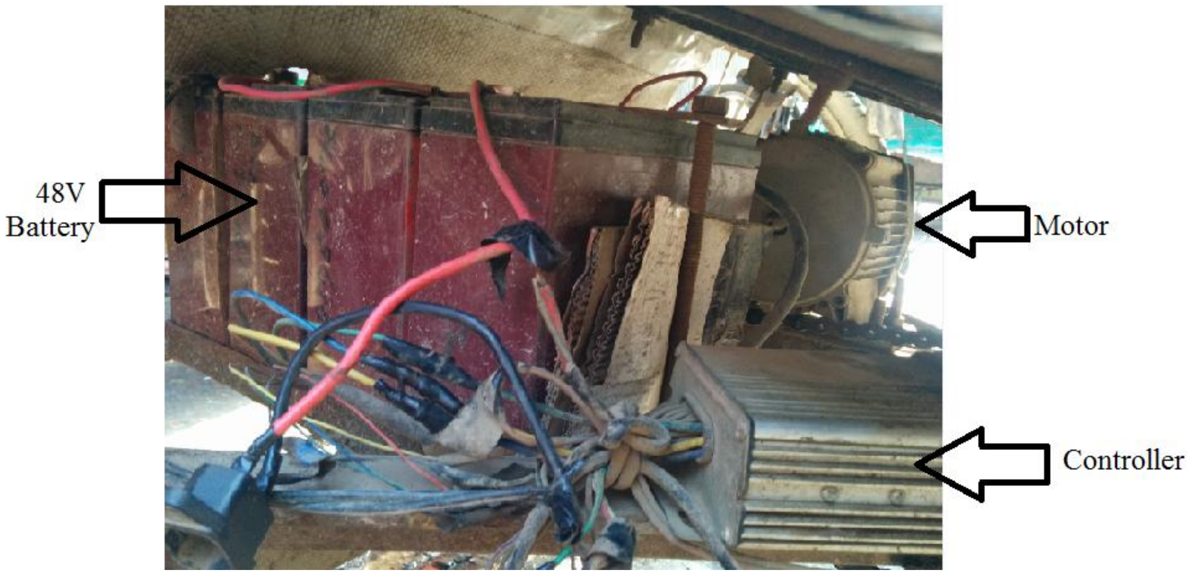


Figure 8.3: Electric vehicle equipments

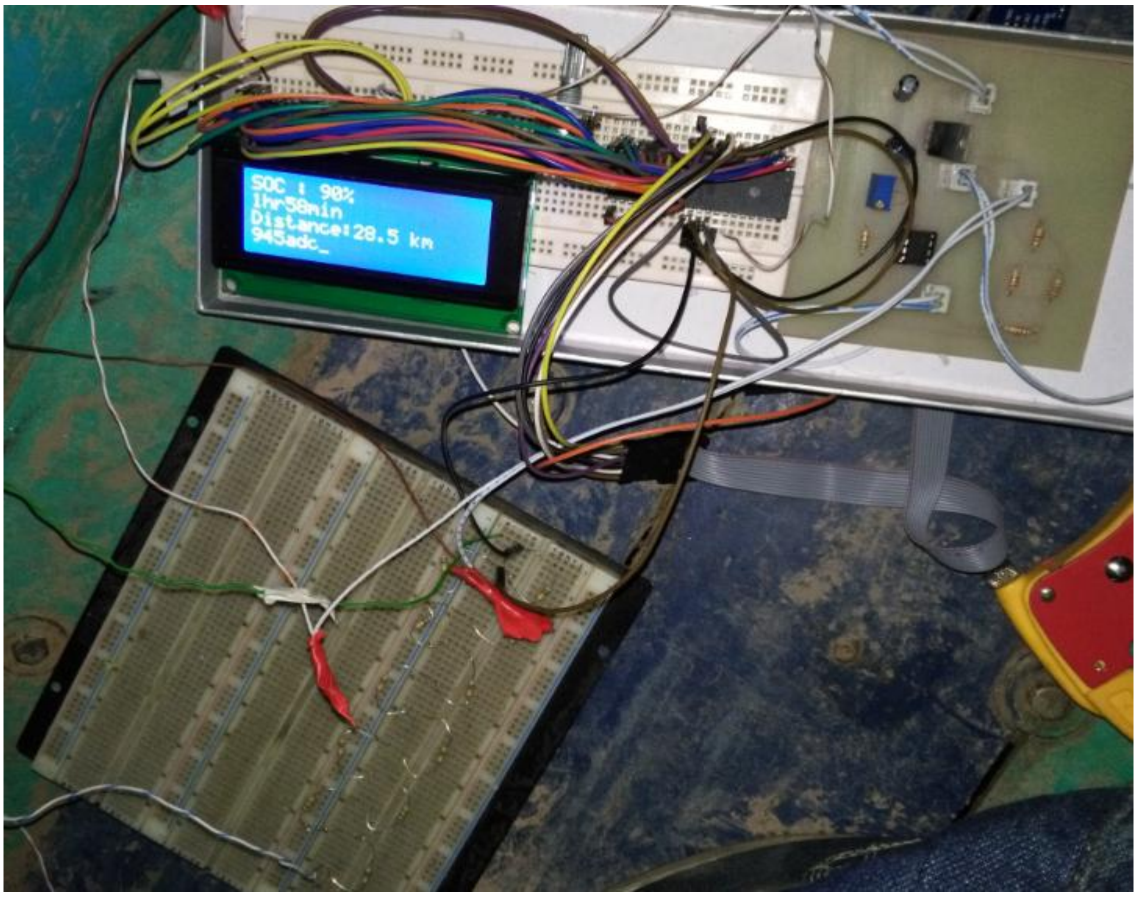


Figure 8.4: Output on the LCD display before the field test

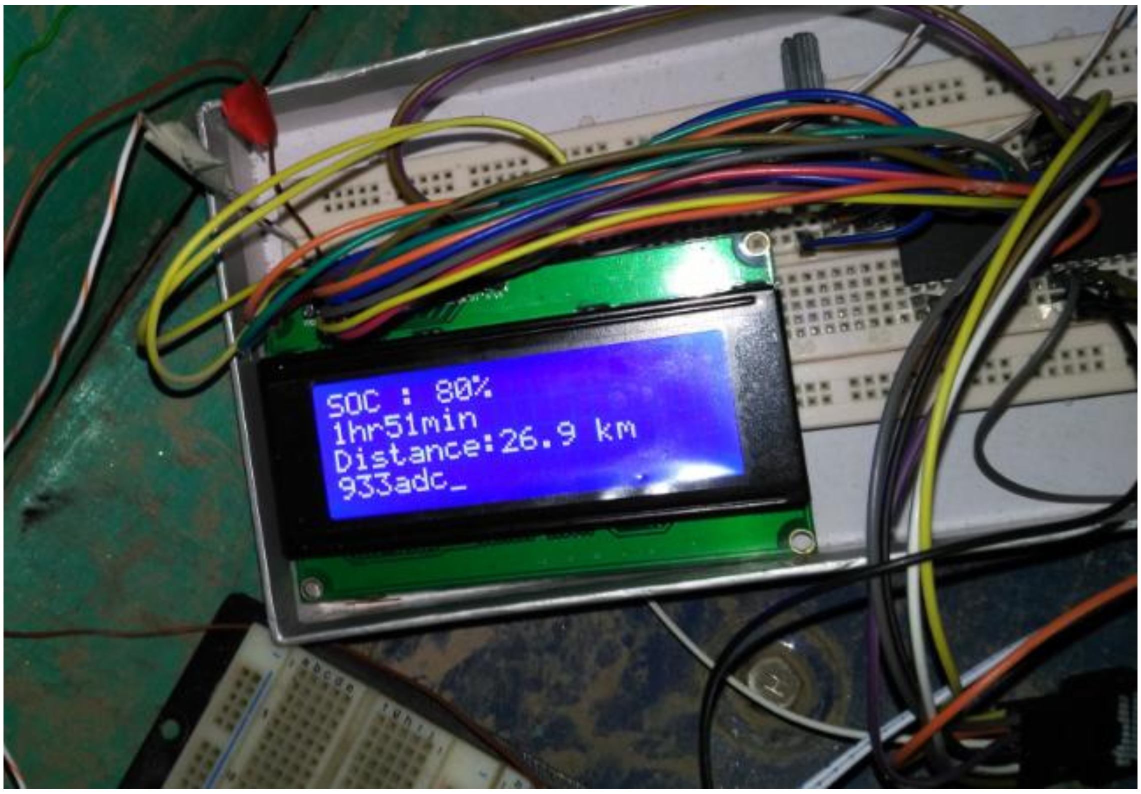


Figure 8.5: Output on the LCD display after the field test

Chapter 9

Conclusion

9.1 Summary

Electric vehicles are drawing an attractive option in the world because of its environmental friendly behavior, high level efficiency and regenerative multiple energy sources. Our project is to make an electrical device which will display SOC, remaining travelling time and the total distance covered by the vehicle for the current SOC to reach the destination. We also took the field test of the electrically assisted cargo hauler and data retrieved from the video of our field test. After plotting the data we got some equation to develop the algorithm for the cargo hauler. For measuring remaining time, we grouped our data for different range of motor current and after then we find out the distance covered with the current SOC. We will keep our work on to improve the electric vehicle for future and will install this electrical device in all electric vehicle to know the current battery condition; otherwise in the middle of the road, the battery can be fully discharged without the driver’s concern which can be a severe nuisance for both passengers and drivers. We have implemented our electrical device on a electric vehicle and the performance of the device is exactly the way it has been designed and developed and thus to a greater extent the microcontroller based device can be said as a successful one.

9.2 Limitations and challenges

An embedded system has been designed and developed to monitor the real time condition of all the batteries of electrically assisted vehicle while carrying out the field test using the electrically assisted cargo hauler, due to the road like traffic jam, bad condition of the road, random comfort of paddling of the driver. The motor current fluctuated rapidly in every seconds for which it was difficult to record these readings manually and frequently that is one of the challenges we faced during the data acquisition of our project to get the accurate result.

The microcontroller cannot take any input value of more than 5V where the full charged battery voltage of the cargo hauler is 50.92V. Hence we have to step down the battery voltage within 5V range to feed into the microcontroller. Thus, here we also get deprived to some extent from the exact result due to the step down of the microcontroller input voltage. In addition, the microcontroller we used has some internal circuit problems which create signal oscillation and noise. Because of this problems we get fluctuation while doing ADC (Analog to digital conversion) for obtaining a given fixed input voltage value.

The current sensor testing also took us almost to the edge of imperfection as we tested it in the laboratory before implementing on the vehicle. We tested our current sensor with an electrical stove as a load where maximum current is around 20A and cargo hauler can draw maximum 60A current. Since the algorithm for time remaining and distance covered is dependent on the current, so to get the desired result, we were able to do tests for the range 0-10A, 10-20A but not more than this range. Another problem we faced to measure the output voltage of the current sensor. For every 1A current, the change in output voltage of current sensor is 10mV and the offset voltage is also there which is around 2V. We cannot sense the change of output voltage by using the multimeter because the multimeter cannot show change in voltage in such small mV value. The solution of this problem has been found by implementing another circuit using the Data Acquisition Card (DAQ) with an electrical stove to sense the change of the output voltage of the current sensor as it can show 8 digit after the decimal value.

As we are displaying SOC, remaining discharging time of the battery and distance, all these information states the current condition of the electric vehicle. This project is about real time monitoring system of the battery which is a error prone process. For this reason, we cannot get the accurate result when we are displaying current status of the battery of the electric vehicle.

9.3 Plan of improvement for future

This project can be improved by displaying the present status of the electric vehicle in Bengali language for the sake of the simplicity so that the drivers in our country those who cannot understand English language. We can also display the exact speed of the electric vehicle in which the vehicle is moving. The problem we faced for real time system can be minimized by calibrating the system; this will improve the performance of our electrical device. We can set an alarm which will alert the driver that the vehicle is fully charged, half charged and almost zero charge. We can also set a Bluetooth module which will send message to drivers/passengers via mobile phone about the battery condition and how longer distance the passenger can travel.

Reference

[1] Z. Guirong, Z. Henghai, L. Houyu, "The driving control of pure electric vehicle", Proc. Environ. Sci. 10 (2011) pp.433-438; 2011, Accessed on: Jul. 1, 2016

[2] Ringkesh, "Advantages and disadvantages of electric cars - conserve energy future," in *Energy Articles*, Conserve-Energy-Future. [Online]. Available: http://www.conserve-energy-future.com/advantages-and-disadvantages-of-electric-cars.php, 2014.Accessed on: Jul. 1, 2016

[3] R. Zhang and E. Yao, "Electric vehicles' energy consumption estimation with real driving condition data," pp.177-187. [Online]. Available: http://opensample.info/electric-vehicles-energy-consumption-estimation-with-real-driving-condition-data, 2015.Accessed on: Jul. 3, 2016

[4] C. K. Wai, Y. Y. Rong, and S. Morris, "Simulation of a distance estimator for battery electric vehicle," *Alexandria Engineering Journal*, vol. 54, no. 3, pp.359-371. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1110016815000563,Sep. 2015.Accessed on: Jul. 3, 2016

[5]"Wheel tax," in *Dhaka North City Corporation* [Online]. Available: http://www.dncc.gov.bd/index.php/act-rules/wheel-tax.html, 2014 Accessed on: Jul. 6, 2016

[6]N. Zaman, "Rickshaw," in *Banglapedia*. [Online]. Available: http://en.banglapedia.org/index.php?title=Rickshaw, Mar. 9, 2015.Accessed on: Jul. 3, 2016

[7] R. Tarek, A. Anjum, M. Abrar-Ul-Hoque, F. A. Rahim, 'Solar Electric Ambulance Van to Assist Rural Emergencies of Bangladesh- A Complete Off Grid Solution'. Available: http://dspace.bracu.ac.bd/xmlui/handle/10361/4870, 2015. Accessed on: Jul. 14, 2016

[8] "Atmel AVR," in *Wikipedia,* Wikimedia Foundation. Available: https://en.wikipedia.org/wiki/Atmel_AVR, June 9, 2016.Accessed on: Jul. 6, 2016

[9] "Simple AVR Programmers". Internet: http://elm-chan.org/works/avrx/report_e.html, June 16, 2015.Accessed on: Jul.7, 2016

[10] 2016freedownloadmanager, "Extreme burner - AVR (free) download windows version," *Google* [Online]. Available: http://en.freedownloadmanager.org/Windows-PC/eXtreme-Burner-AVR-FREE.html, 2016.Accessed on: Jul. 6, 2016

[11] A. Tutorials. "AVR tutorials". Available: http://www.avr-tutorials.com/avr-studio-5/avr-studio-5, 2012.Accessed on: Jul. 6, 2016

[12] A. Tutorials, "Generating / creating the AVR HEX file in AVR studio 5". Available: http://www.avr-tutorials.com/avr-studio-5/generating-hex-file-avr-studio-5, 2010.Accessed on: Jul. 6, 2016

[13] Ex. Electronics and India, "Extreme burner - PIC. Get the software safe and easy," *Software Informer,* [Online]. Available: http://extreme-burner-pic.software.informer.com, 2015.Accessed on: Jul. 6, 2016

[14] Ex. Electronics and India, "Extreme burner - AVR. Get the software safe and easy," *Informer Technologies. Inc.* [Online]. Available: http://extreme-burner-avr.software.informer.com, 2016.Accessed on: Jul. 6, 2016

[15] A. Gupta. "Tutorial on USBASP AVR Programmer Documentation". Internet: www.extremeelectronics.co.in/downloads/usbavrprogrammer/docs/Tutorial.pdf, 2007-2008 Accessed on: Jul. 7, 2016

[16]"USBASP AVR Programmer," in *Future Electronics*. [Online]. Available: http://www.fut-electronics.com/wp-content/uploads/2015/11/USBASP_AVR_Programmer_tutorial.pdf. Accessed on: Jul. 7, 2016

[17]"AC-PG-USBASP USBASP AVR Programmer" in *ProtoStack*[Online]. Available: http://eecs.oregonstate.edu/education/docs/ece375/USBASP-UG.pdf, Jan. 18, 2012. Accessed on: Jul. 9, 2016

[18] "How to program an AVR chip using a USBASP (10-pin cable)," in *Learning About Electronics*. [Online]. Available: http://www.learningaboutelectronics.com/Articles/Program-AVR-chip-using-a-USBASP-with-10-pin-cable.php, 2015.Accessed on: Jul. 9, 2016

[19] Rakesh, "A note on character LCD displays" in *CircuitsToday* [Online]. Available: http://www.circuitstoday.com/a-note-on-character-lcd-displays, Feb. 29, 2012. Accessed on: Jul. 9, 2016

[20] "Atmel- 8255G 8-bit AVR Microcontroller ATmega32A datasheet complete," in *Atmel Corporation*. Available: http://www.atmel.com/images/atmel-8155-8-bit-microcontroller-avr-atmega32a_datasheet.pdf, 2015. Accessed on: Jul. 10, 2016

[21] G. Gridling and B. Weiss, "Introduction to Microcontrollers".Available: https://ti.tuwien.ac.at/ecs/teaching/courses/mclu/theory-material/Microcontroller.pdf, Feb.26, 2007. Accessed on: Jul. 10, 2016

[22] Kushagra, "ATmega32," in *EngineersGarage* [Online] Available: http://www.engineersgarage.com/electronic-componenets/atmega32-avr-microcontroller, 2015.Accessed on: Jul. 10, 2016

[23]"Rotary Potentiometer- 10k Ohm, Linear," in *sparkfun*, [Online]. Available: https://www.sparkfun.com/products/9939, 2015.Accessed on: Jul. 10, 2016

[24] "USB-4716, 200 KS/s, 16-bit, USB Multifunction Module User Manual," in *Advantech Co., Ltd*. Available: http://www.lima.com.tr/Advantech%20eAutomation/Manuals/USB-4716%20Manual. PDF, 2006.Accessed on: Aug. 10, 2016

[25] A. A. Chowdhury, M. K. Priyanka, and B. Mahmud, "Real time monitoring of solar battery charging station (SBCS)," *BRAC University*. Available: http://hdl.handle.net/10361/4872,2015.Accessed on: Aug. 10, 2016

[26] "Thermally Enhanced, Fully Integrated, Hall Effect-Based Linear Current Sensor IC with 100 μΩ Current Conductor," in *Allegromicro USA* [Online]. Available: file:///C:/Users/USER/Documents/Downloads/ACS758-Datasheet%20.pdf. Accessed on: Jul. 24, 2016

[27] DanYell, "Hall effect sensor and how magnets make it works," *Basic Electronics Tutorials*. Available: http://www.electronics-tutorials.ws/electromagnetism/hall-effect.html, 2013.Accessed on: Jul. 24, 2016

[28] R. Repas, "Sensor sense: Hall-effect current sensors," 2007. [Online]. Available: http://machinedesign.com/sensors/sensor-sense-hall-effect-current-sensors. Accessed on: Aug. 12, 2016

[29] W. Yen and W. Chen, "Estimate SOC capacity of power battery on the dynamic working voltage," *2013 Fourth International Conference on Digital Manufacturing & Automation*, pp. 559–562, Jun. 2013.Accessed on: Jul. 24, 2016

[30] F. R. Khan, A. Rahman, and A. T. Aurony, "Power conservation for electrically assisted rickshaw-vans with pv support, torque sensor pedal and the solar battery charging station: A complete off-grid solution (conducting field tests and data analysis for performance and feasibility study)," *BRAC University*, Available: http://hdl.handle.net/10361/4333, 2015.Accessed on: Jul. 24, 2016

[31] T. K. Hareendran, "AVR ADC analog to digital conversion - Tutorial #13,". [Online]. Available: http://www.electroschematics.com/10053/avr-adc/.Accessed on: Jul. 24, 2016

APPENDIX A

Field test data of Cargo Hauler (C5):

Time (s)	Motor Current (A)	Battery Voltage (V)
20	41.5	47.3
40	38.2	47.6
60	38.3	47.4
80	38.6	47.2
100	34.5	47.6
120	32.1	48.2
140	54.4	46.7
160	0.2	51.3
180	35.0	45.9
200	53.9	45.8
220	53.6	45.4
240	50.9	46.1
260	53.9	45.5
280	0.10	51.9
300	0.1	52.0
320	0.3	52.0
340	43.8	47.1
360	37.3	47.5
380	24.9	48.3
400	54.1	45.5
420	53.8	45.7
440	35.2	47.5
460	53.4	45.6
480	37.7	46.7
500	0.5	51.2
520	21.0	47.2
540	33.3	41.1

560	36.7	46.8
580	15.9	49.6
600	53.4	45.2
620	0.5	50.6
640	1.7	50.0
660	41.5	47.1
680	54.3	45.4
700	52.0	45.4
720	54.7	45.2
740	53.3	45.2
760	35.0	48.7
780	43.2	46.7
800	2.3	50.4
820	54.5	45.4
840	47.8	45.6
860	14.1	47.5
880	43.4	46.6
900	5.5	50.6
920	1.1	51.5
940	39.8	47.4
960	55.0	45.3
980	54.3	44.5
1000	54.8	45.2
1020	54.6	45.1
1040	53.4	45.5
1060	54.3	45.0
1080	54.2	45.0
1100	1.3	50.6
1120	53.8	45.2
1140	29.8	47.8
1160	8.6	50.1
1180	1.3	51.1

1200	1.3	50.8
1220	1.3	51.0
1240	1.3	50.7
1260	1.2	51.0
1280	41.4	46.5
1300	40.2	46.4
1320	40.2	46.7
1340	2.1	50.2
1360	39.3	46.3
1380	36.0	46.7
1400	3.2	50.4
1420	1.5	50.7
1440	26.3	48.1
1460	21.5	48.0
1480	55.3	44.8
1500	54.2	44.5
1520	51.6	45.3
1540	55.1	44.8
1560	1.8	50.6
1580	1.8	51.0
1600	1.9	51.4
1620	42.3	46.0
1640	6.8	49.5
1660	54.3	44.6
1680	51.0	44.4
1700	55.4	44.5
1720	55.3	44.1
1740	55.4	44.2
1760	36.5	46.1
1780	1.9	50.1
1800	47.0	45.2
1820	1.5	50.0

1840	1.7	49.9
1860	22.3	48.9
1880	3.5	50.2
1900	56.7	44.2
1920	55.0	44.5
1940	56.6	44.0
1960	2.2	49.9
1980	50.3	44.7
2000	56.6	44.2
2020	2.6	49.8
2040	2.2	49.6
2060	55.0	44.0
2080	29.2	46.4
2100	50.3	44.8
2120	55.5	43.6
2140	2.0	50.0
2160	2.0	50.2
2180	26.5	46.8
2200	20.8	47.7
2220	29.9	46.5
2240	2.0	49.6
2260	2.6	49.6
2280	54.6	43.7
2300	2.0	49.6
2320	55.4	43.4
2340	30.0	46.0
2360	23.5	47.3
2380	53.5	43.6
2400	45.4	43.9
2420	1.9	49.3
2440	0.1	49.8
2460	0.2	49.8

2480	41.7	46.5
2500	52.0	46.6
2520	54.7	43.7
2540	54.9	43.3
2560	53.8	43.3
2580	54.1	43.2
2600	53.5	43.6
2620	40.2	44.5
2640	3.7	48.9
2660	54.3	43.2
2680	53.9	43.1
2700	5.2	48.5
2720	54.1	43.0
2740	0.3	49.0
2760	2.8	48.7
2780	54.4	43.1
2800	0.3	48.8
2820	54.0	42.9
2840	43.3	44.0
2860	54.1	42.7
2880	53.5	42.4
2900	52.8	43.5
2920	39.6	43.8
2940	53.8	42.2
2960	0.1	48.6
2980	53.7	42.2
3000	0.1	48.4
3020	6.1	46.5
3040	54.5	42.7
3060	54.2	42.1
3080	24.8	47.6
3100	54.4	42.0

3120	47.0	42.9
3140	54.5	42.2
3160	16.6	46.7
3180	53.9	41.7
3200	53.8	41.9
3220	0.4	47.8
3240	53.0	41.8
3260	26.2	44.1
3280	54.2	41.6
3300	53.0	41.7
3320	0.1	47.5
3340	53.2	41.6
3360	46.8	42.1
3380	53.5	41.2
3400	53.3	41.2
3420	53.4	40.7
3440	19.1	45.5
3460	50.4	41.5
3480	53.6	40.8
3500	21.3	45.9
3520	0.2	47.5
3540	55.4	41.9
3560	5.5	47.1
3580	33.6	43.4
3600	55.0	40.7
3620	30.8	43.4
3640	50.5	41.0
3660	54.1	40.4
3680	55.0	40.6
3700	54.6	40.3
3720	54.5	39.9
3740	26.0	43.5

3760	0.4	47.3
3780	0.4	47.4
3800	0.3	47.5
3820	58.1	42.9
3840	5.0	47.5
3860	20.1	47.0
3880	2.5	48.2
3900	57.7	42.1
3920	44.4	45.4
3940	2.5	47.5
3960	27.8	45.9
3980	2.4	47.6
4000	2.3	47.9
4020	2.4	47.9
4040	30.0	44.0
4060	56.5	40.6
4080	35.8	43.0
4100	50.7	41.1
4120	36.8	42.6
4140	28.0	43.6
4160	27.7	43.6
4180	27.5	43.5
4200	27.4	43.3
4220	2.4	46.6
4240	1.9	47.2
4260	22.5	44.4
4280	29.4	43.3
4300	56.2	39.6
4320	50.3	38.0
4340	42.8	38.9
4360	37.9	39.4
4380	1.8	44.5

4400	31.7	40.0
4420	13.6	43.0
4440	36.4	39.3
4460	45.6	38.0
4480	1.4	44.8

APPENDIX B

Codes used in MATLAB software:

Codes are written to produce graphical representations of Battery Voltage (y-axis) versus Time Duration elapsed (x –axis).

For 0-10A current:

```
clc;
clearall;
closeall;
x=[160,280,300,320,500,620,640,800,900,920,1100,1160,1180,1200,1220,1240,1260,1340,1
400,1420,1560,1580,1600,1640,1780,1820,1840,1880,1960,2020,2040,2140,2160,2240,2260
,2300,2420,2440,2460,2640,2700,2740,2760,2800,2960,3000,3020,3220,3320,3520,3560,37
60,3780,3800,3840,3880,3940,3980,4000,4020,4220,4240,4380,4480];
y=[51.3,51.9,52,52,51.2,50.6,50,50.4,50.6,51.5,50.6,50.1,51.1,50.8,51,50.7,51,50.2,50.4,50.7
,50.6,51,51.4,49.5,50.1,50,49.9,50.2,49.9,49.8,49.6,50,50.2,49.6,49.6,49.6,49.3,49.8,49.8,48.
9,48.5,49,48.7,48.8,48.6,48.4,46.5,47.8,47.5,47.5,47.1,47.3,47.4,47.5,47.5,48.2,47.5,47.6,47.
9,47.9,46.6,47.2,44.5,44.8];
plot(x,y),xlabel('Time(s)'),ylabel('Battery Voltage(V)'),title('Battery voltage VS Time for 0-
10A motor current')
gridon
```

For 10-20A current:

```
clc;
clearall;
closeall;
x=[580,860,3160,3440,4420];
y=[49.6,47.5,46.8,45.5,43.0];
```

```
plot(x,y),xlabel('Time(s)'),ylabel('Battery Voltage(V)'),title('Battery voltage VS Time for 10-20A motor current')
gridon
```

For 20-30A current:

```
clc;
clearall;
closeall;
x=[380,520,1140,1440,1460,1860,2080,2180,2200,2220,2360,3080,3260,3500,3760,3860,3960,4140,4160,4180,4200,4260,4280];
y=[48.3,47.2,47.8,48.1,48.0,48.9,46.4,46.8,47.7,46.5,47.3,47.6,44.1,45.9,43.5,47.0,45.9,43.6,43.6,43.5,43.3,44.4,43.3];
plot(x,y),xlabel('Time(s)'),ylabel('Battery Voltage(V)'),title('Battery voltage VS Time for 20-30A motor current')
gridon
```

For 30-40A current:

```
clc;
clearall;
closeall;
x=[40,60,80,100,120,180,360,440,480,540,560,760,940,1360,1380,1760,2920,3580,3620,4080,4120,4360,4400,4440];
y=[47.6,47.4,47.2,47.6,48.2,45.9,47.5,47.5,46.7,41.1,46.8,48.7,47.4,46.3,46.7,46.1,43.8,43.4,43.8,43.0,42.6,39.4,40.0,39.3];
plot(x,y),xlabel('Time(s)'),ylabel('Battery Voltage(V)'),title('Battery voltage VS Time for 30-40A motor current')
gridon
```

For 40-50A current:

```
clc;
clearall;
closeall;
x=[20,340,660,780,840,880,1280,1300,1320,1620,1800,2400,2480,2620,2840,3120,3360,3920,4340];
```

```
y=[47.3,47.1,47.1,46.7,45.6,46.6,46.5,46.4,46.7,46.0,45.2,43.9,46.5,44.5,44.0,42.9,42.1,45.4,38.9];
plot(x,y),xlabel('Time(s)'),ylabel('Battery Voltage(V)'),title('Battery voltage VS Time for 40-50A motor current')
gridon
```

For 50-60A current:

```
clc;
clearall;
closeall;
x=[140,200,220,240,260,400,420,460,600,680,700,720,740,820,960,980,1000,1020,1040,1060,1080,1120,1480,1500,1520,1540,1660,1680,1700,1720,1740,1900,1920,1940,1980,2000,2060,2100,2120,2280,2320,2380,2500,2520,2540,2560,2580,2600,2660,2680,2720,2780,2820,2860,2880,2900,2940,2980,3040,3060,3100,3140,3180,3200,3240,3280,3300,3340,3380,3400,3420,3460,3480,3540,3600,3640,3660,3680,3700,3720,3820,3900,4060,4100,4300,4320];
y=[46.7,45.8,45.4,46.1,45.5,45.5,45.7,45.6,45.2,45.4,45.4,45.2,45.2,45.4,45.3,44.5,45.2,45.1,45.5,45.0,45.0,45.2,44.8,44.5,45.3,44.8,44.6,44.4,44.5,44.1,44.2,44.2,44.5,44.0,44.7,44.2,44.0,44.8,43.6,43.7,43.4,43.6,46.6,43.7,43.3,43.3,43.2,43.6,43.2,43.1,43.0,43.1,42.9,42.7,42.4,43.5,42.2,42.7,42.7,42.1,42.0,42.2,41.7,41.9,41.8,41.6,41.7,41.6,41.2,41.2,40.6,41.5,40.8,41.9,40.7,41.0,40.4,40.6,40.3,39.9,42.9,42.1,40.6,41.1,39.6,38.0];
plot(x,y),xlabel('Time(s)'),ylabel('Battery Voltage(V)'),title('Battery voltage VS Time for 50-60A motor current')
gridon
```

Code used in MATLAB software for current sensor:

```
clc;
clearall;
closeall;
x=[2,4,6,8,10,12,14,16,18,19.2];
y=[2.0236210,2.0460510,2.0573430,2.0687870,2.0906770,2.1131900,2.1426390,2.1598820,2.1748350,2.1815490];
plot(x,y),xlabel('Current(A)'), ylabel('Output voltage of Current Sensor(V)'), title('Output voltage of Current Sensor VS current')
```

APPENDIX C

This is the coding part what we have written to display SOC, remaining travelling time and distance of the electric vehicle.

```
#include<avr/io.h>
#include<util/delay.h>
charfirstColumnPositionsForMrLCD[4]={0,64,20,84};
#defineMrLCDsCribPORTD
#defineDataDir_MrLCDsCribDDRD
#defineMrLCDsControlPORTB
#defineDataDir_MrLCDsControlDDRB
#defineLightSwitch2
#defineReadWrite1
#defineBiPolarMood0
voidCheck_IF_MrLCD_isBusy(void);
voidPeek_A_Boo(void);
voidSend_A_Command(unsignedcharcommand);
voidSend_A_Character(unsignedcharcharacter);
voidSend_A_String(char*StringOfCharacters);
voidGotoMrLCDsLocation(uint8_tx,uint8_ty);
voidInitADC()
{
ADMUX=(1<<REFS0);// For Aref=AVcc;
ADCSRA=(1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0);//Rrescalar div factor =128
}
uint16_tReadADC(uint8_tch)
{
//Select ADC Channel ch must be 0-7
ch=ch&0b00000111;
ADMUX|=ch;
//Start Single conversion
ADCSRA|=(1<<ADSC);

//Wait for conversion to complete
while(!(ADCSRA&(1<<ADIF)));

//Clear ADIF by writing one to it
//Note you may be wondering why we have write one to clear it
//This is standard way of clearing bits in io as said in datasheets.
//The code writes '1' but it result in setting bit to '0' !!!
```

```
ADCSRA|=(1<<ADIF);
return(ADC);
}
voidWait()
{
uint8_ti;
for(i=0;i<20;i++)
_delay_loop_2(0);
}
intmain(void)
{
        floatsum1;
        floatsum2;
        floatv1;
        floatv2;
        floatvout;
        floatt;
        floattout;
        intj;
        intk;
        floatd;
        floatd1;
        intz;
        inta;
        floatm;
        intp;
        floatw;
        floaty;
        floatx;
        floati;
        floatt1;
        intt2;
        floatt3;
        floatt4;
        intt5;
        intc;
        floatq;
        uint16_tv;
DataDir_MrLCDsControl|=1<<LightSwitch|1<<ReadWrite|1<<BiPolarMood;
_delay_ms(15);
//Initialize ADC
```

```
InitADC();
charpositionString[4];
Send_A_Command(0x01);//Clear Screen 0x01 = 00000001
_delay_ms(2);
Send_A_Command(0x38);
_delay_us(50);
Send_A_Command(0b00001110);
_delay_us(50);
while(1)
{
Send_A_Command(0x01);//Clear Screen 0x01 = 00000001
_delay_ms(2);
       sum1=0;
       for(j=0;j<30;j++)
       {
       sum1=sum1+ReadADC(2);
_delay_ms(2);
       }
       sum1=(sum1/30);
       vout=(4.476*11.568*sum1)/1000;
       sum2=0;
       for(k=0;k<30;k++)
       {
       sum2=sum2+ReadADC(1);
_delay_ms(2);
       }
       sum2=(sum2/30);
       y=sum2*(2.59/548);
       i=(y-2.0016)/0.0095486;

if(vout>50.92)
{
Send_A_String("SOC : 100%");
}
else
{
if((vout<50.92)&&(vout>50.48))
{
Send_A_String("SOC : 90%");
}
else
{
```

```
if((vout<50.48)&&(vout>50.00))
{
Send_A_String("SOC : 80%");
}
else
{
if((vout<50.00)&&(vout>49.48))
{
Send_A_String("SOC : 70%");
}
else
{
if((vout<49.48)&&(vout>48.96))
{
Send_A_String("SOC : 60%");
}
else
{
if((vout<48.96)&&(vout>48.40))
{
Send_A_String("SOC : 50%");
}
else
{
if((vout<48.40)&&(vout>47.84))
{
Send_A_String("SOC : 40%");
}
else
{
if((vout<47.84)&&(vout>47.24))
{
Send_A_String("SOC : 30%");
}
else
{
if((vout<47.24)&&(vout>46.64))

{
Send_A_String("SOC : 20%");
}
else
{
if((vout<46.64)&&(vout>46.04))
{
Send_A_String("SOC : 10%");
}
else
{
Send_A_String("SOC : Critical");
}
}
}
}
}
}
}
}
}
}
Send_A_Command(0xC0);
if((0<=i)&&(i<10))
{
t=(vout-52.23)/(-0.0012824);
tout=4480-t;
}
else
{
if((10>=i)&&(i<20))
{
t=(vout-49.776)/(-0.0013228);
tout=4420-t;
}
else
{
if((20>=i)&&(i<30))
{
t=(vout-49.461)/(-0.0012236);
tout=4280-t;
}
else
{
if((30>=i)&&(i<40))
{
```

```
t=(vout-47.606)/(-0.0014394);
tout=4440-t;
}
else
{
if((40>=i)&&(i<50))
{
t=(vout-47.907)/(-0.001415);
tout=4340-t;
}
else
{
t=(vout-46.706)/(-0.0014653);
tout=4320-t;
}
}
}
}
}
q=tout;
t1=q/3600;
t2=t1;
c=t2;
itoa(c,positionString,10);
Send_A_String(positionString);
Send_A_String("hr");
t3=t1-t2;
t4=60*t3;
t5=t4;
p=t5;
itoa(p,positionString,10);
Send_A_String(positionString);
Send_A_String("min");
Send_A_Command(0x94);
Send_A_String("Distance:");
d=(4.01786*tout);
d1=d/1000;
a=d1;
itoa(a,positionString,10);
Send_A_String(positionString);
Send_A_String(".");
w=(d1-a);
```

```
m=10*w;
z=m;
itoa(z,positionString,10);
Send_A_String(positionString);
Send_A_String(" ");
Send_A_String("km");
Send_A_Command(0xD4);
intu=i;
itoa(u,positionString,10);
Send_A_String(positionString);
Send_A_String(" A");
Wait();
                     }
}
voidCheck_IF_MrLCD_isBusy()
{
DataDir_MrLCDsCrib=0;
MrLCDsControl|=1<<ReadWrite;
MrLCDsControl&=~1<<BiPolarMood;
while(MrLCDsCrib>=0x80)
{
Peek_A_Boo();
}
DataDir_MrLCDsCrib=0xFF;//0xFF means 0b11111111
}
voidPeek_A_Boo()
{
MrLCDsControl|=1<<LightSwitch;
asmvolatile("nop");
asmvolatile("nop");
MrLCDsControl&=~1<<LightSwitch;
}
voidSend_A_Command(unsignedcharcommand)
{
Check_IF_MrLCD_isBusy();
MrLCDsCrib=command;
MrLCDsControl&=~((1<<ReadWrite)|(1<<BiPolarMood));
Peek_A_Boo();
MrLCDsCrib=0;
}
voidSend_A_Character(unsignedcharcharacter)
{
```

```
Check_IF_MrLCD_isBusy();
MrLCDsCrib=character;
MrLCDsControl&=~(1<<ReadWrite);
MrLCDsControl|=1<<BiPolarMood;
Peek_A_Boo();
MrLCDsCrib=0;
}
voidSend_A_String(char*StringOfCharacters)
{
while(*StringOfCharacters>0)
{
Send_A_Character(*StringOfCharacters++);
}
voidGotoMrLCDsLocation(uint8_tx,uint8_ty)
{
Send_A_Command(0x80+firstColumnPositionsForMrLCD[y-1]+(x-1));
}
}
```

APPENDIX D

Field test data of our device:

Time (s)	SOC, %	Remaining time	Distance, km
0	80	1hr 52min	27.2
5	70	1hr 45min	25.4
10	90	1hr 55min	27.8
15	90	1hr 55min	27.8
20	80	1hr 54min	27.4
25	80	1hr 53min	27.4
30	80	1hr 50min	26.5
35	70	1hr 49min	26.3
40	80	1hr 52min	27.1
45	80	1hr 50min	26.7
50	80	1hr 52min	27.0
55	80	1hr 52min	27.0
60	80	1hr 52min	27.0
65	80	1hr 53min	27.4
70	80	1hr 53min	27.4
75	50	1hr 35min	23.0
80	50	1hr 34min	22.8
85	50	1hr 34min	22.8
90	80	1hr 49min	26.4
95	70	1hr 47min	25.9
100	40	1hr 26min	20.9
105	70	1hr 45min	25.5
110	80	1hr 50min	26.6
115	40	1hr 28min	21.3
120	80	1hr 51min	26.7
125	80	1hr 51min	26.7
130	80	1hr 52min	27.0
135	50	1hr 31min	22.0

140	40	1hr 25min	20.6
145	70	1hr 47min	25.9
150	80	1hr 50min	26.6
155	40	1hr 25min	20.6
160	40	1hr 25min	20.6
165	80	1hr 50min	26.6
170	30	1hr 23min	20.1
175	50	1hr 35min	22.9
180	30	1hr 24min	20.3
185	40	1hr 25min	20.5
190	70	1hr 47min	26.0
195	70	1hr 47min	26.0
200	40	1hr 25min	20.5
205	30	1hr 24min	20.4
210	70	1hr 48min	26.2
215	30	1hr 23min	20.0
220	70	1hr 49min	26.3
225	70	1hr 49min	26.3
230	80	1hr 50min	26.6
235	40	1hr 27min	21.0
240	70	1hr 47min	25.9
245	50	1hr 32min	22.3
250	40	1hr 25min	20.6
255	80	1hr 50min	26.6
260	80	1hr 51min	26.7
265	80	1hr 51min	26.7
270	70	1hr 47min	25.8
275	70	1hr 47min	25.8
280	30	1hr 24min	20.4
285	70	1hr 48min	26.0
290	80	1hr 49min	26.4
295	80	1hr 49min	26.4

300	30	1hr 23min	20.2
305	30	1hr 23min	20.2
310	70	1hr 45min	25.3
315	80	1hr 49min	26.4
320	80	1hr 49min	26.4
325	80	1hr 51min	26.8
330	50	1hr 31min	22.1
335	50	1hr 31min	22.1
340	40	1hr 27min	21.0
345	70	1hr 47min	25.9
350	60	1hr 41min	24.5
355	70	1hr 46min	25.6
360	70	1hr 46min	25.5
365	70	1hr 46min	25.5
370	30	1hr 24min	20.4
375	30	1hr 23min	20.0
380	70	1hr 48min	26.1
385	80	1hr 50min	26.6
390	80	1hr 51min	26.7
395	40	1hr 25min	20.5
400	70	1hr 49min	26.3
405	70	1hr 49min	26.3
410	80	1hr 50min	26.6
415	80	1hr 51min	26.9
420	80	1hr 51min	26.9
425	80	1hr 51min	26.7
430	70	1hr 45min	25.3
435	70	1hr 45min	25.3
440	40	1hr 25min	20.5
445	80	1hr 49min	26.5
450	70	1hr 46min	25.6
455	30	1hr 23min	20.0

460	30	1hr 22min	19.9
465	60	1hr 38min	23.7
470	30	1hr 23min	20.1
475	70	1hr 48min	26.2
480	70	1hr 48min	26.2
483	80	1hr 50min	26.6