5/2/2016

Steve Wilson

CodeClub.im

Code Club EAT Project

Teaching Notes

EAT Platform Build Notes

Pi 3 Setup

Install Hostapd & interface

Setup dhcp

Test with phone

Connect ESP8266 to Arduino

Code SoftwareSerial AT Command TestEAT

Test Access Point

Code Python Socket server

Test with Telnet

Test with TestEATv2

Contents

[Pi 3 Setup 3](#_Toc455307589)

[Install Hostapd & interface 3](#_Toc455307590)

[Setup dhcp 3](#_Toc455307591)

[Test with phone 3](#_Toc455307592)

[Connect to ESP8266 with FTDI from PC USB 3](#_Toc455307593)

[Connect ESP8266 to Arduino 3](#_Toc455307594)

[Code SoftwareSerial AT Command TestEAT 3](#_Toc455307595)

[Arduino Programming 3](#_Toc455307596)

[Arduino Overview 3](#_Toc455307597)

[Arduino IDE: Introduction 3](#_Toc455307598)

[Writing programs 3](#_Toc455307599)

[Testing Programs 3](#_Toc455307600)

[Test Access Point Commands 4](#_Toc455307601)

[Attach Pi 3 LCD 4](#_Toc455307602)

[Install Drivers 4](#_Toc455307603)

[Configure 4](#_Toc455307604)

[Code Python Socket server 5](#_Toc455307605)

[Install Geany 5](#_Toc455307606)

[Test with Telnet 5](#_Toc455307607)

[Install Telnet 5](#_Toc455307608)

[Test the Arduino Code 5](#_Toc455307609)

[Write the Website 9](#_Toc455307610)

[Install Apache2 9](#_Toc455307611)

[Create the HTML page 9](#_Toc455307612)

[Design the HTML page 9](#_Toc455307613)

[Install the page 9](#_Toc455307614)

[Make it do something… 9](#_Toc455307615)

[GPIO 10](#_Toc455307616)

[Installing RPi.GPIO 10](#_Toc455307617)

[Install PHP 12](#_Toc455307618)

[Test Raspberry Pi PHP 13](#_Toc455307619)

[Testing the MicroSwitch 13](#_Toc455307620)

[Appendices 16](#_Toc455307621)

[Appendix 1 – References 16](#_Toc455307622)

[Appendix 2 – Example Code 17](#_Toc455307623)

[Arduino Example Code 17](#_Toc455307624)

[Appendix 3 25](#_Toc455307625)

[Python Socket Server Works with TestEATv” 25](#_Toc455307626)

[Appendix 3 - ESP Connections 27](#_Toc455307627)

[Appendix 4 - ESP Command Set 28](#_Toc455307628)

[Appendix 5 - Raspberry Pi PHP 28](#_Toc455307629)

[Appendix 6 - Backup RPi 29](#_Toc455307630)

# Pi 3 Setup

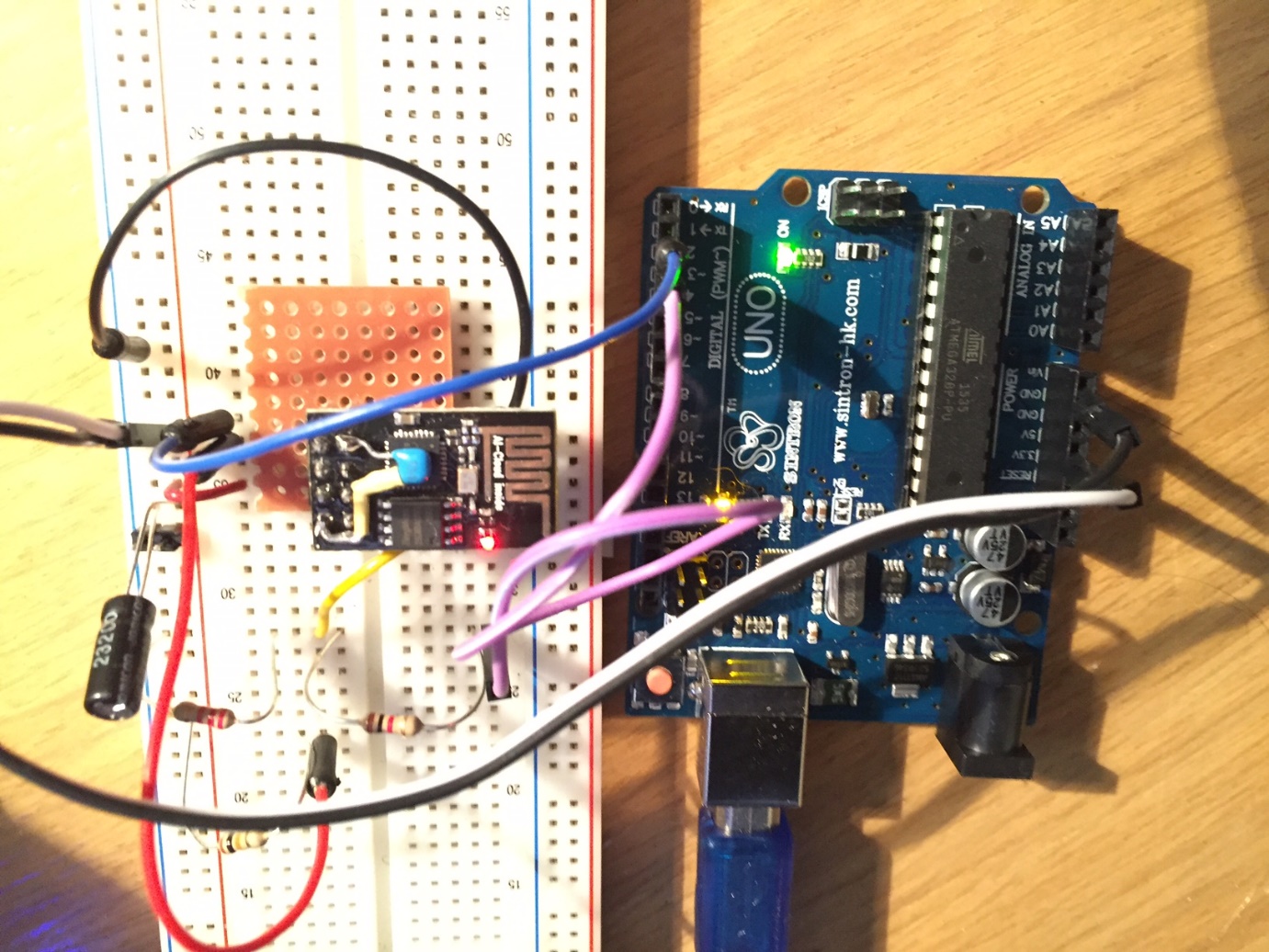
## Install Hostapd & interface

## Setup dhcp

## Test with phone

# Connect to ESP8266 with FTDI from PC USB

# Connect ESP8266 to Arduino



# Code SoftwareSerial AT Command TestEAT

## Arduino Programming

### Arduino Overview

### Arduino IDE: Introduction

### Writing programs

### Testing Programs

Serial Monitor

Debug statements

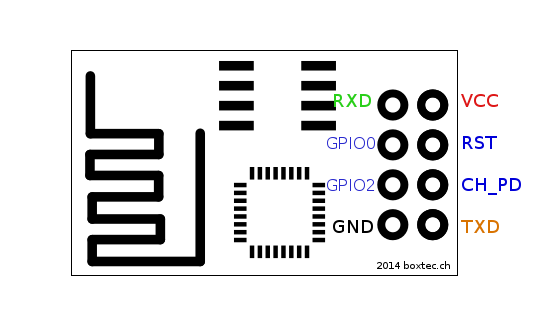
Tcpdump

Netcat – used to listen or send TCP connections

https://learn.adafruit.com/raspipe-a-raspberry-pi-pipeline-viewer-part-2/networked-pipes-with-netcat

SSH

Arduino code constructor: SoftwareSerial(rxPin, txPin, inverse\_logic)



|  |  |  |
| --- | --- | --- |
|  | Arduino Uno | ESP8266 |
| VCC | 3.3v | VCC |
| GND | GND | GND |
| GND | GND | CH\_PD through 10k R |
|  | 3 | RXD through 1K+220 potential divider |
|  | 2 | TXD |

[Wiring Diagram]

## Test Access Point Commands

# Attach Pi 3 LCD

## Install Drivers

## Configure

[see Appendix]

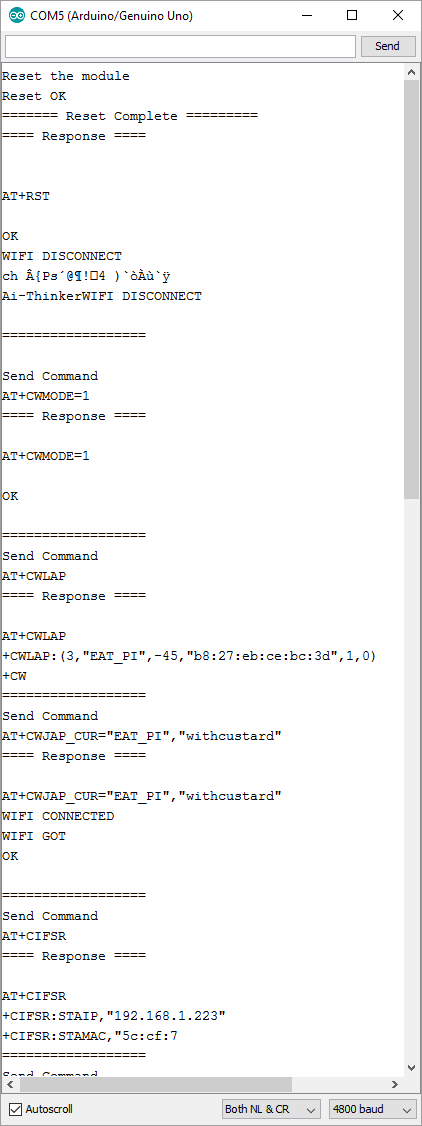
# Code Python Socket server

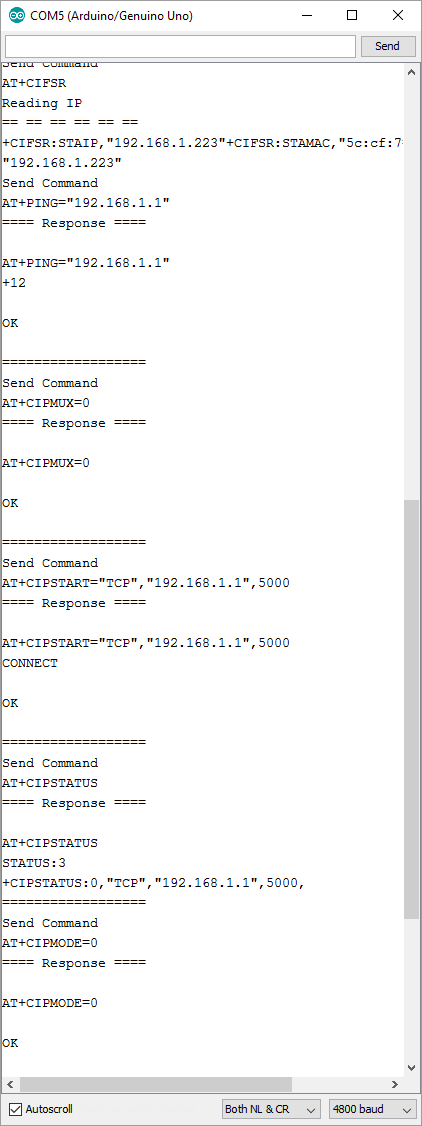
### Install Geany

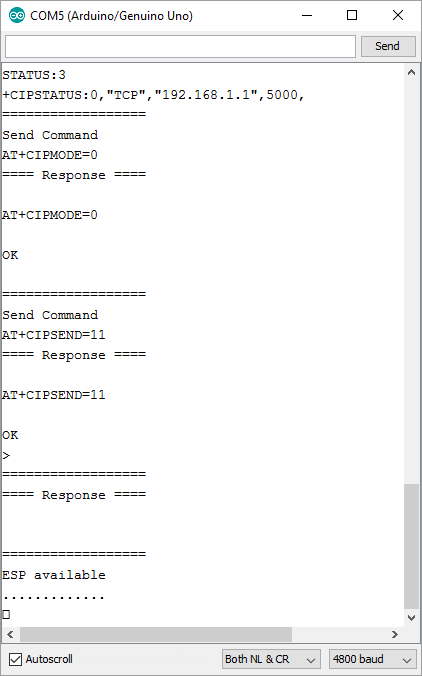
## Test with Telnet

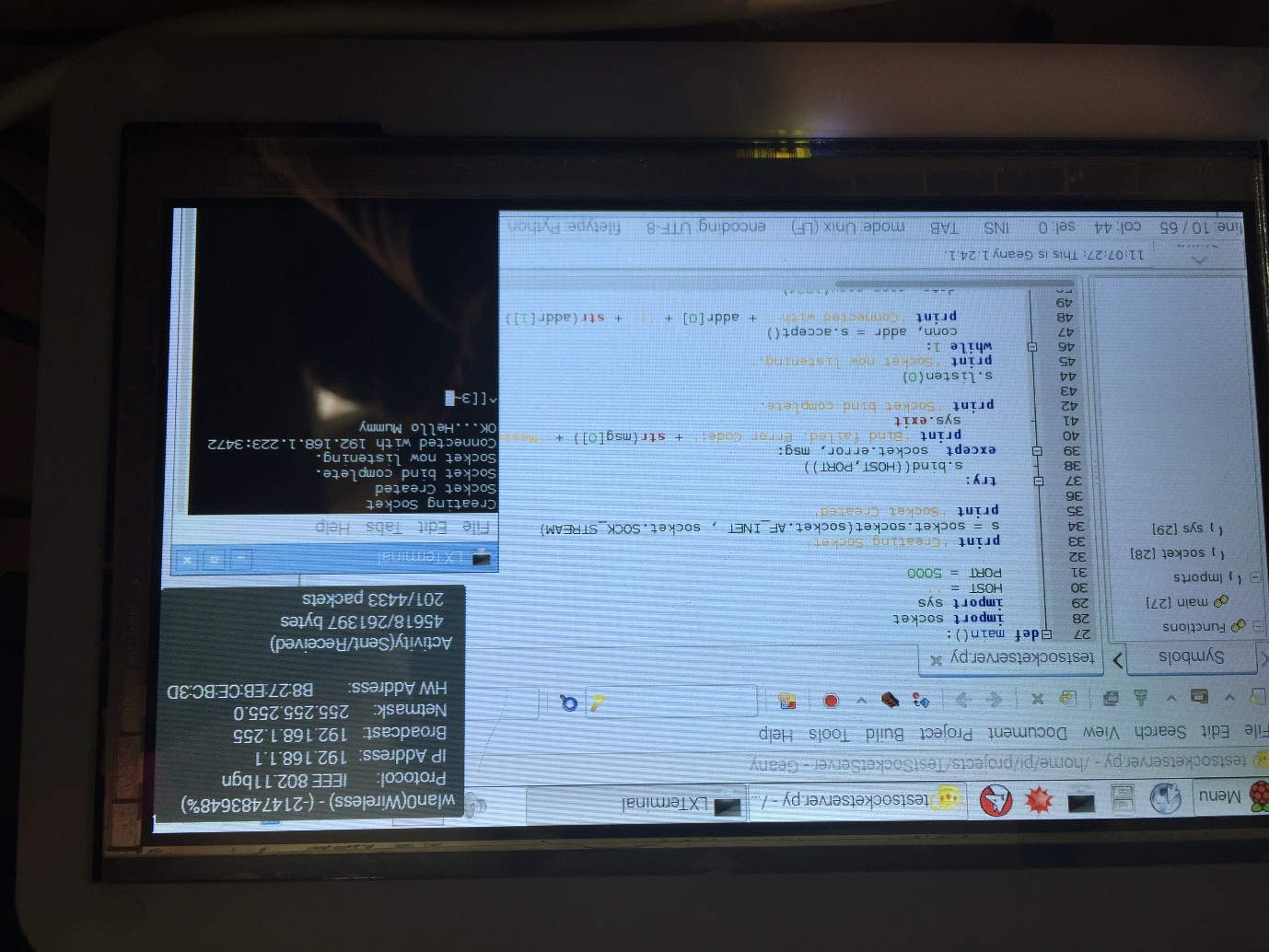
### Install Telnet

## Test the Arduino Code

Test with TestEATv2 







# Write the Website

## Install Apache2

Apt-get install Apache2

Change config.txt(?)

## Create the HTML page

### Design the HTML page

### Install the page

Copy the directory contents to /var/www/html with

sudo cp –r /media/pi/XBOOT/EAT/TestHtmlPage/\* /var/www/html

list it sudo ls /var/www/html

set the permissions up with chmod – the easiest way is to copy the permissions from the demo index.html page using:

sudo chmod –reference=/var/www/html/index.html /var/www/html/\*

connect to it with a smart phone:

### Make it do something…

Add some JavaScript to refresh the page periodically

Add some PHP to highlight different buttons

Add some PHP to check the status:

Pseudo Code= If (click occurred & button type = to Oneshot & button!=start)

Stop Refreshcounter

If (click occurred & button type = to Oneshot & button=start)

Refreshcounter=Refreshcounter++

Stop Refreshcounter

Elseif(click occurred & button type != to Oneshot)

Refreshcounter = refreshcounter

Elseif(No click and Refrehcounter running)

Refreshcounter=Refreshcounter++

Elsif(Noclick & Refreshcounter not running)

Refreshcounter =1

Endif

# GPIO

## Installing RPi.GPIO

First Run "sudo apt-get install python-dev" else it won’t install properly.

NOTE 1: The instructions here refer to an early version of RPi.GPIO. Please search the web for the latest version and replace the version numbers in the instructions below.

NOTE 2: If you have purchased Ciseco's[*4G Raspberry Wheezy SD card, configured for Ciseco products*](http://shop.ciseco.co.uk/4gb-wheezy-raspberry-pi-sd-card-configured-for-ciseco-products/), or used the Ciseco Wheezy image to create an SD card yourself, then the RPi.GPIO should already be installed and you can skip this step.

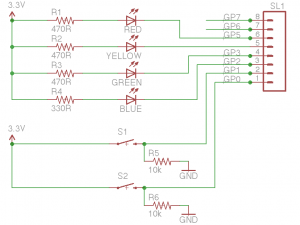
[RPi.GPIO](http://pypi.python.org/pypi/RPi.GPIO) is a small python library that take some of the complexity out of driving the GPIO pins, once install a single LED can be lit with 3 lines of python. Installing the library is almost as simple, either at a text console or using LXTerminal enter the following

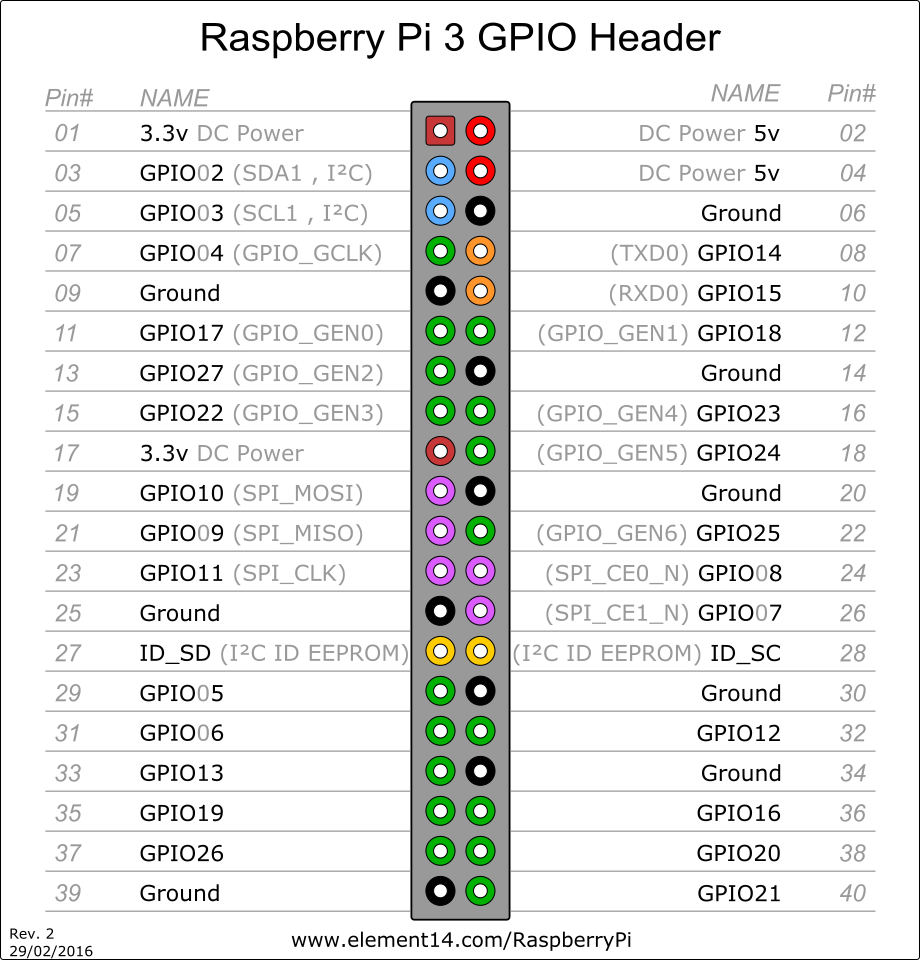
$ wget http://pypi.python.org/packages/source/R/RPi.GPIO/RPi.GPIO-0.6.2.tar.gz

$ tar zxf RPi.GPIO-0.6.2.tar.gz

$ cd RPi.GPIO-0.6.2

$ sudo python setup.py install





# Install PHP

sudo apt-get install apache2 php5

Test php

Create gpio.php and copy to var/www/html

Use wiringPi with PHP

sudo apt-get install git-core

sudo apt-get update

sudo apt-get upgrade

git clone git://git.drogon.net/wiringPi

cd wiringPi

git pull origin

cd wiringPi

./build

OR alternatively ….

Find: <https://git.drogon.net/?p=wiringPi;a=summary>

tar xfz wiringPi-98bcb20.tar.gz

cd wiringPi-98bcb20

./build

## Test Raspberry Pi PHP

Enable the execution of the php file using sudo visudo to add

www-data ALL=NOPASSWD: ALL

to the file.

Also, make sure that the /var/www directory belongs to the www-data user and group (use sudo chown -R www-data:www-data /var/www to set the correct owner).

Run the test program and verify that the LED is turned off and on.

Upgrade the Test html page to behave the same way as the gpio test.

NB. The pull/down resistors on the PI are software configurable so it is possible that the io is working but that LED is not visible due to a floating output state.

## Testing the MicroSwitch

Reference for the callback:

<http://raspberry.io/projects/view/reading-and-writing-from-gpio-ports-from-python/>

# Build the Remote Control

## Basic remote Data Acquisition

### IRLIB

https://learn.adafruit.com/using-an-infrared-library/hardware-needed

C:\Users\Dad\Desktop\CodeClub\EAT\IRLib-master

Download and compile up C:\Users\Dad\Desktop\CodeClub\EAT\IRLib-master\examples\IRrecord

### Circuit

IR transmitter LED + 220 ohm

IR Receiver

PSU

Red LED + 220 ohm resistor

Arduino

Microswitch

Green LED+ 220 ohm

### Code

Essentially IRecord with modifications:

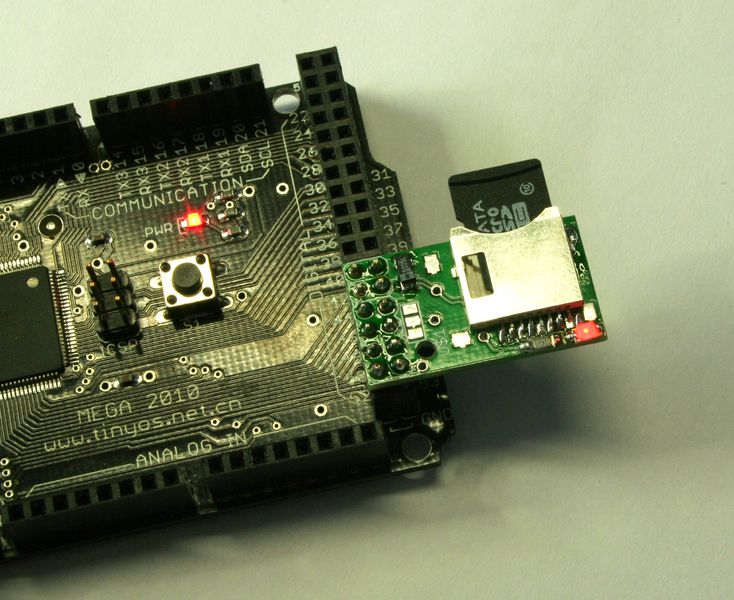
1. Red LED on record
2. Green LED on Send
3. RAW “unknown” transmitted at 40KHz
4. Unknown repeated 4 times
5. Delay of 75ms between repeats.
6. Send Code is activated by the microswitch instead of the Seial Monitor input.

### Testing

1. Play the chosen command
2. Replay the command using the Microswitch

## SDCard

### Connection and testing

[](http://reprap.org/mediawiki/images/9/90/Sd8.jpg)

http://reprap.org/wiki/File:Sdrampstest.zip

### Storing and reading

The communication between the microcontroller and the SD card uses [SPI](https://www.arduino.cc/en/Reference/SPI), which takes place on digital pins 11, 12, and 13 (on most Arduino boards) or 50, 51, and 52 (Arduino Mega). Additionally, another pin must be used to select the SD card. This can be the hardware SS pin - pin 10 (on most Arduino boards) or pin 53 (on the Mega) - or another pin specified in the call to SD.begin(). **Note that even if you don't use the hardware SS pin, it must be left as an output or the SD library won't work.**

You’ll need a SD reader and computer to format your card. The library supports the FAT16 and FAT32 filesystems, but use FAT16 when possible. The process to format is fairly straightforward

SPI connections for memory.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Arduino / Genuino Board | MOSI | MISO | SCK | SS (slave) | SS (master) | Level |
| Uno or Duemilanove | 11 or ICSP-4 | 12 or ICSP-1 | 13 or ICSP-3 | 10 | - | 5V |
| Mega1280 or Mega2560 | 51 or ICSP-4 | 50 or ICSP-1 | 52 or ICSP-3 | 53 | - | 5V |
| Leonardo | ICSP-4 | ICSP-1 | ICSP-3 | - | - | 5V |
| Due | ICSP-4 | ICSP-1 | ICSP-3 | - | 4, 10, 52 | 3,3V |
| Zero | ICSP-4 | ICSP-1 | ICSP-3 | - | - | 3,3V |
| 101 | 11 or ICSP-4 | 12 or ICSP-1 | 13 or ICSP-3 | 10 | 10 | 3,3V |
| MKR1000 | 8 | 10 | 9 | - | - | 3,3V |

### Playing the instruction file

# Appendices

## Appendix 1 – References

**Arduino**

<https://cdn.sparkfun.com/assets/learn_tutorials/4/0/3/4A-ESP8266__AT_Instruction_Set__EN_v0.30.pdf>

<http://www.microtechnica.tv/support/manual/ESP8266_WiFi_Module_Quick_Start_Guide_v_1.0.4.pdf>

<http://linksprite.com/wiki/index.php5?title=Advanced_Sensors_Kit_for_Arduino>

http://www.instructables.com/id/How-to-control-your-TV-with-an-Arduino/step2/Receiving-the-Signals

**Pi**

<http://www.waveshare.com/wiki/7inch_HDMI_LCD_(C)>

Python Socket Server

<http://www.binarytides.com/python-socket-server-code-example/>

<http://www.binarytides.com/python-socket-programming-tutorial>

**GPIO**

<http://www.raspberry-pi-geek.com/Archive/2014/07/PHP-on-Raspberry-Pi>

## Appendix 2 – Example Code

### Arduino Example Code

#### Code SoftwareSerial AT Command TestEATv2 @4800 baud.

The receiver is testsocketserver.py – runs in terminal on RPi3

#include <SoftwareSerial.h>

SoftwareSerial esp2866(2,3);

void setup() {

// put your setup code here, to run once:

Serial.begin(4800);

esp2866.begin(4800);

}

void loop() {

// put your main code here, to run repeatedly:

if(esp2866.available())

{

Serial.write("ESP available");

while(esp2866.available())

{

char c =esp2866.read();

Serial.write(c);

}

}

if(Serial.available())

{

delay(1000);

String command="";

while(Serial.available())

{

command+= (char)Serial.read();

}

esp2866.println(command);

}

}

#### TestEATv2

#include "ESP2866ATcmd.h"

SoftwareSerial esp2866(2,3);

String jap;

boolean doneit = false;

String current\_ip;

void setup() {

// put your setup code here, to run once:

Serial.begin(4800);

esp2866.begin(4800);

esp2866.println("AT");

delay(1000);

}

void loop() {

if (doneit == false)

{

Serial.println("Reset the module");

ESP\_Reset();

Displayresponse();

doneit = true;

//delay(5000);

Serial.println("");

delay(2000);

Echo("AT+CWMODE=1");

Displayresponse();

delay(2000);

Echo("AT+CWLAP");

delay(8000);

Displayresponse();

//

delay(2000);

String jap;

jap="AT+CWJAP\_CUR=";

jap.concat("\"");

jap.concat("EAT\_PI");

jap.concat("\"");

jap.concat(",");

jap.concat("\"");

jap.concat("withcustard");

jap.concat("\"");

//Serial.println(jap);

Echo(jap);

Displayresponse();

Echo("AT+CIFSR");

Displayresponse();

Echo("AT+CIFSR");

// Displayresponse();

current\_ip = GetIP();

////////////////////////////

Echo("AT+PING=\"192.168.1.1\"");

Displayresponse();

////////////////////////////

ESP\_SEND();

Displayresponse();

}

// put your main code here, to run repeatedly:

if(esp2866.available()>0)

{

Serial.println("ESP available");

Serial.println(".............");

delay(50);

while(esp2866.available())

{

delay(10);

char c =esp2866.read();

// Serial.print(c);

Serial.write(c);

}

}

if(Serial.available())

{

delay(1000);

String command="";

while(Serial.available())

{

command+= (char)Serial.read();

}

esp2866.flush();

esp2866.println(command);

}

//SetupAP

//AT+CWMODE=3

//Diagnostic

//Firmware version AT+GMR

//List Access Points AT+CWLAP

//Get IP Address AT+CIFSR

//WiFi Mode AT+CWMODE?

//TCP/UDP Connections AT+ CIPMUX?

//Check IP address AT+CIFSR

//Baud Rate\* AT+CIOBAUD? Supported: 9600, 19200, ....not yet supported

//Join Access Point AT+CWJAP\_CUR="EAT\_PI", "withcustard"

//Leave Access Point AT+CWQAP

ESP8266 Library

#include "ESP2866ATcmd.h"

boolean ESP\_SEND()

{

//ESP\_IpConfig();

//ESP\_SendData(1, "192.168.1.223", 23,"Hello Mum");

ESP\_SendData(1, "192.168.1.1", 5000,"Hello Mummy");

ESP\_TcpUdpClose();

}

boolean ESP\_SendData(int modeUDPTCP,String ip,int port, String data)

{

boolean result = false;

result = ESP\_MultiConStart();

result = ESP\_TcpUdpConnect(modeUDPTCP, ip, port);

String cmd;

cmd.concat("AT+CIPSEND=");

cmd.concat(data.length());

//try

Echo("AT+CIPMODE=0");

Displayresponse();

Echo(cmd);

//esp2866.println(cmd);

Displayresponse();

bool found;

unsigned long start;

// while (millis()-start<5000)

// {

// if(esp2866.find(">")==true )

// {

// found = true;

// break;

// }

// }

// if(found)

//esp2866.print(">");

esp2866.print(data);

// else

// {

// ESP\_TcpUdpClose();

// return false;

// }

String responsedata;

start = millis();

while (millis()-start<5000)

{

if(esp2866.available()>0)

{

char a =esp2866.read();

responsedata=responsedata+a;

}

if (responsedata.indexOf("SEND OK")!=-1)

{

return true;

}

}

return false;

}

boolean ESP\_MultiConStart(void)

{

Echo("AT+CIPMUX=0");

Displayresponse();

}

//

//

boolean ESP\_TcpUdpConnect(byte type, String addr, int port)

{

//esp2866.flush();

//esp2866.print("AT+CIPSTART=");

String cmd;

cmd="AT+CIPSTART=";

if(type>0)

{

// esp2866.print("\"TCP\"");

cmd.concat("\"");

cmd.concat("TCP");

cmd.concat("\"");

cmd.concat(",");

cmd.concat("\"");

cmd.concat(addr);

cmd.concat("\"");

cmd.concat(",");

cmd.concat(port);

}else

{

//esp2866.print("\"UDP\"");

cmd.concat("\"");

cmd.concat("UDP");

cmd.concat("\"");

cmd.concat(",");

cmd.concat("\"");

cmd.concat(addr);

cmd.concat("\"");

cmd.concat(",");

cmd.concat(port);

cmd.concat(",");

cmd.concat(11);

cmd.concat(",");

cmd.concat(0);//transparent transmission

}

Echo(cmd);

Displayresponse();

Echo("AT+CIPSTATUS");

Displayresponse();

return true;

}

void ESP\_Reset()

{

// int loopi;

// loopi=0;

// do{

//Serial.print(".");

// if(esp2866.available())

// {

esp2866.println("AT+RST");

unsigned long start;

start = millis();

while(millis() - start < 5000)

{

if(esp2866.find("OK") == true)

{

Serial.println("Reset OK");

break;

}

}

// }

delay(100);

Serial.println("======= Reset Complete =========");

}

String GetIP()

{

String ip\_data;

do

{

unsigned long start;

start = millis();

while (millis()-start<3000)

{

Serial.println("Reading IP");

while(esp2866.available()>0)

{

char a =esp2866.read();

ip\_data=ip\_data+a;

// Serial.println(ip\_data);

}

if (ip\_data.indexOf("AT+CIFSR")!=-1)

break;

}

if(ip\_data.indexOf(".") != -1)

{

break;

}

ip\_data = "";

}while(1);

char head[4] = {0x0D,0x0A};

char tail[7] = {0x0D,0x0D,0x0A};

ip\_data.replace("AT+CIFSR","");

ip\_data.replace(tail,"");

ip\_data.replace(head,"");

Serial.println("== == == == == ==");

Serial.print(ip\_data);

ip\_data.replace("+CIFSR:STAIP,","");

ip\_data.remove(ip\_data.indexOf("+CIFSR:STAMAC"));

Serial.println("== == == == == ==");

Serial.print(ip\_data);

Serial.println();

return ip\_data;

}

void Displayresponse()

{

Serial.println("==== Response ====");

Serial.println("");

while(esp2866.available()>0)

{

delay(80);

char c =esp2866.read();

Serial.write(c);

}

Serial.println("");

Serial.println("==================");

}

void Echo(String cmd)

{

Serial.println("Send Command");

Serial.println(cmd);

esp2866.flush();

esp2866.println(cmd);

delay(4000);

// unsigned long start;

// while(millis() - start < 5000)

// {

// if(esp2866.find("OK") == true)

// {

// Serial.print("command executed");

// break;

// }

// }

}

//turn off TCP or UDP

void ESP\_TcpUdpClose(void)

{

esp2866.flush();

esp2866.println("AT+CIPCLOSE");

String data;

unsigned long start;

start = millis();

while (millis()-start<3000)

{

if(esp2866.available()>0)

{

char a =esp2866.read();

data=data+a;

}

if (data.indexOf("Linked")!=-1 || data.indexOf("ERROR")!=-1 || data.indexOf("we must restart")!=-1)

break;

}

}

Header Library Header

#include <Arduino.h>

#include <SoftwareSerial.h>

extern SoftwareSerial esp2866;

//extern String ip\_data;

extern void Displayresponse();

extern void ESP\_Reset();

extern void Echo(String cmd);

extern String GetIP();

extern boolean ESP\_MultiConStart();

extern boolean ESP\_TcpUdpConnect(byte type, String addr, int port);

extern boolean ESP\_SEND();

//extern boolean ESP\_SendData(String data);

extern boolean ESP\_SendData(int modeUDPTCP,String ip,int port, String data);

extern void ESP\_TcpUdpClose(void);

## Appendix 3

### Python Socket Server Works with TestEATv2”

import socket

import sys

HOST = ''   # Symbolic name meaning all available interfaces

PORT = 5000 # Arbitrary non-privileged port

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

print 'Socket created'

try:

    s.bind((HOST, PORT))

except socket.error , msg:

    print 'Bind failed. Error Code : ' + str(msg[0]) + ' Message ' + msg[1]

    sys.exit()

print 'Socket bind complete'

s.listen(10)

print 'Socket now listening'

#now keep talking with the client

while 1:

    #wait to accept a connection - blocking call

    conn, addr = s.accept()

    print 'Connected with ' + addr[0] + ':' + str(addr[1])

    data = conn.recv(1024)

    reply = 'OK...' + data

    if not data:

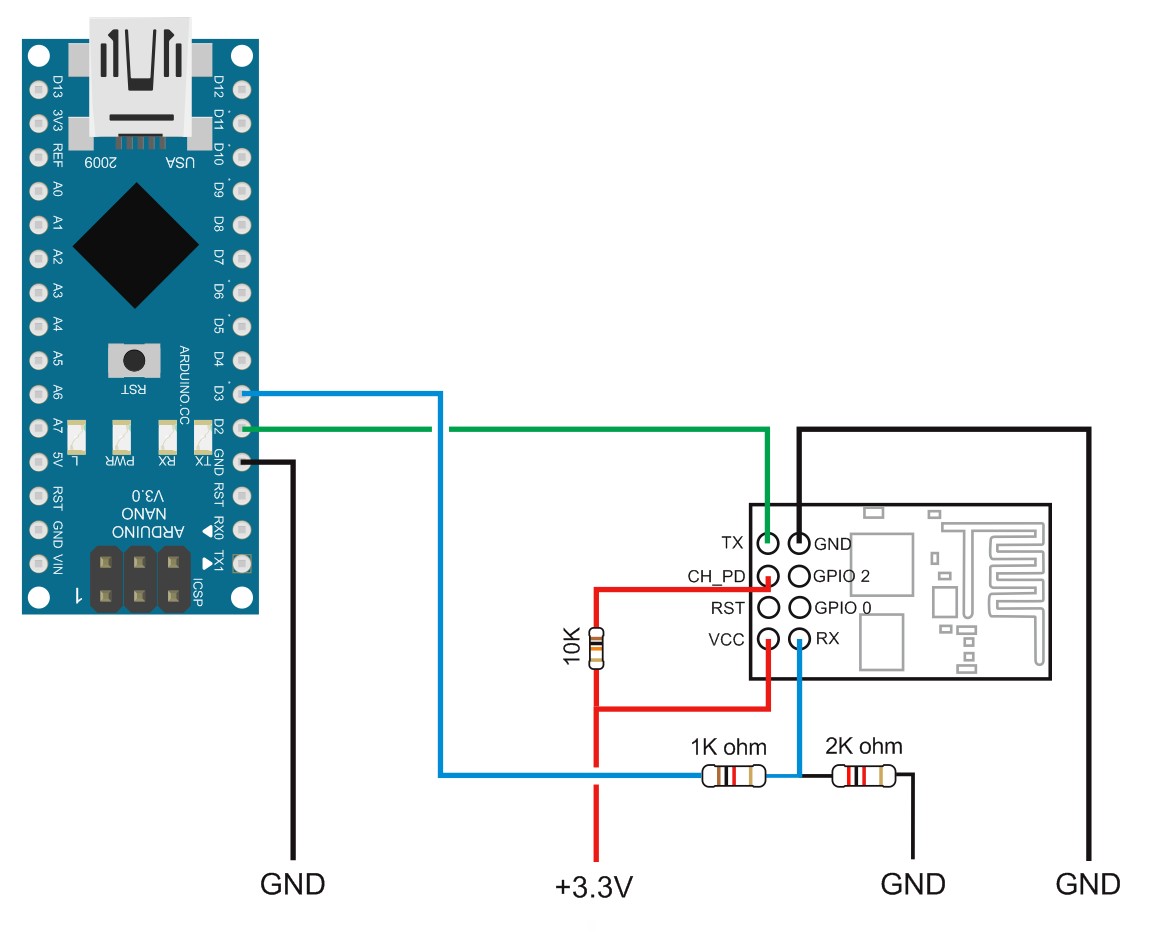
        break

    conn.sendall(reply)

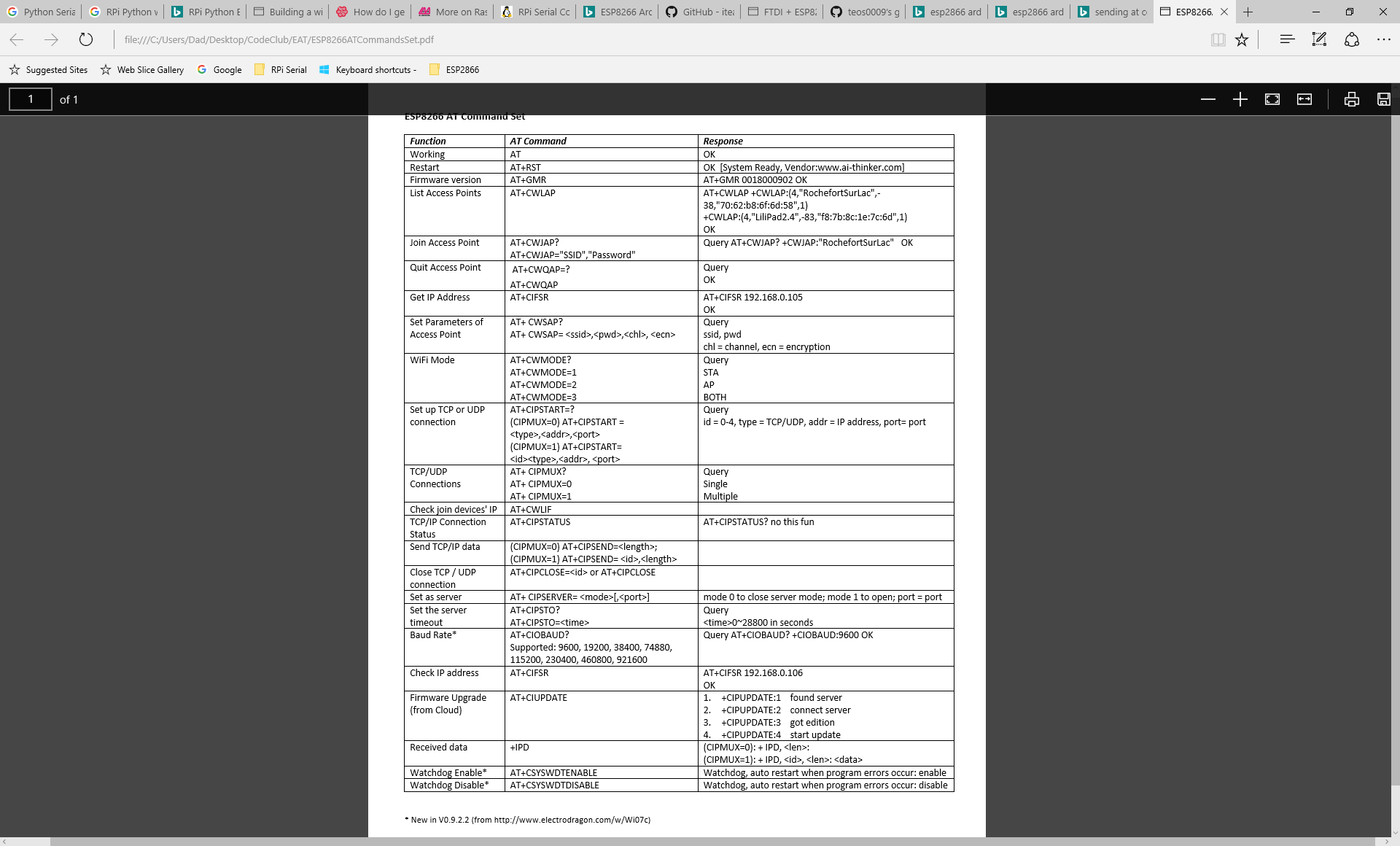
conn.close()

s.close()

## Appendix 3 - ESP Connections



## Appendix 4 - ESP Command Set



Configure LCD in Config

max\_usb\_current=1

hdmi\_group=2

hdmi\_mode=1

hdmi\_mode=87

hdmi\_cvt 1024 600 60 6 0 0 0

## Appendix 5 - Raspberry Pi PHP

1 <html>

02 <head>

03 <meta name="viewport" content="width=device-width" />

04 <title>LED Control</title>

05 </head>

06 <body>

07 LED Control:

08 <form method="get" action="gpio.php">

09 <input type="submit" value="ON" name="on">

10 <input type="submit" value="OFF" name="off">

11 </form>

12 <?php

13 $setmode17 = shell\_exec("/usr/local/bin/gpio -g mode 17 out");

14 if(isset($\_GET['on'])){

15 $gpio\_on = shell\_exec("/usr/local/bin/gpio -g write 17 1");

16 echo "LED is on";

17 }

18 else if(isset($\_GET['off'])){

19 $gpio\_off = shell\_exec("/usr/local/bin/gpio -g write 17 0");

20 echo "LED is off";

21 }

22 ?>

23 </body>

24 </html>

## Appendix 6 - Backup RPi

Cloning the SD card is simple. Just follow these steps:

1. Get everything set up just the way you want it on your Raspberry Pi, whatever you're using it for. Then shut down the Pi and remove the SD card. Insert the SD card into your computer.
2. Start up [Win32DiskImager](http://sourceforge.net/projects/win32diskimager/), a program that you probably have from [when you first set up your Pi](http://lifehacker.com/5976912/a-beginners-guide-to-diying-with-the-raspberry-pi). (If you're on OS X or Linux, you'll have to [use the dd command as described here](http://raspberrypi.stackexchange.com/questions/311/how-do-i-backup-my-raspberry-pi) instead of these steps).
3. In the "Image File" box, enter the path of your soon-to-be image file. For example, I put mine in C:\Users\Whitson\images\myraspbmc.img
4. Under the "Device" box, select your SD card.
5. Click the "Read" button to create the image file from your card.
6. When it's done creating the image file, you can eject your SD card and put it back in your Raspberry Pi. Keep that IMG file in a safe place.

Now, if anything ever goes wrong with your Pi, you can restore your fully-set-up image using the reverse instructions:

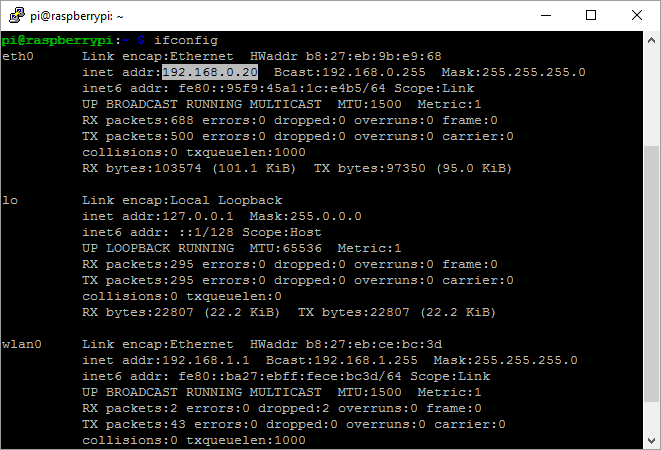
1. Insert the SD card back into your computer.
2. Head to the start menu or screen and type "disk management." Open the disk management program and find your SD card in the list.
3. Right-click and delete all the partitions on your SD card. When it's empty, right-click on it and format it (it doesn't matter what filesystem you format it to, your computer just needs to recognize it).
4. Open Win32DiskImager again and browse for your image file. Select your device from the Device dropdown just as you did before.
5. This time, click "Write" to write the image to the SD card.
6. When it finishes, eject the SD card and re-insert it into your Raspberry Pi. When you boot it up, it should be in the exact same state it was in when you first cloned the SD card.

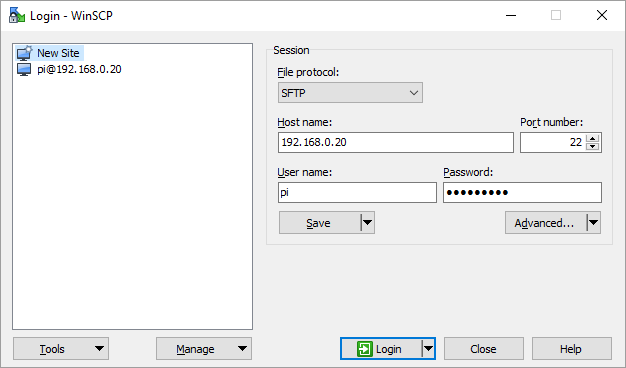
Copy File

## Using PUTTY & WinSCP

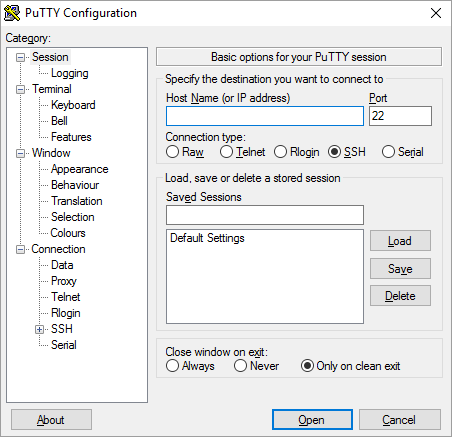
Connect Ethernet cable ( and check for Link & TX lights)

Find the Ip address on the network: run ifconfig at the command prompt in Terminal window on Pi.





Use the same address for PuTTY



# TODO

## Function Adapters

### IR LED driver

### Micro Test Switch

### LED feed back.

* 1. On and running – green
  2. Transmission - red