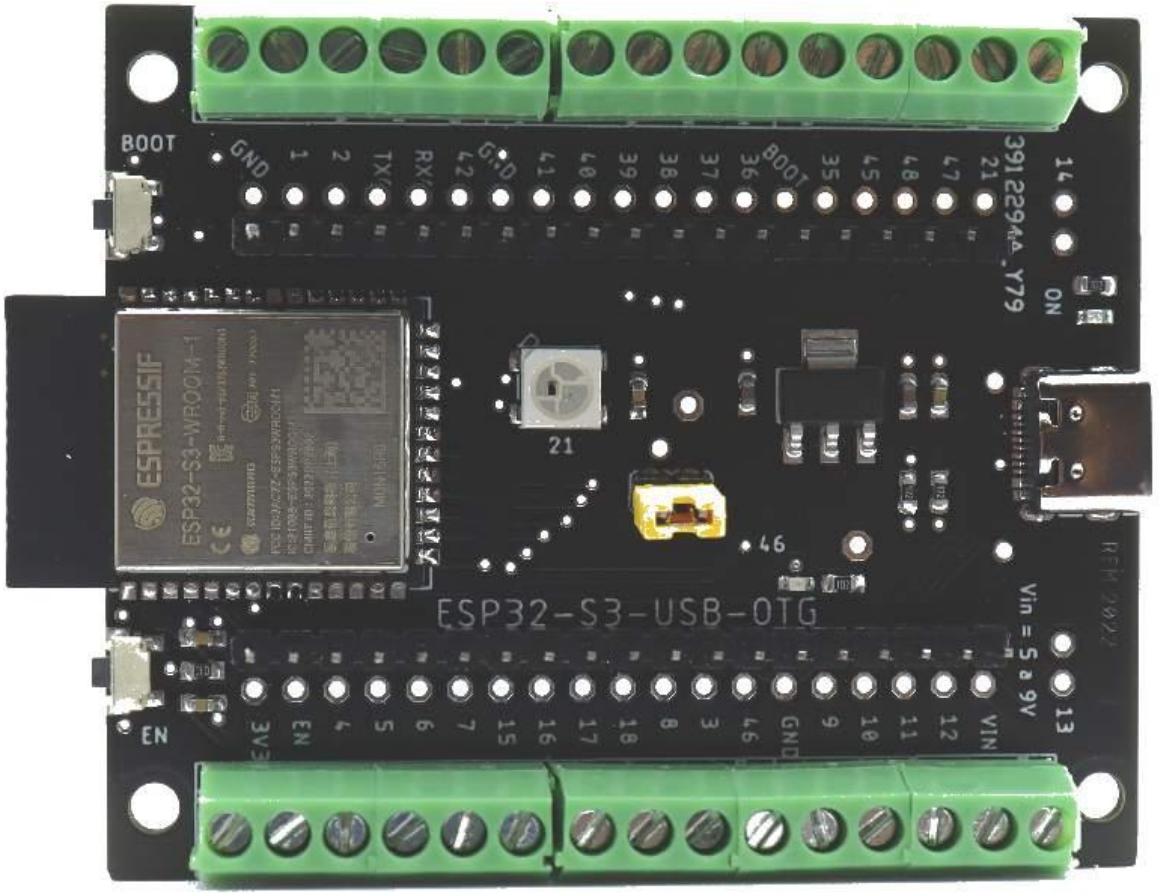


El módulo ESP32-S3-USB-OTG es una tarjeta de desarrollo, esta versión está basada en el módulo ESP32-S3-WROOM-1-N16R8, el cual es una tarjeta para el procesador ARM Xtensa® el cual es un procesador de doble núcleo de 32-bit LX7. Este cuenta con wifi 2.4 GHz 802.11 b/g/n y bluetooth® LE. La tarjeta cuenta con 36 pines de propósito general de entrada y salida (GPIO). Además cuenta con 16MB de memoria FLASH y 8MB de memoria SRAM.

Especificaciones

- ESP32-S3 SoCs, procesador Xtensa® de 32 bits.
- Perforaciones para adaptadores a riel DIN.
- 384KB ROM.
- 512KB SRAM.
- 16KB SRAM en el RTC.
- 8MB PSRAM.
- Wifi: 802.11 b/g/n hasta 150Mbs.
- Bluetooth 5 LE. 2Mpbs PHY.
- Modulo: ESP32-S3-WROOM-1-N16R8
- Dimensiones: 54mm x 65mm.

- Conexión: USB-C.
- Incluye cable: Si.
- Acceso a terminales tornillo para algunos de los pines.
- Conexiones macho y pines adicionales para soldar o conectar pines hembra (pines hembra no incluidos).
- Niveles de voltaje lógico: 3.3Vdc.
- Led RGB digital: WS2812B conectado al PIN 21.
- Led rojo conectado al pin 46.

Código de prueba
https://gist.github.com/esmarr58/8450086391930b2e667b1c6d68065db9

```
#include <Adafruit_NeoPixel.h>
#include <OneWire.h>

Adafruit_NeoPixel pixels(1, 21, NEO_GRB + NEO_KHZ800);

void setup() {
  pinMode(46, OUTPUT);
  pixels.begin();
  Serial.begin(115200);
}

void loop() {
  // put your main code here, to run repeatedly:
    digitalWrite(46, LOW);
    pixels.clear();
    pixels.setPixelColor(0, pixels.Color(0, 0, 255));
    pixels.show();
    delay(1000);
    pixels.clear();
    pixels.setPixelColor(0, pixels.Color(00, 255, 0));
    pixels.show();
    delay(1000);
    pixels.clear();
    pixels.setPixelColor(0, pixels.Color(255, 0, 0));
    pixels.show();
    delay(1000);

     // contador++;
     digitalWrite(46, HIGH);
     delay(1000);
     Serial.println("Hola");

     //delay(1000);
```

}
Programación en Arduino

Para programar la tarjeta en Arduino, solo hay que hacer referencia al número de pin impreso en la tarjeta.

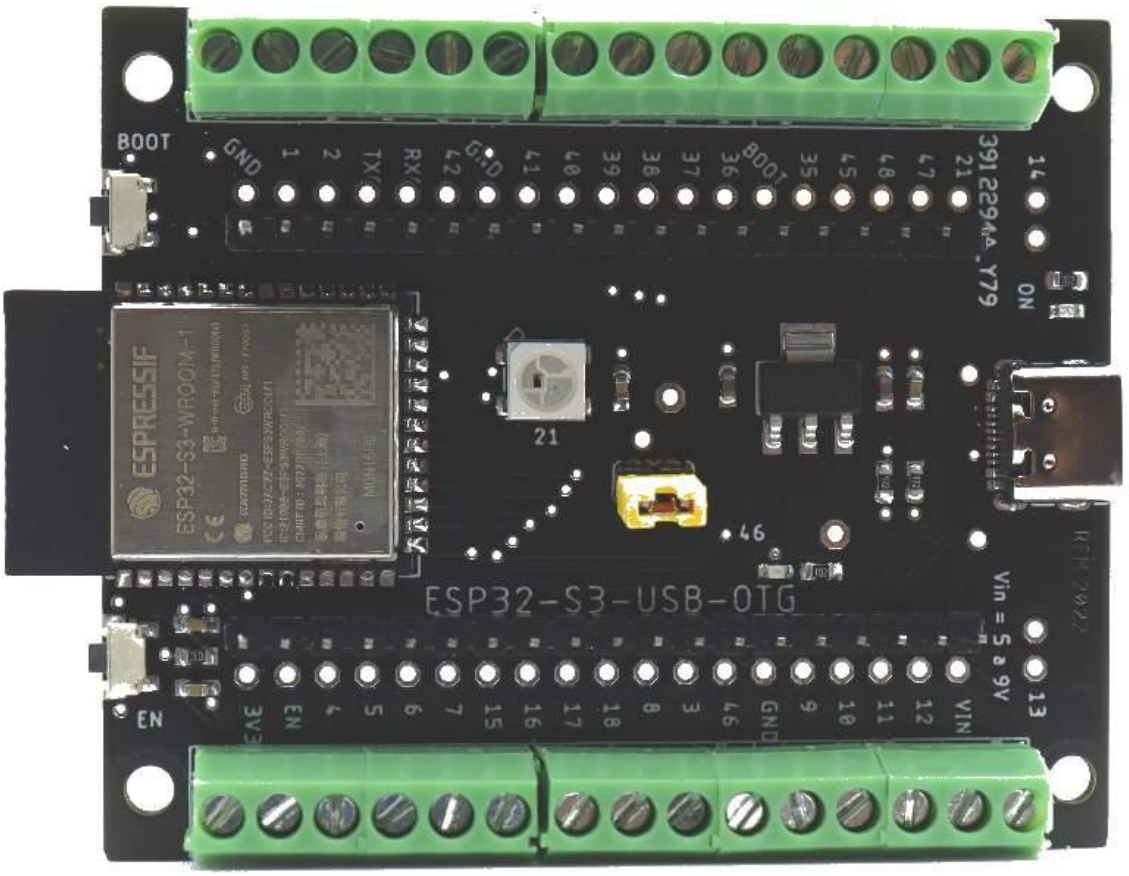


Figura 2. En la serigrafía de la cara superior vienen los números de los pines de los conectores macho.

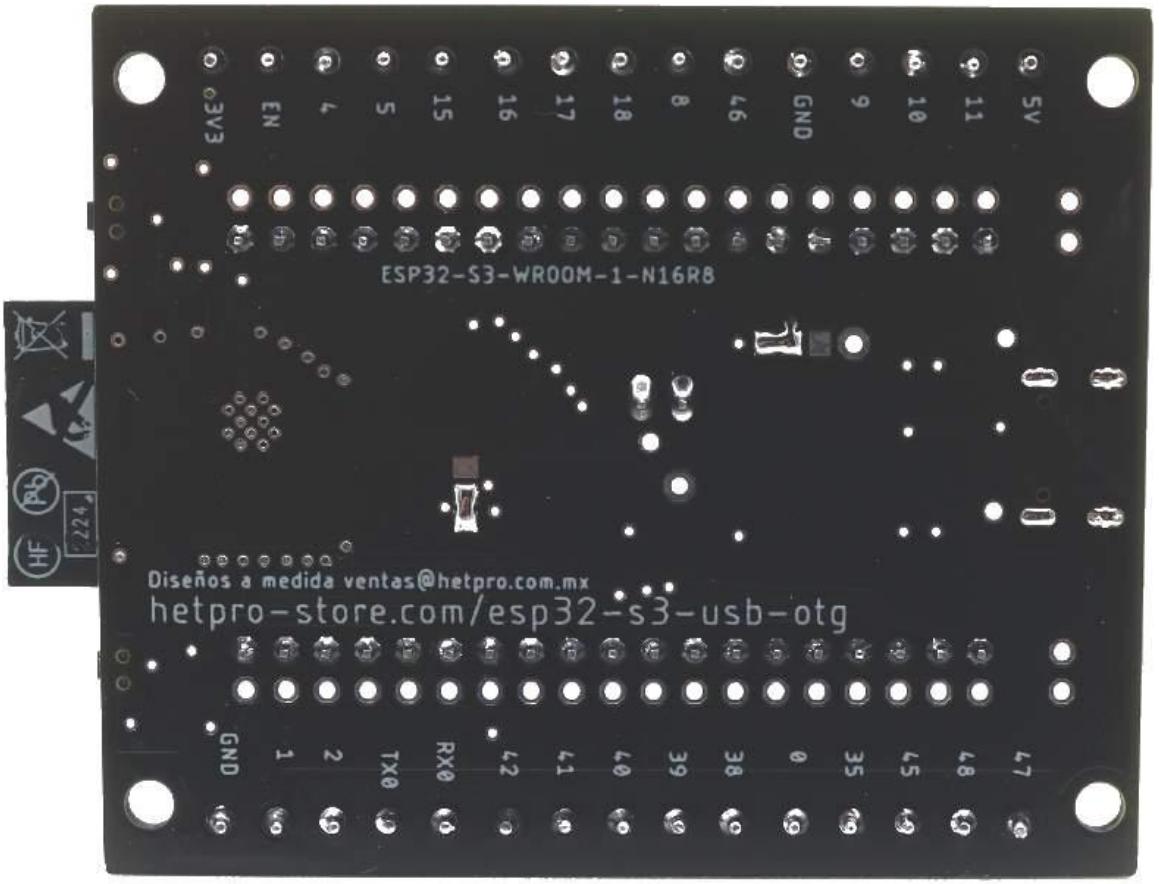


Figura 3. En la cara inferior de la tarjeta se muestran los números de pines que se tiene acceso a través de las terminales tornillo.

Descripción de pines

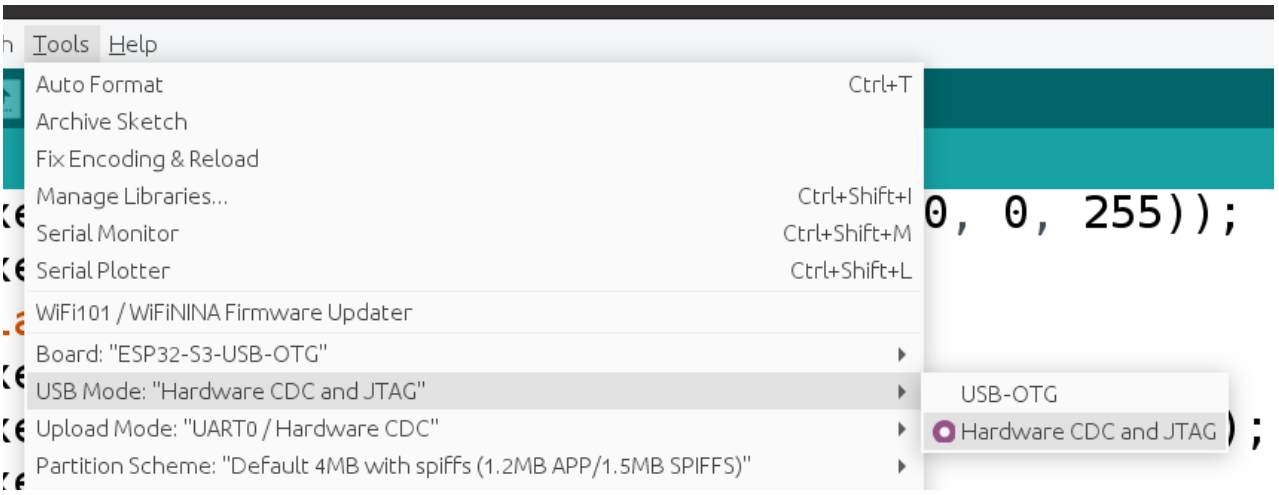
Serigrafia de tarjeta	Funcionalidad
GND	Tierra GND
3V3	Voltaje de salida del regulador de 3.3V. El cual se usa para alimentar al módulo.
EN	Habilitador del módulo, voltaje alto, activa al módulo, voltaje bajo, lo apaga.
4	IO4, Sensor táctil 4, entrada 3 del ADC1.
5	IO5, Sensor táctil 5, ADC1_CH4
6	IO6, Sensor táctil 6, ADC1_CH5
7	IO7, Sensor táctil 7, ADC1_CH6

15	IO15, U0RTS, ADC2_CH4, XTAL_32K_P
16	IO16, U0CTS, ADC2_CH5, XTAL_32K_N
17	IO17, U1TXD, ADC2_CH6
18	IO18, U1RXD, ADC2_CH7, CLK_OUT3
19	IO19, U1RTS, ADC2_CH8, CLK_OUT2, USB_D-
20	IO20, U1CTS, ADC2_CH9, CLK_OUT1, USB_D+
3	IO3, Sensor táctil 3, ADC1_CH2, RTC_GPIO3
46	IO46
9	IO9, Sensor táctil 9, ADC1_CH8, FSPIHD, SUBSPIHD, RTC_GPIO9
10	IO10, Sensor táctil 10, ADC1_CH9, FSPICS0, FSPIIO4, RTC_GPIO10
11	IO11, Sensor táctil 11, ADC2_CH0, FSPID, FSPIIO5, RTC_GPIO11, SUBSPID
12	IO12, Sensor táctil 12, ADC2_CH1, FSPICLK, FSPIIO6, SUBSPICLK
13	IO13, Sensor táctil 13, ADC2_CH2, FSPIQ, FSPIIO7, RTCGPIO13
14	IO14, Sensor táctil 14, ADC2_CH3, FSPIWP, FSPIDQS, SUBSPIWP, RTC_GPIO14
21	IO21, RTC_GPIO21
47	IO47, SPI_CLK, SUBSPI_CLK_P_DIFF
48	IO48, SPI_CLK_N_DIFF
45	IO45
0	RTC_GPIO0, botón boot
35	IO35, SPII06, FSPID, SUBSPID
36	IO36, SPIIO7, FSPICLK, SUBSPICLK
37	IO37, SPIDQS, FSPIQ
38	IO38, FSPIWP, SUBSPIWP
39	IO39, MTCK, CLK_OUT3, SUBSPICS1
40	IO40, CLK_OUT2, MTDO
41	IO41, CLK_OUT1, MTDI
42	IO42, MTMS
RXD0	Pin RX serial, este no está conectado a la comunicación USB. Niveles de

	voltaje de 3.3V, para comunicar con una computadora, se requiere un convertidor usb a serial.
TXD0	Pin de transmisión serial, independiente de la comunicación USB-OTG.
2	IO2, Sensor táctil 2, ADC1_CH1, RTC_GPIO2
1	IO1, Sensor táctil 1, ADC1_CH0, RTC_GPIO1

Comunicación con Arduino y la función Serial.print.

Para poder visualizar en el monitor serial de Arduino, la información impresa con la instrucción Serial.print o Serial.println, es necesario configurar la funcionalidad del puerto USB-OTG para ser usado para dicho propósito.Esto en el menú tools y USB Mode, tendrá que estar como Hardware CDC and JTAG. Es importante mencionar que para que el programa funcione correctamente, siempre debe de haber una comunicación con la computadora, si por ejemplo, desconectamos el cable USB-C es posible que el programa no funcione correctamente, por lo que se debería de usar la función serial print, solamente cuando siempre tengamos conexión. Si queremos lograr lo que tradicionalmente se hace con un Arduino UNO, entonces se deberá de conectar un convertidor USB-Serial a los pines RX y TX.

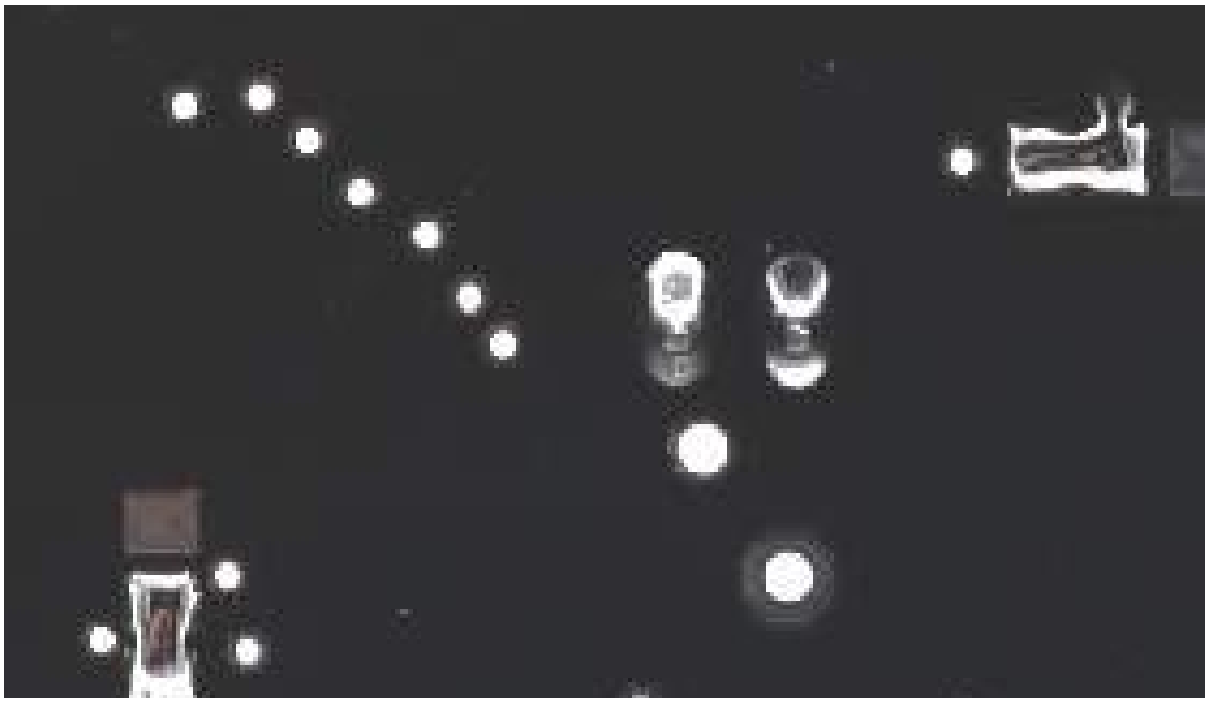


Conexión con las entradas analogicas

Para dar lectura a las entradas analógicas, es necesario solamente tomar en cuenta el pin GPIO al que se quiere leer. Hay que tomar en cuenta que únicamente los siguientes GPIO tienen acceso al módulo ADC: 4,5,6,7,8,9,10,11,12,13,14,15,17,18,19,20,21,22,38 y 39. Aunque los pines 19 y 20 se usan para la programación por el puerto USB-OTG y en algunos casos cuando se usa la comunicación WiFi o bluetooth algunos estarán deshabilitados para el propósito de entrada analógica.

Periféricos

La tarjeta cuenta con 2 periféricos, 2 leds uno digital RGB y el otro un led rojo. Ambos leds pueden ser deshabilitados, al quitar el jumper de soldadura que se encuentra en la cara inferior de la tarjeta.



Adicionalmente pueden ser puenteados a la conexión de una vía, la cual puede ser usada para soldar un cable, un pin macho o un pin hembra y poder puentear la conexión a cualquier otro pin de la tarjeta.

Vías de 0.85mm de diámetro para la conexión externa de los leds periféricos.