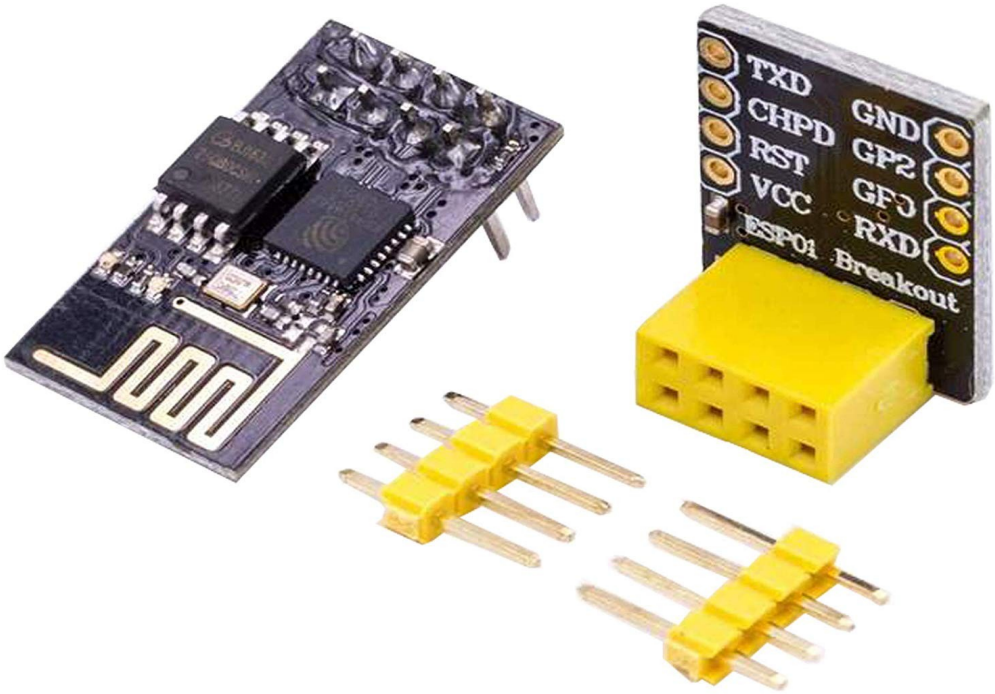
¡Bienvenido!

Muchas gracias por comprar nuestro AZ-Delivery Módulo ESP8266-01S con Adaptador Breadboard. En las siguientes páginas, se presentará como utilizar y configurar este práctico dispositivo.

¡Diviértase!



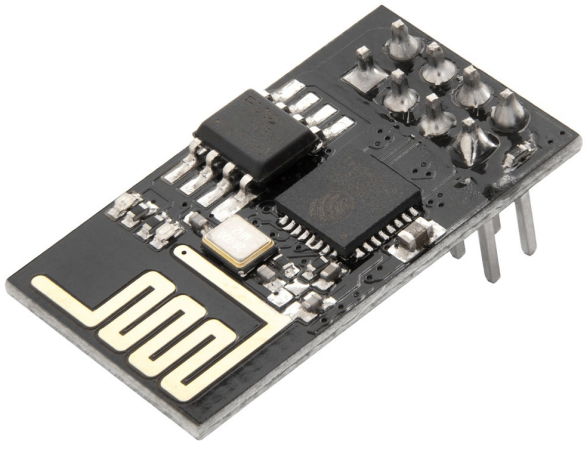
Ihr Experte für Mikroelektronik!


AZ-Delivery
Ihr Experte für Mikroelektronik!


AZ-Delivery
Ihr Experte für Mikroelektronik!

El módulo ESP8266 es un Sistema en un Chip (SoC), fabricado por la empresa china Espressif. Consiste en un microcontrolador Tensilica L106 de 32 bits y un transceptor WiFi. Tiene 11 pines GPIO (General Purpose Input/Output), y también una entrada analógica. Esto permite que pueda programar como cualquier otro microcontrolador normal de placa de microcontrolador o cualquier otro. Lo mejor de ESP8266 es que tiene comunicación WiFi con éste, así que puede utilizarlo para conectarse a la red WiFi, conectarse a internet, alojar un servidor web con páginas web, conectar su smartphone con él, etc.

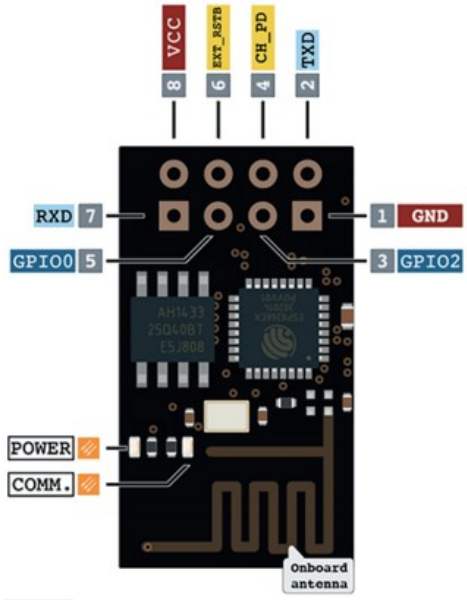
El ESP8266 es un increíble chip WiFi que se puede utilizar en varias aplicaciones de automatización del hogar. Debido al potente procesador de 80MHz y a una gran memoria de 1MB, el ESP8266 también puede funcionar de forma autónoma.



Características

- 802.11 b/g/n
- MCU de 32 bits de baja potencia integrada
- CAD de 10 bits integrado
- Pila de protocolo TCP/IP integrada
- Interruptor TR, balun, LNA, amplificador de potencia y red de adaptación integrados
- PLL, reguladores y unidades de gestión de energía integrados
- Soporta diversidad de antenas
- Wi-Fi 2.4 GHz, soporta WPA/WPA2
- Soporta los modos de operación STA/AP/STA+AP
- Soporta la función Smart Link para dispositivos Android e iOS
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IRDA, PWM, GPIO
- STBC, 1x1 MIMO, 2x1 MIMO
- Agregación de A-MPDU y A-MSDU e intervalo de guarda de 0.4s
- Energía de Deep Sleep <10uA, corriente de fuga de apagado <5uA
- Despierta y transmite paquetes en < 2ms
- Consumo de energía en espera de < 1.0mW (DTIM3)
- +20dBm de potencia de salida en el modo 802.11b
- Rango de temperatura de funcionamiento: -40 °C ~ 125 °C

Se encuentran diversas placas de desarrollo basadas en ESP8266, y la ESP8266-01S es una de ellas. Esta placa tiene 8 pines (disposición de los pines en la imagen de abajo).

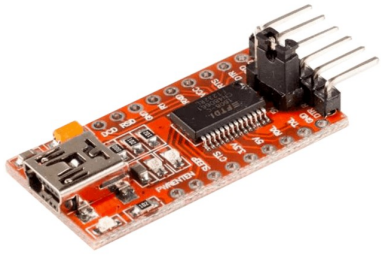


Como se puede observar, se dispone de dos pines GPIO libres, GPIO0 y GPIO2. Así que se puede utilizar GPIO0 y GPIO2 como entradas o salidas digitales como en un microcontrolador normal. En el ejemplo (más adelante en el E-Book) se utilizará GPIO2 para leer los datos del módulo sensor de temperatura DHT22.

Se puede programar el ESP8266-01S con varios métodos diferentes, pero sólo se revisará el método utilizando el Arduino IDE. Si previamente se han utilizado las placa de microcontrolador, entonces esto será muy sencillo. Sólo tomar en cuenta que no se limita a esta opción, hay otras maneras de programar el ESP8266 (el oficial ESP SDK para la programación en C, el intérprete de Lua, el firmware de MicroPython, son algunas de éstas).

Para programar el ESP8266-01S se deben utilizar los pines RX y TX, y se debe conectarlos al dispositivo con interfaz serial como el módulo FT232RL o el adaptador USB escogido, o la placa de microcontrolador.

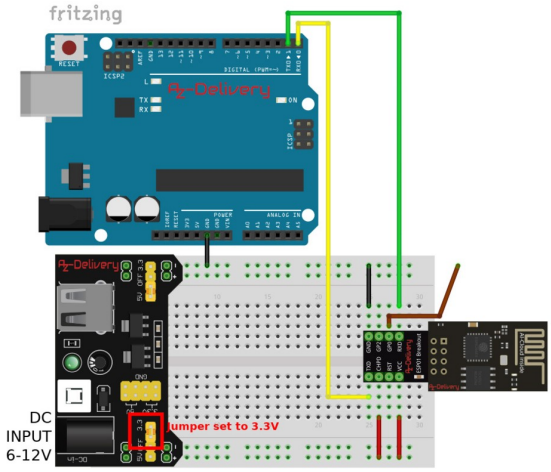
El FT232RL es muy popular, ya que puede cambiar entre la lógica de 5V y 3.3V. **¡Es esencial que el convertidor de USB a Serie funcione a 3.3V!** **¡Si utiliza el ESP8266 en un modelo FT232RL a 5V, se dañará el ESP8266!**



Adaptador FT232RL

https://www.az-delivery.de/products/ftdi-adapter-ft232rl?ls=en

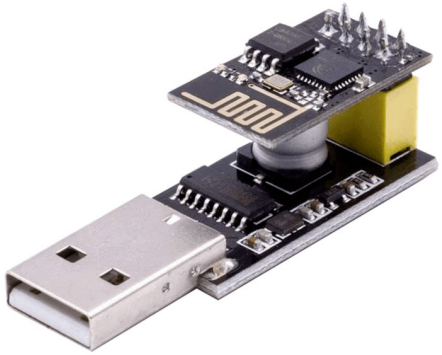
Si se utiliza placa de microcontrolador para programar el ESP8266-01S, lo siguiente siguiente es el diagrama de conexión (no se lo utilizará en este E-Book):



Se utiliza éste en nuestro E-Book del ESP8266-01S con Relé:

https://www.az-delivery.de/products/esp8266-01-mit-relais-kostenfreies-e-book?_pos=2&_sid=ffbf81394&_ss=r&ls=de

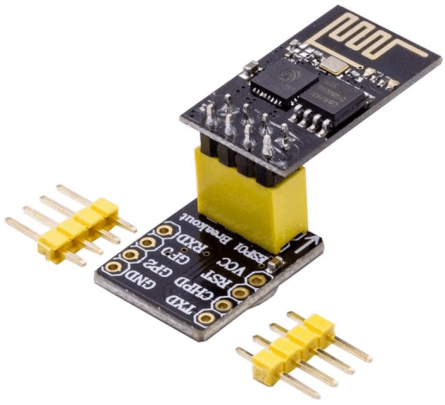
Hay otra forma de programar el ESP8266-01S y es con un adaptador USB dedicado (imagen inferior), pero tampoco se la revisará en este E-Book.



Adaptador USB con ESP8266-01S:

https://www.az-delivery.de/products/esp8266-01s-mit-usb-adapter?ls=en

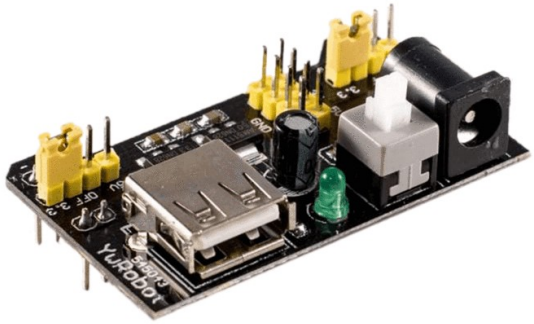
En este E-book se utilizará la placa breakout breadboard y el adaptador FT232RL para programar el ESP8266-01S, debido a que es más sencillo. Sin embargo, cualquier forma que utilice es similar.



Placa breakout breadboard con ESP8266-01S:

https://www.az-delivery.de/products/esp8266-01s-mit-breadboardadapter?ls=en

Se debe asegurar de suministrar energía a ESP8266-01S con una fuente de alimentación separada. Se puede utilizar nuestra fuente de alimentación MB102 breadboard. Es una fuente de alimentación muy simple y práctica que acepta entradas de 6 a 12V DC y puede emitir tanto +3.3V como +5V.



Fuente de alimentación MB102 para el breadboard
https://www.az-delivery.de/products/mb102-breadboard?ls=en

Hay dos formas muy diferentes de programar el ESP8266-01S. La primera es utilizando comandos "AT" (comandos de atención) a través de la Interfaz Serial, y la segunda es programando el chip ESP8266 como un microcontrolador (utilizando Arduino IDE).

Debido a que la segunda forma brinda más libertad para crear, sólo se revisará esta forma de programación ESP8266-01S en este E-Book.

Habilitando el chip

Primero, se tienen que agregar algunas resistencias para encender el ESP8266-01S, y también para seleccionar el modo de arranque correcto (programación o modo normal). Para asegurar el uso correcto del módulo ESP8266-01S se deben seguir los siguientes pasos:

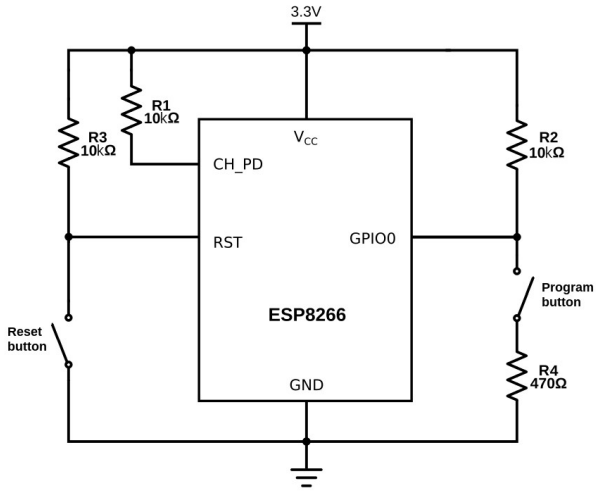
1. Habilitar el chip conectando el pin **CH_PD** (Chip Power Down, etiquetado como *CH_EN*, o *CH_PC*) a **VCC** a través de un resistencia **10kΩ**.

2. Seleccionar el modo *NORMAL* conectando el **GPIO0** a **VCC** a través de una resistencia **10KΩ**. (en el modo *PROGRAM* se conecta a **GND**, y para el modo *NORMAL* se lo deja desconectado)

3. Prevenir los restablecimientos aleatorios conectando el pin **RST** (reinicio) a **VCC** través de una resistencia **10KΩ**. (se puede agregarlo, pero no es necesario, se lo deja desconectado)

Agregando los botones de RESET y PROGRAM

Nuestras placas ESP8266-01S no tienen un botón de reinicio o un botón para el modo de programación, pero se podría (aunque no es necesario) agregarlos, como en la imagen inferior.



Para el botón de reinicio conecte un botón pulsador entre el pin *RST* y *GND*, y agregar una resistencia PULL-UP 10kΩ (R3 en la imagen superior) entre RST y VCC.

Para colocar el chip en modo de programación, se tiene que jalar el pin *GPIO0* hacia abajo durante el arranque (conectarlo al GND). Si está hacia arriba o no está conectado, el chip comenzará a arrancar en modo normal. Debido a que es posible utilizar *GPIO0* como salida, no se puede hacer un puente directo a tierra, que podría dañar el chip. Para evitar esto, conecte la resistencia 470Ω (R4 en la imagen superior) entre este botón y el GND.

Es importante que esta resistencia sea lo suficientemente baja, de lo contrario, será jalada hacia arriba por la resistencia 10KΩ (R2 en la imagen superior) si elige agregar R2 como en la sugerencia de la página anterior (en 2.).

Para el objetivo de este E-Book, no se utiliza ningún botón. Para poner el ESP en modo de programación, se utiliza un cable puente, que conecta el *GPIO0* a *GND* para el modo de programación*;* y se lo desconecta del *GND* en el modo normal.

Para reiniciar ESP8266-01S, apague/encienda la fuente de alimentación.

Se debe tener en cuenta lo siguiente cuando se utiliza un ESP8266-01S: Lo más importante es que funcione a 3.3V, así que si se lo conecta a una fuente de alimentación de 5V, dejará de funcionar. A diferencia de algunas placa de microcontrolador de 3.3V, los pines de I/O del ESP8266 no soportan 5V, así que si se utiliza un 5V convertidor USB a Serial, o sensores de 5V, etc. dejará de funcionar.

También se debe tener en cuenta que el ESP8266 sólo puede generar o recoger 12mA por cada pin de salida, comparado con 20 - 40mA de la mayoría de los placa de microcontrolador!

Por último se debe tener en cuenta que el ESP8266 tiene que compartir los recursos del sistema y el tiempo de la CPU entre su sketch y el controlador WiFi. Además, características como PWM, interrupciones o I2C son emuladas en el software. Por otro lado, la mayoría de los placa de microcontrolador tienen partes de hardware dedicadas solo para estas

tareas. Sin embargo, para la mayoría de las aplicaciones, esto no es un problema, pero se debe analizar ésto cuando se diseña un sketch.

El ESP8266 como microcontrolador - Hardware

Mientras que el ESP8266 se utiliza a menudo como un "inútil" puente Serial a WiFi, por su cuenta es un microcontrolador muy poderoso.

Digital I/O

Al igual que un placa de microcontrolador normal, el ESP8266 tiene pines de entrada/salida digitales o GPIO - General Purpose Input/Output pins. Como su nombre indica, se pueden utilizar tanto entradas digitales para leer un voltaje digital, o como salidas digitales para la salida de 0V (recogida de corriente) o 3.3V (fuente de corriente).
El ESP8266 es un microcontrolador de 3.3V, por lo que sus I/O también funcionan a 3.3V.
- Los pines no toleran 5V, si se aplica más de 3.6V **EN CUALQUIER PIN, el chip dejará de funcionar!**
- ¡La máxima corriente que se puede extraer de un solo pin GPIO es de 12mA!

Pines utilizables

El ESP8266 tiene 17 pines GPIO (0-16), sin embargo, sólo se pueden utilizar 2 de ellos, porque para el ESP8266-01S sólo hay dos disponibles en los pines de ruptura. Si trata de utilizar otros pines, podría dañar el programa. Los pines *GPIO1* y *GPIO3* se utilizan como *TX* y *RX* del puerto Serial del hardware (*UART*). Se pueden utilizar como GPIOs, pero en la

mayoría de los casos, no se deberían utilizar como GPIOs al enviar/recibir datos seriales.

Modos de arranque

Como se mencionó en el capítulo anterior, algunos pines GPIO tienen una función especial durante el arranque, seleccionan 1 de 3 modos de arranque:

GPIO0	GPIO2	Modo
0V	3.3V	Cargador de arranque Uart (modo PROGRAM)
3.3V	3.3V	Sketch de arranque (SPI flash)
x	x	Modo SDIO (no se utiliza para Atmega328P Board)

Nota: No se tiene que agregar una resistencia externa pull-up a GPIO2, la interna se habilita en el arranque.

Se debe asegurar de que estas condiciones se cumplan agregando resistencias externas en el capítulo anterior. Sin embargo, esto tiene algunas implicaciones:
» GPIO0 se jala hacia arriba durante el modo de operación normal, por lo que no se puede utilizar como una entrada Hi-Z.
» GPIO2 no debe estar hacia abajo al arrancar, así que no puede conectarle un interruptor.

Resistencias internas pull-up/-down

Todos los pines GPIO de ESP8266 tienen una resistencia pull-up integrada, como en un Atmega328P Board. A diferencia, el GPIO16 tiene una

resistencia de pull-down integrada (pero no se puede acceder a ella en el ESP8266-01S, así que no importa).

PWM

A diferencia de la mayoría de los chips Atmel, el ESP8266 no soporta el hardware PWM; sin embargo, el software PWM es compatible con todos los pines digitales. El rango predeterminado de PWM es de 10 bits a 1kHz, pero esto se puede cambiar (hasta >14 bits a 1kHz).

Interfaz Serial

El ESP8266 tiene dos hardware *UARTS* (puertos Serial), pero sólo se puede utilizar uno en el módulo ESP8266-01S, *UART0* en los pines 1 y 3 (*TX0* y *RX0* respectivamente).

El ESP8266 como un microcontrolador - Software

La mayor parte de la funcionalidad del microcontrolador del ESP utiliza exactamente la misma sintaxis que un placa de microcontrolador normal, lo que permite que sea muy fácil empezar.

Digital I/O

Al igual que con un placa de microcontrolador normal, puede configurar la función de un pin utilizando el pin *Mode (pin, mode)*, donde "*pin"* es el número de *GPIO* , y "*mode"* puede ser *INPUT* (que es el valor por defecto), o *OUTPUT*, o *INPUT_PULLUP* para habilitar las resistencias pull-up incorporadas para *GPIO 0-15*. Para habilitar las resistencias pull-down para GPIO16, tiene que utilizar *INPUT_PULLDOWN_16* (para ESP8266-01S, no puede acceder a GPIO16 así que esto es en general para ESP8266). Para establecer un pin de salida *HIGH* (3.3V) o *LOW* (0V), utilice *digitalWrite(pin, value)* donde "*pin"* es el pin digital, y "*value"* es 1 o 0 (o *HIGH* y *LOW*). Para leer una entrada, utilice *digitalRead(pin).* Para habilitar PWM en un determinado pin, utilice *analogWrite(pin, value)* donde "*pin"* es el pin digital, y "*value"* un número entre 0 y 1023. Puede cambiar el rango de la salida PWM utilizando *analogWriteRange(new_range).* La frecuencia se puede cambiar utilizando *analogWriteFreq(new_frequency)*, "*new_frequency"* debe estar entre 100Hz y 1000Hz.

Comunicación Serial

Para utilizar *UART0* (*TX* = *GPIO1*, *RX* = *GPIO3*), puede utilizar el objeto Serial, justo como en un placa de microcontrolador: *Serial.begin(baud_reate)*. **Todas las funciones de placa de microcontrolador Stream, como reed*, write, print, println*, etc., también se soportan.**

Compartiendo el tiempo de CPU con la parte RF

Un aspecto que se debe tener en cuenta al escribir programas para el ESP8266 es que su sketch tiene que compartir recursos (tiempo de CPU y memoria) con el WiFi y los TCP-stacks (el software que se ejecuta en segundo plano y se encarga de todas las conexiones WiFi e IP). Si su código tarda demasiado en ejecutarse, y no permite que los TCP-stacks cumplan con su labor, esto puede fallar, o puede perder datos. Es mejor mantener el tiempo de ejecución de su bucle por debajo de un par de cientos de milisegundos. Cada vez que el bucle principal se repite, su sketch cede al WiFi y TCP para manejar todas las solicitudes de WiFi y TCP. Si su bucle tarda más tiempo que esto, tendrá que proporcionar exclusivamente tiempo de CPU a WiFi/TCP stacks, utilizando *delay(0)* o *yield()*. Si no lo hace, la comunicación de red no funcionará como se espera, y si es más larga de 3 segundos, el soft WDT (Watchdog Timer) reiniciará el ESP. Si el soft WDT está desactivado, después de un poco más de 8 segundos, el hardware WDT reiniciará el chip. Sin embargo,

desde la posición de un microcontrolador, 3 segundos es un tiempo excesivo (240 millones de ciclos de reloj), así que a menos que haga un número extremadamente pesado de crujidos, o envíe cadenas extremadamente largas a través de la serie, no se verá afectado por esto. Sólo tenga en cuenta que si agrega el *yield()* dentro de sus bucles *for* o *while* eso podría tomar más de 100ms.

WiFi

Utilizar el ESP8266 como un simple microcontrolador es excelente, pero la razón por la que la mayoría de la gente lo utiliza, es por su capacidad WiFi. Soporta protocolos de red como WiFi, TCP, UDP, HTTP, DNS, etc.

Cargando los sketches en el ESP8266

El procedimiento de carga de las placas ESP8266 es un poco diferente del procedimiento normal de placa de microcontrolador. La mayoría de las placa de microcontrolador se reinician automáticamente cuando se carga un nuevo programa, y entran automáticamente en modo de programación. Mientras que, en las placas ESP8266-01S tiene que entrar manualmente en el modo de programación, e incluso hay que reiniciarlas manualmente después de programar el módulo.

REINICIO Manual y PROGRAMACIÓN Manual

Si no dispone de un convertidor USB-a-Serie con líneas *DTR* y *RTS*, también puede utilizar los botones *RESET* y *PROGRAM* que se escribió en uno de los capítulos anteriores.

Para colocar el ESP8266 en el modo *PROGRAM*, *GPIO0* debe estar abajo al arrancar. Para el objetivo de esta guía, se utilizan cables de puente en lugar de botones, así que, este es el proceso sobre cómo cambiar entre el modo *NORMAL* y el modo *PROGRAM*:

» Para ingresar al modo *PROGRAM*:

1. Apague el módulo (desconecte la fuente de alimentación)
2. Conecte el pin *GPIO0* a tierra (GND) a través de un cable de puente.
3. Encienda el módulo (conecte la fuente de alimentación), y ESP8266-01S ingresa al modo *PROGRAM*.

» Para ingresar al modo *NORMAL*:

1. Apague el módulo (desconecte la fuente de alimentación)
2. Deje el pin *GPI0* sin conectar o desconéctelo de GND, si estaba previamente conectado. (¡sólo asegúrese de que el cable de puente no esté conectado a nada!)
3. Encienda el módulo (conecte la fuente de alimentación), y ESP8266-01S ingresa al modo *NORMAL*.

Por supuesto, puede agregar los botones *RESET* y *PROGRAM*, y así simplificar el proceso. Si elige seguir este camino, entonces los procesos para entrar en los modos *PROGRAM o NORMAL* son:

» Para ingresar al modo *PROGRAM*:

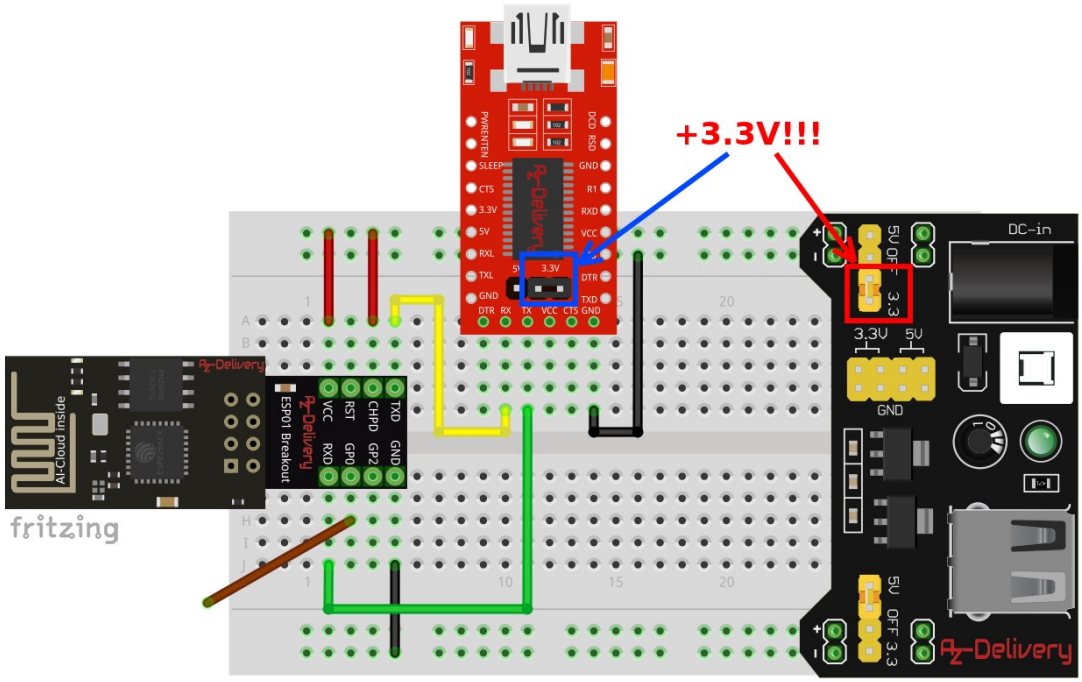
1. Presione y mantenga el botón *RESET*
2. Presione y mantenga el botón *PROGRAM*
3. Suelte el botón *RESET*, el ESP se iniciará en el modo *PROGRAM*
4. Suele el botón *PROGRAM*
5. Cargue el sketch

» Para ingresar al modo *NORMAL*:

1. Presione y libere el botón *RESET* para reiniciar el ESP8266-01S. El botón *PROGRAM* debe permanecer libre durante este proceso.

Diagrama de conexión ESP8266-01S

Para hacer todas las sugerencias como en los capítulos anteriores, conecte el ESP8266-01S como se observa en el siguiente diagrama de conexión:



¡Para los tres módulos conecte los pines GND juntos! (**Cables negros**)

Para FT232RL y ESP8266-01S:

1. Asegúrese de establecer el puente de referencia de voltaje a 3.3V (como para FT232RL esto también se aplica para MB102)

2. Conecte el pin **GND** del FT232RL al pin **GND** del ESP8266-01S.

3. Conecte el pin **RX** del FT232RL al pin **TX** del ESP8266-01S.
   Cable amarillo en el diagrama de conexión superior.
4. Conecte el pin **TX** del FT232RL al pin **RX** del ESP8266-01S. **
   Cable verde en el diagrama de conexión superior.

VCC del ESP8266-01S está conectado a +3.3V (**Cable rojo**).
CHPD pin del ESP8266-01S está conectado a VCC (**Cable rojo**).

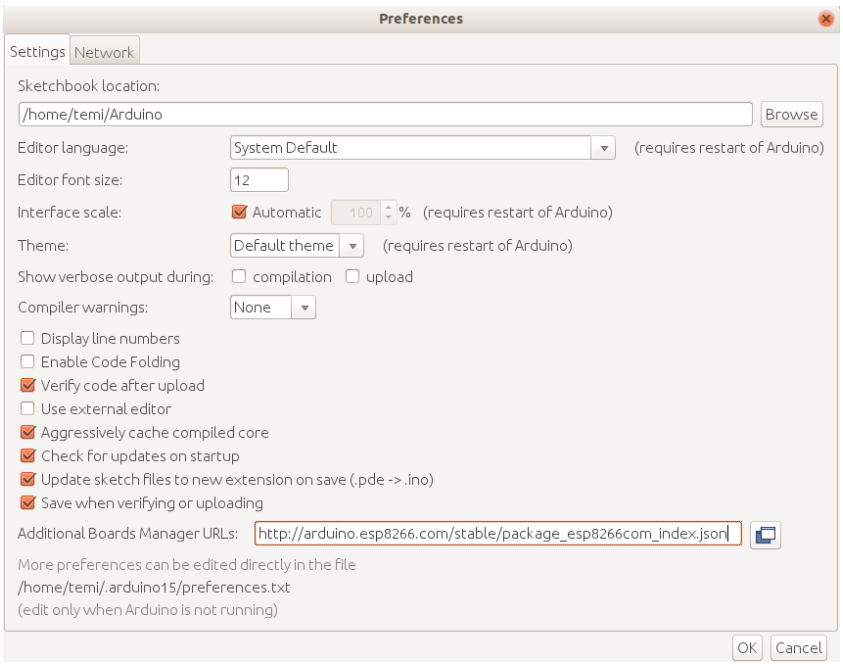
Para el modo *NORMAL*, el pin *GPIO0* tiene que estar desconectado al momento de arrancar (**Cable café**), pero cuando se quiere programar el ESP8266-01S, primero se tiene que apagar el módulo (desconectándolo de la fuente de alimentación), conecte el pin *GPIO0* a GND (**Cable café**) y después vuelva a conectar el ESP a la fuente de alimentación.

Para cambiar al modo *NORMAL* de nuevo, se necesita apagar el módulo de nuevo, y luego desconectar el pin *GPIO0* (**Cable café** mantener desconectado como en el diagrama de conexión), y encienda el módulo ESP de nuevo, conectarlo de nuevo a la fuente de alimentación.

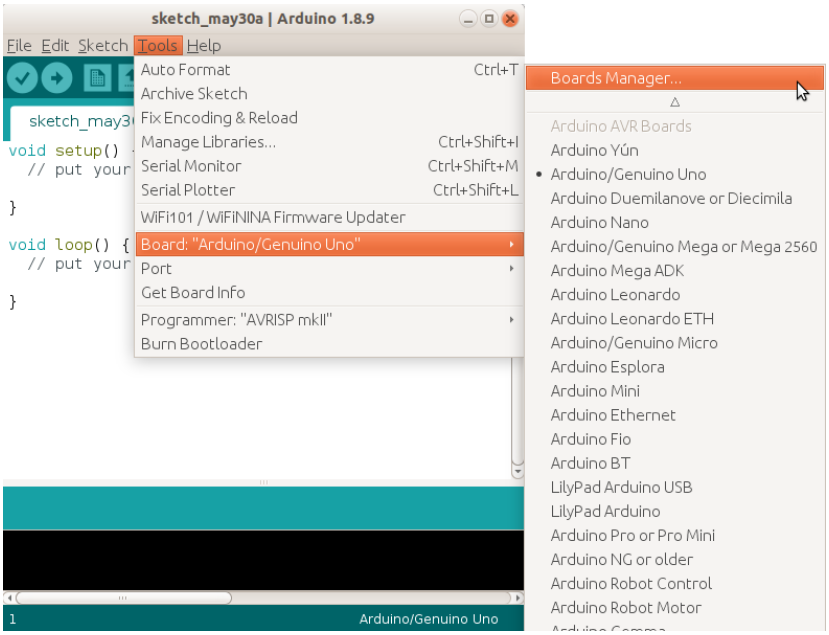
Software

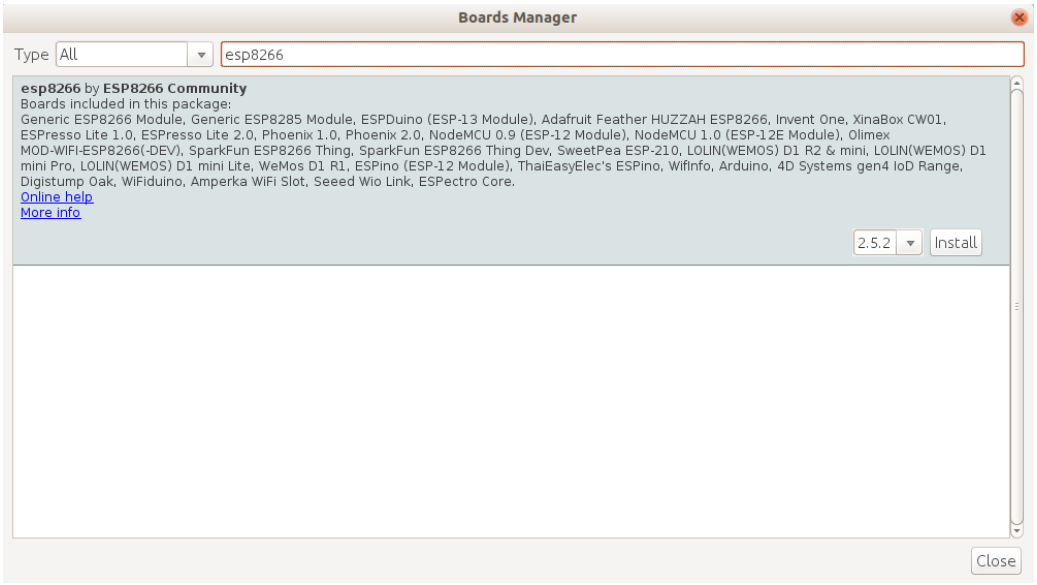
Solo se mostrará como utilizar el ESP8266-01S con Arduino IDE.
El primer paso es descargar e instalar Arduino IDE. Si ya lo tiene descargado, genial, en caso que aún no lo esté, ir al siguiente link: https://www.arduino.cc/en/Main/Software para descargarlo e instalarlo.

Para programar el ESP8266-01S, necesitará un plugin para el Arduino IDE, que se puede descargar desde GitHub https://github.com/esp8266/Arduino manualmente, pero es más fácil simplemente agregar la URL en el Arduino IDE. Abrir el IDE, ir a *File* > *Preferences*, pegar la URL:
http://arduino.esp8266.com/stable/package_esp8266com_index.json
en el campo de URLs *"Additional Board Manager"*. (Puede agregar múltiples URLs, separándolas con comas.)



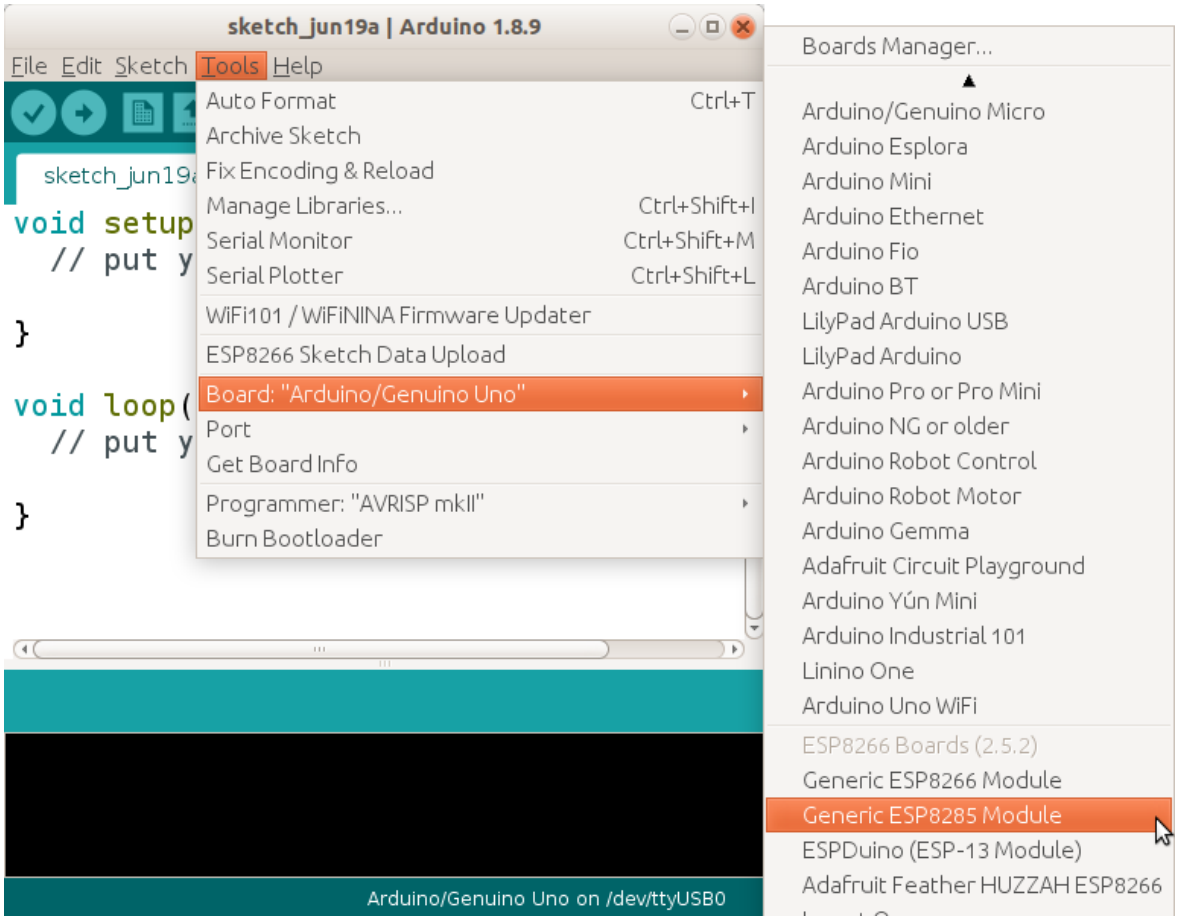
A continuación, ir a: *Tools* > *Board* > *Board Manager* y buscar "*esp8266*". Seleccionar la última versión, y dar clic en *Install*. Después de esto, se instalarán varios ejemplos de sketch, y placas.



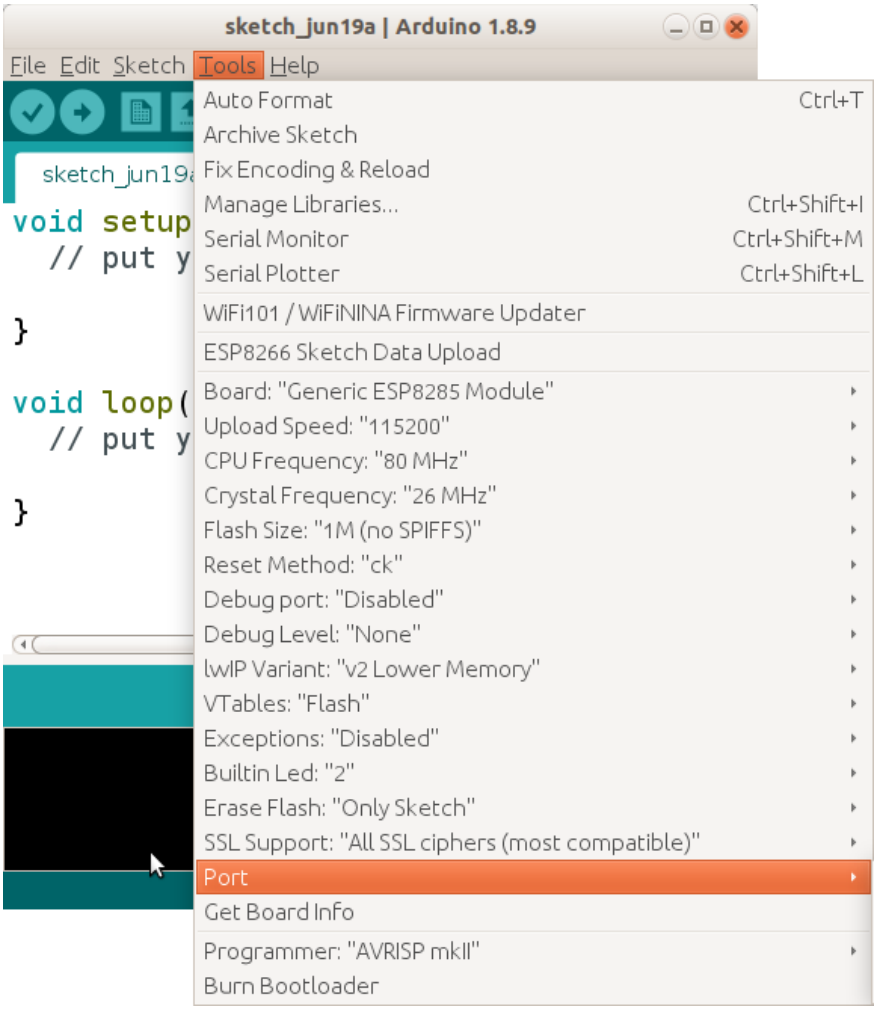


Ahora se tiene que configurar qué placa programar.

Ir a: *Tools > Board > {board name}* y escoger *"Generic ESP8266 Module".*



Cuando escoge esta placa hay otras opciones que también puede configurar. Cuando abra los menús desplegables *"Tools"* después de escoger *"Generic ESP8266 Module"*, habrán varias opciones que pueda configurar. Para el propósito de este E-book déjelo como está, porque no se va a entrar en detalles con esto. Sólo escoja el puerto al que está conectado su placa.



Ejemplo de aplicación

En este ejemplo se mostrará cómo conectar el ESP8266-01S con el sensor de temperatura y humedad del DHT22.

Escribimos sobre el sensor DHT22 en nuestro E-Book - Guía de inicio rápido del sensor DHT22:

https://www.az-delivery.de/products/dht-22-modul-kostenfreies-e-book?_pos=3&_sid=8bfd4bac3&_ss=r&ls=en.

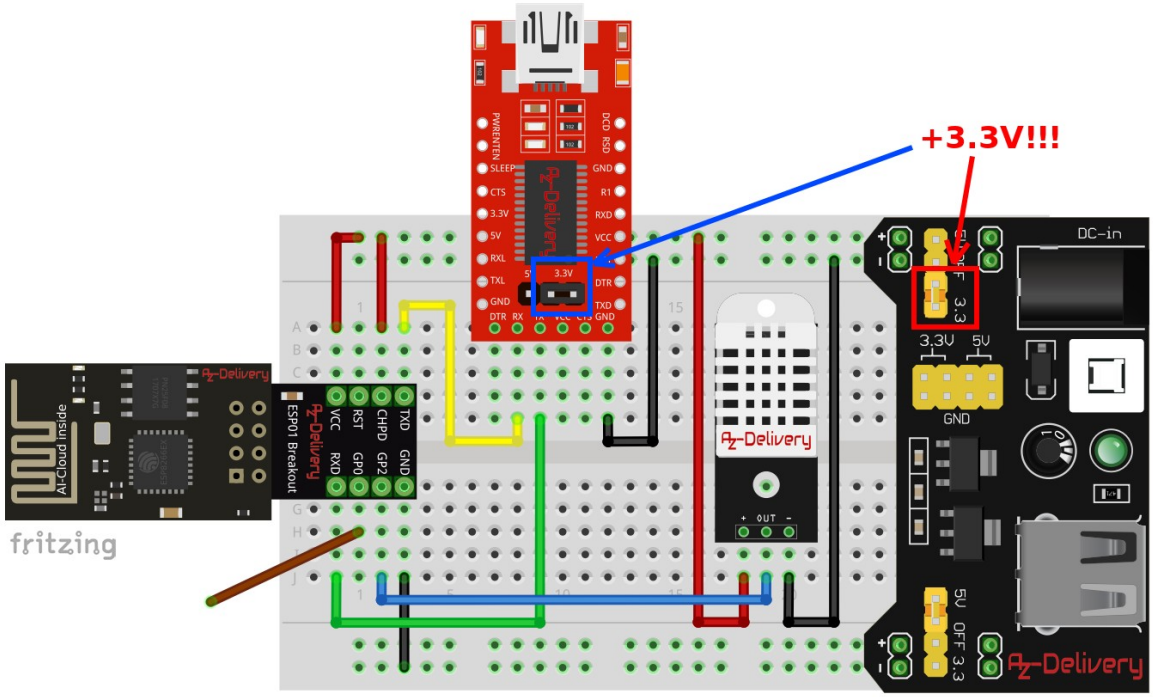
Así que no se entrará en detalles con este sensor. Todo lo que tiene que hacer es descargar la librería *"SimpleDHT"*, y agregarla al Arduino IDE si no utilizó DHT22 antes de esto. Para descargar la librería, ir al link: https://github.com/winlinvip/SimpleDHT .

Cuando descargue el archivo *".zip"*, abrir el placa de microcontrolador e ir a: *Sketch* > *Include Library* > *Add .ZIP Library*, y agregar el archivo zip descargado. Después de esto, ir a: *File* > *Examples* > *SimpleDHT* > *DHT22Default* y abrir el sketch. Se utilizará este código de este sketch en nuestro sketch.

El sensor DHT22 puede trabajar tanto en +3.3V y +5V así que se conecta el DHT22 a +3.3V debido a que ESP trabaja en +3.3V y se dañaría si se conecta el DHT22 a +5V. Cuando el DHT22 está conectado a +5V emitirá una señal de +5V en el pin OUT.

¡Así que asegúrese de conectar el sensor DHT22 a la fuente de alimentación de +3.3V!

Conecte todo como en el diagrama de conexión inferior:



Todas las conexiones son las mismas que en el último diagrama de conexión de este E-Book, excepto la parte con DHT22.

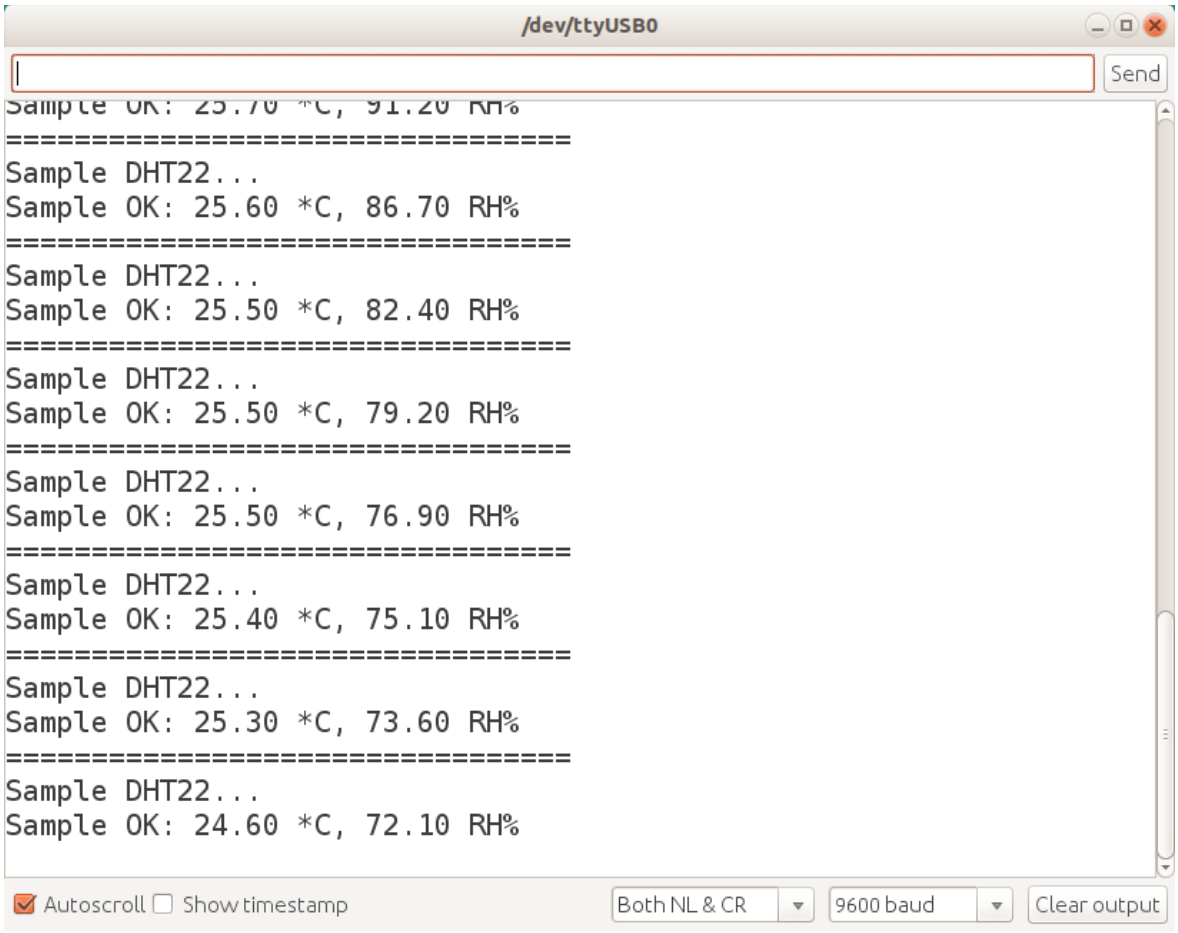
Pin DHT22	**>**	**Pin de ESP y pines fuente de alimentación**		
Pin "+"	>	+3.3V	[fuente de alimentación]	**Cable rojo**
Pin OUT	>	GPIO pin 2	[ESP]	**Cable azul**
Pin "-"	>	GND	[fuente de alimentación]	**Cable negro**

Al abrir este sketch *"DHT22Default"*, y luego elegir en *Tools* > *Board* > *{board name}*, primero *“Generic ESP8266 Module”*. Luego, cuando empiece a cargar el sketch, este será compilado para ESP8266.

Sólo se cambian algunas cosas, sólo para una mejor legibilidad, pero el sketch es el mismo. Aquí está nuestro código del sketch:

```
#include <SimpleDHT.h>
int pinDHT22 = 2;
SimpleDHT22 dht22(pinDHT22);
float temperature = 0;
float humidity = 0;
void setup() {
  pinMode(pinDHT22, INPUT);
  Serial.begin(9600);
}
void loop() {
  temperature = 0;
  humidity = 0;
  Serial.println("=================================");
  Serial.println("Sample DHT22...");
  int err = SimpleDHTErrSuccess;
    if((err = dht22.read2(&temperature, &humidity, NULL)) !=
SimpleDHTErrSuccess) {
    Serial.print("Read DHT22 failed, err=");
    Serial.println(err);
    delay(2000);
    return;
  }
  Serial.print("Sample OK: ");
  Serial.print((float)temperature); Serial.print(" *C, ");
  Serial.print((float)humidity); Serial.println(" RH%");
  delay(2500);
}
```

Y cuando inicia el Serial Monitor (*Tools* > *Serial Monitor*) la salida se debe observar algo como: (como con Atmega328P Board o Raspberry Pi)



Se utiliza una velocidad de 9600 baudios en el sketch, así que asegúrese de que en Serial Output, la velocidad de 9600 baudios esté establecida.

Lo ha logrado.

Ahora puede utilizar su módulo para sus proyectos

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

Si está buscando microelectrónica y accesorios de alta calidad , AZ-Delivery Vertriebs GmbH es la compañía adecuada donde podrá obtenerlos. Se le proporcionarán numerosos ejemplos de aplicación, guías completas de instalación, libros electrónicos, librerías y la asistencia de nuestros expertos técnicos.

https://az-delivery.de

¡Disfrute!

Impressum

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