NORVI GSM-AE04-I-L – USER GUIDE

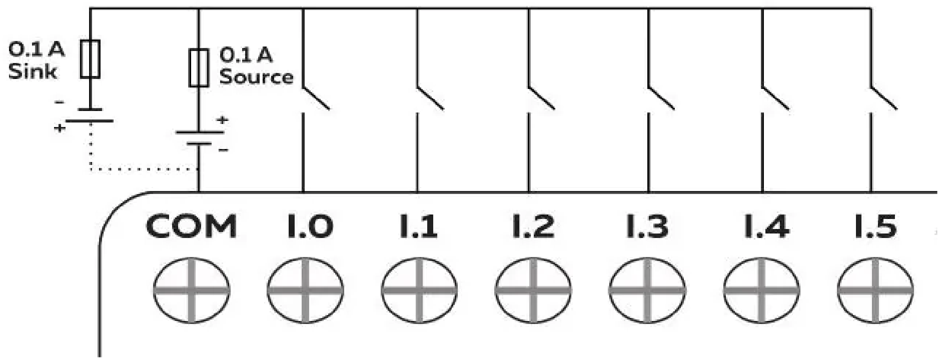
Programming

The NORVI GSM-AE04-I-L has a mini USB Port for serial connection with the SoC for programming. Any ESP32-supported programming IDE can be used to program the controller. Follow this Guide to programming NORVI ESP32-based controllers with the Arduino IDE.

SoC: ESP32-WROOM32
Programming Port: USB UART

Digital Inputs

Wiring Digital Inputs



NORVI GSM-AE04-I-L Digital Input Wiring

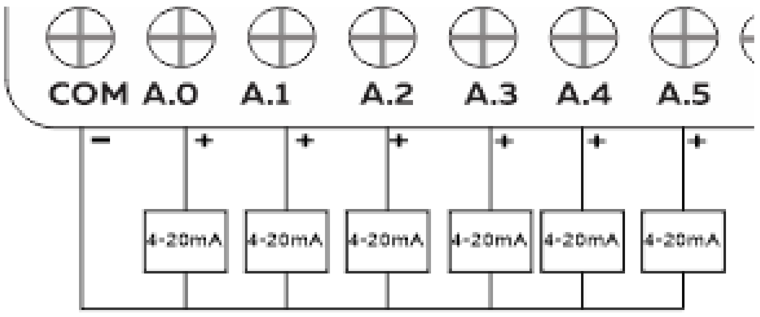
Programming Digital Inputs

Reading the relevant GPIO of the ESP32 gives the value of the Digital Input. When the inputs are in the OFF state, the GPIO goes HIGH, and when the input is in the ON stage, the GPIO goes LOW. Refer to the GPIO allocation table in the datasheet for the digital input GPIO.

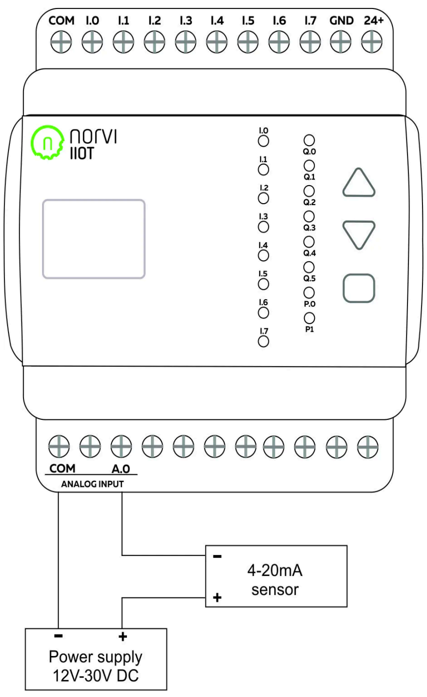
```
#define INPUT1 27

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

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

4 – 20 mA Analog Input

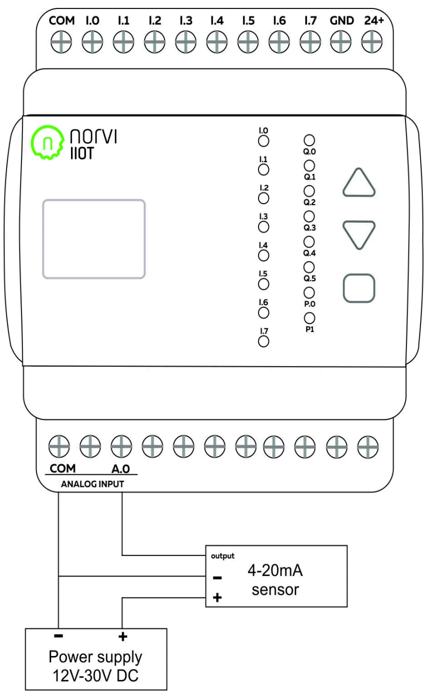
Wiring Analog Inputs



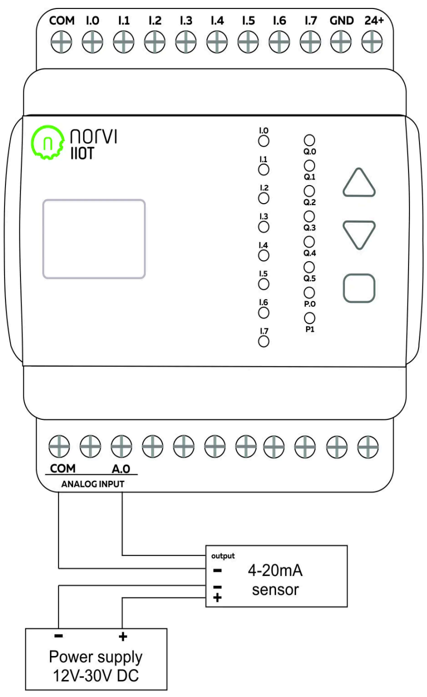
NORVI GSM-AE04-I-L Analog Input Wiring



2 WIRE CONNECTION



3 WIRE CONNECTION

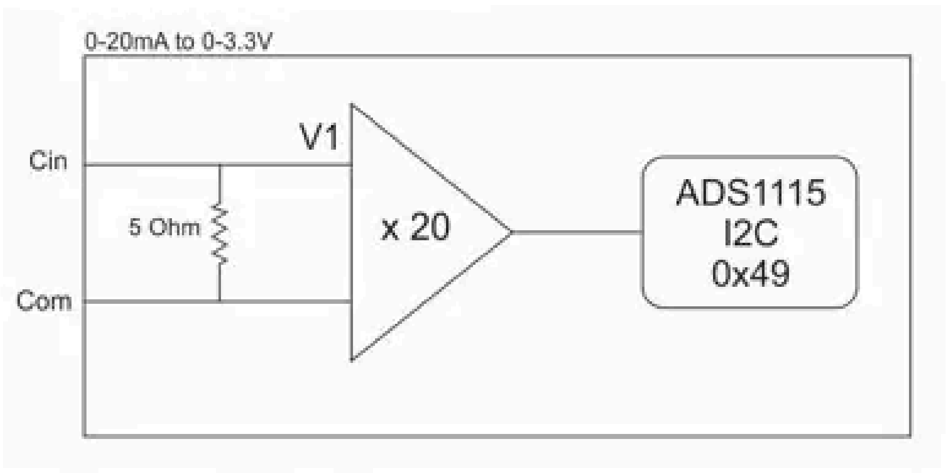


4 WIRE CONNECTION

Reading Analog Input

Reading the relevant I2C address of the ADC gives the value of the Analog Input.

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,

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

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

Programming Analog Inputs

```
#include <Adafruit_ADS1X15.h>
#include <Wire.h>
Adafruit_ADS1115 ads1;
Adafruit_ADS1115 ads2;

void setup() {
   Serial.begin(9600);
```

```
    Serial.println("Device Starting");
    Wire.begin(16,17);
    ads1.begin(0x48);
    ads2.begin(0x49);
    ads1.setGain(GAIN_ONE);
    ads2.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.print("Analog 4 ");
  Serial.println(ads2.readADC_SingleEnded(0));
  delay(10);
  Serial.print("Analog 5 ");
  Serial.println(ads2.readADC_SingleEnded(1));
  delay(10);
  Serial.println("");
  delay(500);
}
```

RS-485 Communication

RS-485 Wiring

Driver	MAX485
UART RX	GPIO25
UART TX	GPIO26
Flow Control	GPIO22

GPIO Connections of RS-485

Programming RS-485

NORVI-GSM-AE04 series RS-485 connection uses a half-duplex mode of MAX485 transmitter with UART Communication.

```
#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);
  }
```

Built-in OLED Display

Display driver	SSD1306
Communication	I2C
Module Address	0x3C
Resolution	128 x 64

0.96 OLED Display Specification

Refer to the GPIO allocation table in the Datasheet for the I2C GPIO of the OLED Display.

Library supported by the Adafruit_SSD0306 Library.

Wire.begin (SDA, SCL); is required to initialize I2C on the correct pins.

Programming OLED Display

```
#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() {
}
```

Built-in Buttons

Read mode	ADC (Analog to Digital Conversion)
Analog IO	GPIO36

GPIO Connection of buttons

Built-in button Internal Schematic

Programming Buttons

```
#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);
}
```

Internal RTC

RTC Chip	DS3231
Backup Battery Type	CR2032
Interface	I2C
I2C Address	0x68
SCL Pin	GPIO17
SDA Pin	GPIO16

Programming 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();
}
```

Micro SD Card Support

SD CARD CS	GPIO15
MISO	GPIO19
MOSI	GPIO23

SCLK	GPIO18

GPIO Connections of SD Card

Programming SD Card

```
#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() {
}
```

LTE Communication

Model of GSM Modem	QUECTEL EC25

FCC ID	XMR202008EC25AFXD
TAC	86675804
RXD	GPIO33
TXD	GPIO32
RESET	GPIO21

Model of GSM Modem	SIM7500
FCC ID	2AQ9M-SIM7500
TAC	86147503
RXD	GPIO33
TXD	GPIO32
RESET	GPIO21

Programming LTE Communication

```
#define GSM_RX 33
#define GSM_TX 32
#define GSM_RESET 21
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);
  pinMode(GSM_RESET, OUTPUT);
  digitalWrite(GSM_RESET, HIGH);
  timer1 = millis();
  Serial2.println("AT");
    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:
  }
```

RESET and USB