



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

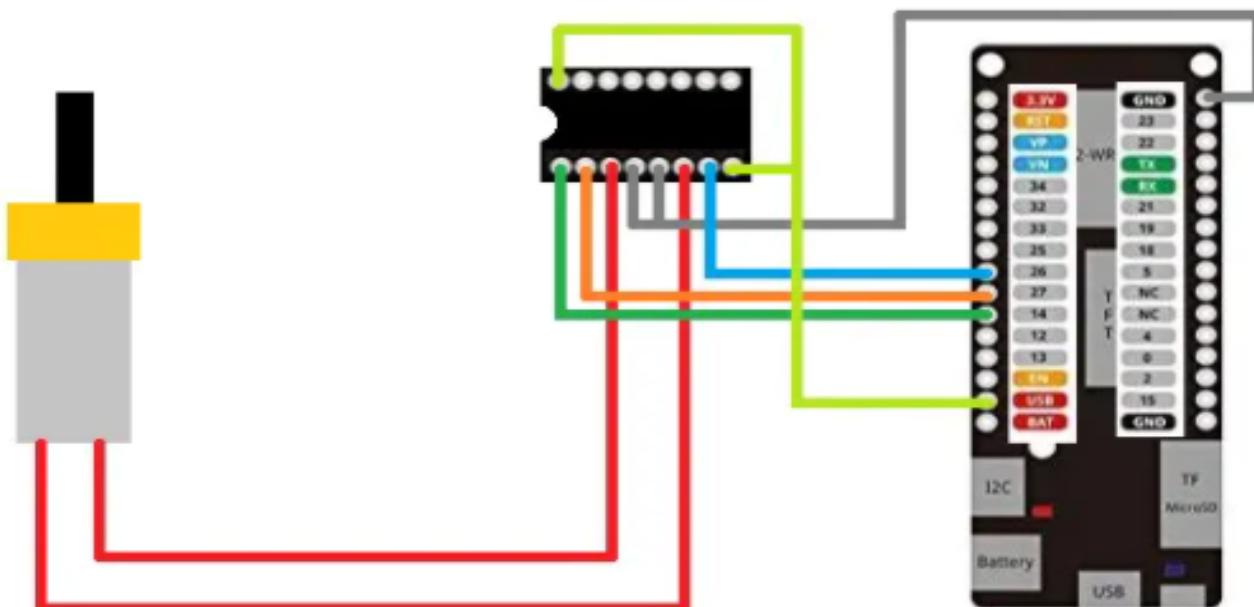
1. Utilizar el siguiente listado de materiales para la elaboración de la actividad

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

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, es decir la actividad debe tener la secuencia siguiente: El **stop** puede ser ejecutado en cualquier instante, y el motor estará ejecutando 5s en forward, 1s stop, 5s reverse, 1s stop, 5s forward, 1s stop, 5s reverse,...

Figura 1 Circuito ESP32 IC L293 Motor DC

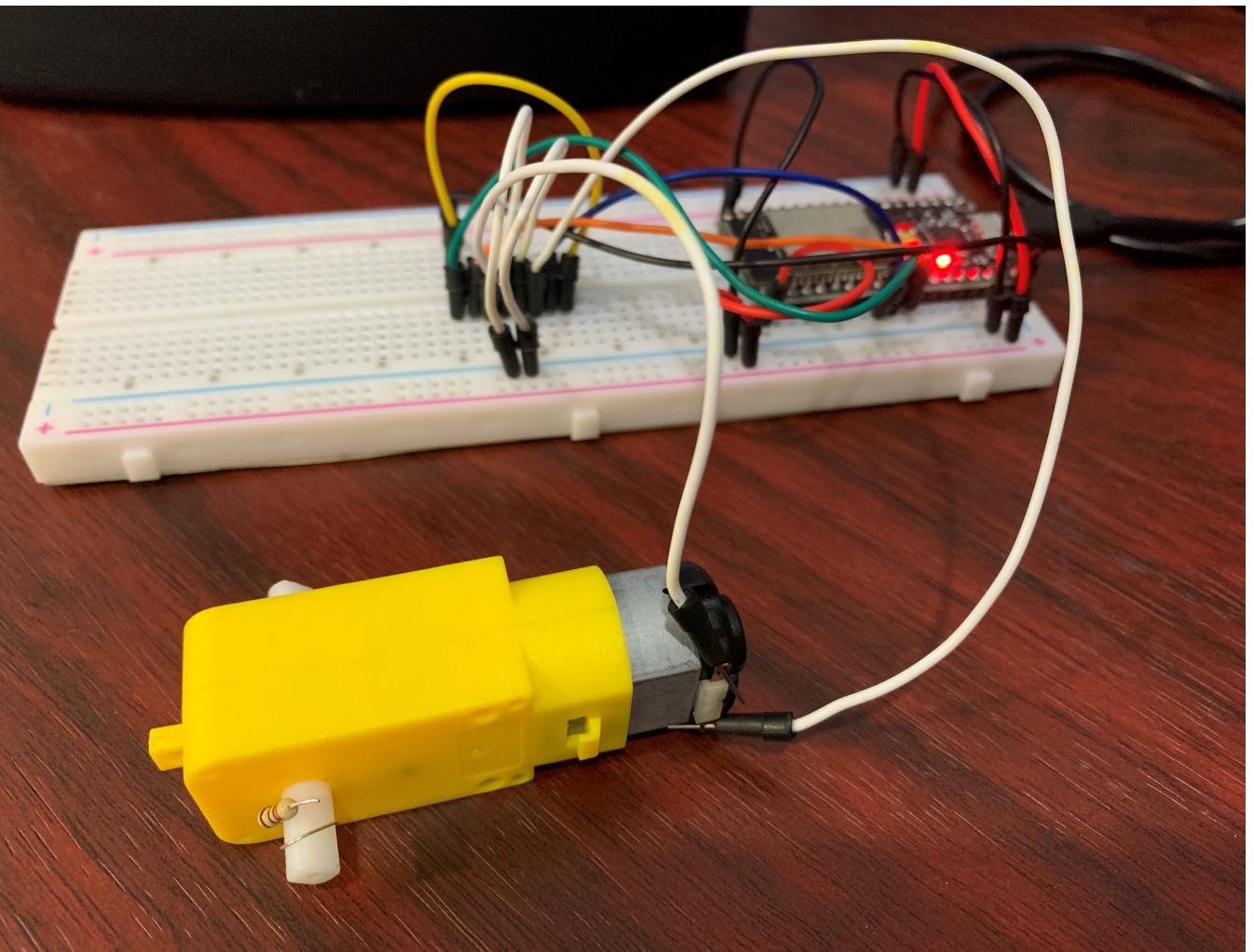
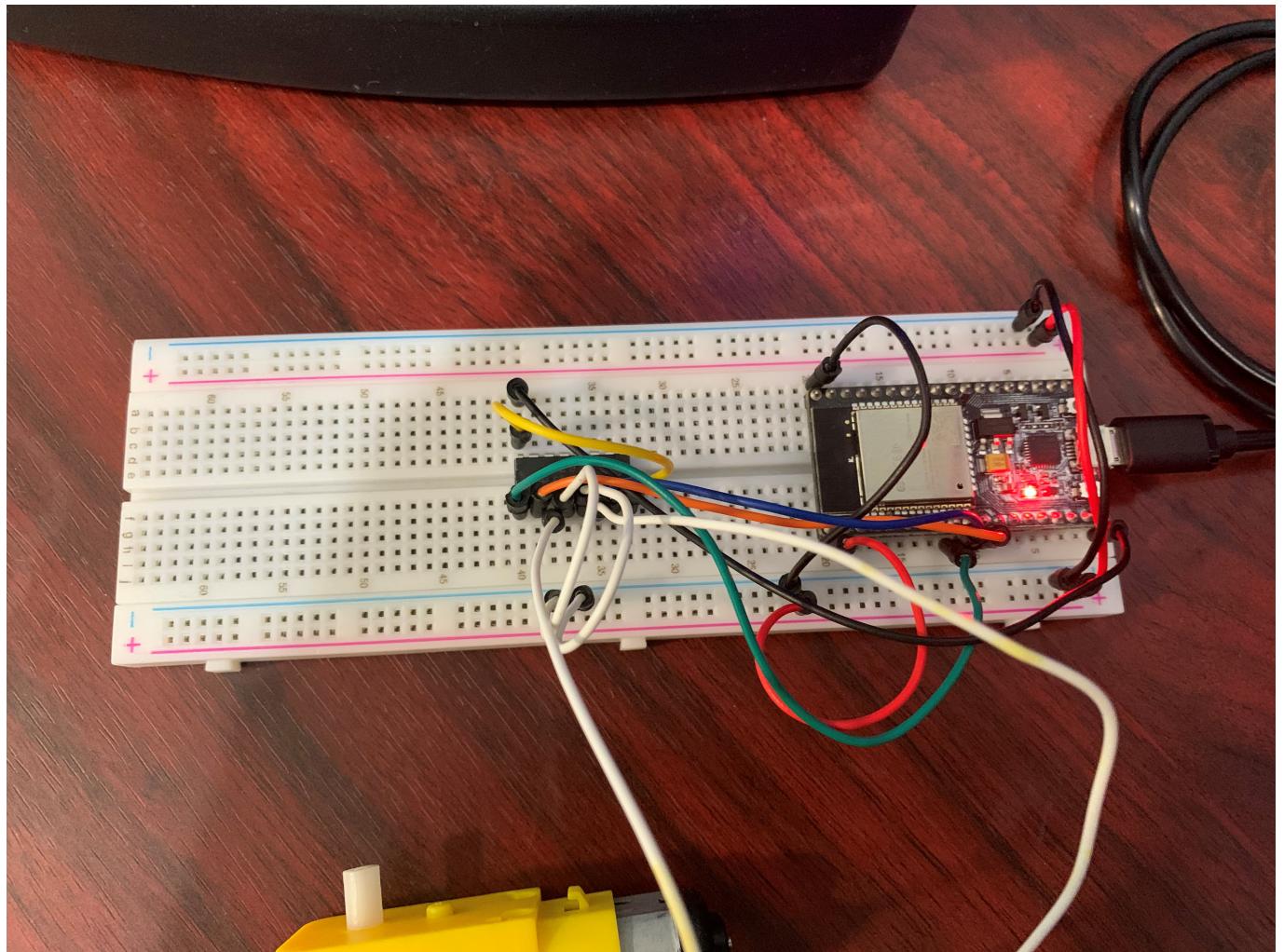


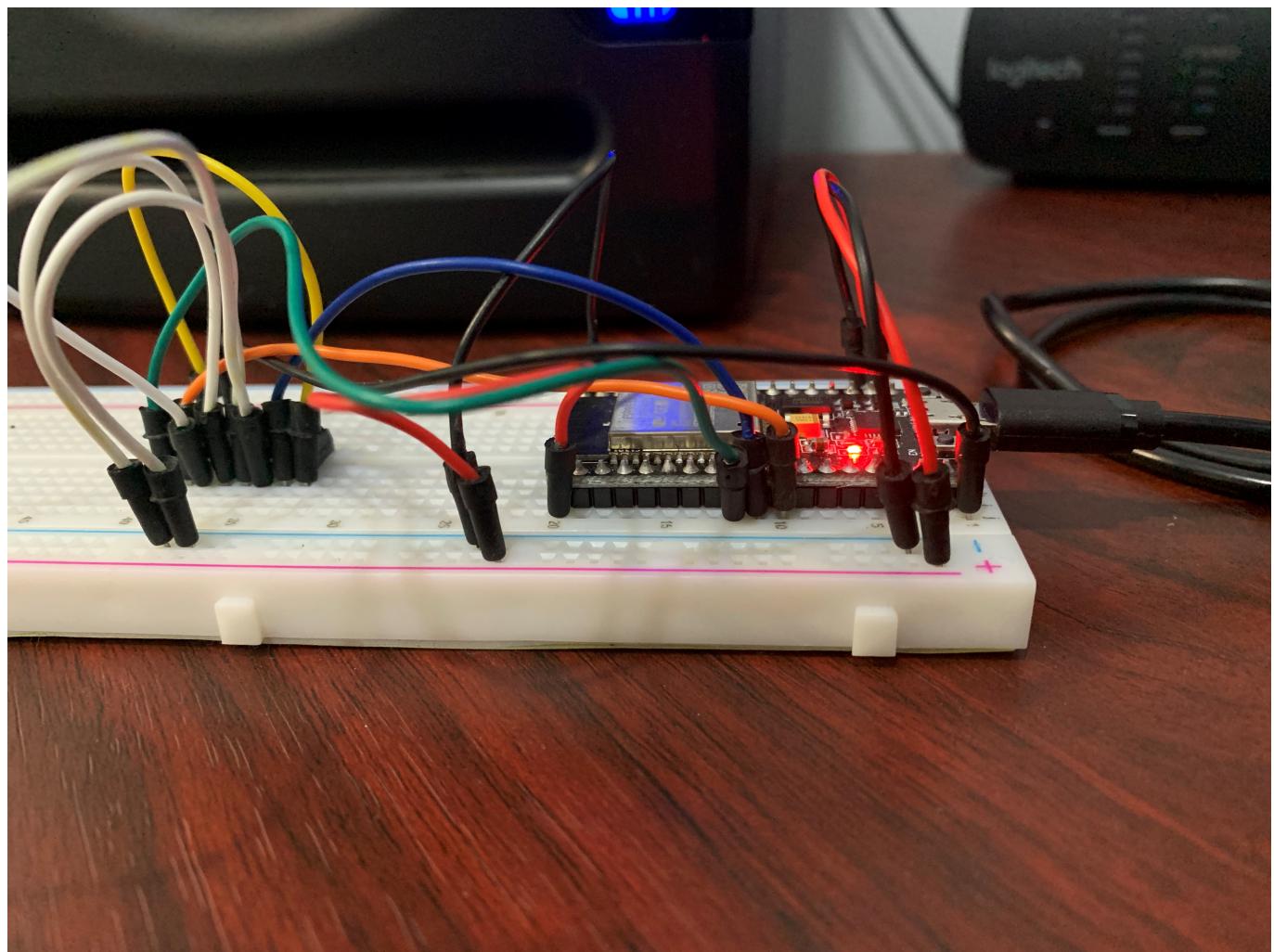
3. Coloque aquí la imagen del circuito ensamblado

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

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

Link de video a Youtube





```
#include <DabbleESP32.h>      // importacion de la libreria

const int motor1Pin1 = 27; // Declaracion de los pines
const int motor1Pin2 = 26;
const int enablePin = 25;

int StartMotor = 0;          // Tiempo de inicio
bool MotorOn = false;        // Variable boleana para encender y apagar el motor
int message = ' ';

void setup()
{
    // Se declaran las salidas del motor y el puente
    pinMode(motor1Pin1, OUTPUT);
    pinMode(motor1Pin2, OUTPUT);
    pinMode(enablePin, OUTPUT);

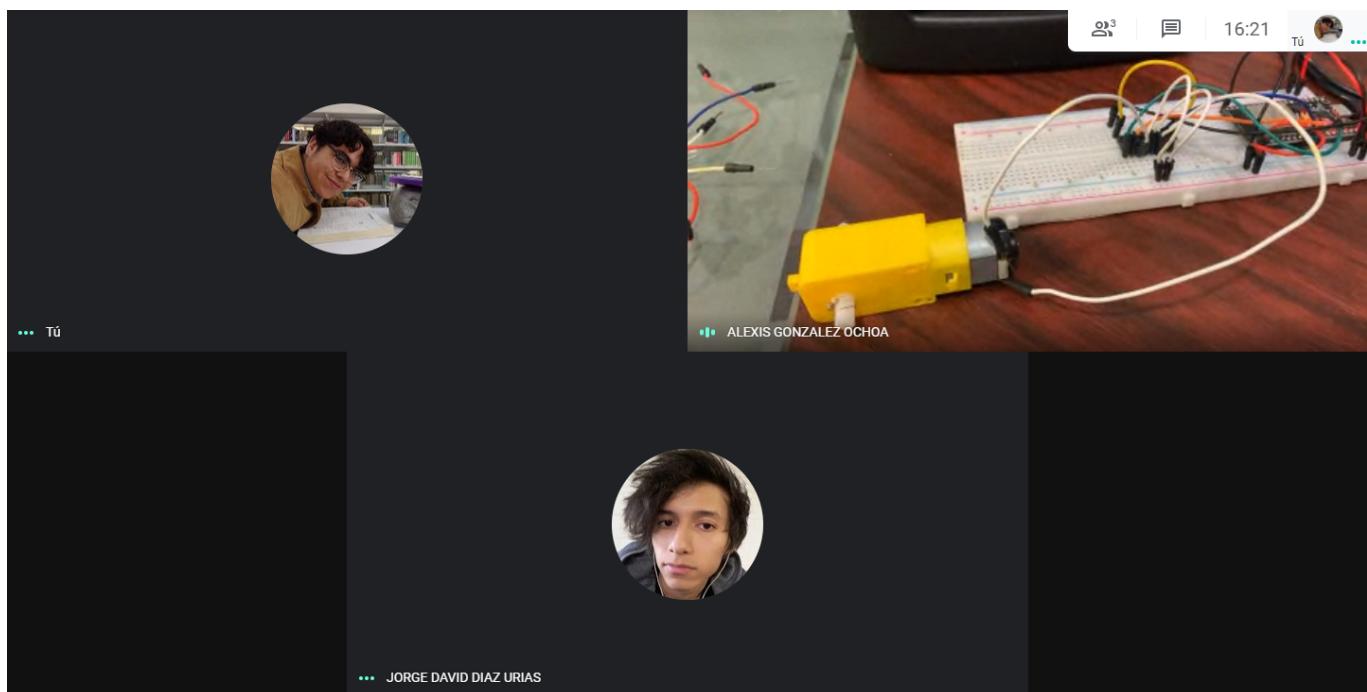
    Dabble.begin("MiESP32"); // Se declara dabble para la conexion
    Dabble.waitForAppConnection();
    Notification.clear();
    Terminal.print("Listo para usar");
    Serial.begin(115200);
}

void loop()
{
    Dabble.processInput(); // Se inicio dabble dentro del ciclo
    if(Terminal.available()) // Verifica si se mando algo en la terminal del celular
    {
        message = Terminal.read(); // Se le dara a messaje lo leido ya sea 1 o 0
        Serial.write(message); // Se escribe lo mandado en la terminal de la IDE
        if(message == '1') // Si es '1' se cambiara el valor bool a true
        { startime(); }
        else if(message == '0') // Si es '0' se dentendra
        { stoped(); } // Valor bool = false|
    }
    if(MotorOn) // Como el valor bool es true entrara aqui
    { statusM(); }
}

void startime() // Se le da a la variable millis para medir el tiempo
{
    StartMotor = millis(); // Y el valor booleano se cambia a true
    MotorOn = true;
}

void stoped() // Sirve para deneter el motor
{
    digitalWrite(enablePin, LOW);
    digitalWrite(motor1Pin1, LOW);
    digitalWrite(motor1Pin2, LOW);
    MotorOn = false; // Cambia el valor bool a false para que no se encienda el motor
}
```

```
void statusM()
{
    int elapsetime = millis() - StartMotor; // Se determina el tiempo desde que se encendio el motor
    if (elapsetime < 5000) // Si es menor a 5 mil este girara a direccion del reloj
    {
        digitalWrite(enablePin, HIGH);
        digitalWrite(motor1Pin1, HIGH);
        digitalWrite(motor1Pin2, LOW);
    }
    else if(elapsetime < 6000) // Si es menor a 6 mil este se detendra
    {
        digitalWrite(enablePin, LOW);
        digitalWrite(motor1Pin1, LOW);
        digitalWrite(motor1Pin2, LOW);
    }
    else if(elapsetime < 11000) // Si es menor a 11 mil este girara a contra el reloj
    {
        digitalWrite(enablePin, HIGH);
        digitalWrite(motor1Pin1, LOW);
        digitalWrite(motor1Pin2, HIGH);
    }
    else if(elapsetime < 12000) // Si es menor a 12 mil se detendra
    {
        digitalWrite(enablePin, LOW);
        digitalWrite(motor1Pin1, LOW);
        digitalWrite(motor1Pin2, LOW);
    }
    else // Cuando supere los 12 mil se volvera a iniciar
    { starttime(); }
}
```



```

// AlexisGonzalezOchoa_TheChallengers
// Motor DC con Bluetooth
// Control de velocidad del motor DC
// Utilizando el L293D H
// Motor DC con Bluetooth
// Control de velocidad del motor DC
// Utilizando el L293D H

#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11); // TX, RX

void setup() {
    mySerial.begin(9600); // Se establece la velocidad de baudios
    Serial.begin(9600); // Se establece la velocidad de baudios
}

void loop() {
    if (mySerial.available() > 0) { // Si se ha recibido un mensaje en la terminal
        String message = mySerial.readString(); // Se lee el mensaje que viene de la terminal
        if (message == "1") { // Si el mensaje es "1"
            digitalWrite(motorEnable1, HIGH); // Se enciende el motor
            digitalWrite(motorEnable2, LOW);
        } else if (message == "0") { // Si el mensaje es "0"
            digitalWrite(motorEnable1, LOW); // Se apaga el motor
            digitalWrite(motorEnable2, HIGH);
        }
    }
}

void startMotor() {
    digitalWrite(motorEnable1, HIGH); // Se enciende el motor
    digitalWrite(motorEnable2, LOW);
    MotorSpeed = 100;
}

void stopMotor() {
    digitalWrite(motorEnable1, LOW); // Se apaga el motor
    digitalWrite(motorEnable2, HIGH);
    MotorSpeed = 0;
}

void setSpeed() {
    if (millis() - startMotorTime > 5000) { // Si ha pasado mas de 5 segundos
        if (MotorSpeed < 1000) { // Si es menor a 1000
            digitalWrite(motorEnable1, HIGH);
            digitalWrite(motorEnable2, LOW);
            digitalWrite(motorPwm1, HIGH);
            digitalWrite(motorPwm2, HIGH);
        } else if (MotorSpeed < 10000) { // Si es menor a 10000
            digitalWrite(motorEnable1, HIGH);
            digitalWrite(motorEnable2, LOW);
            digitalWrite(motorPwm1, HIGH);
            digitalWrite(motorPwm2, LOW);
        } else if (MotorSpeed < 100000) { // Si es menor a 100000
            digitalWrite(motorEnable1, HIGH);
            digitalWrite(motorEnable2, LOW);
            digitalWrite(motorPwm1, HIGH);
            digitalWrite(motorPwm2, LOW);
        }
    }
}

void stopAllMotors() {
    digitalWrite(motorEnable1, LOW);
    digitalWrite(motorEnable2, HIGH);
}

```

6. Conclusiones

- Alexis Gonzalez Esta actividad realizada me sirvio para darme cuenta del uso que tienen las funciones y de lo utiles que pueden ser dentro de arduino. El manejo del motorDC en conjunto con el puente L293D H para poder hacer uso del motor, y con el mismo poder hacer que gire o que pare por completo y ajustar la velocidad del mismo. Fue un tanto divertido por que tuve que investigar bastante y me encontre con foros donde hacian uso de tareas multiples. Pero era mas complicado por lo que se opto por hacer funciones permitiendo que el void loop siguiera su ciclo despues de haber ejecutado alguno de las funciones.
- Jorge Diaz Con la realización de esta práctica se pudo ver el funcionamiento del esp 32 con un motor DC, al hacer uso de estos motores podemos generar movimiento, una principal característica de estos motores es la velocidad la cual nos da muchas opciones de como aplicar su funcionamiento y con la ayuda del esp32 y su función de bluetooth podemos controlar el arranque y apagado del motor con un dispositivo así como también manipular el sentido de rotación y así con el esp32 y el uso de sus pines GPIO y con la programación de este podemos manipular su sentido ya sea en un sentido o en dos como se aplicó en esta práctica.

-Julio Jimenez En conclusión la práctica realizada me enseña y sigue sorprendiendo con todas las posibles funciones que tiene el ESP32, pero está en específico el poder controlar de manera precisa un motor CC, esto manejando ciclos,direcciones, detener y avanzar todo este manejo desde una aplicación y bluetooth. Esto me viene a la idea de cuantas aplicaciones en el campo de la vida real podemos realizar con esta forma de manipular un motor mediante bluetooth, y no solo un motor sino tambien cualquier otras herramientas.

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

 [Ir a GitHub de Julio Jimenez](#)

 [Ir a GitHub de Jorge Diaz](#)

 [Ir a GitHub de Alexis Gonzalez](#)