

Az-Delivery

¡Bienvenido!

Muchas gracias por comprar nuestro *AZ-Delivery ESP-32 Dev Kit C V2*. En las siguientes páginas se presentará como utilizar y configurar este práctico dispositivo.

¡Diviértase!

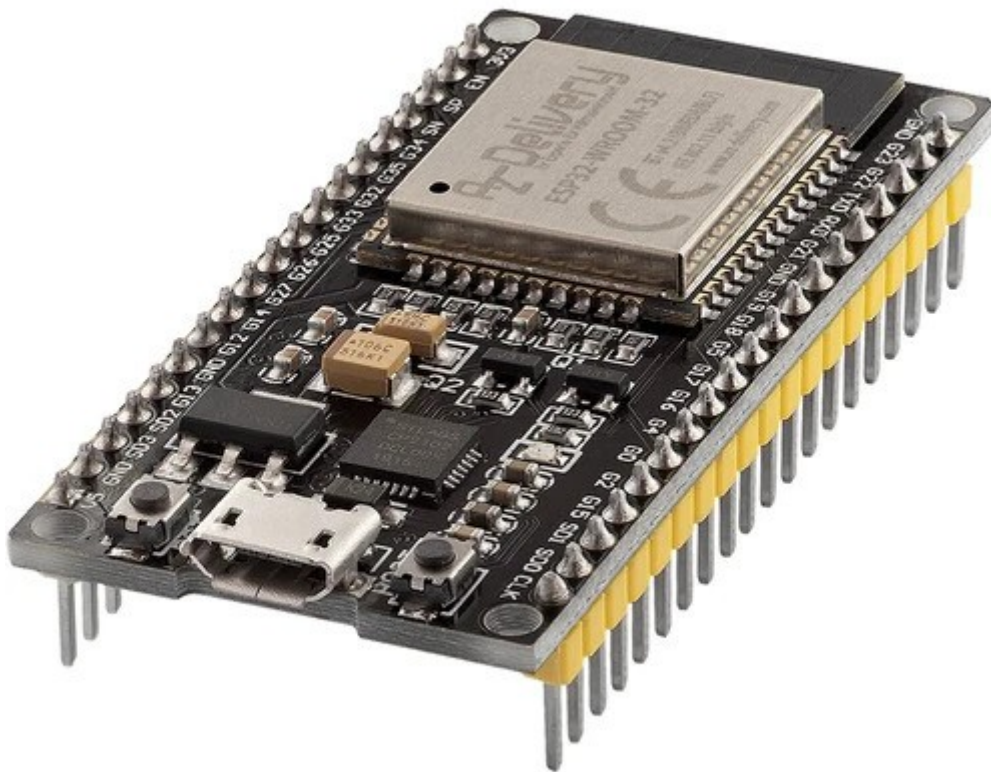




Tabla de Contenidos

Introducción.....	3
Especificaciones.....	4
ESP32 Dev Kit C V2.....	5
Disposición de los pines.....	6
Descripción de los pines.....	7
Pines del Sensor Táctil Capacitivo.....	8
Pines del Convertidor Analógico a Digital	9
Pines del Convertidor Digital a Analógico	9
Pines GPIO del Reloj en Tiempo Real.....	10
Pines del PWM (Pulse Width Modulation).....	11
Pines de la interfaz I2C.....	11
Pines de la interfaz SPI.....	12
Pines de conexión.....	12
Pines HIGH en el Arranque.....	13
Habilitación (EN).....	13
USB a Comunicación en Serie.....	14
Comunicación WiFi.....	15
Comunicación Bluetooth.....	16
Otras características.....	18
Configuración del Arduino IDE.....	19
Configuración Adicional.....	23
Ejemplo de cableado del ESP32 Dev Kit C V2	27
Ejemplos de sketch.....	28



Introducción

El ESP32 Dev Kit C V2 es una placa de desarrollo creada en torno al chip ESP32 WROOM 32, que contiene un regulador de voltaje y un circuito programador USB para el chip ESP32, y otras características.

Para el desarrollo de aplicaciones se puede elegir entre Arduino IDE o ESP-IDF (plataforma nativa). La mayoría de los usuarios escogen el Arduino IDE por su simplicidad y compatibilidad. La comunidad de usuarios de Arduino es muy activa y soporta plataformas como ESP32.

ESP32 Dev Kit C V2 viene con un firmware preinstalado que permite trabajar con el lenguaje interpretado, enviando comandos a través del puerto serie (chip CP2102). Las placas ESP32 son una de las plataformas más utilizadas para proyectos de Internet de las Cosas (IoT).

La placa ESP32 Dev Kit C V2 está especialmente diseñada para trabajar en breadboard. Tiene un regulador de voltaje que le permite alimentarse directamente del puerto USB. Los pines de entrada/salida funcionan a 3.3V. El chip CP2102 se encarga de la comunicación USB a serie.

Especificaciones

Voltaje de la fuente de alimentación (USB)	5V DC
Voltaje Entrada/Salida	3.3V DC
Corriente de funcionamiento requerida	mínimo. 500mA
SoC	ESP32-WROOM 32
CPU	Xtensa® single-dual-core 32-bit LX6
Rango de frecuencias del reloj	80MHz / 240MHz
RAM	512kB
Memoria Flash Externa	4MB
Pines I/O	34
Canales ADC	18
Resolución ADC	12-bit
Canales DAC	2
Resolución DAC	8-bit
Interfaces de Comunicación	SPI, I2C, I2S, CAN, UART
Protocolos Wi-Fi	802.11 b/g/n (802.11n hasta 150 Mbps)
Frecuencia Wi-Fi	2.4 GHz - 2.5 GHz
Bluetooth	V4.2 - BLE y Bluetooth Classic
Antena Wireless	PCB
Dimensiones	56x28x13mm(2.2x1.1x0.5in)

ESP32 Dev Kit C V2

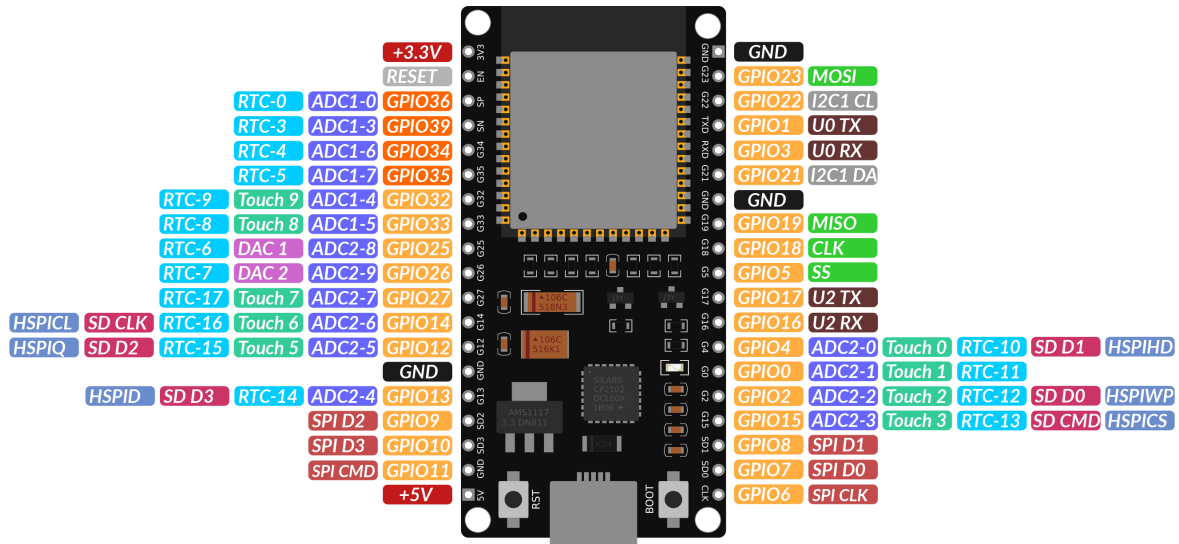
La serie de chips Wi-Fi ESP32 WROOM 32 es producida por Espressif Systems. ESP32 WROOM-32 es un módulo Wi-Fi adecuado para proyectos DIY en el Internet de las cosas (IoT). Este módulo viene con muchos GPIOs y soporte para una variedad de protocolos como SPI, I2C, I2S, UART, y más. Lo mejor es que viene con red inalámbrica incluida, lo que lo hace diferente a otros microcontroladores como el Arduino. Esto significa que puede controlar y monitorear fácilmente los dispositivos de forma remota a través de Wi-Fi y Bluetooth® a un precio asequible.

ESP32 WROOM 32 es un sistema en chip (SoC) que integra un microcontrolador Tensilica de 32 bits, interfaces periféricas digitales estándar, conmutadores de antena, balun de RF, amplificador de potencia, amplificador de recepción de bajo ruido, filtros y módulos de gestión de energía en un pequeño paquete. Ofrece Wi-Fi de 2.4 GHz (802.11 b/g/n, que soporta velocidades de hasta 150 MB/s), comunicación inalámbrica BLE y Bluetooth® clásica, 34 pines de I/O, interfaces I2C e I2S, ADC (conversión analógica a digital), DAC (conversión digital a analógica), interfaz SPI, UART en pines dedicados y PWM (Pulse Width Modulation).

El núcleo del procesador, denominado LX6 por Espressif, se basa en el controlador del procesador Xtensa® de doble núcleo LX6 de 32 bits y funciona a una frecuencia de entre 80-240MHz. Tiene 448kB de ROM para arranque, 520kB de SRAM en el chip y 4MB de memoria flash externa a la que se puede acceder a través de la interfaz SPI.

Disposición de los pines

El ESP32 Dev Kit C V2 tiene 38 pines. La disposición de los pines se presenta en la siguiente imagen:



- Orange Digital In/Out ports (all support PWM)
- Red Digital Input ports
- Purple Analog Input 12 bits, 0 to 3.3V
- Blue Analog Output 8 bits, 0 - 3.3V
- Green Capacitive Touch Sensor ports
- Light Blue I/O -pins from RTC ultra low power processor, usable in deep sleep mode
- Pink SD card interface
- Dark Red SPI bus for Flash-memory, do not use

The following pins show the default assignment. All these signals can be changed to any In/Out port. This applies also to UART0 and UART1, which cannot be accessed in the default assignment.

- Grey I2C bus (Wire)
- Green VSPI bus
- Brown Serial interfaces
- Blue HSPI bus

Para una descripción detallada del pinout y las capacidad de I/O, por favor consulte la hoja de datos que puede encontrar en el siguiente [link](#).

NOTA: La corriente máxima absoluta consumida por un GPIO es de 40mA según la sección "Recommended Operating Conditions" de la hoja de datos del ESP32.



Descripción de los pines

Al igual que una placa Arduino normal, el ESP32 Dev Kit C V2 tiene pines digitales de entrada/salida (pines GPIO - General Purpose Input/Output pins). Estas entradas/salidas digitales funcionan a 3.3V.

El voltaje de 5V no se debe conectar a ningún pin del chip ESP32!

Los pines no soportan 5V, si se aplica más de 3.3V en cualquier pin se destruirá el chip.

Los pines GPIO 34 a 39 son GPIOs - solamente pines de entrada. Estos pines no tienen resistencias internas de pull-up o pull-down. No se pueden usar como salidas, así que use estos pines sólo como entradas: GPIO 34, GPIO 35, GPIO 36, GPIO 39

Hay una flash SPI integrada en el chip ESP-WROOM-32. Los pines GPIO6 a GPIO 11 están expuestos en ciertas placas de desarrollo ESP32. Estos pines están conectados a la flash SPI integrada en el chip y no se recomiendan para otros usos.

GPIO 6 (SCK/CLK), GPIO 7 (SDO/SD0), GPIO 8 (SDI/SD1), GPIO 9 (SHD/SD2), GPIO 10 (SWP/SD3), GPIO 11 (CSC/CMD).



Pines del Sensor Táctil Capacitivo

El ESP32 tiene 10 sensores táctiles capacitivos internos. Los pines táctiles capacitivos también se pueden utilizar para despertar el ESP32 del Deep Sleep. Estos sensores táctiles internos están conectados a estos GPIOs: T0 (GPIO 4), T1 (GPIO 0), T2 (GPIO 2), T3 (GPIO 15), T4 (GPIO 13), T5(GPIO 12), T6 (GPIO 14), T7 (GPIO 27), T8 (GPIO 33), T9 (GPIO 32).

Pines del Convertidor Analógico a Digital

El ESP32 tiene 18x12 bits ADC (Analog to Digital converter) canales de entrada (mientras que el ESP8266 sólo tiene 1x 10 bits ADC). Estos son los GPIOs que se pueden utilizar como ADC y sus respectivos canales:

ADC1_CH0 (GPIO 36), ADC1_CH1 (GPIO 37), ADC1_CH2 (GPIO 38),
ADC1_CH3 (GPIO 39), ADC1_CH4 (GPIO 32), ADC1_CH5 (GPIO 33),
ADC1_CH6 (GPIO 34), ADC1_CH7 (GPIO 35), ADC2_CH0 (GPIO 4),
ADC2_CH1 (GPIO 0), ADC2_CH2 (GPIO 2), ADC2_CH3 (GPIO 15),
ADC2_CH4 (GPIO 13), ADC2_CH5 (GPIO 12), ADC2_CH6 (GPIO 14),
ADC2_CH7 (GPIO 27), ADC2_CH8 (GPIO 25), ADC2_CH9 (GPIO 26).

Pines del Convertidor Digital a Analógico

Hay 2 canales DAC (Digital to Analog converter) de 8 bits en el ESP32 para convertir las señales digitales en salidas de señales analógicas de voltaje. Estos son los canales DAC:

DAC1 (GPIO25), DAC2 (GPIO26).



Pines GPIO del Reloj en Tiempo Real

El ESP32 dispone de soporte GPIO para el RTC (reloj en tiempo real). Los GPIOs dirigidos al subsistema de bajo consumo del RTC pueden utilizarse cuando el ESP32 está en Deep Sleep. Estos GPIOs RTC pueden utilizarse para despertar al ESP32 del Deep Sleep cuando se está ejecutando el coprocesador de Ultra Bajo Consumo (ULP). Los siguientes GPIOs se pueden utilizar como fuente de activación externa: RTC_GPIO0 (GPIO36), RTC_GPIO3 (GPIO39), RTC_GPIO4 (GPIO34), RTC_GPIO5 (GPIO35), RTC_GPIO6 (GPIO25), RTC_GPIO7 (GPIO26), RTC_GPIO8 (GPIO33), RTC_GPIO9 (GPIO32), RTC_GPIO10 (GPIO4), RTC_GPIO11 (GPIO0), RTC_GPIO12 (GPIO2), RTC_GPIO13 (GPIO15), RTC_GPIO14 (GPIO13), RTC_GPIO15 (GPIO12), RTC_GPIO16 (GPIO14), RTC_GPIO17 (GPIO27).



Pines del PWM (Pulse Width Modulation)

El controlador PWM (modulación de ancho de pulso) del LED ESP32 tiene 16 canales independientes que se pueden configurar para generar señales PWM con diferentes propiedades. Todos los pines que pueden actuar como salidas pueden ser utilizados como pines PWM (los GPIOs 34 a 39 no pueden generar PWM). Para establecer una señal PWM, es necesario definir estos parámetros en el código: Frecuencia de la señal, ciclo de trabajo, canal PWM, GPIO por donde se quiere dar salida a la señal.

Pines de la interfaz I2C

El ESP32 tiene dos canales I2C y cualquier pin puede ser configurado como SDA o SCL. Cuando se utiliza el ESP32 con el Arduino IDE, los pines I2C por defecto son: GPIO 21 (SDA), GPIO 22 (SCL).

Pines de la interfaz SPI

Por defecto, la asignación de pines para los pines SPI es:

SPI	MOSI	MISO	CLK	CS
VSPI	GPIO 23	GPIO 19	GPIO 18	GPIO 5
HSPI	GPIO 13	GPIO 12	GPIO 14	GPIO 15

Pines de conexión

Los siguientes pines se utilizan para poner el ESP32 en modo bootloader o flashing: GPIO 0, GPIO 2, GPIO 4, GPIO 5 (debe estar "HIGH" durante el arranque), GPIO 12 (debe estar "LOW" durante el arranque), GPIO 15 (debe estar "HIGH" durante el arranque).

La mayoría de las placas de desarrollo ponen los pines en el estado correcto para el modo de flasheo o de arranque. Si algunos periféricos están conectados a los pines de conexión y el IDE no puede cargar el código o flashear el ESP32, puede ser porque esos periféricos están impidiendo que el ESP32 entre en el modo correcto. Después de reiniciar, flashear o arrancar, esos pines funcionan como se espera. Hay una guía de documentación sobre la selección del modo de arranque en el siguiente [link](#). Más explicaciones extensas no están en el alcance de este E-Book, así que por favor, consulte la hoja de datos.



Pines HIGH en el Arranque

Algunos GPIOs cambian su estado a HIGH o emiten señales PWM al arrancar o reiniciar. Esto significa que si se conectan salidas a estos GPIOs se pueden obtener resultados inesperados cuando el ESP32 se reinicia o arranca.

GPIO 1, GPIO 3, GPIO 5, GPIO 6 a GPIO 11 (conectados a la memoria flash SPI integrada del ESP32 - no se recomienda su uso), GPIO 14, GPIO 15.

Habilitación (EN)

Habilitación (Enable-EN) es el pin de habilitación del regulador de 3,3V.

Tiene un estado pull up y necesita ser conectado a tierra para desactivar el regulador de 3.3V. Esto significa que este pin puede ser conectado a un botón para reiniciar su ESP32, por ejemplo.



USB a Comunicación en Serie

El ESP32 Dev Kit C V2 tiene un puerto de conexión microUSB. Está diseñado alrededor del chip CP21202 fabricado por Silicon Laboratories que permite la comunicación USB a UART en serie. El chip tiene la característica de puerto COM virtual (VCP) que aparece como puerto COM en las aplicaciones de PC. La interfaz UART del CP2102 implementa todas las señales RS-232, incluidas las de control y handshaking, por lo que no es necesario modificar el firmware del sistema existente. Para poder utilizar el ESP32 es necesario instalar el controlador.



Comunicación WiFi

ESP32 Dev Kit C V2 tiene integrada la interfaz de comunicación Wi-Fi y puede funcionar en tres modos diferentes: Estación Wi-Fi, punto de acceso Wi-Fi y ambos al mismo tiempo. Soporta las siguientes características:

- Velocidades de datos 802.11b y 802.11g
- 802.11n MCS0-7 en los anchos de banda de 20MHz y 40MHz
- 802.11n MCS32
- Intervalo de guarda 802.11n de 0.4 μ S
- Velocidad de datos de hasta 150 Mbps
- Recepción STBC 2x1
- Potencia de transmisión de hasta 20 dBm
- Potencia de transmisión ajustable
- Diversidad y selección de antenas (hardware gestionado por software)

Comunicación Bluetooth

El ESP32 Dev Kit C V2 tiene un integrado Radio Bluetooth y soporta las siguientes características:

- Potencias de salida de transmisión de clase 1, clase 2 y clase 3 y más de 30 dB de rango de control dinámico
- Modulación $\pi/4$ DQPSK y 8 DPSK
- Alto rendimiento en la sensibilidad del receptor NZIF con más de 98 dB de rango dinámico
- Funcionamiento de clase 1 sin PA externo
- La SRAM interna permite la transferencia de datos a máxima velocidad, la mezcla de voz y datos y el funcionamiento completo de la piconet
- Lógica para la corrección de errores hacia delante, control de errores de cabecera, correlación de códigos de acceso, CRC, demodulación, generación de flujos de bits de encriptación, blanqueo y conformación de impulsos de transmisión
- ACL, SCO, eSCO y AFH
- CODEC de audio digital A-law, μ -law y CVSD en interfaz PCM
- CODEC de audio SBC
- Gestión de la energía para aplicaciones de bajo consumo
- SMP con AES de 128-bit



Además, la Radio Bluetooth es compatible con los siguientes protocolos de interfaz de comunicación:

- Interfaz UART HCI, hasta 4 Mbps
- Interfaz HCI SDIO / SPI
- Interfaz I2C
- Interfaz de Audio PCM / I2S.

Otras características

El chip ESP32-WROOM 32D tiene integrado un sensor de efecto Hall que detecta los cambios en el campo magnético de su entorno.


El sensor Hall se basa en una resistencia de N-carriers. Cuando el chip está en el campo magnético, el sensor Hall desarrolla un pequeño voltaje en la resistencia, que puede ser medido directamente por el convertidor analógico-digital (ADC), o amplificado por el preamplificador analógico de ultra bajo ruido y luego medido por el ADC.

El sensor de temperatura genera un voltaje que varía con la temperatura. El voltaje se convierte internamente a través de un convertidor analógico-digital en un código digital. El sensor de temperatura tiene un rango de -40°C a 125°C . Como el desplazamiento del sensor de temperatura varía de un chip a otro debido a la variación del proceso, junto con el calor generado por el propio circuito Wi-Fi (que afecta a las mediciones), el sensor de temperatura interno sólo es adecuado para aplicaciones que detectan cambios de temperatura en lugar de temperaturas absolutas y también para fines de calibración. Sin embargo, si el usuario calibra el sensor de temperatura y utiliza el dispositivo en una aplicación mínimamente encendida, los resultados podrían ser lo suficientemente precisos.

Configuración del Arduino IDE

Si el Arduino IDE no está instalado, seguir el siguiente [link](#) y descargar el archivo de instalación para el sistema operativo de su elección. La versión Arduino IDE utilizada para este E-Book es la **1.8.13**.

Download the Arduino IDE



The screenshot shows the Arduino IDE download page. On the left, there is a large teal circle containing the Arduino logo (an infinity symbol with a minus sign on the left and a plus sign on the right). To the right of the logo, the text reads: **ARDUINO 1.8.13**. Below this, it says: "The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions." On the right side of the page, there is a teal sidebar with white text. It lists the following options: "Windows Installer, for Windows 7 and up", "Windows ZIP file for non admin install", "Windows app Requires Win 8.1 or 10" (with a small icon), "Mac OS X 10.10 or newer", "Linux 32 bits", "Linux 64 bits", "Linux ARM 32 bits", "Linux ARM 64 bits", "Release Notes", "Source Code", and "Checksums (sha512)".

Para los usuarios de *Windows*, haga doble clic en el archivo descargado *.exe* y seguir las instrucciones de la ventana de instalación.

Az-Delivery

Para los usuarios de *Linux*, descargar un archivo con la extensión *.tar.xz*, que se debe extraer. Cuando se la extrae, ir al directorio extraído y abrir el terminal en ese directorio. Se deben ejecutar dos scripts *.sh*, el primero es *arduino-linux-setup.sh* y el segundo es *install.sh*.

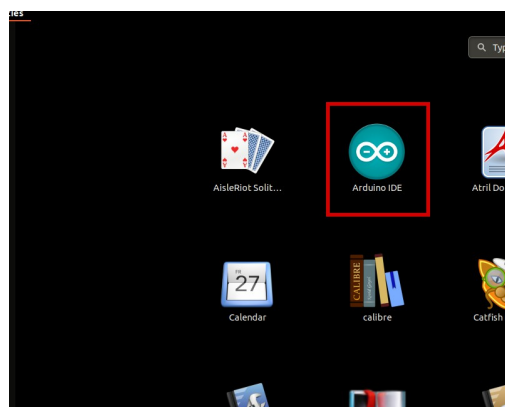
Para ejecutar el primer script en el terminal, abrir el terminal en el directorio extraído y ejecutar el siguiente comando:

```
sh arduino-linux-setup.sh user_name
```

user_name - es el nombre de un superusuario en el sistema operativo Linux. Una contraseña para el superusuario se debe introducir cuando se inicia el comando. Se debe esperar unos minutos para que el script complete todo.

El segundo script *install.sh* se debe utilizar después de la instalación del primer script. Ejecutar el siguiente comando en el terminal (directorio extraído): **sh install.sh**

Después de la instalación de estos scripts, ir a la sección *All Apps*, donde *Arduino IDE* está instalado.



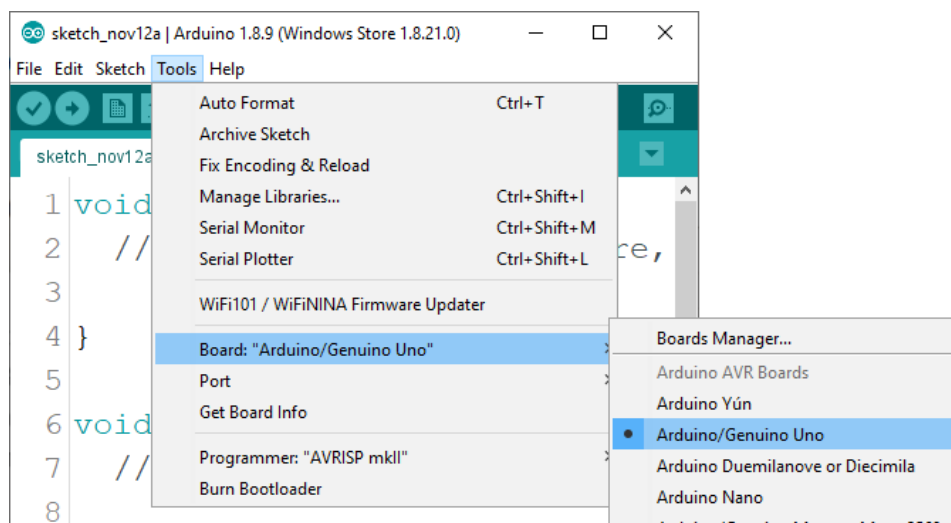
Az-Delivery

La mayoría de los sistemas operativos incluyen un editor de texto preinstalado (por ejemplo, *Windows* incluye *Notepad*, *Linux Ubuntu* incluye *Gedit*, *Linux Raspbian* incluye *Leafpad*, etc.). Todos estos editores de texto se pueden utilizar perfectamente para el objetivo de este E-Book.

Lo siguiente es comprobar si el PC puede detectar una placa Arduino. Para esto, se debe abrir el Arduino IDE recién instalado, e ir a:

Tools > Board > {your board name here}

{your board name here} debe ser el *Arduino/Genuino Uno*, como se puede observar en la imagen a continuación:



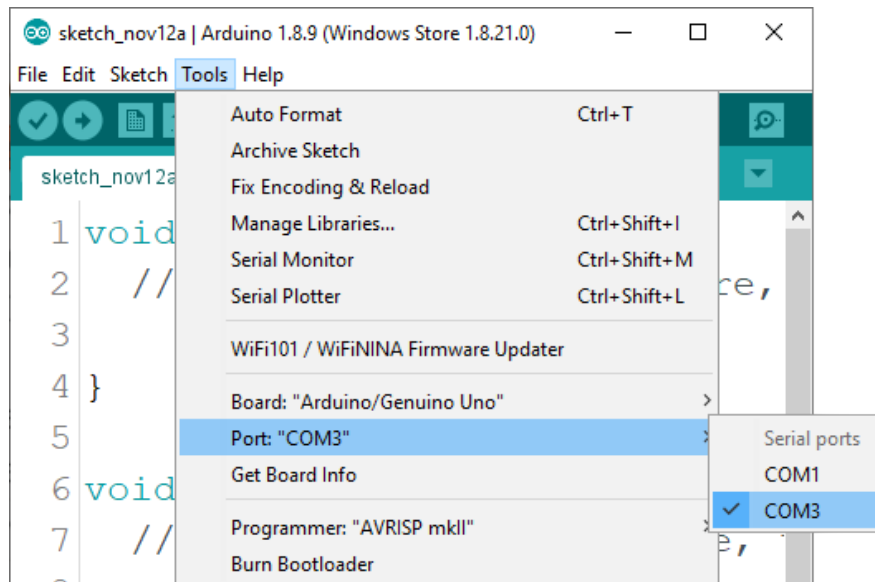
Se debe seleccionar el puerto al que está conectado la placa de Arduino. Ir

a: *Tools > Port > {port name goes here}*

y cuando la placa de Arduino está conectada al puerto USB, el nombre del puerto se puede observar en el menú desplegable de la imagen anterior.

Az-Delivery

Si el Arduino IDE se utiliza en *Windows*, los nombres de los puertos son los siguientes:



Para los usuarios de *Linux*, por ejemplo, el nombre del puerto es */dev/ttyUSBx*, donde *x* representa un número entero entre 0 y 9.



Configuración Adicional

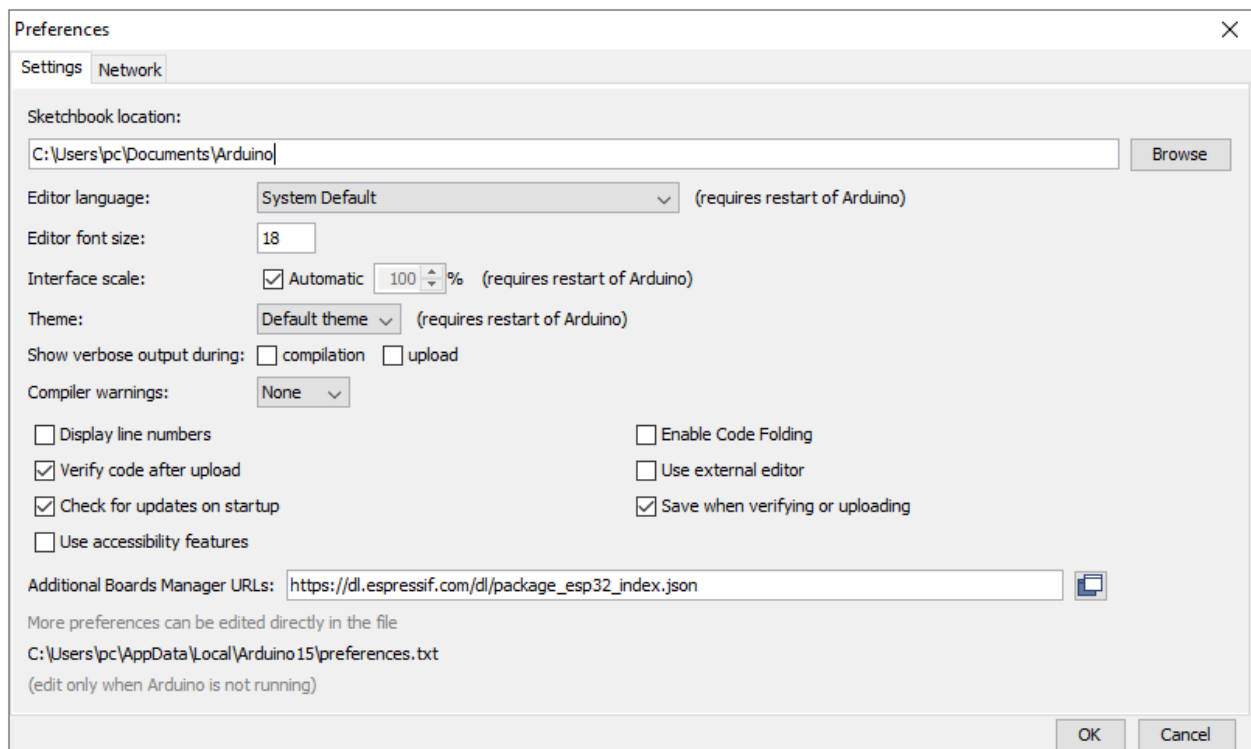
Para utilizar el ESP32 Dev Kit C V2 con el Arduino IDE, siga unos sencillos pasos. Antes de configurar el Arduino IDE, se debe instalar el controlador para la USB a comunicación Serial. Si el controlador no se instala automáticamente, hay una página de soporte que contiene los controladores para Windows/Mac o Linux y se puede elegir dependiendo de cuál se utilice. Los controladores se pueden descargar desde el siguiente [link](#).

Az-Delivery

A continuación, para instalar el soporte para la plataforma ESP32, abra Arduino IDE y vaya a: *File > Preferences*, y busque el campo Additional URLs.

A continuación, copie la siguiente URL:

https://dl.espressif.com/dl/package_esp32_index.json



Az-Delivery

Pegue este enlace en el campo Additional URLs. Si hay uno o más enlaces dentro de este campo, simplemente añada una coma después del último enlace, pegue el nuevo enlace después de la coma y haga clic en el botón **OK**.



Abra de nuevo el Arduino IDE y vaya a:

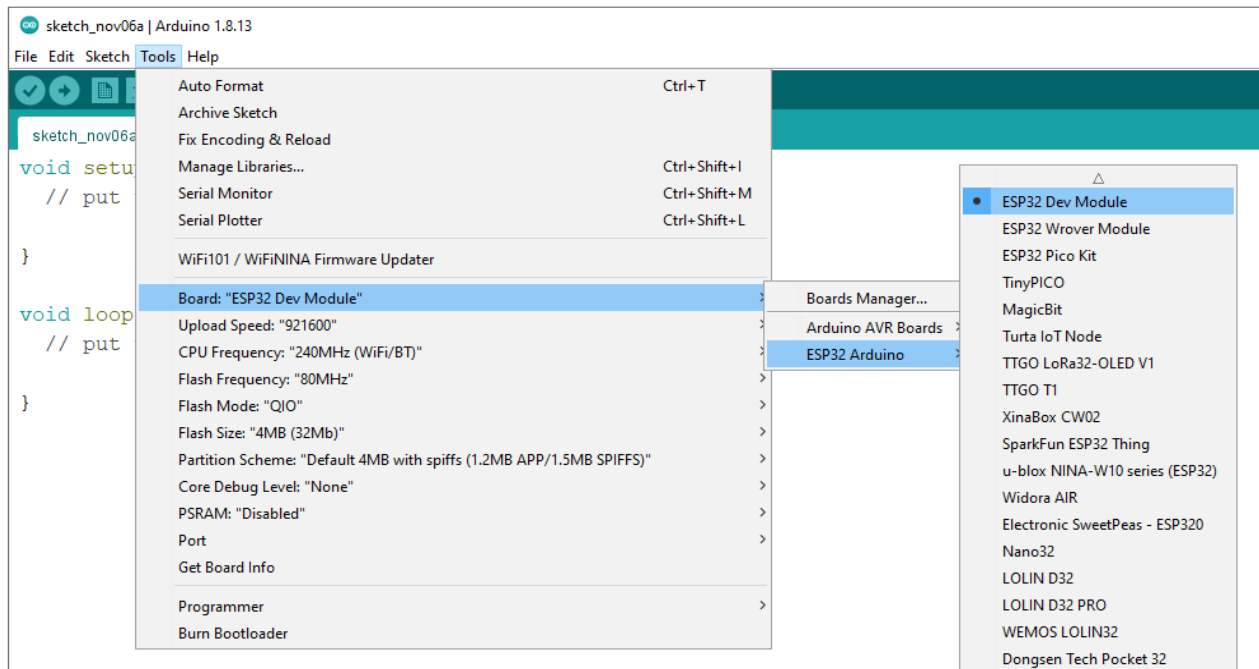
Tools > Board > Boards Manager

Cuando se abra la nueva ventana, escriba *esp32* en la caja de búsqueda e instale la placa llamada *esp32* hecha por Espressif Systems, como se observa en la siguiente imagen:

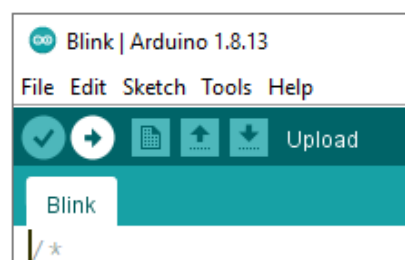


Para seleccionar la placa ESP32, vaya a:

Tools > Board > ESP32 Arduino > ESP32 Dev Module



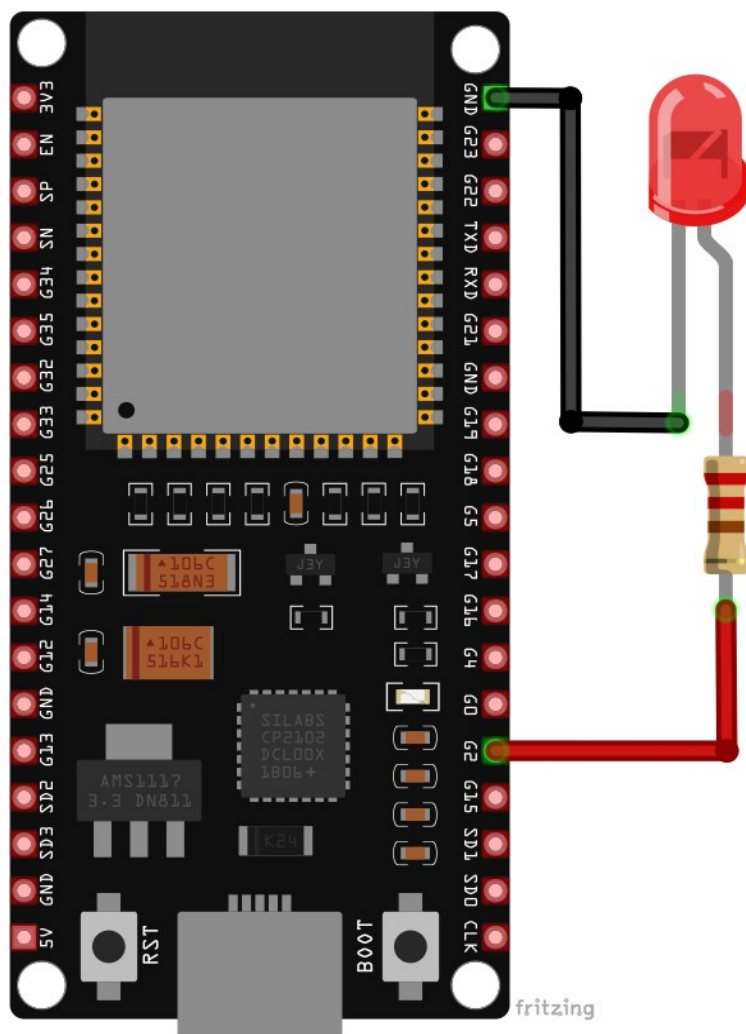
Para cargar el código del sketch a la placa ESP32, primero seleccione el puerto en el que conectó la placa. Vaya a: *Tools > Port > {port name}*



Si la carga no funciona en el primer intento, puede ser útil pulsar el "botón de arranque" del módulo durante la carga. Por favor, utilice un cable USB 2.0 certificado para la programación.

Ejemplo de cableado del ESP32 Dev Kit C V2

Conecte el ESP32 Dev Kit C V2 con un LED y una resistencia como se observa en el siguiente diagrama de conexión:



Pin de ESP32 Dev Kit C V2	Pin de LED	Color del cable
GPIO2 (pin2)	Ánodo (+) a través de la resistencia	Cable Rojo
GND	Cátodo (-)	Cable Negro

Az-Delivery

Ejemplos de sketch

LED Parpadeante

```
int ledPin = 2;

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  digitalWrite(ledPin, HIGH);
  delay(1000);
  digitalWrite(ledPin, LOW);
  delay(1000);
}
```



PWM - Pulse Width Modulation

```
#define LEDC_CHANNEL_0 0
#define LEDC_TIMER_13_BIT 13
#define LEDC_BASE_FREQ 5000
#define LED_PIN 2

int brightness = 0;
int fadeAmount = 5;

void ledcAnalogWrite(uint8_t channel, uint32_t value, uint32_t valueMax = 255)
{
    uint32_t duty = (8191 / valueMax) * min(value, valueMax);
    ledcWrite(channel, duty);
}

void setup() {
    ledcSetup(LEDC_CHANNEL_0, LEDC_BASE_FREQ, LEDC_TIMER_13_BIT);
    ledcAttachPin(LED_PIN, LEDC_CHANNEL_0);
}

void loop() {
    ledcAnalogWrite(LEDC_CHANNEL_0, brightness);
    brightness = brightness + fadeAmount;
    if (brightness <= 0 || brightness >= 255) {
        fadeAmount = -fadeAmount;
    }
    delay(30);
}
```



Ahora es el momento de aprender y hacer sus propios proyectos. Esto puede hacerlo con la ayuda de muchos scripts de ejemplo y otros tutoriales que usted puede encontrar fácilmente en Internet.

Si está buscando productos de alta calidad para Arduino y Raspberry Pi, AZ-Delivery Vertriebs GmbH es el lugar apropiado. Por su compra usted recibirá varios ejemplos de aplicación, guías completas de instalación, EBooks, librerías y la asistencia de nuestros expertos técnicos.

<https://az-delivery.de>

¡Disfrute!

Impressum

<https://az-delivery.de/pages/about-us>