# Fase Final - Splitter de Vídeo Compuesto*

Josue Eduardo Nimatuj Piedrasanta, 201632173,[1,**] Luis Ajkim, Tepaz Toj, 201630870,[***] and Eliezer Ajuchán Xovin, 200611470[****]

[1] *Facultad de Ingeniería, Laboratorio de comunicaciones 3, Universidad de San Carlos,Ciudad Universitaria, Zona 12, Guatemala.*

En el presente proyecto se realizó la implementación física de un circuito de audio y vídeo que en conjunto forman un splitter de vídeo compuesto, la implementación del proyecto se muestra detalladamente a continuación, así como detalles y explicación más relevante de los dispositivos y etapas en el circuito propuesto.

## I. OBJETIVOS

### A. General

- Realizar la implementación física de un Splitter de Video Compuesto.

### B. Especifico

* Aplicar los conceptos de vídeo y audio.

* Reforzar la teoría de multiplicación.

* Manejar adecuadamente el ruido en transmisiones, y corregir los posibles errores.

## II. MARCO TEÓRICO

### A. Amp TL084CN

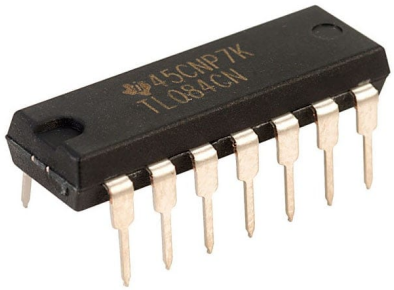


Figura 1: Amp. TL084CN

El Amplificador Operacional TL084CN DIP-14 es un circuito integrado (CI) de 14 pines que internamente esta compuesto por cuatro amplificadores operacionales de entrada JFET de alta velocidad que incorporan JFET de alto voltaje y transistores bipolares en un circuito integrado monolítico. Los cuatro amplificadores cuentan con altas velocidades de rotación, baja polarización de entrada y corrientes de compensación, y bajo coeficiente de temperatura de voltaje de compensación.

* Comunicaciones 3
** e-mail: @gmail.com
*** e-mail: ajkim00h@gmail.com
**** e-mail: aircuenta@gmail.com

### B. Optoacoplador PC817

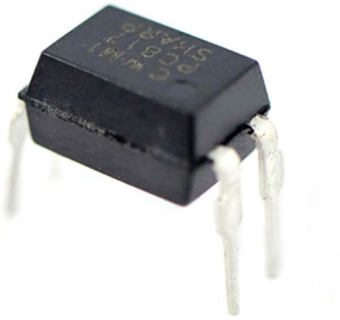

Figura 2: Optoacoplador PC817

El PC817 es un optoacoplador de propósito general que consiste en un diodo que emite luz infrarroja hacia un fototransistor, logrando excitar al transistor en forma óptica, cuenta con un encapsulado DIP-4 de un canal.

### C. Transistor BD140


Figura 3: Transistor BD140

Transistor bipolar compuesto por silicio con un amplio uso en diversos circuitos y aparatos electrónicos.

Consiste en una plaquita de semiconductor con tres regiones consecutivas de diferente tipo de conductibilidad eléctrica los cuales forman dos uniones p-n-p, con la característica de que las dos regiones extremas tienen un mismo tipo de conductibilidad, y la región intermedia posee otro tipo de conductibilidad. Estas regiones son llamadas emisor, colector y base.

### D. Microcontrolador FireBeetle ESP32 IoT

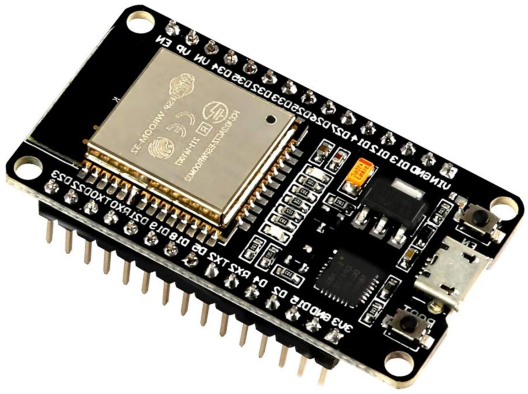

Figura 4: Microcontrolador FireBeetle ESP32 IoT

La tarjeta FireBeetle de DFRobot es un microcontrolador de bajo consumo diseñado intencionalmente para proyectos de Internet de las cosas (IoT). La FireBeetle integra un módulo ESP-WROOM-32 de doble núcleo, que admite MCU y comunicación de modo dual Wi-Fi y Bluetooth. La corriente eléctrica es de solo 10 $\mu$A en el modo de reposo profundo. El controlador principal admite dos métodos de suministro de energía: USB y batería de litio externa de 3.7V. Y tanto USB como DC externo pueden cargar la batería Lipo directamente.

FireBeetle Board-ESP32 ha realizado un diseño de hardware especial para Arduino IDE. Puede realizar una descarga sin cambiar el modo de arranque manualmente. Es compatible con Arduino, IDF (linux), micropython, etc. Además, FireBeetle hizo una asignación de pines para Arduino IDE. Se puede configurar con transporte Dx, compatible con Arduino UNO y reducir la barrera de entrada.

## III. COSTE DEL PROYECTO

En la siguiente tabla se da la lista de materiales con su respectivo precio y el monto total para la realizacion del proyecto:

| No | Producto | Precio |
|---|---|---|
| 1 | Microcontrolador Esp32 | Q120.00 |
| 2 | Amp UA741CN | Q7.5.00 |
| 2 | Amp TL084CN | Q12.00 |
| 8 | Capacitores $47\mu f$ | Q8.00 |
| 2 | Capacitores $1\mu f$ | Q2.00 |
| 2 | Capacitores $10\mu f$ | Q2.00 |
| 5 | Capacitores $100\mu f$ | Q5.00 |
| 3 | Potenciometros $2k\Omega$ | Q11.25 |
| 8 | Optoacopladores PC817 | Q32.00 |
| 1 | Transistor BD140 | Q2.75 |
| 1 | Resistencia 1W $1k\Omega$ | Q1.00 |
| 14 | Resistencia 1/4W $220\Omega$ | Q14.00 |
| 5 | Resistencia 1/4W $75\Omega$ | Q5.00 |
| 4 | Resistencia 1/4W $10k\Omega$ | Q4.00 |
| 6 | Resistencia 1/4W $100k\Omega$ | Q6.00 |
| 2 | mt. Cable UTP | Q6.00 |
| 6 | mt. Alambre Proto | Q16.50 |
| 1 | Pulsador Rojo 3A | Q8.00 |
| 15 | Conecotores RCA | Q30.00 |
| 1 | Impresion 3D | Q16.00 |
| | Total | Q309.00 |

Cuadro I: Presupuesto

## IV. RESULTADOS

### A. Diagrama de control de salidas Esp-32

Se presentó el funcionamiento del circuito que se planteó en la fase 1, con la diferencia de que a la salida negativa de cada conector están conectados a transistores que funcionan en corte o en saturación y que son controlados desde un servidor web a través del el microcontrolador ESP32.
Diagrama de control de salidas utilizando los pines digitales 32, 33, 4 y 2 del microcontrolador Esp32, se manda un pulso en alto para encender el led interno del Optoacoplador transmitiendo la señal por medio de luz al transistor poniendo en saturacion el transitor interno del optoacoplador permitiendo la conexion a negativo para la habilitacion y deshabilitacion de nuestras salidas de auido y video.

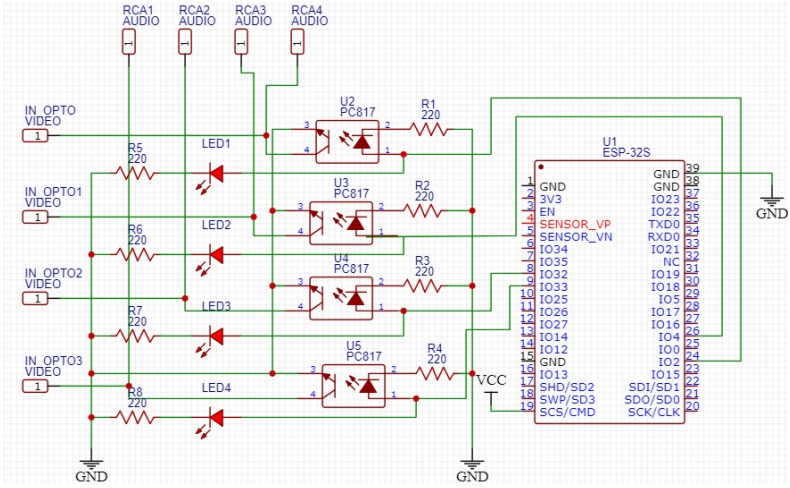


Figura 5: Diagrama Control de Salidas de AUDIO microcontrolador Esp-32

En el siguiente digrama se puede observar las salidas de los optoacopladores hacia otros optoacopladores del control de Auido que sirven para controlar la salida de la señal de video:

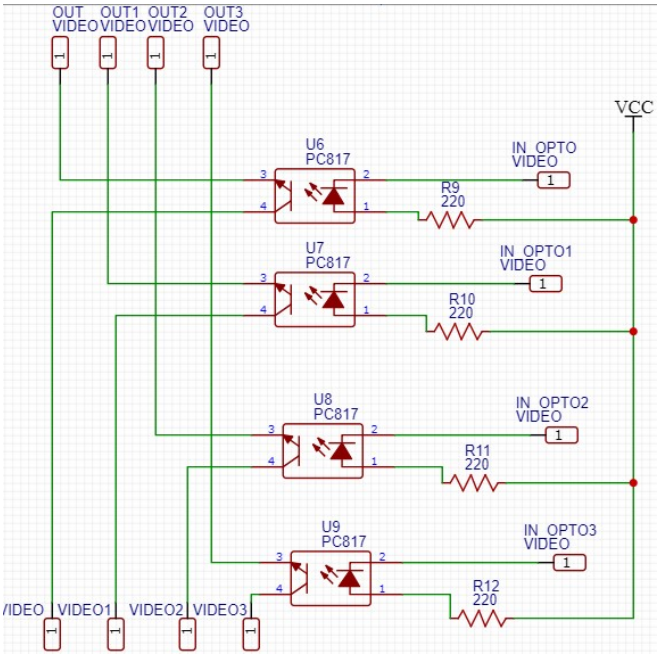


Figura 6: Diagrama Control de Salidas de VIDEO con microcontrolador Esp-32

### B. Servidor Web

Utilizamos lo que es un servidor Web asíncrono para controlar las salidas del Splitter, con la ayuda de la biblioteca ESPAsyncWebServer que proporciona una manera fácil de construir un servidor web

asincrónico en la IDE Arduino. Ventajas Servidor web asincrónico es que maneja mas de una conexión al mismo tiempo y trabaja en el envió de señales en segundo plano.

En el servidor web se implementaron 4 switchs que habilitan y deshabilitan los canales según lo requiramos. En esta etapa se presentó la parte de audio, y la parte de vídeo tendrá un funcionamiento similar a pesar de que usa un circuito de amplificación diferente.

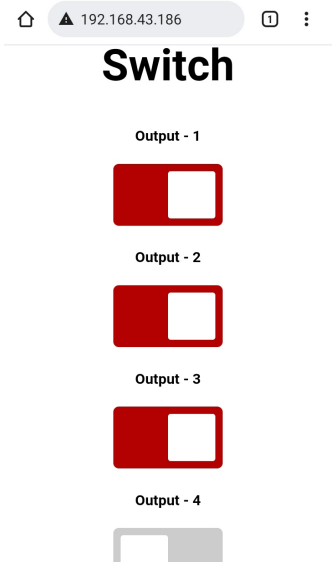


Figura 7: Interruptores digitales

Su funcionamiento basicamente es que cuando se alterna el estado de los botones en la pagina web. Por ejemplo el GPIO 2, se alterna el botón para encender GPIO 2. Cuando eso sucede, el navegador realiza una solicitud HTTP GET, en el/ update? output = 2 & state = 1URL. Según esa URL, el ESP cambia el estado de GPIO 2 a 1 ( ALTO ). Y si deseamos cambiar el estado de GPIO 2 a 0 ( LOW ), realizamos una solicitud HTTP GET en el/ update? output = 2 & state = 0 URL

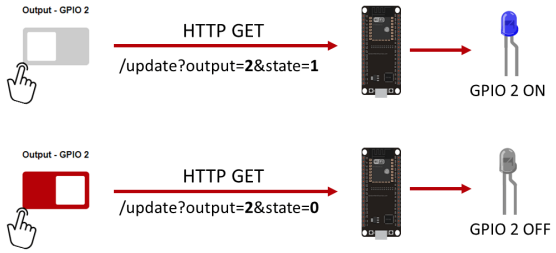


Figura 8: Funcionamiento de los botones

En el servidor web solo se hará necesario implementar 4 switchs, ya que cada uno controlará los 4 canales a la vez. Por lo que el código variará mínimamente con respecto al presentado en esta fase, a menos que hay algún cambio sustancial en la lógica.

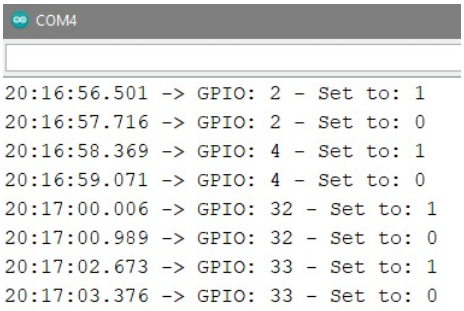


Figura 9: Estado de los Interruptores digitales en un monitor serial

### C. App Splitter

A continuacion se presenta la ralizacion de un simple App con el fin de un simplificar el manejo del Splitter de audio y video. Utilizando el entorno de desarrojo integrado de aplicaciones Web MIT App Inventor se crea una aplicacion sencilla en el cual nos conectamos automaticamente o manualmenta a la IP de nuestro servidor web el cual controla de manera inalambrica las salidas del Splitter:

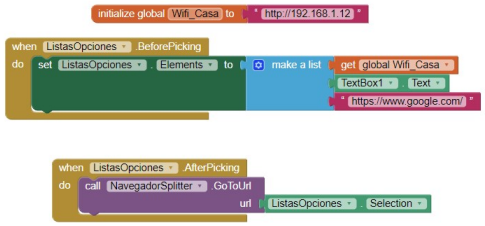


Figura 10: Diagram de BLoques App Inventor

La direccion IP registrada depende le asige el Router al microcontrolador Esp-32.

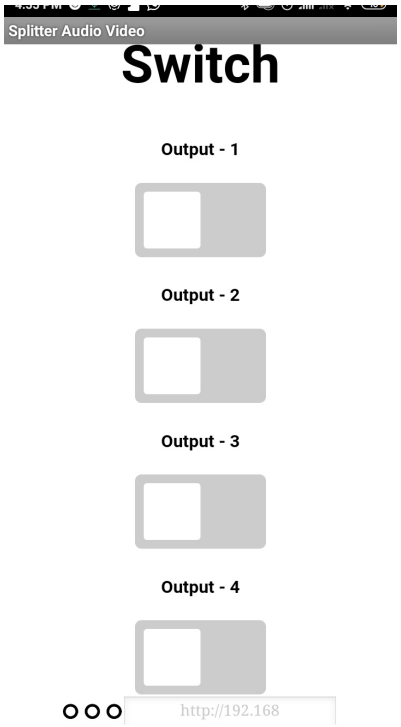


Figura 11: APP Splitter

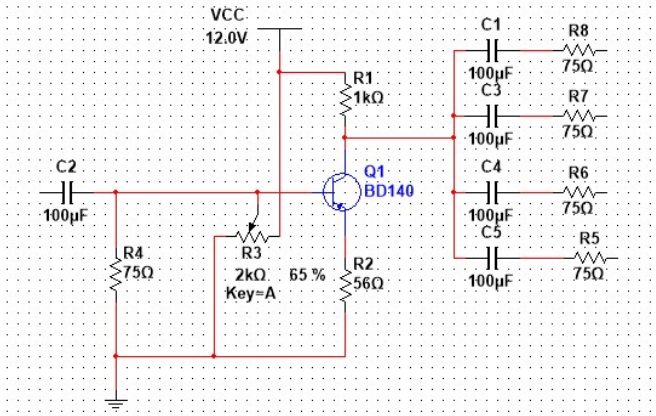


Figura 13: Simulación del circuito de vídeo

Circuito amplificador de Clase A o de polarizacion de tension, la amplificacion de la señal es controlada por el potenciometro conectado a la base del transisotror BD140, la salidas del amplificador estan en la parte del colector conectada a una resistencia de 75Ω el cual sera la resistencia de salida del mismo. La salida de video esta conectado a una resistencia de $10k\Omega$ para obtener una señal mas clara y estas salidas van a un hacia los optoacopladores. Este tipo de circuito es completo y sugerido en circuitos amplificadores de alta fidelidad.

### D. Diagrama utilizado para el Splitter de Audio

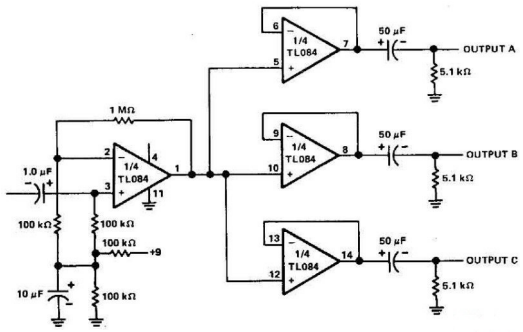


Figura 12: Digrama utilizado para Splitter Audio

### E. Diagrama utilizado para el Splitter de Video

Amplificador video con transistor Clase A:

## V. ANEXOS

### A. Vídeo de Funcionamiento

https://drive.google.com/file/d/1IplynSjL74pubwBZ144CK5FUx0wH_oNi/view?usp=sharing

### B. Codigo utilizado

https://drive.google.com/drive/folders/1g5AjZjX8WePr8Y6mQ5EodqxovwIUUzS2?usp=sharing

### C. APP

https://drive.google.com/file/d/1BQo7wQovPhdtxOElyafoR2wzGrz_JrWU/view?usp=sharing

### D. Imagen parte fisica fase final

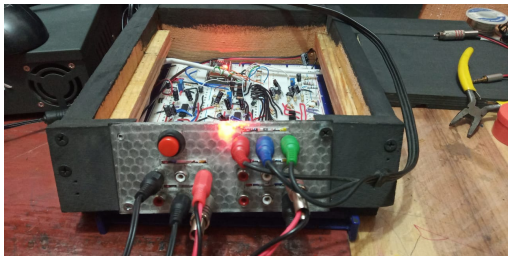

Figura 14: Splitter Audio y Video

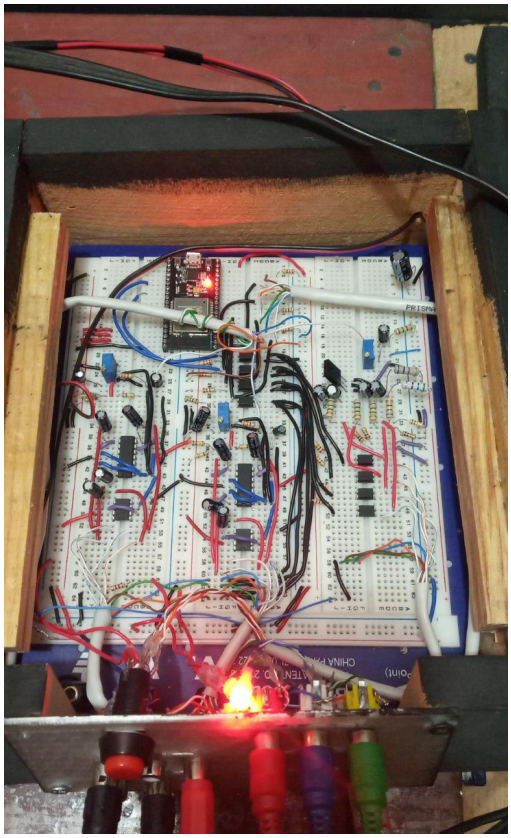

Figura 15: Splitter Auido y Video

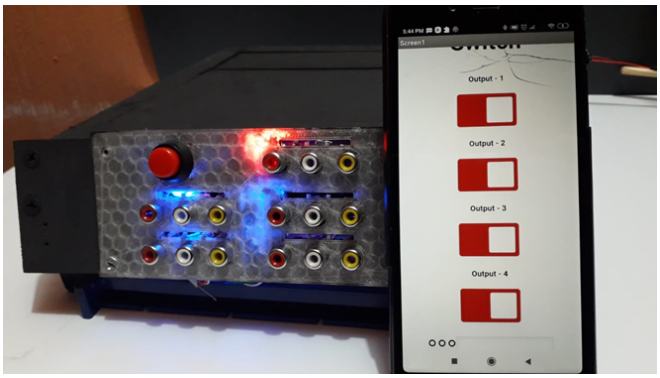


Figura 16: Implementación final

### E. Código de la implementación física del circuito de audio

## VI. FUENTES

https://randomnerdtutorials.com/esp32-async-web-server-espasyncwebserver-library/
http://circuittreecal.blogspot.com/2017/03/3-channels-audio-splitter-amplifier.html

```
// Import required libraries
#include <WiFi.h>
#include <AsyncTCP.h>
#include <ESPAsyncWebServer.h>

// Replace with your network credentials
const char* ssid = "Redmi";
const char* password = "5acbe082";

const char* PARAM_INPUT_1 = "output";
const char* PARAM_INPUT_2 = "state";

// Create AsyncWebServer object on port 80
AsyncWebServer server(80);

const char index_html[] PROGMEM = R"rawliteral(
<!DOCTYPE HTML><html>
<head>
  <title>Splitter</title>
  <meta name="viewport"
      content="width=device-width, initial-scale=1">
  <link rel="icon" href="data:,">
  <style>
    html {font-family: Arial; display:
        inline-block; text-align: center;}
    h2 {font-size: 3.0rem;}
    p {font-size: 3.0rem;}
    body {max-width: 600px; margin:0px auto;
        padding-bottom: 25px;}
    .switch {position: relative; display:
        inline-block; width: 120px; height: 68px}
    .switch input {display: none}
    .slider {position: absolute; top: 0; left: 0;
        right: 0; bottom: 0; background-color:
        #ccc; border-radius: 6px}
    .slider:before {position: absolute; content:
```

```
        ""; height: 52px; width: 52px; left: 8px;
        bottom: 8px; background-color: #fff;
        -webkit-transition: .4s; transition: .4s;
        border-radius: 3px}
    input:checked+.slider {background-color:
        #b30000}
    input:checked+.slider:before
        {-webkit-transform: translateX(52px);
        -ms-transform: translateX(52px); transform:
        translateX(52px)}
  </style>
</head>
<body>
  <h2>Switch</h2>
  %BUTTONPLACEHOLDER%
<script>function toggleCheckbox(element) {
  var xhr = new XMLHttpRequest();
  if(element.checked){ xhr.open("GET",
      "/update?output="+element.id+"&state=1",
      true); }
  else { xhr.open("GET",
      "/update?output="+element.id+"&state=0",
      true); }
  xhr.send();
}
</script>
</body>
</html>
)rawliteral";


// Replaces placeholder with button section in your
    web page
String processor(const String& var){
  //Serial.println(var);
  if(var == "BUTTONPLACEHOLDER"){
    String buttons = "";
    buttons += "<h4>Output - 1</h4><label
        class=\"switch\"><input type=\"checkbox\"
        onchange=\"toggleCheckbox(this)\" id=\"2\"
        " + outputState(2) + "><span
        class=\"slider\"></span></label>";
    buttons += "<h4>Output - 2</h4><label
        class=\"switch\"><input type=\"checkbox\"
        onchange=\"toggleCheckbox(this)\" id=\"4\"
        " + outputState(4) + "><span
        class=\"slider\"></span></label>";
    buttons += "<h4>Output - 3</h4><label
        class=\"switch\"><input type=\"checkbox\"
        onchange=\"toggleCheckbox(this)\" id=\"32\"
        " + outputState(32) + "><span
        class=\"slider\"></span></label>";
    buttons += "<h4>Output - 4</h4><label
        class=\"switch\"><input type=\"checkbox\"
        onchange=\"toggleCheckbox(this)\" id=\"33\"
        " + outputState(33) + "><span
        class=\"slider\"></span></label>";
    return buttons;
  }
  return String();
}

String outputState(int output){
  if(digitalRead(output)){
    return "checked";
  }
  else {
    return "";
  }
}
void setup(){
  // Serial port for debugging purposes
  Serial.begin(115200);

  pinMode(2, OUTPUT);//2
  digitalWrite(2, LOW);
  pinMode(4, OUTPUT);//4
  digitalWrite(4, LOW);
  pinMode(32, OUTPUT);//new
  digitalWrite(32, LOW);
  pinMode(33, OUTPUT);
  digitalWrite(33, LOW);

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi..");
  }

  // Print ESP Local IP Address
  Serial.println(WiFi.localIP());

  // Route for root / web page
  server.on("/", HTTP_GET, [](AsyncWebServerRequest
      *request){
    request->send_P(200, "text/html", index_html,
        processor);
  });

  // Send a GET request to
      <ESP_IP>/update?output=<inputMessage1>&state
  //=<inputMessage2>
  server.on("/update", HTTP_GET, []
      (AsyncWebServerRequest *request) {
    String inputMessage1;
    String inputMessage2;
    // GET input1 value on <ESP_IP>/update?output
    //=<inputMessage1>&state=<inputMessage2>
    if (request->hasParam(PARAM_INPUT_1) &&
        request->hasParam(PARAM_INPUT_2)) {
      inputMessage1 =
          request->getParam(PARAM_INPUT_1)->value();
      inputMessage2 =
          request->getParam(PARAM_INPUT_2)->value();
      digitalWrite(inputMessage1.toInt(),
          inputMessage2.toInt());
    }
    else {
      inputMessage1 = "No message sent";
      inputMessage2 = "No message sent";
    }
    Serial.print("GPIO: ");
    Serial.print(inputMessage1);
    Serial.print(" - Set to: ");
    Serial.println(inputMessage2);
    request->send(200, "text/plain", "OK");
  });
```

```
  // Start server
  server.begin();
}

//////////////////
```

```
void loop() {
digitalWrite(15, LOW);
}
```

[1] Apichet Garaipoom. (2 de enero de 2019). *Video amplifier splitter using transistor*. ElecCircuit. https://www.eleccircuit.com/video-amplifier-splitter/