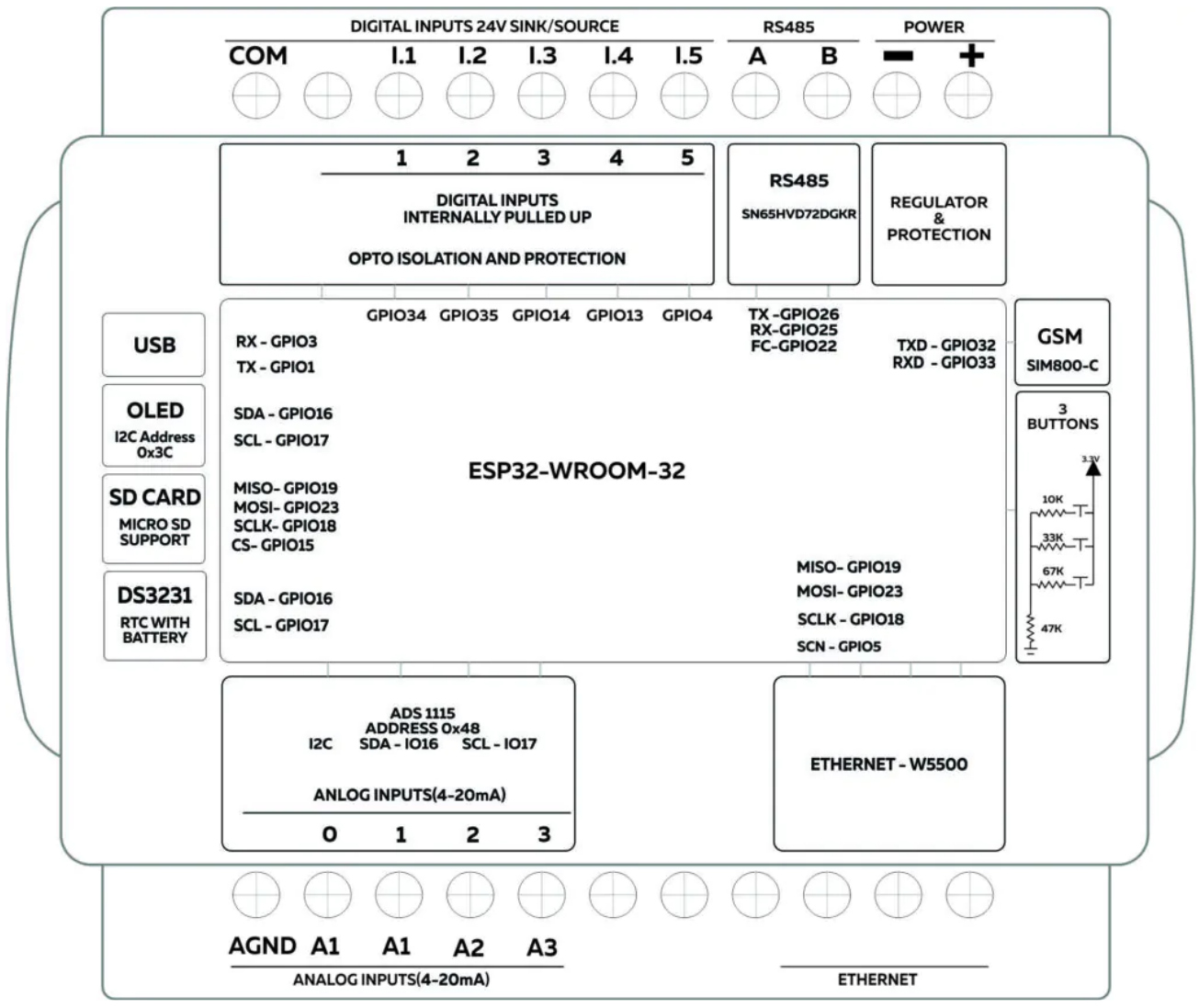
# NORVI GSM-AE08-I-G – Ficha Técnica

## Características del Producto




- Módulo ESP32-WROOM32
- Conexión GSM
- Pantalla OLED integrada de 0,96
- Soporte de tarjeta microSD
- DS3231 RTC con batería de reserva
- Ethernet a través de W5500
- Botón incorporado en el panel frontal
- Entradas digitales
- Entradas analógicas
- Montaje en carril DIN

### Comunicación Celular

- Modulo: SIM800-C
- Marca: SIMCom FCC
- ID: UDU-SIM800C TAC:
- 86610402

### Expansiones compatibles

- Entrada Analógica
- Entrada Digital

# Inicio

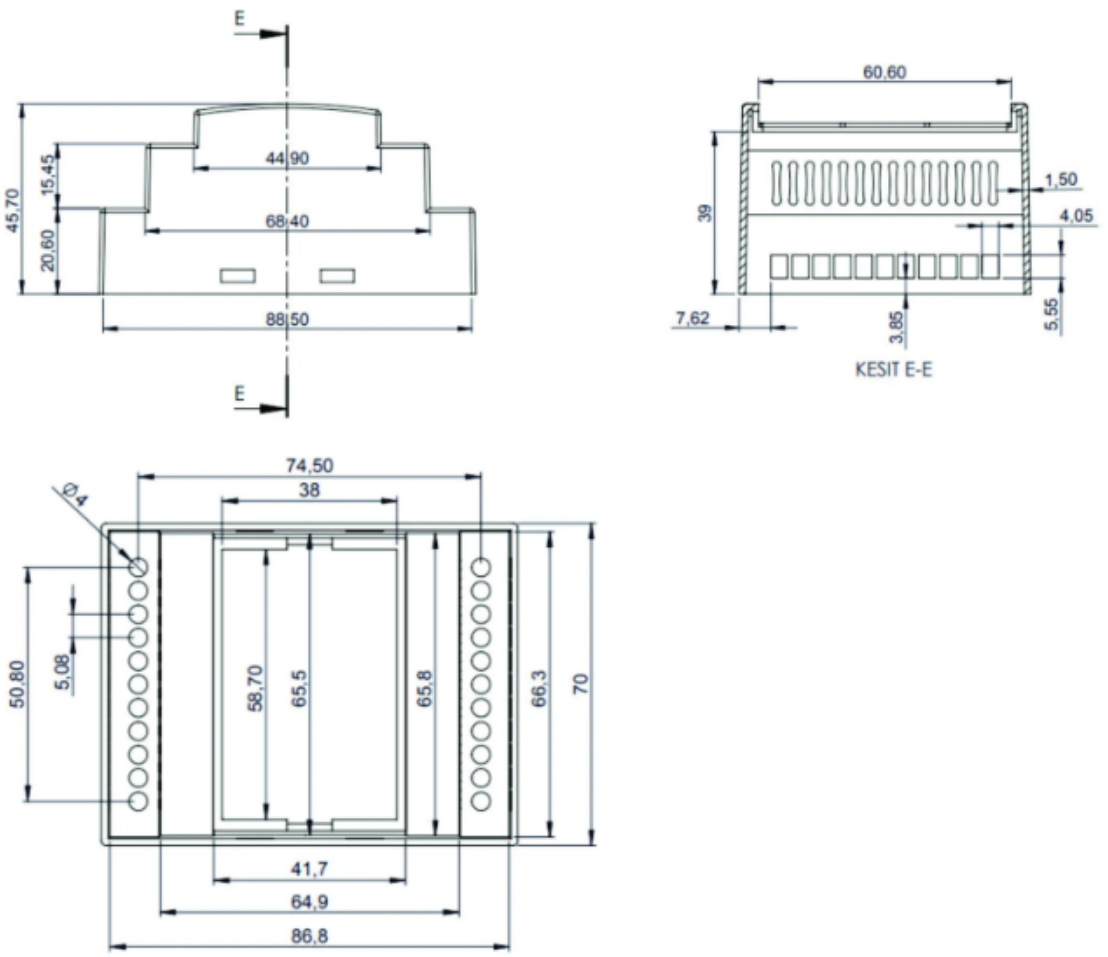
| | |
|---|---|
| Gama del Producto | NORVI GSM |
| Tipo de Producto | Controlador Programable |
| Certificaciones | EN 61131-2:2007<br>EN 61010-1:2010+A1:2019<br>EN IEC 61010-2-201:2018<br>2014/30/EU- Compatibilidad Electromagnética (EMC) Anexo III, Parte B, Modulo C |
| Alimentacion | 24V DC |
| Comunicación | WiFI / Bluetooth<br>GSM / GPRS – SIMCOM SIM800C<br>RS485 |
| Entradas y Salidas | 5 x Entradas Digitales<br>4 x Entradas Analógicas con 4-20mA |
| Pantalla e Indicadores Visuales | 0.96 Pantalla OLED e Indicadores |

# Complementarios

| | |
|---|---|
| Producto Código Unificado | NORVI GSM-AE08-I-G |
| Números de Pieza de Producto | NORVI GSM-AE08-I-G |

# Características Mecánicas

| | |
|---|---|
| Carcasa | NORVI 204 |
| Montaje / Método de Instalación | RAIL DIN / MOUNTING TABS |
| Tipo de Terminal | TERMINAL DE TORNILLOS |
| Disposición de los Terminales | Arriba y Abajo |
| Largo | 90.50 mm |
| Alto | 56.60 mm |
| Ancho | 60.60 mm |



# Entorno

| | |
|---|---|
| Grado de Protección IP | IP20 |
| Altitud de Operación | 0–2000 metros |
| Temperatura de Operación | – -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 aire |
| Resistencia a Campos Electromagnéticos | 10 V/m (80 MHz …… 1GHz)<br>3 V/m (1.4 MHz …… 2 GHz)<br>1 V/m (2 MHz …… 3 GHz) |

# Características Eléctricas

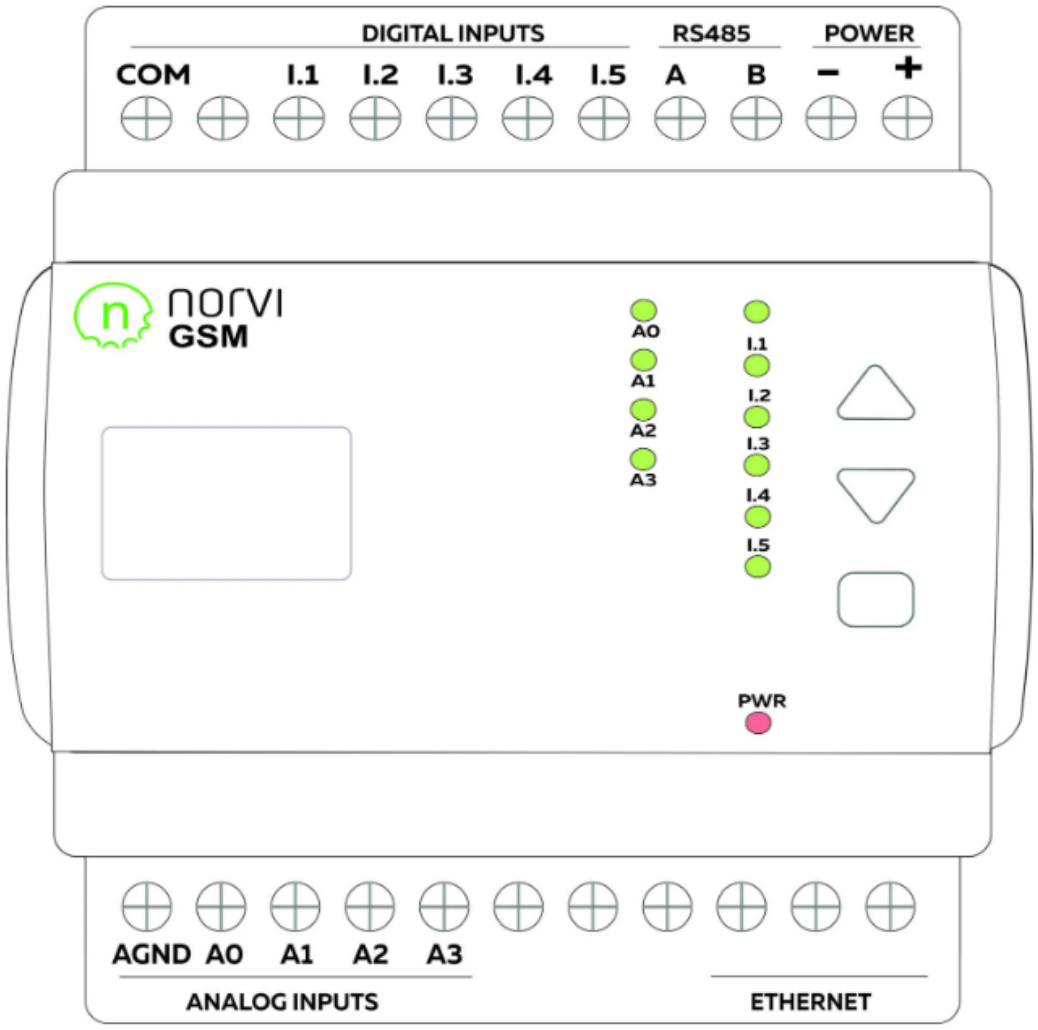
### *Dispositivos Alimentados por Red*

| | |
|---|---|
| Tensión de Alimentación (V) | 24V DC |
| Consumo 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 |

# Disposición de los Indicadores



# Periféricos

## Compatible con tarjeta micro SD

| | |
|---|---|
| Tipo de Tarjeta | microSD |
| Interfaz | SPI |
| SD CARD CS | GPIO15 |
| MISO | GPIO19 |
| MOSI | GPIO23 |
| SCLK | GPIO18 |
| SD Detect | NO CONECTADO |

## RTC Interno

| | |
|---|---|
| Chip RTC | DS3231 |
| Tipo de Batería de Respaldo | CR2032 |
| Interfaz | I2C |
| Dirección I2C | 0x68 |
| Pin SCL | GPIO17 |
| Pin SDA | GPIO18 |

## Botones Integrados

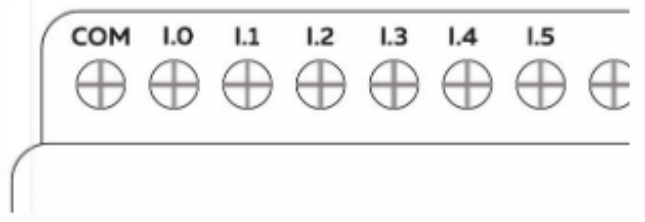
| | |
|---|---|
| Botón 1 Pin | GPIO36 Nivel de entrada analógica 1 |
| Botón 2 Pin | GPIO36 Nivel de entrada analógica 2 |
| Botón 3 Pin | GPIO36 Nivel de entrada analógica 3 |

## Pantalla OLED

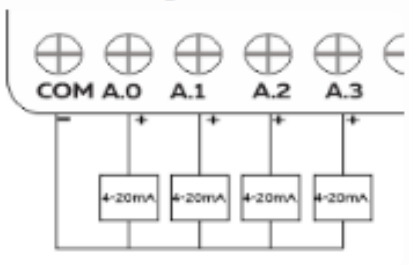
| | |
|---|---|
| Controlador de Pantalla | SSD1306 |
| Tamaño de Pantalla | 0.96 pulgadas |
| Pin SCL | GPIO17 |
| Pin SDA | GPIO16 |
| RESET Pin | NO CONECTADO |

# ENTRADAS y SALIDAS

## Entradas Digitales

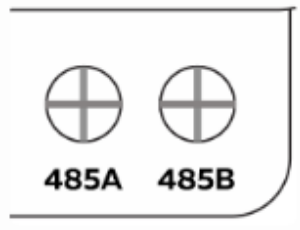
| | |
|---|---|
| Numero de Entradas Digitales | 5 |
| Polaridad de Entradas Digitales | Sink and Source |
| Tensión Máxima de Entrada Digital | 32V DC |
| Tensión Mínima de Entrada Digital | 18V DC |
| Frecuencia Máxima de Conmutación | 1 kHZ |
| Disposición de Terminales | Entrada Digital 1 – GPIO34<br>Entrada Digital 2 – GPIO35<br>Entrada Digital 3 – GPIO14<br>Entrada Digital 4 – GPIO13<br>Entrada Digital 5 – GPIO4<br>  |

## Entradas Analógicas

| | |
|---|---|
| Numero de Entradas Analógicas | 4 |
| Rango de Medición de la Entrada Analógica | 4-20mA |
| Conversor Analógico a Digital (ADC) | ADS1115 |
| Comunicación de Conversor Analógico a Digital (ADC) | I2C |
| Dirección Conversor Analógico a Digital (ADC) | 0x48 |
| Disposición de Terminales<br>  | A0 : Entrada Analogica 0 – ADS1115 – 0x48 – AIN0<br>A1 : Entrada Analogica 1 – ADS1115 – 0x48 – AIN1<br>A2 : Entrada Analogica 2 – ADS1115 – 0x48 – AIN2<br>A3 : Entrada Analogica 3 – ADS1115 – 0x48 – AIN3 |

# Canales de Comunicación

## RS-485 Comunicación

| | |
|---|---|
| Modo de Comunicacion | HALF-DUPLEX |
| Transceptor | MAX485 |
| Unidad de Carga | 1/4 |
| Control de Flujo / Pin de Control de Dirección | GPIO22 |
| Pin TX | GPIO26 |
| Pin RX | GPIO25 |
| Disposición de Terminales |  |

## Ethernet SPI – W5500

| | |
|---|---|
| Transceptor | W5500 |
| Velocidad | 10BaseT/100BaseTX |
| Admite Negociación Automática | Si |
| Tamaño del  Buffer TX/RX | Memoria Interna de 32 Kbytes |
| Protocolos TCP/IP Cableados Compatibles | TCP, UDP, ICMP, IPv4, ARP, IGMP, and PPPoE |
| Número de Tomas Independientes Simultáneas | 8 |
| SCSn | GPIO5 |
| MISO | GPIO19 |
| MOSI | GPIO23 |
| SCLK | GPIO18 |
| RSTn | GPIO22 |
| INTn | GPIO2 |

## Comunicación GSM

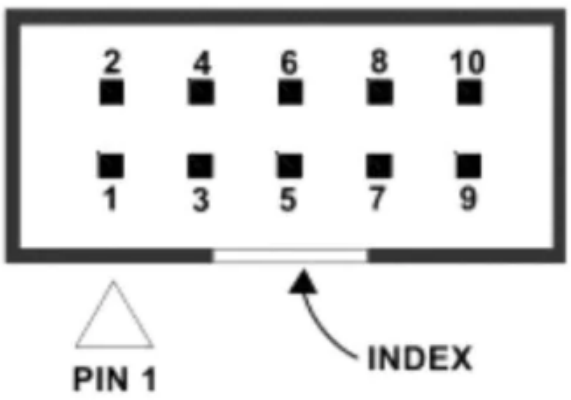
| | |
|---|---|
| Modelo de Módem GSM | SIM800C |
| FCC ID | UDU-SIM800C |
| TAC | 86610402 |
| RXD | GPIO33 |
| TXD | GPIO32 |

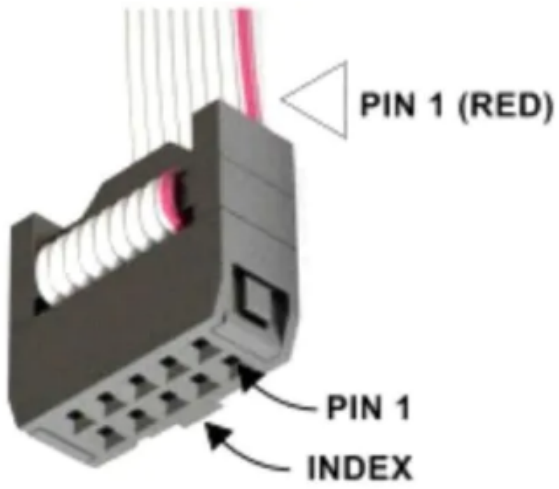
# Mapa GPIO

| GPIO | Descripción | Uso |
|---|---|---|
| 0 | emite una señal PWM en el arranque | NRST |
| 1 | | TX0 |
| 2 | | |
| 3 | | RX0 |
| 4 | sólo entrada | Entrada Digital 5 |
| 5 | | SCSN |
| 12 | fallo de arranque si se activa | |
| 13 | sólo entrada | Entrada Digital 4 |
| 14 | sólo entrada | Entrada Digital 3 |
| 15 | | SD CARD CS |
| 16 | | SDA |
| 17 | | SCL |
| 18 | | SCLK |
| 19 | | MISO |
| 20 | | |
| 21 | | |

| | | |
|---|---|---|
| 22 | | RS485- Control de Flujo |
| 23 | | MOSI |
| 24 | | |
| 25 | | RS485-RX |
| 26 | | RS485-TX |
| 31 | | |
| 32 | | GSM TX |
| 33 | | GSM RX |
| 34 | sólo entrada | Entrada Digital 1 |
| 35 | sólo entrada | Entrada Digital 2 |
| 36 | | Botones |
| 37 | | |

# Puerto de Expansión




| PIN | ESP32 Conexión |
|---|---|
| 1 | NO CONECTADO |
| 2 | NO CONECTADO |
| 3 | 5V |
| 4 | NO CONECTADO |
| 5 | BOOT GPIO0 |
| 6 | NO CONECTADO |
| 7 | 3.3V |
| 8 | SCL2 GPIO17 |
| 9 | GND |
| 10 | SDA2 GPIO16 |

# NORVI GSM-AE08-I-G – USER GUIDE

## Programación

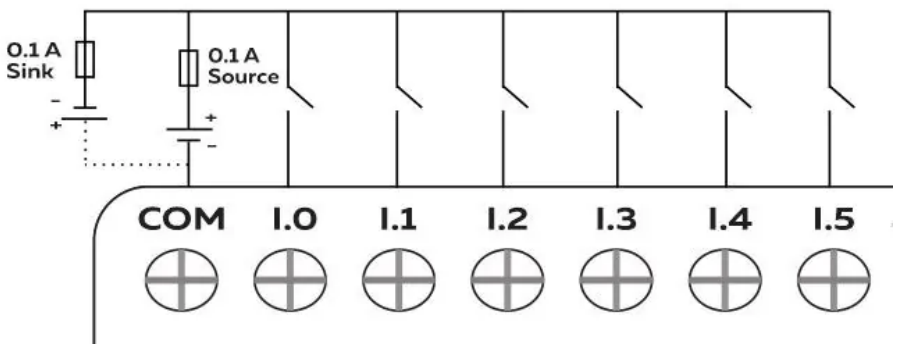
El NORVI GSM-AE08-I-G tiene un mini puerto USB para la conexión en serie con el SoC para la programación. Cualquier IDE de programación compatible con ESP32 se puede utilizar para programar el controlador. Siga esta Guía para programar los controladores NORVI basados en ESP32 con el IDE Arduino.

SoC: ESP32-WROOM32
Puerto Programable: USB UART

## Entradas Digitales

### Cableado de Entradas Digitales

Las entradas digitales del NORVI GSM-AE08-I-G pueden configurarse como conexiones de Fuente y de Sumidero. La polaridad inversa de la entrada digital debe suministrarse al terminal común.



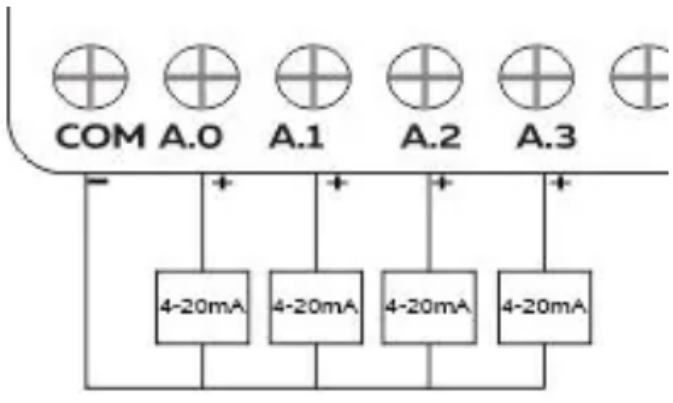

### Programación de Entradas Digitales

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 de GPIO en la hoja de datos para el GPIO de entrada digital.

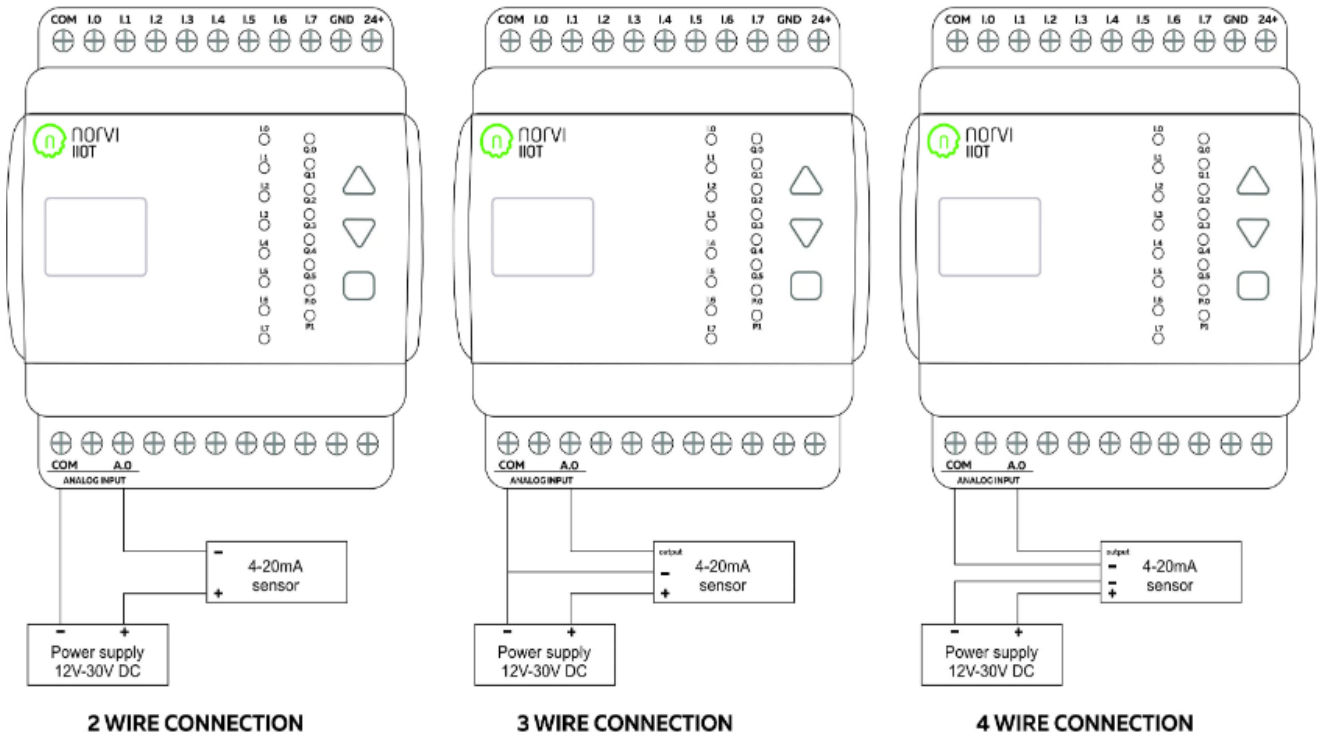
```
#define INPUT1 34
void setup() {
   Serial.begin(9600);
   Serial.println("Device Starting");
   pinMode(INPUT1, INPUT);
}

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

## Entrada Analógica 4 – 20 mA

### Cableado de Entradas Analógicas



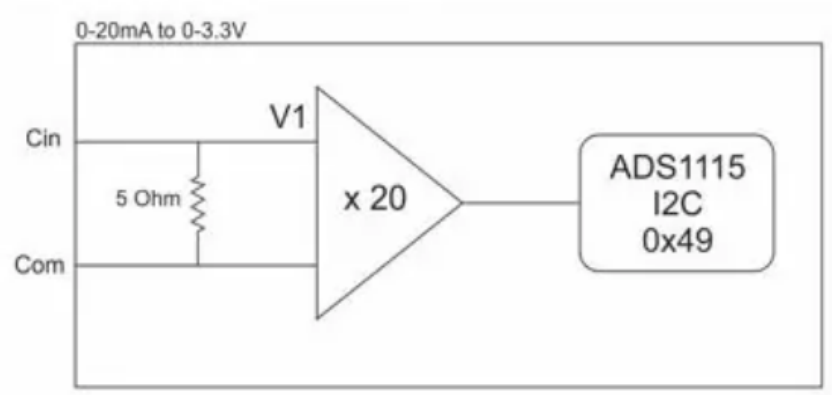

NORVI GSM-AE08-I-G Cableado de entrada analógica




## Lectura de la Entrada Analógica

Leyendo la dirección I2C correspondiente del ADC se obtiene el valor de la Entrada Analógica.

ADS1115 Current reading calibration.




Here's a general formula:

$$Cout = \left(\frac{ADC\ Reading}{32767}\right) \times \left(\frac{Vref}{20 \times 5\Omega}\right)$$

Cout = the output current you want to calculate.

ADC Reading = the serial reading obtained from the ADS115.

32767 = the maximum positive value in the digital output range.

Vref = the Full-scale Input Voltage of the ADS1115. Vref is set to ±4.096 volts.

5Ω = the shunt resistor value.

20 = the OPAMP multiplier.

If the ADC reading is 4478,

$$Cout = \left(\frac{4478}{32767}\right) \times \left(\frac{4096mV}{20 \times 5\Omega}\right)$$

$$Cout = 5.6\text{mA}$$

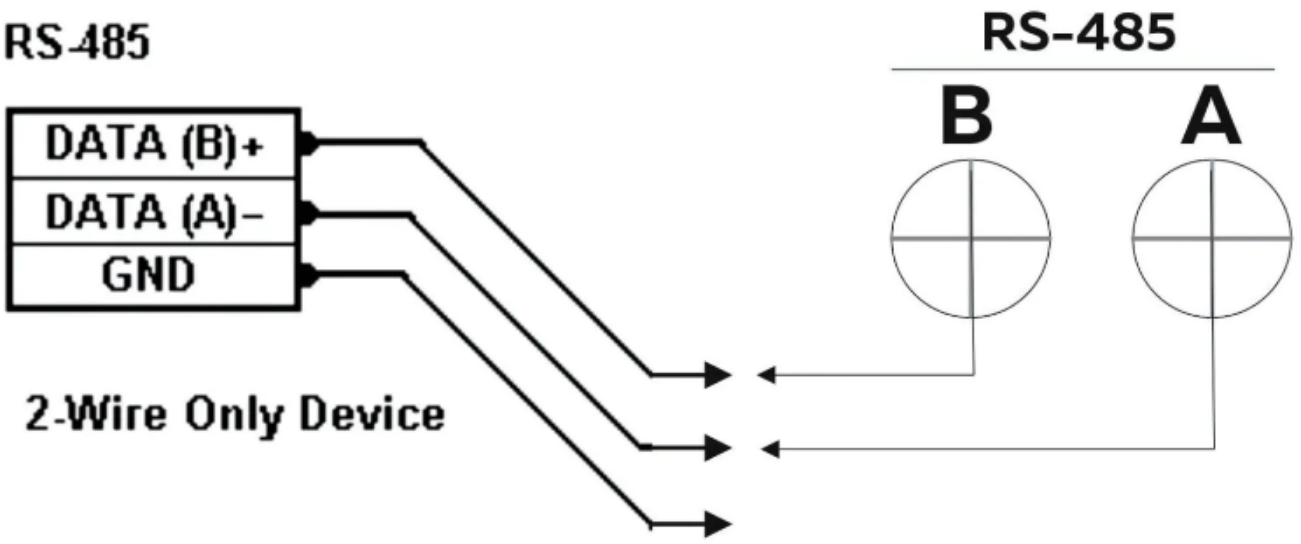
## Programación de Entradas Analógicas

```
#include <Adafruit_ADS1X15.h>
#include <Wire.h>
Adafruit_ADS1115 ads1;

void setup() {
   Serial.begin(9600);
   Serial.println("Device Starting");
   Wire.begin(16,17);
   ads1.begin(0x48);
   ads1.setGain(GAIN_ONE);
```

```
}

void loop() {
  Serial.print("Analog 0 ");
  Serial.println(ads1.readADC_SingleEnded(0));
  delay(10);
  Serial.print("Analog 1 ");
  Serial.println(ads1.readADC_SingleEnded(1));
  delay(10);
  Serial.print("Analog 2 ");
  Serial.println(ads1.readADC_SingleEnded(2));
  delay(10);
  Serial.print("Analog 3 ");
  Serial.println(ads1.readADC_SingleEnded(3));
  delay(10);
  Serial.println("");
  delay(500);
}
```

# Comunicación RS-485

## Cableado RS-485




| | |
|---|---|
| Controlador | MAX485 |
| UART RX | GPIO25 |
| UART TX | GPIO26 |
| Control de Flujo | GPIO22 |



## Programación RS-485

La conexión RS-485 de la serie NORVI-GSM-AE08 utiliza un modo semidúplex del transmisor MAX485 con comunicación UART.

```
#define RXD 25
#define TXD 26
#define FC  22

void setup() {
 Serial.begin(9600);
 pinMode(FC, OUTPUT);
 Serial1.begin(9600, SERIAL_8N1,RXD,TXD);
}
```

```
void loop() {
  digitalWrite(FC, HIGH);                  // Make FLOW CONTROL pin HIGH
  Serial1.println("RS485 01 SUCCESS");     // Send RS485 SUCCESS serially
  delay(500);                              // Wait for transmission of data
  digitalWrite(FC, LOW);                   // Receiving mode ON

  while (Serial1.available()) {  // Check if data is available
    char c = Serial1.read();     // Read data from RS485
    Serial.write(c);             // Print data on serial monitor
  }
  delay(1000);
}
```

# Pantalla OLED Incorporada

| | |
|---|---|
| Controlador de Pantalla | SSD1306 |
| Comunicación | I2C |
| Dirección de Modulo | 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.

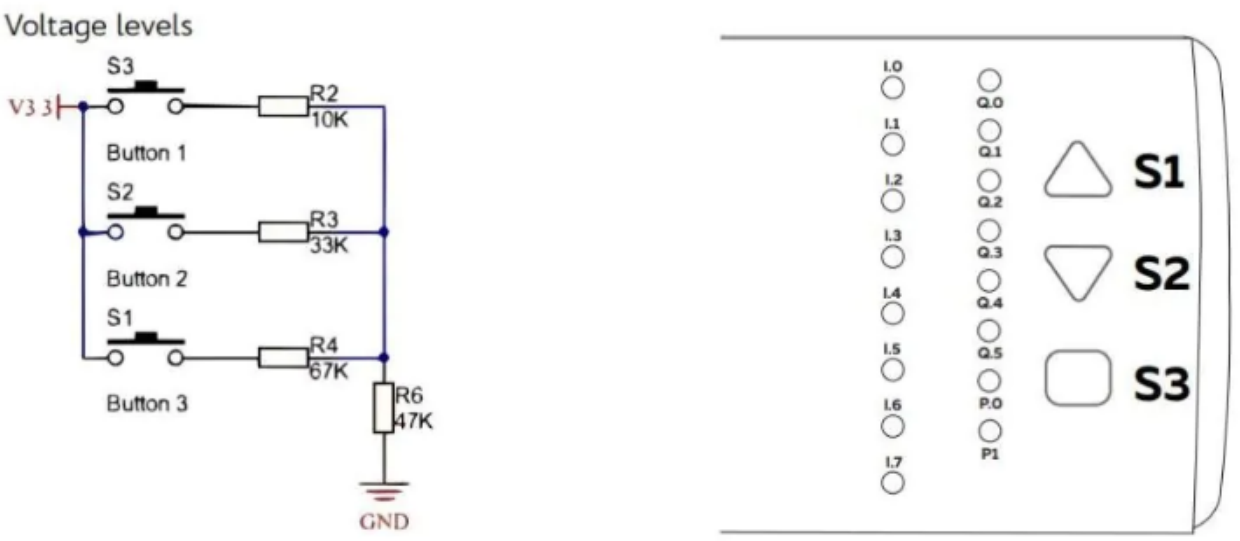
## Programación de Pantalla OLED

```
#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 Integrados

| | |
|---|---|
| Modo de Lectura | ADC (Conversión Analógica a Digital) |
| IO Analógico | GPIO36 |

GPIO Conexión de los botones




Botón incorporado Esquema interno

## Programación de Botones

```
#define buttonPin 36
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);
}
```

# RTC Interno

| | |
|---|---|
| Chip RTC | DS3231 |
| Tipo de Batería de Respaldo | CR2032 |
| Interfaz | I2C |
| Direccion I2C | 0x68 |
| Pin SCL | GPIO17 |
| Pin SDA | GPIO16 |

## Programación RTC

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

```
#include <Adafruit_SSD1306.h>
#include "RTClib.h"

RTC_DS3231 rtc;
char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"};

#define SCREEN_WIDTH 128  // OLED display width, in pixels
#define SCREEN_HEIGHT 64  // OLED display height, in pixels
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

void setup()  {
  Serial.begin(9600);
  if (! rtc.begin()) {
    Serial.println("Couldn't find RTC");
    while (1);
  }
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C))  {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }
  rtc.adjust(DateTime(__DATE__, __TIME__));
  display.display();
  delay(2);
  display.clearDisplay();
  display.clearDisplay();
  display.setTextColor(WHITE);   //display.startscrollright(0x00, 0x0F);
  display.setTextSize(2);
  display.setCursor(0,5);
  display.print("  Clock ");
  display.display();
  delay(3000);
}

void loop() {
  DateTime now = rtc.now();
  display.clearDisplay();
  display.setTextSize(2);
  display.setCursor(75,0);
  display.println(now.second(), DEC);

  display.setTextSize(2);
  display.setCursor(25,0);
  display.println(":");

  display.setTextSize(2);
  display.setCursor(65,0);
  display.println(":");
  display.setTextSize(2);
  display.setCursor(40,0);
  display.println(now.minute(), DEC);

  display.setTextSize(2);
  display.setCursor(0,0);
  display.println(now.hour(), DEC);

  display.setTextSize(2);
  display.setCursor(0,20);
  display.println(now.day(), DEC);
```

```
display.setTextSize(2);
display.setCursor(25,20);
display.println("-");

display.setTextSize(2);
display.setCursor(40,20);
display.println(now.month(), DEC);

display.setTextSize(2);
display.setCursor(55,20);
display.println("-");

display.setTextSize(2);
display.setCursor(70,20);
display.println(now.year(), DEC);

display.setTextSize(2);
display.setCursor(0,40);
  display.print(daysOfTheWeek[now.dayOfTheWeek()]);
  display.display();
}
```

# Compatible con Tarjetas Micro SD

| | |
|---|---|
| TARJETA SD  CS | GPIO15 |
| MISO | GPIO19 |
| MOSI | GPIO23 |
| SCLK | GPIO18 |

Conexiones GPIO de la tarjeta SD

## Programación de Tarjeta SD

```
#include <SD.h>
#define PIN_SPI_CS 15 // The ESP32 pin GPIO15
File myFile;

void setup() {
  Serial.begin(9600);
  if (!SD.begin(PIN_SPI_CS)) {
    Serial.println(F("SD CARD FAILED, OR NOT PRESENT!"));
    while (1); // stop the program
  }
  Serial.println(F("SD CARD INITIALIZED."));
  if (!SD.exists("esp32.txt")) {
    Serial.println(F("esp32.txt doesn't exist. Creating esp32.txt file..."));
    // create a new file by opening a new file and immediately close it
    myFile = SD.open("esp32.txt", FILE_WRITE);
    myFile.close();
  }
  // recheck if file is created or not
  if (SD.exists("esp32.txt"))
    Serial.println(F("esp32.txt exists on SD Card."));
  else
    Serial.println(F("esp32.txt doesn't exist on SD Card."));
```

```
}

void loop() {
}
```

# Ethernet SPI – W5500

| | |
|---|---|
| Transceptor | W5500 |
| SCSn | GPIO5 |
| MISO | GPIO19 |
| MOSI | GPIO23 |
| SCLK | GPIO18 |
| RSTn | GPIO22 |

## Programación Ethernet

```
#include <Ethernet.h>
#include <EthernetUdp.h>

byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};
unsigned int localPort = 8888;
const char timeServer[] = "time.nist.gov";
const int NTP_PACKET_SIZE = 48;
byte packetBuffer[NTP_PACKET_SIZE];
EthernetUDP Udp;

void setup() {
  Serial.begin(9600);
  Ethernet.init(5);  // ESP32 with Adafruit Featherwing Ethernet
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    if (Ethernet.hardwareStatus() == EthernetNoHardware) {
      Serial.println("Ethernet shield was not found.  Sorry, can't run without hardware. :(");
    }
    if (Ethernet.linkStatus() == LinkOFF) {
      Serial.println("Ethernet cable is not connected.");
    }
  }
  else {
    Udp.begin(localPort);
    sendNTPpacket(timeServer);
    delay(1000);
    if (Udp.parsePacket()) {
      Udp.read(packetBuffer, NTP_PACKET_SIZE);
      unsigned long highWord = word(packetBuffer[40], packetBuffer[41]);
      unsigned long lowWord = word(packetBuffer[42], packetBuffer[43]);
      unsigned long secsSince1900 = highWord << 16 | lowWord;

      const unsigned long seventyYears = 2208988800UL;
      unsigned long epoch = secsSince1900 - seventyYears;

      Serial.print("Unix time = ");
      Serial.println(epoch);
```

```
      Serial.print("The UTC time is ");
      Serial.print((epoch  % 86400L) / 3600);
      Serial.print(':');
      if (((epoch % 3600) / 60) < 10) {
        Serial.print('0');
      }
      Serial.print((epoch  % 3600) / 60);
      Serial.print(':');
      if ((epoch % 60) < 10) {
        Serial.print('0');
      }
      Serial.println(epoch % 60);
    }
    delay(3000);
    Ethernet.maintain();
  }
}

void loop() {
  // Your loop code for Ethernet can go here if needed.
}

void sendNTPpacket(const char * address) {
  memset(packetBuffer, 0, NTP_PACKET_SIZE);
  packetBuffer[0] = 0b11100011;
  packetBuffer[1] = 0;
  packetBuffer[2] = 6;
  packetBuffer[3] = 0xEC;
  packetBuffer[12]  = 49;
  packetBuffer[13]  = 0x4E;
  packetBuffer[14]  = 49;
  packetBuffer[15]  = 52;
  Udp.beginPacket(address, 123);
  Udp.write(packetBuffer, NTP_PACKET_SIZE);
  Udp.endPacket();
}
```

## Comunicación GSM

| | |
|---|---|
| Modelo del Módem GSM | SIM800C |
| FCC ID | UDU-SIM800C |
| TAC | 86610402 |
| RXD | GPIO33 |
| TXD | GPIO32 |

Conexiones GPIO de la comunicación GSM

## Programación de Comunicación GSM

```
#define GSM_RX 33
#define GSM_TX 32
unsigned long int timer1 = 0;

void setup() {
  // put your setup code here, to run once:
```

```
Serial.begin(9600);
Serial.println("Hello");
Serial2.begin(9600, SERIAL_8N1, GSM_RX, GSM_TX);

timer1 = millis();
Serial2.println("AT");
while(millis()<timer1+10000){
  while (Serial2.available()) {
    int inByte = Serial2.read();
    Serial.write(inByte);
  }
}
timer1 = millis();
Serial2.println("AT+CPIN?");
while(millis()<timer1+10000){
  while (Serial2.available()) {
    int inByte = Serial2.read();
    Serial.write(inByte);
  }
  timer1 = millis();
  Serial2.println("AT+CFUN?");
  while(millis()<timer1+10000){
    while (Serial2.available()) {
      int inByte = Serial2.read();
      Serial.write(inByte);
    }
  }
  Serial.println("AT TIMEOUT");
}

void loop() {
    while (Serial.available()) {
      int inByte = Serial.read();
      Serial2.write(inByte);
  }
  while (Serial2.available()) {
      int inByte = Serial2.read();
      Serial.write(inByte);
  }
  // put your main code here, to run repeatedly:
}
```

# REINICIO y USB