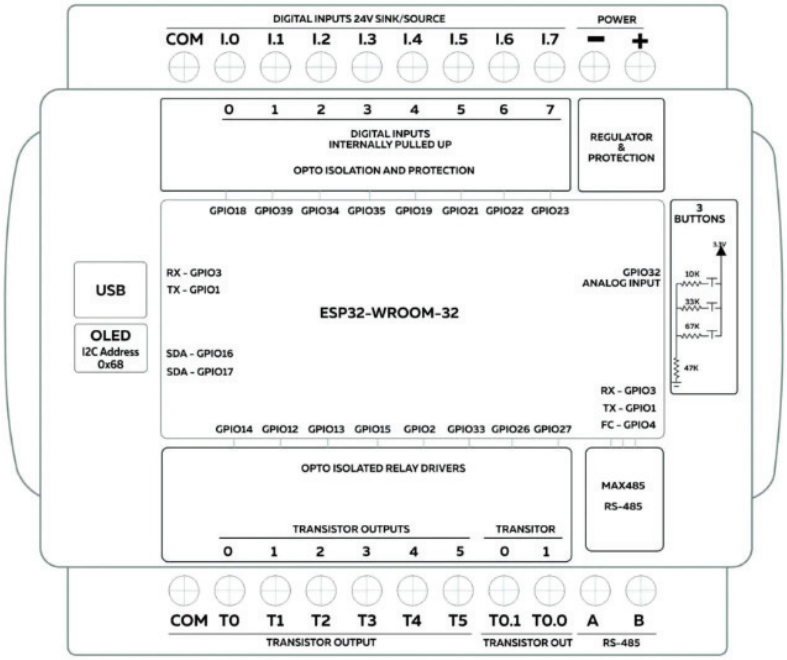
NORVI IIOT-AE01-T – Ficha de Datos

Características del Producto



- Módulo ESP32-WROOM32
- Pantalla OLED de 0,96 incorporada
- Botón incorporado en el panel frontal
- Entradas digitales
- Salidas de transistor
- Montaje en carril DIN

Ampliaciones Compatibles

- Entrada digital
- Salida de Transistor

Inicio

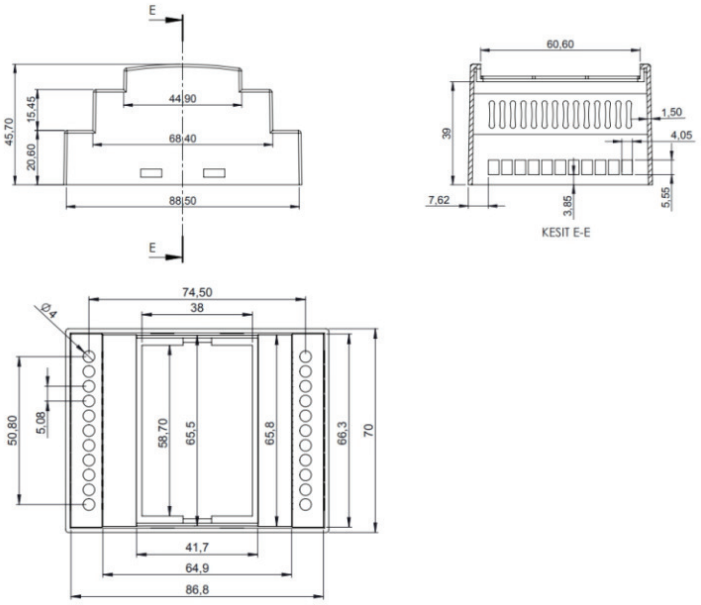
Gama de Producto	NORVI IIOT
Tipo de Producto	Controlador Programable
Certificaciones	EN 61131-2:2007 EN 61010-1:2010+A1:2019 EN IEC 61010-2-201:2018 2014/30/EU- Compatibilidad Electromagnética (EMC) Anexo III, Parte B, Modulo C
Tensión Nominal de Alimentación	24V DC
Comunicación	WiFI 2.5GHz / Bluetooth RS-485
Entradas y Salidas	8 x Entradas Digitales 8 x Salidas Transistor
Pantallas e Indicadores Visuales	0,96 Pantalla OLED e indicadores

Complementarios

Código Unificado de Producto	NORVI IIOT-AE01-T
Números de Pieza de Producto	NORVI IIOT-AE01-T

Características Mecánicas

Carcasa	NORVI 204
Montaje / Método de Instalaci· n	RIEL DIN / PESTAÑAS DE MONTAJE
Tipo de Terminal	TERMINAL DE TORNILLO
Disposición de los terminales	Arriba y Abajo
Largo	90.50 mm
Altura	56.60 mm
Ancho	60.60 mm



Entorno

Grado de Protección IP	IP20
Altitud de Funcionamiento	0–2000 metros
Temperatura de Funcionamiento	– -10… +85° C (14…185 °F)
Altitud de Almacenaje	0–3000 metros
Resistencia a Golpes	15 gn para 11ms
Resistencia a Descargas Electrostáticas	4 kV en contacto 8 kV en el aire
Resistencia a Campos Electromagnéticos	10 V/m (80 MHz …… 1GHz) 3 V/m (1.4 MHz …… 2 GHz) 1 V/m (2 MHz …… 3 GHz)

Características Eléctricas

Dispositivos Alimentados por la Red

Tensión nominal de alimentación (V)	24V DC
Consumo de Corriente (mA)	400mA
Fuente de Alimentación Recomendada	1A, 24V DC

Procesamiento

SOC / MCU	ESP32-WROOM32
Flash Memory	4MB
ROM	448 KB
SRAM	520 KB
PSRAM	NO DISPONIBLE

Periféricos

Botones Incorporados

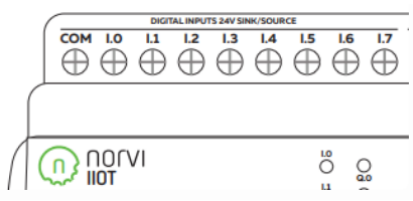
Botón 1 Pin	GPIO32 Entrada Analógica Nivel 1
Botón 2 Pin	GPIO32 Entrada Analógica Nivel 2
Botón 3 Pin	GPIO32 Entrada Analógica Nivel 3

Pantalla OLED

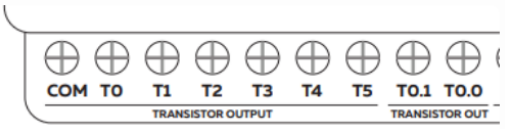
Controlador de Pantalla	SSD1306
Tamaño de Pantalla	0.96 pulgadas
SCL PiN	GPIO17
SDA Pin	GPIO16
RESET Pin	NO CONECTADO

ENTRADAS y SALIDAS

Entradas Digitales

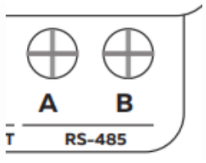
Numero de Entradas Digitales	8
Polaridad de Entradas Digitales	Ambos, Receptor de Corriente y Fuente (Sink and Source)
Tensión Máxima de Entrada Digital	32V DC
Tensión Mínima de Entrada Digital	18V DC
Frecuencia Máxima de Conmutación	1 khZ
Disposición de los Terminales	Entrada Digital 0 – GPIO18 Entrada Digital 1 – GPIO39 Entrada Digital 2 – GPIO34 Entrada Digital 3 – GPIO35 Entrada Digital 4 – GPIO19 Entrada Digital 5 – GPIO21 Entrada Digital 6 – GPIO22 Entrada Digital 7 – GPIO23 

Salidas Transistor

Numero de Salidas Transistor	8
Tipo de Salida Transistor	COLECTOR ABIERTO
Corriente Máxima Sink/Source (mA)	100mA
Voltaje Máximo Aplicable	36V DC
Frecuencia Máxima de Conmutación	1 kHz
Disposición de los Terminales	T0.1 – GPIO26 T0.0 – GPI027 T0 – GPI014 T1 – GPI012 T2 – GPI013 T3 – GPI015 T4 – GPI02 T5 – GPI033 

Canales de Comunicación

Comunicación RS-485

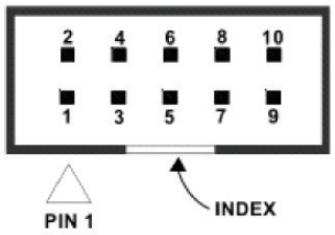
Modo de Comunicación	HALF-DUPLEX
Transceptor	MAX485
Unidad de Carga	1/4
Control de flujo / Pin de Control de Dirección	GPIO4
Pin TX	GPIO1
Pin RX	GPIO3
Disposición de los Terminales	

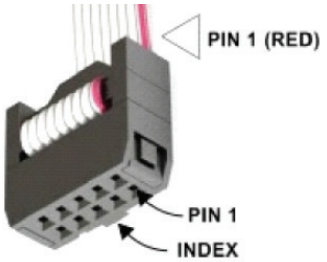
Mapa GPIO

GPIO	Descripcion	Uso
0	Emite señal PWM al arrancar	NRST
1	RS-485	TX
2		Salida de Transistor 4
3	RS-485	RX
4	RS-485	Flow Control
5		
12		Salida de Transistor 1
13		Salida de Transistor 2

14		Salida de Transistor 0
15		Salida de Transistor 3
16	SDA	I2C
17	SCL	I2C
18	sólo entrada	Entrada Digital 0
19	sólo entrada	Entrada Digital 4
20		
21	sólo entrada	Entrada Digital 5
22	sólo entrada	Entrada Digital 6
23	sólo entrada	Entrada Digital 7
24		
25	Puerto de Expansión	Pin 1
26		Salida de Transistor 0.0
27		Salida de Transistor 0.1
28		
32	Entrada Análoga	Botones
33		Salida de Transistor 5
34	sólo entrada	Entrada Digital 2
35	sólo entrada	Entrada Digital 3
38		
39	sólo entrada	Entrada Digital 1

Puerto de Expansión





PIN	ESP32 Conexión
1	IO25
2	TXD0
3	SIN CONEXION
4	RXD0
5	BOOT IO0
6	IO32
7	3.3V
8	SCL – GPIO17
9	TIERRA
10	SDA – GPIO16

NORVI IIOT-AE01-T – Programa de Pruebas

Introducción

- Esta guía está destinada a probar las características y el funcionamiento básico del dispositivo, el NORVI IIOT-AE01-T (modelo de transistor).
- Programa de pruebas NORVI IIOT-AE01-T.

Tabla de Instrucciones de Ensayo

Grabe el firmware de prueba antes de probar el dispositivo. Siga las instrucciones dadas en la guía "Guía para Grabar el Firmware de Prueba" para grabar el código binario.

Pruebas de componentes / características	**Test - Prueba**	**Resultados Previstos**
Alimentación	Proporcione una alimentación de 24 V CC.	El LED rojo dentro del dispositivo se ilumina. La pantalla se enciende.
Pantalla	Encienda el dispositivo usando un cable USB o una fuente de alimentación de 24V CC.	La pantalla comienza con el logotipo de Norvi. Se muestra el modelo del dispositivo. Aparece una última pantalla con el estado de las Entradas, Salidas y Pulsadores. Los indicadores LED del lado de salida se iluminan siguiendo un patrón.
Entradas Digitales	Encienda el dispositivo con una fuente de alimentación de 24 V CC. Conecte los pines GND y COM del lado de la entrada digital y suministre 24 V CC a cada entrada digital una a una.	Consulte los resultados esperados de la comprobación de pantalla anterior. En el estado de entrada, el estado de las ocho entradas digitales será 1.(Ya que están internamente pull up) El estado de la entrada cambia de 1 a 0, y el indicador LED del lado de entrada comienza a iluminarse en consecuencia.
Salidas Transistor	Encienda el dispositivo con una fuente de alimentación de 24 V CC.	La conmutación del estado de salida (de 0 a 1) se observa en la pantalla para las 8 salidas de transistor, que sigue el patrón de parpadeo del indicador LED del lado de salida. Siempre que estos LEDs estén encendidos, significa que el transistor respectivo está encendido.
Salidas de Transistor (continuación)	Para comprobar el funcionamiento de los transistores, se realiza una prueba de tensión con un multímetro. Para ello, mantén la sonda positiva del multímetro en la patilla +24V del dispositivo. A continuación, toca la sonda negativa con las dos patillas del transistor, una a una, tras un intervalo de 15 segundos.	El multímetro muestra una lectura de 24 V CC siempre que el transistor esté encendido (el estado del transistor se indica mediante el indicador LED del lado de salida respectivo y el estado de salida en la pantalla).
Pulsadores	Pulse los tres pulsadores, de uno en uno.	El estado analógico de 4 dígitos del pulsador se visualiza correspondientemente en la pantalla. Estado analógico 1_ _ _ para el botón superior Estado analógico 2_ _ _ para el botón central Estado analógico 3_ _ _ para el botón inferior
Comunicación RS-485	Para esta prueba se necesita un conversor USB a RS-485. Conecte los pines A y B RS-485 del dispositivo Norvi con los respectivos pines A y B del convertidor USB a RS-485.	En el monitor serie se imprime la indicación «RS485 SUCCESS». Esto indica que la operación de envío RS-485 funciona correctamente en el dispositivo Norvi.

Comunicación RS-485 (continuación)	Conecte el extremo USB del conversor USB a RS-485 al PC. Encienda el dispositivo Norvi con un cable USB. Abre la aplicación Arduino IDE. Selecciona el puerto COM correcto del conversor USB a RS-485 en el IDE de Arduino y abre el Monitor serie. Envía el número '5' en el monitor serie.	Una vez recibido el número «5», todos los indicadores LED del lado de salida se iluminarán simultáneamente durante unos segundos. Luego, más tarde, continuarán brillando en su patrón anterior. Esto indica que la operación de recepción RS-485 funciona correctamente en el dispositivo Norvi.

Registros de Pruebas

Programa de Pruebas

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include "FS.h"

#define ANALOG_PIN_0 32

#define INPUT1 18
#define INPUT2 39
#define INPUT3 34
#define INPUT4 35
#define INPUT5 19
#define INPUT6 21
#define INPUT7 22
#define INPUT8 23

#define OUTPUT1 26
#define OUTPUT2 27
#define OUTPUT3 14
#define OUTPUT4 12
#define OUTPUT5 13
#define OUTPUT6 15
#define OUTPUT7 2
#define OUTPUT8 33

#define RS485_RXD 3
#define RS485_TXD 5
#define RS485_FC 4

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

#define OLED_RESET     -1 // Reset pin # (or -1 if sharing Arduino reset pin)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

int analog_value = 0;
int readSwitch(){
  analog_value = analogRead(ANALOG_PIN_0);

  return analog_value;
}
```

```
void setup() {

  Serial.begin(9600);
  Serial.println("Hello");

 pinMode(RS485_FC, OUTPUT);
 digitalWrite(RS485_FC, HIGH);

 Serial1.begin(9600, SERIAL_8N1,RS485_RXD,RS485_TXD);
  digitalWrite(RS485_FC, HIGH);   // RS-485

  pinMode(OUTPUT1, OUTPUT);
  pinMode(OUTPUT2, OUTPUT);
  pinMode(OUTPUT3, OUTPUT);
  pinMode(OUTPUT4, OUTPUT);
  pinMode(OUTPUT5, OUTPUT);
  pinMode(OUTPUT6, OUTPUT);
  pinMode(OUTPUT7, OUTPUT);
  pinMode(OUTPUT8, OUTPUT);

  pinMode(INPUT1, INPUT);
  pinMode(INPUT2, INPUT);
  pinMode(INPUT3, INPUT);
  pinMode(INPUT4, INPUT);
  pinMode(INPUT5, INPUT);
  pinMode(INPUT6, INPUT);
  pinMode(INPUT7, INPUT);
  pinMode(INPUT8, INPUT);

  Wire.begin(16,17);

  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // Address 0x3D for 128x64
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }
  display.display();
  adcAttachPin(32);
}
void loop() {
 Serial.println("");
      Serial.print(digitalRead(INPUT1));
      Serial.print(digitalRead(INPUT2));
      Serial.print(digitalRead(INPUT3));
      Serial.println(digitalRead(INPUT4));
      Serial.print(digitalRead(INPUT5));
      Serial.print(digitalRead(INPUT6));
      Serial.print(digitalRead(INPUT7));
      Serial.println(digitalRead(INPUT8));
      Serial.println("");

  Serial.print("Push button  ");Serial.println(readSwitch());
  Serial.println("");
  digitalWrite(OUTPUT1, HIGH);
  digitalWrite(OUTPUT2, LOW);
  digitalWrite(OUTPUT3, LOW);
  digitalWrite(OUTPUT4, LOW);
  digitalWrite(OUTPUT5, LOW);
```

```
  digitalWrite(OUTPUT6, LOW);
  digitalWrite(OUTPUT7, LOW);
  digitalWrite(OUTPUT8, LOW);
  delay(200);
  digitalWrite(OUTPUT1, LOW);
  digitalWrite(OUTPUT2, HIGH);
  digitalWrite(OUTPUT3, LOW);
  digitalWrite(OUTPUT4, LOW);
  digitalWrite(OUTPUT5, LOW);
  digitalWrite(OUTPUT6, LOW);
  digitalWrite(OUTPUT7, LOW);
  digitalWrite(OUTPUT8, LOW);
  delay(200);
  digitalWrite(OUTPUT1, LOW);
  digitalWrite(OUTPUT2, LOW);
  digitalWrite(OUTPUT3, HIGH);
  digitalWrite(OUTPUT4, LOW);
  digitalWrite(OUTPUT5, LOW);
  digitalWrite(OUTPUT6, LOW);
  digitalWrite(OUTPUT7, LOW);
  digitalWrite(OUTPUT8, LOW);
  delay(200);
  digitalWrite(OUTPUT1, LOW);
  digitalWrite(OUTPUT2, LOW);
  digitalWrite(OUTPUT3, LOW);
  digitalWrite(OUTPUT4, HIGH);
  digitalWrite(OUTPUT5, LOW);
  digitalWrite(OUTPUT6, LOW);
  digitalWrite(OUTPUT7, LOW);
  digitalWrite(OUTPUT8, LOW);
  delay(200);
  digitalWrite(OUTPUT1, LOW);
  digitalWrite(OUTPUT2, LOW);
  digitalWrite(OUTPUT3, LOW);
  digitalWrite(OUTPUT4, LOW);
  digitalWrite(OUTPUT5, HIGH);
  digitalWrite(OUTPUT6, LOW);
  digitalWrite(OUTPUT7, LOW);
  digitalWrite(OUTPUT8, LOW);
  delay(200);
  digitalWrite(OUTPUT1, LOW);
  digitalWrite(OUTPUT2, LOW);
  digitalWrite(OUTPUT3, LOW);
  digitalWrite(OUTPUT4, LOW);
  digitalWrite(OUTPUT5, LOW);
  digitalWrite(OUTPUT6, HIGH);
  digitalWrite(OUTPUT7, LOW);
  digitalWrite(OUTPUT8, LOW);
  delay(200);
  digitalWrite(OUTPUT1, LOW);
  digitalWrite(OUTPUT2, LOW);
  digitalWrite(OUTPUT3, LOW);
  digitalWrite(OUTPUT4, LOW);
  digitalWrite(OUTPUT5, LOW);
  digitalWrite(OUTPUT6, LOW);
  digitalWrite(OUTPUT7, HIGH);
  digitalWrite(OUTPUT8, LOW);
  delay(200);
```

```
  digitalWrite(OUTPUT1, LOW);
  digitalWrite(OUTPUT2, LOW);
  digitalWrite(OUTPUT3, LOW);
  digitalWrite(OUTPUT4, LOW);
  digitalWrite(OUTPUT5, LOW);
  digitalWrite(OUTPUT6, LOW);
  digitalWrite(OUTPUT7, LOW);
  digitalWrite(OUTPUT8, HIGH);
  delay(200);

digitalWrite(RS485_FC, HIGH);                    // Make FLOW CONTROL pin HIGH
  delay(500);
  Serial1.println(F("RS485 01 SUCCESS"));    // Send RS485 SUCCESS serially
  delay(500);                                // Wait for transmission of data
  digitalWrite(RS485_FC, LOW) ;                    // Receiving mode ON
  delay(1000);

  while (Serial1.available()) {  // Check if data is available
    char c = Serial1.read();     // Read data from RS485
    Serial.write(c);             // Print data on serial monitor
  }
}
```

NORVI IIOT-AE01-T – Guía del Usuario

Programación

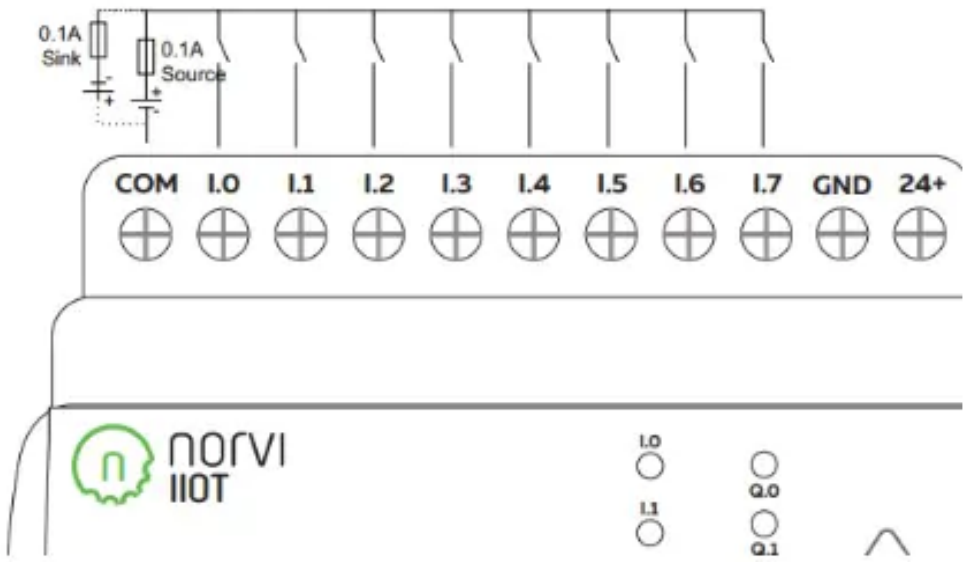
El NORVI IIOT-AE01-T tiene un puerto mini USB para la conexión en serie con el SoC para la programación. Se puede utilizar cualquier IDE de programación compatible con ESP32 para programar el controlador. Siga esta guía para programar los controladores NORVI ESP32 con el IDE Arduino.

SoC: ESP32-WROOM32

Programming Port: USB UART

Entradas Digitales

Cableado de Entradas Digitales



NORVI IIOT-AE01-T Cableado de la entrada digital

Programación de las Entradas Digitales

La lectura del GPIO correspondiente del ESP32 da el valor de la Entrada Digital. Cuando las entradas están en estado OFF, el GPIO pasa a HIGH, y cuando la entrada está en estado ON, el GPIO pasa a LOW.

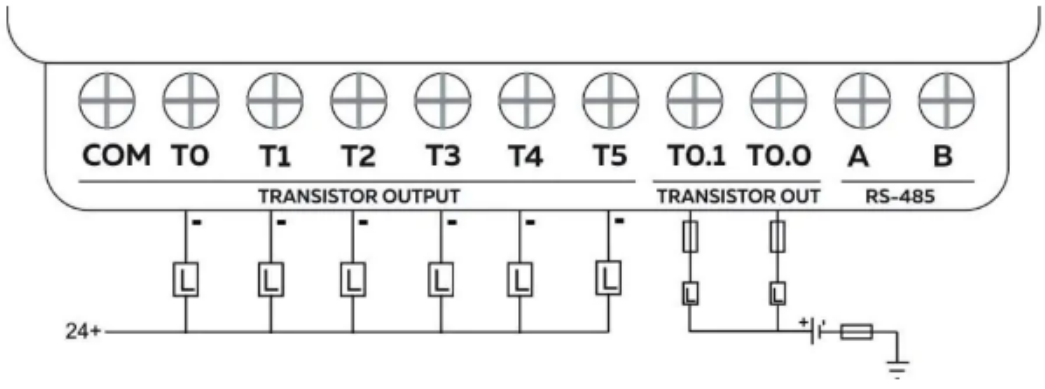
Consulte la tabla de asignación GPIO en la hoja de datos para la entrada digital GPIO.

```
#define INPUT1 18
void setup() {
   Serial.begin(115200);
   Serial.println("Device Starting");
   pinMode(INPUT1, INPUT);
}

void loop() {
   Serial.print(digitalRead(INPUT1));
   Serial.println("");
   delay(500);
}
```

Salida Transistor

Cableado de Salida de Transistor



NORVI-IIOT-AE01-T Cableado de Salida del Transistor

Programación de Salidas de Transistor

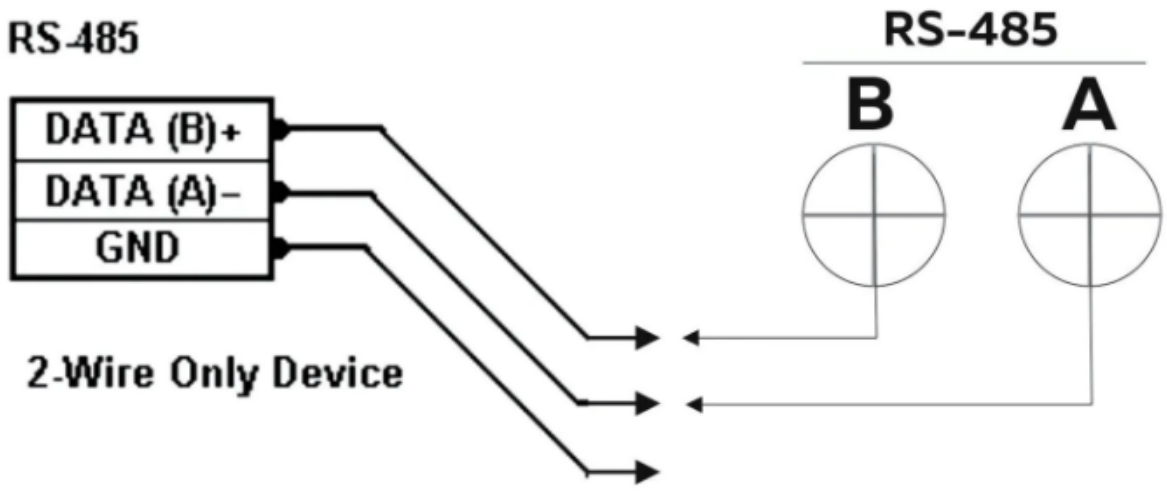
La lectura del GPIO correspondiente del ESP32 proporciona el valor de la salida del relé o transistor. Consulte la tabla de asignación de GPIO en la hoja de datos para el GPIO de salida de relé / transistor.

```
#define OUTPUT1 14

void setup() {
   Serial.begin(115200);
   Serial.println("Device Starting");
   pinMode(OUTPUT1 , OUTPUT);
}
void loop() {
   digitalWrite(OUTPUT1, HIGH);
   delay(500);
   digitalWrite(OUTPUT1, LOW);
   delay(500);
   Serial.println("");
   delay(500);
}
```

Comunicación RS-485

Cableado RS-485



Driver	MAX485
UART RX	GPIO3
UART TX	GPIO1
Control de Flujo	GPIO4

Conexiones GPIO de RS-485

Programación RS-485

Las conexiones UART del RS-485 de la serie NORVI IIOT AE01 son compartidas con las conexiones UART del USB Serial.

Las gamas de productos NORVI-IIOT-AE01 y NORVI-IIOT-AE02 utilizan comunicación RS485 en la que los pines TX (Transmisión) y RX (Recepción) se comparten con los pines RX y TX del USB. Esto significa que se utilizan los mismos pines físicos tanto para la comunicación serie con un ordenador a través de USB como para la comunicación RS485.
Como resultado, el uso directo de las funciones de impresión en serie para la comunicación no es posible cuando se conecta a tales dispositivos. Una solución alternativa sería enviar el resultado a una pantalla OLED, lo que proporcionaría información visual sobre los datos.

```
#include <Wire.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_GFX.h>

#define FC 4
#define RXD 3
#define TXD 1

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET    -1
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

void setup() {
  Serial.begin(9600);
  pinMode(FC, OUTPUT);
  digitalWrite(FC, LOW);

  Wire.begin(16, 17);
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.display();
}

void loop() {
  Serial1.println(F("RS485 01 SUCCESS"));
  while (Serial.available()) {  // Check if data is available
    char c = Serial.read();     // Read data from RS485
    display.clearDisplay();
    display.setTextColor(WHITE);
    display.setTextSize(1);
    display.setCursor(0, 12);
    display.print(c);
    delay(100);
    display.display();
  }
}
```

Pantalla OLED Integrada

Controlador de Pantalla	SSD1306
Comunicación	I2C
Dirección del módulo	0x3C
Resolución	128 x 64

0,96 Especificaciones de la pantalla OLED

Consulte la tabla de asignación de GPIO en la hoja de datos para el GPIO I2C de la pantalla OLED.
Librería soportada por la Librería Adafruit_SSD0306.
Wire.begin (SDA, SCL) es necesario para inicializar I2C en los pines correctos

Programación Pantalla OLED

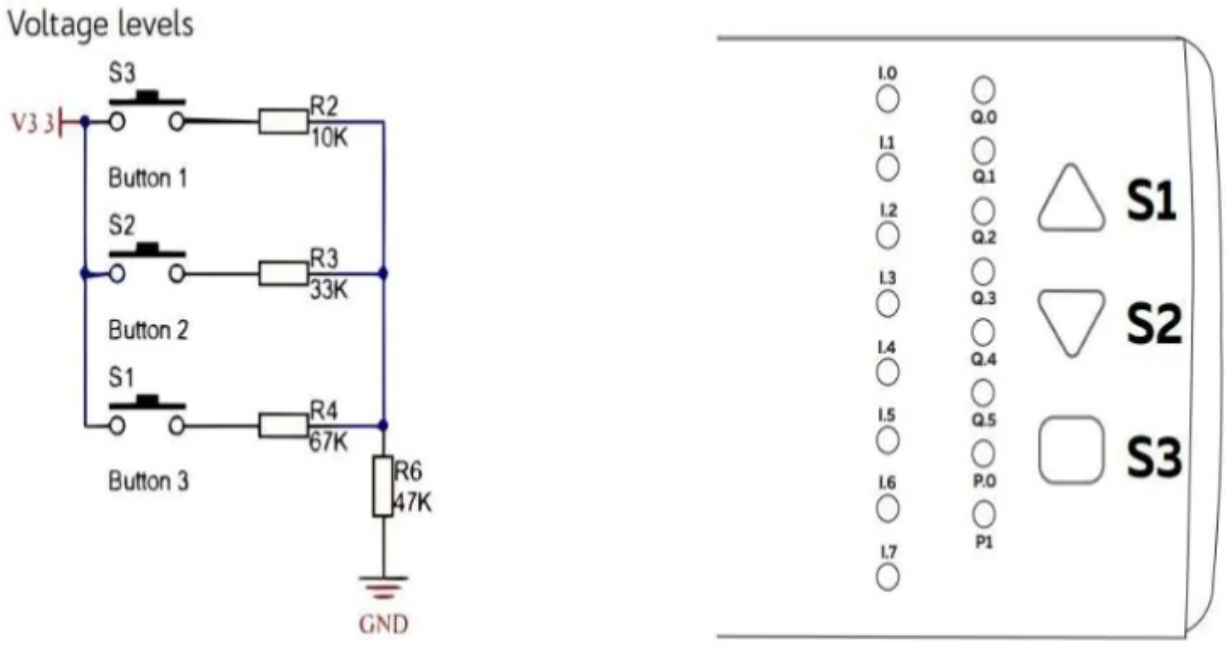
```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
#define OLED_RESET -1 // Reset pin # (or -1 if sharing Arduino reset pin) Adafruit_SSD1306 display(SCREEN_WIDTH,
SCREEN_HEIGHT, &Wire, OLED_RESET);

void setup() {
   Wire.begin(16,17);
   if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
   // Address 0x3C for 128x64 Serial.println(F("SSD1306 allocation failed"));
      for(;;); // Don't proceed, loop forever
   }
   // Show initial display buffer contents on the screen –
   // the library initializes this with an Adafruit splash screen.
   display.display(); delay(2000); // Pause for 2 seconds
}

void loop() {
}
```

Botones Incorporados

Modo de Lectura	ADC (Conversión Analógica a Digital)
Analógico IO (In/Out)	GPIO32



Botón incorporado Esquema Interno

Botones de Programación

```
#define buttonPin 32
int  buttonState = 0;

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

}
void loop() {
  Serial.print("Button: ");
  buttonState = analogRead(buttonPin);
  delay(50);
  Serial.print(analogRead(buttonPin));
  Serial.print("\tAnalog: ");
  delay(1000);
}
```

USB y Reinicio