**HOME AUTOMATION SYSTEM USING RASPBERRY PI**

**SEMINAR / PROJECT REPORT**

### **Submitted by**

Deepak J Puthukkaden ,20

Giridhar A K ,31

Govindh B ,32

Rohit Sreekumar ,53

Shine Ali ,59

***in partial fulfillment for the award of the degree***

***of***

**Bachelor of Technology**

in

ELECTRONICS AND COMMUNICATION ENGINEERING

*of*

**COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY**

****

**DEPARTMENT OF ELECTRONICS ENGINEERING**

**MODEL ENGINEERING COLLEGE**

**COCHIN 682 021**

**NOVEMBER 2014**

****

**MODEL ENGINEERING COLLEGE**

**THRIKKAKARA, KOCHI-21**

**DEPARTMENT OF ELECTRONICS ENGINEERING**

**COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY**

***BONAFIDE CERTIFICATE***

***This is to certify that the seminar report entitled***

**. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**. . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . .**

***Submitted by***

**. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

***is a bonafide account of the work done by him/her under our supervision***

***Dr. Mini M G Ms. Jibi John Mr./Ms.***

***Head of department Project coordinator Project guide***

**ACKNOWLEDGEMENT**

At this moment of accomplishment, we are presenting our work with

great pride and pleasure, we would like to express our sincere gratitude to all

those who helped us in the successful completion of our venture. First of all,

we would like to thank our Principal **Prof. V.P Devassia** who provided us

with all facilities and amenities for the development of our project. We would like to thank our HOD, **Dr Mini M G** for helping us in the successful

accomplishment of our project. We are exceedingly grateful to our project

coordinator Assistant Professor, **Ms. Jibi John** for her timely and valuable

suggestions. We would also like to thank our project guide Senior Lecturer

**Mr.Rashid** who gave us constant guidance and support throughout this

journey of turbulence.

We also sincerely thank **Mr.Sujith**, Lab Technician Department of Electronics and Communication for their constant support and encouragement for our project. We would also like to thank our parents and friends for their over-whelming and whole hearted encouragement and support without which this would not have been successful.

Above all we thank God almighty for constantly motivating us with His love, and giving courage to move forward with confidence and self –belief.

**ABSTRACT**

*Home automation is one of the most researched topics in the present technological scenario. It is basically an application of* ***Internet of Things*** *or* ***IoT****. Internet of things is an interconnection of all the appliances and devices that we use in our everyday life, it may be coffee vending machines, smart phones, medical monitors to huge weather stations. In the approach of internet of things, all devices work in a synchronised manner, i.e they are interdependent.*

*This system consists of a main processing unit(Raspberry Pi), microcontroller to interface the different modules, and the independent modules.In our project the area which we are going to concentrate is home automation that works based on* ***real time data****. It does the following functions:*

* *Adaptive internal lighting and automatic external lighting system*
* *Automatic water pumping system*
* *Password enabled door lock using keypad*
* *Interactive Music player*
* *Logging of all the control events*

**Table of Contents**

Chapter1 Project Overview

1.1 Introduction

1.2 Objectives

1.3 System Specifications

Chapter2 System Overview

2.1 Modules

2.1.1 Raspberry Pi

2.1.2 Microcontroller Atmega328P

2.1.3 Passive InfraRed (PIR) Sensor

2.1.4 Real-Time Clock (RTC-DS1307) module

2.2 Systems

2.2.1 Adaptive Internal Lighting system

2.2.2 Automatic External Lighting system

2.2.3 Automatic Water Pump system

2.2.4 Password Enabled Door lock system

2.2.5 Interactive Music System

2.2.6 Data logging system

2.3 Block Diagram

Chapter3 System Design

3.1 Adaptive Internal Lighting system

3.2 Automatic External Lighting system

3.3 Automatic Water Pump system

3.4 Password Enabled Door lock system

3.5 Interactive Music System

3.6 Power Supply

3.7 Real Time Clock (RTC-DS1307)

**Chapter 1**

**PROJECT OVERVIEW**

**1. 1 Introduction**

The proposed idea for our Mini project for the Fifth Semester (2014-2015)

is Home Automation based on Raspberry Pi. Automation is a broad

concept in engineering and home automation is but a small subsidiary of this vast idea. Recently, the Internet Of Things has gathered momentum and many interesting applications have surfaced making many of the daily chores simple and efficient. We intend to implement a small scale home automation project using the Raspberry Pi. Raspberry Pi is a powerful credit card sized computer capable of running as a normal computer and also capable of running the functions of a microcontroller.

Due to the hardware limitations (limited no of input and output pins) of the

Raspberry Pi, we are forced to interface additional microcontrollers to

control the various sensors involved in the automation.

**1. 2 Objectives**

The system comprises automation of some daily tasks in our home. The

different functions of the system are:

• Adaptive Internal Lighting and automatic external lighting System

• Automatic Water Pumping System

• Password Enabled Door Lock using keypad

• Interactive Music Player

• Logging of all the control events

**1.3 System Specifications**

* Raspberry Pi
* Atmega 328P (2)
* Serial Interface
* Power supply 5V, 2A
* Passive InfraRed (PIR)
* Real Time Clock (RTC)

**Chapter 2**

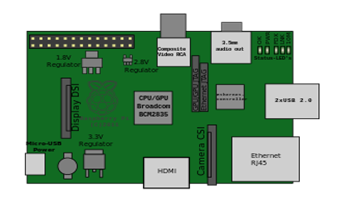
**SYSTEM OVERVIEW**

**2.1 Modules**

We use 4 individual modules in our project namely Raspberry Pi, Microcontroller (Atmega 328P), PIR sensor and an RTC (DS1307).

**2.1.1 Raspberry Pi**

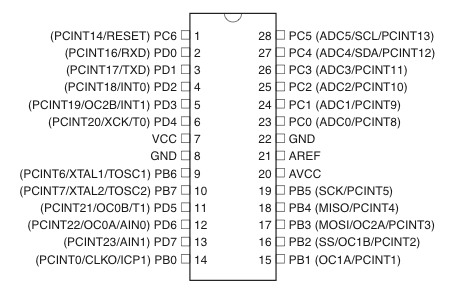
Raspberry Pi is a single board computer. It has a Broadcom BCM2835 system on a chip (SoC), which includes an ARM1176JZF-S 700 MHz processor, VideoCore IV GPU with 256MB of RAM. But it does not come with a Real-Time Clock, though it can be interfaced with DS1307 through I2C.

****Figure 1

**2.1.2 Microcontroller Atmega328P**

The high-performance Atmel picoPower 8-bit AVR RISC-based microcontroller combines 32KB ISP flash memory with read-while-write capabilities, 1024B EEPROM, 2KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, a 6-channel 10-bit A/D converter (8-channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts.

By executing powerful instructions in a single clock cycle, the device achieves throughputs approaching 1 MIPS per MHz, balancing power consumption and processing speed.

Figure 2

**2.1.3 Passive Infrared (PIR) sensor**

A passive infrared sensor (PIR sensor) is an electronic sensor that measures infrared (IR) light radiating from objects in its field of view. An individual PIR sensor detects changes in the amount of infrared radiation impinging upon it, which varies depending on the temperature and surface characteristics of the objects in front of the sensor. When an object, such as a human, passes in front of the background, such as a wall, the temperature at that point in the sensor's field of view will rise from room temperature to body temperature, and then back again. The sensor converts the resulting change in the incoming infrared radiation into a change in the output voltage, and this triggers the detection. Moving objects of similar temperature to the background but different surface characteristics may also have a different infrared emission pattern, and thus sometimes trigger the detector.



Figure 3

**2.1.4 Real Time Clock (RTC)**

A real-time clock (RTC) is a computer clock that keeps track of the current time.

Main advantages are:-

1. Low Power Consumption

2. Frees the main system from critical time related tasks

3. More accurate than other methods



Figure 4

**2.2 Systems**

There are 6 systems or individual modules in our project. They are:

**2.2.1 Adaptive Internal Lighting System**

The Adaptive Internal Lighting System controls the amount of illumination in a room by controlling a light source, based on the amount of light present in the room. The sensor involved is a light dependant resistor (LDR) which varies it's resistance depending on the amount of light falling on it. The change in resistance will result in a change in the voltage drop across the resistance (a varying analog signal). For demonstration purposes we intend to use an LED, the brightness of which is controlled by Pulse Width Modulation.

**2.2.2 Automatic External Lighting System**

This feature of our project controls an External Lighting System based on

the amount of external light. When the light outside falls below a

specific value, the light is turned on. It also makes use of a LDR.

**2.2.3 Automatic Water Pumping System**

Through this system we intend to control the water pump based on

the water level inside the water tank. This prevents the overflowing and

wastage of water which is a pristine and scarce resource. The water level is

determined by two sensors and the output is used by Raspberry Pi to

drive a relay that switches the water pumping system.

**2.2.4 Password Enabled Door Lock System**

This feature adds to the security of the smart home. A 4\*3 keypad is interfaced

with the microcontroller which is then connected to a door lock system. The

door lock gets unlocked when the correct password is entered.

**2.2.5 Interactive Music System**

This part of the project adds to the entertainment in the smart home.

A simple music player is integrated with the Raspberry Pi which can be

controlled using small buttons for functions like play, pause etc.

**2.2.6 Data logging**

This feature comprises of the data logging and event tracking feature of

our project. Data logging includes the temperature measurement and event

logging. It can be used to analyze the working of the home.

**2.3 Block Diagram**

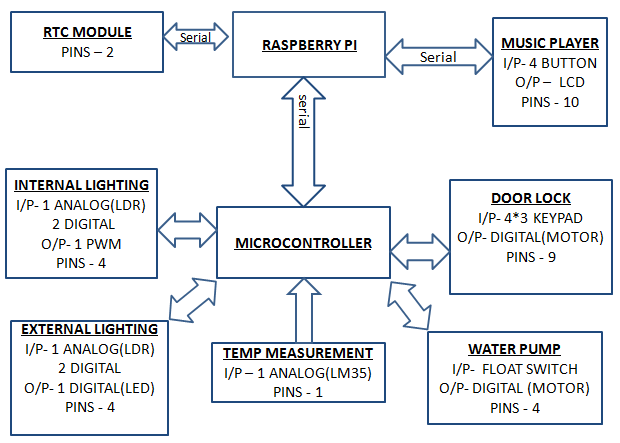
****

Figure 5

**Chapter 3**

**System Design**

**3.1 Adaptive Lighting System**

Hardware Requirements:-

* LDR
* LED
* Power Supply-5V

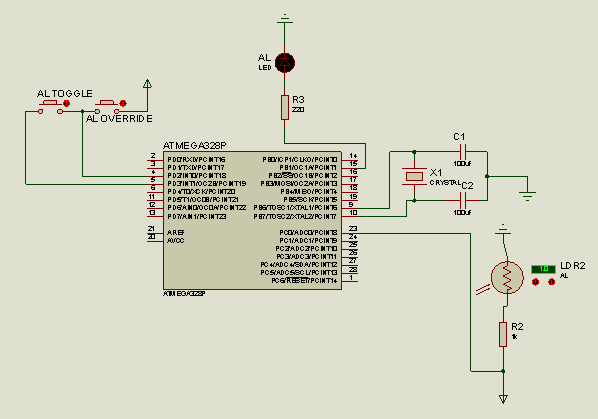
****

Figure 6

**3.2 Automatic External Lighting System**

Hardware Requirements:-

* LDR
* LED
* Power Supply-5V

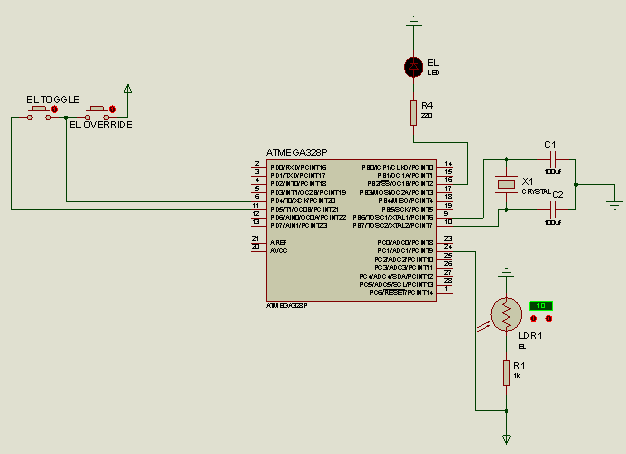
****

Figure 7

**3.3 Automatic Water Pump System**

Hardware Requirements:-

* Relay
* L293D
* Power Supply-5V

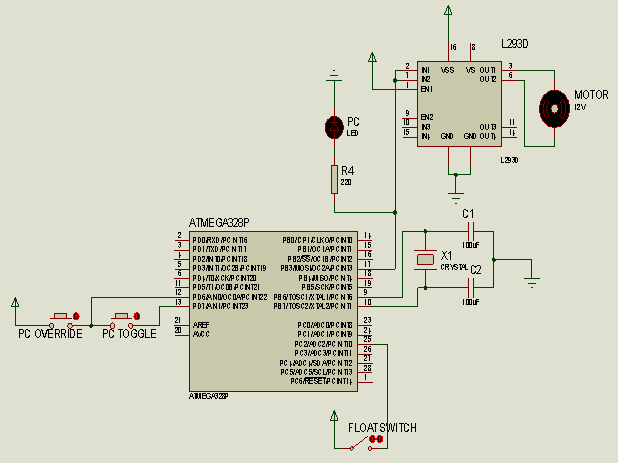


Figure 8

**3.4 Password Enabled Door Lock**

Hardware Requirements:-

* Relay
* L293D
* Power Supply-5V

**3.5 Interactive Music Player System**

Specifications:

**3.6 Power Supply**

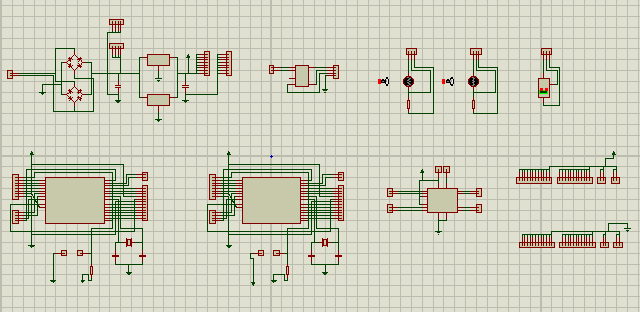
Specifications:

* 5V, 2A

**3.7 Real Time Clock (RTC)**

**Chapter 4**

**PCB Design**

****Figure 9

**Chapter 5**

**Software**

/\*

Main Program

==========

adaptive lighting

al-adaptive lighting

external lighting

el-external lighting

temperature mesuring system

tm-temperature measurement

Atmega1:

PIR:

i/p - A0

\*/

#include <EEPROM.h>

//preprosessor definitions

#define pirinput 9

#define alinput A1

#define aloutput 10

#define aloverride 12

#define altoggle 13

#define althresholdlow 500

#define althresholdmedium 650

#define althresholdhigh 900

#define aleeprom 1

#define elinput A2

#define eloutput 7

#define threshold 500

#define eloverride 5

#define eltoggle 6

#define eleeprom 2

#define tminput A3

//variable defenitions

int serial\_input;

int pirstate = LOW; // we start, assuming no motion detected

int pirvalue = 0; // variable for reading the pin status

int pir\_status=0;

int alstate,alsensor;

int elstate,elsensor;

int tm;

//functions

int pir();

void adaptive\_internal\_lighting\_system(); //adaptive internal lighting system function

void external\_lighting\_system(); //external lighting system function

void temperature\_measurement(); //temperature measurement function

void alstatus();

void elstatus();

//setup function

void setup()

{

pinMode(pirinput, INPUT); // declare sensor as input

pinMode(aloverride,INPUT);

pinMode(altoggle,INPUT);

alstate=EEPROM.read(aleeprom);

pinMode(eloutput,OUTPUT);

pinMode(eloverride,INPUT);

pinMode(eltoggle,INPUT);

elstate=EEPROM.read(eleeprom);

delay(1000);

Serial.begin(9600);

delay(1000);

}

//cloop function

void loop()

{

adaptive\_internal\_lighting\_system(); //call the adaptive internal lighting system function

external\_lighting\_system(); //call the external lighting system function

temperature\_measurement();//call the temperature measurement function

delay(100);

}

void serialEvent()

{

if(Serial.available()==1)

{ serial\_input=Serial.read();

if(serial\_input==20)

alstatus();

else if(serial\_input==30)

elstatus();

else if(serial\_input==60)

temperature\_measurement();

}

}

int pir()

{

pirvalue = digitalRead(pirinput); // read input value

if (pirvalue == HIGH)

{

// check if the input is HIGH

if (pirstate == LOW)

{

// we have just turned on

// We only want to print on the output change, not state

pirstate = HIGH;

return pirstate;

}

}

else

{

if (pirstate == HIGH)

{

// we have just turned of

pirstate = LOW;

return pirstate;

}

}

}

//adaptive internal lighting system function

void adaptive\_internal\_lighting\_system()

{

pir\_status=pir();

if(pir\_status==0&&alstate!=5)

{ Serial.print(25);

alstate=5;

EEPROM.write(aleeprom,alstate);

}

if(pir\_status==0&&alstate==5)

return;

if(!digitalRead(aloverride))

{ alsensor = analogRead(alinput); //read the sensor value

if(alsensor>=althresholdlow)

{ analogWrite(aloutput,20); //output the PWM to the LED

if(alsensor>=althresholdmedium&&alsensor<althresholdhigh)

analogWrite(aloutput,127);

else if(alsensor>=althresholdhigh)

analogWrite(aloutput,255);

if(alstate!=2)

{

Serial.print(22);

alstate=2;

EEPROM.write(aleeprom,alstate);

}

}

else

{ if(alstate!=1)

{ analogWrite(aloutput,0);

alstate=1;

Serial.print(21);

EEPROM.write(aleeprom,alstate);

}

}

}

else //override function

{ if(digitalRead(altoggle)) //toggle - ON

{ analogWrite(aloutput,255);

if(alstate!=4)

{ alstate=4;

Serial.print(24);

EEPROM.write(aleeprom,alstate);

}

}

else //toggle - OFF

{ analogWrite(aloutput,0);

if(alstate!=3)

{ alstate=3;

Serial.print(23);

EEPROM.write(aleeprom,alstate);

}

}

}

return;

}

void alstatus()

{

if(alstate==1)

Serial.print(21);

else if(alstate==2)

Serial.print(22);

else if(alstate==3)

Serial.print(23);

else if(alstate==4)

Serial.print(24);

else

Serial.print(25);

return;

}

//external lighting system function

void external\_lighting\_system()

{

if(!digitalRead(eloverride))

{ elsensor=analogRead(elinput);

if(elsensor>threshold)

{ digitalWrite(eloutput,HIGH);

if(elstate!=1)

{ Serial.print(32);

elstate=1;

EEPROM.write(eleeprom,elstate);

}

}

else if(elsensor<threshold)

{ digitalWrite(eloutput,LOW);

{ if(elstate!=0)

{ Serial.print(31);

elstate=0;

EEPROM.write(eleeprom,elstate);

}

}

}

}

else //override function

{ if(digitalRead(eltoggle))

{ digitalWrite(eloutput,HIGH);

if(elstate!=3)

{ Serial.print(34);

elstate=3;

EEPROM.write(eleeprom,elstate);

}

}

else

{ digitalWrite(eloutput,LOW);

if(elstate!=2)

{ Serial.print(33);

elstate=2;

EEPROM.write(eleeprom,elstate);

}

}

}

return;

}

void elstatus()

{

if(elstate==1)

Serial.print(31);

else if(elstate==2)

Serial.print(32);

else if(elstate==3)

Serial.print(33);

else

Serial.print(34);

return;

}

//temperature measurement system

void temperature\_measurement()

{

tm=analogRead(tminput);//read the temperature value

//tmvoltage = tm \* (5 / 1024);

Serial.print(61);

Serial.print(tm);

return;

}

/\*

Atmega 2

======= =

water pump system

pc-pumpcontrol

\*/

//preprosessor definitions

#include <Keypad.h>

#include<String.h>

#include<EEPROM.h>

#define pcinput A2

#define pcoutput 11

#define pcoverride 6

#define pctoggle 7

#define pcinput11 A0

#define pcinput21 A1

#define pcinput22 A2

#define pcoutput A3

#define pcled A4

#define pcoverride A5

#define pctoggle 13

#define pceeprom 6

const byte ROWS = 4; //four rows

const byte COLS = 3; //three columns

char keys[ROWS][COLS] = {

{'1','2','3'},

{'4','5','6'},

{'7','8','9'},

{'\*','0','#'}

};

char t;

byte rowPins[ROWS] = {5, 4, 3, 2}; //connect to the row pinouts of the keypad

byte colPins[COLS] = {8, 7, 6}; //connect to the column pinouts of the keypad

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

byte ledPin = 13;

byte motorpin1=12;

byte motorpin2=11;

boolean blink = false;

boolean ledPin\_state;

int addr=0,i,c[5];

char b[5],key;

//variable defenitions

int pcstate;

int serial\_input;

//functions

void pump\_control\_system(); //automatic pump control system function

void KEYPAD();

void motorup();

void motordown();

void keypadEvent(KeypadEvent key);

//setup function

void setup()

{

pinMode(pcinput11,INPUT);

pinMode(pcinput21,INPUT);

pinMode(pcinput22,INPUT);

pinMode(pcoverride,INPUT);

pinMode(pctoggle,INPUT);

pinMode(pcoutput,OUTPUT);

pinMode(pcled,OUTPUT);

pcstate=EEPROM.read(pceeprom);

pinMode(ledPin, OUTPUT); // Sets the digital pin as output.

digitalWrite(ledPin, HIGH); // Turn the LED on.

ledPin\_state = digitalRead(ledPin); // Store initial LED state. HIGH when LED is on.

keypad.addEventListener(keypadEvent); // Add an event listener for this keypad

for(i=0,addr=0;i<4;i++,addr++)

{

c[i]=EEPROM.read(addr);

b[i]=c[i]+48;

}

addr=0;

delay(1000);

Serial.begin(9600);

delay(1000);

}

//loop function

void loop()

{ pump\_control\_system(); //call the automatic pump control system function

KEYPAD();

delay(1);

}

void serialEvent()

{

if(Serial.available()==1)

{ serial\_input=Serial.read();

if(serial\_input==40)

pcstatus();

}

}

//pump control system

void pump\_control\_system()

{

if(digitalRead(pcinput11==1)&&pcstate!=5)

{ Serial.print(45);

pcstate=5;

EEPROM.write(pceeprom,pcstate);

digitalWrite(pcled,HIGH);

delay(500);

digitalWrite(pcled,LOW);

}

if(pcstate==5)

return;

pcstate=1;

if(!digitalRead(pcoverride))

{ int a,b;

a=digitalRead(pcinput21);

b=digitalRead(pcinput22);

if(a==1&&b==1&&pcstate!=2)

{ digitalWrite(pcoutput,HIGH);

Serial.print(42);

pcstate = 2;

EEPROM.write(pceeprom,pcstate);

}

else if(a==0&&b==0&&pcstate!=1)

{ digitalWrite(pcoutput,LOW);

Serial.print(41);

pcstate = 1;

EEPROM.write(pceeprom,pcstate);

}

}

else //override function

{ if(digitalRead(pctoggle)==1&&pcstate!=4)

{ digitalWrite(pcoutput,HIGH);

Serial.print(44);

pcstate = 4;

EEPROM.write(pceeprom,pcstate);

}

else if(pctoggle==0&&pcstate!=3)

{ digitalWrite(pcoutput,LOW);

Serial.print(43);

pcstate = 3;

EEPROM.write(pceeprom,pcstate);

}

}

return;

}

void pcstatus()

{

if(pcstate==1)

Serial.print(31);

else if(pcstate==2)

Serial.print(32);

else if(pcstate==3)

Serial.print(23);

else

Serial.print(24);

return;

}

void KEYPAD()

{ int i,j,k,l;

//Serial.println("Enter password:");

char a[5];

for(i=0;i<4;i++)

{ char key = keypad.waitForKey();

//if (key)

//{

// Serial.println(key);

//}

a[i]=key;

}

a[i]='\0';

if(!strcmp(a,b))

{ //Serial.println("Password is correct\n");

Serial.print(53);

for(j=0;j<100;j++)

{ for(k=0;k<2000;k++)

{ if(keypad.getKey()=='\*')

{ //Serial.println("Enter new password");

for(l=0,addr=0;l<4;l++)

{ b[l]=keypad.waitForKey();

c[l]=b[l]-48;

EEPROM.write(addr,c[l]);

//Serial.println(b[l]);

addr++;

}

Serial.print(54);

}

}

}

//Serial.println("Delay over");

motorup();

key = keypad.waitForKey();

if(key=='#')

motordown();

}

else

Serial.print(53);

addr=0;

if (blink)

{ digitalWrite(ledPin,!digitalRead(ledPin)); // Change the ledPin from Hi2Lo or Lo2Hi.

delay(100);

}

}

void motorup()

{

digitalWrite(motorpin1,HIGH);

digitalWrite(motorpin2,LOW);

delay(2500);

digitalWrite(motorpin1,LOW);

}

void motordown()

{

digitalWrite(motorpin2,HIGH);

digitalWrite(motorpin1,LOW);

delay(2500);

digitalWrite(motorpin2,LOW);

}

// Taking care of some special events.

void keypadEvent(KeypadEvent key)

{ switch (keypad.getState()){

case PRESSED:

if (key == '#') {

digitalWrite(ledPin,!digitalRead(ledPin));

ledPin\_state = digitalRead(ledPin); // Remember LED state, lit or unlit.

}

break;

case RELEASED:

if (key == '\*') {

digitalWrite(ledPin,ledPin\_state); // Restore LED state from before it started blinking.

blink = false;

}

break;

case HOLD:

if (key == '\*') {

blink = true; // Blink the LED when holding the \* key.

}

break;

}

}

**Chapter 6**

**CONCLUSION**

**6.1 Observation**

* Program to run the two Atmega’s were obtained
* The individual systems were designed and run successfully
* Serial communication was obtained

**6.2 Future Scope**

The scope for improvement is immense. The various data logs from the sensors can be shared over the internet and an algorithm can be implemented to take intelligent decisions regarding the power consumption etc. It could also be used to monitor the national averages etc. We could also incorporate wireless sensor networks for efficient power management.The Raspberry Pi can be configured to act as a radio transmitter.This acts as an emergency radio service. The keypad can be replaced by a voice recognition system.Also we could integrate voice control to almost all the features. The various sensors can be connected to the central microcontroller unit via Zigbee Modules.

