IOT BASED WATER LEVEL CONTROLLER

A Project Report submitted to Don Bosco Institute of Technology

In partial fulfilment of the requirement for the award of

Degree In

ELECTRONICS AND TELECOMMUNICATION ENGINEERING

Submitted by

JUSTINE AYROOR 6

SUMAN DEB 16

SOHAN NAIK 48

Under the guidance of

Prof. JITHIN ISAAC

MINI-PROJECT I (ETL504)

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATIONS

DON BOSCO INSTITUTE OF TECHNOLOGY

(2015-2016)

Don Bosco Institute of Technology , Kurla West, Mumbai

(2015-2016)

CERTIFICATE

The group project titled "**IOT BASED WATER LEVEL CONTROLLER**" was carried out by the following students:

NAME	ROLL NO	EXAM SEAT NO
JUSTINE AYROOR	**06**	**1505**
SUMAN DEB	**16**	**1515**
SOHAN NAIK	**48**	**1547**

Submitted in partial fulfilment of the requirement for the award of Degree in

Electronics and Telecommunication Engineering

Project Guide **HOD** **Principal**

External Examiner

Acknowledgment

Our very sincere thanks to our project guide **Mr.Jithin Isaac** and **Mr.Girish Chapale** for motivating, co-operating and guiding us throughout the project with their effective skills and huge knowledge base. We would like to thank **Dr. Sudhakar Mande**, Head of Department, for his valuable guidance, support, suggestions and precious time in every possible way throughout the project activity.

We would like to take this opportunity to express our deep gratitude to all the staff who have given their valuable support and co-operation in spite of their busy schedule. Without their advice and help it would have been difficult to complete this work.

We wish to express our deep gratitude towards all our colleagues for their encouragement and moral support

We would like to specially thank Mrs. Joanne Gomes from SFIT who is a member of IEEE to conduct a workshop on raspberry pi and this inspired us to take up this project .

TABLE OF CONTENTS

IOT BASED WATER LEVEL CONTROLLER

Justine Ayroor (Roll No. 6)
Student, Bachelor of Engineering, Department of Electronics & Telecommunication, Don Bosco Institute of Technology, Mumbai
justine2496@gmail.com

Suman Deb (Roll No. 16)
Student, Bachelor of Engineering, Department of Electronics & Telecommunication, Don Bosco Institute of Technology, Mumbai
suman28795@gmail.com

Sohan Niak (Roll No. 48)
Student, Bachelor of Engineering, Department of Electronics & Telecommunication, Don Bosco Institute of Technology, Mumbai

***Abstract*— The technology is a never ending process and these technologies will tend to improve the quality of any product. To be able to design a product using the current technology which is beneficial to the lives of others is a huge contribution to the society. This paper presents the design and implementation of a low cost, Tangible as well as flexible and secure cell phone based device automation system. The design is based on a standalone Raspberry Pi Model B+ board and the home appliances are connected to the input/output ports of this board. The communication between the cell phone and the Raspberry Pi board is wireless due to which the system can be used by any person who can operate an android phone & computer. This system is low cost and scalable that allows variety of devices to be controlled with minimum changes to its core.**

***Keywords*— Rapberry Pi , Electronic Devices, Smartphone, Web Server, PC.**

I. INTRODUCTION

Raspberry Pi is a credit card sized computer used to connect the outside world using the GPIO pins.In this Project we control and monitor the level of water automatically and display the output of the sensor on a webpage and also display the status of the motor i.e the pump on the HTML page

II. THE CONCEPT OF THE PROJECT

.

Raspberry Pi uses a basic programming language called PYTHON . It is used to control the Raspberry pi to the outside world .So basically we program the raspberry pi using Python & FLASK which is a web development framework for python to display the values on a HTML page

2.RASPBERRY PI MODEL B+

1.1.What is RPi

3. The **Raspberry Pi** is a series of credit card–sized single-board computers developed in the United Kingdom by the Raspberry Pi Foundation with the intention of promoting the teaching of basic computer science in schools and developing countries.[5][6][7]

4. The original Raspberry Pi and Raspberry Pi 2 are manufactured in several board configurations through licensed manufacturing agreements with Newark element14 (Premier Farnell), RS Components and Egoman. These companies sell the Raspberry Pi online.[8] Egoman produces a version for distribution solely in Taiwan, which can be distinguished from other Pis by their red colouring and lack of FCC/CE marks. The hardware is the same across all manufacturers.

5. The original Raspberry Pi is based on the Broadcom BCM2835 system on a chip (SoC),[2] which includes an ARM1176JZF-S 700 MHzprocessor, VideoCore IV GPU,[9] and was originally shipped with 256 megabytes of RAM, later upgraded (models B and B+) to 512 MB.[3][10] The system has Secure Digital (SD) (models A and B) or MicroSD (models A+ and B+) sockets for boot media and persistent storage.[11]

6. In 2014, the Raspberry Pi Foundation launched the *Compute Module*, which packages a BCM2835 with 512 MB RA

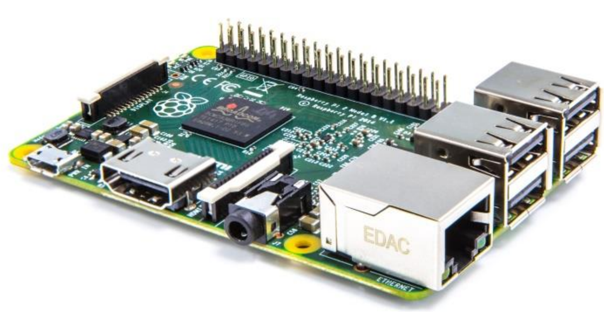
7. M and an eMMC flash chip into a module for use as a part of embedded systems.[12]

8. The Foundation provides Debian and Arch Linux ARM distributions for download.[13] Tools are available for Python as the main programming language, with support for BBC BASIC[14] (via the RISC OS image or the Brandy Basic clone for Linux),[15] C, C++, Java,[16] Perl and Ruby.[17]

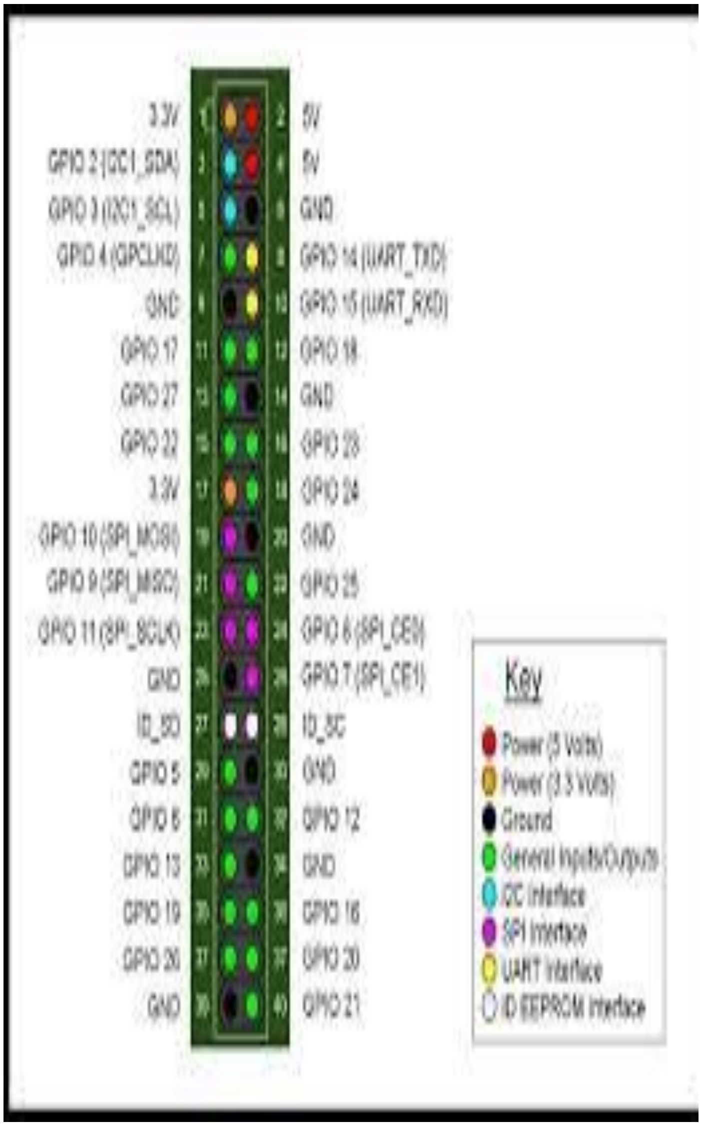
9. As of 8 June 2015, about five to six million Raspberry Pis have been sold.[18][19] While already the fastest selling British personal computer, it has also shipped the second largest number of units behind the Amstrad PCW, the "Personal Computer Word-processor", which sold eight million.

10. In early February 2015, the next-generation Raspberry Pi, Raspberry Pi 2, was released.[20] The new computer board is initially available only in one configuration (model B) and features a Broadcom BCM2836 SoC, with a quad-core ARM Cortex-A7 CPU and a VideoCore IV dual-core GPU; 1 GB of RAM with remaining specifications being similar to those of the previous generation model B+. The Raspberry Pi 2 retains the same US$35 price point of the model B,[21] with the US$20 model A+ remaining on sale.

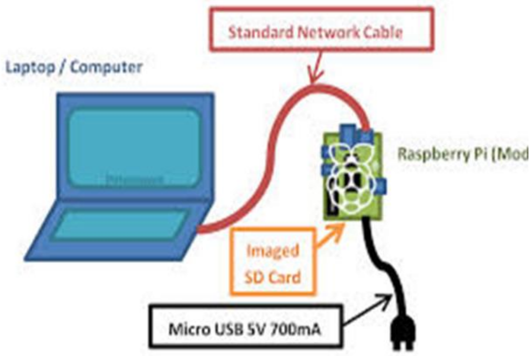
11.



2.2 GPIO PINS


Key
Power (5 Volts)
Power (3.3 Volts)
Ground
General Inputs/Outputs
I2C Interface
SPI Interface
UART Interface
ID EEPROM interface

2.2.Connect to remote desktop

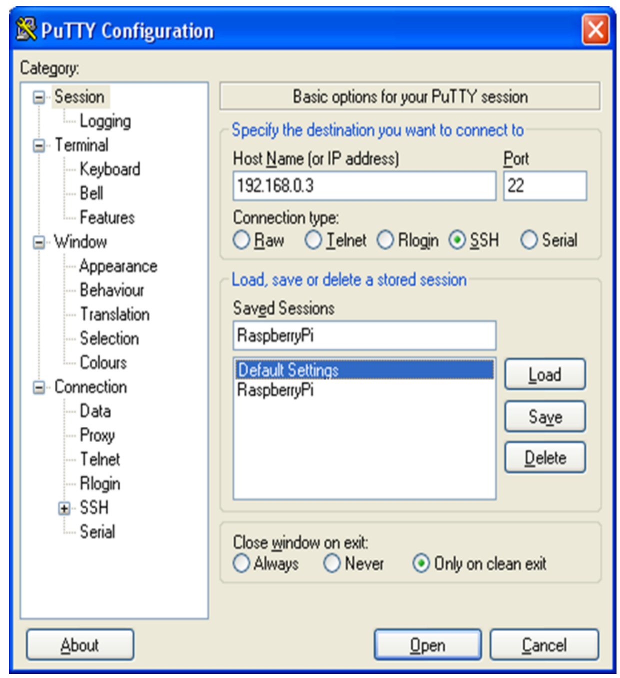


Go to sudo nano /etc/network/interfaces

Edit the file save it and connect it

- *Static and dynamic IP to configure eth0*
- *Open file /etc/network/interfaces*
  $ sudo nano /etc/network/interfaces
  auto lo eth0
  Iface eth0 inet static
  address 192.186.1.48
  netmask 255.255.255.0
  network 192.168.1.0
  gateway 192.168.1.254
  broadcast 192.168.1.255

- *Install puTTY.exe terminal emulator on you PC*
  - *Insert Rpi's IP as Host name*
  - *Give name to the connection*
    - *Choose SSH*
    - *Save..load..open*
  - *It will open terminal window*
    - *Login to Rpi*
- *Gain access to RPi's terminal*



2.3 . CONNECT TO WIFI

- *Dynamic IP to configure wifi*
  - *Open file etc/wpa_supplicant/wpa_supplicant.config*

$ sudo nano /etc/wpa_supplicant/wpa_supplicant.config

Add the following…

Network={

ssid="ur_APname"

psk="ur_APpasswd"

id_str="home"

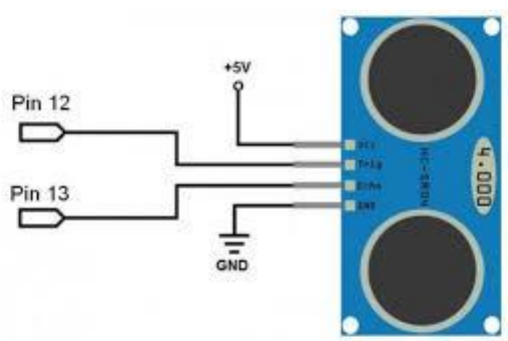
}

3.HC-SR04 Ultrasonic Sensor

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work: (1) Using IO trigger for at least 10us high level signal, (2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back. (3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning. Test distance = (high level time×velocity of sound (340M/S) / 2,

Wire connecting direct as following:

5V Supply
Trigger Pulse Input
Echo Pulse Output
0V Ground



WORKING:

Sound consists of oscillating waves through a medium (such as air) with the pitch being determined by the closeness of those waves to each other, defined as the frequency. Only some of the sound spectrum (the range of sound wave frequencies) is audible to the human ear, defined as the “Acoustic” range. Very low frequency sound below Acoustic is defined as “Infrasound”, with high frequency sounds above, called “Ultrasound”. Ultrasonic sensors are designed to sense object proximity or range using ultrasound reflection, similar to radar, to calculate the time it takes to reflect ultrasound waves between the sensor and a solid object. Ultrasound is mainly used because it’s inaudible to the human ear and is relatively accurate within short distances. You could of course use Acoustic sound for this purpose, but you would have a noisy robot, beeping every few seconds. . . .

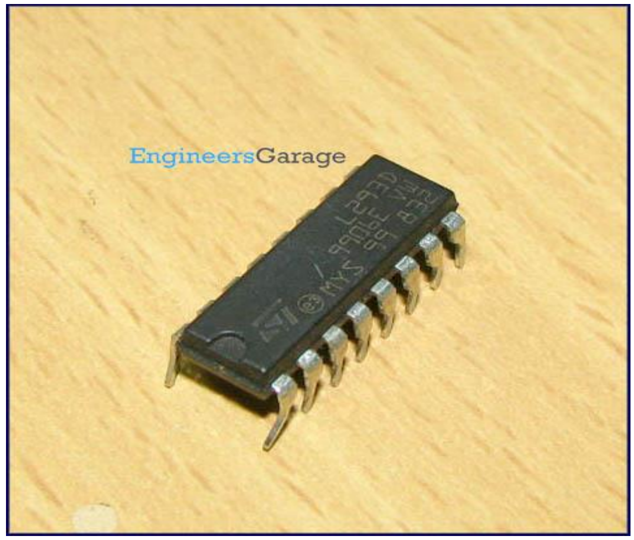
A basic ultrasonic sensor consists of one or more ultrasonic transmitters (basically speakers), a receiver, and a control circuit. The transmitters emit a high frequency ultrasonic sound, which bounce off any nearby solid objects. Some of that ultrasonic noise is reflected and detected by the receiver on the sensor. That return signal is then processed by the control circuit to calculate the time difference between the signal being transmitted and received. This time can subsequently be used, along with some clever math, to calculate the distance between the sensor and the reflecting object.

The HC-SR04 Ultrasonic sensor we’ll be using in this tutorial for the Raspberry Pi has four pins: ground (GND), Echo Pulse Output (ECHO), Trigger Pulse Input (TRIG), and 5V Supply (Vcc). We power the module using Vcc, ground it using GND, and use our Raspberry Pi to send an input signal to TRIG, which triggers the sensor to send an ultrasonic pulse. The pulse waves bounce off any nearby objects and some are reflected back to the sensor. The sensor detects these return waves and measures the time between the trigger and returned pulse, and then sends a 5V signal on the ECHO pin.

ECHO will be “low” (0V) until the sensor is triggered when it receives the echo pulse. Once a return pulse has been located ECHO is set “high” (5V) for the duration of that pulse. Pulse duration is the full time between the sensor outputting an ultrasonic pulse, and the return pulse being detected by the sensor receiver. Our Python script must therefore measure the pulse duration and then calculate distance from this.

IMPORTANT. The sensor output signal (ECHO) on the HC-SR04 is rated at 5V. However, the input pin on the Raspberry Pi GPIO is rated at 3.3V. Sending a 5V signal into that unprotected 3.3V input port could damage your GPIO pins, which is something we want to avoid! We’ll need to use a small voltage divider circuit, consisting of two resistors, to lower the sensor output voltage to something our Raspberry Pi can handle.

4.L293D IC



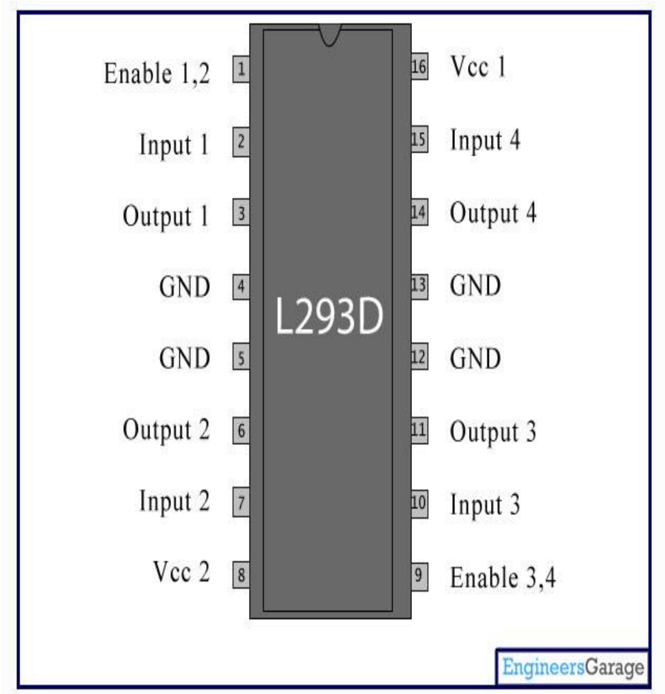
L293D is a dual H-bridge motor driver integrated circuit (IC). Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors.

L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 & 15. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively.

Enable pins 1 and 9 (corresponding to the two motors) must be high for motors to start operating. When an enable input is high, the associated driver gets enabled. As a result, the outputs become active and work in phase with their inputs. Similarly, when the enable input is low, that driver is disabled, and their outputs are off and in the high-impedance state.

1) *Pin Description:*

Pin No	Function	Name
1	Enable pin for Motor 1; active high	Enable 1,2
2	Input 1 for Motor 1	Input 1
3	Output 1 for Motor 1	Output 1
4	Ground (0V)	Ground
5	Ground (0V)	Ground
6	Output 2 for Motor 1	Output 2
7	Input 2 for Motor 1	Input 2
8	Supply voltage for Motors; 9-12V (up to 36V)	Vcc_2
9	Enable pin for Motor 2; active high	Enable 3,4
10	Input 1 for Motor 1	Input 3
11	Output 1 for Motor 1	Output 3
12	Ground (0V)	Ground
13	Ground (0V)	Ground
14	Output 2 for Motor 1	Output 4
15	Input2 for Motor 1	Input 4
16	Supply voltage; 5V (up to 36V)	Vcc_1



5. SOFTWARE

5.1. **Python** is a widely used general-purpose, high-level programming language.[19][20] Its design philosophy emphasizes code readability, and its syntax allows programmers to express conce

pts in fewer lines of code than would be possible in languages such as C++ orJava.[21][22] The language provides constructs intended to enable clear programs on both a small and large scale.[23]

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or proceduralstyles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library.[24]

Python interpreters are available for installation on many operating systems, allowing Python code execution on a wide variety of systems. Using third-party tools, such as Py2exe or Pyinstaller,[25] Python code can be packaged into stand-alone executable programs for some of the most popular operating systems, allowing the distribution of Python-based software for use on those environments without requiring the installation of a Python interpreter.

CPython, the reference implementation of Python, is free and open-source software and has a community-based development model, as do nearly all of its alternative implementations. CPython is managed by the non-profit Python Software Foundation.

5.2 *FLASK-Flask is a microframework for Python based on Werkzeug, Jinja 2 and good intentions. And before you ask: It's BSD licensed!*

Latest Version: 0.10.1

B. *Flask is Fun*

```
from flask import Flask

app = Flask(__name__)


@app.route("/")

def hello():

    return "Hello World!"


if __name__ == "__main__":

    app.run()
```

C. *And Easy to Setup*

```
$ pip install Flask

$ python hello.py

 * Running on http://localhost:5000/
```

Flask
web development,
one drop at a time

5.3. PUTTY

PuTTY is an SSH and telnet client, developed originally by Simon Tatham for the Windows platform. PuTTY is open source software that is available with source code and is developed and supported by a group of volunteers.

Bitvise SSH Client is an SSH and SFTP client for Windows. It is developed and supported professionally by Bitvise. The SSH Client is robust, easy to install, easy to use, and supports all features supported by PuTTY, as well as the following:

- graphical SFTP file transfer;
- single-click Remote Desktop tunneling;
- auto-reconnecting capability;
- dynamic port forwarding through an integrated proxy;
- an FTP-to-SFTP protocol bridge.

7.DESIGN COST TABLE

NAME	QUANTITY	COST
Raspberry Pi Tool Kit	1	5000
HC-SR04 Sensor	1	80
L293D IC	2	120
Breadboard	1	150
Resistors,BJT,LED	-	-
Jumper wires	2(packets)	150
----------------------	TOTAL=	5500

8. Future scope:

→Overhead water tanks
→Dam's water level controller
→Automatic plant watering system
→Fish tanks

9.Conclution:

→Hence we control the level of water by simple programming in python
Using the GPIO pins of Raspberry Pi
→We can also view the status of the sensor and the motor on a webpage to make it convenient for the user using the concept of IOT provided bt the Raspberry Pi

10.Reference:

- http://en.wikipedia.org/wiki/Raspberry_Pi
- http://elinux.org/Rpi_Datasheet_201_Raspberry_Pi_Computer
- http://elinux.org/RPi_Hardware
- http://elinux.org/R-Pi_Hub
- http://www.element14.com/community/docs/DOC-43016/l/broadcom-datasheet-for-bcm2835-soc-used-in-raspberry-pi
- http://www.raspberrypi.org/help/
- http://www.cpdforteachers.com/resources