



A4.1 Actividad de aprendizaje

Circuito de control para activar y desactivar un motor DC, utilizando NodeMCU ESP32 por medio de Bluetooth

Instrucciones

- Realizar un sistema ensamblado de control por medio de **Bluetooth**, capaz de control a un motor DC, utilizando un NodeMCU **ESP32**, un y un **IC L293D**.
- Toda actividad o reto se deberá realizar utilizando el estilo **MarkDown con extension .md** y el entorno de desarrollo VSCode, debiendo ser elaborado como un documento **single page**, es decir si el documento cuanta con imágenes, enlaces o cualquier documento externo debe ser accedido desde etiquetas y enlaces, y debe ser nombrado con la nomenclatura **A4.1_NombreApellido_Equipo.pdf**.
- Es requisito que el .md contenga una etiqueta del enlace al repositorio de su documento en GITHUB, por ejemplo **Enlace a mi GitHub** y al concluir el reto se deberá subir a github.
- Desde el archivo **.md** exporte un archivo **.pdf** que deberá subirse a classroom dentro de su apartado correspondiente, sirviendo como evidencia de su entrega, ya que siendo la plataforma **oficial** aquí se recibirá la calificación de su actividad.
- Considerando que el archivo **.PDF**, el cual fue obtenido desde archivo **.MD**, ambos deben ser idénticos.
- Su repositorio ademas de que debe contar con un archivo **readme.md** dentro de su directorio raíz, con la información como datos del estudiante, equipo de trabajo, materia, carrera, datos del asesor, e incluso logotipo o imágenes, debe tener un apartado de contenidos o indice, los cuales realmente son ligas o **enlaces a sus documentos .md**, *evite utilizar texto* para indicar enlaces internos o externo.
- Se propone una estructura tal como esta indicada abajo, sin embargo puede utilizarse cualquier otra que le apoye para organizar su repositorio.

```
- readme.md
- blog
  - C4.1_TituloActividad.md
  - C4.2_TituloActividad.md
  - C4.3_TituloActividad.md
  - C4.4_TituloActividad.md
- img
- docs
  - A4.1_TituloActividad.md
  - A4.2_TituloActividad.md
  - A4.3_TituloActividad.md
```

Fuentes de apoyo para desarrollar la actividad

- [Random Nerd Tutorial DHT Humedad y temperatura](#)
- [Motor DC con IC L293 y ESP32](#)

Desarrollo

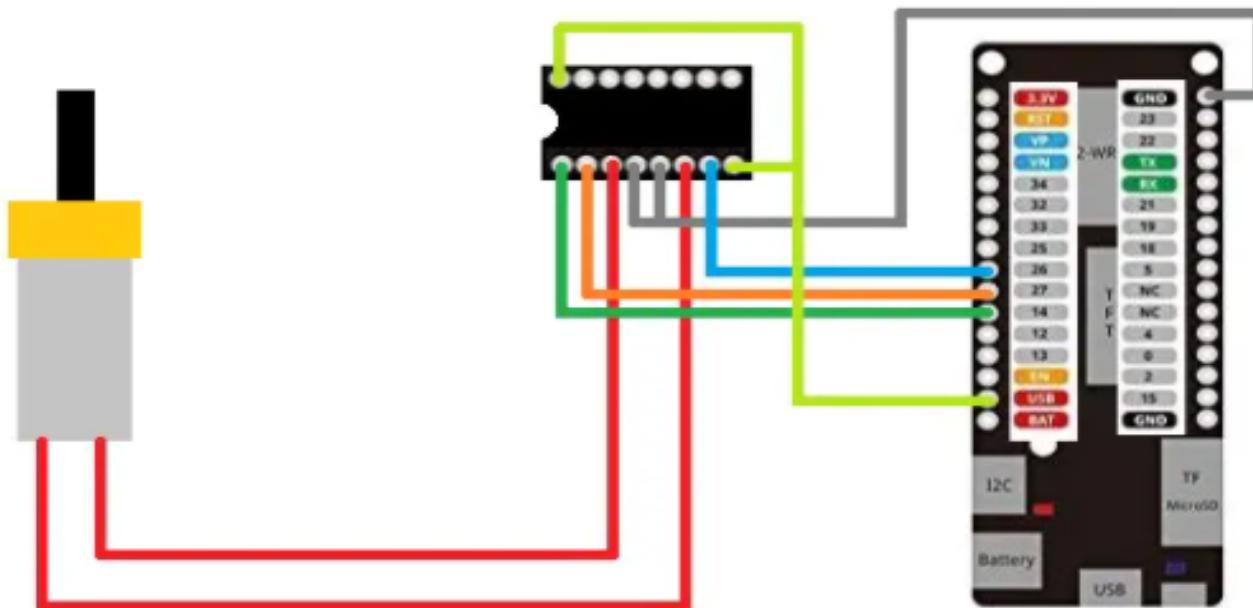
1. Utilizar el siguiente listado de materiales para la elaboración de la actividad

Cantidad	Descripción
1	IC L293D
1	Fuente de voltaje de 5V
1	NodeMCU ESP32
1	BreadBoard
1	Jumpers M/M

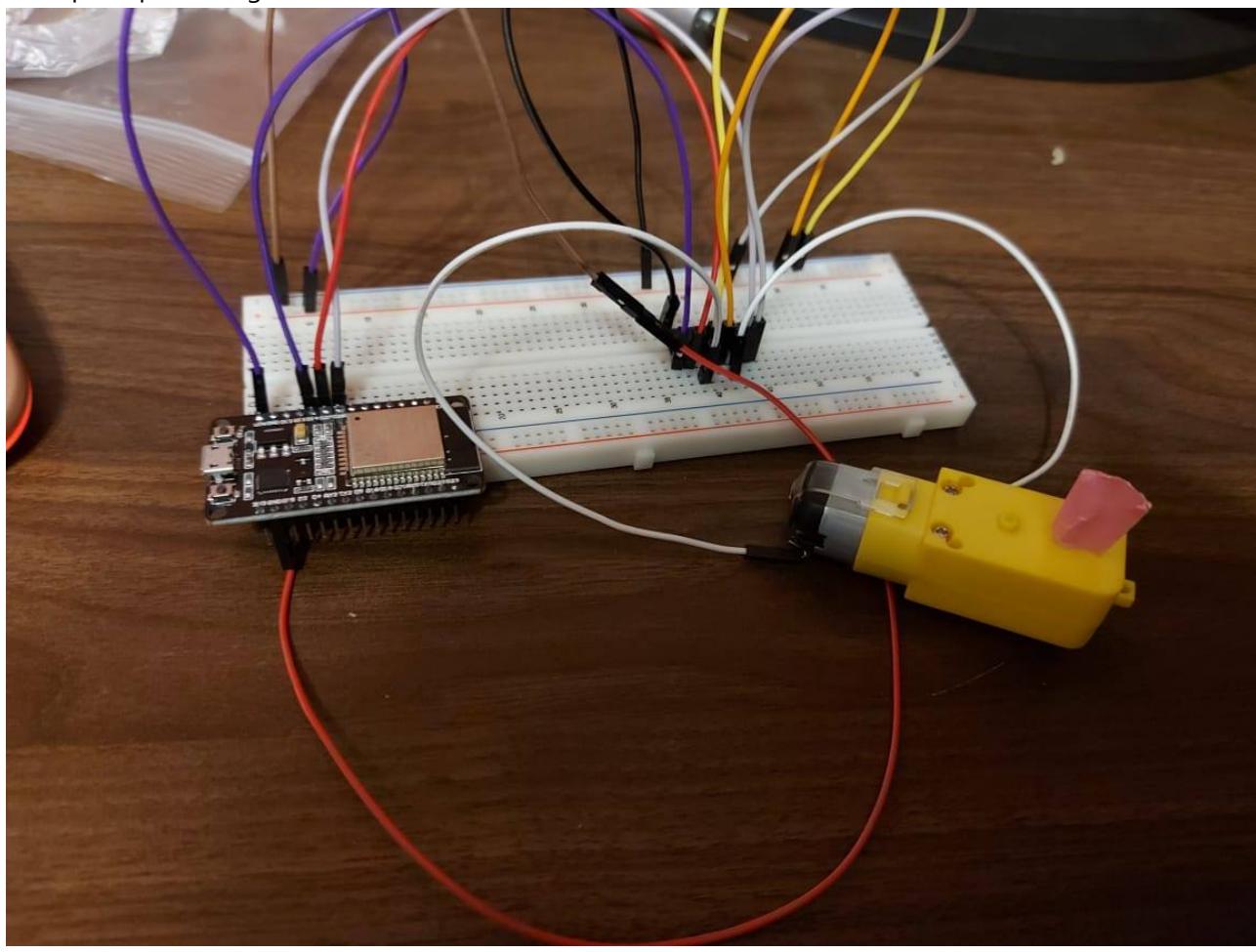
2. Basado en las imágenes que se muestran en las **Figura 1**, ensamblar un circuito electrónico, con la finalidad de obtener un sistema capaz de cumplir con las instrucciones siguientes:

- Por medio de la aplicación "Serial Bluetooth terminal" que puede ser descargada del play Store de google o incluso cualquier otra que considere, se deberá controlar el arranque y apagado de un motor DC, es decir se contara con dos peticiones, la cual una de ellas representara el "START" y la otra opción "STOP"
- El motor debe ser capaz de girar a favor de las manecillas del reloj durante 5 segundos, al cumplirse ese tiempo debe frenar 1 segundo e invertirá su giro durante otros 5 segundos.

Figura 1 Circuito ESP32 IC L293 Motor DC



3. Coloque aqui la imagen del circuito ensamblado



4. Coloque en este lugar el programa creado dentro del entorno de Arduino

```
#include "BluetoothSerial.h"

#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run make menuconfig to and enable it
#endif

BluetoothSerial SerialBT;
int received;// valor se guardara en esta variable
char receivedChar;// el valor recibido CHAR se guarda en esta variable

const char turnON ='o'; //variable para encender
const char turnOFF ='f';//Variable para apagar

const int motor1 = 27; //pines del motor
const int motor2 = 26;
const int enable1Pin = 14;//pin para habilitar el L239D
bool motor = false; //bandera para utilizar la secuencia
int Tiempo=0; // para medir el tiempo de activacion despues de que enciende el
motor

const int freq = 30000;
const int pwmChannel = 0;
```

```
const int resolution = 8;
int dutyCycle = 200;

void setup() {

    SerialBT.begin("ESP32"); //nombre del dispositivo BT
    Serial.println("El dispositivo se emparejo!");
    pinMode(enable1Pin, OUTPUT); // ponemos los pines del motor y del L239D como
salida
    pinMode(motor1, OUTPUT);
    pinMode(motor2, OUTPUT);
    //configuracion del pwm del motor
    ledcSetup(pwmChannel, freq, resolution);
    ledcAttachPin(enable1Pin, pwmChannel);
    Serial.begin(115200);//iniciar consola
}

void loop() {
    receivedChar =(char)SerialBT.read();// el valor de la variable es igual al
valor recibido en la consola BT

    if (Serial.available()) { //si la consola esta disponible el BT puede escribir y
la consola lo puede leer
        SerialBT.write(Serial.read());
    }

    if (SerialBT.available()) {
        SerialBT.print("Recibido:");// Escribir en BT app
        SerialBT.println(receivedChar);// Escribir en BT app
        Serial.print ("Recibido:");//imprimir en el monitor serial
        Serial.println(receivedChar);//imprimir en el monitor serial

        if(receivedChar == turnON)iniciar(); // si el caracter que se envia es igual a
la variable "turnON" iniciar la secuencia del motor
        else if (receivedChar == turnOFF) Apagar(); // si el caracter que se envia es
igual a la variablw "turnOFF" detener la secuencia del motor
    }
    if (motor) Secuencia();
}

//Iniciar la secuencia
void iniciar(){
    Tiempo = millis();
    motor=true;
}

//detener la secuencia
void Apagar(){
    Detener();
    motor=False;
}

//secuencia del motor
```

```
void Secuencia(){
    int tiempoTranscurrido= millis() - Tiempo; //El tiempo transcurrido es igual a
    la del tiempo de activacion del motor
    if(tiempoTranscurrido < 5000) Encendido(); // Encender el motor durante 5
    segundos
    else if(tiempoTranscurrido < 6000) Detener(); // Apagar el motor despues de los
    5 segundos
    else if(tiempoTranscurrido < 11000) Reversa(); // Encender el motor en reversa
    otros 5 segundos
    else if(tiempoTranscurrido < 12000) Detener(); // Detener el motor
    else iniciar(); //Reiniciar la secuencia
}

void Encendido(){
dutyCycle = 255;
ledcWrite(pwmChannel, dutyCycle);
Serial.println("Motor hacia delante");
digitalWrite(motor1, LOW);
digitalWrite(motor2, HIGH);
}

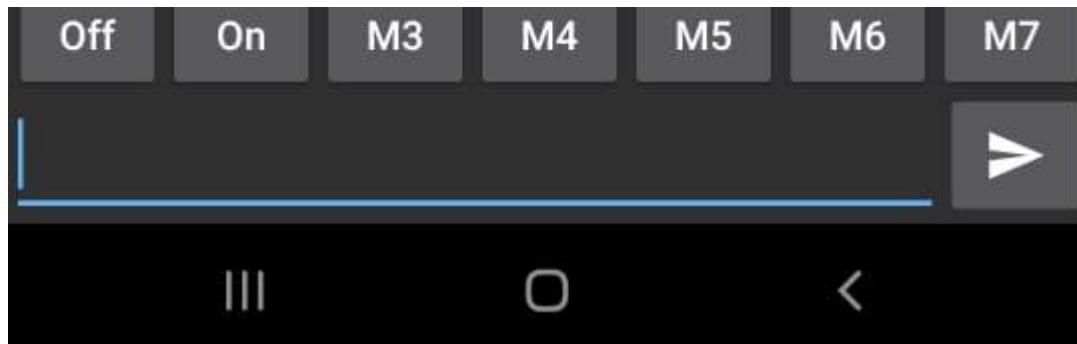
// para mover motor hacia atras
void Reversa(){
Serial.println("El motor se mueve hacia atras");
digitalWrite(motor1, HIGH);
digitalWrite(motor2, LOW);
}

void Detener(){
Serial.println("el motor se detuvo");
digitalWrite(motor1, LOW);
digitalWrite(motor2, LOW);
}
```

5. Coloque aquí evidencias que considere importantes durante el desarrollo de la actividad.

- [Video Demostrativo](#)

The screenshot shows a mobile application interface for a terminal or serial port. At the top, there is a blue header bar with the time "8:28" and several icons for signal strength, battery, and connectivity. Below the header, the word "Terminal" is displayed in white text. To the right of "Terminal" are three icons: a left arrow, a trash can, and a vertical ellipsis. The main area of the screen is a dark gray scrollable text view showing a log of messages. The messages are color-coded: green for most entries, red for some, and yellow for one. The log starts with "20:24:37.035 Connecting to ESP32 ..." in green, followed by "20:24:38.717 Connected" in yellow. Subsequent messages include "20:24:40.638 o", "20:24:40.680 Recibido:♦", "20:24:40.695 Recibido:o", "20:24:40.695 Recibido:^M", "20:24:55.188 f", "20:24:55.747 Recibido:f", "20:24:55.747 Recibido:^M", "20:26:41.379 o", "20:26:41.457 Recibido:♦", "20:26:41.457 Recibido:o", "20:26:41.457 Recibido:^M", "20:27:05.818 f", "20:27:05.889 Recibido:f", "20:27:05.921 Recibido:^M", "20:27:08.441 o", "20:27:08.493 Recibido:o", "20:27:08.518 Recibido:^M", "20:27:09.777 f", "20:27:09.836 Recibido:f", "20:27:09.863 Recibido:^M", "20:27:11.048 o", "20:27:11.147 Recibido:♦", "20:27:11.176 Recibido:o", "20:27:11.176 Recibido:^M", "20:27:12.347 f", "20:27:12.400 Recibido:f", "20:27:12.428 Recibido:^M", "20:27:13.593 o", "20:27:13.652 Recibido:♦", "20:27:13.682 Recibido:o", "20:27:13.682 Recibido:^M", "20:27:15.246 f", "20:27:15.310 Recibido:f", "20:27:15.343 Recibido:^M". The bottom of the screen features a navigation bar with five gray rectangular buttons.



COM8

[Enviar](#)

Autoscroll

Nueva línea

115200 baud

[Clear output](#)

COM8

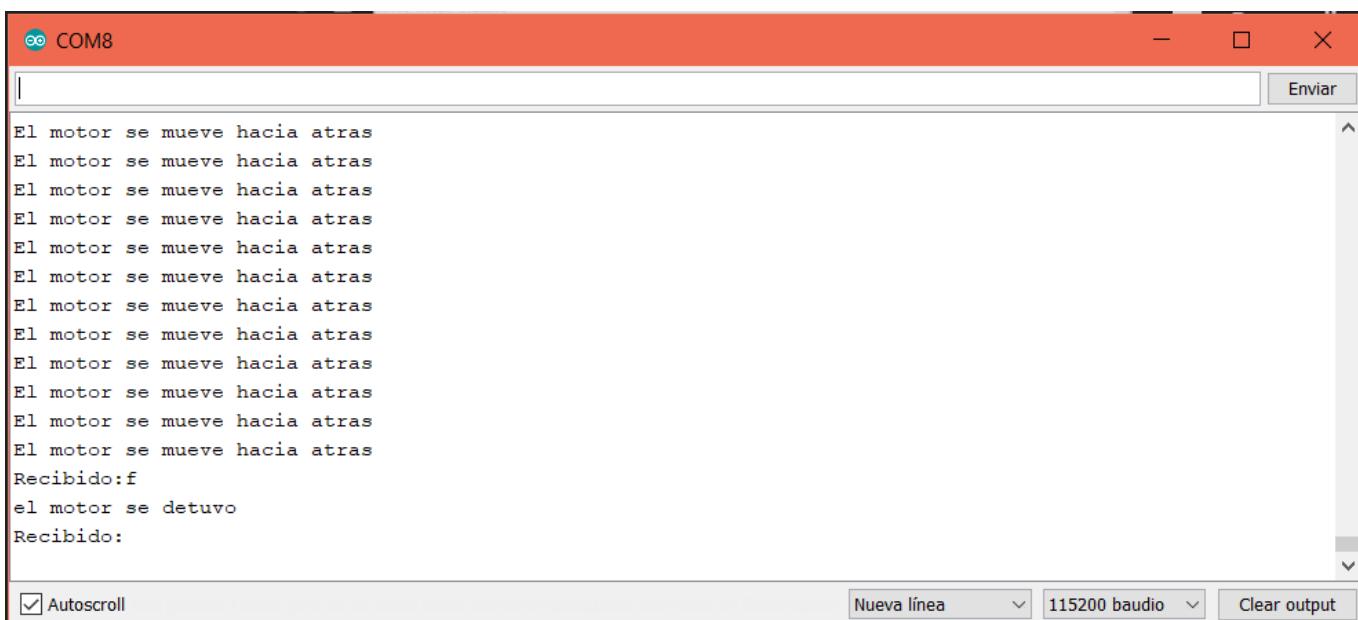
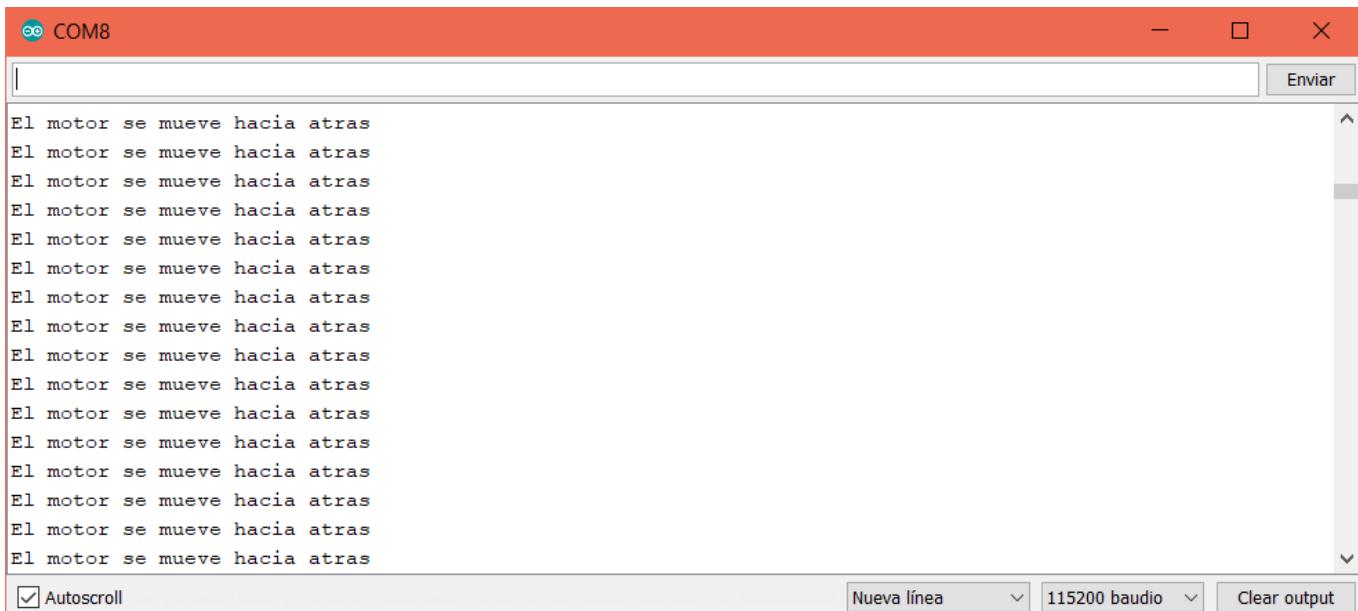
— □ ×

Autoscroll

Nueva línea

115200 baud

[Clear output](#)



Slack

The screenshot shows two screenshots of a Slack channel named 'zerox' within a workspace 'ITN_SistemasProgramables'. The left screenshot displays a conversation between users Vanessa Marlenne Rodriguez Baez, Alejandro Diaz Nava, and another user. The right screenshot shows a code editor with C++ code related to Bluetooth and motor control.

Conversation Screenshot:

- Vanessa Marlenne Rodriguez Baez: Chicos Ya me salio
- ALEJANDRO DIAZ NAVA: genial
- entonces mañana, reunion
- Vanessa Marlenne Rodriguez Baez: Son una de las evidencias
- Ahorita les mando el codigo

Code Editor Screenshot:

```
#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENAE
#error Bluetooth is not enabled! Please run make menuconfig to and enable it
#endif

BluetoothSerial SerialBT;
int received;// valor se guardara en esta variable
char receivedChar;// el valor recibido CHAR se guarda en esta variable

const char turnON = '0'; //variable para encender
const char turnOFF = '1';//Variable para apagar

const int motor1 = 27; //pines del motor
const int motor2 = 26;
const int enable1Pin = 14;//pin para habilitar el L239D
bool motor = false; //bandera para utilizar la secuencia
int tiempo=0; // para medir el tiempo de activacion despues de que enciende el motor

const int freq = 30000;
const int pwmChannel = 0;
const int resolution = 8;
int dutyCycle = 200;

void setup()
{
    SerialBT.begin("ESP32"); //nombre del dispositivo BT
    Serial.println("El dispositivo se emparejó!");
    pinMode(enable1Pin, OUTPUT); // ponemos los pines del motor y del L239D como salida
    pinMode(motor1, OUTPUT);
    pinMode(motor2, OUTPUT);
    //configuracion del num del motor
```

ITN_SistemasProgramables

zerox

```

receivedChar =(char)SerialBT.read(); // el valor de la variable es igual al valor Hoy en la consola BT
if (Serial.available()) { // si la consola esta disponible el BT puede escribir y la consola lo puede leer
    SerialBT.write(Serial.read());
}

if (SerialBT.available()) {
    SerialBT.print("Recibido:"); // Escribir en BT app
    SerialBT.println(receivedChar); // Escribir en BT app
    Serial.print ("Recibido:"); // Imprimir en el monitor serial
    Serial.println(receivedChar); // Imprimir en el monitor serial

    if(receivedChar == turnON)iniciar(); // si el caracter que se envia es igual a la variable "turnON" iniciar la secuencia del motor
    else if (receivedChar == turnOFF) Apagar(); // si el caracter que se envia es igual a la variable "turnOFF" detener la secuencia del motor
}
if (motor) Secuencia();
}

//Iniciar la secuencia
void iniciar(){
    Tiempo = millis();
    motor=true;
}

//detener la secuencia
void Apagar(){
    Detener();
    motor=false;
}

//secuencia del motor

```

Enviar mensaje a zerox

Aa @ ⓘ

20:44 martes
08/12/2020

ITN_SistemasProgramables

zerox

```

else if(tiempoTranscurrido < 6000) Detener(); // Apagar el motor despues de 6 segundos
else if(tiempoTranscurrido < 11000) Reversa(); // Encender el motor en reversa otros 5 segundos
else if(tiempoTranscurrido < 12000) Detener(); // Detener el motor
else iniciar(); //Reiniciar la secuencia

}

void Encendido(){
dutyCycle = 255;
ledcWrite(pwmChannel, dutyCycle);
Serial.println("Motor hacia delante");
digitalWrite(motor1, LOW);
digitalWrite(motor2, HIGH);
}

// para mover motor hacia atras
void Reversa(){
Serial.println("El motor se mueve hacia atras");
digitalWrite(motor1, HIGH);
digitalWrite(motor2, LOW);
}

void Detener(){
Serial.println("el motor se detuvo");
digitalWrite(motor1, LOW);
digitalWrite(motor2, LOW);
}

Si mañana Reunion para explicarles el código y que miren como funciona el programa

```

Enviar mensaje a zerox

Aa @ ⓘ

20:44 martes
08/12/2020

Reunión

A screenshot of a Windows desktop during a video conference. In the top right corner, there's a video feed of a person standing on a beach at sunset. The main window is an Arduino IDE showing code for an ESP32 module. The taskbar at the bottom has several pinned icons, including Google, AliExpress, Facebook, Booking.com, and Cinemex. The system tray shows the date as 09/12/2020 and the time as 12:12 p.m.

A.4.1 Circuito de control para activar y desactivar un motor DC, utilizando NodeMCU ESP32

3 visualizaciones • 8 dic 2020

SUSCRIBIRSE

12:14 p.m.
09/12/2020

(56) A.4.1 Circuito de control para activar y desactivar un motor DC, utilizando NodeMCU ESP32

0:24 / 0:56

12:14
09/12/2020

The screenshot shows a Microsoft Windows desktop environment. At the top, there's a taskbar with several pinned icons. An open browser window displays a presentation slide titled "Conclusões" (Conclusions) with three bullet points: "Diaz Navarro Alejandro", "Rodríguez Báez Vanessa Marlenne", and "Soria Márquez Guillermo". Below the browser is a Visual Studio Code window showing a file named "Preview A4.1 VanessaMarleneRodriguezBaez_Zerox.md" containing code related to "SISTEMAS PROGRAMABLES". To the right of the code editor is a Microsoft Teams meeting interface. The Teams interface shows a video feed of a person named "vanessa rodriguez" and a rubric table with four rows: "Instrucciones", "Desarrollo", "Demostración", and "Conclusiones". The rubric table includes descriptions and scores for each criterion. The bottom right corner of the screen shows the date and time: "12:15 p.m. 09/12/2020".

Conclusiones

- Diaz Navarro Alejandro:** Con esta práctica se logró comprender el manejo del motor con la incorporación y utilización de bluetooth, con apoyo de las prácticas anteriores se maneja la implementación de la señal pwm para generar el movimiento del motor, el cambio de sentido o la detención del mismo, entre las complicaciones de esta práctica fue realizar la programación y la conexión con bluetooth ya que al querer implementar la conexión con estos dos el motor dejaba de funcionar y no se lograba solucionar.
- Rodríguez Báez Vanessa Marlenne:** En esta practica se realizo algo similar con lo de la practica pasada de controlar algo por bluetooth, esta practica fue controlar un motor DC con un puente h o un L293D que funciona para poder controlar motores, yo había utilizado lo que es el puente h pero en placa por que utilice 2 motores DC, la dificultad de esta practica fue a la hora de hacer el código ya que nos basamos del materia de ayuda que venia en la practica pero al momento de incluir el bluetooth no funcionaba el motor lo cual eso fue lo mas complicado de esta practica, despues de ver un poco los programas de nuestros compañeros pudimos resolver nuestro error.
- Soria Márquez Guillermo:** En esta práctica volvimos a utilizar el ESP32 con la implementación de un Motor DC, esta práctica en particular fue la más complicada que se ha presentado ya que tuvimos mucho problemas con la codificación y sobre todo con la conexión con el bluetooth. En nuestro caso desarrollamos un código pero al momento de conectarlo con el bluetooth no encendía el motor por error de código. Después de investigar por nuestra parte no lo resolvimos en el momento ya que todavía no sabíamos porque ocurría el error de código. Tardamos un poco más de lo normal en terminar esta práctica pero al final logramos resolver los errores que teníamos.

Rubrica

Criterios	Descripción	Puntaje
-----------	-------------	---------

Criterios	Descripción	Puntaje
Instrucciones	Se cumple con cada uno de los puntos indicados dentro del apartado Instrucciones?	10
Desarrollo	Se respondió a cada uno de los puntos solicitados dentro del desarrollo de la actividad?	60
Demostración	El alumno se presenta durante la explicación de la funcionalidad de la actividad?	20
Conclusiones	Se incluye una opinión personal de la actividad por cada uno de los integrantes del equipo?	10

 [Link Díaz Navarro Alejandro](#)

 [Link Rodríguez Báez Vanessa Marlenne](#)

 [Link Soria Márquez Guillermo](#)