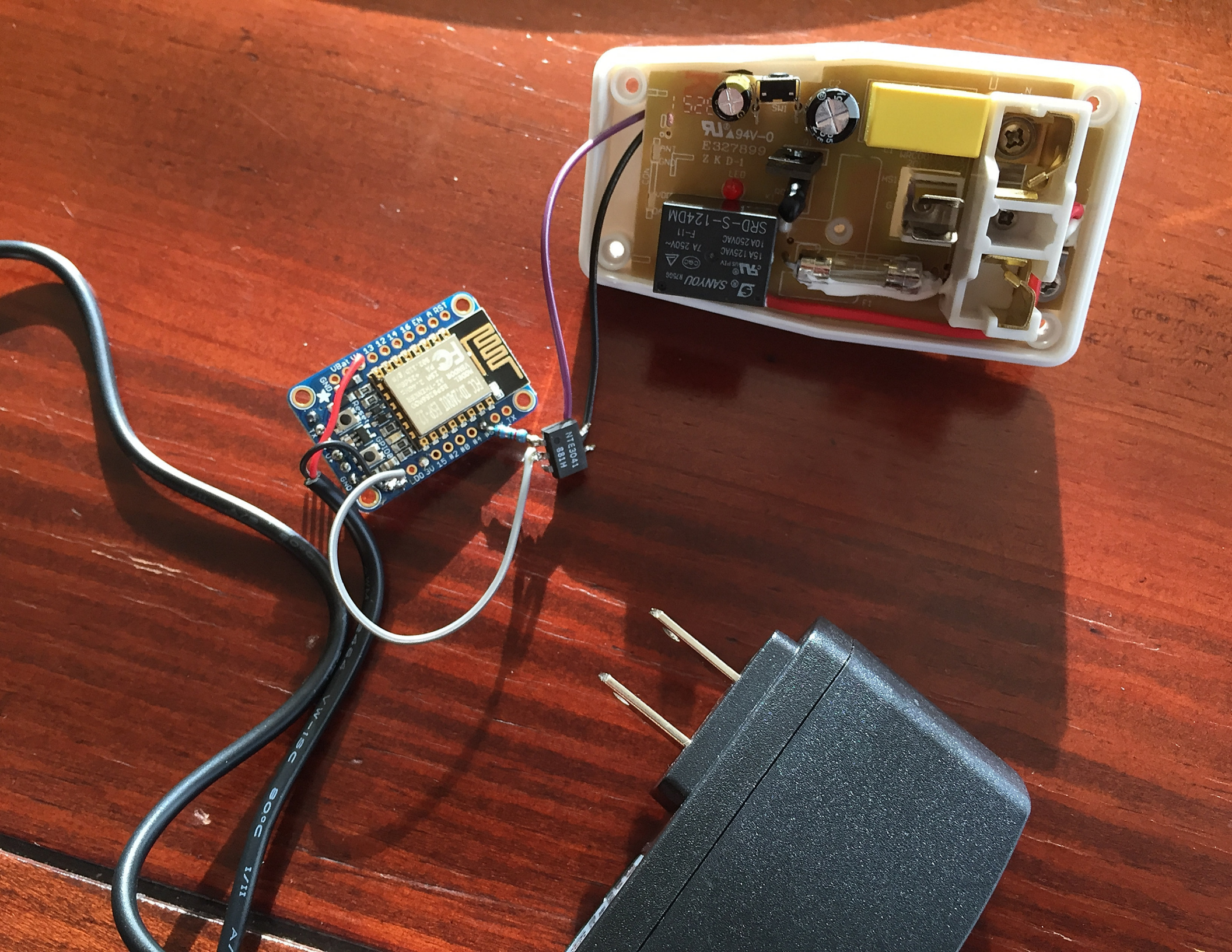
94V-0
E327899
ZKD-1
SRD-S-124DM
SANYOU
NTE3041

```
/*
************************************
*  Module 2                        *
*  ========                        *
*   By Roy H Guerra Jr.            *
*   3/4/17                         *
************************************
```

This code uses an Adafruit ESP8266 "HUZZAH" to control 120VAC modules to turn on/off appliances. The program uses the Blynk iPhone Application and Arduino Library. The results are transferred to a cloud server at "Blynk". The Blynk app is downloaded to your phone where the parameters are displayed. In addition, for trending puposes, a graphing function may be used, and you can send push notifications and E Mails using the Blynk App.

The AC switching modules were bought at Costco and reverse Engineered. Non critical components were removed (refer to photo).

This code also uses a Wemo clone library that can interface with the Amazon Alexa device. In addition, a Wi-Fi Library has been added that creates an interface where the Wi-Fi parameters can be entered via a mobile device or laptop, etc. Device boots up as an "AP" (Access Point), and once connected to the network, turns into a "Wi-Fi Station". If the GUI does not appear, type 192.168.4.1 on any browser to enter Wi-Fi credentials.

Note- The (OTA) Over The Air library is optional. This enables programming the ESP8266 without harware cables via Wi-Fi. The ESP8266 hardware configuration needs to be compatible to enable this feature (Not all ESP8266 boards are the same). Also ensure that the latest ESP8266 board hardware manager is installed.

Note- Without feedback, the operation is "Asynchronous". This is very rare, but If the virtual LED's
on the Blynk AP become out of sync with the operation, place module iin manual mode and turn on/off
two times and it will re-sync. This only happens on loss of power when the module is on.

Two main library links are listed below.
https://github.com/witnessmenow/esp8266-alexa-wemo-emulator
https://github.com/tzapu/WiFiManager

For IFTTT control, use the Maker Channel with the following settings and another toggle software function:
 *   URL: http://blynk-cloud.com:8080/YOUR_TOKEN/VX   Substitute your own token and vitual pin
 *   Method: PUT
 *   Content type: application/json
 *   Body: ["1"]   Use 1 for ON, 0 for OFF

Place the ESP8266 HUZZAH in bootloader mode so you can program it. On the Adafruit ESP8266 board, press the "GPIO" button, and then the Reset” button and release the "Reset" button
and then the "GPIO button. After that, the red LED should dim and upload the code to the

board.
When it is finished uploading, open the Serial monitor, and reset the board (if desired).

Hardware Required:

- Adafruit ESP8266 Huzzah
- 2N7000 FET transistor(s)or optoisolators
- Modified modules from Costco (see external picture)
- MISC (solder, proto board, etc.)

The circuit has the following features:
----
0) Starts as an Access Point, and connects to network via a phone or laptop through a GUI.
1) Connects and Transmits / Receives through the house wireless router as a client server
2) Connects to the Blynk cloud server that could be pulled up anywhere on your phone.
3) Sends Notifications directly to your phone (opyional).
4) The "blue LED" lights when connected to WiFi.
5) The "red LED" blinks showing operation sampling.
6) Has a menu for manual operation or timer mode which can be set by the iPhone.
7) Has feedback from the internal led located in the module to indicate on/off (optinal).
8) Program a seperate module for each room.
9) Contains software flags to assist in asyncronous operations.
10) If module happens to be on and switched to the timer mode, the timer on will have no effect
    for on operation, but will shut the module off as long as the menu remains in timer mode.

11) Alexa can overrule module operation in timer mode (this was the way it was designed).

Note(s):
----
* First download the Blynk App on your phone, and set up your account and Project.
* Set debug to "true" to read WiFi information and coonection status.
* Once website and program are verified to run correctly, set debug to "false", and reprogram (optional)
* Put the ESP8266 board in bootloader mode so you can program it.
* Don’t put Blynk.virtualWrite and any other Blynk command inside void loop() the server connection
will get terminated.
* Call functions with intervals. For example, this SimpleTimer Library is a library for timed events.
* Avoid using long delays with delay() it may cause server connection breaks;
* Do not send more than 1 E-Mail and / or notification in less than 15 seconds, or connection will break.
* If you send more than 100 values per second you may cause Flood Error and your hardware will be
disconnected from the network server.
* Be careful sending a lot of Blynk.virtualWrite commands as most hardware is not very powerful (like ESP8266)
so it may not handle some requests.
*/

// Define Libraries

```
// ================
#include <ESP8266WiFi.h>
#include<BlynkSimpleEsp8266.h>
#include <DNSServer.h>
#include<ESP8266WebServer.h>
#include<WiFiManager.h>
#include<SimpleTimer.h>
#include<WiFiUdp.h>
#include<ArduinoOTA.h>
#include "WemoSwitch.h"
#include "WemoManager.h"
#include"CallbackFunction.h"

// Declare Variables
// =================
char auth[] = "[redacted]";  // Put your Auth Token here.
boolean debug_mode = true; // Choose "true" for degugging
boolean state = LOW;  // Sampling LED state change
boolean flag = 0;   // Module timer flag to denergize relay
boolean flag_1; // Menu program flag
boolean modulestate = 0; // On/off software logic flag (initial state is "0")
boolean butstate = 1; // Virtual manaul button state, and initial condition (!1=0 to
start)
// SwitchReset = true; // Uncomment if using a manual switch on the module
int i = 0; // Counter variable

// Declare Defines
```

```
// ===============
//#define ledind 6  // (optional led used for feedback, may need more coding)(uncoment if
you want feedback)
#define toggleswitch  5  // GPIO pin we are connected to (optional for manual switch)
#define SERIAL_BAUDRATE 115200  // Serial baud rate
#define relayout 4  // ESP8266 output pin number
#define red_led  0  // ESP8266 red led onboard connection
#define blue_led 2  // ESP8266 blue led onboard connection

// ON / OFF Callbacks
// ==================
void switch3On();  // Set up "on" call function
void switch3Off(); // Set up "off" call function

WemoManager wemoManager;  // Invoke library function
WemoSwitch *switch3 = NULL;

WidgetLED led1(V5); // Module ON LED
WidgetLED led2(V6); // Module OFF LED

SimpleTimer timer; // Create a Timer case

BLYNK_WRITE(V7) { // Read Menu Mode (virtual)
  switch (param.asInt()){
    case 1: // Manual Mode
      if (debug_mode == true){ // Print on Serial
        Serial.println(F("Manual Mode"));
```

```
      }
      flag_1 = 1; // Set state of Menu flag to manual
      break;
    case 2: // Timer Mode
      if (debug_mode == true){ // Print on Serial
        Serial.println(F("Timer Mode"));
      }
      flag_1 = 0; // Set state of Menu flag to auto
      break;
   }
}

BLYNK_WRITE(V4){  // Read manual push button routine (virtual)
  int onbutt = param.asInt(); // Create a local variable to store virtual button value
    if ((onbutt == 1)&&(flag_1 == 1)){  // Only operate in manual mode
      // Relay on/off pulse function (will not need if using a bistable device)
      butstate = !butstate;  // Toggle the software flag
      switch (butstate){ // changed from buttstate add to logic?????
        case 0:
          switch3On(); // Goto this function
        break;
        case 1:
         switch3Off(); // Goto this function
        break;
      }
   }
}
```

```
BLYNK_WRITE(V8){ // Read virtual Module Timer (on/off)
  // You'll get HIGH/1 at startTime and LOW/0 at stopTime.  This method will be triggered
every day
  int autoon = param.asInt(); // Create a local variable to store virtual button value
    if ((autoon == 1)&&(flag_1 == 0)&&(modulestate == 0)){  // Check to see if menu was
in auto operation and timer has operated
      switch3On(); // Goto this function
    }
    if ((autoon == 0)&&(flag_1 == 0)&&(modulestate == 1)){  // Check to see if menu was
in auto operation and timer has stopped
      switch3Off();  // Goto this function
    }
}

void moduleoff(){  // Turns off module relay after 1 second (simulates a momentary
pushbutton)
  if (flag == 1){
    i++;  // Count to 1 second (which is 100 times)
  }
  if (i >= 100){ // When count = 100, this is 1 second
  digitalWrite(relayout, LOW);  // Turn off Relay if previusly "on"
  i = 0; // Reset counting variable
  flag = 0; // Reset operate flag
  }
}
```

```
void ledsense(){ // Determine module state (on/off)and change virtual LED status
  state =!state;  // Logic used to blink ESP8266 red LED to show circuit is working and
sampling data
  digitalWrite(red_led, state); // RED ESP8266 Board LED
 // boolean ledvalue = digitalRead(ledind);  // See if module led is "on" or "off"
(uncoment if you want feedback, and adjust code accordingly)
   //if (ledvalue == HIGH){  // Module LED is off (uncoment if you want feedback)
     //led1.off(); // Turn off Widget LED (uncoment if you want feedback)
     //led2.on(); // Turn on Widget LED (uncoment if you want feedback)
   //}
   //if (ledvalue == LOW){ // Module LED is on (uncoment if you want feedback)
    //led1.on(); // Turn on Widget LED (uncoment if you want feedback)
    //led2.off(); // Turn off Widget LED (uncoment if you want feedback)
  //}
}

void setup(){  // Setup initialization
  pinMode(red_led, OUTPUT); // Set on board red LED as an output to circuit is running
(it flashes)
  pinMode(blue_led, OUTPUT); // Set on board blue LED as an output to indicate
satisfactory Wi-Fi connection
  pinMode(relayout, OUTPUT); // Set GPIO as output to turn on relay
  digitalWrite(relayout, LOW);  // Ensure relay is off during power up
  pinMode(toggleswitch,INPUT_PULLUP); // Set interal pullup resistor (optional, used for
manual switch)
  digitalWrite(blue_led, HIGH);  // Turn off blue LED
  Serial.begin(SERIAL_BAUDRATE); // See the connection status in Serial Monitor
```

```
   if (debug_mode == true){ // Print on Serial
    Serial.println(F("Alexa Demo Sketch"));
    Serial.println(F("After connection, ask Alexa/Echo to 'turn device 'on' or 'off'"));
    Serial.println("");
    }
  WiFiManager wifi;  // Start an AP, and a GUI to enter Wi-Fi information if one already
does not exist
  //wifi.resetSettings(); // Uncomment to reset saved wi-fi settings
 wifi.autoConnect("Module_2");
    if (debug_mode == true){ // Print on Serial
    Serial.println(F("Wi-Fi Connected to "));  // Display message
    Serial.println("");
    }
  Blynk.config(auth); // Make the cloud server connection
  ArduinoOTA.begin();
  digitalWrite(blue_led, LOW);  // Turn on BLUE LED to show Wi-Fi and Blynk server
connection
  timer.setInterval(10L, moduleoff); // Setup a counting function to be called every 10mS
  //timer.setInterval(100, ButtonCheck);  // Uncomment to setup a manual button check for
evey 100mS (if used)
  timer.setInterval(1000L, ledsense); // Setup a function to be called every second
  wemoManager.begin();  // Start wemo library emulator
// Format: Alexa invocation name, local port no, on callback, off callback
//----------------------------------------------------------------------
  switch3 = new WemoSwitch("Module two", 82, switch3On, switch3Off);  // Bump port by 1
for next device and change to switch3, etc
 wemoManager.addDevice(*switch3);
```

```
  led1.off(); // Turn off Widget LED for initial conditions
  led2.on(); // Turn on Widget LED  for initial conditions
}

BLYNK_CONNECTED() {
  Blynk.syncVirtual(V14); // Sync the menu status to last value stored on server.
}

void loop() { // Main Loop (can not use the delay function here! See notes above!)
  Blynk.run();
  timer.run();
  wemoManager.serverLoop();  // Enable server to listen for Alexa commands
  ArduinoOTA.handle();  // Enable server handler for OTA programming
}

void switch3On() {  // Function to turn the relay on
    if (modulestate == 0){  // Ensure module is off before turning on (keeps in sync)
      led1.on(); // Turn on Widget LED
      led2.off(); // Turn off Widget LED
      digitalWrite(relayout, HIGH); // Comment out line if using a bistable device
      flag = 1; // // Comment out line if using a bistable device
     if (debug_mode == true){ // Print on Serial
       Serial.println(F("Switch 1 turn on ..."));
      }
     }
    //digitalWrite(relayout, HIGH); // Uncomment line if using a bistable device
    modulestate = 1;  // Software flag set to 1
```

```
    butstate = 0; // Set manual button state so it syncs with Alexa
}

void switch3Off() { // Function to turn the relay off
    if (modulestate == 1){  // Ensure module is on before turning off (keeps in sync)
      led1.off(); // Turn off Widget LED
      led2.on(); // Turn on Widget LED
      digitalWrite(relayout, HIGH); // Comment out line if using a bistable device
      flag = 1; // // Comment out line if using a bistable device
     if (debug_mode == true){ // Print on Serial
       Serial.println(F("Switch 1 turn off ..."));
     }
    }
    //digitalWrite(relayout, LOW); // uncomment line if using a bistable device
    modulestate = 0;  // Software flag set to 0
    butstate = 1; // Set manual button state so it syncs with Alexa
}

/*
Uncomment, and fix code for using a manual switch on the module

void ButtonCheck(){
  // look for new button press (logic = false for a "press")
  boolean SwitchState = (digitalRead(TacSwitch));

  // toggle the switch if there's a new button press
  if (!SwitchState && SwitchReset == true){
```

```
    Serial.println("Hardware switch activated");
    if (modulestate == 1){
      switch2Off();
    }
    else{
      switch2On();
    }

    // Flag that indicates the physical button hasn't been released
    SwitchReset = false;
    delay(50);            //debounce
  }
  else if (SwitchState == 1){
    // reset flag the physical button release
    SwitchReset = true;
  }
}
*/

//*******************************************************

/*
Uncomment, and fix code and Blynk app for use with IFTTT
BLYNK_WRITE(VPIN){
  int SwitchStatus = param.asInt();

  Serial.println(F("Blynk switch activated"));
```

```
  // For use with IFTTT, toggle the relay by sending a "2"
  if (SwitchStatus == 2){
    ToggleRelay();  // This routine was moved to the "manual switch"
  }
  else if (SwitchStatus == 1){
    switch2On();
  }
  else
  switch2Off();
}
*/
```