# **Z** 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.

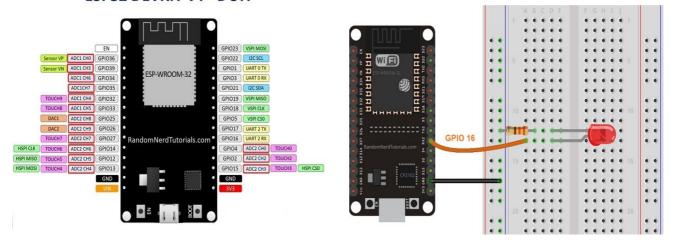
```
readme.md
 blog
 | C3.1 TituloActividad.md
  C3.2_TituloActividad.md
 | C3.3 TituloActividad.md
   C3.4 TituloActividad.md
  C3.5_TituloActividad.md
  C3.6 TituloActividad.md
 C3.7_TituloActividad.md
 | C3.8_TituloActividad.md
| | img
 docs
 | A3.1_TituloActividad.md
 | A3.2_TituloActividad.md
```



## Desarrollo

1. Ensamble el circuito mostrado en la figura siguiente.

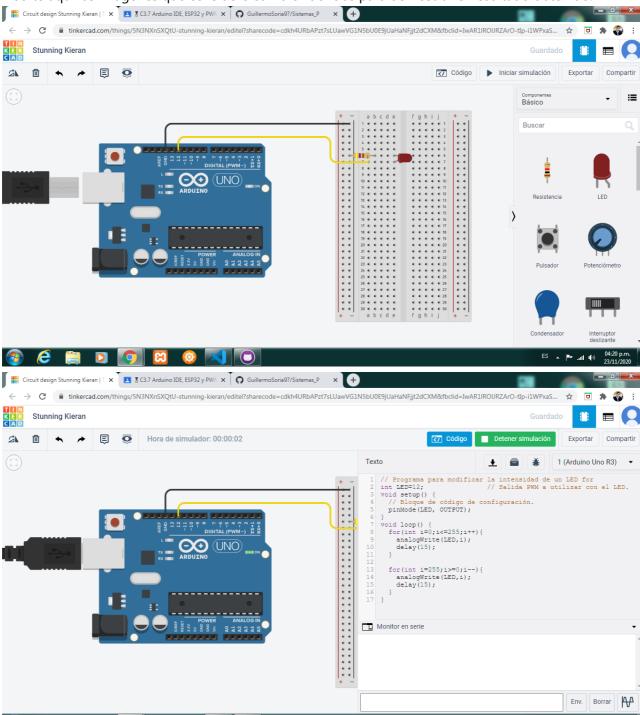
### **ESP32 DEVKIT V1 - DOIT**

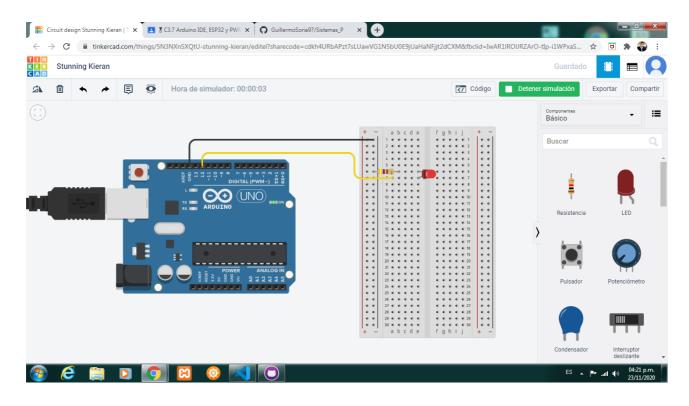


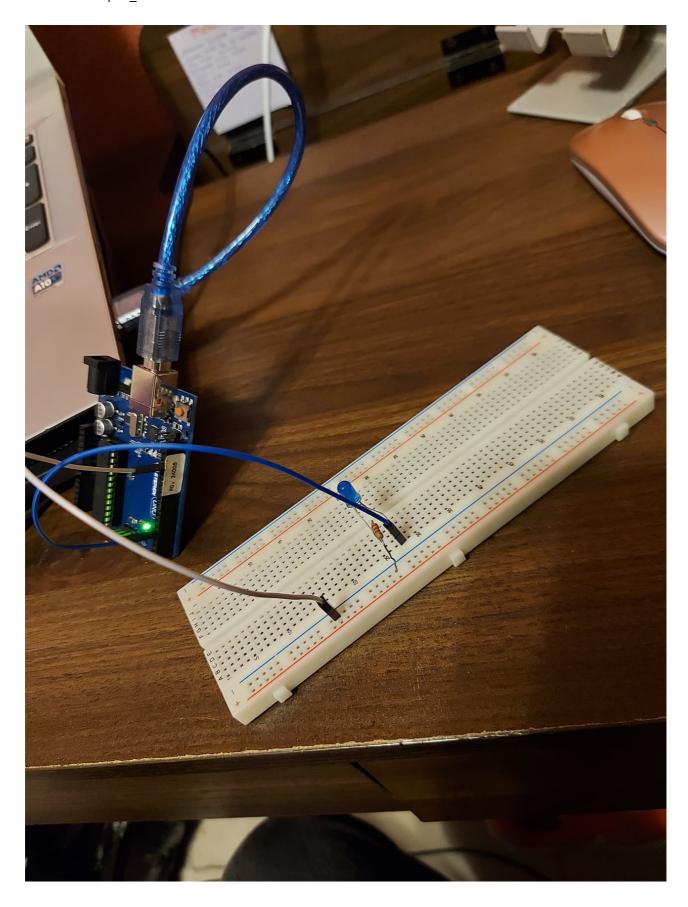
2. Analice y escriba el programa que se muestra a continuación.

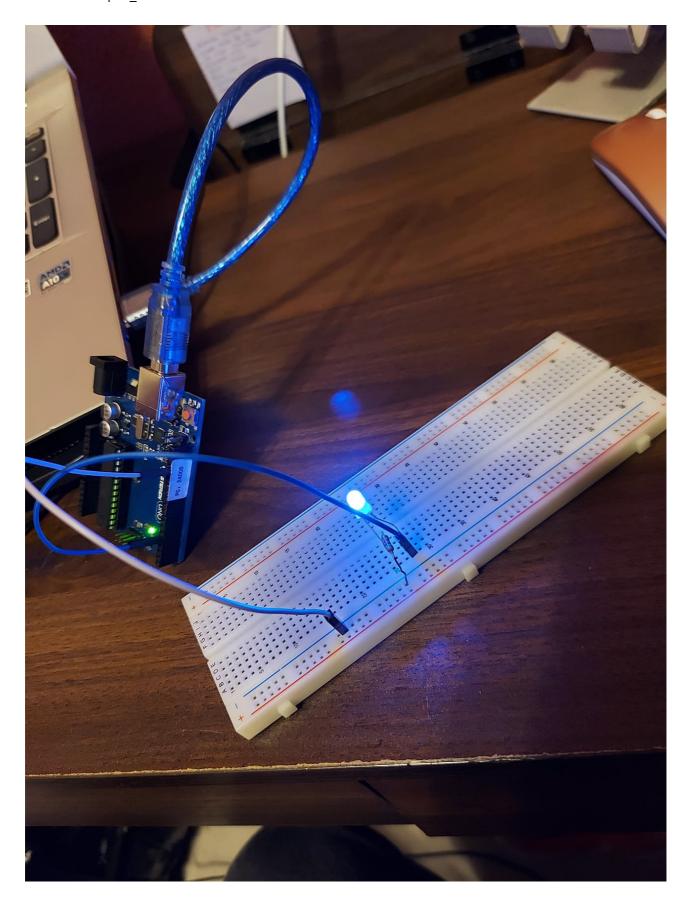
```
// the number of the LED pin
const int ledPin = 16; // 16 corresponds to GPI016
// setting PWM properties
const int freq = 5000;
                                                      Then, you set the PWM signal properties. You define a frequency of 5000 Hz,
                                                      choose channel 0 to generate the signal, and set a resolution of 8 bits. You can
const int ledChannel = 0;
                                                       choose other properties, different than these, to generate different PWM signals.
const int resolution = 8;
void setup(){
                                                       In the setup(), you need to configure LED PWM with the properties you've
  // configure LED PWM functionalitites
                                                       defined earlier by using the ledcSetup() function that accepts as arguments, the
                                                       ledChannel, the frequency, and the resolution, as follows:
  ledcSetup(ledChannel, freq, resolution);
  // attach the channel to the GPIO to be controlled
  ledcAttachPin(ledPin, ledChannel);
                                                       In the loop, you'll vary the duty cycle between 0 and 255 to increase the LED
void loop(){
                                                       brightness
 // increase the LED brightness
  for(int dutyCycle = 0; dutyCycle <= 255; dutyCycle++){</pre>
    // changing the LED brightness with PWM
    ledcWrite(ledChannel, dutyCycle);
    delay(15);
                                                      And then, between 255 and 0 to decrease the brightness.
  // decrease the LED brightness
  for(int dutyCycle = 255; dutyCycle >= 0; dutyCycle--){
    // changing the LED brightness with PWM
                                                       To set the brightness of the LED, you just need to use the <code>ledcWrite()</code> function
    ledcWrite(ledChannel, dutyCycle);
                                                      that accepts as arguments the channel that is generating the signal, and the duty
    delav(15):
                                                      cycle.
```

Fuente de consulta: Random Nerd Tutorials











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

**A** Enlace a mi GitHub