



## C3.7 Programación Microcontrolador NodeMCU ESP32

---

Arduino y modulador de ancho de pulso PWM con salida analógica

---



### Instrucciones

- De acuerdo con la información presentada por el asesor referente al tema, desarrollar lo que se indica dentro del apartado siguiente.
- 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.
- Es requisito que el archivo .md contenga una etiqueta del enlace al repositorio de su documento en Github, por ejemplo **Enlace a mi GitHub**
- Al concluir el reto el reto se deberá subir a github el archivo .md creado.
- Desde el archivo .md se debe exportar un archivo .pdf con la nomenclatura **C3.7\_NombreAlumno\_Equipo.pdf**, el cual deberá subirse a classroom dentro de su apartado correspondiente, para que sirva como evidencia de su entrega; siendo esta plataforma **oficial** aquí se recibirá la calificación de su actividad por individual.
- Considerando que el archivo .pdf, fue obtenido desde archivo .md, ambos deben ser idénticos y mostrar el mismo contenido.
- 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.

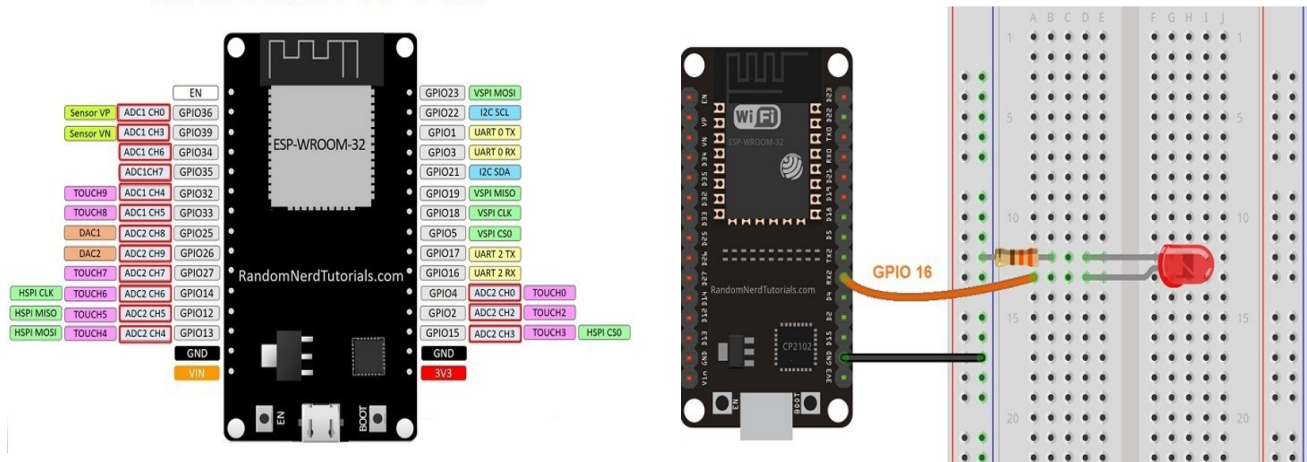
```
| readme.md
| | blog
| | | C3.1_TituloActividad.md
| | | C3.2_TituloActividad.md
| | | C3.3_TituloActividad.md
| | | C3.4_TituloActividad.md
| | | C3.5_TituloActividad.md
| | | C3.6_TituloActividad.md
| | | C3.7_TituloActividad.md
| | | C3.8_TituloActividad.md
| | img
| | docs
| | | A3.1_TituloActividad.md
| | | A3.2_TituloActividad.md
```



## Desarrollo

1. Ensamble el circuito mostrado en la figura siguiente.

### ESP32 DEVKIT V1 - DOIT



2. Analice y escriba el programa que se muestra a continuación.

```
// the number of the LED pin
const int ledPin = 16; // 16 corresponds to GPIO16

// setting PWM properties
const int freq = 5000;
const int ledChannel = 0;
const int resolution = 8;

void setup(){
  // configure LED PWM functionalites
  ledcSetup(ledChannel, freq, resolution);

  // attach the channel to the GPIO to be controlled
  ledcAttachPin(ledPin, ledChannel);
}

void loop(){
  // increase the LED brightness
  for(int dutyCycle = 0; dutyCycle <= 255; dutyCycle++){
    // changing the LED brightness with PWM
    ledcWrite(ledChannel, dutyCycle);
    delay(15);
  }

  // decrease the LED brightness
  for(int dutyCycle = 255; dutyCycle >= 0; dutyCycle--){
    // changing the LED brightness with PWM
    ledcWrite(ledChannel, dutyCycle);
    delay(15);
  }
}
```

Then, you set the PWM signal properties. You define a frequency of 5000 Hz, choose channel 0 to generate the signal, and set a resolution of 8 bits. You can choose other properties, different than these, to generate different PWM signals.

In the `setup()`, you need to configure LED PWM with the properties you've defined earlier by using the `ledcSetup()` function that accepts as arguments, the `ledChannel`, the frequency, and the resolution, as follows:

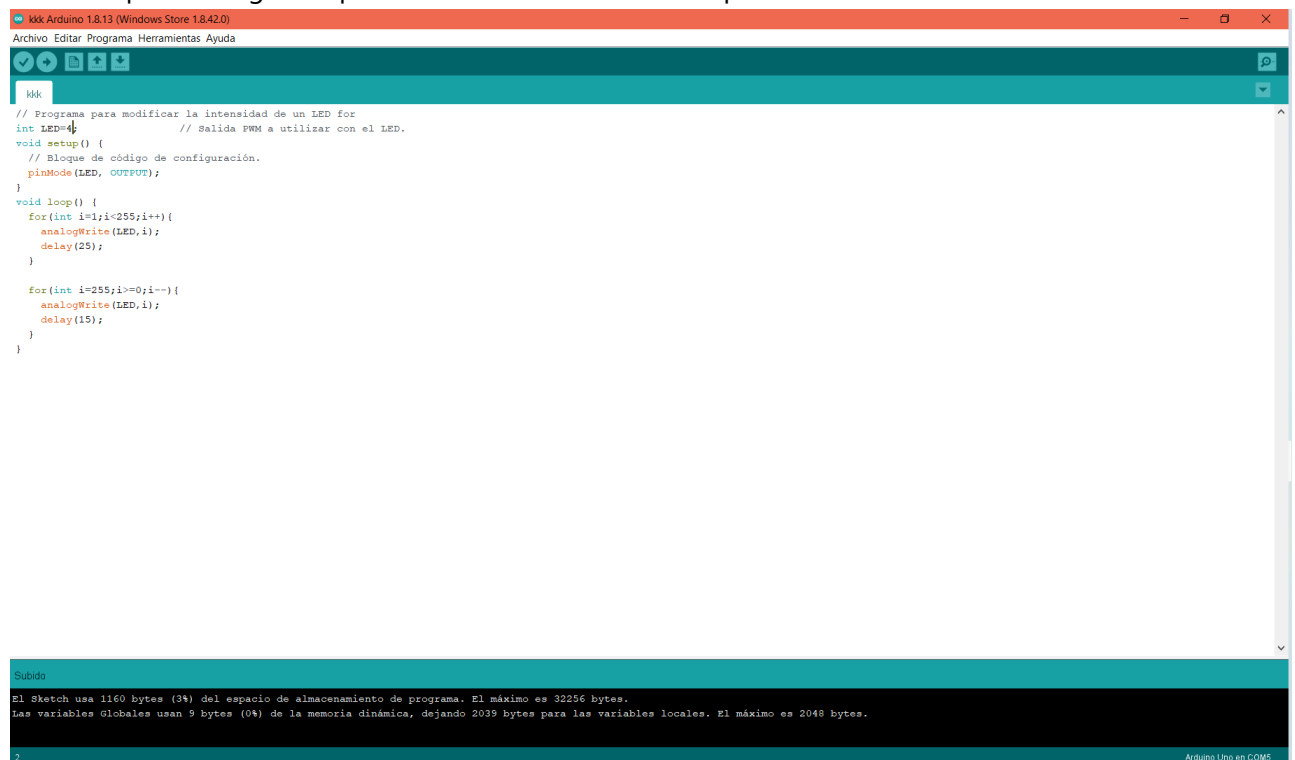
In the loop, you'll vary the duty cycle between 0 and 255 to increase the LED brightness.

And then, between 255 and 0 to decrease the brightness.

To set the brightness of the LED, you just need to use the `ledcWrite()` function that accepts as arguments the channel that is generating the signal, and the duty cycle.

Fuente de consulta: [Random Nerd Tutorials](https://www.randomnerdtutorials.com/)

### 3. Inserte aquí las imágenes que considere como evidencias para demostrar el resultado obtenido.



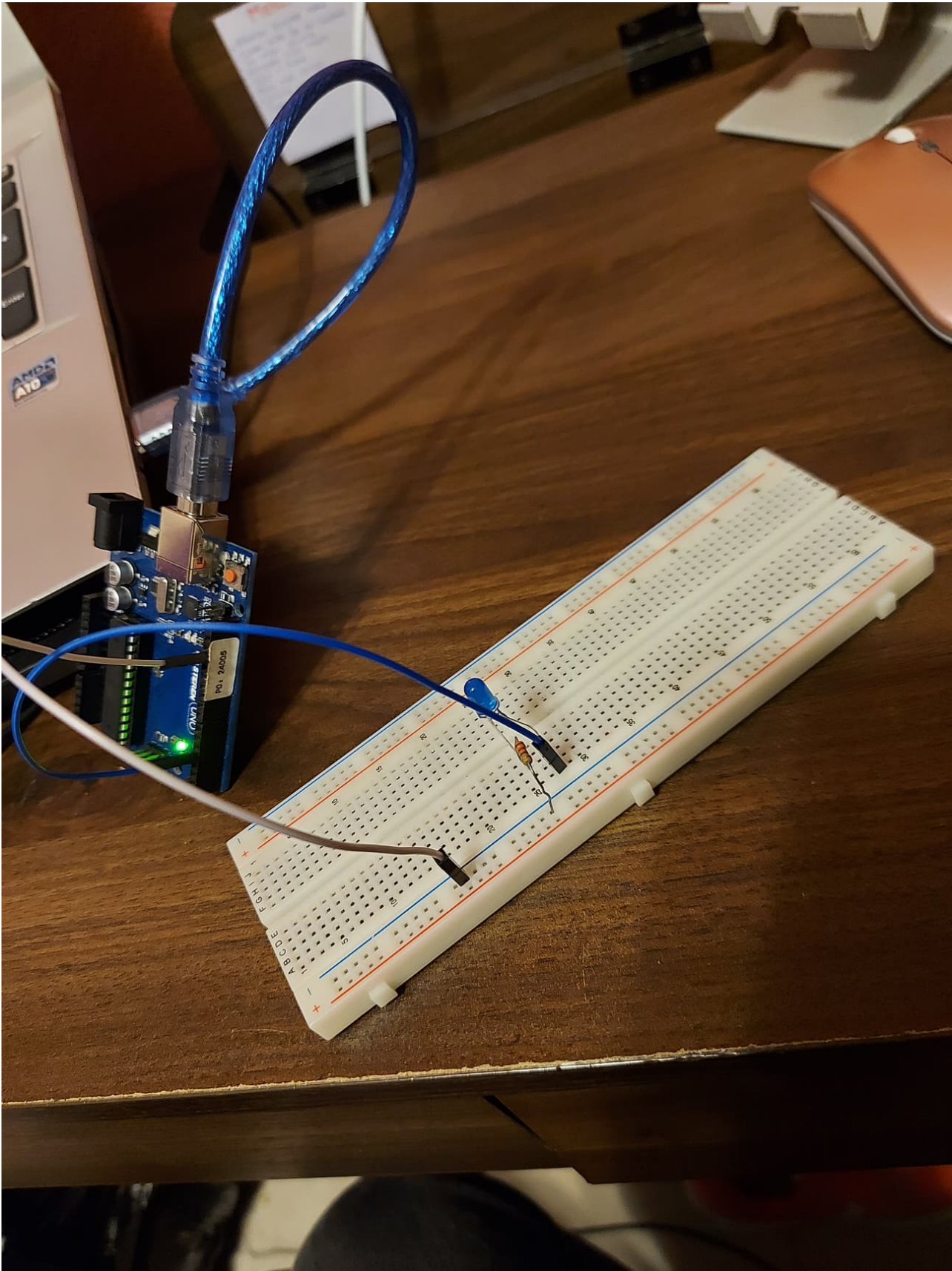
```
Ardu Arduino 1.8.13 (Windows Store 1.8.42.0)
Archivo Editar Programa Herramientas Ayuda

// Programa para modificar la intensidad de un LED for
int LED=4; // Salida PWM a utilizar con el LED.
void setup() {
  // Bloque de código de configuración.
  pinMode(LED, OUTPUT);
}
void loop() {
  for(int i=1;i<255;i++){
    analogWrite(LED,i);
    delay(25);
  }

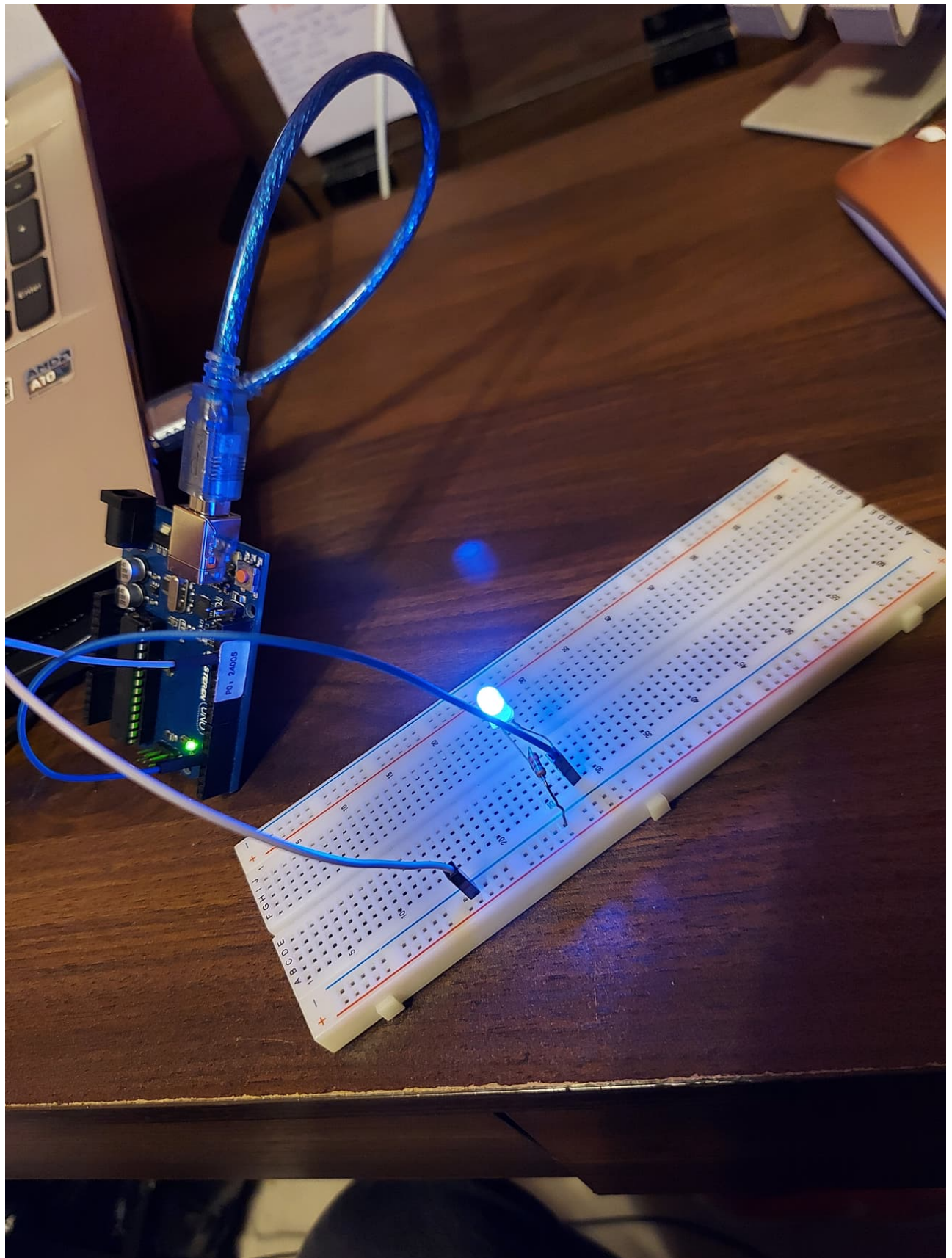
  for(int i=255;i>=0;i--){
    analogWrite(LED,i);
    delay(15);
  }
}

Subido
El Sketch usa 1160 bytes (3%) del espacio de almacenamiento de programa. El máximo es 32256 bytes.
Las variables Globales usan 9 bytes (0%) de la memoria dinámica, dejando 2039 bytes para las variables locales. El máximo es 2048 bytes.

Arduino Uno en COM5
```







---

### Nota

Buen dia profesor, nuestro ESP8266 no lo agarro mi computadora descargue los controladores y todo y no lo agarro al igual que las librerias y el gestor de tarjeta y nada es por eso que lo tuve que hacer en arduino adaptando el codigo un poco el ESP32 nos llegara el dia "4 de noviembre ya las siguientes practicas las realizaremos bien.

Una disculpa.



Rubrica

Criterios	Descripción	Puntaje
Instrucciones	Se cumple con cada uno de los puntos indicados dentro del apartado Instrucciones?	20
Desarrollo	Se respondió a cada uno de los puntos solicitados dentro del desarrollo de la actividad?	80



[Link a GitHub](#)