RFID based Health Care System – Database Management

Adit Bhardwaj – 10002004 Akash Bapat – 10003008 Nishank Jain – 10002016

ABSTRACT

Radio-frequency Identification and Detection (RFID) is the use of a wireless non-contact system that uses radio-frequency electromagnetic fields to transfer data from a tag attached to an object, for the purposes of automatic identification and tracking. The report is focused on the use of RFID technology to map a person's medical history and provide methods to add to the database and edit it. The information can be stored in the cloud or in the tag itself and can be accessed whenever and wherever it's required providing a clean and green way to get medicated even in the remotest of areas with no access to modern information technology.

In many countries, including India, a matter of great concern is public health and its adverse effects on daily life of people with the spread of incurable and deadly diseases. With the use of RFID technology, it is possible to access information about a person's well-being and current health status. Now, going in with the present trend of miniaturization, no one would like an additional piece of equipment to carry for any such purpose. Furthermore, the brand new concept of UID (Unique Identification Card) is a great way to implement this. We just need to attach a micro-sized equipment to the card which has to be carried by everyone and all of the above can be implemented very easily.

THE RFID TECHNOLOGY

A radio-frequency identification system uses *tags*, or *labels* attached to the objects to be identified. Two-way radio transmitter-receivers called *interrogators* or *readers* send a signal to the tag and read its response. The readers generally transmit their observations to a computer system running RFID software or RFID middleware.

The tag's information is stored electronically in a non-volatile memory. The RFID tag includes a small RF transmitter and receiver. An RFID reader transmits an encoded radio signal to interrogate the tag. The tag receives the message and responds with its identification information. This may be only a unique tag serial number, or may be product-related information such as a stock number, lot or batch number, production date, or other specific information.

RFID tags can be either *passive*, *active* or *battery assisted passive*.

- An active tag has an on-board battery and periodically transmits its ID signal.
- A battery assisted passive (BAP) has a small battery on board and is activated when in the presence of a RFID reader.
- A passive tag is cheaper and smaller because it has no battery. Instead, the tag uses the radio energy transmitted by the reader as its energy source. The interrogator must be close for RF field to be strong enough to transfer sufficient power to the tag.

Since tags have individual serial numbers, the RFID system design can discriminate several tags that might be within the range of the RFID reader and read them simultaneously.

Active and semi-passive tags are reserved for costly items that are read over greater distances. They broadcast high frequencies from 850 to 950 MHz that can be read 100 feet (30.5 meters) or more away. If it is necessary to read the tags from even farther away, additional batteries can boost a tag's range to over 300 feet (100 meters). Passive RFID tags are read up to 20 feet (six meters) away.

RFID tags contain at least two parts:

- An integrated circuit for storing and processing information, modulating and demodulating a radio-frequency (RF) signal, collecting DC power, from the incident reader signal, and other specialized functions.
- An antenna for receiving and transmitting the signal.

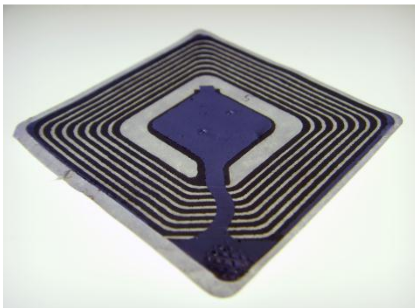
AREAS OF APPLICATIONS

This information can be used for multiple purposes:

1. To avoid contact with people with chronic diseases to prevent further spreading.
2. In case of an accident, instant information about blood group and other allergies etc. relevant for immediate treatment can be accessed.
3. A medical prescription carrying information about the disease can be directly used by a chemist in digital form thereby saving on paper.
4. At a higher technological level, when a person has an appointment with a doctor and enters the hospital, information can be read at the door itself and notified to the doctor which will increase the speed of consultation and save time as well as give opportunity to many others for treatment.
5. It is a great substitution for medical certificates required for many purposes.
6. The veterinarian doctors can also use the same technology to tag animals with their medical history and track them from time to time.

Apart from the health-care system, the technology can also be used for various other purposes such as:

1. It can also be used to automate the lighting and door-lock systems in houses and other places.
2. Access management
3. Tracking of goods
4. Tracking of persons and animals
5. Toll collection and contactless payment
6. Machine readable travel documents
7. Tracking sports memorabilia to verify authenticity
8. Airport baggage tracking logistics



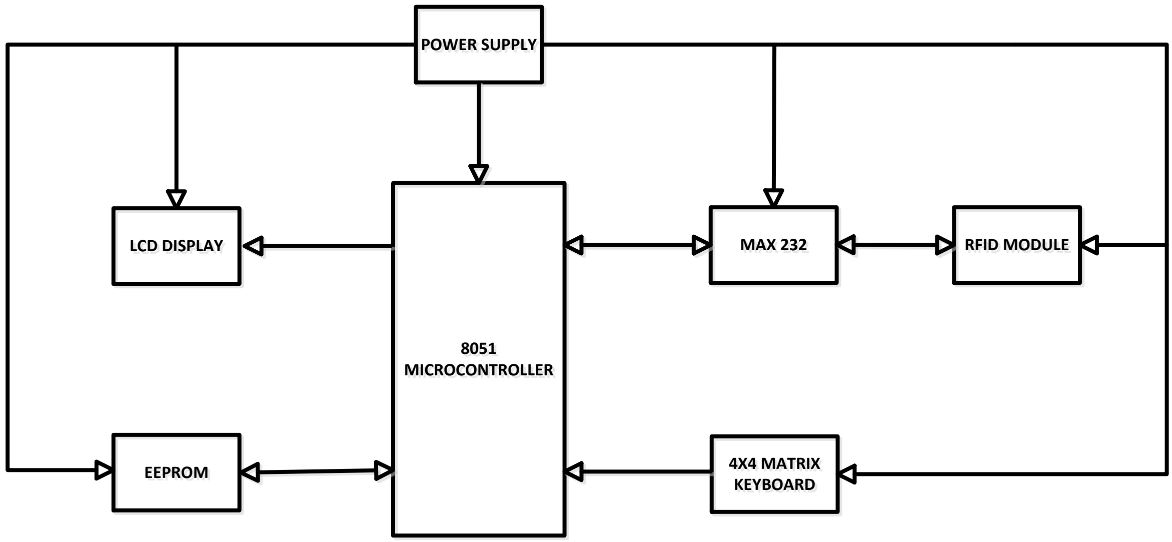
The link below gives an adequate description of the type of applications that can be achieved using RFID. This is an article from the RFID Journal which explains how the Texas Ophthalmology Clinics are planning to address wait times of the patients using the same technology.

http://www.rfidjournal.com/article/view/9966

PROJECT STATEMENT

To build an automated system using RFID technology to detect RFID tags, display the medical history of the corresponding person and enable the user to edit and update the information stored in the microcontroller memory.

PROJECT ANALYSIS

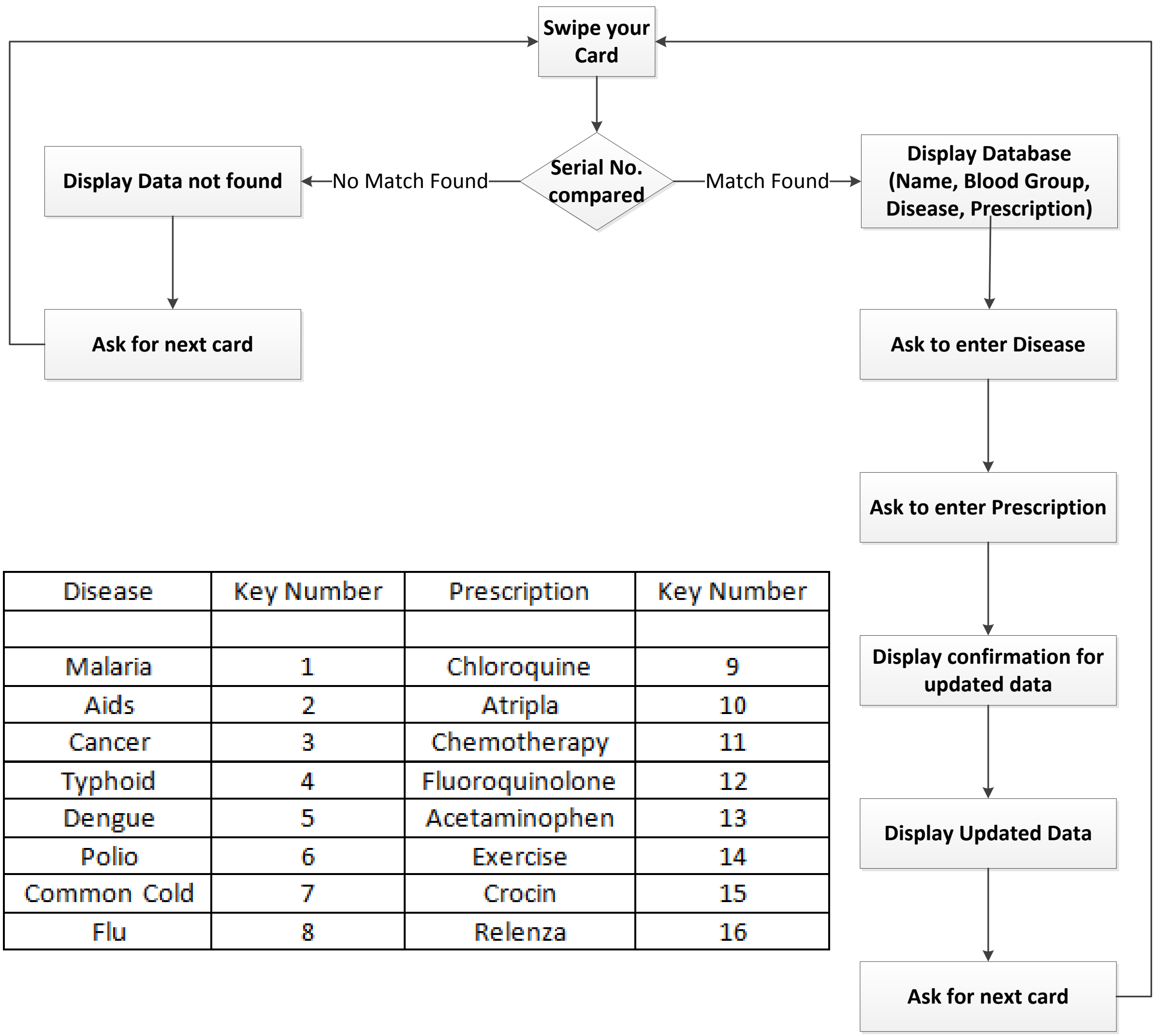


Different RFID tags work on different frequencies. Here low frequency, 125 kHz, RFID tags have been used. These tags work within a range of 10 cm. When an RFID tag comes in this range, the reader detects it and sends a unique code of the tag serially. This serial code, consisting of 12 bytes, is received by the microcontroller. A serial level converter is required for 89V51RD2 (8051 microcontroller IC) to receive these serial signals. IC MAX232 is used for this purpose to interface the RFID reader with microcontroller.

The 8051 converts the tag into a hex number and transmits the bits to the LCD. The LCD converts these data bits into ASCII characters and this serial number is compared with the previously stored serial numbers in the memory. If no match is found, the user is duly notified. If a match is found, then the medical history is displayed (if stored previously) and options to edit the information are also displayed on the LCD.

The user can make use of the 4X4 Matrix Keyboard to enter the disease and the prescription of the patient and update the information. If the information that has to be stored in the memory exceeds the memory limit of the microcontroller an external EEPROM can be interfaced with it to complete the job (not done here).

WORKFLOW OF THE SYSTEM



Disease	Key Number	Prescription	Key Number
Malaria	1	Chloroquine	9
Aids	2	Atripla	10
Cancer	3	Chemotherapy	11
Typhoid	4	Fluoroquinolone	12
Dengue	5	Acetaminophen	13
Polio	6	Exercise	14
Common Cold	7	Crocin	15
Flu	8	Relenza	16

The workflow of the system is quite fluid and user-friendly. Initially the LCD displays a message "Swipe your Card" asking you to bring your tag in close proximity to the reader. On detection of the tag, the reader beeps and transmits the received tag number to the microcontroller which compares it with the numbers already stored in its memory. If a match is not found, it displays "Data not found" and asks for the next card. If a match is found, however, the previous history (if any stored on the card) of the patient is displayed on the LCD. The data displayed consists of the name of the patient, blood group, disease and the prescription by the medical practitioner/doctor.

The user is then asked to edit the data by entering the disease and the prescription. If the user wants to change or add to the existing information, one can make use of the 4X4 matrix keyboard and press the keys 1→8 for entering diseases or 9→16 for entering the prescription by following the look-up table as given above.

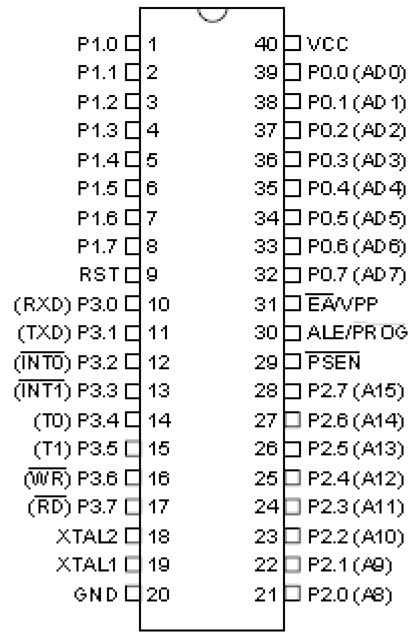
HARDWARE

The components used are:

1. 89V51RD2 (8051 Microcontroller IC) on a development board.
2. LCD Display
3. 4X4 Matrix Keyboard
4. Passive RFID Tags
5. RFID Module (with MAX 232 IC installed)

The other hardware needed to program the microcontroller is: Adapter (9V DC), Serial Communication Cable, Connecting Wires, Breadboard (if needed).

1. 8051 Micro-controller IC



The IC that has been employed is 89V51RD2. It consists of 4 I/O ports of which three have been used in this project (P0, P1, P2). P3 cannot be used as it has dedicated pins for different purposes so it might give erroneous results. P0 has been configured as data-port while P1 has been configured as command-port for the LCD. P2 has been configured for interfacing with the 4X4 matrix keyboard.

2. LCD Display

A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers:

1. ***Command/Instruction Register -*** stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing, clearing the screen, setting the cursor position, controlling display etc.

2. ***Data Register -*** stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.

The commands required to initialize the LCD are given below:

Hex Code	Command to LCD Instruction Register
1	Clear screen display
2	Return home
4	Decrement cursor
6	Increment cursor
E	Display ON, Cursor ON
80	Force the cursor to the beginning of the 1^{st} line
C0	Force the cursor to the beginning of the 2^{nd} line
38	Use 2 lines and 5x7 matrix

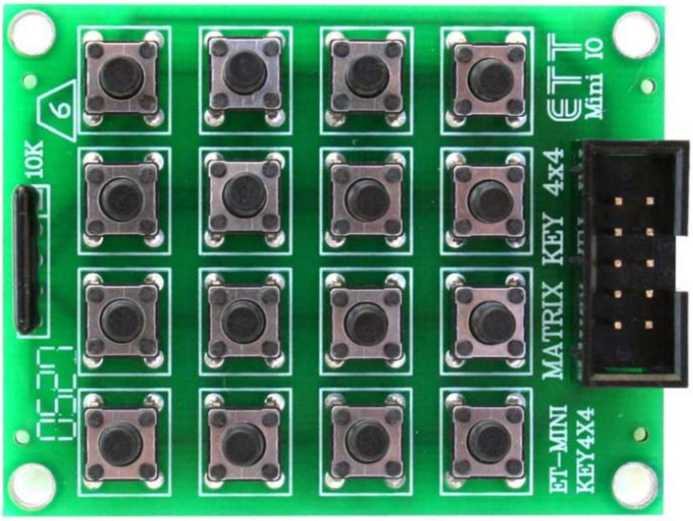
The pin description of this LCD module is given below:

<table>
<tr><th>Pin</th><th>Symbol</th><th>Description</th><th>Comments</th></tr>
<tr><td>1</td><td>V_{SS}</td><td>Ground</td><td>0V</td></tr>
<tr><td>2</td><td>V_{CC}</td><td>Main power supply</td><td>+5V</td></tr>
<tr><td>3</td><td>V_{EE}</td><td>Power supply to control contrast</td><td>Contrast adjustment by providing a variable resistor through V_{CC}</td></tr>
<tr><td>4</td><td>RS</td><td>Register Select</td><td>RS=0 to select Command Register
RS=1 to select Data Register</td></tr>
<tr><td>5</td><td>R/W</td><td>Read/Write</td><td>R/W=0 to write to the register
R/W=1 to read from the register</td></tr>
<tr><td>6</td><td>EN</td><td>Enable</td><td>A high to low pulse (min 450ns wide) is given when data is sent to data pins</td></tr>
<tr><td>7</td><td>DB0</td><td rowspan="8">To display letters or numbers, their ASCII codes are sent to data pins (with RS=1). Also <u>instruction command codes</u> are sent to these pins.</td><td rowspan="8">8-bit Data pins</td></tr>
<tr><td>8</td><td>DB1</td></tr>
<tr><td>9</td><td>DB2</td></tr>
<tr><td>10</td><td>DB3</td></tr>
<tr><td>11</td><td>DB4</td></tr>
<tr><td>12</td><td>DB5</td></tr>
<tr><td>13</td><td>DB6</td></tr>
<tr><td>14</td><td>DB7</td></tr>
<tr><td>15</td><td>LED+</td><td>Backlight V_{CC}</td><td>+5V</td></tr>
<tr><td>16</td><td>LED-</td><td>Backlight Ground</td><td>0V</td></tr>
</table>

To send a command on the LCD, a particular command is first specified to the data pins with R/W = 0 (to specify the write operation) and RS = 0 (to select the command register). A high to low pulse is given at EN pin when data is sent.

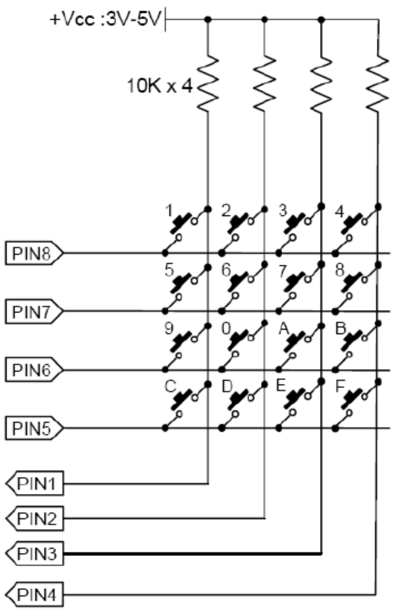
To send data on the LCD, data is first written to the data pins with R/W = 0 (to specify the write operation) and RS = 1 (to select the data register). A high to low pulse is given at EN pin when data is sent. Each write operation is performed on the positive edge of the Enable signal.

3. <u>4X4 Matrix Keyboard</u>



The 4x4 matrix keyboard module has 8 pins to interface with MCU (micro-controller). PIN1-PIN4 (COLUMN) is INPUT for MCU to read the status of the key being pressed. PIN5-PIN8 (ROW) is OUTPUT from MCU to send logic LOW to scan the key pressed on each row, MCU sends logic LOW into each key's ROW (PIN5-PIN8). The MCU then reads the status of the key pressed from PIN1-PIN4. If any key is pressed, the status reads logic LOW, otherwise the logic is HIGH.

The pin connections inside the keyboard are as shown below:



The first 8 keys are dedicated to provide input for entering the disease names and the next 8 keys are dedicated for entering the prescription in the program.

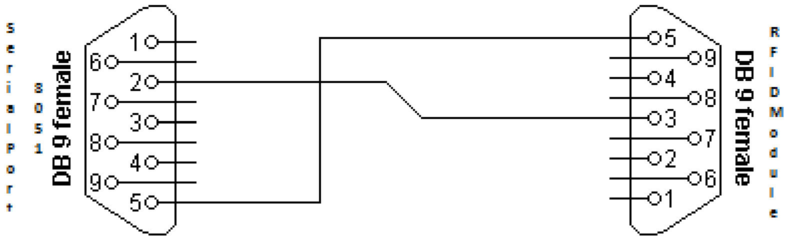
4. **Passive RFID Tags**

The Passive RFID tags have a short detection range (~10cm.) which exercises more control over the user because one has to be in close proximity to the reader for registration.

5. **RFID Module**

The RFID reader as shown below is fixed onto a module which contains the MAX 232 IC and a serial port.

The communication between the RFID Module and the Serial Port of the 8051 Microcontroller is achieved through a connection between the DB9 female connectors as shown below.



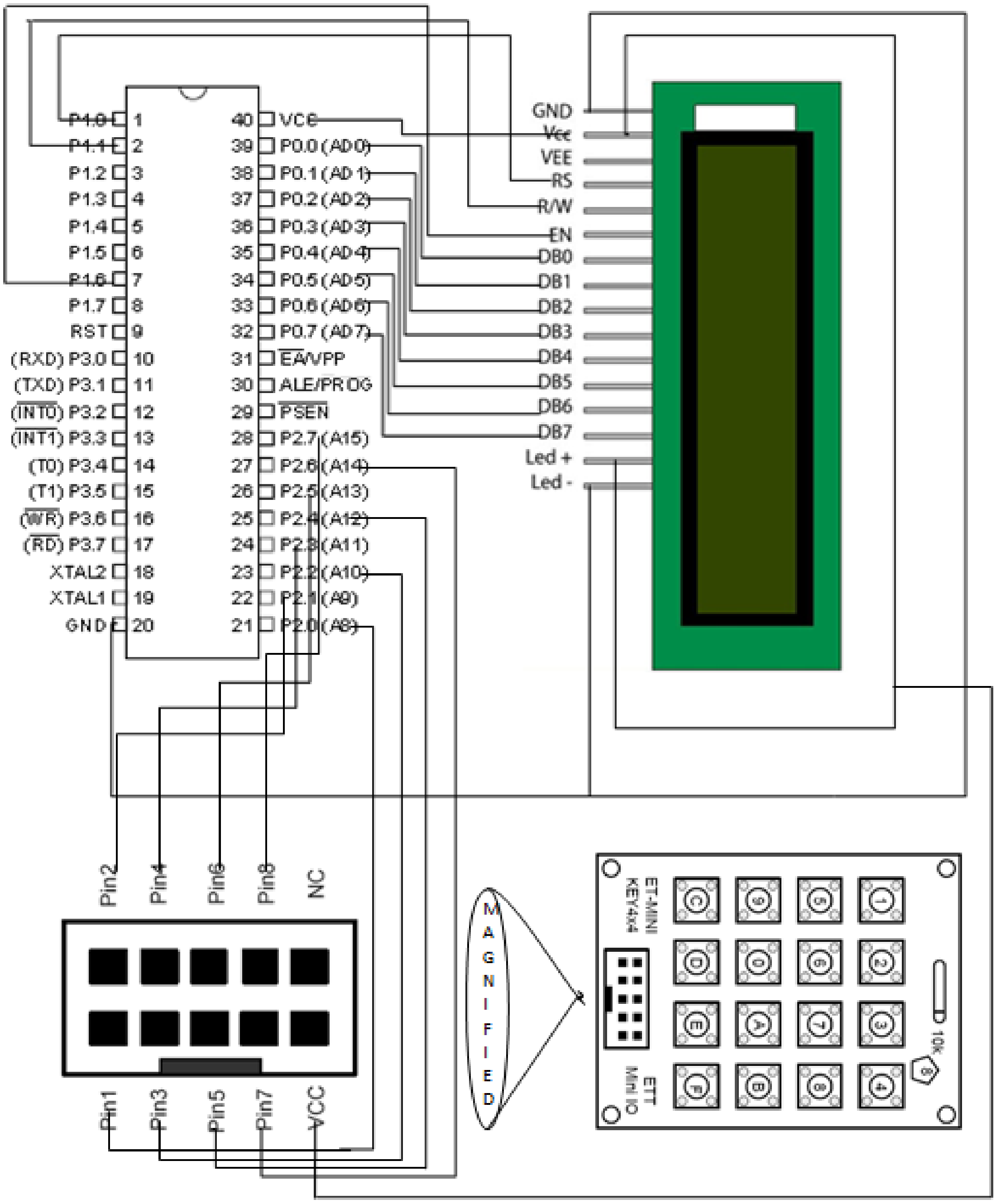
SOFTWARE

Two software are used to carry out the whole operation: Keil µVision3 and Flash Magic.

1. **Keil µVision3** – This software is used to write, assemble and build programs in both assembly language and C language. The program used in this project is coded in C language. On building the project, a hex file is generated which has to be burnt into the microcontroller which understands only binary instructions. This is done by the next software.
2. **Flash Magic** – This software is used to burn the hex file, generated previously, into the microcontroller memory. The procedure for burning the code is described below.

PROCEDURE TO GET THE SYSTEM RUNNING

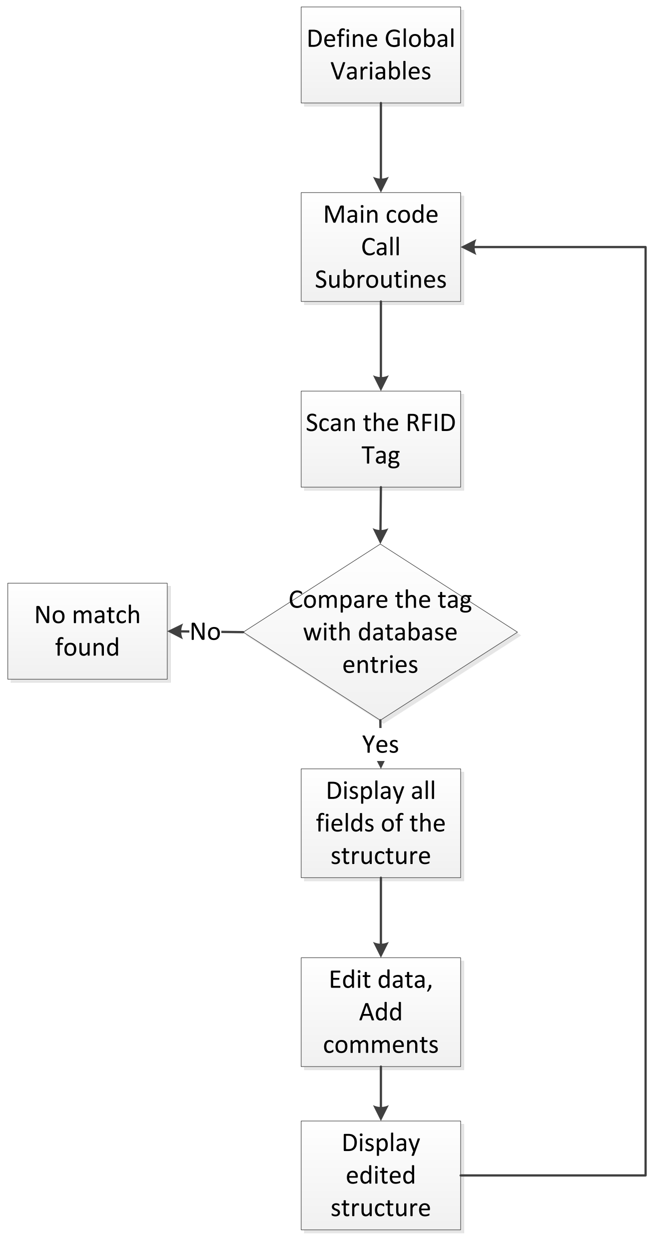
The schematic diagram of the hardware connections is given below:



1. Establish connections between the 8051 microcontroller on the development board, LCD Display and the 4X4 Matrix Keyboard as shown above.
2. Connect power supply to the development board.
3. Check whether the LCD is glowing, if not recheck the connections.
4. Connect the serial communication cable between the serial port of the microcontroller and the computer.
5. Compile and build the code to be burnt into the microcontroller memory and generate a hex file for the same remembering the path where it is being saved.

6. After all the connections are made and the hex file is generated, the steps to be followed for programming the microcontroller using Flash Magic are as follows:
    - Step – 1: Select Microcontroller “89V51RD2”, set COM PORT where you have connected the serial communication cable, set Baud Rate as “9600” and interface as “None (ISP)”.
    - Step – 2: Select Erase all Flash.
    - Step – 3: Select the path for the hex file of the code to be burnt.
    - Step – 4: If you wish to verify the burning after programming, you can select this option.
    - Step – 5: Start the process, if asked to reset the microcontroller press the reset button on the development board.
7. On completion of the process, disconnect the serial communication cable and press the reset button on the development board again.
8. The LCD should start displaying now.

WORKFLOW OF THE CODE



CODE

```
#include<reg51.h>
#include<string.h>
#define keypad P2                    //keypad connected to P2
sbit col1= P2^0;                     //column 1
sbit col2= P2^1;                     //column 2
sbit col3= P2^2;                     //column 3
sbit col4= P2^3;                     //column 4
sbit r1=P2^4;                        //row 1
sbit r2=P2^5;                        //row 2
sbit r3=P2^6;                        //row 3
sbit r4=P2^7;                        //row 4
#define cmdport P1                   //lcd initializers connected to P1
#define dataport P0                  //lcd data pins connected to P0
sbit rs = cmdport^0;                 //register select pin
sbit rw = cmdport^1;                 //read write pin
sbit e = cmdport^6;                  //enable pin


void scan(void);
unsigned char get_key();
void display(unsigned char);
void editRx(unsigned char ,unsigned char);
void editdis(unsigned char,unsigned char);
void disp(unsigned char);


struct Patient{
unsigned char name[8];
unsigned char bg[3];
unsigned char rfid[13];
unsigned char comm[4];
unsigned char Rx[4];
};


struct Patient p[3]                 //database of 3 patients - name, bloodgroup, rfid
                                      no.,disease, prescription

{
{{"ADIT"},{"B+"},{"42006C0DCBE8"},{"ILL"},{"NIL"}},
{{"AKASH"},{"O+"},{"42006C0A3410"},{"ILL"},{"NIL"}},
{{"NISHANK"},{"A+"},{"42006C4DB2D1"},{"ILL"},{"NIL"}}
};


unsigned char in[13];
unsigned char flag;
```

```
code unsigned char str[6][16]={{"SWIPE YOUR CARD"},
                              {"DATA NOT FOUND"},
                              {"ENTER DISEASE:"},
                              {"ENTER Rx:"},
                              {"DATA UPDATED.."},
                              {"NEXT PLZ..."}};


void scan()
{
unsigned char i;
TMOD=0x20;
TH1=0xFD;
SCON=0x50;
TR1=1;
for(i=0;i<12;i++)
{
while(RI==0);
in[i]= SBUF ;
RI=0;
}
TR1=0;
}


void delay(unsigned int msec)            //Function to provide time delay
{
int i,j ;
for(i=0;i<msec;i++)
for(j=0;j<1275;j++);
}


void lcdcmd(unsigned char item)          //Function to send command to LCD
{
dataport = item;
rs= 0;
rw=0;
e=1;
delay(1);
e=0;
}


void lcddata(unsigned char item)         //Function to send data to LCD
{
dataport = item;
rs= 1;
```

```
rw=0;
e=1;
delay(1);
e=0;
}


unsigned char check()
{
unsigned char j,a;
for(j=0;j<3;j++)
{
a=strcmp(p[j].rfid,in);                    //compares a string, if equal, a=0
if(a==0)
return(j);
}
return(3);
}


void display(unsigned char a)              //display struct
{
unsigned char j=0;
while(p[a].name[j]!='\0')                  //name
{
lcddata(p[a].name[j]);
delay(10);
j++;
}
lcdcmd(20);
j=0;

while(p[a].bg[j]!='\0')                    //bloodgroup
{
lcddata(p[a].bg[j]);
delay(10);
j++;
}
lcdcmd(0x0C0);                             //move to next line
j=0;
lcddata(68);                               //ascii nos dis:
lcddata(73);
lcddata(83);
lcddata(58);

while(p[a].comm[j]!='\0')                  //disease
{
lcddata(p[a].comm[j]);
```

```
delay(10);
j++;
}
j=0;

lcdcmd(20);                              //display space rx:
lcddata(82);
lcddata(88);
lcddata(58);

while(p[a].Rx[j]!='\0')                  //rx=prescription
{
lcddata(p[a].Rx[j]);
delay(10);
j++;
}
delay(800);
lcdcmd(20);
}


void editdis(unsigned char id,unsigned char disease)    //lookup options
{
switch(disease)                                          //for entering diseases
{
case 1:
strcpy(p[id].comm,"MAL");
break;
case 2:
strcpy(p[id].comm ,"AID");
break;
case 3:
strcpy(p[id].comm,"CAN");
break;
case 4:
strcpy(p[id].comm,"TYP");
break;
case 5:
strcpy(p[id].comm ,"DEN");
break;
case 6:
strcpy(p[id].comm ,"POL");
break;
case 7:
strcpy(p[id].comm,"CCD");
break;
case 8:
strcpy(p[id].comm,"FLU");
```

```
break;
default: break;
}
}


void editRx(unsigned char id,unsigned char pr)      //lookup options
{pr = pr-8;                                         //options on keyboard from 9 to 16
switch(pr)                                          //for entering prescription
{
case 1:
strcpy(p[id].Rx ,"CHL");
break;
case 2:
strcpy(p[id].Rx ,"ATR");
break;
case 3:
strcpy(p[id].Rx ,"CHE");
break;
case 4:
strcpy(p[id].Rx ,"FLO");
break;
case 5:
strcpy(p[id].Rx ,"ACT");
break;
case 6:
strcpy(p[id].Rx ,"EXE");
break;
case 7:
strcpy(p[id].Rx ,"CRO");
break;
case 8:
strcpy(p[id].Rx ,"REL");
break;
default: break;
}
}


unsigned char get_key()
{while(1)
{
unsigned char k,i,key;
r1=0;
r2=0;
r3=0;
r4=0;
key=0;
```

```
k=1;

for(i=0;i<4;i++)
{
keypad = ~(0x80>>i);
if(col1 == 0)                              //check if key1 is pressed
{
key = k+0;                                 //set key number
while(col1 == 0);                          //wait for release
return key;                                //return key number
}

if(col2 == 0)                              //check if key2 is pressed
{
key = k+1;                                 //set key number
while(col2 == 0);                          //wait for release
return key;                                //return key number
}

if(col3 == 0)                              //check if key3 is pressed
{
key = k+2;                                 //set key number
while(col3 ==0);                           //wait for release
return key;                                //return key number
}

if(col4==0)                                //check if key4 is pressed
{
key = k+3;                                 //set key number
while(col4 ==0);                           //wait for release
return key;                                //return key number
}

k+=4;                                      //next row key number
}

}
return 0;                                  //return false if no key pressed
}


void disp(unsigned char option)            //display fixed strings
{
unsigned char j=0;
delay(20);
lcdcmd(0x01);
while(str[option][j]!='\0')
{
```

```
lcddata(str[option][j]);
delay(10);
j++;
}
}


main()
{
unsigned char a=3,k;
unsigned char ret =0;                    //for get_key
in[12]=0x00;                             //for comparison as a string 0x00=\0 ascii code
P1=0x00;

while(1)
{
lcdcmd(0x38);                            //For using 8-bit 2 row mode of LCD
delay(10);
lcdcmd(0x0E);                            //Turn display ON for cursor blinking
delay(10);

lcdcmd(0x01);                            //clear screen
delay(10);
lcdcmd(0x06);                            //display ON

keypad=0x0F;                             //initialize keypad

disp(0);
scan();                                  //scan the rfid
a=check();                               //check database

if(a!=3)                                 //3 persons a=3 condition
{
lcdcmd(0x01);
display(a);                              //display struct
disp(2);
lcdcmd(0x0C0);                           //enter disease

ret=get_key();                           //which disease
editdis(a,ret);

for(k=0;k<3;k++)                         //display entered disease
{
lcddata(p[a].comm[k]);
delay(10);
}

delay(100);
```

```
disp(3);
lcdcmd(0x0C0);                              //enter Rx

ret=get_key();
editRx(a,ret);                              //select rx

for(k=0;k<3;k++)                            //display entered rx
{
lcddata(p[a].Rx[k]);
delay(10);
}

delay(100);
lcdcmd(0x01);                               //clear screen
disp(4);                                    //successfully updated
delay(100);
lcdcmd(0x01);                               //clear screen
display(a);                                 //updated data
}

else disp(1);                               //if person not found
delay(50);
disp(5);                                    //displays no match
delay(100);
}
}
```

DEMO


SWIPE YOUR CARD


ENTER Rx:


DATA NOT FOUND


ENTER Rx:
VTS


NEXT PLZ...


DATA UPDATED..


SWIPE YOUR CARD


AKASH O+
DIS:CAN RX:VTS


AKASH O+
DIS:CAN RX:VTS


NEXT PLZ...


ENTER DISEASE:


SWIPE YOUR CARD


ENTER DISEASE:
CAN

RESULTS

The aim of the project is achieved. The RFID tag is detected and if match is found the LCD is able to display the name of the patient, blood group, medical history and provides the ability to edit it whereas the user is notified if the match is not found. The user can choose from 8 different diseases and 8 corresponding prescriptions to enter into the record and update the information about the patient for future reference. The system is robust but is limited in storage capabilities. Also, since in the given microcontroller the data is stored in the RAM, it is lost once the IC is reset. So an external ROM needs to be connected for permanent data storage.

FURTHER EXPANSION

RFID technology has been around since 1970, but until recently, it has been too expensive to use on a large scale. Originally, RFID tags were used to track large items, like cows, railroad cars and airline luggage, that were shipped over long distances. This database management system can be extended to store the information in the cloud and display it on a computer. A full-fledged keyboard and an external memory can also be interfaced so that more amount of data can be stored and more flexibility in editing the data can be achieved. The information can also be used for accessing restricted areas and many other purposes as mentioned in the areas of applications.