**CS 321**

**HOME AUTOMATION**

**USING**

**RASPBERRY PI**

***By Group 18***

**Objective:**

Creating a Smart Home Automation System using Raspberry Pi.

**Instruments Used :**

* Raspberry Pi
* Arduino(2),
* RF module(one receiver & one tranmitter)
* Ping Sensor(2)
* lm35 Temperature Sensor
* Breadboard
* Jumper Wires
* DC Motor & Motordriver (L293D)
* LED’s.

**Features of the Smart Home :**

* The Smart Home keeps a count of the number of people in the room. When the number of people in the room becomes 0, it intelligently switches off the lights.
* The Smart Home has a temperature sensor, that can measure the temperature in the room and depending on how hot it is can adjust the speed of the fan.
* The Smart Home can be controlled through a web server, a person can switch on and off the lights from far way using internet.
* There is a webcam in the Smart Home, which allows one to keep a watch at the house remotely. The webcam displays image through a web server.

**Overview of the implementation of the Project:**

* To keep a count of the number of people entering the room, we have two ping sensors that communicate directly with one Arduino1. When a person enters a room ping sensor communicates this to Arduino1 and it transfers this information wirelessly using RF Module to Arduino number2. Ardunio2 gives this information to Raspberry Pi, since it is directly connected which updates the count. Similar procedure is followed when a person leaves the room. When the count of number of people in the room becomes 0, the Raspberry Pi switches off the light, if it was alread on.
* There is a temperature sensor connected to Arduino1. This value of temprature is trasmitted to Arduino2 wirelessly thru rf module. The Raspberry pi takes this value from Arduino2 and uses Pulse Wave Modulation to alter the speed of the fan. Hence if the temperature is higher more proportion of the input voltage is used, and the fan runs faster.
* We have hosted a web server on RaspberryPi which can be accessed online by going to a specific address, hence some operations of the RaspberryPi can be done through the internet. Since the RaspberryPi is directly connected with the LED, on pressing the appropriate button, the light is switched on remotely using the internet.
* The Webcam is directly connected to RaspberryPi and on appropriate command from the webpage it is activated by the user. This Webcam transmits its streaming to the RaspberryPi which being a web server, the image can be seen online.

**Module 1**

**Hosting a Web Server on RPi and controlling LEDs through it**

We used lighttpd service to host a lightweight server at the Raspberry Pi. We had already configured the ethernet to the static IP settings.

INSTALLING LIGHTTPD

To install the lighttpd web server issue the command.

*sudo apt-get install lighttpd*

This will install the web server and also pull in any other packages (called dependencies) that are required. The server will be automatically started and set to start by default after a reboot.

It is useful to change the permissions on the www directory to allow your user to update the web pages without needing to be root.

*sudo chown www-data:www-data /var/www*

*sudo chmod 775 /var/www*

*sudo usermod -a -G www-data pi*

Once the setup is complete you can access the web page by pointing your browser to the router IP address or DNS entry.

By default, lighttpd runs with a single website that lives in the directory /var/www. We can set up our own website in there by creating a new file /var/www/index.html and putting some HTML code in it.

To make the **webserver password protected** we do the following:

we create a password file.

This file can go anywhere, but for simplicity in this walkthrough, we’ll store it *in/etc/lighttpd/.htpasswd .*

Then we created the hidden .htpasswd directory:

*sudo mkdir /etc/lighttpd/.htpasswd*

Next, we created a shell script that will hash a user’s password and put everything in the right format for the file and made it executable.

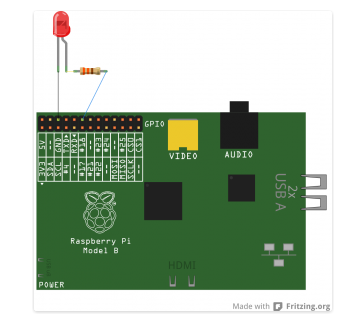
Then we added a user and the hashed password to the password file.

Then, we added “mod auth” to the lighttpd.conf file and specified the path we wanted to protect.

Then, when we restarted the lighttpd server, it asked for authentication.

Finally, we connect the LED and resistor in series between the Pi’s ground pin and one of its GPIO pins, we have used Pin no: 18.

Below is how the connection looks like:

****

Then, we wrote the python script to control the LEDs. We used the **RPI.GPIO library** to control the LEDs. We used the BCM mode to use the Board Pin numbers and we used the Pin number 18.

The Python scripts controls the output of the pin number 18 according to the terminal input. “w” for on and “s” for off.

The web page calls the python script on loading and the on and off buttons are used to give “w” and “s” terminal outputs respectively. So, the button click causes the terminal output which is used to control the pin output.

|  |  |
| --- | --- |
|  |  |
|  |  |

**Webcam on the Server**

We use the **motion service** to control the USB webcam.d

We installed the webcam server software using the following command:

*sudo apt-get install motion*

Then we made the necessary changes in the conf file and turned the Daemon service on and enabled the webcam\_localhost option.

When we restarted the motion service, and browsed to our the link 10.11.2.49:8081(ip address), we could see live streaming from the webcam on the webpage.

We finally added this link to our homepage.

**Module 2**

**DC motor through Temperature**

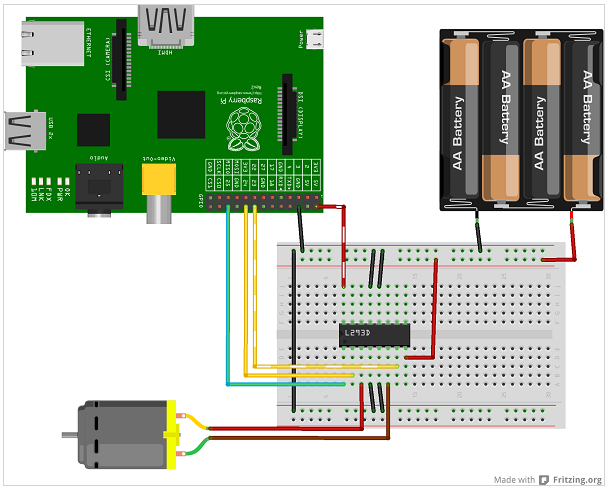
Our aim was to control speed of a fan in a room according to the present temperature in the room. We connected the temperature sensor with the arduino and then the output of it is sent thru the rf modules we used to make a wireless connection in between the raspberry pi and arduino with the temperature sensor.   
We used **Pulse Width Modulation** to control the speed of the motor according to the temperature received.   
Pulse Width Modulation (or PWM) is a technique for controlling power. We use it here to control the amount of power going to the motor and hence how fast it spins.  
  
1.Controlling 12v DC motor thru GPIO pins

As the output from a GPIO pin is 3.3v when the pin is high it is not feasible to supply power to DC motor directly thru the GPIO pin. So what we need is to take power from another source for the DC motor.   
To connect motor with Raspberry pi   
a) Requirements

* A Breadboard to connect everything on
* An L293D motor driver chip
* Jumper cables to connect everything up (Male to male and female to male)
* DC motors rated for 6-12v
* 12v Adapter as a power source

We used the following link for this particular task :  
<http://computers.tutsplus.com/tutorials/controlling-dc-motors-using-python-with-a-raspberry-pi--cms-20051>

Below is how the connection should be to connect a dc motor :



2. Controlling speed thru PWM (pulse width modulation)

Next task is to control the speed according to the temperature. So we defined three levels of stepping. We will get the value of the temprature thru the receiver of the rf module.

motor = GPIO.PWM(25, 100) # create object motor for PWM on port 25 at 100 Hertz

motor.start(0) # start motor on 0 percent duty cycle (off)

For temperature > 29(in celsius)   
Motor runs at highest speed   
i.e motor.ChangeDutyCycle(100)

For temperature > 26(in celsius)   
Motor runs at moderate speed   
i.e motor.ChangeDutyCycle(75)

For temperature > 23(in celsius)   
Motor runs at half speed   
i.e motor.ChangeDutyCycle(50)

For temperature > 21(in celsius)   
Motor is off   
i.e motor.ChangeDutyCycle(0)

With these changes in the Duty Cycle we were able to achieve the change in speed of the DC motor according to the variation in temperature.

**Module 3**  
**Control LED based on number of people in the room**

The value of count (i.e the number of people in the room) is directly transmitted to raspberry pi thru the rf modules. And an LED is connected to raspberry pi. If count is greater than zero than GPIO pins goes high and the led blows and if count is 0 the GPIO pin goes low, thus switching off the led.  
Arduino1 finds the value of count which is further transmitted to rpi. Arduino1 uses PING sensor to identify if any objects crosses thru the door and our system has different exit and entry door. So, based on the PING sensor Arduino easily finds out the value of count in the room and then transmit it to rpi thru rf module transmitter.

**Connection with Arduino**

There are two types of connection with the Arduino. One is the connection of Sensors with the Arduino and another is the wireless connections of Arduino1 and Arduino2 so that the sensors can communicate with the Raspberry Pi.

**Feature1:** Automatic light on/off**:**

The purpose of this feature is to switch the lights of the home on/off depending on the whether someone is present in the home**.**

**Hardware modules used:** Ping sensor, RF module

**Software modules used:** NewPing (Arduino Library), RCSwitch (Arduino Library)

**Hardware modules**:

Ping sensor (HC-SR04): The [HC-SR04](http://cytron.com.my/p-sn-hc-sr04) ultrasonic sensor uses sonar to determine distance to an object like bats or dolphins do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. From 2cm to 400 cm or 1” to 13 feet. It operation is not affected by sunlight or black material like Sharp rangefinders are (although acoustically soft materials like cloth can be difficult to detect). It comes complete with ultrasonic transmitter and receiver module.

**Features:**

* Power Supply :+5V DC
* Quiescent Current : <2mA
* Working Currnt: 15mA
* Effectual Angle: <15°
* Ranging Distance : 2cm – 400 cm/1" - 13ft
* Resolution : 0.3 cm
* Measuring Angle: 30 degree
* Trigger Input Pulse width: 10uS
* · Dimension: 45mm x 20mm x 15mm

Here is an image of the sensor



**RF module**: Here is an image of the two sensors



The left one is the transmitter and the right one is the receiver. These are used to wirelessly tansmit and receive signals over 433MHz transmission channel. The ATAD pin of the transmitter is the data pin which needs to be connected to the pin of arduino passed to mySwitch.enableTransmit(). The VCC and GND needs to be connected to +5V and ground of arduino. Similarly the 2nd and 3rd pin of receiver are data pins and any one can be used to receive the signals.

**Software Modules**:

**NewPing**: Allows to calculate time taken for ultrasonic waves to return back from the time they were sent.

**RCSwitch**: Allows to transmit data in our code over 433MHz transmission channel. Also allows to receive the signals at the receiver end of the RF module.

Code for count thru the Ping Sensor:

#include <NewPing.h>

#include <RCSwitch.h>

#define TRIGGER\_PIN 3 // Arduino pin tied to trigger pin on the ultrasonic sensor.

#define ECHO\_PIN 4 // Arduino pin tied to echo pin on the ultrasonic sensor.

#define MAX\_DISTANCE 200 // Maximum distance we want to ping for (in centimeters). Maximum sensor distance is rated at 400-500cm.

#define TRIGGER\_PIN1 6

#define ECHO\_PIN1 7

NewPing sonar(TRIGGER\_PIN, ECHO\_PIN, MAX\_DISTANCE); // NewPing setup of pins and maximum distance.

NewPing sonar1(TRIGGER\_PIN1, ECHO\_PIN1, MAX\_DISTANCE);

RCSwitch mySwitch = RCSwitch();

int count = 1;

int cycle = 0;

int cycle1 = -10;

int count1;

boolean flag = 0;

void setup() {

Serial.begin(9600); // Open serial monitor at 115200 baud to see ping results.

mySwitch.enableTransmit(10);

}

void loop() {

//delay(1);

unsigned int uS = sonar.ping(); // Send ping, get ping time in microseconds (uS).

Serial.print("Ping: ");

Serial.print(uS / US\_ROUNDTRIP\_CM); // Convert ping time to distance in cm and print result (0 = outside set distance range)

Serial.println("cm");

if(uS / US\_ROUNDTRIP\_CM < 30 && uS / US\_ROUNDTRIP\_CM > 0 && cycle-cycle1 > 4)

{

if (count > 1)

{

count--;

flag = 1;

}

}

unsigned int uS1 = sonar1.ping(); // Send ping, get ping time in microseconds (uS).

Serial.print("Ping: ");

Serial.print(uS1 / US\_ROUNDTRIP\_CM); // Convert ping time to distance in cm and print result (0 = outside set distance range)

Serial.println("cm");

if(uS1 / US\_ROUNDTRIP\_CM < 30 && uS1 / US\_ROUNDTRIP\_CM > 0 && cycle-cycle1 > 4)

{

count++;

flag = 1;

}

if(flag)

{

cycle1 = cycle;

flag = 0;

}

Serial.print("count =");

Serial.println(count-1);

mySwitch.send(count, 24);

cycle++;

cycle = cycle%10000;

}

**Explanation:**

TRIGGER\_PIN is the corresponding pin of your arduino board and similarly ECHO\_PIN is the pin of arduino needed to be connected to echo of the ping sensor. We have two sets of ping sensors and they have their differnet pin configurations.

The library NewPing.h is used to calculate the distance from the ping sensor using just a single function. We create an object of type NewPing called sonar. It takes 3 parameters Trigger Pin, Echo Pin and maximum distance that can be detected. It has a function called sonar.ping() that returns the time it needed for ultrasonic wave to return back to receiver from the time it was send using the transmitter of the ping sensor. A division by a constant US\_ROUNDTRIP\_CM gives the distance of the object from the ping sensor.

If the distance of an object from sensor1 goes below 30, it implies that someone is coming in and we increase the count by 1. Similarly if the distance of an object goes below 30 from sensor2, it implies someone is leaving the room and hence we decrement the count by 1. We also have included the feature of avoiding delay. We make sure that the number of cycles of signal passed between previous change of count and this pending change is greater than 4 because otherwise it can be just someone coming/going slowly.

Finally we transmit this count to the RF Module receiver (433 MHz transmission) using the software module RCSwitch. We created an object mySwitch of class RCSwitch. We then transmit using this object. Here we have used mySwitch.enableTransmit(10) because the data pin of the RF transmitter is connected to pin 10 of arduino we used. We use mySwitch.Send(count, 24) to send the count. 24 is just a coding format to send the data over 433MHz transmission.

code of the receiver of the RF module:

#include <RCSwitch.h>

RCSwitch mySwitch = RCSwitch();

void setup() {

Serial.begin(9600);

mySwitch.enableReceive(0); // Receiver on inerrupt 0 => that is pin #2

}

void loop() {

if (mySwitch.available()) {

int value = mySwitch.getReceivedValue();

if (value == 0) {

Serial.print("Unknown encoding");

} else {

//Serial.print("Received ");

Serial.print( mySwitch.getReceivedValue() );

Serial.print("\n");

//Serial.print( mySwitch.getReceivedBitlength() );

//Serial.print("bit ");

//Serial.print("Protocol: ");

//Serial.println( mySwitch.getReceivedProtocol() );

}

mySwitch.resetAvailable();

}

}

**Explanation:**

Here also we make an instance of the RCSwitch class, mySwitch. We use mySwitch.enableReceive(0) to make the pin #2 of the arduino as the receiver pin and hence this needs to be connected to the data pin of the receiver. Also mySwitch.reciveValue() returns the data received at the receiver which is stored in the variable val.

**Feature**: Controlling speed of fan depending on the temperature.

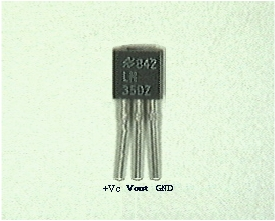
**Logic**: We are sending the temperature to the raspberry-pi from arduino using the temperature sensor. We have three levels of fan speed which are controlled using Pulse Wave Modulation(PWM)

**Hardware Modules used**: Temperature sensor, RF Module

**Softwafre Modules used**: RCSwitch

**Hardware Modules**:

Temperature sensor (LM35): The LM35 is an integrated circuit sensor that can be used to measure temperature with an electrical output proportional to the temperature (in **o**C). Here is an image of the sensor.



RF Module: Covered before

**Software Modules**:

RCSwitch: Covered before

Code for temprature and transmiiting it to rpi:

#include <RCSwitch.h>

const int sensorPin = A0; // select the input pin for the potentiometer

int sensorValue = 0; // variable to store the value coming from the sensor

RCSwitch mySwitch = RCSwitch();

void setup() {

Serial.begin(9600);

mySwitch.enableTransmit(10);

}

void loop() {

// read the value from the sensor:

sensorValue = analogRead(sensorPin);

sensorValue = sensorValue\*500/1024;

Serial.println(sensorValue) ;

Serial.print("CELSIUS: ");

Serial.print(sensorValue);

Serial.println("\*C ");

mySwitch.send(sensorValue, 24);

delay(200);

}

**Explanation**:

sensorValue variable takes in the voltage experienced across the temperature sensor. We then multiply this with the constant 500/1024 to get the temperature in degree Celsius.

SensorPin variable takes the value of the pin on arduino to which the sensor pin of temperature sensor would be connected. AnalogRead(sensorPin) is the function used to get the voltage stored in sensorValue.

By using RCSwitch, we transmit this temperature to the receiving arduino which in turns pass it to the raspberry pi.

**Problems Faced in Implementation:**

1. Configurring Proxy Settings

At the very start we faced the problem to connect our raspberry pi to the internet due to the proxy configuration problems. Making IP settings and setting up the proxy settings took a lot of time. But at the end we finally overcame this problem and then used wifi module to work on the pi. And also using 'wget' command in rpi terminal is not working due to this proxy and server problems.

2. Connecting Raspberry pi and Arduino

what we wanted is to have wireless connections in between the sensors we are using and the raspberry pi. Initially we thought to use Xbee to overcome this problem. So, we required one more arduino which is connected to sensor and connected with Xbee transmitter. But we find it very difficult to connect the Xbee receiver with the rpi. We tried but nothing was working. So, finally we used rf modules, one receiver and one trasmitter. Advantage of rf modules is that first it is very cheap than xbee for wireless transmission and another is that it is easy to configure these with arduino. We used one more arduino and then connected a transmitter to one arduino with sensor and connected another arduino with receiver with raspberry pi.

3. Using Ping Sensor (than LDR sensor)

Our aim was to switch on/ off the lights depending on the number of peoples in the room. So, we kept count by putting Ping sensors on the entry and exit section of door. Intially the Ping sensor was too much fluctuating, so we decided to use the LDR sensors to identify the object(if passed). While making a demo on a breadboard if worked fine as we had put led's in front of LDR sensors and if some object pass thru it then amount of light on LDR decreases and by noting that we can increase/decrease count. But when we installed it on a door then it was not working properly. So, again we decided to use the PING sensor and then after installing it to the door it was working fine.

Also, due to lack of the PIR sensors we just identified the objects not only human beings if passed thru the gates. If given a pair of PIR sensors we could have identified only humans.

4. Running DC motor using Transistor as a Switch

At the starting we didn't had motor driver. So to run motor thru GPIO pin and external power supply we first tried to use transistor as a switch. But this was not working. Then finally we arranged a motor driver and it worked just fine.

5. Using Temperature Sensor

We used lm35 temperature sensor. When we first used it, it shows the correct temperature but whenever we used it again it showed some random temperature. Then we realised that to put a resistance in between the vcc and the sensor. Although this was not suggested anywhere but eventually it worked.

6. Voice Recognition

We could not configure voice recognition because first, microphone quality was not excellent and secondly the proxy configuration meant that we could not use Google APi which is the best voice recognition software available.

6. Making the System

One of the main aim of our project is to control lights using the count of the peoples in the room. So, for this we need to think of a system for which we can easily count the number of persons. Finally, we decided to have a different entry and exit section in a room and installed one PING sensor on the entry gate and one on exit gate.

**Limitations of the Project:**

* We require two doors to be there, one for entry and for the exit. Ideally it should be single gate for both entry/exit.
* In our system, it is not possible to detect the entry/exit of two or more persons if they enter/exit together i.e, within a time gap of 2 seconds. In such a situation, count of the number of persons inside the room will always increase/decrease by one.
* We cannot be sure by looking at the app, whether the lights of the room are On/Off.

But we have a camera for that.

* It is not possible to check the past feed of the webcam. We are not storing the video feed from the camera.
* The fan speeds are fixed according to the temperature. There is no way a user which does not understand the code can change the temperature speed settings.
* In our implementation, the light which is controlled through the web app and the light whose condition is determined by the count of persons in the room are different.This is a rather inefficient implementation.
* The no of lights are fixed. If you want to control the lights separately, you need to change the code.

**Extra Features that could be implemented using minimal changes in our existing system**

* We could add an extra feature depending such that the light is only switched on when it is evening/night time. We only need an extra check condition that depends on the time of the day.
* We could keep statistics of the number of people entering the room and the number of times light/fan is switched on. This can be seen online. This will help us in finding how much energy efficient our home is and what we can do for more energy savings.
* We could add another security feature that, we could lock the room for a period of time. As soon as someone breaks it by crossing the ping sensors, we could alert the owner.
* We could add a feature for fire safety if the temperature crosses a particular limit, alert the owner. We already have a temperature sensor and depending upon it the speed of the fan is adjusted, we just need to have an extra condition that if the temperature exceeds a particular threshold alert.
* Using Web Camera, we can do face detection which can be used for security purposes and notify the owner if an unidentified person tries to break in. Also we could use it to monitor children.
* We could use dimmable lights to change the lighting according to the mood and the occasion.
* We could use a temperature sensor with the geyser and configure it to heat the water to a preset temperature.