BLOOD DONOR AND USER COMMUNICATION SYSTEM USING IOT AND GSM

A PROJECT REPORT

Submitted by

DHARANI KANNA KV (714217106009)

KALIDHAS S (714217106015)

SREERAG P (714217106034)

THARANI DHARAN PR (714217106036)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

TAMILNADU COLLEGE OF ENGINEERING,

KARUMATHAMPATTI, COIMBATORE - (641659)

ANNA UNIVERSITY: CHENNAI 600 025

MARCH 2021

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"BLOOD TYPE DETECTION AND DONAR COMMUNICATION SYSTEM USING IOT AND GSM"** is the bonafide work of "**DHARANI KANNA KV, KALIDHAS S, SREERAG P, THARANI DHARAN PR"** who carried out the project work under my supervision.

SIGNATURE

Dr.S.LATHA SHANMUGA VADIVU

HEAD OF THE DEPARTMENT

Electronics and communication engineering
Tamilnadu college of engineering,
Karumathampatti,
Coimbatore – 641659.

SIGNATURE

S. NAGAMMAI M.E

SUPERVISOR

Electronics and communication engineering
Tamilnadu college of engineering,
Karumathampatti,
Coimbatore – 641659.

Submitted for the anna university examination held on ……...

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENTS

Abstract

Today, in the present scenario, many people were facing a lot of problems in getting blood for patients at right time. The needy does not know where to get the blood or how to get access to a required quantity of blood quickly in an emergency. Blood donation is a big issue and consumes a lot of time to the donor. The blood bank systems have no proper tool for managing the blood collected from the camps. The needy in an emergency could not find the compatible blood or platelets with the patient. There are many existing mobile apps for blood donation. Though smartphones are available it is not possible to find a right donor who has compatible blood group with the patient in right time. We propose an integrated solution which will connect blood banks, donors, and the needy. This solution provides efficiency and convenience for the needy to search for required blood group or platelets in your neighborhood easy and simple. Get instant help form blood banks without signing up required. The solution works for the encouragement of blood donation and ensures the availability of blood or platelets accessible to the needy. It also assists quality management programs for blood platelets transfusion services. Finally, this solution is used to notify/communicate with the nearest blood bank organizations timely to ensure the availability of safe and quality blood accessible to the needy with minimum effort.

INTRODUCTION

Growing population has increased the need for the blood supply for various diseases. In every two seconds, some person required blood transfusion and currently India facing problem of the blood shortage. Automate the communication to make emergency alert for the donor. Low cost and Effective, efficient method for searching healthy and nearly blood donor.
Direct communication between the blood donor and blood requiring person.
t – IOT based Blood Bank is an associate work that brings voluntary blood donors and those in need of blood on to a common platform. The proposed system facilitates communication between blood donors and blood donation centers so that the appropriate donor can be reached just on time.

To address the problem an effective system is designed using the Internet of things. The system provides a methodology to fulfil the requirement of blood to the patients/victims without rushing to the blood bank to know the availability of the blood. An emergency buttons are connected to the Arduino board which continuously monitors the status of the available bloodstock. The output data provided by the Arduino is displayed on the webpage using the wi-fi module so anyone accesses the firebase and obtained the information of available bloodstock in real time. It will reduce the manpower required at the blood bank to update the online data also reduces the efforts of blood seeker of searching bloodstock at each blood bank.
When bloodstock reaches to zero system helps to send a request message to the donor and nearest blood bank. By using IoT the real-time available bloodstock is displaying on the website it minimizes the efforts of blood seeker.

CHAPTER 1

1.1 Model overview:

1.1.1 Existing model:

Such as just touch the button donor will be ask to enter an individual's details like name, phone number, age, weight, date of birth blood group, address etc.

At the emergency time of blood needed we can check for blood donor nearby by using GPS. Once the app user enters the blood group which he/she needed it will automatically show the donor nearby and send an alert message to the donor.

In case if the first donor is not available it will automatically search the next donor which is present in queue.

If the donor accepts the request, then a one-time password (OTP) will be sent to the donor to verify. Blood donation app provider list of donors in your city/area.

Once the donor donates the blood it will automatically remove the donor detail for next three months.

Some of existing models are

I. Android blood bank,

II. Blood Bank Venture Framework web application

Dis-advantages:

- Pre installation of this app is required for perform the task e.g., Request blood, Accept blood request.
- This app requires internet for Receiving and sending the request for blood.
- User need smart phone for use this application.
- Search in queue user method led to take long time for reach suitable donor.

1.1.2 Android blood bank:

Blood is a saver of all existing lives in case of emergency needs. The task of blood bank is to receive blood from various donors, to monitor the blood groups database and to send the required blood during the need to the hospital in case of emergencies. The problem is not insufficient number of donors, but finding a willing donor at the right time. We want to build a network of people who can help each other during an emergency. This application timely updates the information regarding the donors where the administrator accesses the whole information about blood bank management system. Donor will be prompted to enter an individual's details, like name, phone number, and blood group. In the urgent time of a blood requirement, you can quickly check for blood banks or hospitals matching a particular or related blood group and reach out to them through the App. Blood bank App provides list of blood banks in your area. A large number of blood donors are attracted using an Android application. Since almost everyone carries a mobile phone with him, it ensures instant location tracking and communication. Only a registered person, with willingness to donate blood, will be able to access the service. In this application we are using the GPS technology that will be used to trace the way to the blood bank. The user will get the route to reach the desired location and he won't have to ask manually, therefore time can be saved.

1.1.3 Blood Bank Venture Framework web application:

Blood Bank Venture Framework web application keeps up an online library of blood donors in a city/state or region. There are times when Doctors and Blood bank venture need to confront the trouble in finding the blood among Donors at the appropriate time. Blood donation venture framework has endeavored to give the required response by taking upon itself the process of gathering Blood donors across the country for the cure of those who are in need. At any moment the patients who are in need can apply for a donation through the web application and connect with the right donor. Based on humanity, anybody can enlist as a blood donor. The donors are accepted only after their blood sample tests and also based on their medical history. The

general population who are non-benefactors can likewise assume an essential part of the proposed framework. Simply by registering their blood type, they can help in identifying the proportion donor of blood types needed in an area. Blood Bank Venture Framework (BBVF) is a web application program that is intended to store, process, retrieve and analyze information concerned with the administrative and inventory management within a blood bank. This project aims at maintaining all the information pertaining to blood donors, different blood groups available in each blood bank and help them manage in a better way. The aim is to provide transparency in this field, make the process of obtaining blood from a blood bank hassle free and corruption free and make the system of blood bank management effective.,

1.1.4 Android Based Health Application in Cloud Computing for Blood Bank:

In many emergency situations, such as accidents, there is an immediate need for specific blood type. According to survey as increasing requirements for blood, only about 5% of the Indian population donates blood. In our proposed system, user will just touch the button on mobile phone. Then he will automatically connect to the cloud and his location will be tracked by GPS. All information about the donors and blood bank is stored on the cloud. As per blood requirement, user can quickly check for contacts matching a particular or related blood group and sends notification to blood bank within the radius of 5-10 km. If requested blood group is available in the blood bank then it will send positive reply message to the users. If requested stock is not available in the blood bank then blood bank send notification to all donors. If anyone is able to donate then he will reply to blood bank. Only a registered person who wish to donate blood, will be able to access the service. Cloud-based services can prove important in emergency blood delivery since they are able to access data.

1.1.5 Optimization of Blood Donor Information and Management System:

Emergency situations, such as accidents, create an immediate, critical need for specific blood type. In addition to emergency requirements, advances in medicine have increased the need for blood in many ongoing treatments and elective surgeries. Despite increasing requirements for blood, only about 5% of the Indian population donates blood. In this paper we propose a new and efficient way to overcome such scenarios with our project. We have to create a new idea, just touch the button. Donor will be prompted to enter an individual's details, like name, phone number, and blood type. After that your contact details will appear in alphabetical order on the screen; the urgent time of a blood requirement, you can quickly check for contacts matching a particular or related blood group and reach out to them via Phone Call/SMS through the Blood donor App.

1.1.6 BCloud App: Blood Donor Application for Android Mobile:

Blood service operations are a key component of the healthcare system all over the world and yet the modeling and the analysis of such systems from a complete supply chain network optimization perspective have been lacking due to their associated unique challenges. To provide web based communication there are numbers of online web based blood bank management system exists for communicating between department of blood centers and hospitals, to satisfy blood necessity, to buy, sale and stock the blood, to give information about this blood. Manual systems as compared to Computer Based Information Systems are time consuming, laborious, and costly. This paper introduces the, BCloud Blood Donor Contact Manager Web Services allows you to maintain a free, easy-to- organize database of contacts and their blood groups. With this convenient—and potentially lifesaving--app, you can quickly check your Android device for someone with a certain blood type and contact them immediately

1.2 Proposed Model:

In our device we have provided different buttons which represents the blood groups, after finding the blood group of the patient, press the desired blood group button in the device so this device will find the blood stock in blood bank automatically. using the pre provided data base.

If there is no stock of that particular blood group it will retrieve contact data of blood donor with similar blood group from blood bank server. The blood required data was sent to all the via Internet of things app and SMS. And this message having GPS location, contact number.

After sending the alert to same blood group donor as primary part. Alert will send to the other blood group person also because the reference purpose.

Waiting for the "Yes" or "No" response from the donor which is shown as pop-up notification on our mobile app installed on donor's mobile.

If the response from the donor is "YES", then data of the donor which contains contact information such as, address, contact number, location etc., shown to this same device which is previously used to detect blood group of the person.

If we got the required blood unit, we will press a notice button that will send message that "Need of blood is satisfied" to all the users. To avoid spamming with same requests over and again.

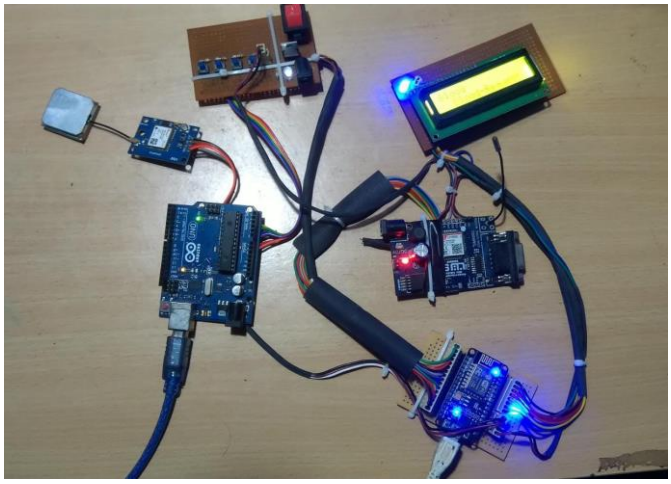
Advantages:

- The installation of mobile application is not compulsory.
- There is no need of internet to communicate with user and donor in our system.
- This system doesn't need any smart phone to communicate with donor and user. It can also be operated with low end mobiles.
- Primary device only needs an internet connection even low power, Low bandwidth of internet is enough to perform the task efficiently and save life.
- This system will efficiently reach 1000+ people within 1 minute.so, the communication between donor and receiver is made more faster.

CHAPTER 2

2.1 Technical explanation:

Motivation of this system is to reduce time, reach number of donors as quick and low power and cost of maintain. Embedded System and Mobile application are the two systems integrated commonly to a database which is called as Google firebase.



1. Donor communication Primary system is this hardware kit.
2. Donor communication Secondary system is mobile application.

Explanation:

Accident occurs in the road and lose of blood is common thing happens now a days. As well for arranging blood for the patient is still being a traditional task like Contacting with friends and requesting in blood bank and sometimes this leads miscommunication and irresponsible people's action may lead to be a loss of patient's life.

- Collecting blood group data of patient from their license, Id card
- We can request blood from our system using the collected data.
- This is very simple operating device. We have separate buttons for each blood group. Just pressing the concern button will immediately start the requesting process automatically.
- After pressing the button, the particular required blood type was searched in the blood bank database. If that found immediately that was arranged to the patient else the contact data of donors will be collected by the system automatically and undergo further steps.
- The contact data of similar blood type donors get emergency alert via simple

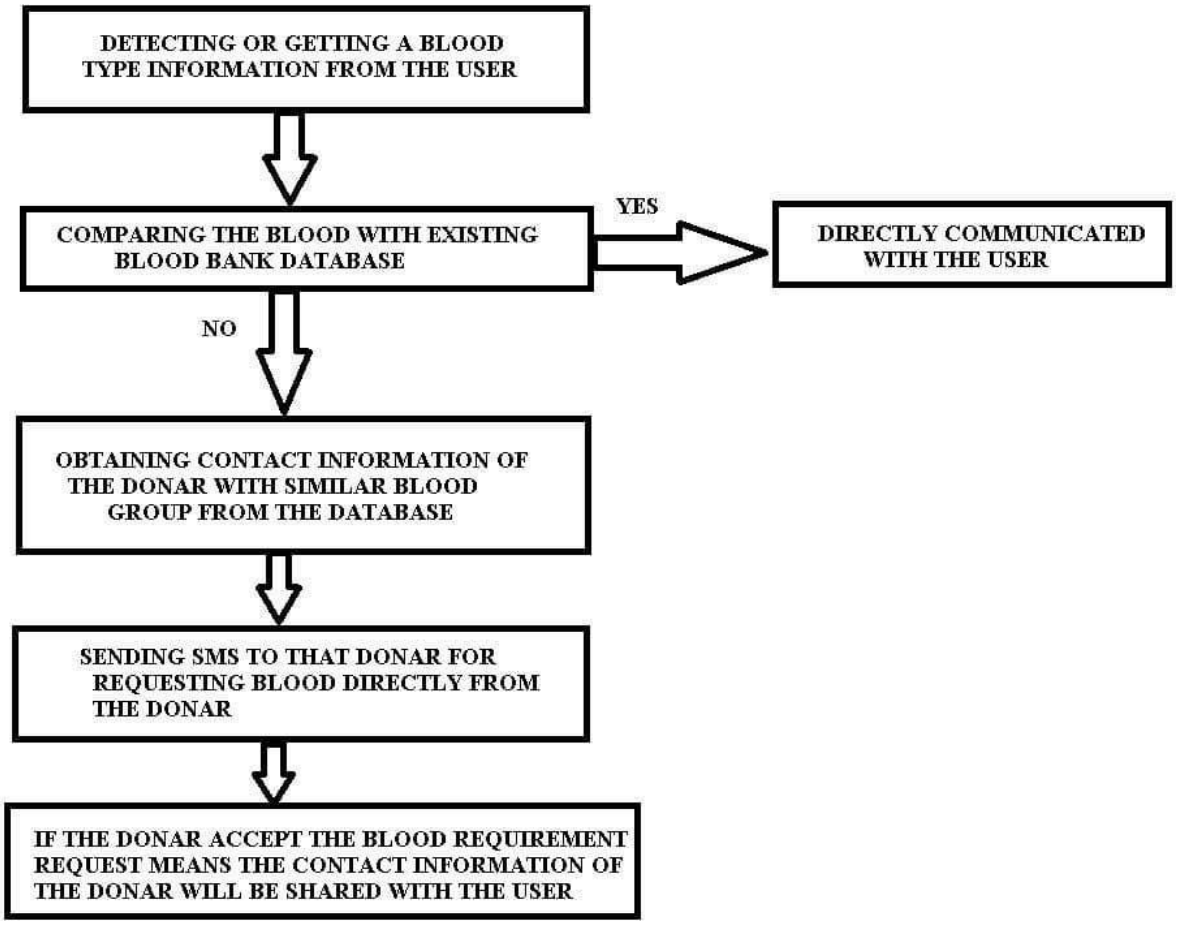
message as SMS. Simple way of alert has no dependencies like internet, smart phone etc.

- Contact of non-similar blood type persons have notified by an emergency alert via Mobile application. Here we focus on their family members with similar blood group reach easily the blood required patient. Also focus to reach large number of donors via IOT.
- After getting notified the donor work is to Accept the request or get back to the system via SMS or Call. If they accept the request means their contact details will shared automatically to this primary Communication device.
- The Accepted donors contact details are displayed in the primary device's LCD display.

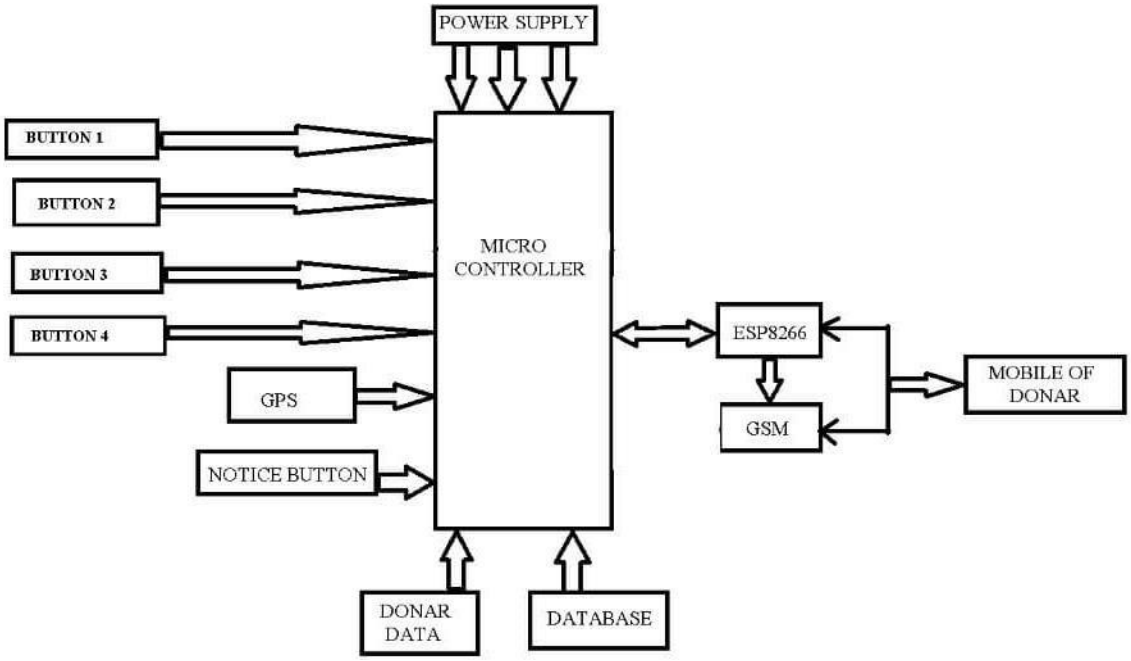
Methodology

- I have used Embedded C programming to develop the hardware primary device as Donor communication system.
- In this primary system I have integrated ESP8266, Arduino Uno, GSM, GPS, LCD 16x2, Power supply module.
    1. ESP8266 is used for connect the device with local WIFI and connect with database server. Especially this is master controller of this system.
    2. Arduino is connected as slave device to get the data's from GSP and Respective blood type buttons and transfer the data with master controller.
    3. GSM and LCD was integrated with Master controller ESP8266.
    4. The Master and Slave was integrated using the I2C concept for performance efficiency.
    5. GPS is integrated for filter the donor contact within nearby locations to send Alert.
    6. GSM is used to send SMS alert to the donors.
- Secondary system of donor communication system is android application and that was developed using the Flutter. This application is used to get notification and also give request for blood in case of emergency.
- Primary and secondary system are integrated with Google firebase.

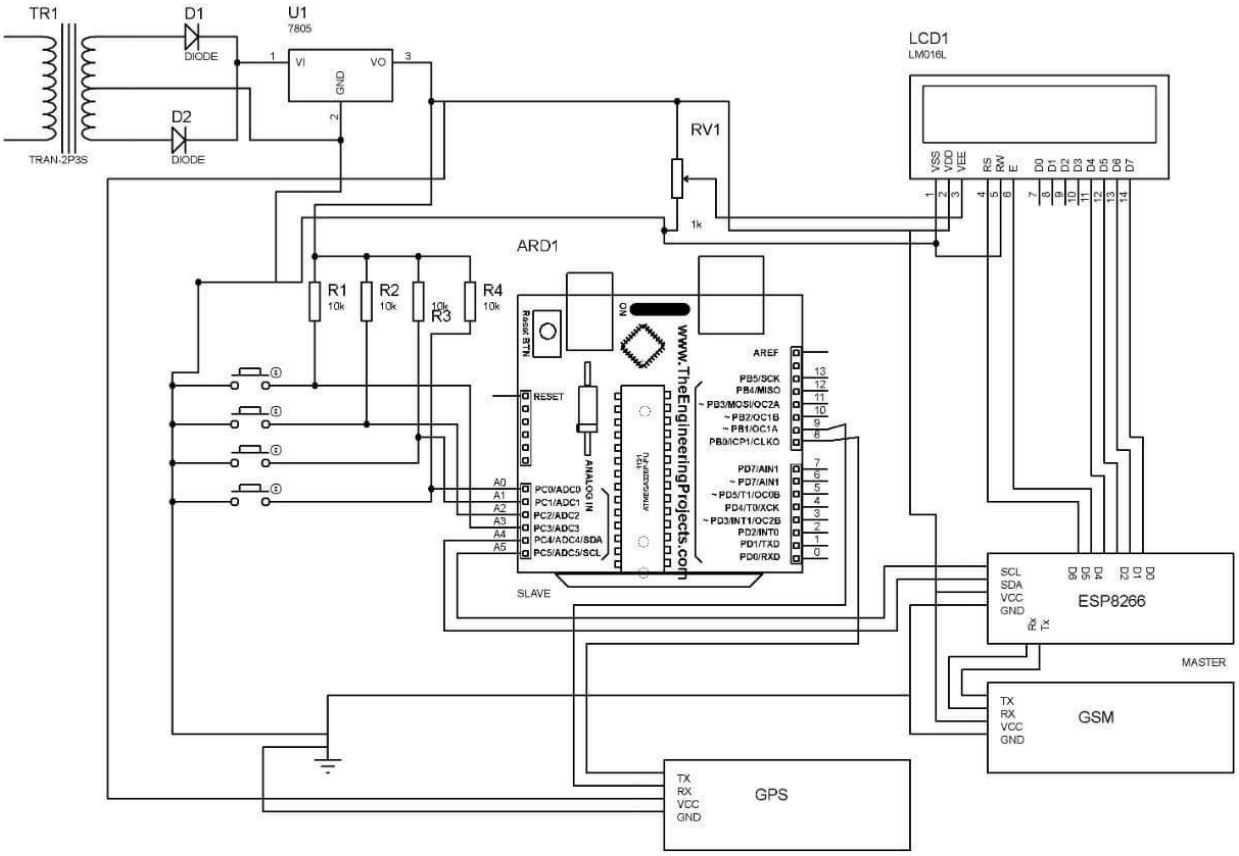
2.2Work flow diagram:



2.3 Block diagram:



2.4 Circuit diagram:



CHAPTER 3

SOFTWARE:

The software used in the project are explained below

3.1 Flutter:

Flutter is an open-source UI software development kit created by Google. It is used to develop applications for Android, iOS, Linux, Mac, Windows, Google Fuchsia, and the web from a single codebase. The first version of Flutter was known as codename "Sky" and ran on the Android operating system. It was unveiled at the 2015 Dart developer summit, with the stated intent of being able to render consistently at 120 frames per second. During the keynote of Google Developer Days in Shanghai, Google announced Flutter Release Preview 2, which is the last big release before Flutter 1.0. On December 4, 2018, Flutter 1.0 was released at the Flutter Live event, denoting the first "stable" version of the Framework. On December 11, 2019, Flutter 1.12 was released at the Flutter Interactive event. On May 6, 2020, the Dart SDK in version 2.8 and the Flutter in version 1.17.0 were released, where support was added to the Metal API, improving performance on iOS devices (approximately 50%), new Material widgets, and new network tracking. On March 3, 2021, Google released Flutter 2 during an online Flutter Engage event. This major update brought official support for web-based applications as well as early-access desktop application support for Windows, MacOS, and Linux. Framework architecture Dart platform Flutter engine Foundation library Design-specific widgets Widgets See also References External links The major components of Flutter include: Dart platform Contents Framework architecture Flutter engine Foundation library Design-specific widgets Flutter DevTools Flutter apps are written in the Dart language and make use of many of the language's more advanced features. On Windows, macOS, and Linux Flutter runs in the Dart virtual machine, which features a just-in-time execution engine. While writing and debugging an app, Flutter uses Just In Time compilation, allowing for "hot reload", with which modifications to source files can be injected into a running application. Flutter extends this with support for stateful hot reload, where in most cases changes to source code are reflected immediately in the running app without requiring a restart or any loss of state. Release versions of Flutter apps are compiled with ahead-of-time (AOT) compilation on both Android and iOS, making Flutter's high performance on mobile devices possible. Flutter's engine, written primarily in C++, provides low-

level rendering support using Google's Skia graphics library. Additionally, it interfaces with platform-specific SDKs such as those provided by Android and iOS. The Flutter Engine is a portable runtime for hosting Flutter applications. It implements Flutter's core libraries, including animation and graphics, file and network I/O, accessibility support, plugin architecture, and a Dart runtime and compile toolchain. Most developers interact with Flutter via the Flutter Framework, which provides a reactive framework and a set of platform, layout, and foundation widgets. The Foundation library, written in Dart, provides basic classes and functions that are used to construct applications using Flutter, such as APIs to communicate with the engine. The Flutter framework contains two sets of widgets that conform to specific design languages: Material Design widgets implement Google's design language of the same name, and Cupertino widgets implement Apple's iOS Human interface guidelines. Flutter uses a variety of widgets to deliver a fully functioning application. These widgets are Flutter's framework architecture.

3.2 Android Studio:

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems or as a subscription-based service in 2020. It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development. Android Studio was announced on May 16, 2013 at the Google I/O conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0. On May 7, 2019, Kotlin replaced Java as Google's preferred language for Android app development. Java is still supported, as is C++.

3.2.1 Feature :

The specific feature of the Android Studio is an absence of the possibility to switch autosave feature off. The following features are provided in the current stable version: Gradle-based build support Android-specific refactoring and

quick fixes Lint tools to catch performance, usability, version compatibility and other problems Contents Features License Binaries: Freeware, Source code: Apache License Website developer.android.com /studio/index.html ProGuard integration and app-signing capabilities Template-based wizards to create common Android designs and components A rich layout editor that allows users to drag-anddrop UI components, option to preview layouts on multiple screen configurations Support for building Android Wear apps Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine [19] Android Virtual Device (Emulator) to run and debug apps in the Android studio. Android Studio supports all the same programming languages of IntelliJ (and CLion) e.g. Java, C++, and more with extensions, such as Go; and Android Studio 3.0 or later supports Kotlin and "all Java 7 language features and a subset of Java 8 language features that vary by platform version." External projects backport some Java 9 features. While IntelliJ states that Android Studio supports all released Java versions, and Java 12, it's not clear to what level Android Studio supports Java versions up to Java 12 (the documentation mentions partial Java 8 support). At least some new language features up to Java 12 are usable in Android

3.3 EMBEDDED C :

Embedded C is one of the most popular and most commonly used Programming Languages in the development of Embedded Systems. So, in this article, we will see some of the Basics of Embedded C Program and the Programming Structure of Embedded C.

Embedded C is perhaps the most popular languages among Embedded Programmers for programming Embedded Systems. There are many popular programming languages like Assembly, BASIC, C++ etc. that are often used for developing Embedded Systems but Embedded C remains popular due to its efficiency, less development time and portability.

Before digging in to the basics of Embedded C Program, we will first take a look

at what an Embedded System is and the importance of Programming Language in Embedded Systems.

3.4 EMBEDDED SYSTEM:

An Embedded System can be best described as a system which has both the hardware and software and is designed to do a specific task. A good example for an Embedded System, which many households have, is a Washing Machine.

We use washing machines almost daily but wouldn't get the idea that it is an embedded system consisting of a Processor (and other hardware as well) and software.

Embedded Systems can not only be stand-alone devices like Washing Machines but also be a part of a much larger system. An example for this is a Car. A modern day Car has several individual embedded systems that perform their specific tasks with the aim of making a smooth and safe. Some of the embedded systems in a Car are Anti-lock Braking System (ABS), Temperature Monitoring System, Automatic Climate Control, Tyre Pressure Monitoring System, Engine Oil Level Monitor, etc.

3.4.1 PROGRAMMING IN EMBEDDED SYSTEM:

As mentioned earlier, Embedded Systems consists of both Hardware and Software. If we consider a simple Embedded System, the main Hardware Module is the Processor. The Processor is the heart of the Embedded System and it can be anything like a Microprocessor, Microcontroller, DSP, CPLD (Complex Programmable Logic Device) and FPGA (Field Programmable Gated Array).

All these devices have one thing in common: they are programmable i.e. we can write a program (which is the software part of the Embedded System) to define how the device actually works.

Embedded Software or Program allow Hardware to monitor external events (Inputs) and control external devices (Outputs) accordingly. During this process, the program for an Embedded System may have to directly manipulate the

internal architecture of the Embedded Hardware (usually the processor) such as Timers, Serial Communications Interface, Interrupt Handling, and I/O Ports etc.

From the above statement, it is clear that the Software part of an Embedded System is equally important to the Hardware part. There is no point in having advanced Hardware Components with poorly written programs (Software).

There are many programming languages that are used for Embedded Systems like Assembly (low-level Programming Language), C, C++, JAVA (high-level programming languages), Visual Basic, JAVA Script (Application-level Programming Languages), etc.

In the process of making a better embedded system, the programming of the system plays a vital role and hence, the selection of the Programming Language is very important.

3.4.2 INTRODUCTION TO EMBEDDED C PROGRAMMING LANGUAGE

Before going in to the details of Embedded C Programming Language and basics of Embedded C Program, we will first talk about the C Programming Language.

The C Programming Language, developed by Dennis Ritchie in the late 60's and early 70's, is the most popular and widely used programming language. The C Programming Language provided low level memory access using an uncomplicated compiler (a software that converts programs to machine code) and achieved efficient mapping to machine instructions.

The C Programming Language became so popular that it is used in a wide range of applications ranging from Embedded Systems to Super Computers.

Embedded C Programming Language, which is widely used in the development of Embedded Systems, is an extension of C Program Language. The Embedded C Programming Language uses the same syntax and semantics of the C Programming Language like main function, declaration of datatypes, defining variables, loops, functions, statements, etc.

The extension in Embedded C from standard C Programming Language include I/O Hardware Addressing, fixed point arithmetic operations, accessing address spaces, etc.

3.4.3 DIFFERENCE BETWEEN C AND EMBEDDED C

There is actually not much difference between C and Embedded C apart from few extensions and the operating environment. Both C and Embedded C are ISO Standards that have almost same syntax, datatypes, functions, etc.

Embedded C is basically an extension to the Standard C Programming Language with additional features like Addressing I/O, multiple memory addressing and fixed-point arithmetic, etc.

C Programming Language is generally used for developing desktop applications whereas Embedded C is used in the development of Microcontroller based applications.

3.4.4 EMBEDDED SYSTEM AND ITS REAL TIME APPLICATIONS

The World is filled with Embedded Systems. The development of Microcontroller has paved path for several Embedded System application and they play a significant role (and will continue to play in the future as well) in our modern day life in one way or the other.

Starting from consumer electronics like Digital Cameras, DVD Players to high end and advanced systems like Flight Controllers and Missile Guidance Systems, embedded systems are omnipresent and became an important part of our life.

The way we live our life has been significantly improved with the utilization of Embedded Systems and they will continue to be an integral part of our lives even tomorrow.

Another important concept we are hearing these days is Real – Time Systems. In a real time system, Real Time Computing takes place, where a computer (an Embedded System) must generate response to events within certain time limits.

Before going in to the details of Real Time Applications of Embedded Systems, we will first see what an Embedded System is, what is a real time system and what is real time operating system.

Some Other Real Time Applications of Embedded Systems

3.4.4.1 Latest Smart TVs
3.4.4.2 GPS Navigation Systems
3.4.4.3 Almost all Modern Day Smart Phones
3.4.4.4 Missile Guidance Systems
3.4.4.5 Space Exploration (Rovers)
3.4.4.6 Automobiles (ABS, Airbags)
3.4.4.7 Industries (Assembly Robots)
3.4.4.8 Road Safety Systems (Traffic Monitoring and Collision Alert Systems)
3.4.4.9 and many other.

3.5 PIC C COMPILER:

CCS provides a method to attempt to make sure you can compile code written in older versions of CCS with minimal difficulty by altering the methodology to best match the desired version. Currently, there are 4 levels of compatibility provided: CCS V2.XXX, CCS V3.XXX, CCS V4.XXX and ANSI.

Notice: this only affects the compiler methodology, it does not change any drivers, libraries and include files that may have been available in previous versions.

#device CCS2

- ADC default size is set to the resolution of the device (#device ADC=10, #device

  ADC=12, etc)

- boolean = int8 is compiled as: boolean = (int8 != 0)

- Overload directive is required if you want to overload functions

- Pointer size was set to only access first bank (PCM *=8, PCB *=5)

  var16 = NegConst8 is compiled as: var16 = NegConst8 & 0xFF (no sign extension)

- Compiler will NOT automatically set certain #fuses based upon certain code conditions.

- rom qualifier is called _rom

#device CCS3

- ADC default is 8 bits (#device ADC=8)

- boolean = int8 is compiled as: boolean = (int8 & 1)

- Overload directive is required if you want to overload functions

- Pointer size was set to only access first bank (PCM *=8, PCB *=5)

- var16 = NegConst8 is compiled as: var16 = NegConst8 & 0xFF (no sign extension)

- Compiler will NOT automatically set certain #fuses based upon certain code conditions.

- rom qualifier is called _rom

#device CCS4

• ADC default is 8 bits (#device ADC=8)

• boolean = int8 is compiled as: boolean = (int8 & 1)

• You can overload functions without the overload directive

• If the device has more than one bank of RAM, the default pointer size is now 16

(#device *=16)

• var16 = NegConst8 is will perform the proper sign extension

• Automatic #fuses configuration (see next section)

#device ANSI

• Same as CCS4, but if there are any discrepancies are found that differ with the ANSI

standard then the change will be made to ANSI

• Data is signed by default

• const qualifier is read-only RAM, not placed into program memory (use rom qualifier to

place into program memory)

• Compilation is case sensitive by default

• Constant strings can be passed to functions

#device (PASS_STRINGS_IN_RAM)

This C compiler, is fully optimized for use with PIC microcontrollers. Built in functions make coding the software very easy. Based on original K&R, the integrated C development environment gives developers a fast method to produce efficient code from an easily Maintainable high-level language.

3.5.1 CAPABILITIES

· Arrays up to 5 subscripts

· Structures and Unions may be nested.

· Custom bit fields (1-8 bits) within structures.

· ENUMurated types,

· CONSTant variables, arrays and strings.

· Full function parameter support (any number).

· Some support for C++ reference parameters.

3.5.2 STANDARD C FUNCTIONS:

· IF, ELSE, WHILE, DO, SWITCH, CASE, FOR,

RETURN, GOTO, BREAK, CONTINUE

· !, ~, ++, --, *, /, %, +, -, <<, >>, <. <=, >, >=,

==, !=, &, ^, |, &&, ||, ?:, =, +=, -=, *=, /=, %=,

>>=, <<=, &=, ^=, |=

· TYPEDEF, STATIC, AUTO, CONST, ENUM, STRUCT, UNION

3.5.3 FEATURES:

3.5.3.1 Built in Libraries for RS232 serial I/O library, I/O, I2C, discrete I/Oand precision delays.

3.5.3.2 Integrates with MPLAB and other simulators/emulators for source level debugging.

3.5.3.3 Standard Hex file and debug files ensure compatibility with all programmers.

3.5.3.4 Formatted Printf allows easy formatting and display in Hex or decimal.

3.5.3.5 Efficient function implementation allows call trees deeper than the hardware stack.

3.5.3.6 Access to hardware from easy to use C functions, Timers, A/D, E2, SSP, PSP, I2C & more.

3.5.3.7 1,8, and 16 bit types.

3.5.3.8 Assembly code may be inserted anywhere in source and may reference C variables.

3.5.3.9 Automatic linking handles multiple code pages.

3.5.3.10 Inline procedures supported; Linker automatically determines optimum architecture or it can be manually specified.

3.5.3.11 Compiler directives determine if tri-state registers are refreshed on

every I/O

3.5.3.12 Constants (including strings and arrays) are saved in program memory.

3.5.3.13 Standard one bit type (Short Int) permits the compiler to generate efficient Bit oriented code.

3.5.3.14 #BIT and #BYTE allow C variables to be placed at absolute addresses to map register to C variables.

3.5.3.15 Reference parameters may be used to improve code readability and inline procedure efficiency.

3.5.3.16 Both an Integrated editor/compiler and a cmd line compiler.

3.5.3.17 Special windows show the RAM memory map, C/Assembly listing and the calling tree.

3.5.3.18 Interrupt procedures supported on PCM. The compiler generates all startup and cleanup code as well as identifying the correct interrupt procedure to be called.

3.5.3.19 Updates via modem for 30 days included.

3.5.4 DESCRIPTION

This integrated C development environment gives developers the capability to quickly produce very efficient code from an easily maintainable high level language. The compiler includes built in functions to access the PIC hardware such as READ_ADC to read a value from the A/D converter. Discrete I/O is handled by describing the port characteristics in a PRAGMA. Functions such as INPUT and OUTPUT_HIGH will properly maintain the tri-state registers. Variables including structures may be directly mapped to memory such as I/O ports to best represent the hardware structure in C. The microcontroller clock speed may be specified in a PRAGMA to permit built in functions to delay for a given number of microseconds or milliseconds. Serial I/O functions allow standard functions such as GETC and PRINTF to be used for RS-232 like I/O. The hardware serial transceiver is used for applicable parts when possible. For all other cases a software serial transceiver is generated by the compiler. The standard C operators and the special built in functions are optimised to produce very efficient code for the bit and I/O functions. Functions may be implemented inline or separate. Function parameters are passed in reusable registers. Inline functions with reference parameters are implemented efficiently with no memory

overhead. During the linking process the program structure including the call tree is analyzed. Functions that call one another frequently are grouped together in the same page. Calls across pages are handled automatically by the tool transparent to the user. Functions may be implemented inline or separate. RAM is allocated efficiently by using the call tree to determine how locations can be re-used. Constant strings and tables are saved in the device ROM. The output hex and debug files are selectable and compatible with popular emulators & programmers including MPLAB for source level debugging. The Professional Package (PCW) provides both compilers in a powerful Windows environment.

3.6 ARDUINO IDE:

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A Software Serial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

3.6.1 PROGRAMMING

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

3.6.1.1 On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.

3.6.1.2 On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

3.6.2 Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

3.7 HARDWARE:

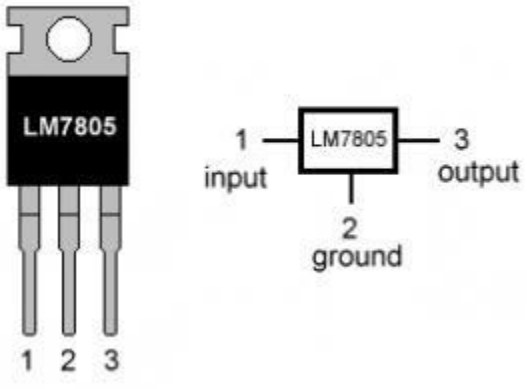
The hardware used in the development of device is explained in detail below.

3.8 REGULATOR 7805:

Voltage sources in a circuit may have fluctuations resulting in not providing fixed voltage outputs. A voltage regulator IC maintains the output voltage at a constant value. 7805 IC, a member of 78xx series of fixed linear voltage regulators used to maintain such fluctuations, is a popular voltage regulator integrated circuit (IC). The xx in 78xx indicates the output voltage it provides. 7805 IC provides +5 volts regulated power supply with provisions to add a heat sink.

All voltage sources cannot able to give fixed output due to fluctuations in the circuit. For getting constant and steady output, the voltage regulators are implemented. The integrated circuits which are used for the regulation of voltage are termed as voltage regulator ICs. Here, we can discuss about IC 7805.The **voltage regulator IC 7805** is actually a member of 78xx series of voltage regulator ICs. It is a fixed linear voltage regulator. The xx present in 78xx represents the value of the fixed output voltage that the particular IC provides. For 7805 IC, it is +5V DC regulated power supply. This regulator IC also adds a provision for a heat sink. The input voltage to this voltage regulator can be up to 35V, and this IC can give a constant 5V for any value of input less than or equal to 35V which is the threshold limit.

LM7805 PINOUT DIAGRAM



3.8.1 7805 IC Rating

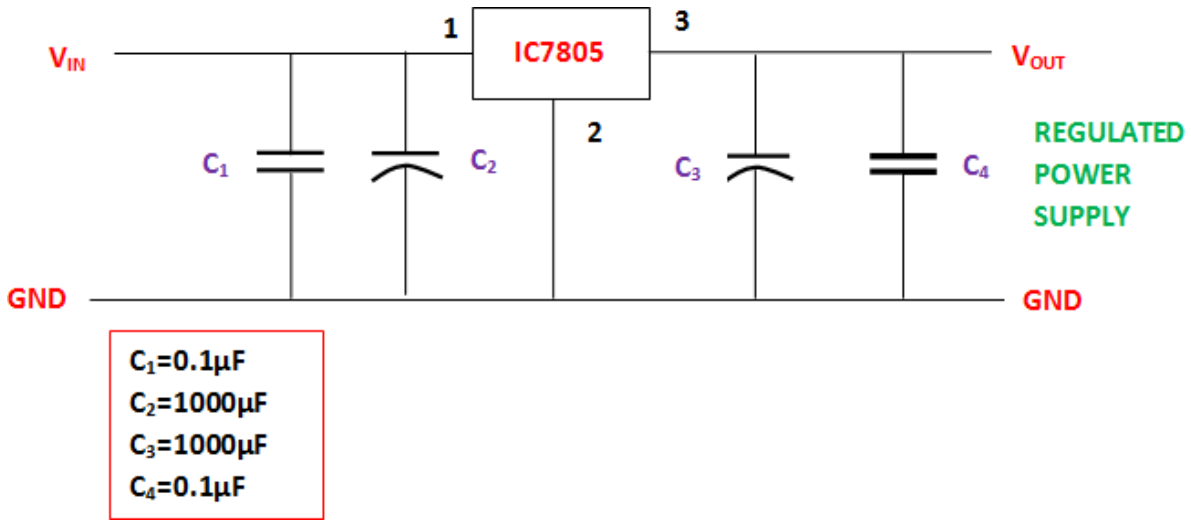
3.8.1.1 Input voltage range 7V- 35V

3.8.1.2 Current rating I_c =1A

3.8.1.3 Output voltage range $V_{Max=5.2V}$, $V_{Min=4.8V}$

Regulated Power Supply Circuit

3.8.1.4 The **voltage regulator 7805** and the other components are arranged in the circuit as shown in figure.



The purposes of coupling the components to the IC7805 are explained below. C_1- It is the bypass capacitor, used to bypass very small extent spikes to the earth. C_2 and C_3- They are the filter capacitors. C_2 is used to make the slow changes in the input voltage given to the circuit to the steady form. C_3 is used to make the slow changes in the output voltage from the regulator in the circuit to the steady form. When the value of these capacitors increases, stabilization is enlarged. But these capacitors single-handedly are unable to filter the very minute changes in the input and output voltages. C_4- like C_1, it is also a bypass capacitor, used to bypass very small extent spikes to the ground or earth. This is done without influencing other components.

Applications of Voltage Regulator 7805 IC

3.8.1.5 Current regulator

3.8.1.6 Regulated dual supply

3.8.1.7 Building circuits for Phone charger, UPS power supply circuits, portable CD player etc

3.8.1.8 Fixed output regulator

3.8.1.9 Adjustable output regulator etc.,

3.9 PUSH BUTTON:

A push-button (also spelled pushbutton) or simply button is a
simple switch mechanism to control some aspect of a machine or a process. Buttons are typically made out of hard material, usually plastic or metal. The surface is usually flat or shaped to accommodate the human finger or hand, so as to be easily depressed or pushed. Buttons are most often biased switches, although many un-biased buttons (due to their physical nature) still require a spring to

return to their un-pushed state. Terms for the "pushing" of a button include pressing, depressing, mashing, slapping, hitting, and punching

USES

The "push-button" has been utilized in calculators, push-button telephones, kitchen appliances, and various other mechanical and electronic devices, home and commercial.

In industrial and commercial applications, push buttons can be connected together by a mechanical linkage so that the act of pushing one button causes the other button to be released. In this way, a stop button can "force" a start button to be released. This method of linkage is used in simple manual operations in which the machine or process has no electrical circuits for control.

Red pushbuttons can also have large heads (called mushroom heads) for easy operation and to facilitate the stopping of a machine. These pushbuttons are called emergency stop buttons and for increased safety are mandated by the electrical code in many jurisdictions. This large mushroom shape can also be found in buttons for use with operators who need to wear gloves for their work and could not actuate a regular flush-mounted push button.

Button shaped as an octagon.

As an aid for operators and users in industrial or commercial applications, a pilot light is commonly added to draw the attention of the user and to provide feedback if the button is pushed. Typically this light is included into the center of the pushbutton and a lens replaces the pushbutton hard center disk. The source of the energy to illuminate the light is not directly tied to the contacts on the back of the pushbutton but to the action the pushbutton controls. In this way a

start button when pushed will cause the process or machine operation to be started and a secondary contact designed into the operation or process will close to turn on the pilot light and signify the action of pushing the button caused the resultant process or action to start.

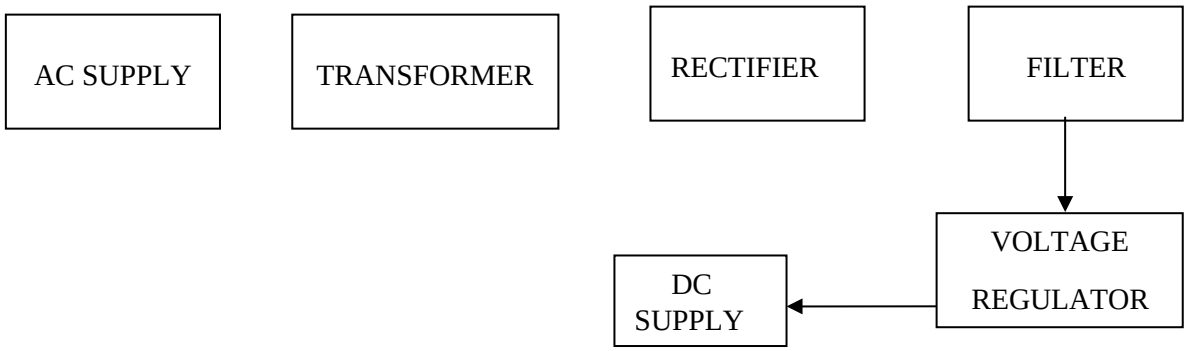
3. 10 POWER SUPPLY

3.10.1 Definition:

A power supply (sometimes known as a power supply unit or PSU) is a device or system that supplies electrical or other types of energy to an output load or group of loads. The term is most commonly applied to electrical energy supplies, less often to mechanical ones, and rarely to others**.**

3.10.2 POWER SUPPLY UNIT BLOCK

All digital circuits work only with low DC voltage. A power supply unit is required to provide the appropriate voltage supply. This unit consists of transformer, rectifier, filter and a regulator. AC voltage typically of 230Vrms is connected to a transformer which steps that AC voltage down to the desired AC voltage level. A diode rectifier then provides a full wave rectified voltage that is initially filtered by a simple capacitor filter to produce a DC voltage. This resulting DC voltage usually has some ripple or AC voltage variations. Regulator circuit can use this DC input to provide DC voltage that not only has much less ripple voltage but also remains in the same DC value, even when the DC voltage varies, or the load connected to the output DC voltage changes. The required DC supply is obtained from the available AC supply after rectification, filtration and regulation. Block diagram of power supply unit is shown in Figure .



Power Supply Block Diagram

The main components used in the power supply unit are Transformer,

Rectifier, Filter and Regulator. The 230V AC supply is converted into 9V AC supply through the transformer. The output of the transformer has the same frequency as in the input AC power. This AC power is converted into DC power through diodes. Here the bridge diode is used to convert AC supply to the DC power supply. This converted DC power supply has the ripple content and for normal operation of the circuit, the ripple content of the DC power supply should be as low as possible. Because the ripple content of the power supply will reduce the life of the circuit. So to reduce the ripple content of the DC power supply, the large value of capacitance filter is used.

This filtered output will not be the regulated voltage. For this purpose IC7805 regulator IC is used in the circuit.

3.11 TRANSFORMER

Transformer is a device used either for stepping-up or stepping-down the AC supply voltage with a corresponding decreases or increases in the current. Here, a transformer is used for stepping-down the voltage so as to get a voltage that can be regulated to get a constant 5V. **0**A rectifier is a device like semiconductor, capable of converting sinusoidal input waveform units into a unidirectional waveform, with a nonzero average component.

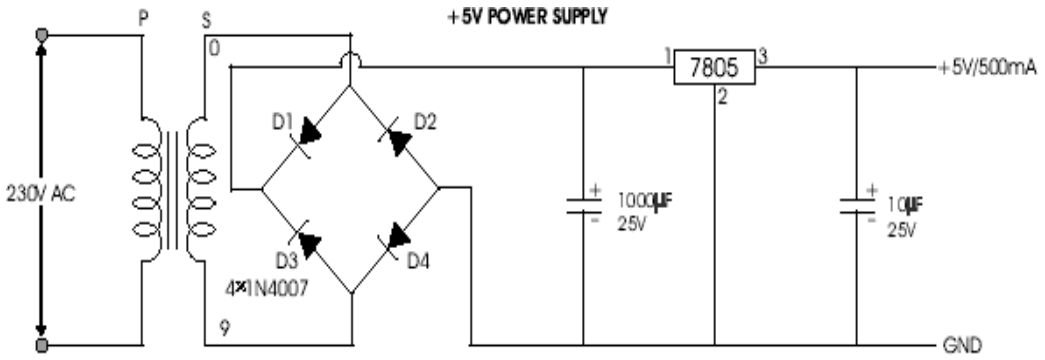
3.12 FILTERS

Capacitors are used as filters in the power supply unit. The action of the system depends upon the fact, that the capacitors stores energy during the conduction period and delivers this energy to the load during the inverse or non-conducting period. In this way, time during which the current passes through the load is prolonged and ripple is considerably reduced.

3.13 VOLTAGE REGULATOR

The LM78XX is three terminal regulator available with several fixed output voltages making them useful in a wide range of applications. IC7805 is a fixed voltage regulators used in this circuit.

Circuit diagram of such power supply is as shown in Figure 3.2



3.14 LIQUID CRYSTAL DISPLAY (LCD): -

LCD is a type of display used in digital watches and many portable computers. LCD displays utilize to sheets of polarizing material with a liquid crystal solution between them. An electric current passed through the liquid causes the crystals to align so that light cannot pass through them. LCD technology has advanced very rapidly since its initial inception over a decade ago for use in lap top computers. Technical achievements has resulted in brighter displace, higher resolutions, reduce response times and cheaper manufacturing process.

The liquid crystals can be manipulated through an applied electric voltage so that light is allowed to pass or is blocked. By carefully controlling where and what wavelength (color) of light is allowed to pass, the LCD monitor is able to display images. A backlight provides LCD monitor's brightness.

Over the years many improvements have been made to LCD to help enhance resolution, image, sharpness and response times.

One of the latest such advancement is applied to glass during acts as switch allowing control of light at the pixel level, greatly improving LCD's ability to display small-sized fonts and image clearly.

Other advances have allowed LCD's to greatly reduce liquid crystal cell response times. Response time is basically the amount of time it takes for a pixel to "change colors", in reality response time is the amount of time it takes a liquid crystal cell to go from being active to inactive.

This is due to following reasons:
The declining prices of LCDs.

The ability to display numbers, characters and graphics. This is in contrast to LEDs, which are limited to numbers and a few characters.

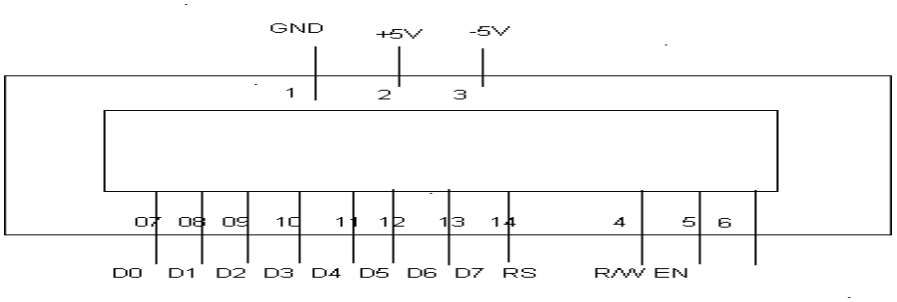
An intelligent LCD display of two lines, 20 characters per line that is interfaced to the pic16f72 microcontroller.

Incorporation of a refreshing controller into the LCD, thereby relieving the CPU to keep displaying the data. Ease of programming for characters and graphics.

Most of the LCD modules conform to a standard interface specification. A 14-pin access is provided having eight data lines, three control lines and three power lines. The connections are laid out in one of the two common configurations, either two rows of seven pins, or a single row of 14 pins.

One of these pins is numbered on the LCD's printed circuit board (PCB), but if not, it is quite easy to locate pin1. Since this pin is connected to ground, it often has a thicker PCB track, connected to it, and it is generally connected to metal work at same point.

3.14.1 PIN DIAGRAM OF LCD: -



3.14.2 PIN DESCRIPTIONS: -

Vcc, Vss and Vee: -

While Vcc and Vss provide +5V and ground respectively, Vee is used for controlling LCD contrast.

RS Register Select: -

There are two very important registers inside the LCD. The RS pin is used for their selection as follows.

If RS=0, the instruction command code register is selected, allowing the user to send a command such as clear display, cursor at home, etc.

If RS=1, the data register is selected, allowing the user to send data to be displayed on the LCD.

R/W, read/write: -

R/W input allows the user to write information to the LCD or read information from it.

R/W = 1 for reading.

R/W= 0 for writing.

EN, enable: -

The LCD to latch information presented to its data pins uses the enable pin. When data is supplied to data pins, a high–to-low pulse must be applied to this pin in

order for the LCD to latch in the data present at the data pins. This pulse must be a minimum of 450 ns wide.

D0 – D7: -

The 8–bit data pins, DO – D7, are used to send information to the LCD or read the contents of the LCD's internal registers.

To display letters and numbers, we send ASCII codes for the letters A–Z, a-z numbers 0-9 to these pins while making RS=1.

There are also instruction command codes that can be sent to the LCD to clear the display or force the cursor to home position or blink the instruction command codes.

We also use RS = 0 to check the busy flag bit to see if the LCD is ready to receive information. The busy flag is D7 and can be read when R/W=1 and RS=0, as follows: if R/W = 1, RS = 0. When D7= 1 (busy flag = 1), the LCD is busy taking care of internal operations and will not accept any information.

Applications

- Automatic Street Light
- Detect Day or Night
- Automatic Head Light Dimmer
- Position sensor
- Used along with LED as obstacle detector
- Automatic bedroom Lights
- Automatic Rear view mirror

3.15 GSM

The GSM system is the most widely used cellular technology in use in the world today. It has been a particularly successful cellular phone technology for a variety of reasons including the ability to roam worldwide with the certainty of being able to be able to operate on GSM networks in exactly the same way - provided billing agreements are in place.

The letters GSM originally stood for the words Groupe Speciale Mobile, but as it became clear this cellular technology was being used world wide the meaning of GSM was changed to Global System for Mobile Communications. Since this cellular technology was first deployed in 1991, the use of GSM has grown steadily, and it is now the most widely cell phone system in the world. GSM reached the 1 billion subscriber point in February 2004, and is now well over the 3 billion subscriber mark and still steadily increasing.

3.15.1 GSM - SYSTEM OVERVIEW

The GSM system was designed as a second generation (2G) cellular phone technology. One of the basic aims was to provide a system that would enable greater capacity to be achieved than the previous first generation analogue systems. GSM achieved this by using a digital TDMA (time division multiple access approach). By adopting this technique more users could be accommodated within the available bandwidth. In addition to this, ciphering of the digitally encoded speech was adopted to retain privacy. Using the earlier analogue cellular technologies it was possible for anyone with a scanner receiver to listen to calls and a number of famous personalities had been "eavesdropped" with embarrassing consequences.

3.15.2 GSM SERVICES

Speech or voice calls are obviously the primary function for the GSM cellular system. To achieve this the speech is digitally encoded and later decoded using a vocoder. A variety of vocoders are available for use, being aimed at different scenarios.

In addition to the voice services, GSM cellular technology supports a variety of other data services. Although their performance is nowhere near the level of those provided by 3G, they are nevertheless still important and useful. A variety of data services are supported with user data rates up to 9.6 kbps. Services including Group 3 facsimile, videotext and teletex can be supported.

One service that has grown enormously is the short message service. Developed as part of the GSM specification, it has also been incorporated into other cellular technologies. It can be thought of as being similar to the paging service but is far more comprehensive allowing bi-directional messaging, store and forward delivery, and it also allows alphanumeric messages of a reasonable length. This service has become particularly popular, initially with the young as it provided a simple, low fixed cost.

3.15.3 GSM BASICS

The GSM cellular technology had a number of design aims when the development started:

- It should offer good subjective speech quality
- It should have a low phone or terminal cost
- Terminals should be able to be handheld
- The system should support international roaming
- It should offer good spectral efficiency

- The system should offer ISDN compatibility

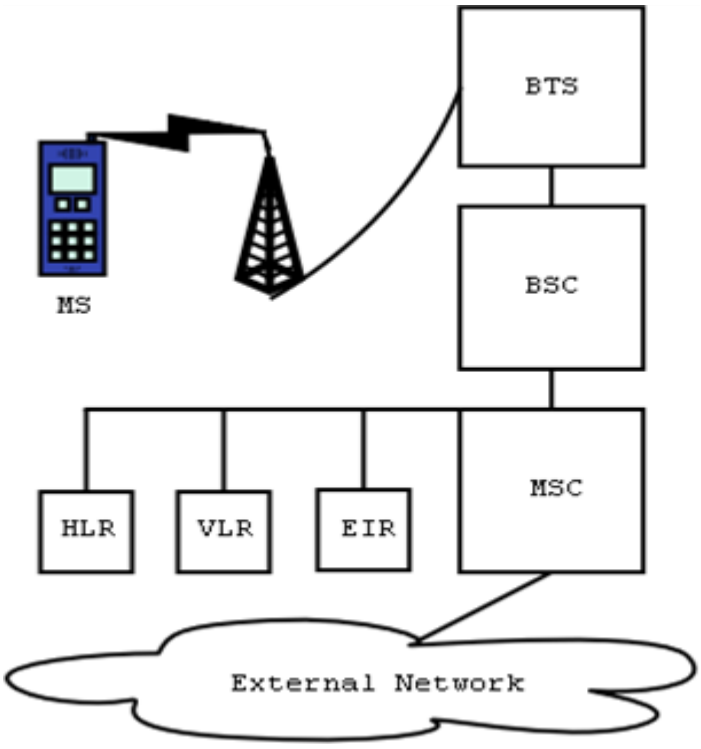
The resulting GSM cellular technology that was developed provided for all of these. The overall system definition for GSM describes not only the air interface but also the network or infrastructure technology. By adopting this approach it is possible to define the operation of the whole network to enable international roaming as well as enabling network elements from different manufacturers to operate alongside each other, although this last feature is not completely true, especially with older items.

GSM cellular technology uses 200 kHz RF channels. These are time division multiplexed to enable up to eight users to access each carrier. In this way it is a TDMA / FDMA system.

The base transceiver stations (BTS) are organised into small groups, controlled by a base station controller (BSC) which is typically co-located with one of the BTSs. The BSC with its associated BTSs is termed the base station subsystem (BSS).

Further into the core network is the main switching area. This is known as the mobile switching centre (MSC). Associated with it is the location registers, namely the home location register (HLR) and the visitor location register (VLR) which track the location of mobiles and enable calls to be routed to them. Additionally there is the Authentication Centre (AuC), and the Equipment Identify Register (EIR) that are used in authenticating the mobile before it is allowed onto the network and for billing. The operation of these are explained in the following pages.

Last but not least is the mobile itself. Often termed the ME or mobile equipment, this is the item that the end user sees. One important feature that was first implemented on GSM was the use of a Subscriber Identity Module. This card carried with it the users identity and other information to allow the user to upgrade a phone very easily, while retaining the same identity on the network. It was also used to store other information such as "phone book" and other items. This item alone has allowed people to change phones very easily, and this has fuelled the phone manufacturing industry and enabled new phones with additional features to be launched. This has allowed mobile operators to increase their average revenue per user (ARPU) by ensuring that users are able to access any new features that may be launched on the network requiring more sophisticated phones.



3.15.4 FURTHER DEVELOPMENTS OF GSM

GSM was a particularly successful mobile telecommunications system. Initially it had been intended for use within Europe, but within a relatively short while the system was being used well beyond the borders of Europe, becoming an internationally accepted system.

In addition to its success as a voice communications system, it was developed beyond the basic voice capability to be able to carry data. With the Internet becoming more widely used, GSM was developed to provide a packet data capability. The first major development was in the form of GPRS.

3.16 ESP8266

Espressif's ESP8266EX delivers highly integrated Wi-Fi SoC solution to meet users' continuous demands for efficient power usage, compact design and reliable performance in the Internet of Things industry. With the complete and self-contained Wi-Fi networking capabilities, ESP8266EX can perform either as a standalone application or as the slave to a host MCU. When ESP8266EX hosts the application, it promptly boots up from the flash. The integrated highspeed cache helps to increase the system performance and optimize the system memory. Also, ESP8266EX can be applied to any microcontroller design as a Wi-Fi adaptor through SPI / SDIO or I2C / UART interfaces. ESP8266EX integrates antenna switches, RF balun, power amplifier, low noise receive amplifier, filters and power management modules. The compact design minimizes the PCB size and requires minimal external circuitries. Besides the Wi-Fi functionalities, ESP8266EX also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor and on-chip SRAM. It can be interfaced with external sensors and other devices through the GPIOs. Software Development Kit (SDK) provides sample codes for various applications. Espressif Systems' Smart Connectivity Platform (ESCP) enables sophisticated features including fast switch between sleep and wakeup mode for energy-efficient purpose, adaptive radio biasing for low-power operation, advance signal processing, spur cancellation and radio co-existence mechanisms for common cellular, Bluetooth, DDR, LVDS, LCD interference mitigation

3.17 BATTERY:

A lead-acid storage battery is an electrochemical device that produces voltage and delivers electrical current. The battery is the primary "source" of electrical energy used in our device. It's important to remember that a battery does not store electricity, but rather it stores a series of chemicals, and through a chemical process electricity is produced. Basically, two different types of lead in an acid mixture react to produce an electrical pressure called voltage. This electrochemical reaction changes chemical energy to electrical energy and is the basis for all automotive batteries.

BATTERIES - Primary or Secondary

Batteries can either be **a primary cell**, such as a flashlight battery once used, throw it away, or a **secondary cell**, such as a car battery (when the charge is gone, it can be recharged).

PRIMARY CELL: Because the chemical reaction totally destroys one of the metals after a period of time, primary cells cannot be recharged. Small batteries such as flashlight and radio batteries are primary cells.

SECONDARY CELL: The metal plates and acid mixture change as the battery supplies voltage. As the battery drains the metal plates become similar and the acid

strength weakens. This process is called discharging. By applying current to the battery in the reverse direction, the battery materials can be restored, thus recharging the battery.

This process is called charging. Automotive lead-acid batteries are secondary cells and can be recharged.

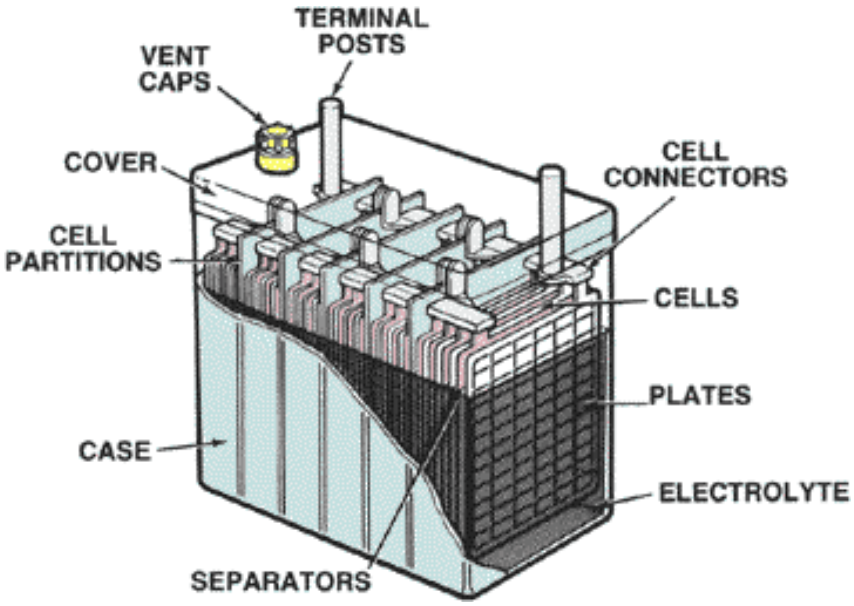
BATTERIES - Wet or Dry Charged

Batteries can be produced as Wet-Charged, such as current automotive batteries are today, or they can be Dry-Charged, such as a motorcycle battery where an electrolyte solution is added when put into service.

WET-CHARGED: The lead-acid battery is filled with electrolyte and charged when it is built. During storage, a slow chemical reaction will cause self-discharge. Periodic charging is required. Most batteries sold today are wet charged.

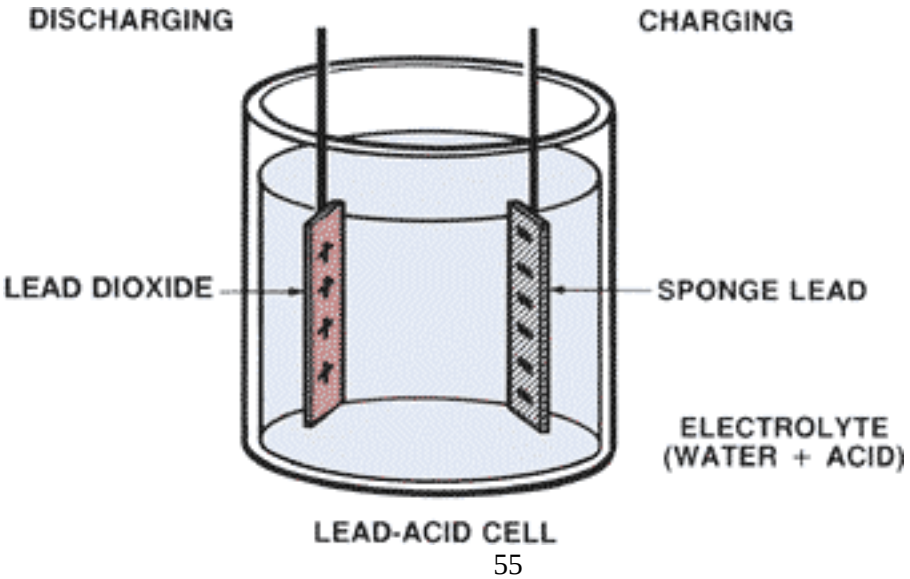
DRY-CHARGED: The battery is built, charged, washed and dried, sealed, and shipped without electrolyte. It can be stored for up to 18 months. When put into use, electrolyte and charging are required. Batteries of this type have a long shelf life. Motorcycle batteries are typically dry charged batteries.

An automobile battery contains a diluted sulfuric acid electrolyte and positive and negative electrodes, in the form of several plates. Since the plates are made of lead or lead-derived materials, this type of battery is often called a lead acid battery. A battery is separated into several cells (usually six in the case of automobile batteries), and in each cell there are several battery elements, all bathed in the electrolyte solution.

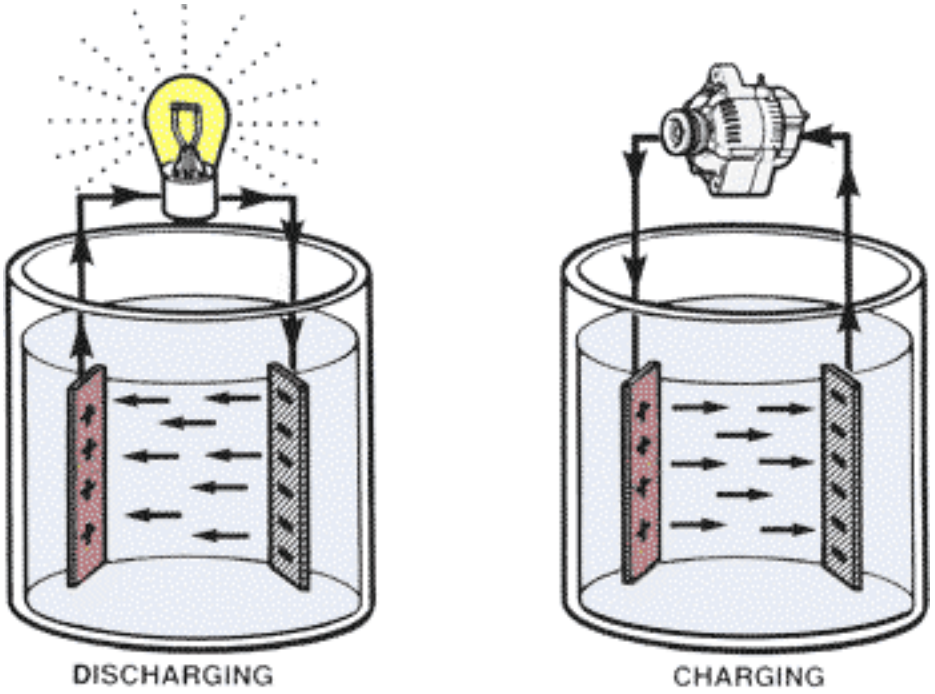


Two dissimilar metals placed in an acid bath produce electrical potential across the poles. The cell produces voltage by a chemical reaction between the plates and the electrolyte. The positive plate is made of reddish-brown material such as Lead Dioxide (PBO2) while the negative plate is made of grayish material called Sponge Lead (PB). The acid bath is a mixture of sulfuric acid and water cell electrolyte. Together a cell element is formed.

chemical reaction processes the battery creates and releases electricity as needed by the



electrical system or devices. Since the battery loses its chemical energy in this process, the battery must be recharged by the alternator. By reversing electrical current flow through the battery the chemical process is reversed, thus charging the battery. The cycle of discharging and charging is repeated continuously and is called "battery cycling".



Battery plates are constructed of a lead alloy containing a percentage of either Antimony or Calcium. The plates are designed as a thin flat grid, grids crossing at right angles (shown below) or grids crossing diagonally at different angles which reduces internal resistance. The grid provides the necessary framework for active material to be pasted onto the plate, making either a positive or a negative plate. The active material on a charged positive plate is a reddish-brown Lead Dioxide (PBO2), while the active material on a charged negative plate is a grayish Sponge Lead (PB)

Advantages:

1. Longer service life than Calcium batteries.
2. Easier to recharge when completely discharged.
3. Lower cost.

CHAPTER 4

HARWARE SOURCE CODE

In this chapter the programs used in the device are showcased.

4.1.1 PROGRAMS:

MASTER CODE:

```
#include <Wire.h>
#include <BearSSLHelpers.h>
#include <CertStoreBearSSL.h>
#include <ESP8266WiFi.h>
#include <ESP8266WiFiAP.h>
#include <ESP8266WiFiGeneric.h>
#include <ESP8266WiFiGratuitous.h>
#include <ESP8266WiFiMulti.h>
#include <ESP8266WiFiScan.h>
#include <ESP8266WiFiSTA.h>
#include <ESP8266WiFiType.h>
#include <WiFiClient.h>
#include <WiFiClientSecure.h>
#include <WiFiClientSecureAxTLS.h>
#include <WiFiClientSecureBearSSL.h>
#include <WiFiServer.h>
#include <WiFiServerSecure.h>
#include <WiFiServerSecureAxTLS.h>
#include <WiFiServerSecureBearSSL.h>
#include <WiFiUdp.h>

#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
```

```
SoftwareSerial mySerial(3,1);
#include "FirebaseESP8266.h"
#define FIREBASE_HOST "final-year-proj-a1883-default-rtdb.firebaseio.com/"   //
"homeautomati-544f5.firebaseio.com"  "home-control-cfa72.firebaseio.com/" //Without
http:// or https:// schemes
#define FIREBASE_AUTH "ccKbPAqd3mRa6k9MscpPH4DVcb3j0Ps1J8JU4l00"    //
"8zXXD07r0aZWf5OAeglB6ghGulmW1v6OdZTK3dcm"
const char* ssid = "DHARANI KANNA"; //replace this with your WiFi network name
const char* password = "Dh@ran1kanna"; //replace this with your WiFi network
password
const int rs = D5, en = D4, d4 = D3, d5 = D2, d6 = D1, d7 = D0;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

FirebaseData firebaseData;
FirebaseData BLOOD_GROUP_Data;
FirebaseData ADDRESS_Data;
FirebaseData CONTACT_Data;
FirebaseData LOCATION_Data;

FirebaseJson json;
String c = "";
String BLOOD_TYPE = "";
//############################################################//
void setup() {
  mySerial.begin(115200);
  Serial.begin(9600); /* begin serial for debug */
  Wire.begin(D6, D7); /* join i2c bus with SDA=D6 and SCL=D7 of NodeMCU */
  Serial.print("Master Ready");
```

```
  lcd.begin(16, 2);

  /*
    Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
    Firebase.reconnectWiFi(true);
  */
  // We start by connecting to a WiFi network
  WiFi_access();
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  //firebase_start();
}
//#####################################################################//

String protocol() {
  Wire.beginTransmission(8); /* begin with device address 8 */
  //Wire.write("Hello Arduino");  /* sends hello string */
  delay(100);
  Wire.endTransmission();    /* stop transmitting */

  Wire.requestFrom(8, 13); /* request & read data of size 13 from slave */
  //Serial.print("Blood Group : ");
  while (Wire.available()) {
    c = Wire.readString();
    //Serial.print(c);
    return c;
  }
  Serial.println();
  //delay(1000);
}
```

```
//###################################################################//

void WiFi_access() {

  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  /* Explicitly set the ESP8266 to be a WiFi-client, otherwise, it by default,
     would try to act as both a client and an access-point and could cause
     network-issues with your other WiFi-devices on your WiFi-network. */

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
}
//###################################################################//
void lcd_data(String data) {
  lcd.begin(16, 2);
  lcd.clear();
  lcd.print("Blood:");
  lcd.println(c);
  delay(1000);
```

```
}
//#####################################################################//
void serial_data(String data) {
  Serial.print("Blood Group:");
  Serial.println(data);
}
//#####################################################################//

void gsm_data(String data) {

  if (data == "A POSITIVE")
  {
    SendMessage_AP();
  }
  else if (data == "A NEGATIVE")
  {
    SendMessage_AN();
  }
  else if (data == "B POSITIVE")
  {
    SendMessage_BP();
  }
  else if (data == "B NEGATIVE")
  {
    SendMessage_BN();
  }
  else
  {
    if (Serial.available() > 0) {
```

```
      if (Serial.read() > 0) {

        RecieveMessage();

      }
    }
  }
}
void SendMessage()
{
  mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
  delay(1000);  // Delay of 1000 milli seconds or 1 second
  mySerial.println("AT+CMGS=\"+919698208798\"\r"); // Replace x with mobile number
  delay(1000);
  mySerial.println("Any Need blood ?");// The SMS text you want to send
  delay(100);
  mySerial.println((char)26);// ASCII code of CTRL+Z
  delay(1000);
}
void SendMessage_BP()
{
  mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
  delay(1000);  // Delay of 1000 milli seconds or 1 second
  mySerial.println("AT+CMGS=\"+919698208798\"\r"); // Replace x with mobile number
  delay(1000);
  mySerial.println("Need of B-Positive blood Group");// The SMS text you want to send
  delay(100);
  mySerial.println((char)26);// ASCII code of CTRL+Z
  delay(1000);
```

```
}

void SendMessage_BN()
{
  mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
  delay(1000);  // Delay of 1000 milli seconds or 1 second
  mySerial.println("AT+CMGS=\"+918248194955\"\r"); // Replace x with mobile number
  delay(1000);
  mySerial.println("Need of B-NEGATIVE blood Group");// The SMS text you want to send
  delay(100);
  mySerial.println((char)26);// ASCII code of CTRL+Z
  delay(1000);
}

void SendMessage_AP()
{
  mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
  delay(1000);  // Delay of 1000 milli seconds or 1 second
  mySerial.println("AT+CMGS=\"+919944523597\"\r"); // Replace x with mobile number
  delay(1000);
  mySerial.println("Need of B-Negative blood Group");// The SMS text you want to send
  delay(100);
  mySerial.println((char)26);// ASCII code of CTRL+Z
  delay(1000);
}

void SendMessage_AN()
{
```

```
  mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
  delay(1000);  // Delay of 1000 milli seconds or 1 second
  mySerial.println("AT+CMGS=\"+917510264405\"\r"); // Replace x with mobile number
  delay(1000);
  mySerial.println("Need of A-Negative blood Group");// The SMS text you want to send
  delay(100);
  mySerial.println((char)26);// ASCII code of CTRL+Z
  delay(1000);
}
void RecieveMessage()
{
  mySerial.println("AT+CNMI=2,2,0,0,0"); // AT Command to receive a live SMS
  delay(1000);
}
//################################################################//
void firebase_data(String data) {
  //if (data!="Blood Request"){
  Firebase.setString(firebaseData, "/Blood Group/", data);

  //}
  //else{
  //   Firebase.setString(firebaseData, "/Pinstate/BLOOD_GROUP", "No Blood request found");
  //}
}
//################################################################//

void loop() {
  BLOOD_TYPE = protocol();  //declaring I2C data value into a variable
```

```
  serial_data(BLOOD_TYPE);          //serial visualisation
  lcd_data(BLOOD_TYPE);    //LCD visualisation
  gsm_data(BLOOD_TYPE);
  firebase_data(BLOOD_TYPE);

}

//#####################################################################//
```

4.1.2 SLAVE CODE:

```
#include <TinyGPS.h>
#include <SoftwareSerial.h>
#include <Wire.h>

SoftwareSerial gpsSerial(8,9);//rx,tx
TinyGPS gps;
float lat = 0, lon = 0; //float lat = 28.5458,lon = 77.1703;
// fix lat, long value

int AP_=A0;
int AN_=A1;
int BP_=A2;
int BN_=A3;
String gd="";
void setup() {
 gpsSerial.begin(9600);
 Wire.begin(8);                /* join i2c bus with address 8 */
 Serial.begin(9600);           /* start serial for debug */
 //Wire.onReceive(receiveEvent); /* register receive event */
 Wire.onRequest(requestEvent); /* register request event */

  //sensors
  pinMode(AP_,INPUT);
  pinMode(AN_,INPUT);
  pinMode(BP_,INPUT);
  pinMode(BN_,INPUT);
```

```
}
//#########################################################################//
String latdata = GPS_DATA_lat();
String londata = GPS_DATA_long();
void loop() {
  int AP=digitalRead(AP_);
  int AN=digitalRead(AN_);
  int BP=digitalRead(BP_);
  int BN=digitalRead(BN_);
  //GPS_DATA();
  //gd = GPS_DATA();
  Serial.println(gd);
 if(AP==0){
  A_POSITIVE();
 }
 else if(AN==0){
  A_NEGATIVE();
 }
 else if(BP==0){
  B_POSITIVE();
 }
 else if(BN==0){
  B_NEGATIVE();
 }
 else{
  Serial.println("Waiting For Blood Request...");
 }
 delay(500);
}
```

```
//#################################################################//
// function that executes whenever data is received from master
void receiveEvent(int howMany) {
 while (0 <Wire.available()) {
    char c = Wire.read();      /* receive byte as a character */
    Serial.print(c);           /* print the character */
  }
 Serial.println();             /* to newline */
}
//#################################################################//
// function that executes whenever data is requested from master
void requestEvent() {
 //Wire.write("Hello NodeMCU");  /*send string on request */

  int AP=digitalRead(AP_);
  int AN=digitalRead(AN_);
  int BP=digitalRead(BP_);
  int BN=digitalRead(BN_);

 if(AP==0){
  Wire.write("A POSITIVE   ");
 }
 else if(AN==0){
  Wire.write("A NEGATIVE   ");
 }
 else if(BP==0){
  Wire.write("B POSITIVE   ");
 }
 else if(BN==0){
```

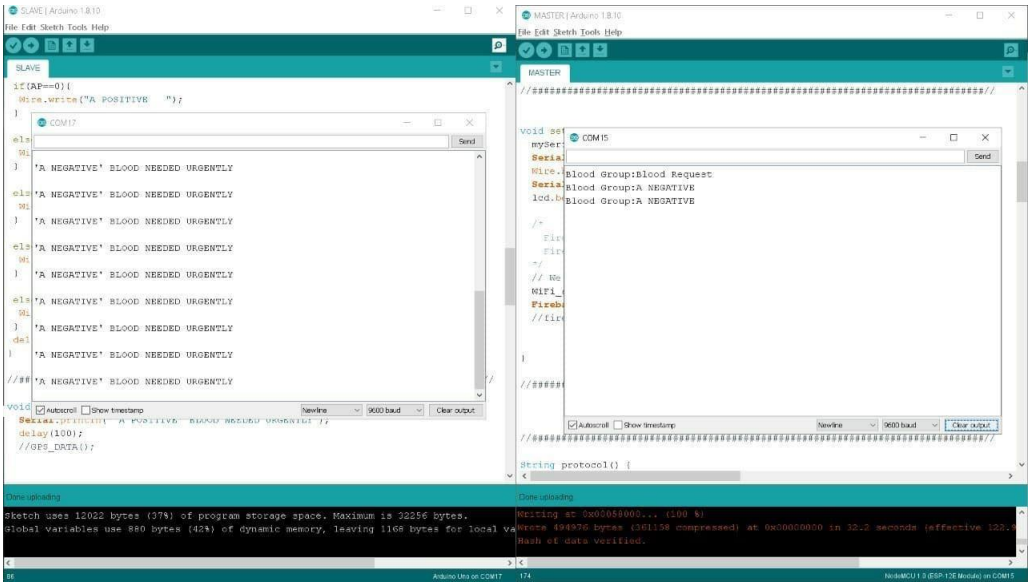
```
  Wire.write("B NEGATIVE    ");
 }
 else{
  Wire.write("Blood Request...");
 }
 delay(100);
}
//#################################################################//
void A_POSITIVE(){
  Serial.println("'A POSITIVE' BLOOD NEEDED URGENTLY");
  delay(100);
  //GPS_DATA();
}
//#################################################################//

void A_NEGATIVE(){
  Serial.println("'A NEGATIVE' BLOOD NEEDED URGENTLY");
  delay(100);
  //GPS_DATA();
}
//#################################################################//

void B_POSITIVE(){
  Serial.println("'B POSITIVE' BLOOD NEEDED URGENTLY");
  delay(100);
  //GPS_DATA();
  }
//#################################################################//
```

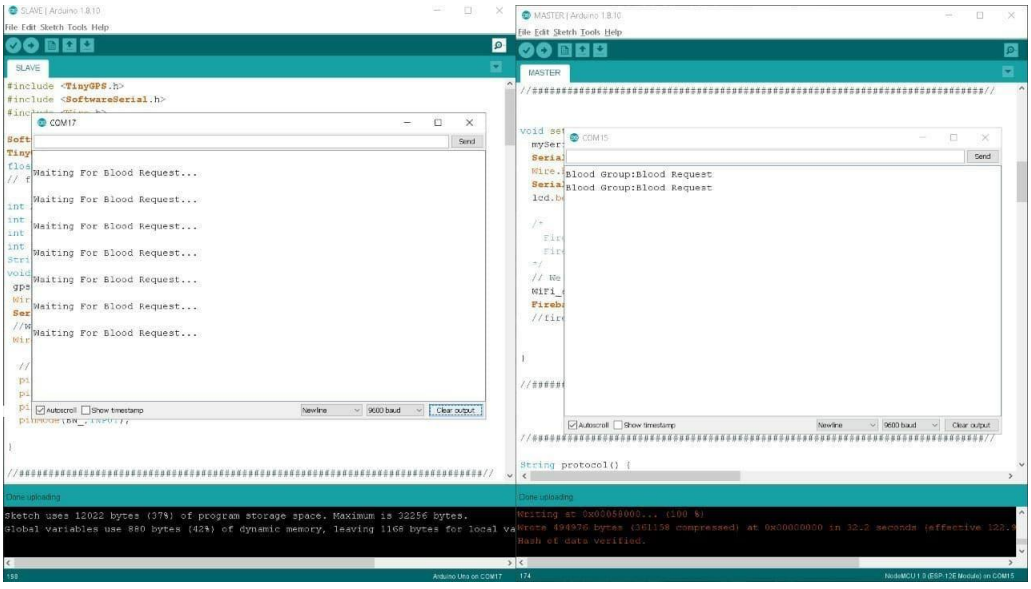
```
void B_NEGATIVE(){
  Serial.println("'B NEGATIVE' BLOOD NEEDED URGENTLY");
  delay(100);
  //GPS_DATA();
}
//######################################################################//
String GPS_DATA_lat(){
   while(gpsSerial.available()){ // check for gps data
  if(gps.encode(gpsSerial.read()))// encode gps data
  {
  gps.f_get_position(&lat,&lon); // get latitude and longitude
  // display position
  //lcd.clear();
  //lcd.setCursor(1,0);
  //lcd.print("GPS Signal");
  /*Serial.print("Position: ");
  Serial.print("Latitude:");
  Serial.print(lat,6);
  Serial.print(";");
  Serial.print("Longitude:");
  Serial.println(lon,6); */

  String latitude = String(lat,6);
  String longitude = String(lon,6);
  String located_value = (latitude+";"+longitude);
 Serial.println(located_value);
 return located_value;
  delay(500);
     } } }
```

```
//#########################################################################//
String GPS_DATA_long(){
   while(gpsSerial.available()){ // check for gps data
  if(gps.encode(gpsSerial.read()))// encode gps data
  {
  gps.f_get_position(&lat,&lon); // get latitude and longitude
  // display position
  //lcd.clear();
  //lcd.setCursor(1,0);
  //lcd.print("GPS Signal");
 /* Serial.print("Position: ");
  Serial.print("Latitude:");
  Serial.print(lat,6);
  Serial.print(";");
  Serial.print("Longitude:");
  Serial.println(lon,6); */

  String latitude = String(lat,6);
  String longitude = String(lon,6);
  String located_value = (latitude+";"+longitude);
 Serial.println(located_value);
 return longitude;
  delay(500);
      }
   }
}
```

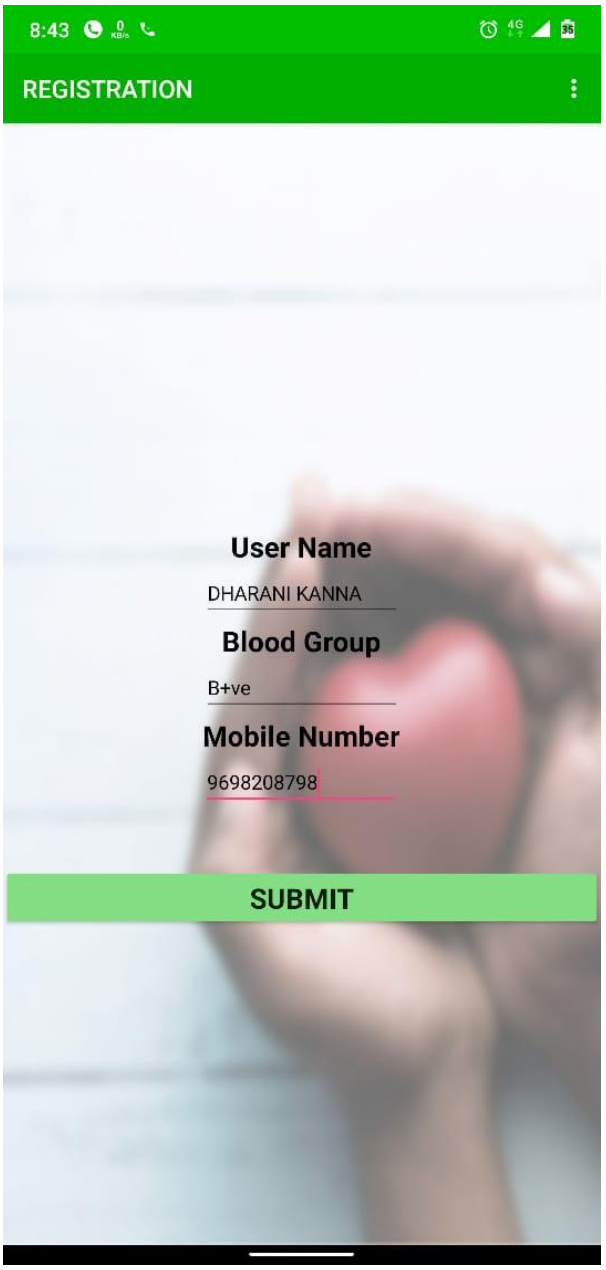
4.2 OUTPUTS

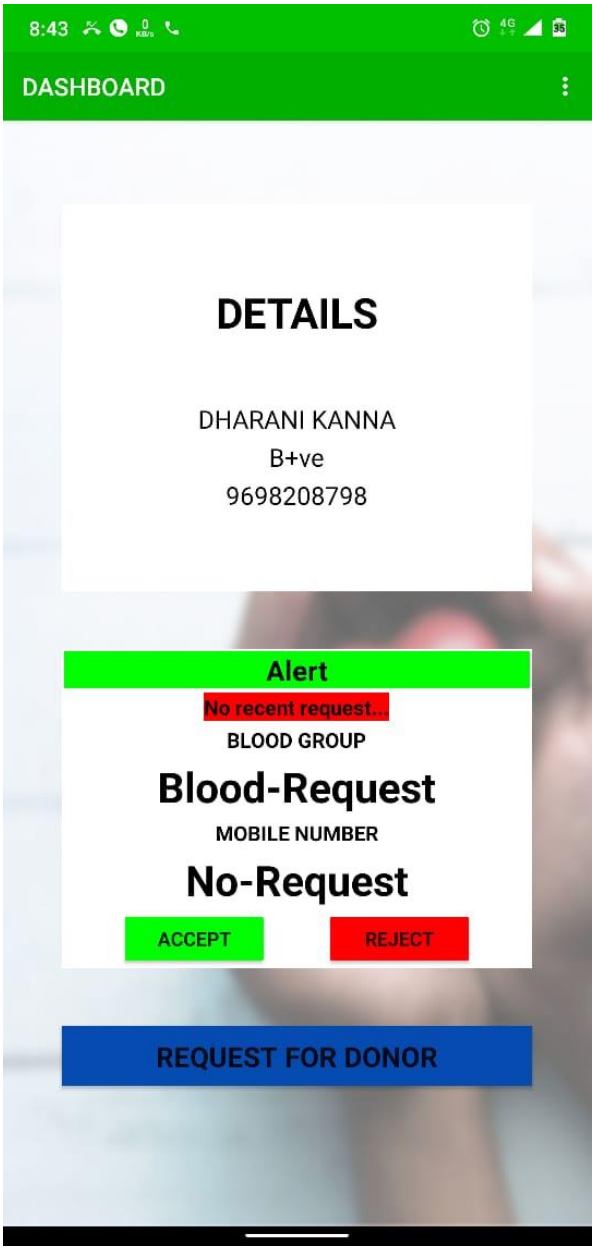
4.2.1 Serial Output:

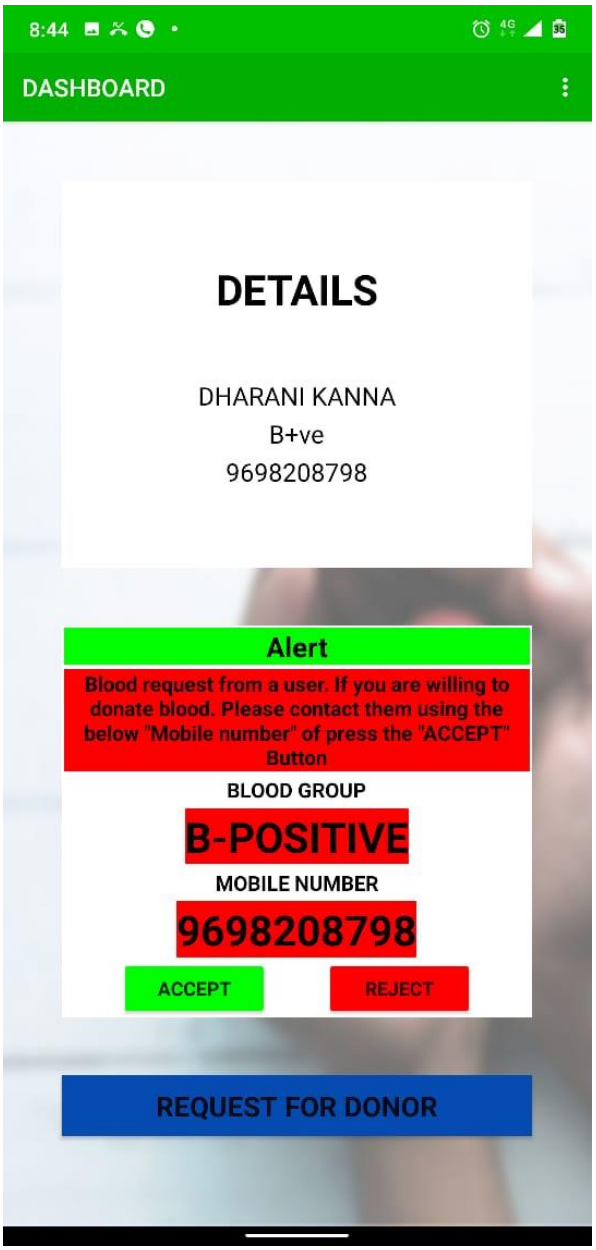


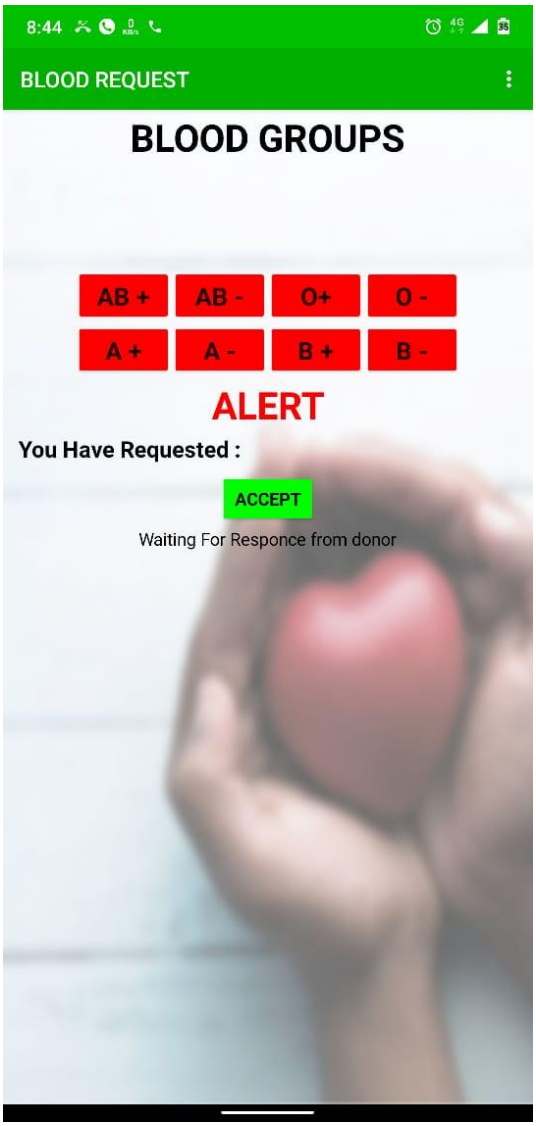


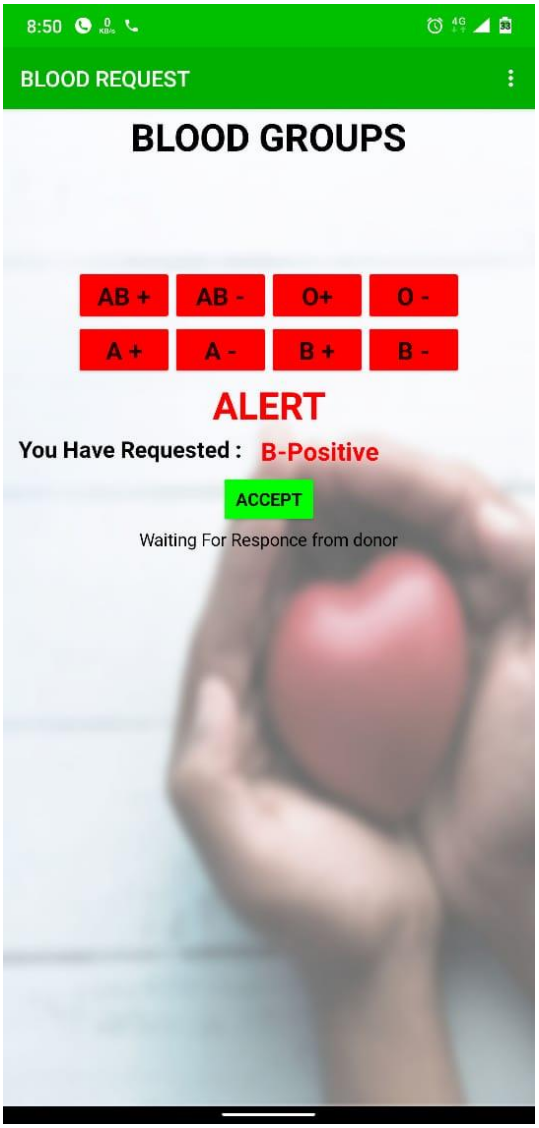
4.2.2 mobile application outputs:



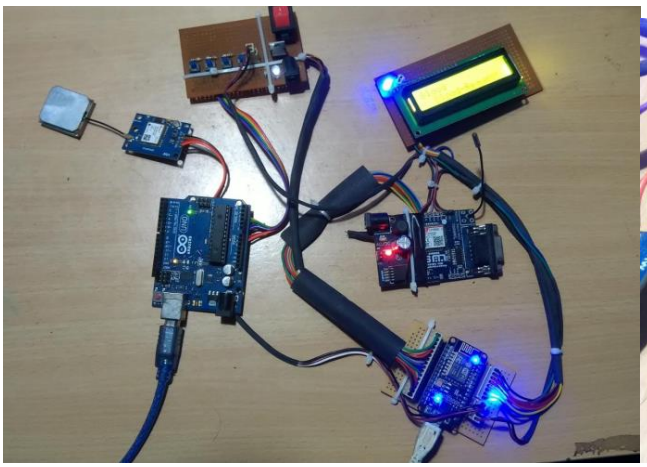

8:43
DASHBOARD
DETAILS
DHARANI KANNA
B+ve
9698208798
Alert
No recent request...
BLOOD GROUP
Blood-Request
MOBILE NUMBER
No-Request
ACCEPT
REJECT
REQUEST FOR DONOR

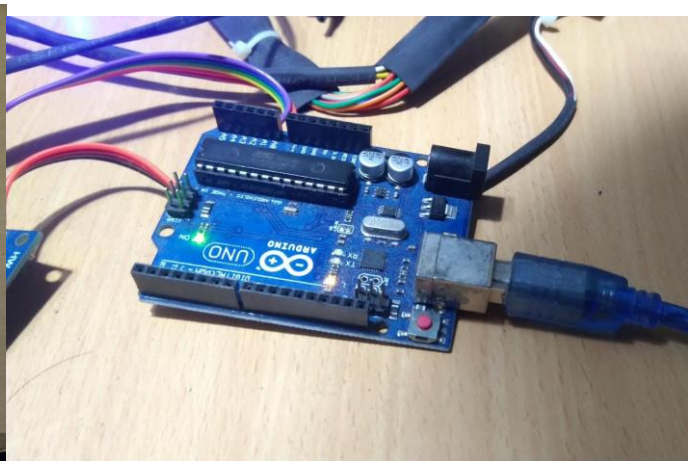

8:44
DASHBOARD
DETAILS
DHARANI KANNA
B+ve
9698208798
Alert
Blood request from a user. If you are willing to donate blood. Please contact them using the below "Mobile number" of press the "ACCEPT" Button
BLOOD GROUP
B-POSITIVE
MOBILE NUMBER
9698208798
ACCEPT
REJECT
REQUEST FOR DONOR

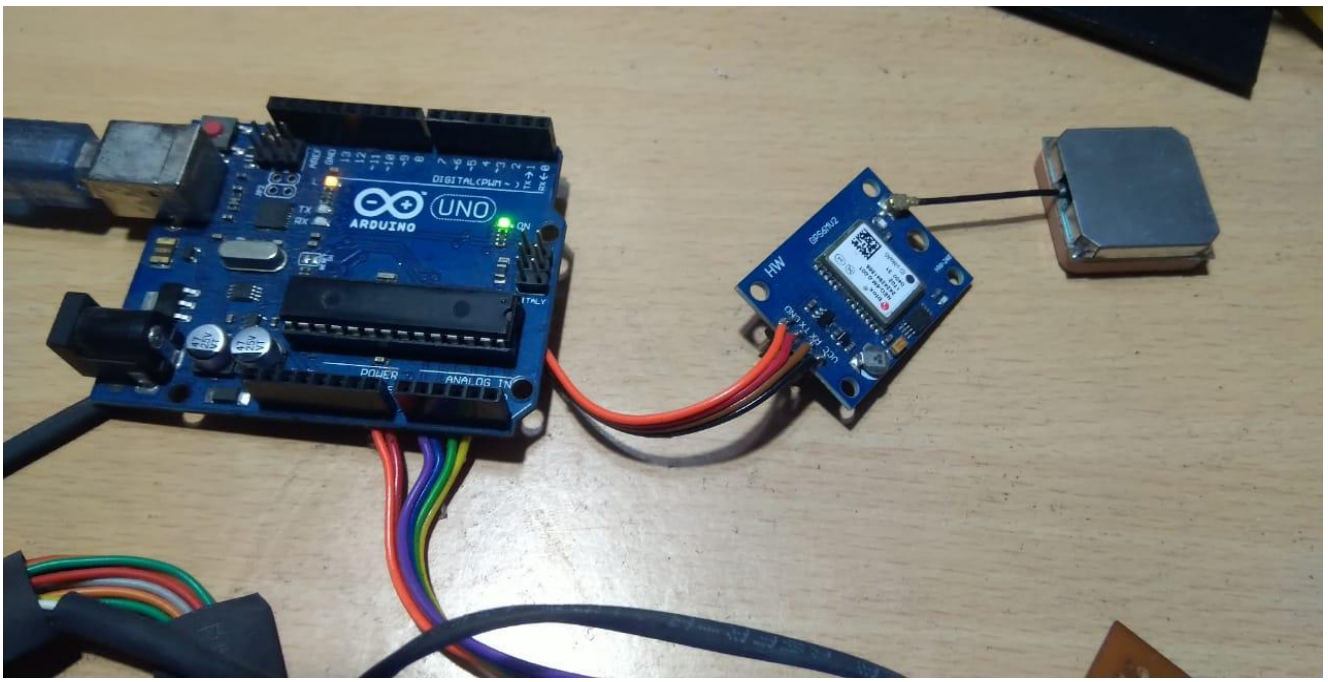

8:44
BLOOD REQUEST
BLOOD GROUPS
AB +
AB -
O+
O -
A +
A -
B +
B -
ALERT
You Have Requested :
ACCEPT
Waiting For Responce from donor

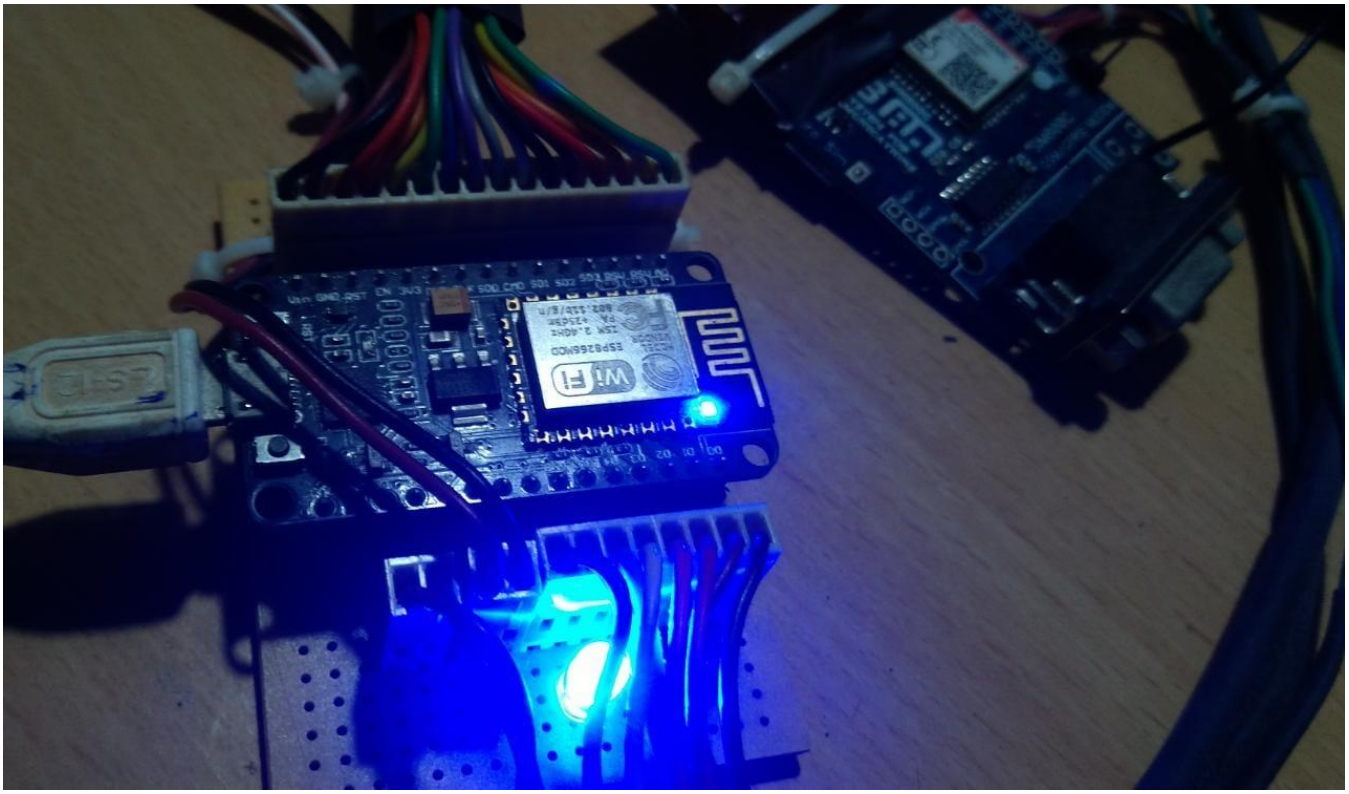

8:50
BLOOD REQUEST
BLOOD GROUPS
AB +
AB -
O+
O -
A +
A -
B +
B -
ALERT
You Have Requested : B-Positive
ACCEPT
Waiting For Responce from donor

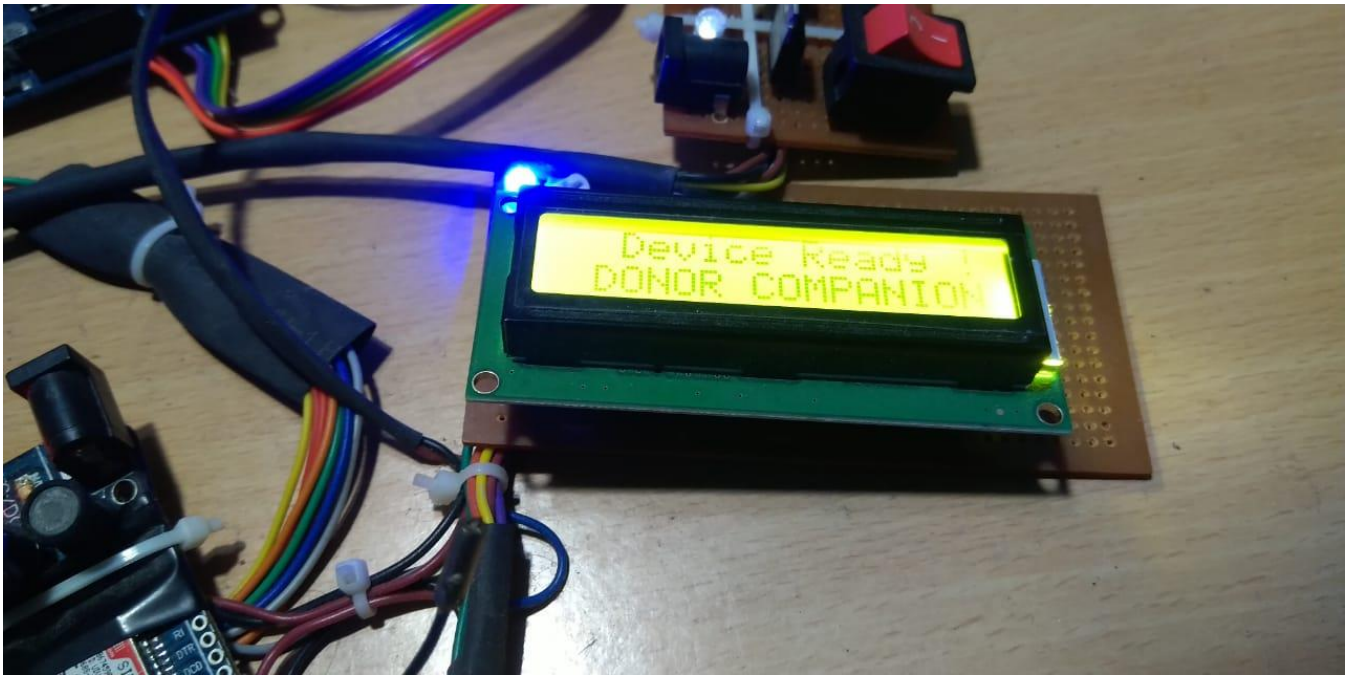
4.2.3 HARDWARE KIT OUTPUTS:

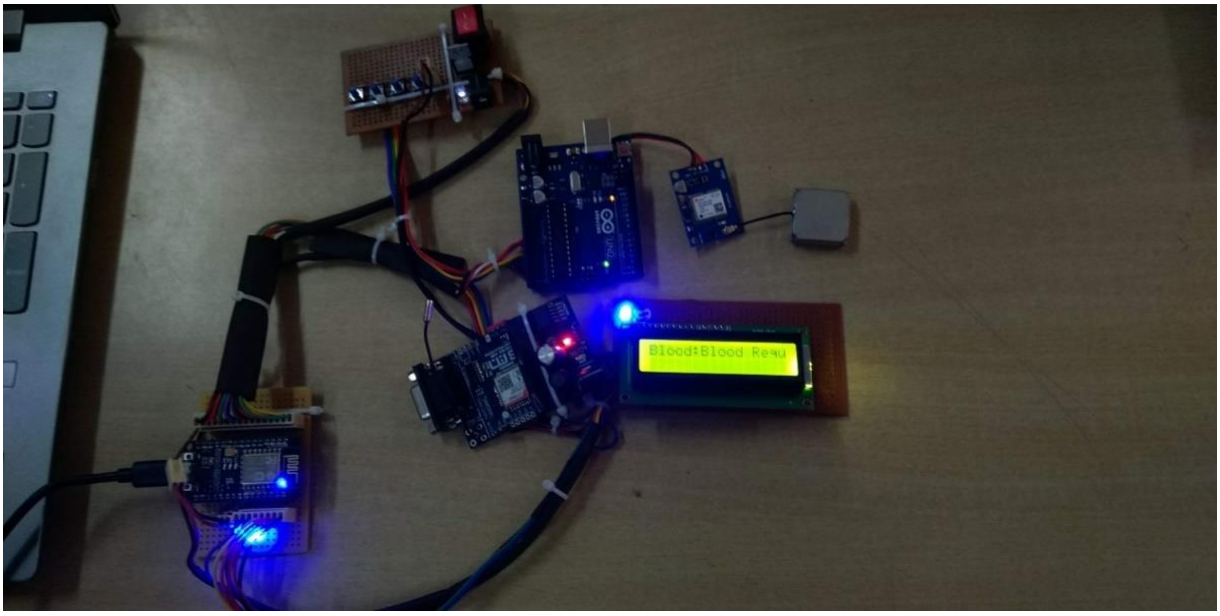


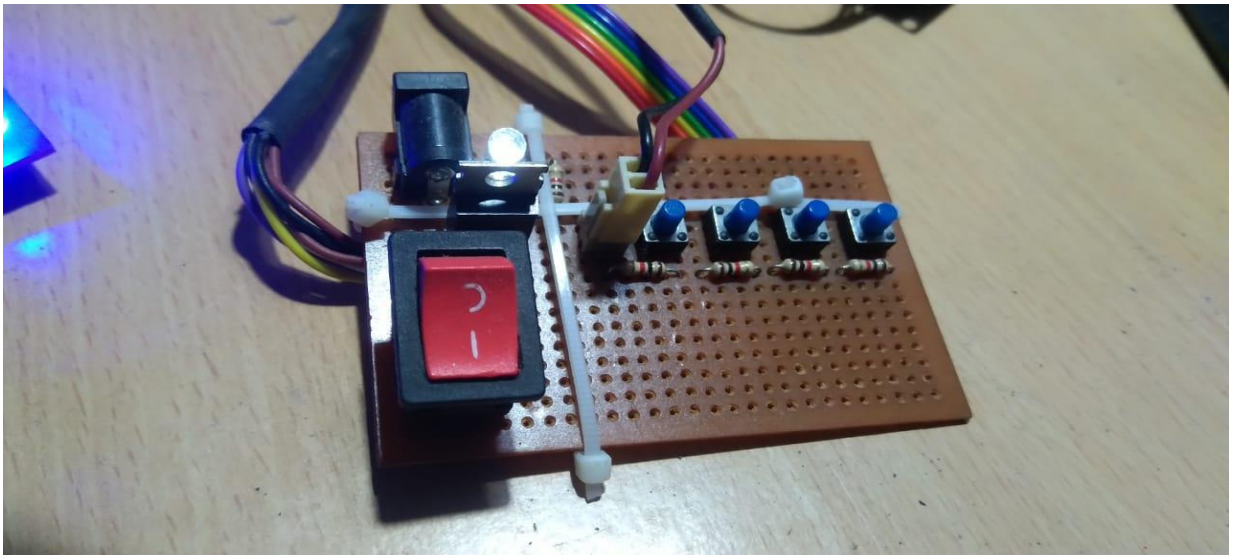







Device Ready
DONOR COMPANION


Blood:Blood Requ




Device Ready !
DONOR COMPANION


Device Ready !
DONOR COMPANION


Blood
B-POSITIVE


Contact
9698208798


Blood
B-NEGATIVE

REFERENCE:

[1] Prof . Snigdha et.al, " Android Blood Bank ",International Journal of Advanced Research in Computer and Communication Engineering , vol 4 , Issue 11,November 2015 , pp:86-88 , ISSN(online) 2278-1021,ISSN(print):2319 5940.

[2] Narendra Gupta et.al , "MBB:A Life Saving Application " , International Journal For Research in Emerging Science And Technology , vol 2 . Issue-1,March-2015 , pp:326330 , ISSN:2349- 7610 .

[3] Sultan Turhan , "An Android Application Volunteer Blood Donors " , ICBB-2015 , DOI:10.5121/csit .2015.51103,pp:23-30

[4] Sayali Dhond et.al , "Android Based Health Application in Cloud Computin For Blood Bank " ,International Engineering Research Journal(IERJ) vol 1 , Issue 9, 2015 , pp:868-870 , ISSN 2395- 1621.

[5] P.Priya et.al , "The Optimization of Blood Donor Information and Management System by Technopedia " , International Journal of Innovative Research in Science Engineering Technology , vol 3, Issue 1,February 2014 ,pp:390-395, ISSN(online):2319-8753, ISSN(print): 2347-6710.

[6] R.Vanitha and P.Divyarani ," BCloud App: Blood Donor Application For Android Mobile", International Journal Of Innovations in Engineering and Technology(IJIET) ,vol.2 ,Issue 1,February 2013, pp:396-401 , ISSN:2319-1058

CONCLUSION:

- Conventional methods used to determine the blood group involves risk of infection, time consuming and reagents are required.
- Alternate method for the Conventional method is Noninvasive method used to detect the blood type that cannot cause any infection, not required any reagent, time consuming
- In future upgrade of project, we can also include to detect sugar level, BP range with this device for domestic health care purpose.
- Adding tablet monitoring system with this system also make more reliable in future.