



C3.8 Programación Microcontrolador NodeMCU ESP32

Arduino y sensor de tacto integrado al NodeMCU ESP32



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 **C3.8_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.

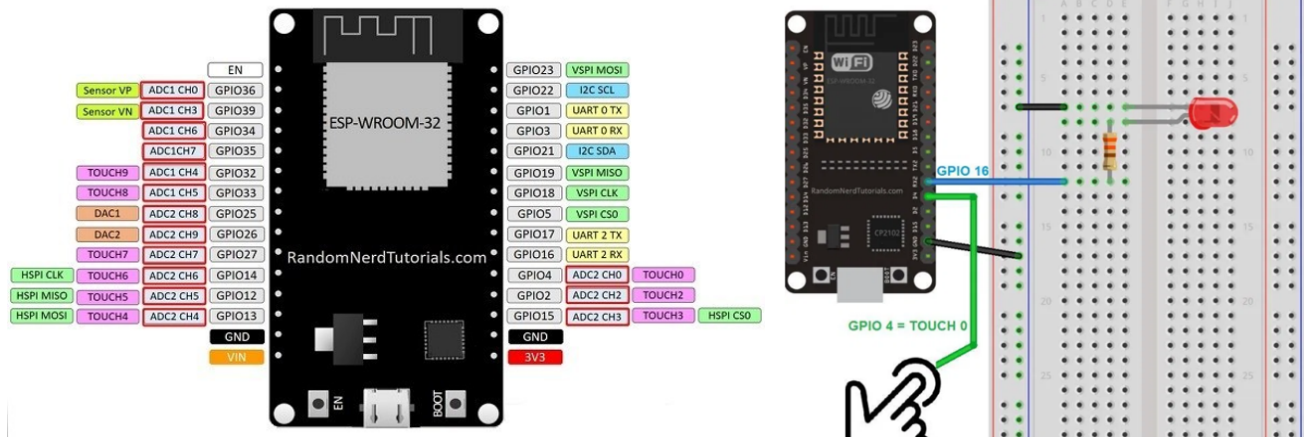
```
| readme.md
| | blog
| | | C3.1_TituloActividad.md
| | | C3.2_TituloActividad.md
| | | C3.3_TituloActividad.md
| | | C3.4_TituloActividad.md
| | | C3.5_TituloActividad.md
| | | C3.6_TituloActividad.md
| | | C3.7_TituloActividad.md
| | | C3.8_TituloActividad.md
| | img
| | docs
| | | A3.1_TituloActividad.md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



Desarrollo

1. Basado en el siguiente circuito y ensamblarlo, utilizando alguno de los simulados propuesto, explicando el resultado que se desea obtener del mismo.

ESP32 DEVKIT V1 - DOIT



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

```
// set pin numbers
const int touchPin = 4;
const int ledPin = 16;

// change with your threshold value
const int threshold = 20;
// variable for storing the touch pin value
int touchValue;

void setup(){
  Serial.begin(115200);
  delay(1000); // give me time to bring up serial monitor
  // initialize the LED pin as an output:
  pinMode (ledPin, OUTPUT);
}

void loop(){
  // read the state of the pushbutton value:
  touchValue = touchRead(touchPin);
  Serial.print(touchValue);
  // check if the touchValue is below the threshold
  // if it is, set ledPin to HIGH
  if(touchValue < threshold){
    // turn LED on
    digitalWrite(ledPin, HIGH);
    Serial.println(" - LED on");
  }
  else{
    // turn LED off
    digitalWrite(ledPin, LOW);
    Serial.println(" - LED off");
  }
  delay(500);
}
```

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

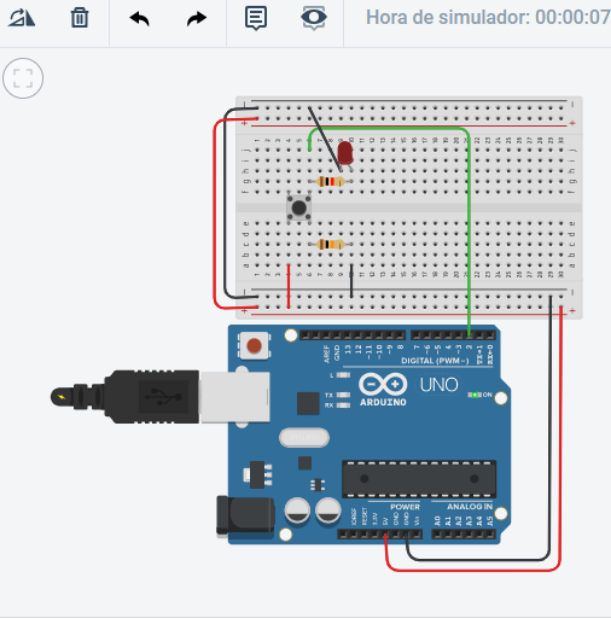
En realidad el código que use es mucho más sencillo dado a que use un botón en vez de el sensor de touch para que realizara la misma función y el resultado fuera algo similar.

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

TIN
KER
CAD

Brave Hillar-Blad

Se han guardado todos los cambios.



Hora de simulador: 00:00:07

Código

Detener simulación

Exportar

Compartir

Texto

1 (Arduino Uno R3)

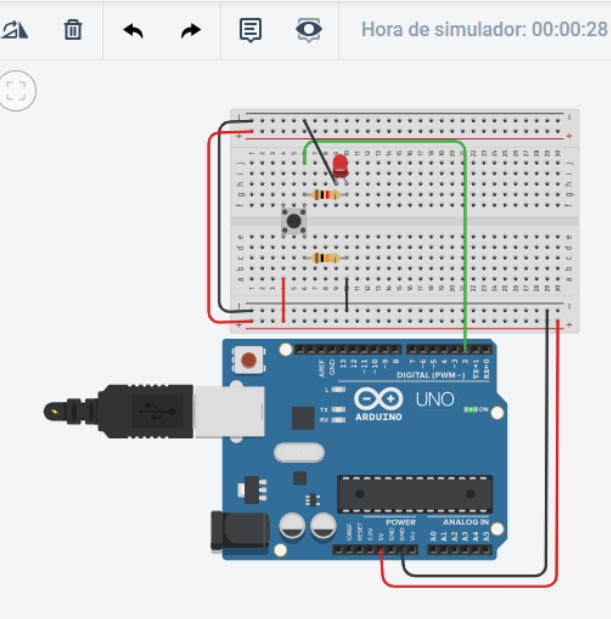
```
1
2 int pushBotton = 2;
3
4 void setup()
5 {
6   Serial.begin(9600);
7   pinMode(pushBotton, INPUT);
8 }
9
10 void loop()
11 {
12   int buttonState = digitalRead(pushBotton); Serial.println(buttonState);
13   delay(100);
14 }
```

Monitor en serie

TIN
KER
CAD

Brave Hillar-Blad

Se han guardado todos los cambios.



Hora de simulador: 00:00:28

Código

Detener simulación

Exportar


Compartir


Texto

1 (Arduino Uno R3)

```
1
2 int pushBotton = 2;
3
4 void setup()
5 {
6   Serial.begin(9600);
7   pinMode(pushBotton, INPUT);
8 }
9
10 void loop()
11 {
12   int buttonState = digitalRead(pushBotton); Serial.println(buttonState);
13   delay(100);
14 }
```

Monitor en serie

 [Ir a repositorio](#)

 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

3 / 4

 [Ir a readme](#)