



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

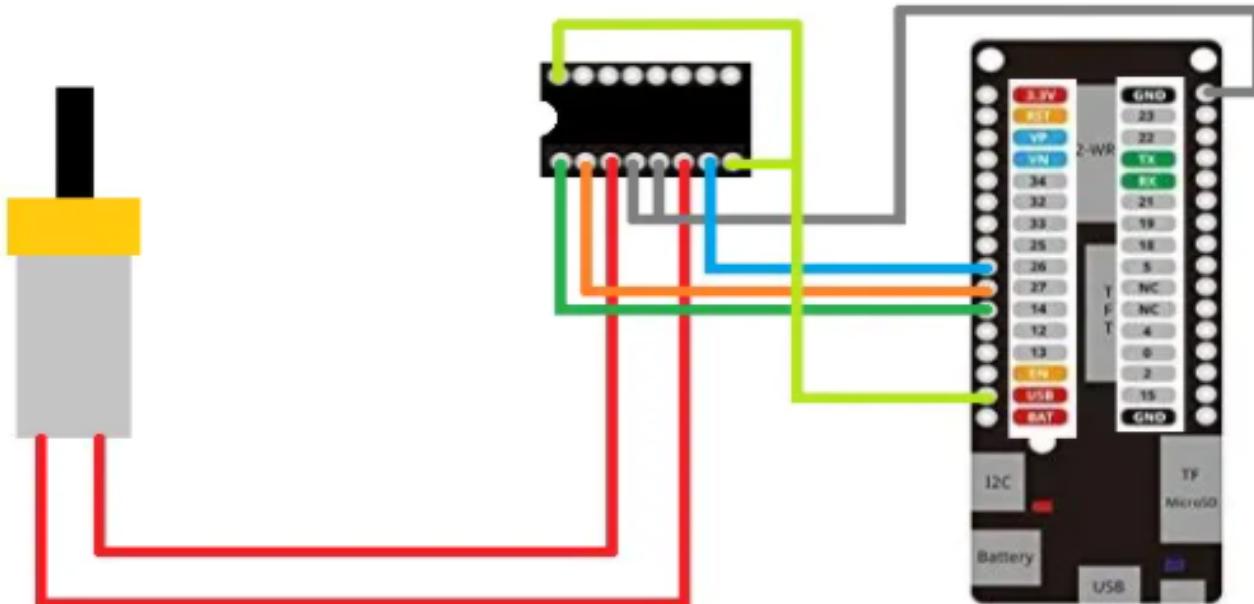
1. Utilizar el siguiente listado de materiales para la elaboración de la actividad

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

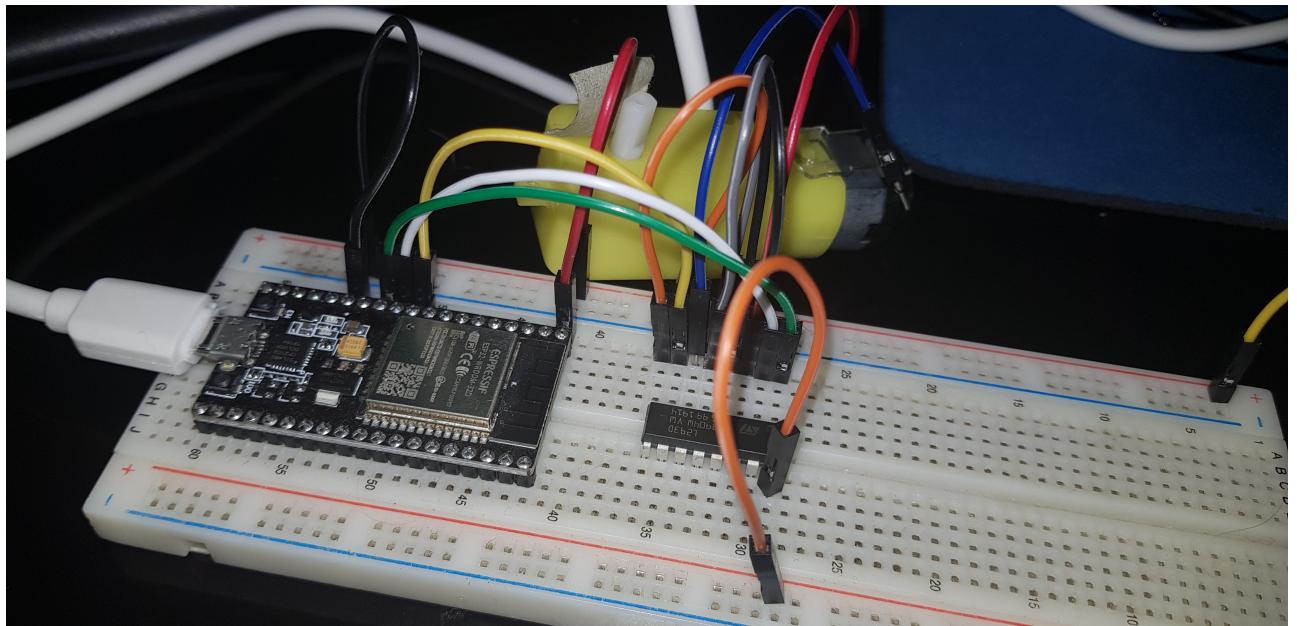
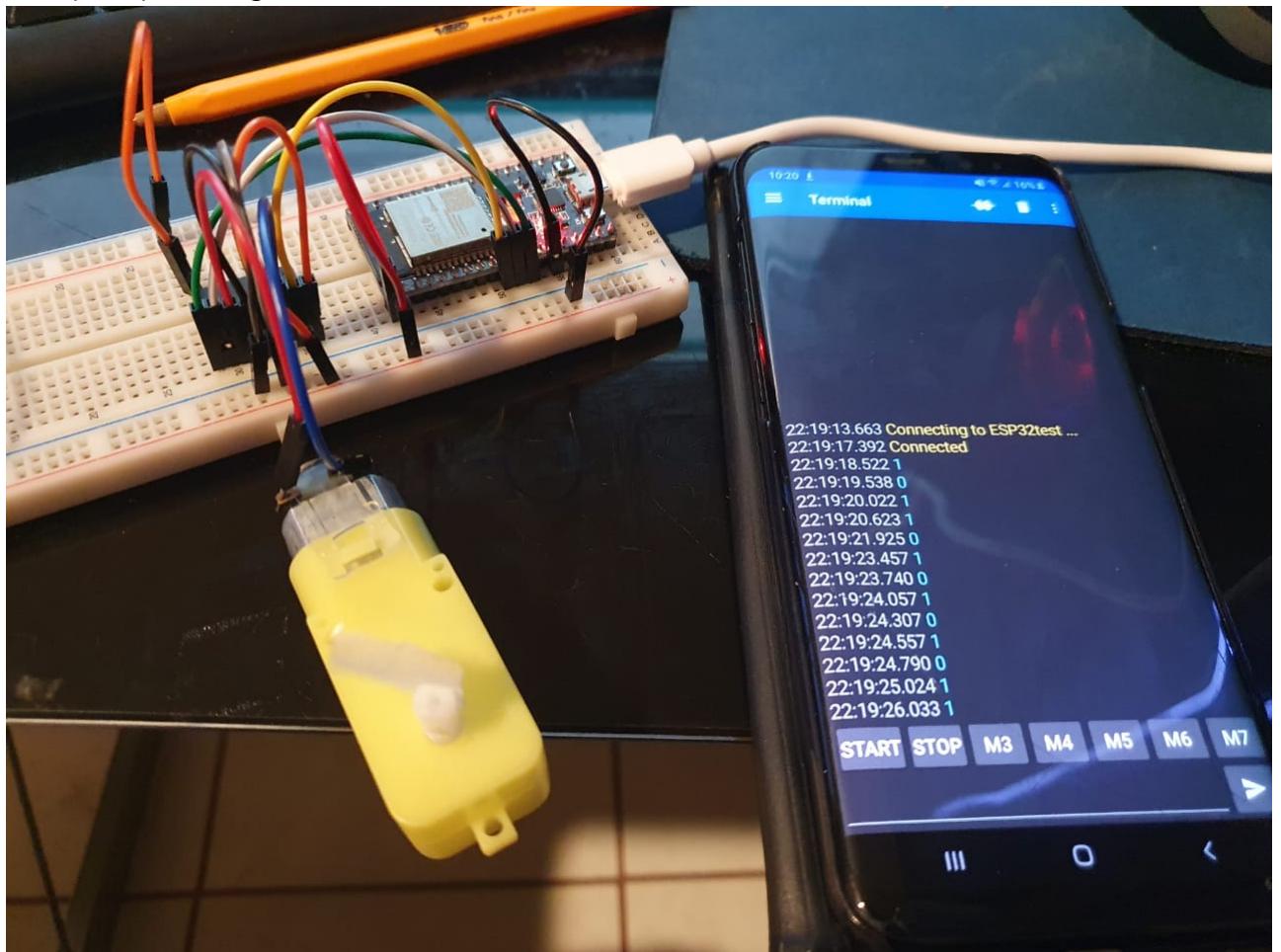
2. Basado en las imágenes que se muestran en las **Figura 1**, ensamblar un circuito electrónico, con la finalidad de obtener un sistema capaz de cumplir con las instrucciones siguientes:

- 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

```
//Libreria para utilizar Bluetooth
#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;

// Variables del mensaje
String message = "";
char incomingChar;
bool valor = true;

// Pin del motor
int motor1Pin1 = 27;
int motor1Pin2 = 26;
int enable1Pin = 14;

void setup() {
    // Salidas del motor
    pinMode(motor1Pin1, OUTPUT);
    pinMode(motor1Pin2, OUTPUT);
    pinMode(enable1Pin, OUTPUT);

    Serial.begin(115200);

    //Se asigna nombre para encontrar el ESP32 en Bluetooth
    SerialBT.begin("ESP32test"); //Bluetooth device name

    //Mensaje para indicar que se inicio la libreria Bluetooth y motor
    Serial.println("Iniciando conexion con el Bluetooth");
    Serial.print("Iniciando motor");
}

void loop() {
    //Si la libreria de Bluetooth esta activada va a leer el puerto serial
    if (SerialBT.available()){
        char incomingChar = SerialBT.read();
        if (incomingChar != '\n'){
            message += String(incomingChar);
        }
        else{
            message = "";
        }
        Serial.write(incomingChar);
    }

    // Si el mensaje es 1 inicia el ciclo
    if (message == "1"){
        while(valor == true){
            // Forward
            Serial.println("");
            Serial.println("Motor en movimiento");
            digitalWrite(motor1Pin2, LOW);
            digitalWrite(motor1Pin1, HIGH);
            for(int i = 0; i <= 50; i++){
                {
                    delay(100);
                    Leer();
                    if (message == "0")

```

```
{  
    goto detener;  
}  
  
// Se detiene el motor  
Serial.println("Motor detenido");  
digitalWrite(motor1Pin2, LOW);  
digitalWrite(motor1Pin1, LOW);  
for(int i = 0; i <= 10; i++)  
{  
    delay(100);  
    Leer();  
    if (message == "0")  
    {  
        goto detener;  
    }  
}  
  
// Reversa  
Serial.println("Motor en reversa");  
digitalWrite(motor1Pin2, HIGH);  
digitalWrite(motor1Pin1, LOW);  
for(int i = 0; i <= 50; i++)  
{  
    delay(100);  
    Leer();  
    if (message == "0")  
    {  
        goto detener;  
    }  
}  
  
// Se detiene el motor  
Serial.println("Motor detenido");  
digitalWrite(motor1Pin2, LOW);  
digitalWrite(motor1Pin1, LOW);  
for(int i = 0; i <= 10; i++)  
{  
    delay(100);  
    Leer();  
    if (message == "0")  
    {  
        goto detener;  
    }  
}  
}  
}  
//Si el mensaje es STOP se apaga el motor  
if (message == "0"){  
    detener:  
    //valor = false;  
    Serial.println("");  
    Serial.println("Motor detenido");
```

```
    digitalWrite(motor1Pin2, LOW);
    digitalWrite(motor1Pin1, LOW);
    delay(1000);
}
delay(20);
}

void Leer()
{
    if (SerialBT.available()){
        char incomingChar = SerialBT.read();
        if (incomingChar != '\n'){
            message += String(incomingChar);
        }
        else{
            message = "";
        }
        Serial.write(incomingChar);
    }
}
```

21:57:39.533 -> Iniciando conexión con el Bluetooth
21:57:39.533 -> Iniciando motor1
21:57:53.570 -> Motor en movimiento
21:57:53.672 ->
21:57:58.644 -> Motor detenido
21:57:59.766 -> Motor en reversa
21:58:04.874 -> Motor detenido
21:58:05.962 ->
21:58:05.962 -> Motor en movimiento
21:58:11.074 -> Motor detenido
21:58:12.165 -> Motor en reversa
21:58:14.962 -> 0
21:58:14.962 -> Motor detenido
21:58:15.984 ->
21:58:16.018 -> 0
21:58:16.018 -> Motor detenido
21:58:17.040 ->
21:58:17.756 -> 1
21:58:17.756 -> Motor en movimiento
21:58:17.824 ->
21:58:20.826 -> 0

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

- Evidencia física realizada por: Jose Alfredo Venegas Medina
- [Presentacion del circuito \(VIDEO\)](#)

reunion_final

Levantar la mano

Jose Venegas está presentando

3:11 PM 12/6/2020

44. Augusto 18.10

Archivo | Edor | Programa | Herramientas | Ayuda

Serial.begin(115200);

```

//Se asigna nombre para encontrar el ESP32 en Bluetooth
SerialBT.begin("ESP32test"); //Bluetooth device name

//Mensaje para indicar que se inicio la libreria Bluetooth y motor
Serial.println("Iniciandoconexion con el Bluetooth");
Serial.print("Iniciando motor");

void loop() {
//Si la libreria de bluetoothesta activada va a leer el punto
if (SerialBT.available()){
    char incomingChar = SerialBT.read();
    if (incomingChar != '\n'){
        message += String(incomingChar);
    }
    else{
        message = "";
    }
    Serial.write(incomingChar);
}

// Si el mensaje es I inicia el ciclo
if (message == "I"){
    while(valor == true){
        // Forward
        Serial.println("Motor en movimiento");
        digitalWrite(motor1Pin2, LOW);
        digitalWrite(motor1Pin1, HIGH);
        for(int i = 0; i < 50; i++)
    }
}

```

15:13:29.669 -> ets Jan 8 2016 00:22:57

15:13:29.669 ->

15:13:29.669 -> rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)

15:13:29.669 -> configip: 0, SPMN:0xee

15:13:29.669 -> CLK_drv:0x00,q_drv:0x00,a_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00

15:13:29.669 -> mode:DIO, clock div:1

15:13:29.669 -> load:0x0fff0016,len:4

15:13:29.669 -> load:0x0fff001c,len:1044

15:13:29.669 -> load:0x40078000,len:8996

15:13:29.669 -> load:0x40000400,len:5016

15:13:29.669 -> entry 0x4000064c

15:13:30.451 -> Iniciandoconexion con el Bluetooth

15:13:30.451 -> Iniciando motor

SOFT_ESP32_DEMO_V_1.0.0.20190524_Wemos_E5MA 3:11 PM 12/6/2020

DULCE JASMIN VILLALOBOS PEREZ

8 / 10

- Ramirez Cervantes Cesar Manuel:**

Para esta práctica se requería realizar un circuito con el ESP232 que fuera capaz de establecer conexión con un móvil por vía Bluetooth para que a través del dispositivo se enviara una señal al ESP232 y este dependiendo de lo que reciba realizará alguna instrucción, para este caso se debía contar con dos envíos de señal una "START" la cual daría inicio al circuito, es decir, se activaría el pequeño motor el cual daría durante 5 segundos hacia un sentido seguido se detendrá para después volver a girar pero al sentido opuesto, también la señal de "STOP" la cual sin importar en qué estado se encuentre el motor (girando en cualquier sentido) este se debe detener por completo, es decir, apagarse el circuito. Para poder hacer la conexión del motor al ESP232 se utilizó un puente H IC L293 el cual sirve para manejar cargas de potencia media, en especial pequeños motores y cargas inductivas, con la capacidad de controlar corriente hasta 600 mA en cada circuito y una tensión entre 4,5 V a 36 V.

- Venegas Medina Jose Alfredo:**

En la realización de esta práctica comprobamos el control de un motor por medio de un puente H y un ESP32 que lo controlaría recibiendo una señal Bluetooth. Para su realización se tuvo que comprender y analizar el funcionamiento de un puente H, ya que se desconocía. Una vez realizada la conexión, lo complejo fue la implementación del código, ya que, aunque hacer el ciclo de giro fue fácil (dar vuelta 5 segundos, detenerse 1 segundo, ir en reversa 5 segundos y detenerse 1 segundo), la duda que teníamos era a la hora de poder detener el motor en cualquier momento, para esto decidimos que únicamente se enviaría un 0 y un 1 de la terminal conectada por Bluetooth al ESP32 para encender y detener el motor. Esto ya que enviar una palabra tardaría más tiempo y por lo tanto el motor tardaría en detenerse una vez haya sido mandada la instrucción. Así que se creó un método que detectara si llegaba un 0 al puerto serial y lo haría cada 100 milisegundos. La aplicación que yo podría encontrar de esto sería en un carro o tal vez un dron a control remoto, donde tú le mandes una señal para detenerlo e igualmente poder mandarle una señal para volverlo a encenderlo.

- Villalobos Perez Dulce Jasmin:**

En esta práctica se utilizó un motor de DC un ejemplo sencillo de la aplicación de este motor sería los motores qué usa un carrito de juguete, al usar eso de la conexión de bluetooth para controlar el movimiento, podemos ver cómo se puede controlar desde el uso de un control remoto, gracias a una aplicación, ya que existe una comunicación entre el móvil y la placa con bluetooth, dicho esto para poder usar el bluetooth en la práctica es necesario el uso de una librería qué nos permite darle uso, en esta práctica se tuvo qué conectar el motor y al Esp32 por medio de un puente IC L293 H, como se muestra en el diagrama, cuando se trabajó con el código se presentó una pequeña complicación qué si llevo tiempo el encontrar una solución, ya que como sabemos en esta práctica el objetivo era hacer qué el motor se moviera por 5 segundos hacia delante, luego se detiene un segundo y luego va de reversa por otros 5 segundos, luego se vuelve a detener y sigue hacia delante y así en forma de ciclo, pero una vez resultó esto, al final se logró poder detener el movimiento del motor en cualquier momento qué se le envía stop con la aplicación.

 **Rubrica**

Criterios	Descripción	Puntaje
Instrucciones	Se cumple con cada uno de los puntos indicados dentro del apartado Instrucciones?	10

Criterios	Descripción	Puntaje
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

EQUIPO VERDE**Acevedo Ensiso Pedro Gabriel:**
 [Ir a mi Github](#)
Ramirez Cervantes Cesar Manuel:
 [Enlace a mi repositorio](#) 
Venegas Medina Jose Alfredo:
 [Mi Github](#)
Villalobos Perez Dulce Jasmin:
 [ENLACE - MI GITHUB](#)