UNIVERSIDAD EIA®

Ser, Saber y Servir

Con Acreditación Institucional

ESP32 - ARDUINO

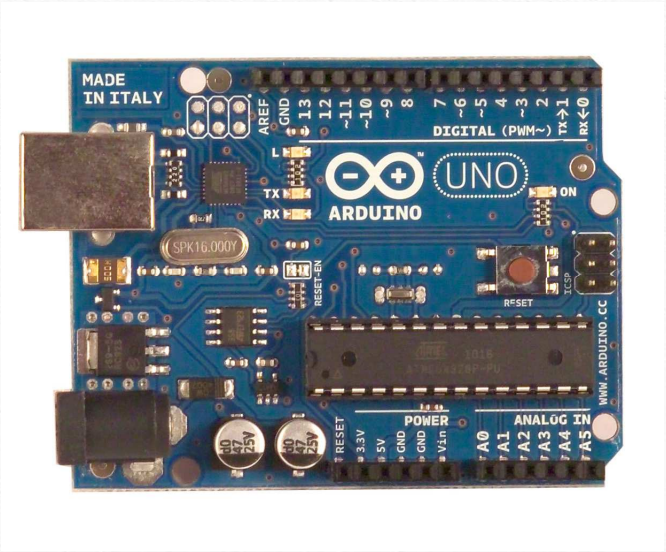
YEISON JAVIER MONTAGUT FERIZZOLA

¿QUÉ ES ARDUINO?

ARDUINO

¿QUÉ ES ARDUINO?

Es una plataforma o sistema de desarrollo, que une Hardware y Software con el objetivo de permitir mayor facilidad y rapidez en el diseño de aplicaciones basadas en µC.



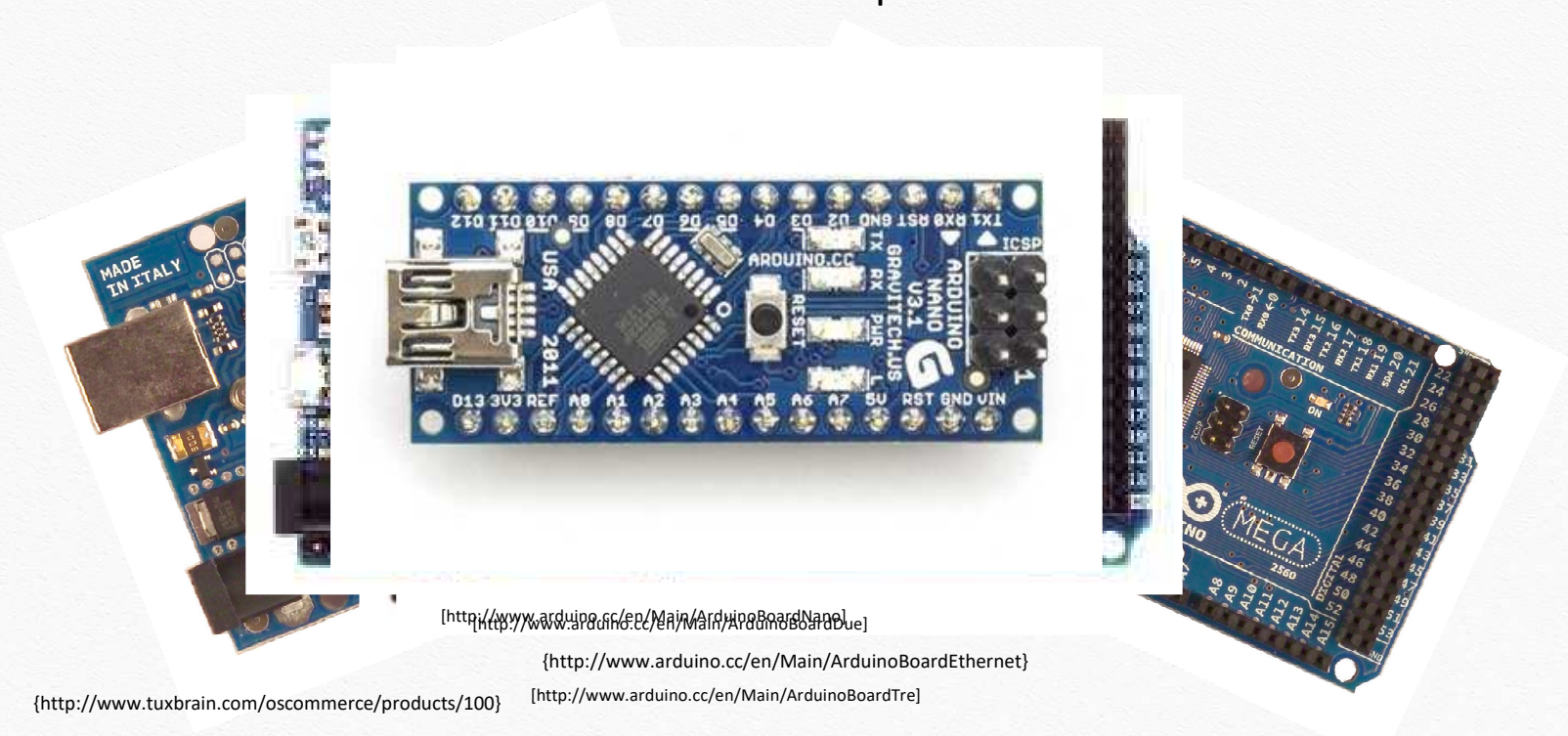
VENTAJAS

- Código abierto (Open source).
- Pueden ser ensamblados a mano.
- Bajo costo.
- Multiplataforma.
- Entorno de programación simple.

ARDUINO

HARDWARE

Existen varias versiones de Arduino. Entre las que se destacan:



{http://www.tuxbrain.com/oscommerce/products/100}

[http://www.arduino.cc/en/Main/ArduinoBoardNano]

[http://www.arduino.cc/en/Main/ArduinoBoardDue]

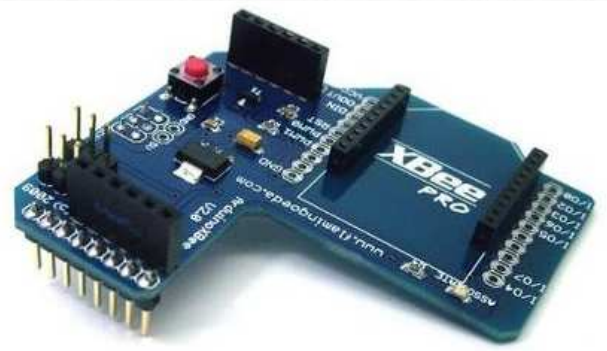
{http://www.arduino.cc/en/Main/ArduinoBoardEthernet}

[http://www.arduino.cc/en/Main/ArduinoBoardTre]

ARDUINO

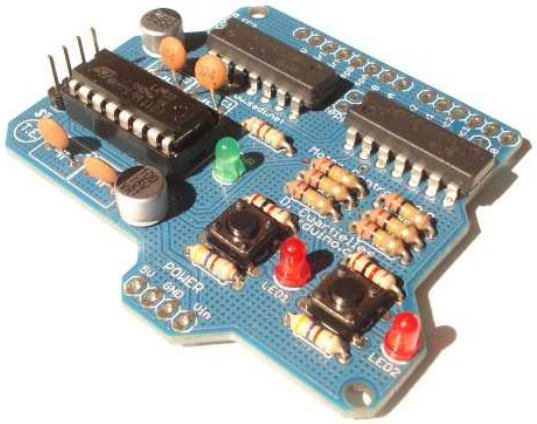
HARDWARE (módu

XBee

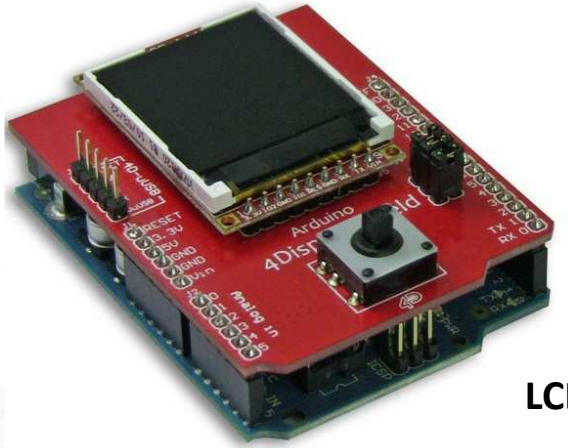


Reconocimiento de voz

Control de motores



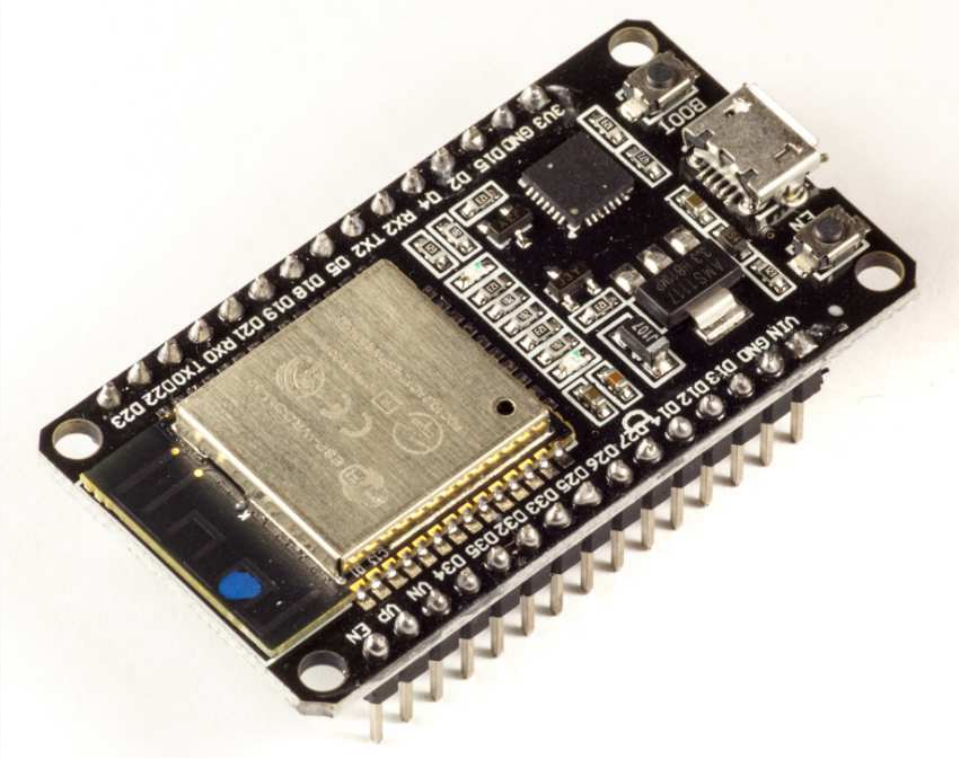
Bluetooth



LCD

¿QUÉ ES ESP32?

ESP32



https://www.espressif.com/

[https://commons.wikimedia.org/wiki/File:ESP32_Espressif_ESP-WROOM-32_Dev_Board_(2).jpg]
[https://upload.wikimedia.org/wikipedia/commons/1/1d/ESP32_Espressif_ESP-WROOM-32_Dev_Board_%282%29.jpg]

ESP32

https://www.espressif.com/

[https://commons.wikimedia.org/wiki/File:ESP32_Espressif_ESP-WROOM-32_Shielded.jpg]
[https://upload.wikimedia.org/wikipedia/commons/7/7b/ESP32_Espressif_ESP-WROOM-32_Shielded.jpg]

¿POR QUÉ ESP32?

¿POR QUÉ ESP32?

- Bajo costo
- Wi-Fi y Bluetooth
- Disponible en un módulo.
- No requiere programador (cuando viene en un módulo)
- Aplicaciones IoT
- Usado en asignatura: Adquisición de señales electrofisiológicas (Ingeniería Biomédica), Procesamiento de señales (Maestría Ingeniería Biomédica) e IoT (Ingeniería Mecatrónica)

CARACTERÍSTICAS

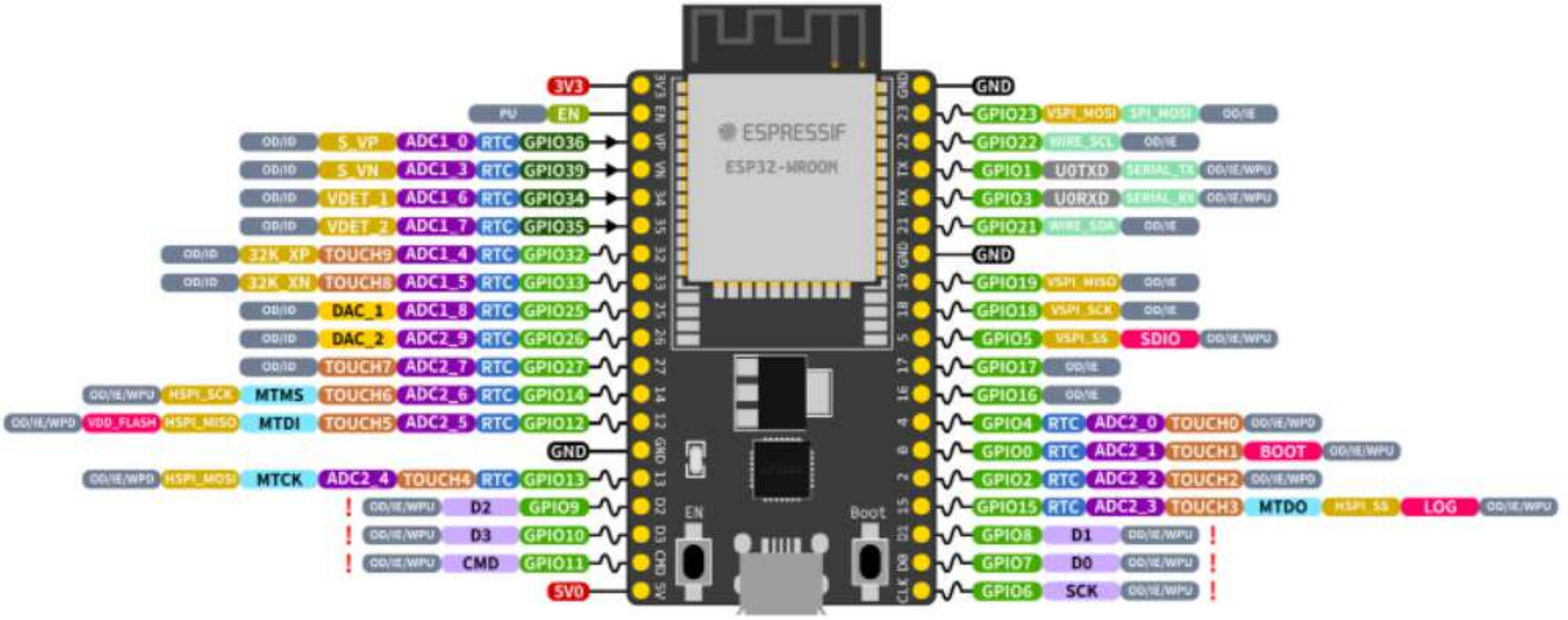
CARACTERÍSTICAS

- ✓ CPU 32-bits Xtensa LX6 dual-core 240 MHz
- ✓ ROM 448 KB
- ✓ RAM 520 KB
- ✓ Wi-Fi 802.11 n (2.4 GHz)
- ✓ Bluetooth v4.2
- ✓ GPIOs
- ✓ ADC
- ✓ DAC

CARACTERÍSTICAS

- ✓ SPI
- ✓ I2C
- ✓ I2S
- ✓ UART
- ✓ PWM
- ✓ CAN
- ✓ RTC (Real-Time clock)
- ✓ ULP (Ultra-Low-Power) 100uA

ESP32-DevKitC



ESP32 Specs

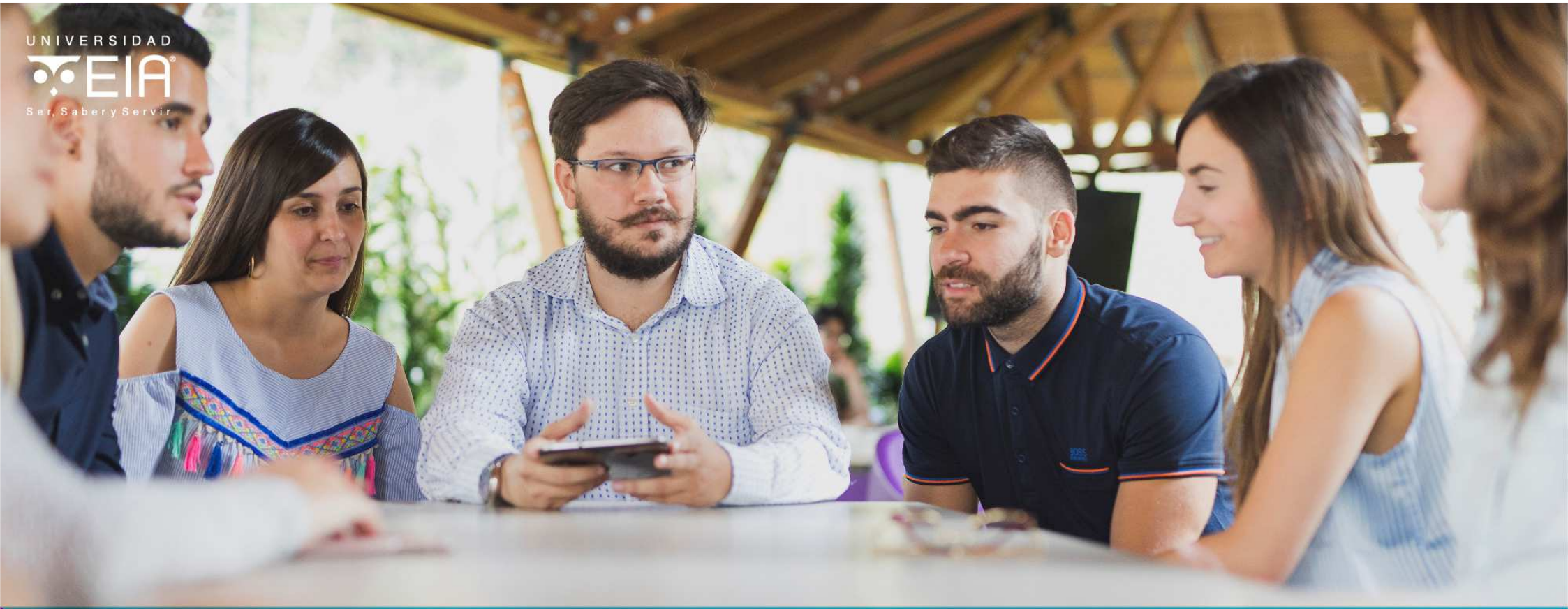
32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
BLuetooth 4.2 BR/EDR and BLE
520 KB SRAM (16 KB for cache)
448 KB ROM
34 GPIOs, 4x SPI, 3x UART, 2x I2C,
2x I2S, RMT, LED PWM, 1 host SD/eMMC/SDIO,
1 slave SDIO/SPI, TWAI®, 12-bit ADC, Ethernet

PWM Capable Pin
GPIOX GPIO Input Only
GPIOX GPIO Input and Output
DAC_X Digital-to-Analog Converter
DEBUG JTAG for Debugging
FLASH External Flash Memory (SPI)
ADCX_CH Analog-to-Digital Converter
TOUCHX Touch Sensor Input Channel
OTHER Other Related Functions
SERIAL Serial for Debug/Programming
ARDUINO Arduino Related Functions
STRAP Strapping Pin Functions

RTC RTC Power Domain (VDD3P3_RTC)
GND Ground
PWD Power Rails (3V3 and 5V)
! Pin Shared with the Flash Memory
Can't be used as regular GPIO

GPIO STATE
WPU: *Weak Pull-up (Internal)*
WPD: *Weak Pull-down (Internal)*
PU: *Pull-up (External)*
IE: *Input Enable (After Reset)*
ID: *Input Disabled (After Reset)*
OE: *Output Enable (After Reset)*
OD: *Output Disabled (After Reset)*

[https://docs.espressif.com/projects/esp-idf/en/latest/esp32/_images/esp32-devkitC-v4-pinout.png]

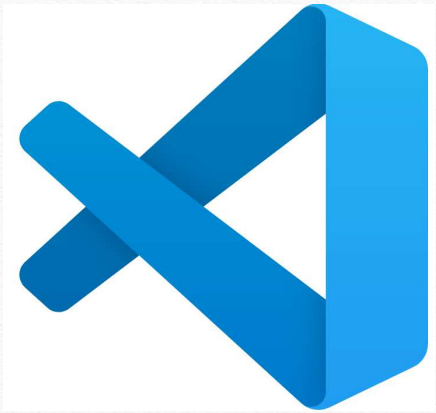


¿CÓMO SE PROGRAMA?

¿CÓMO SE PROGRAMA?

- ✓ Compilador C
- ✓ Arduino
- ✓ MicroPython

¿CÓMO SE PROGRAMA?



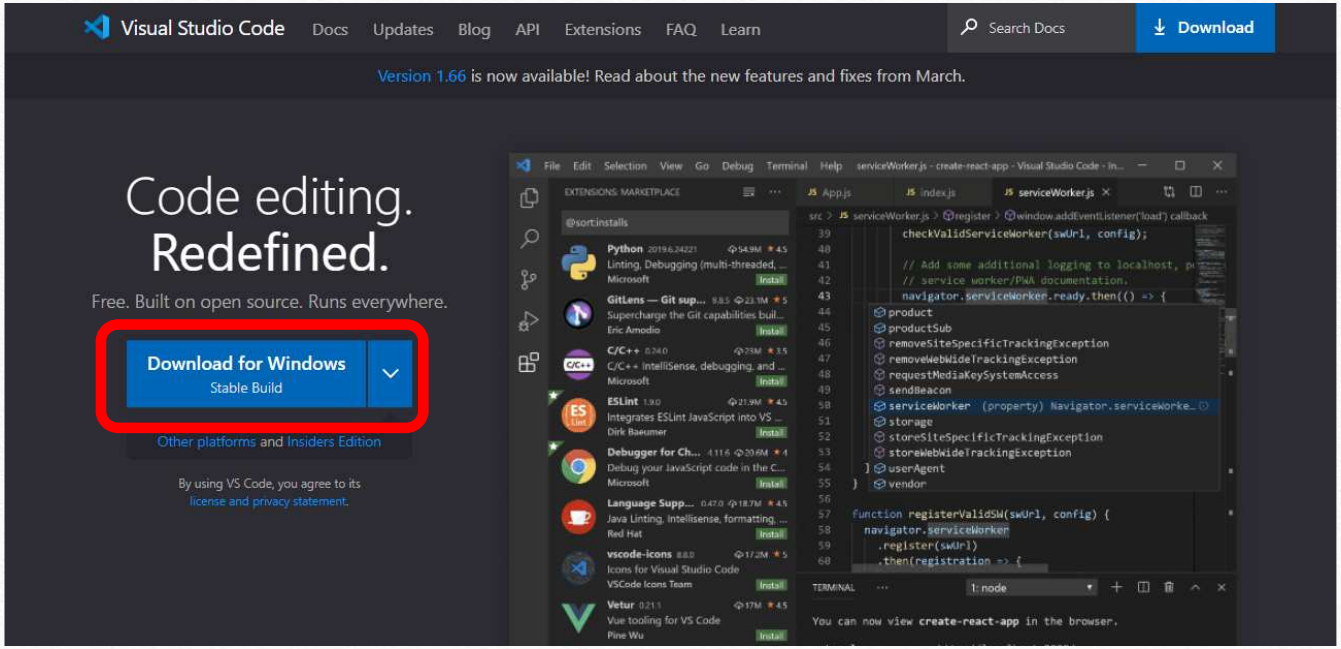
Visual Studio Code

[https://commons.wikimedia.org/wiki/File:Visual_Studio_Code_0.10.1_icon.png]

¿CÓMO SE PROGRAMA?

Visual Studio Code

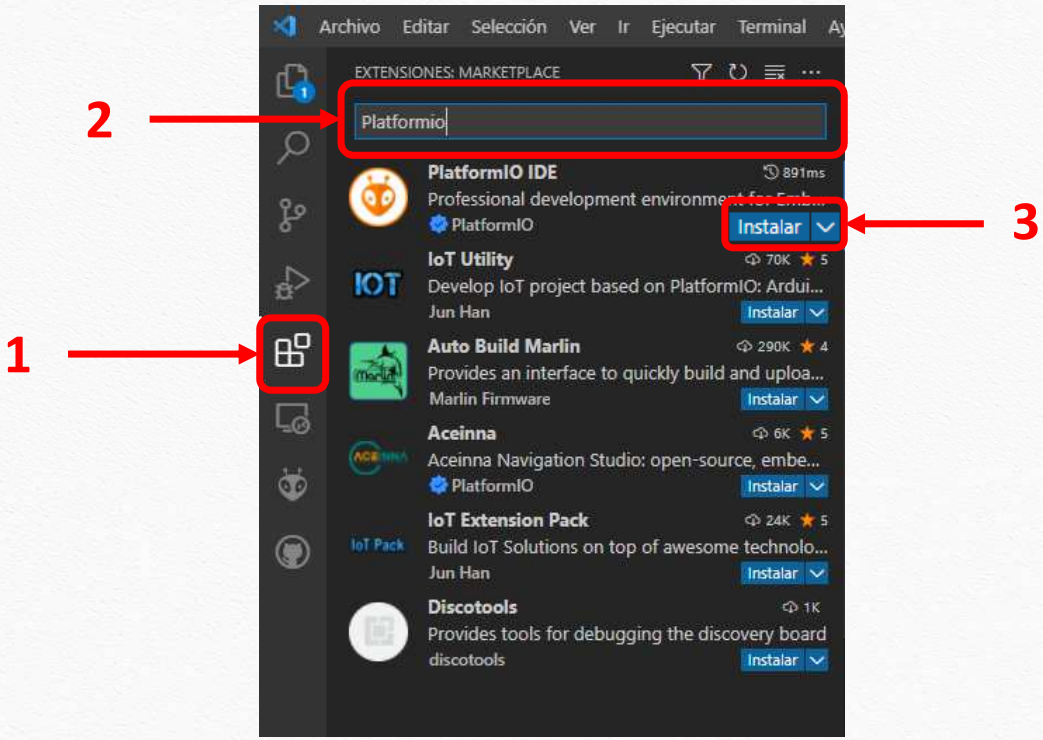
[https://commons.wikimedia.org/wiki/File:Visual_Studio_Code_0.10.1_icon.png]



https://code.visualstudio.com/

¿CÓMO SE PROGRAMA?

PLATFORMIO.ORG

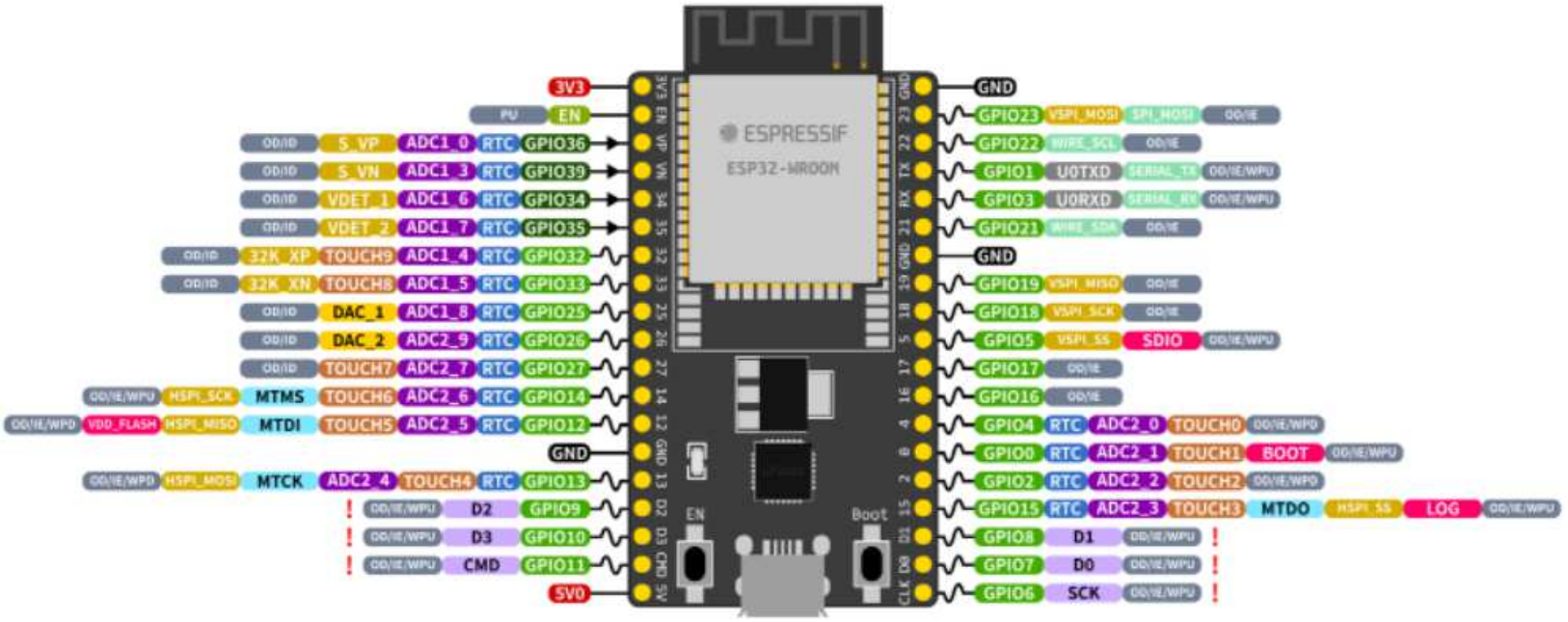


EJEMPLO 1

EJEMPLO 1

Diseñar un código que permita encender y apagar un led conectado al pin GPIO23 del ESP32

ESP32-DevKitC



ESP32 Specs

32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
BLuetooth 4.2 BR/EDR and BLE
520 KB SRAM (16 KB for cache)
448 KB ROM
34 GPIOs, 4x SPI, 3x UART, 2x I2C,
2x I2S, RMT, LED PWM, 1 host SD/eMMC/SDIO,
1 slave SDIO/SPI, TWAI®, 12-bit ADC, Ethernet

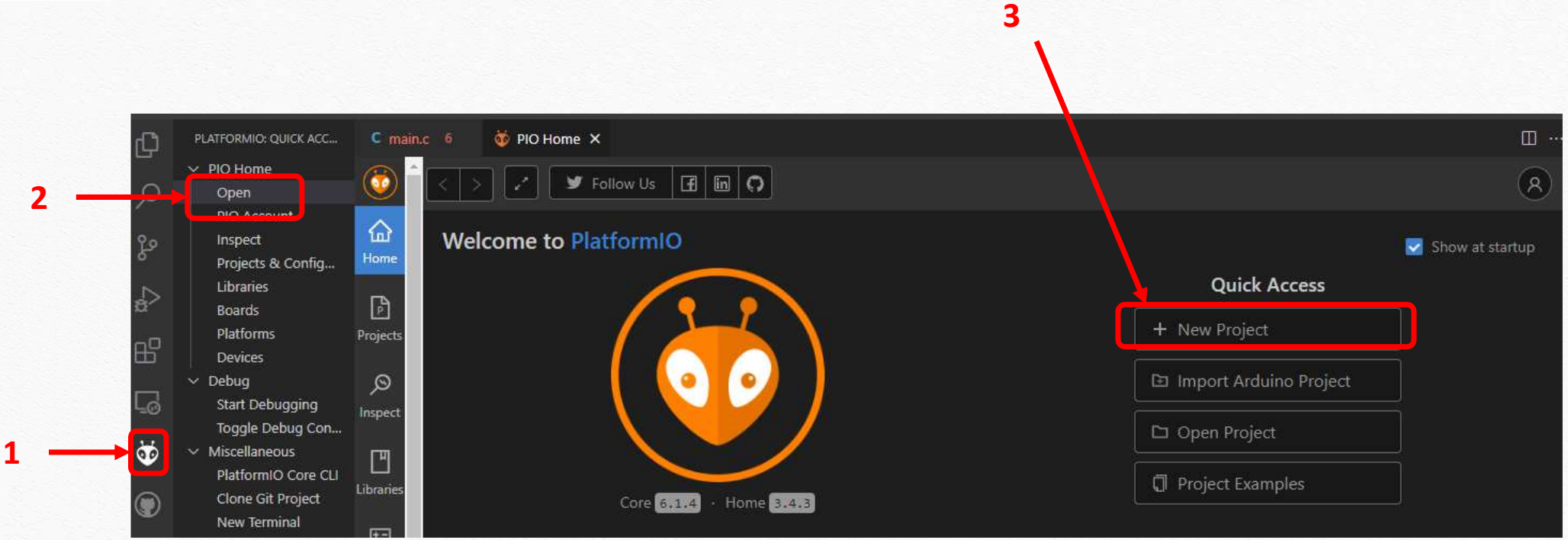
PWM Capable Pin
GPIOX GPIO Input Only
GPIOX GPIO Input and Output
DAC_X Digital-to-Analog Converter
DEBUG JTAG for Debugging
FLASH External Flash Memory (SPI)
ADCX_CH Analog-to-Digital Converter
TOUCHX Touch Sensor Input Channel
OTHER Other Related Functions
SERIAL Serial for Debug/Programming
ARDUINO Arduino Related Functions
STRAP Strapping Pin Functions

RTC RTC Power Domain (VDD3P3_RTC)
GND Ground
PWD Power Rails (3V3 and 5V)
! Pin Shared with the Flash Memory
Can't be used as regular GPIO

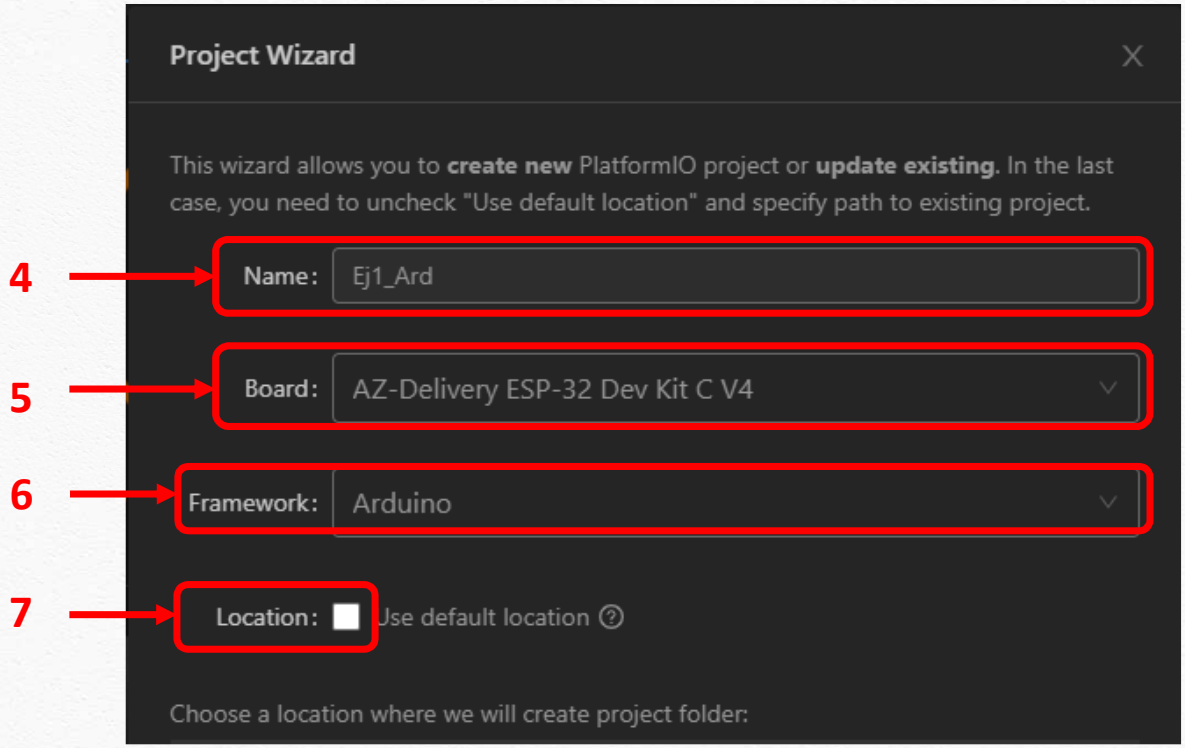
GPIO STATE
WPU: *Weak Pull-up (Internal)*
WPD: *Weak Pull-down (Internal)*
PU: *Pull-up (External)*
IE: *Input Enable (After Reset)*
ID: *Input Disabled (After Reset)*
OE: *Output Enable (After Reset)*
OD: *Output Disabled (After Reset)*

[https://docs.espressif.com/projects/esp-idf/en/latest/esp32/_images/esp32-devkitC-v4-pinout.png]

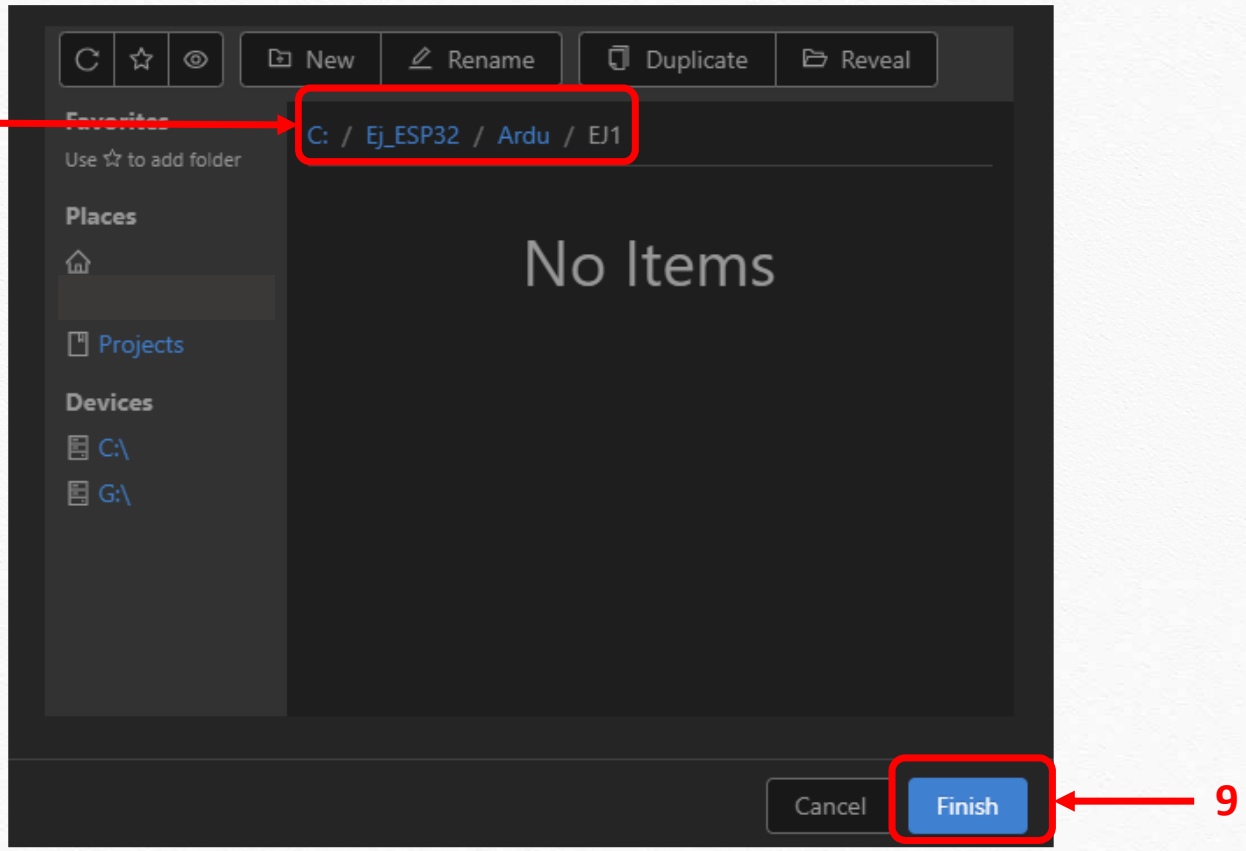
EJEMPLO 1


3
2
1
PLATFORMIO: QUICK ACC...
PIO Home
Open
Inspect
Projects & Config...
Libraries
Boards
Platforms
Devices
Debug
Start Debugging
Toggle Debug Con...
Miscellaneous
PlatformIO Core CLI
Clone Git Project
New Terminal
main.c 6
PIO Home
Follow Us
Home
Projects
Inspect
Libraries
Welcome to PlatformIO
Show at startup
Quick Access
New Project
Import Arduino Project
Open Project
Project Examples
Core 6.1.4 · Home 3.4.3

EJEMPLO 1


Project Wizard
This wizard allows you to create new PlatformIO project or update existing. In the last case, you need to uncheck "Use default location" and specify path to existing project.
4
Name: Ej1_Ard
5
Board: AZ-Delivery ESP-32 Dev Kit C V4
6
Framework: Arduino
7
Location: Use default location
Choose a location where we will create project folder:

EJEMPLO 1


8
New
Rename
Duplicate
Reveal
C: / Ej_ESP32 / Ardu / EJ1
Use ☆ to add folder
Places
Projects
Devices
C:\
G:\
No Items
Cancel
Finish
9

EJEMPLO 1

Diseñar un código que permita encender y apagar un led conectado al pin GPIO23 del ESP32

```
1   #include <Arduino.h>
```

```
3   void setup() {
4     // put your setup code here, to run once:
5     pinMode(23,OUTPUT);
6   }
```

```
8   void loop() {
9     // put your main code here, to run repeatedly:
10    digitalWrite(23,HIGH);
11    delay(200);
12    digitalWrite(23,LOW);
13    delay(100);
14  }
```

EJERCICIO 1

EJERCICIO 1

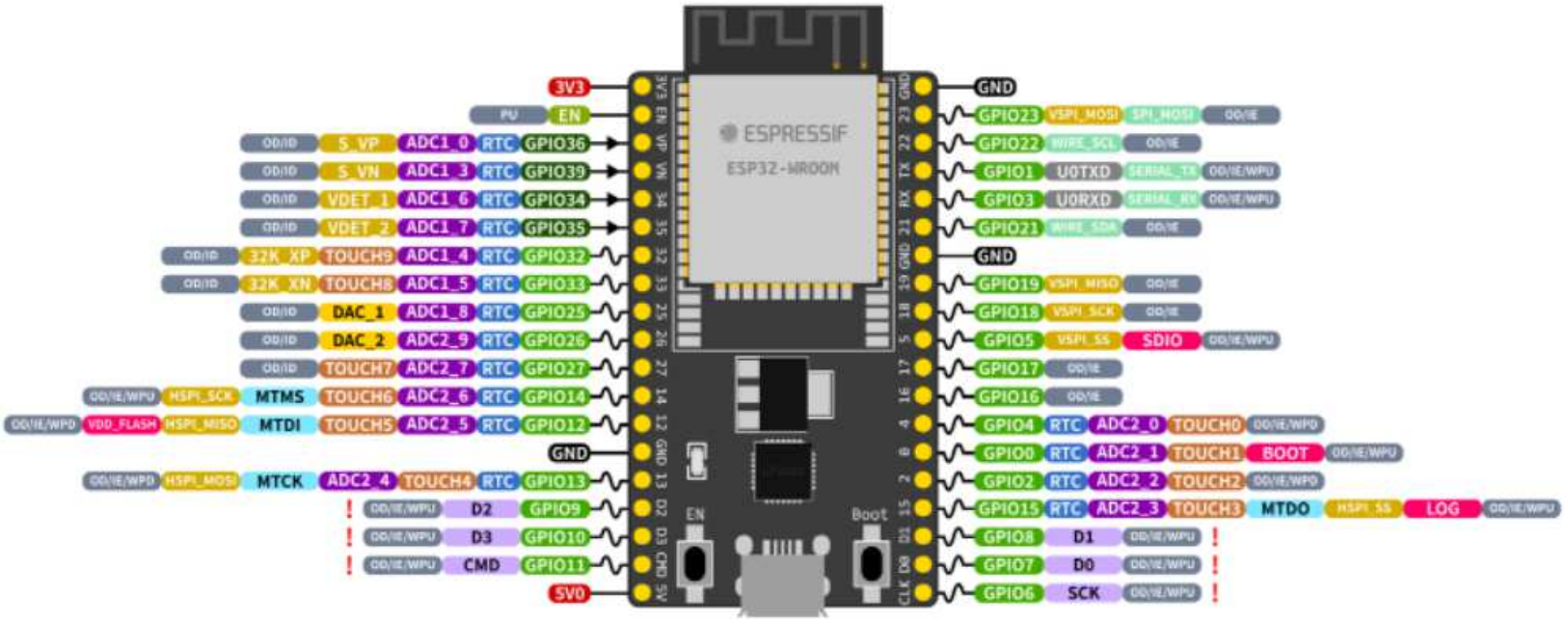
Diseñar un secuenciador de luces usando 4 leds y un ESP32

EJEMPLO 2

EJEMPLO 2

Por medio de un pulsador conectado al pin GPIO5, controlar el encendido y apagado de un led conectado al pin GPIO23 del microcontrolador ESP32

ESP32-DevKitC



ESP32 Specs

32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
BLuetooth 4.2 BR/EDR and BLE
520 KB SRAM (16 KB for cache)
448 KB ROM
34 GPIOs, 4x SPI, 3x UART, 2x I2C,
2x I2S, RMT, LED PWM, 1 host SD/eMMC/SDIO,
1 slave SDIO/SPI, TWAI®, 12-bit ADC, Ethernet

PWM Capable Pin
GPIOX GPIO Input Only
GPIOX GPIO Input and Output
DAC_X Digital-to-Analog Converter
DEBUG JTAG for Debugging
FLASH External Flash Memory (SPI)
ADCX_CH Analog-to-Digital Converter
TOUCHX Touch Sensor Input Channel
OTHER Other Related Functions
SERIAL Serial for Debug/Programming
ARDUINO Arduino Related Functions
STRAP Strapping Pin Functions

RTC RTC Power Domain (VDD3P3_RTC)
GND Ground
PWD Power Rails (3V3 and 5V)
! Pin Shared with the Flash Memory
Can't be used as regular GPIO

GPIO STATE
WPU: *Weak Pull-up (Internal)*
WPD: *Weak Pull-down (Internal)*
PU: *Pull-up (External)*
IE: *Input Enable (After Reset)*
ID: *Input Disabled (After Reset)*
OE: *Output Enable (After Reset)*
OD: *Output Disabled (After Reset)*

[https://docs.espressif.com/projects/esp-idf/en/latest/esp32/_images/esp32-devkitC-v4-pinout.png]

EJEMPLO 2

```
1   #include <Arduino.h>

3    boolean estado = false;

5   void setup() {
6     // put your setup code here, to run once:
7     pinMode(23,OUTPUT);
8     pinMode(5,INPUT);
9   }
```

```
11  void loop() {
12    // put your main code here, to run repeatedly:
13
14    if (digitalRead (5) == LOW)
15    {
16      delay(200);
17      estado = !estado;
18      digitalWrite(23,estado);
19
20    }
21
22  }
```

EJEMPLO 2

```
1   #include <Arduino.h>

3    boolean estado = false;

5   void setup() {
6     // put your setup code here, to run once:
7     pinMode(23,OUTPUT);
8     pinMode(5,INPUT);
9   }
```

Resistencias Pull-UP
pinMode(5, INPUT_PULLUP);

```
11  void loop() {
12    // put your main code here, to run repeatedly:
13
14    if (digitalRead (5) == LOW)
15    {
16      delay(200);
17      estado = !estado;
18      digitalWrite(23,estado);
19
20    }
21
22  }
```

EJERCICIO 2

EJERCICIO 2

Por medio de un interruptor conectado al pin GPIO5, seleccionar la secuencia de luces que se muestra en los cuatro leds conectados al microcontrolador ESP32

UART

MÓDULO DE COMUNICACIÓN UART

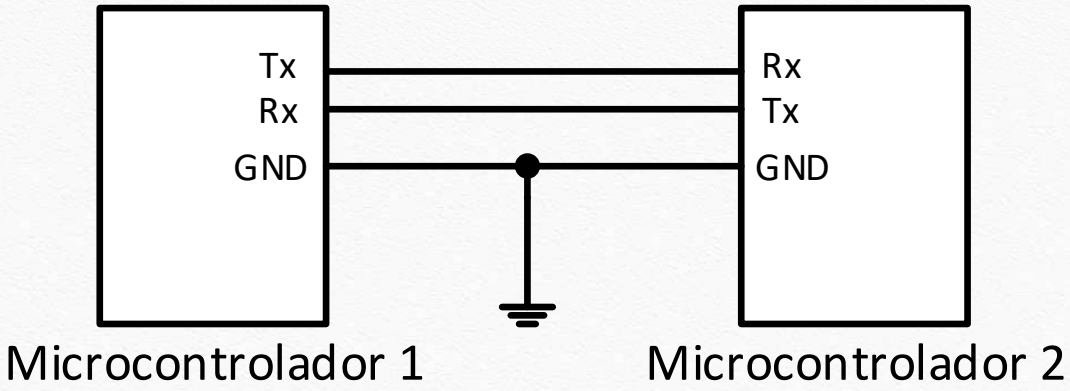
USART:

Comunicación serial, permite la comunicación entre dispositivos digitales o entre un microcontrolador y un PC.

MÓDULO DE COMUNICACIÓN UART

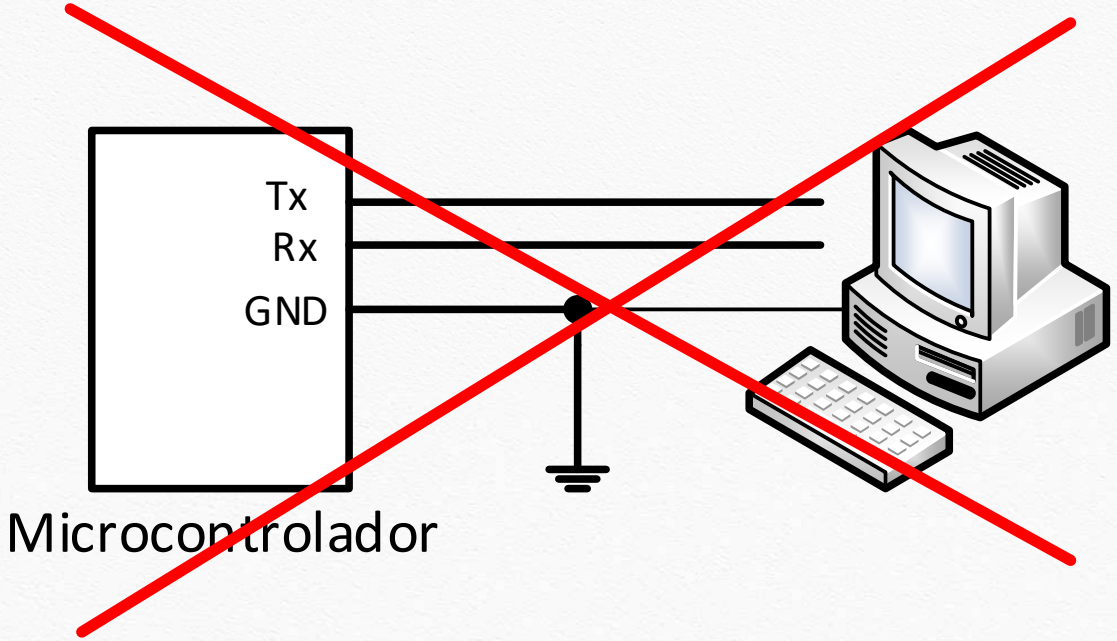
ASINCRONA:

- No requiere señal de reloj
- Comunicación full dúplex
- Solo necesita 2 cables: Rx y Tx



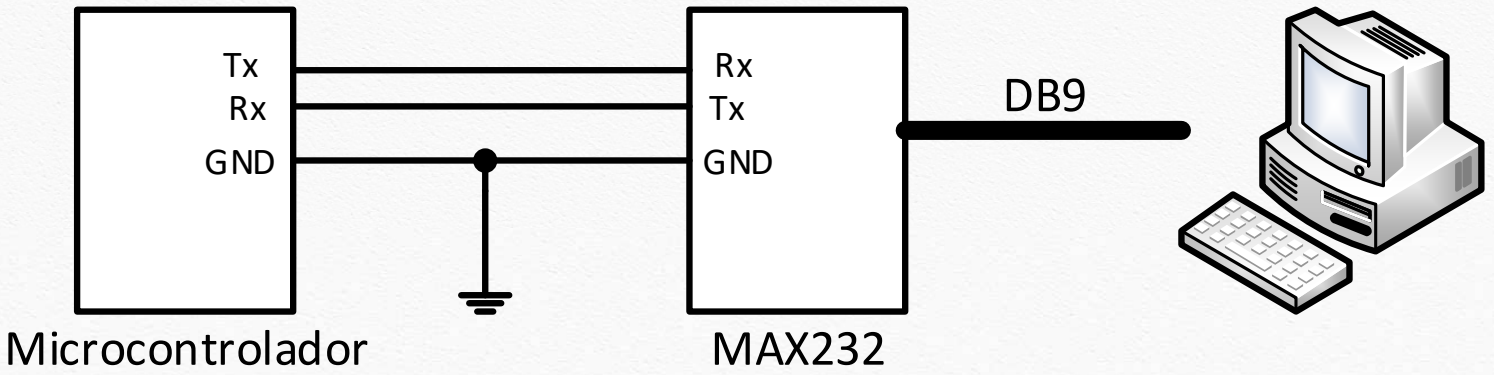
MÓDULO DE COMUNICACIÓN UART

COMUNICACIÓN CON UN PC



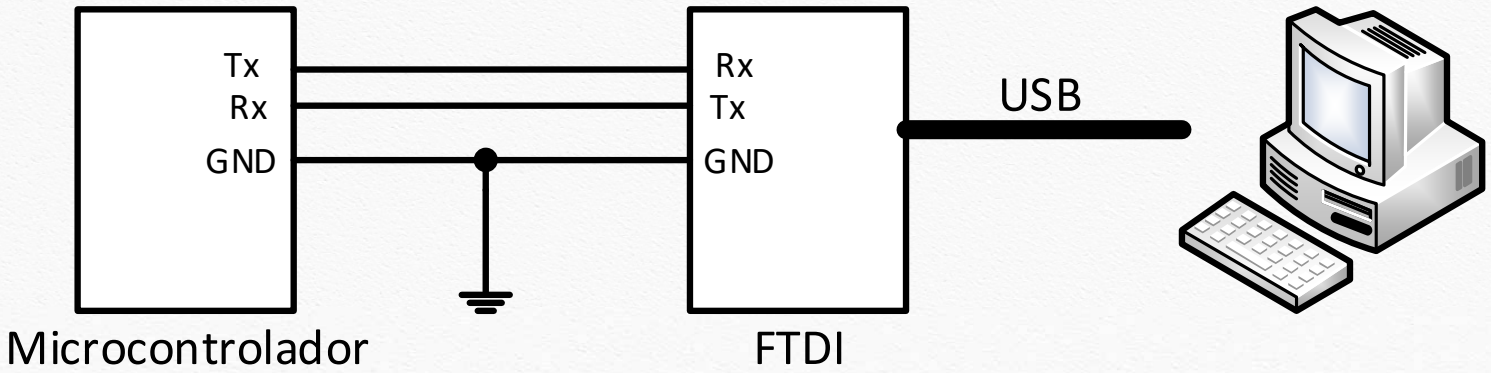
MÓDULO DE COMUNICACIÓN UART

COMUNICACIÓN CON UN PC



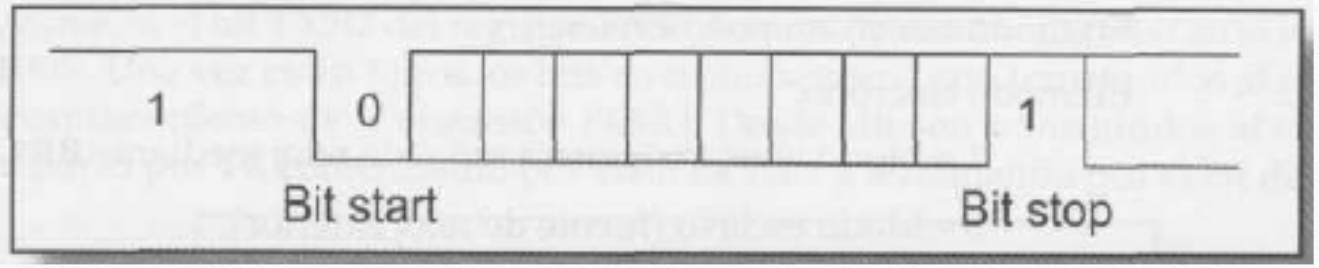
MÓDULO DE COMUNICACIÓN UART

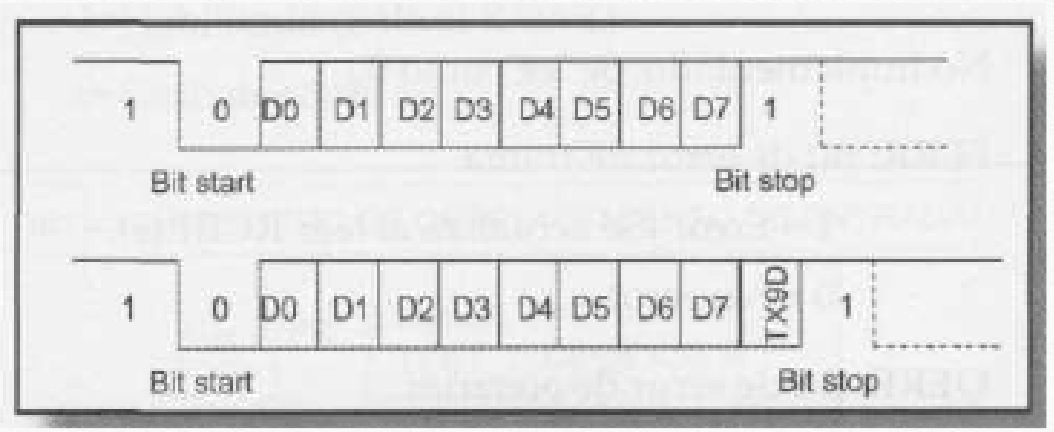
COMUNICACIÓN CON UN PC



MÓDULO DE COMUNICACIÓN UART

TRANSMISIÓN ASINCRONA





EJEMPLO 3

EJEMPLO 3

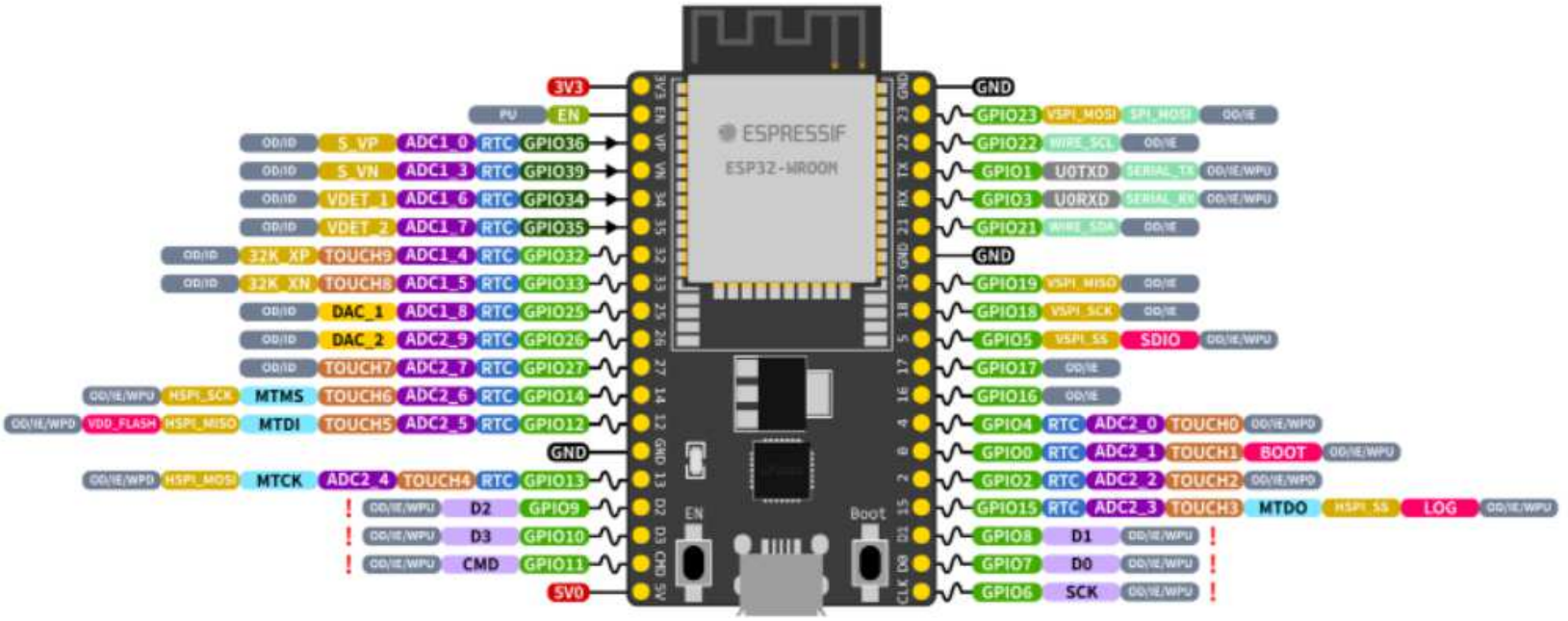
Generar un contador en el ESP32 y transmitirlo por el puerto UART al PC. Mostrar el valor del contador en la consola.

UNIVERSIDAD EIA
Ser, Saber y Servir

KIRON

ESP32-DevKitC

ESPRESSIF



ESP32 Specs

32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
BLuetooth 4.2 BR/EDR and BLE
520 KB SRAM (16 KB for cache)
448 KB ROM
34 GPIOs, 4x SPI, 3x UART, 2x I2C,
2x I2S, RMT, LED PWM, 1 host SD/eMMC/SDIO,
1 slave SDIO/SPI, TWAI®, 12-bit ADC, Ethernet

PWM Capable Pin
GPIOX GPIO Input Only
GPIOX GPIO Input and Output
DAC_X Digital-to-Analog Converter
DEBUG JTAG for Debugging
FLASH External Flash Memory (SPI)
ADCX_CH Analog-to-Digital Converter
TOUCHX Touch Sensor Input Channel
OTHER Other Related Functions
SERIAL Serial for Debug/Programming
ARDUINO Arduino Related Functions
STRAP Strapping Pin Functions

RTC RTC Power Domain (VDD3P3_RTC)
GND Ground
PWD Power Rails (3V3 and 5V)
! Pin Shared with the Flash Memory
Can't be used as regular GPIO

GPIO STATE
WPU: *Weak Pull-up (Internal)*
WPD: *Weak Pull-down (Internal)*
PU: *Pull-up (External)*
IE: *Input Enable (After Reset)*
ID: *Input Disabled (After Reset)*
OE: *Output Enable (After Reset)*
OD: *Output Disabled (After Reset)*

[https://docs.espressif.com/projects/esp-idf/en/latest/esp32/_images/esp32-devkitC-v4-pinout.png]

EJEMPLO 3

Generar un contador en el ESP32 y transmitirlo por el puerto UART al PC. Mostrar el valor del contador en la consola.

```
1   #include <Arduino.h>
```

```
3    int cont = 0;
```

```
5   void setup() {
6     Serial.begin(9600);
7   }
```

```
 9   void loop() {
10     Serial.println(cont);
11     cont = cont + 1;
12     delay(1000);
13   }
```

EJEMPLO 4

EJEMPLO 4

Encender y apagar un led conectado al GPIO23 del ESP32.

0: Apaga el led.

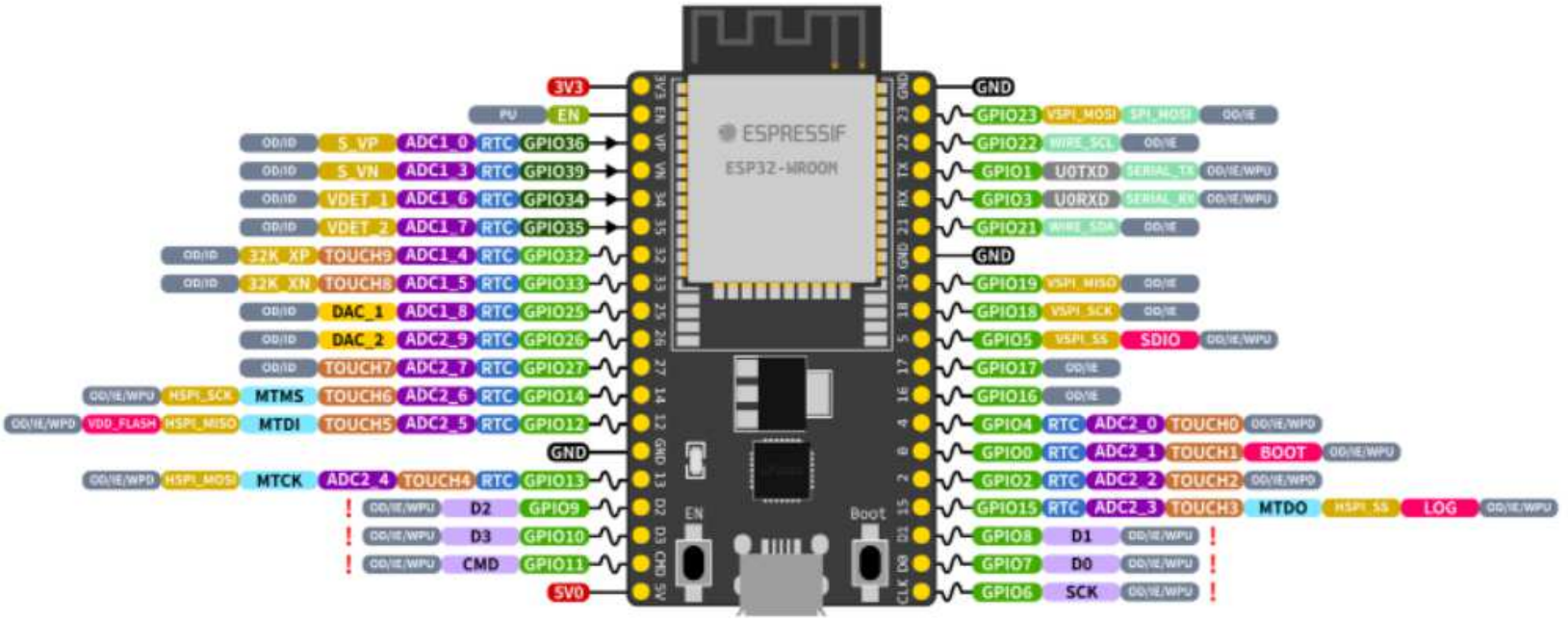
1: Enciende el led.

UNIVERSIDAD EIA
Ser, Saber y Servir

KIRON

ESP32-DevKitC

ESPRESSIF



ESP32 Specs

32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
BLuetooth 4.2 BR/EDR and BLE
520 KB SRAM (16 KB for cache)
448 KB ROM
34 GPIOs, 4x SPI, 3x UART, 2x I2C,
2x I2S, RMT, LED PWM, 1 host SD/eMMC/SDIO,
1 slave SDIO/SPI, TWAI®, 12-bit ADC, Ethernet

[https://docs.espressif.com/projects/esp-idf/en/latest/esp32/_images/esp32-devkitC-v4-pinout.png]

EJEMPLO 4

```
/*****************************************
  Título: Ejemplo 4
  Descripción:
  Recepción de datos a través de puerto
  UART desde consola del PC.
  Autor:
  Fecha:
  Observaciones:
*****************************************/
#include <Arduino.h>
```

```
/*----------------------------------------
MASCARAS DE PINES
----------------------------------------*/
int LED = 23;
```

```
/*----------------------------------------
Definición de Variables
----------------------------------------*/
int X = 0;
```

```
/*----------------------------------------
CONFIGURACIÓN
----------------------------------------*/
void setup() {
  Serial.begin(9600);
  pinMode(LED,OUTPUT);
}
```

```
/*----------------------------------------
BUCLE INFINITO
----------------------------------------*/
void loop() {
  if (Serial.available() > 0){
    X = Serial.read();
    if (X == 49)
    {
      digitalWrite(LED,HIGH);
    }
    if (X == 48)
    {
      digitalWrite(LED,LOW);
    }

  }
}
```

EJEMPLO 4

```
/*****************************************
  Título: Ejemplo 4
  Descripción:
  Recepción de datos a través de puerto
  UART desde consola del PC.
  Autor:
  Fecha:
  Observaciones:
*****************************************/
#include <Arduino.h>
```

```
/*----------------------------------------
MASCARAS DE PINES
----------------------------------------*/
int LED = 23;
```

```
/*----------------------------------------
Definición de Variables
----------------------------------------*/
int X = 0;
```

```
/*----------------------------------------
CONFIGURACIÓN
----------------------------------------*/
void setup() {
  Serial.begin(9600);
  pinMode(LED,OUTPUT);
}
```

```
/*----------------------------------------
BUCLE INFINITO
----------------------------------------*/
void loop() {
  if (Serial.available() > 0){
    X = Serial.read();
    if (X == '1')
    {
      digitalWrite(LED,HIGH);
    }
    if (X == '0')
    {
      digitalWrite(LED,LOW);
    }

  }
}
```

EJEMPLO 4

```
/*****************************************
  Título: Ejemplo 4
  Descripción:
  Recepción de datos a través de puerto
  UART desde consola del PC.
  Autor:
  Fecha:
  Observaciones:
*****************************************/
#include <Arduino.h>
```

```
/*----------------------------------------
MASCARAS DE PINES
----------------------------------------*/
int LED = 23;
```

```
/*----------------------------------------
Definición de Variables
----------------------------------------*/
int X = 0;
```

```
/*----------------------------------------
CONFIGURACIÓN
----------------------------------------*/
void setup() {
  Serial.begin(9600);
  pinMode(LED,OUTPUT);
}
```

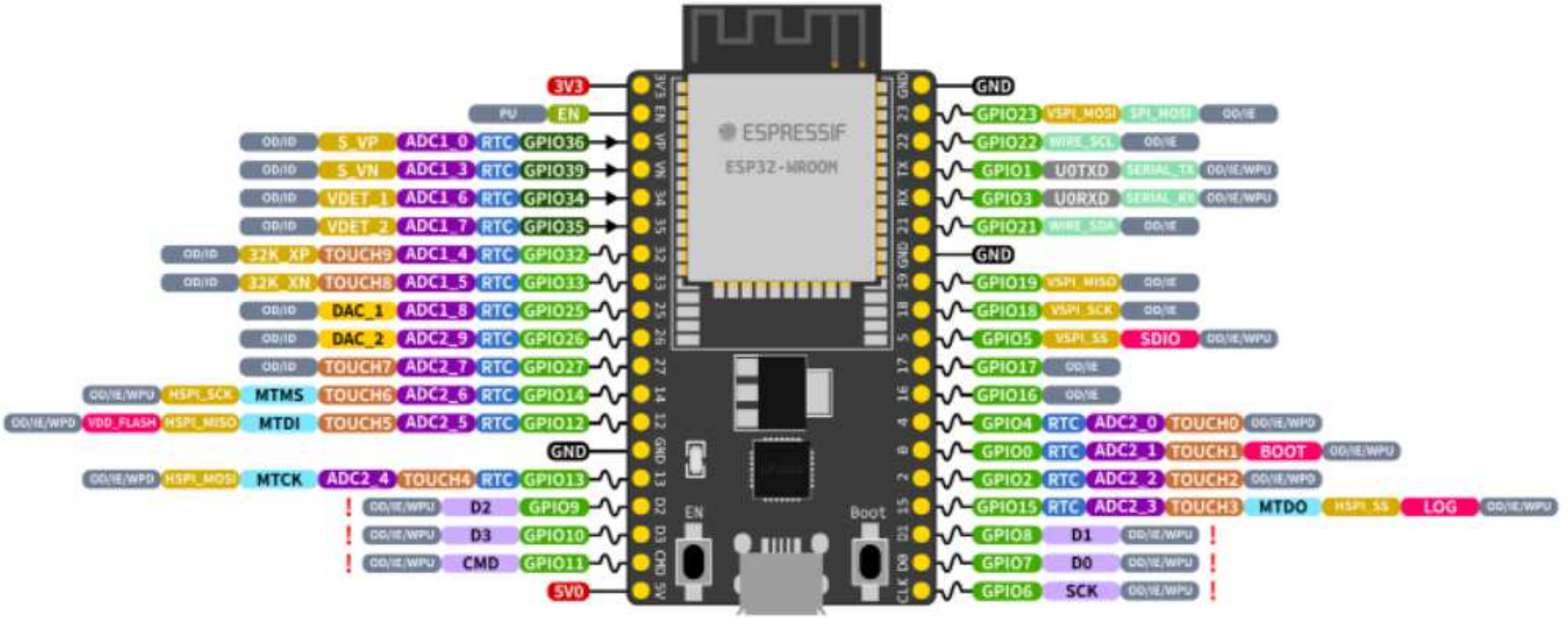
```
/*----------------------------------------
BUCLE INFINITO
----------------------------------------*/
void loop() {
  if (Serial.available() > 0){
    X = Serial.read();
    if (X == 'a')
    {
      digitalWrite(LED,HIGH);
    }
    if (X == 'b')
    {
      digitalWrite(LED,LOW);
    }

  }
}
```

EJEMPLO 5

EJEMPLO 5

Diseñar un voltimetro que muestre el valor medido en la consola del PC

ESP32-DevKitC



ESP32 Specs

32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
BLuetooth 4.2 BR/EDR and BLE
520 KB SRAM (16 KB for cache)
448 KB ROM
34 GPIOs, 4x SPI, 3x UART, 2x I2C,
2x I2S, RMT, LED PWM, 1 host SD/eMMC/SDIO,
1 slave SDIO/SPI, TWAI®, 12-bit ADC, Ethernet

[https://docs.espressif.com/projects/esp-idf/en/latest/esp32/_images/esp32-devkitC-v4-pinout.png]

EJEMPLO 5

```
#include <Arduino.h>

const int Sensor = 34;

int ADC = 0;
float Vol = 0.0;
int mVol = 0;

void setup() {
  Serial.begin(9600);
  pinMode(Sensor,INPUT);
}

void loop() {

  ADC = analogRead(Sensor);
  Vol = (ADC * 3.3)/4095.0;
  mVol = Vol * 1000;

  Serial.print("ADC = ");
  Serial.println(ADC);
  Serial.print("Voltaje = ");
  Serial.print(mVol);
  Serial.println(" mV");

  delay(1000);
}
```

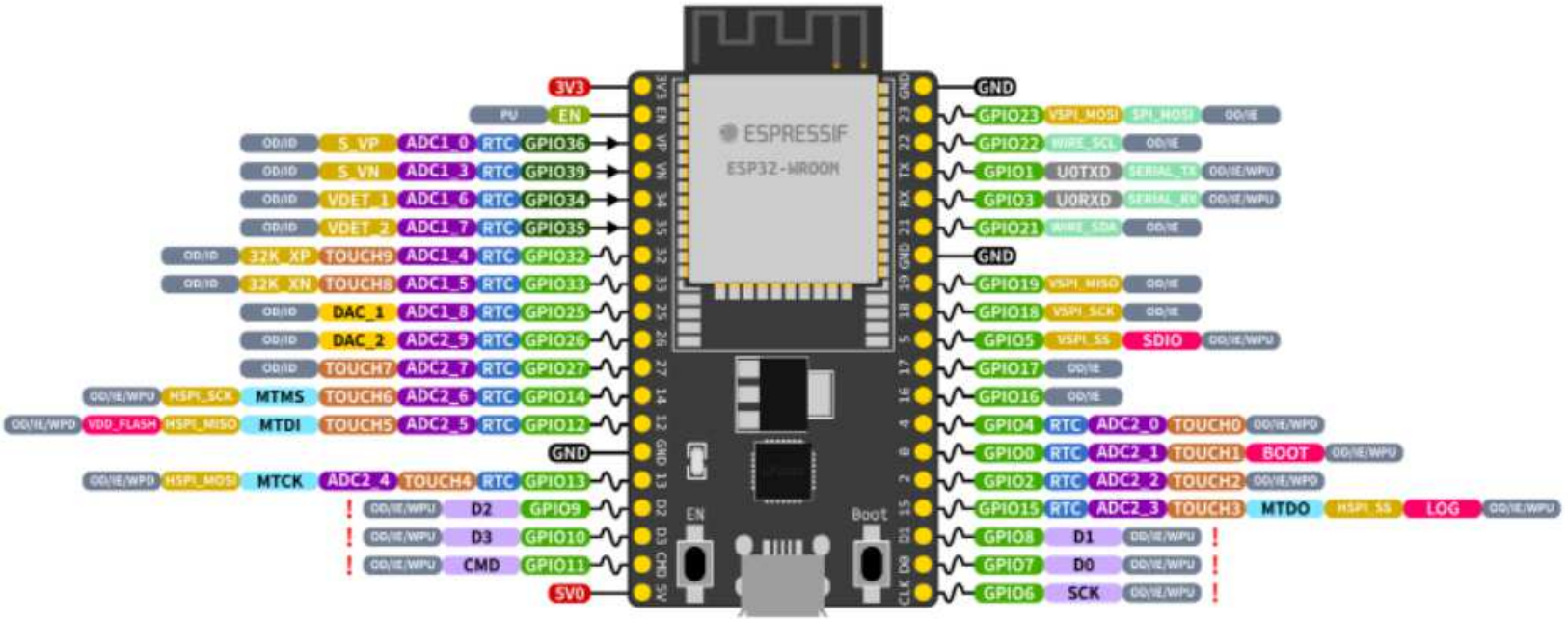
EJEMPLO: PWM

EJEMPLO

Diseñar un circuito que controle la intensidad de iluminación de un led conectado al GPIO23 del ESP32.

El led iniciará apagado y la intensidad comenzará a aumentar hasta que llegue a su intensidad máxima.

ESP32-DevKitC



ESP32 Specs

32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
BLuetooth 4.2 BR/EDR and BLE
520 KB SRAM (16 KB for cache)
448 KB ROM
34 GPIOs, 4x SPI, 3x UART, 2x I2C,
2x I2S, RMT, LED PWM, 1 host SD/eMMC/SDIO,
1 slave SDIO/SPI, TWAI®, 12-bit ADC, Ethernet

PWM Capable Pin
GPIOX GPIO Input Only
GPIOX GPIO Input and Output
DAC_X Digital-to-Analog Converter
DEBUG JTAG for Debugging
FLASH External Flash Memory (SPI)
ADCX_CH Analog-to-Digital Converter
TOUCHX Touch Sensor Input Channel
OTHER Other Related Functions
SERIAL Serial for Debug/Programming
ARDUINO Arduino Related Functions
STRAP Strapping Pin Functions

RTC RTC Power Domain (VDD3P3_RTC)
GND Ground
PWD Power Rails (3V3 and 5V)
! Pin Shared with the Flash Memory *Can't be used as regular GPIO*

GPIO STATE
WPU: *Weak Pull-up (Internal)*
WPD: *Weak Pull-down (Internal)*
PU: *Pull-up (External)*
IE: *Input Enable (After Reset)*
ID: *Input Disabled (After Reset)*
OE: *Output Enable (After Reset)*
OD: *Output Disabled (After Reset)*

[https://docs.espressif.com/projects/esp-idf/en/latest/esp32/_images/esp32-devkitC-v4-pinout.png]

EJEMPLO

```
#include <Arduino.h>

int ciclo_util = 0;

void setup() {

  ledcSetup(0,5000,8); //canal, frecuencia, resolución

ledcAttachPin(23,0);

 Serial.begin(9600);

  delay(1000);

}
```

```
void loop() {


  ledcWrite(0, ciclo_util);

  Serial.println(ciclo_util);

  ciclo_util = ciclo_util + 1;
  if (ciclo_util > 255)
  {
    ciclo_util = 0;
  }

  delay(1000);

}
```

EJEMPLO: BLUETOOTH

EJEMPLO

Diseñar un circuito donde se conecte un teléfono celular con el ESP32 y este a su vez se conecte con un PC a través del puerto USB.

El dato que trasmite el PC al ESP32 será transmitido al celular y el dato transmitido por el celular será enviado al PC.

EJEMPLO

```
#include <Arduino.h>
```

```
#include "BluetoothSerial.h"
```

```
BluetoothSerial SerialBT;
```

```
int Dato_BT = 0;
int Dato_UART = 0;
```

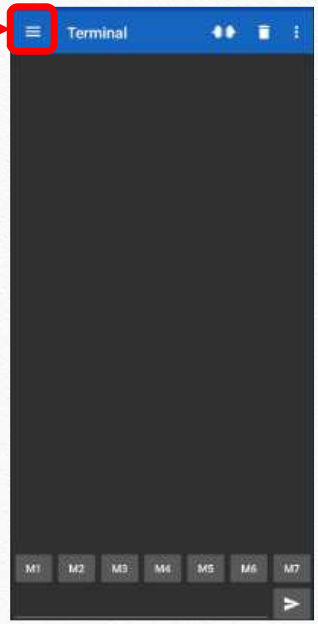
```
void setup() {
  Serial.begin(9600);
  delay(1000);
  SerialBT.begin("ESP32_YM");
  Serial.println("Emparejar el BT");
  delay(1000);
  }
```

```
void loop() {

  if (Serial.available() > 0)
  {
    Dato_UART = Serial.read();
    SerialBT.print("Dato desde PC = ");
    SerialBT.println(Dato_UART);
  }
```

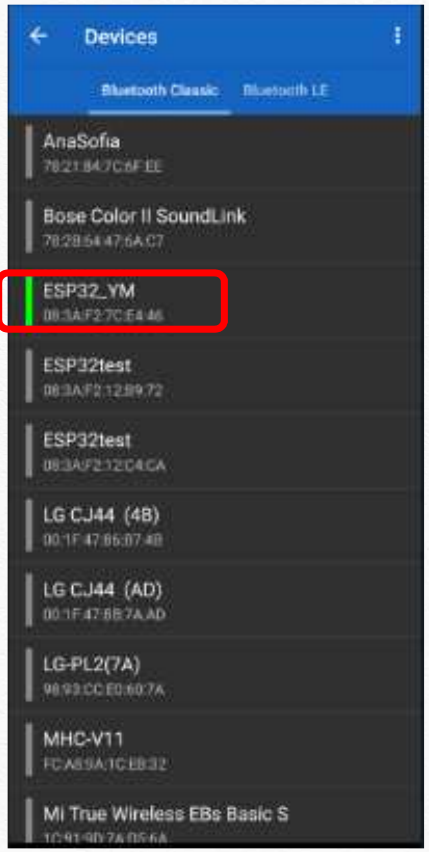
```
  if (SerialBT.available() > 0)
  {
    Dato_BT = SerialBT.read();
    Serial.print("Dato desde BT = ");
    Serial.println(Dato_BT);
  }


}
```

APP DE PRUEBA

1



2



3