```
/*
Laboratorio Integral de Electrónica
Proyecto: Cargador

Integrantes:
Sergio Ramírez Zaldivar
Oscar Alejandro Torres Maya
Daniel Córdova Bermúdez
David Alejandro Nicolás Palos

Descripción: Alternativa digital de proyecto, cargador inteligente capáz de medir el nivel de carga de un dispositivo Android/Iphone a través de una aplicación móvil/Asistente digital Alexa y desconectar
            la alimentción para prevenir sobrecargar y alargar la vida útil de la batería y el cargador.
            Cuenta también con una medición analógica de corriente, con la cual se puede determinar de manera aproximada el nivel de carga del dispositivo conectado, de esta manera
            se puede utilizar no solo con dispositivos Android y Iphone pero cualquier batería con su respectivo cargador.

*/



//Librerias utilizadas para la conexión WiFi al dispositivo móvil
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include<BlynkSimpleEsp8266.h>
#ifdef ARDUINO_ARCH_ESP32
#include <WiFi.h>
#else
#include <ESP8266WiFi.h>
#endif
#include <Espalexa.h>

// Pines de los sensores y actuadores conectados al microcontrolador
int led = LED_BUILTIN;
int relay = 16;
int sens = A0;

// Código de autenticación de BLYNK, la aplicación móvil
char auth[] = "lValXIwA-t6W8InVcv0Kjd2ssrVsPO38";

// Variables a utilizar como el nivel de batería, la corriente actual y el modo de operación
```

```
int modo = 1;
int bateria = 0;
int corriente = 0;

// SSID y contraseña de la red WIFI a utilizar
char ssid[] = "IZZ2_4";
char pass[] = "1952436502";
const char* password = "1952436502";


Espalexa alexa;// Definición de la libreria Espalexa con el nombre "alexa"

void alwaysON(uint8_t brightness);//Define el dispositivo alwaysON y le pone como
única variable la "Intensidad" que posteriormente será utilizada para encender y
apagar el cargador (0% y 100%)
void smartCharge(uint8_t brightness);//Define el dispositivo smartCharge y le pone
como única variable la "Intensidad" que posteriormente será utilizada para encender
y apagar el cargador (0% y 100%)

void setup()
{
  Serial.begin(9600); //Inicializa la comuncación serial a 9600 bps para poder
observar el estatus del dispositivo en la computadora

  pinMode(led, OUTPUT); //Inicializa el pin led como salida
  pinMode(relay,OUTPUT);//Inicializa el relay led como salida
  pinMode(sens,INPUT);//Inicializa el sens led como entrada

  ConectarWifi(); //hace uso de la función conectarWifi definida posteriormente,
para realizar el protocolo de conexión WiFi con los datos definidos arriba
  alexa.addDevice("Cargador",alwaysON); //Agrega el dispositivo definido como
alwaysON a los dispositivos a controlar, le asigna como nombre "Cargador"
  alexa.addDevice("Smart",smartCharge);//Agrega el dispositivo definido como
smartCharge a los dispositivos a controlar, le asigna como nombre "Smart"

  alexa.begin();//Inicializa el modo de compatiblidad con el asistente Alexa
  Blynk.begin(auth, ssid, pass);//Inicializa el modo de compatibilidad con la
aplicación móvil Blynk
  Blynk.virtualWrite(0,modo);//Fija el valor del pin virtual 0 de Blynk con el valor
de la variable modo
}

void loop()
{
  ConectarWifi();//hace uso de la función conectarWifi definida posteriormente, para
realizar el protocolo de conexión WiFi con los datos definidos arriba
```

```
  alexa.loop();//Realiza un ciclo completo de los comandos necesarios para
mantenerse pendiente de cualquier petición vía asistente
  Blynk.run();//Realiza un ciclo completo de los comandos necesarios para mantenerse
pendiente de cualquier petición vía Blynk aplicación móvil
  delay(1);//Delay necesario de acuerdo a los manuales de uso de las librerias.

 corriente = (analogRead(sens)-512)*(analogRead(sens)-512);//Realiza una medición
del pin A0, en el cual se encuentra el sensor de corriente
  /*
    De acuerdo a la hoja de especificaciones, el sensor otorga un valor de voltaje
de salida de 2.5V cuando la corriente es 0, de esta forma, la corriente negativa se
puede medir con valores
    menores a 2.5V, en el entorno de arduino el ADC otorga la salida máxima(1023) a
5V, por lo que a la señal de A0 se le restan 512 para obtener el verdadero valor de
lectura de la corriente.

     esto es necesario porque se está trabajando con corriente alterna, por lo que
en el semiciclo negativo del voltaje, obtendremos también una corriente negativa.

     Se eleva el valor al cuadrado para obtener solo valores positivos, sin este
método, no se puede realizar ningun promedio, pues el promedio de una señal senoidal
es 0.
  */


  for(int i =0; i<20;i++){ //Ciclo for que realiza 20 iteraciones agregandolas al
contador "corriente", realizando el mismo cálculo mencionado anteriormente.
    corriente = corriente + ((analogRead(sens)-512)*(analogRead(sens)-512));
  }
  corriente = corriente/20; //División del contador "corriente" entre 20 para
realizar un promedio de 20 muestreos.
  corriente = sqrt(corriente);//Se obtiene la raíz cuadrada de los valores
calculados anteriormente para poder hacer uso de este como medición de corriente.


  if(modo == 1){ //Si se encuentra en modo 1, se apaga el LED y el Relay para
interrumpir el circuito al cargador (Led activo en Bajo y elay activo en Alto).
    digitalWrite(led,HIGH);
    digitalWrite(relay,LOW);
  }else if(modo == 2){ //Si se encuentra en modo 2, se enciende el LED y el Relay
para reconectar el circuito al cargador (Led activo en Bajo y elay activo en Alto).
    digitalWrite(led,LOW);
    digitalWrite(relay,HIGH);
  }
  /*
    El modo 3 es el modo automático, se hace dos revisiones, el valor de la batería
```

del celular en caso de que la aplicación móvil se encuentre realizando acutalizaciones y el valor de la corriente

Para el caso del nivel de batería, si este llega a 100, se interrumpe el circuito y se pone el cargador en modo 0.
Para el caso del nivel de corriente, si este pasa por debajo del valor 30, se interrumpe también el circuito y se pone el cargdor en modo 0.

El valor 30, fue un valor obtenido experimentalmente, encontramos un ligero consumo de los cargadores aún cuando la batería se encuentra desconectada, el valor 30 fue entonces el
que más se aprocimó en los cargadores probados al nivel completo de carga.
*/
```
 else if(modo==3){
  if(bateria == 100){//Compara el nivel de batería, si se encuentra en 100 se realizan las instrucciones dentro del IF
    modo=1;//pasa al modo 1
    Blynk.virtualWrite(0,modo);//Escribe el valor del modo en el pin virtual 0 de blynk, de esta manera se puede saber en la aplicación cuando la carga ha terminado
   }
  else if(corriente<30){//Compara el nivel de corriente, si se encuentra debajo de 30 se realizan las instrucciones dentro del IF
    modo=1;//pasa al modo 1
    Blynk.virtualWrite(0,modo);//Escribe el valor del modo en el pin virtual 0 de blynk, de esta manera se puede saber en la aplicación cuando la carga ha terminado
   }
   else{
  digitalWrite(LED_BUILTIN,LOW);//Si no se cumplen ninguna de las condiciones anteriores, el cargador se mantiene encendido
  digitalWrite(relay,HIGH);
   }
 }
}

BLYNK_WRITE(0){ //Función de la libreria Blynk, con el cual se obtiene el valor del pin digital 0 y lo despliega en comunicación serial, herramienta utilizada para Troubleshooting
 modo = param.asInt();
 Serial.println(modo);
}
BLYNK_WRITE(16){ //Función de la libreria Blynk, con el cual se obtiene el valor de la carga de batería, lo asigna a la variable bateria
 bateria = param.asInt();
}

void ConectarWifi() {//Función utilizada para conectarse a wifi
```

```
  if (WiFi.status() != WL_CONNECTED) {//Si ya se encuentra conectado, omite los
siguientes procesos
    WiFi.mode(WIFI_STA);//Funcion de la libreria wifi para configurar el modo de
operación
    WiFi.begin(ssid, password);//Con los valores de SSID y password realizá el
protocolo de comunicación con la red
    Serial.println("");
    Serial.println("Connecting to WiFi");//despliega el mensaje en comunicación
serial
    while (WiFi.status() != WL_CONNECTED) {//Realiza parpadeos del LED mientras se
esta conectando
      digitalWrite(LED_BUILTIN, 1);
      delay(250);
      digitalWrite(LED_BUILTIN, 0);
      delay(250);
      Serial.print(".");
     }
    Serial.print("Connected to ");//Herramientras de trouble shooting, despliega el
nombre de la Red y la direccion IP que le fue asignada.
    Serial.println(ssid);
    Serial.print("IP address: ");
    Serial.println(WiFi.localIP());
  }
}

void alwaysON(uint8_t brightness) {//Funcion del dispositivo alwaysOn para controlar
el encendido y apagado del relay con el asistente digital Alexa y actualiza el
estado en la aplicación móvil
  if (brightness) { //Si el comando recibido desde alexa incluye un valor de
"Brillo", cambia al modo encendido
    modo = 2;
    Blynk.virtualWrite(0,2);//actualiza el estado en la aplicación
  }
  else {//Si el comando recibido desde alexa NO incluye un valor de "Brillo", cambia
al modo encendido
    modo = 1;
    Blynk.virtualWrite(0,1);//actualiza el estado en la aplicación
  }
}

void smartCharge(uint8_t brightness) {//Funcion del dispositivo smartCharge para
controlar el encendido y apagado del relay con el asistente digital Alexa y
actualiza el estado en la aplicación móvil
  if (brightness) {//Si el comando recibido desde alexa incluye un valor de
"Brillo", cambia al modo inteligente
    modo = 3;
```

```
   Blynk.virtualWrite(0,3);//actualiza el estado en la aplicación
  }
  else {//Si el comando recibido desde alexa NO incluye un valor de "Brillo", cambia
al modo inteligente
    modo = 1;
   Blynk.virtualWrite(0,1);//actualiza el estado en la aplicación
  }
}
```