



C3.6 Programación Microcontrolador NodeMCU ESP32

Arduino y entrada analógica, utilizando un potenciómetro



Instrucciones

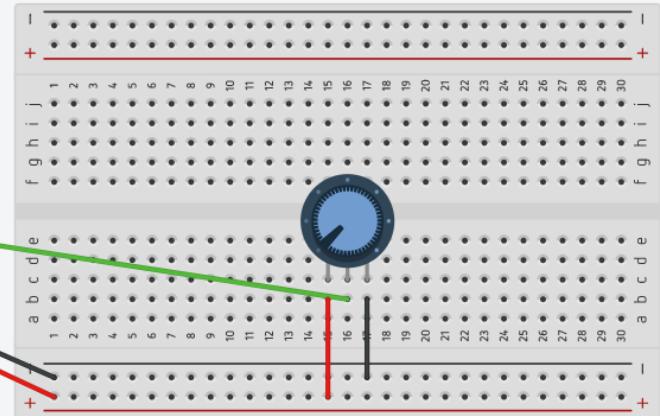
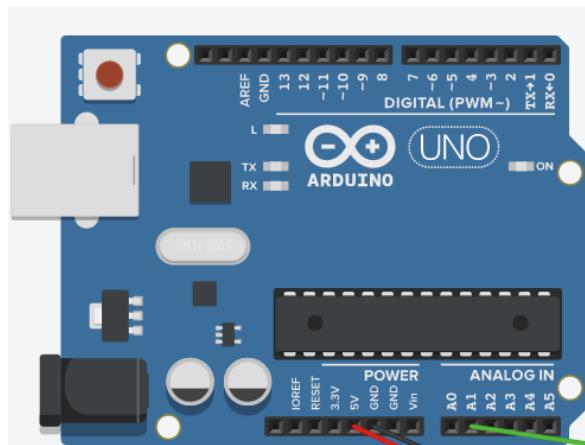
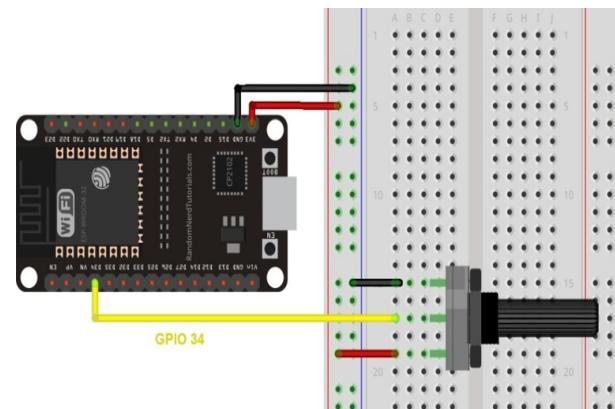
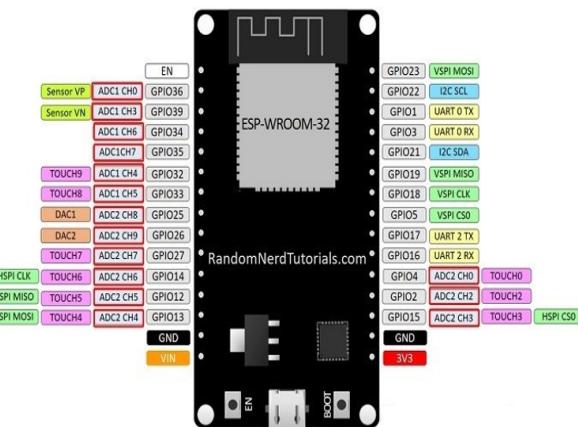
- De acuerdo con la información presentada por el asesor referente al tema, desarrollar lo que se indica dentro del apartado siguiente.
- Toda actividad o reto se deberá realizar utilizando el estilo **MarkDown con extensión .md** y el entorno de desarrollo VSCode, debiendo ser elaborado como un documento **single page**, es decir si el documento cuenta 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 **C3.6_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 índice, 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  
| | img  
| | docs  
| | | A3.1_TituloActividad.md  
| | | A3.2_TituloActividad.md
```

Desarrollo

1. Ensamble el circuito mostrado en la figura siguiente.

ESP32 DEVKIT V1 - DOIT



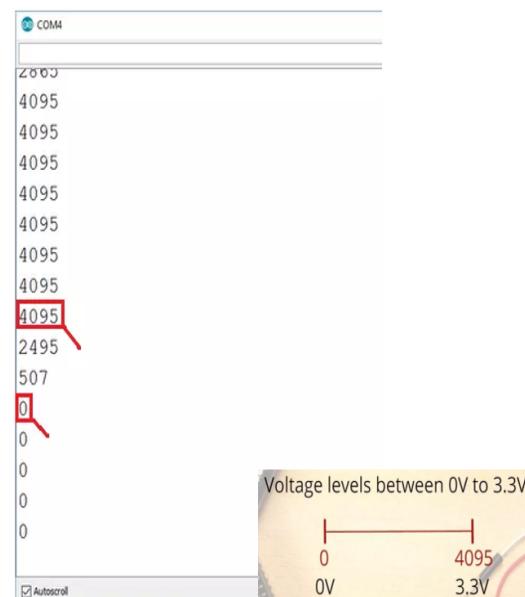
2. Analice y escriba el programa que se muestra a continuación.

```
// Potentiometer is connected to GPIO 34 (Analog ADC1_CH6)
const int potPin = 34;

// variable for storing the potentiometer value
int potValue = 0;

void setup() {
  Serial.begin(115200);
  delay(1000);
}

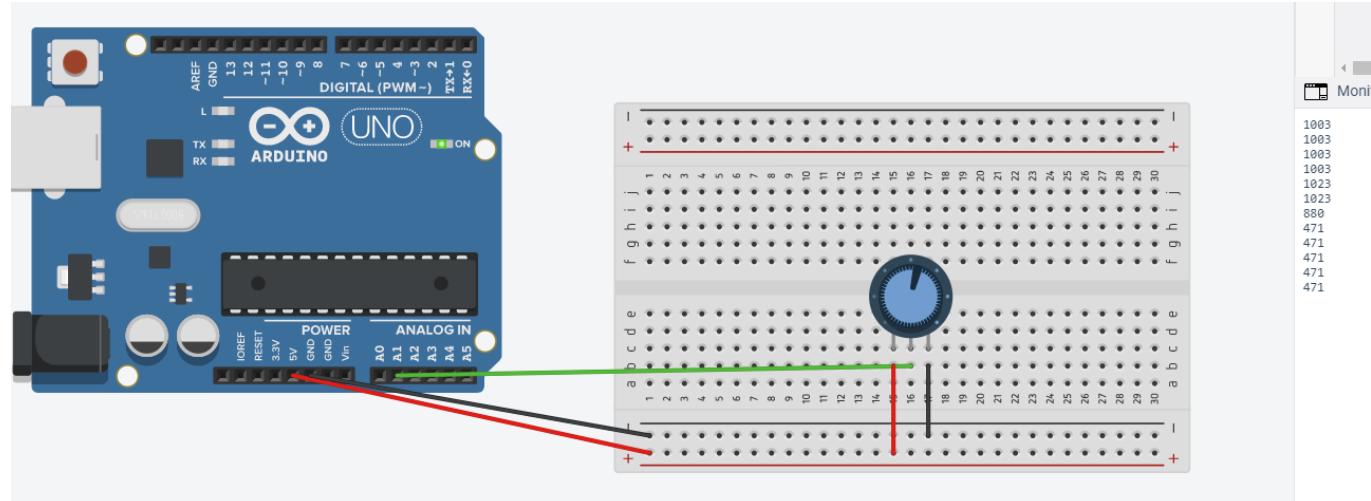
void loop() {
  // Reading potentiometer value
  potValue = analogRead(potPin);
  Serial.println(potValue);
  delay(500);
}
```



Fuente de consulta: [Random Nerd Tutorials](#)

```
1 const int potPin = A1; //el potenciómetro va conectado al pin analógico 1 de Arduino
2 int potValue = 0; //se declara una variable para guardar el valor del potenciómetro
3
4
5 void setup()
6 {
7   Serial.begin(115200); //se inicia la comunicación serial a 115200 bits por segundo
8   delay(1000); // ocurre un delay de 1 segundo
9 }
10
11 void loop() //método loop que se repite indefinidamente
12 {
13   //se lee el valor de potenciómetro
14   potValue = analogRead(potPin); //el valor leído se almacena en potValue
15   Serial.println(potValue); //se imprime el valor leído del potenciómetro
16   delay(500); //se añade un delay de 500 microsegundos para evitar tener problemas al cambiar el valor del potenciómetro
17 }
```

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



4. Evidencias físicas del ESP32 realizadas por Morgado Jacome Eduardo:

The screenshot shows the Arduino IDE interface. The top menu bar includes 'Archivo', 'Editar', 'Programa', 'Herramientas', and 'Ayuda'. The title bar says 'ESP32-Potenciómetro Arduino 1.8.13 (Windows Store 1.8.42.0)'. The main area contains the following code:

```
//Declaracion de variables
const int potenPin = 34;
int potenValor = 0;

void setup() {
  Serial.begin(9600);
  delay(1000);

}

void loop() {
  //Se lee lo que esta recibiendo el pin GPIO34
  potenValor = analogRead(potenPin);
  //Se imprime en pantalla la equivalencia en bits el
  //voltaje de 0 a 3.3V
  Serial.println(potenValor);
  delay(500);
}
```

To the right, the serial monitor window titled 'COM7' displays the following data:

Value
4095
4095
3919
3167
2275
1670
1457
1005
600
193
0
0
0
0
0

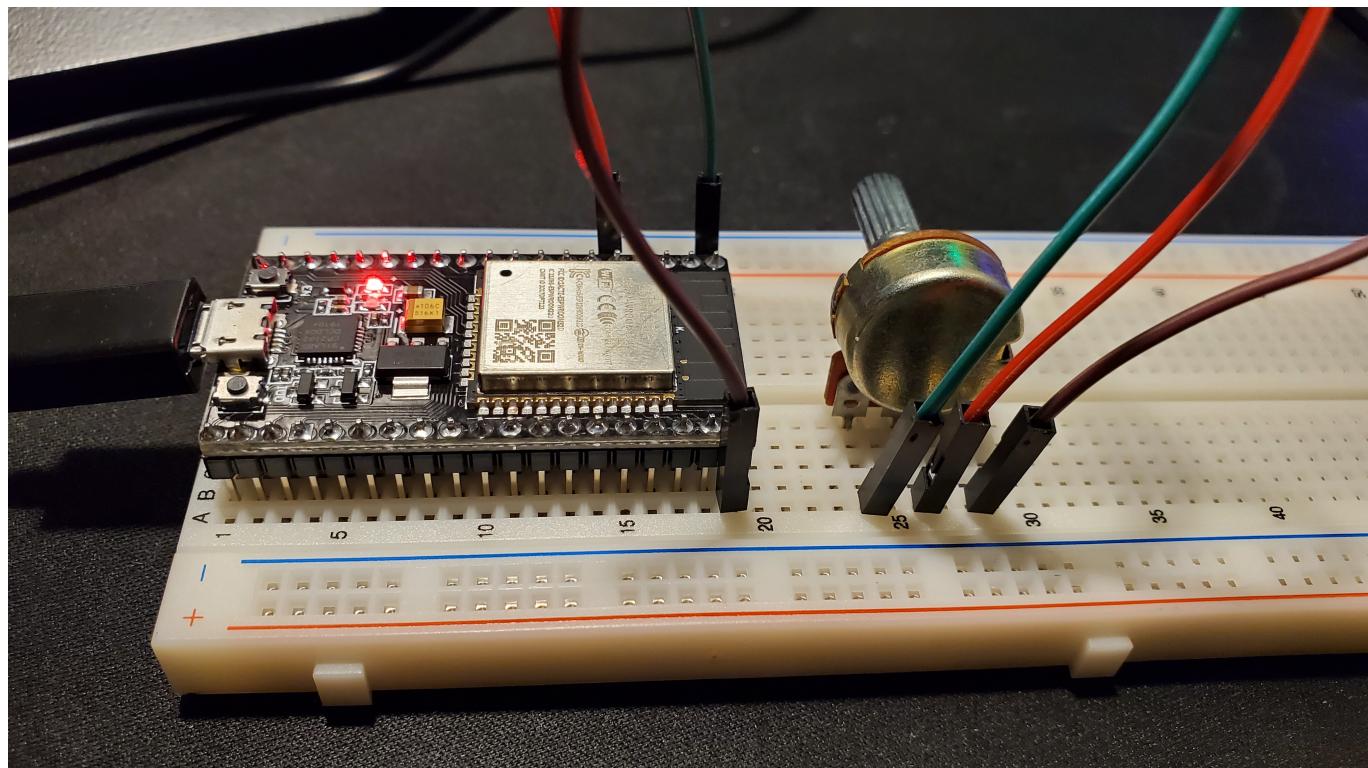
At the bottom of the serial monitor, there are checkboxes for 'Autoscroll' and 'Mostrar marca temporal'.

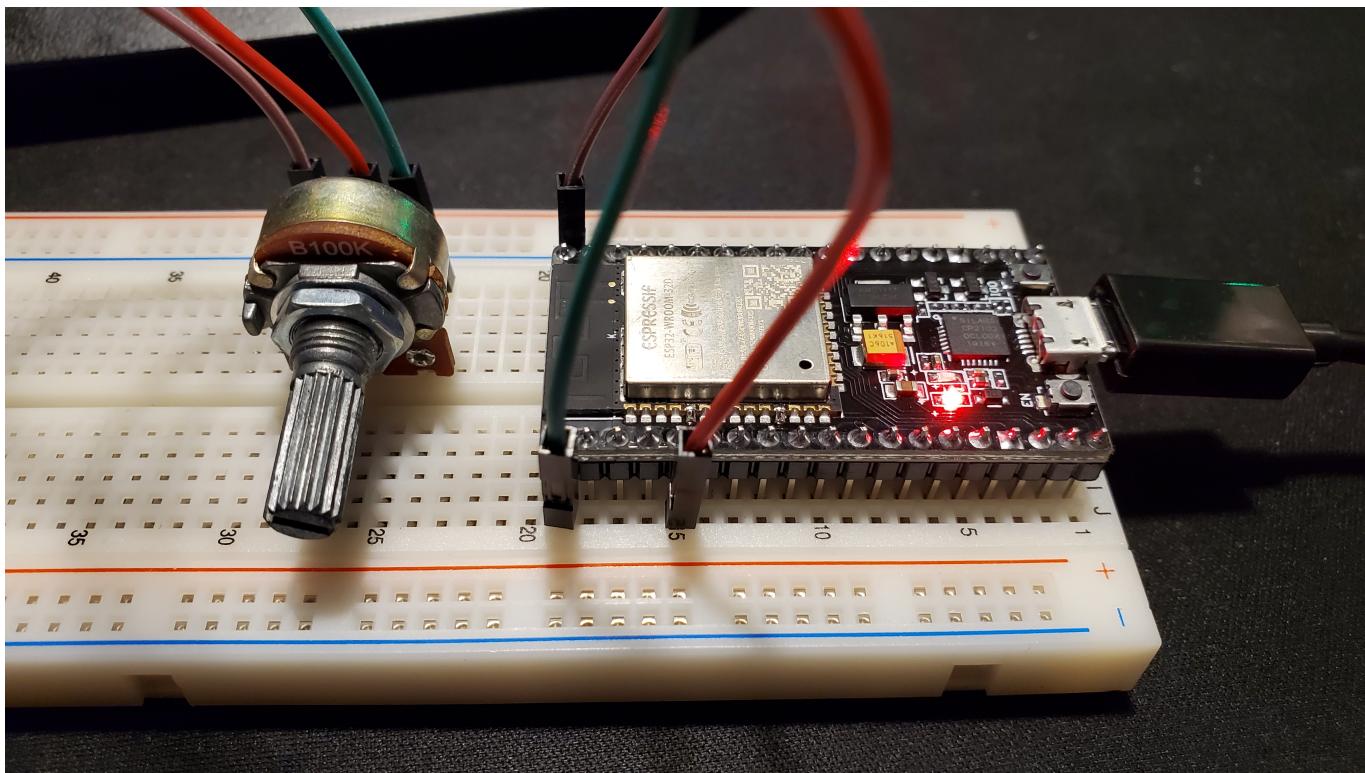
The bottom part of the IDE shows the terminal output:

```
Hash of data verified.
Compressed 3072 bytes to 128...
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.1 seconds (effective 409.6 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

The status bar at the bottom indicates '15' and 'ESP32 Dev Module en COM7'.





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

[Mi repositorio de Github](#)