Distribuidor ARDUINO autorizado

A000024

SHIELD ETHERNET V2 ARDUINO

Descripción:

Este módulo permite realizar acciones de control a través de internet por medio de la conexión entre la placa Arduino y el cable RJ-45. Es un elemento de código abierto basado en el chip Ethernet Wiznet W5500. Admite el módulo Power over Ethernet (PoE) diseñado para extraer energía de un cable Ethernet categoría 5 de par trenzado convencional.

Contenido de producto:

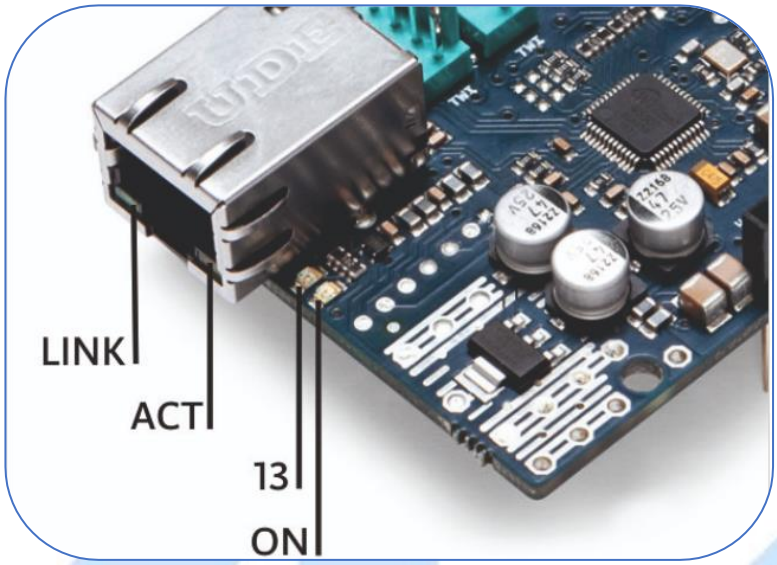
- Módulo Ethernet Shield 2 para Arduino.

No contiene placa Arduino ni módulo PoE.

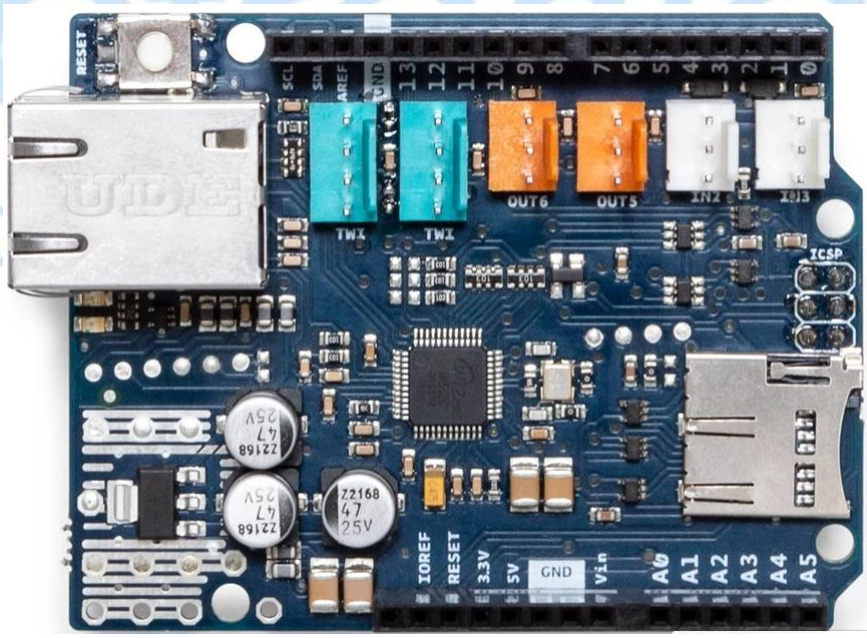
Voltaje de funcionamiento	5V
Controlador Ethernet	W5500 con buffer interno de 32k
Velocidad de conexión	10/ 100 Mb
Compatibilidad	Arduino UNO y Mega
Especificaciones anexas	• Admite conexión RJ-45 estándar (cable RJ-45 no incluido). • Ranura para tarjeta micro-SD incorporada. • Incluye controlador de reinicio para módulo Ethernet W5500 (funcionamiento óptimo al encenderlo). • Admite 8 conexiones de enchufe simultaneas.

AG Electrónica ¿Qué vamos a innovar hoy?

Estructura de módulo

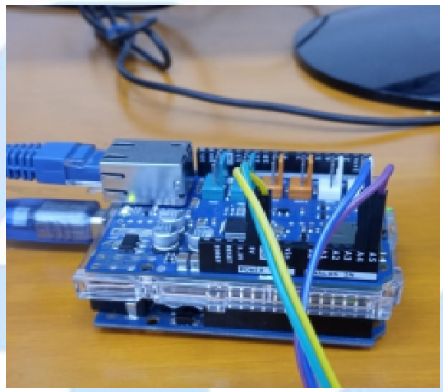


Indicador	Descripción de actividad
LINK	Indica la presencia de un enlace de red y parpadea cuando el módulo transmite o recibe datos.
ACT	Parpadea cuando hay actividad recepción (RX) o transmisión (TX) presente.
13	LED integrado estándar de Arduino
ON	Indica que la placa y el módulo están encendidos
Anexos	• 2 conectores TinkerKit para 2 entradas analógicas conectadas a A2 y A3 (blanco). • 2 conectores TinkerKit para 2 salidas analógicas conectados a salidas PWM en pines D5 y D6 (Naranja). • 2 conectores TinkerKit (uno para entrada y otro para salida) para interfaz TWI de 4 pines.



Funcionamiento:

Se realiza conexión del hat sobre el arduino uno o mega según sea el caso



Descargamos las librerias pertinentes de ethernet en el IDE de Arduino: y posteriormente colocamos el siguiente código:

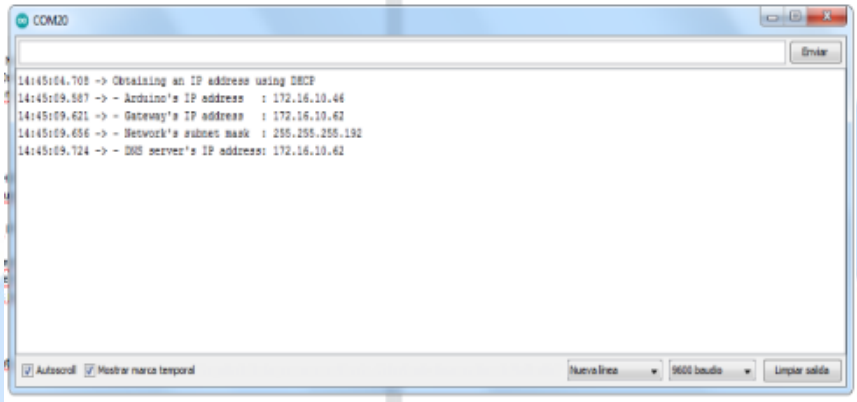
```
#include <SPI.h>
#include <Ethernet.h>
byte mac[] = {  0xA8, 0x61, 0x0A, 0xAE, 0x88, 0xA4}; //la MAC esta impresa debajo del escudo
void setup() {
  Serial.begin(9600);//imprimiremos la DHCP con el fin de validar las direcciones disponibles
Serial.println("Obtaining an IP address using DHCP");
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to obtaining an IP address");

    // Valida si el escudo esta conectado
    if (Ethernet.hardwareStatus() == EthernetNoHardware)
      Serial.println("Ethernet shield was not found");
  // Valida si el cable esta conectado
    if (Ethernet.linkStatus() == LinkOFF)
   Serial.println("Ethernet cable is not connected.");
    while (true);
  }
  // Muestra la IP, submascara, la dirección gateway IP , y la direccion IP del DNS
  Serial.print("- Arduino's IP address   : ");
  Serial.println(Ethernet.localIP());
Serial.print("- Gateway's IP address   : ");
  Serial.println(Ethernet.gatewayIP());
  Serial.print("- Network's subnet mask  : ");
  Serial.println(Ethernet.subnetMask());
  Serial.print("- DNS server's IP address: ");
  Serial.println(Ethernet.dnsServerIP());

  // TODO:  en esta parte se coloca el codigo HTML dependiendo la aplicacion}
void loop() {
  // TODO: en esta parte se coloca el codigo HTML dependiendo la aplicacion
}
```

Funcionamiento:

Abrimos el COM



La direccion IP que ingresaremos en cualquier navegador sera la que nos muestre aqui.

Creando un WEB server

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

// Enter a MAC address and IP address for your controller below.
// The IP address will be dependent on your local network:
byte mac[] = {0xA8, 0x61, 0x0A, 0xAE, 0x88, 0xA4};
IPAddress ip(172, 16, 10, 47);//esta ip la tomamos del resutado del codigo anterior

// Initialize the Ethernet server library
// with the IP address and port you want to use
// (port 80 is default for HTTP):
EthernetServer server(80);
void setup() {
  // You can use Ethernet.init(pin) to configure the CS pin
  //Ethernet.init(10);  // Most Arduino shields
  //Ethernet.init(5);   // MKR ETH shield
  //Ethernet.init(0);   // Teensy 2.0
  //Ethernet.init(20);  // Teensy++ 2.0
  //Ethernet.init(15);  // ESP8266 with Adafruit Featherwing Ethernet
  //Ethernet.init(33);  // ESP32 with Adafruit Featherwing Ethernet

  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }
  Serial.println("Ethernet WebServer Example");
```

Funcionamiento:

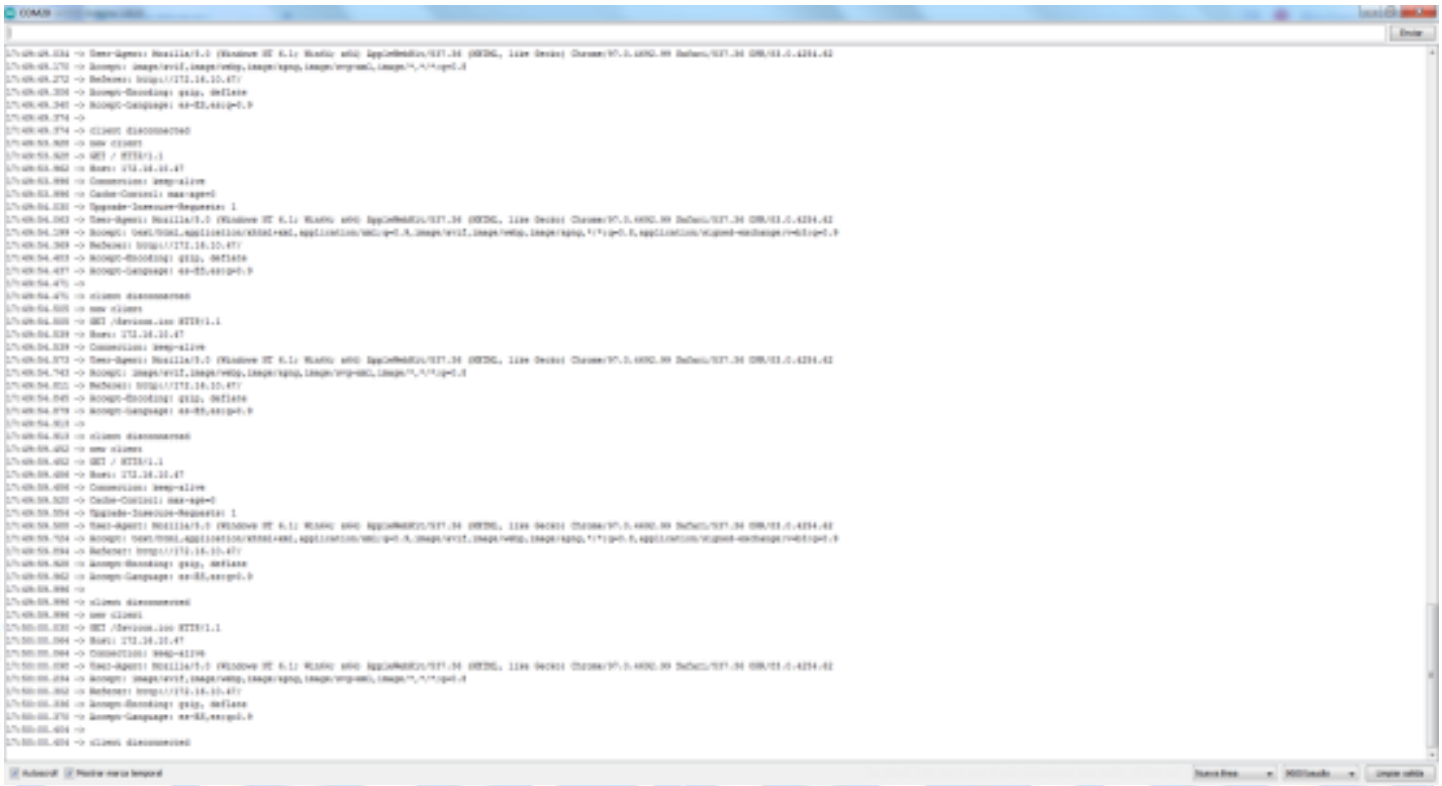
```
// start the Ethernet connection and the server:
  Ethernet.begin(mac, ip);
  // Check for Ethernet hardware present
  if (Ethernet.hardwareStatus() == EthernetNoHardware) {
    Serial.println("Ethernet shield was not found.  Sorry, can't run without hardware. :(");
    while (true) {
      delay(1); // do nothing, no point running without Ethernet hardware
    }
  }
  if (Ethernet.linkStatus() == LinkOFF) {
    Serial.println("Ethernet cable is not connected.");
  }
  // start the server
  server.begin();
  Serial.print("server is at ");
  Serial.println(Ethernet.localIP());
}
void loop() {
  // listen for incoming clients
  EthernetClient client = server.available();
  if (client) {
    Serial.println("new client");
    // an http request ends with a blank line
    boolean currentLineIsBlank = true;
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        Serial.write(c);
        // if you've gotten to the end of the line (received a newline
        // character) and the line is blank, the http request has ended,
        // so you can send a reply
   if (c == '\n' && currentLineIsBlank) {
          // send a standard http response header
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: close");  // the connection will be closed after
          completion of the response
          client.println("Refresh: 5");  // refresh the page automatically every 5 sec
          client.println();
          client.println("<!DOCTYPE HTML>");
          client.println("<html>");
```

Funcionamiento:

```
 // output the value of each analog input pin
          for (int analogChannel = 0; analogChannel < 6; analogChannel++) {
            int sensorReading = analogRead(analogChannel);
            client.print("analog input ");
            client.print(analogChannel);
            client.print(" is ");
            client.print(sensorReading);
            client.println("<br />");
          }
          client.println("</html>");
          break;
        }
        if (c == '\n') {
          // you're starting a new line
          currentLineIsBlank = true;
        } else if (c != '\r') {
          // you've gotten a character on the current line
          currentLineIsBlank = false;
        }
      }
    }
    // give the web browser time to receive the data
    delay(1);
    // close the connection:
    client.stop();
    Serial.println("client disconnected");
  }
}
```

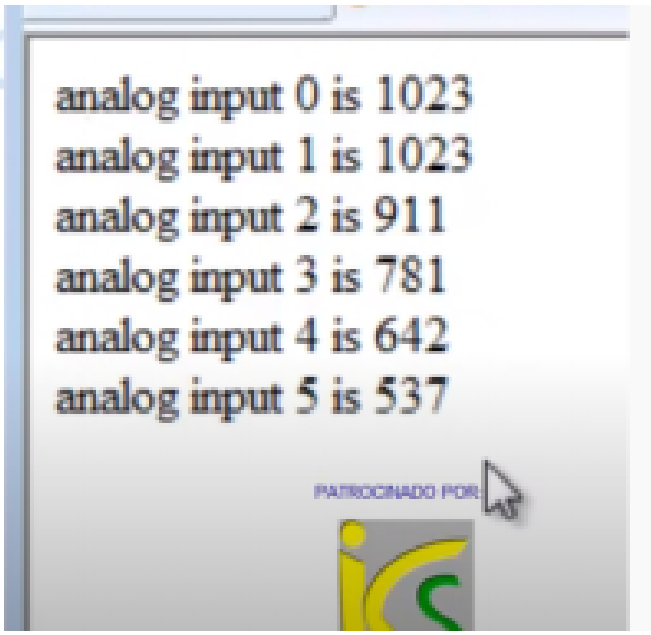
Funcionamiento:

Abrimos el COM:



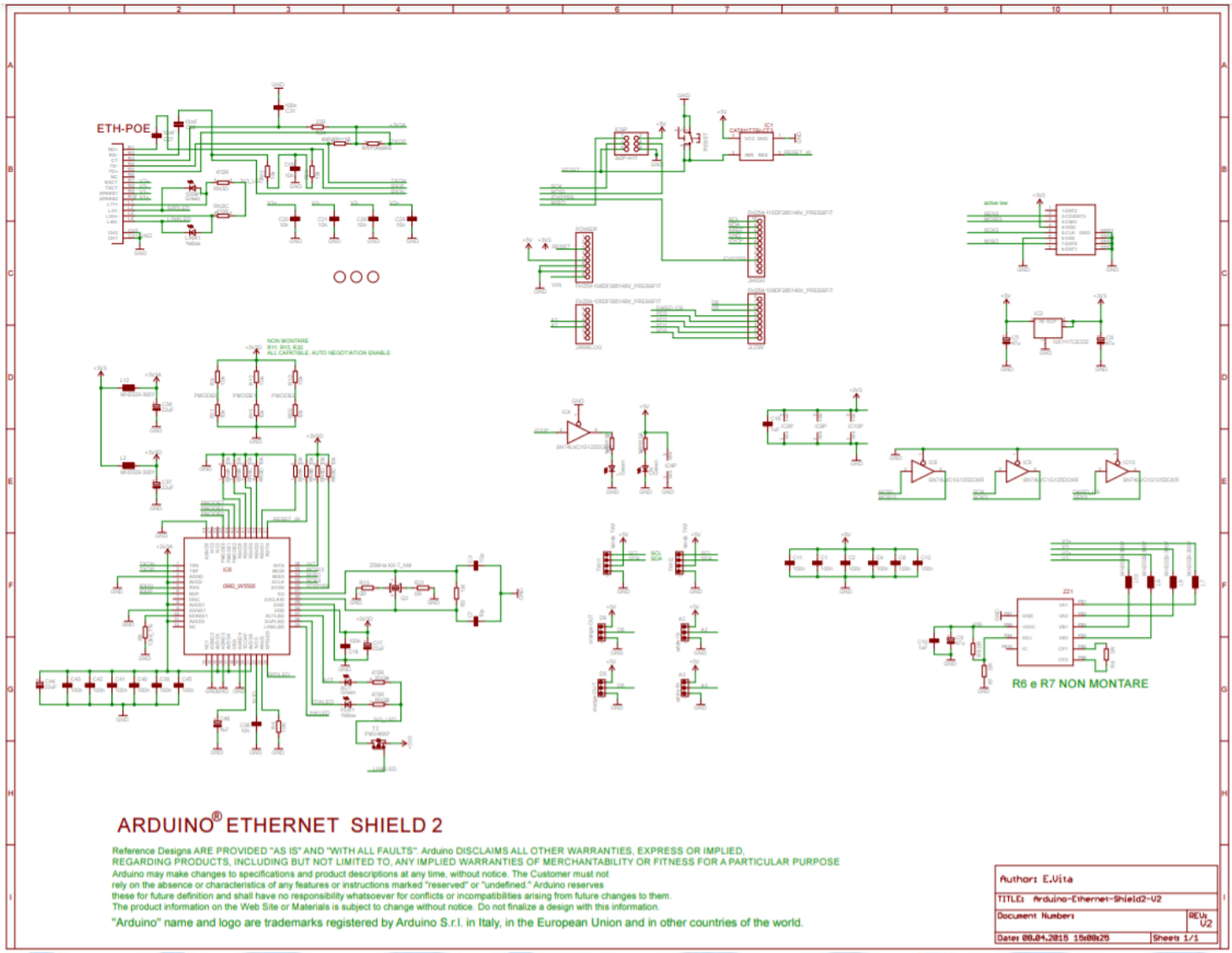
Este cuadro nos indicará quien se ha conectado a nuestro servidor WEB

Y este será el resultado de colocar en nuestro navegador la IP que generamos previamente

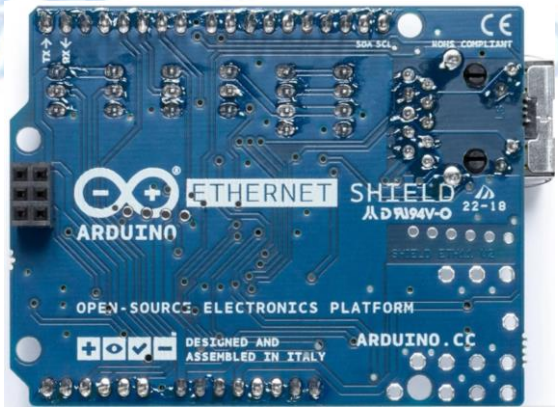


Distribuidor ARDUINO autorizado

Esquema módulo Ethernet Shield V2



Referencia:
https://www.arduino.cc/en/uploads/Main/arduino-Ethernet-Shield2-V2-sch.pdf?_gl=1*1oqsxbq*_ga*MTk1NDgwNTU3Ni4xNjM5NDIwNTU1*_ga_NEXN8H46L5*MTYzOTQyMDU1NC4xLjEuMTYzOTQyMDYwMi4w.



AG Electrónica S.A.P.I. de C.V. República del Salvador N° 20 Segundo Piso Teléfono: 55 5130 – 7210			
ACOTACIÓN: N/A	http://www.agelectronica.com	ESCALA: N/A	REALIZO: CNLS REV: ARSL
TOLERANCIA: N/A	SHIELD ETHERNET V2 ARDUINO		
TOLERANCIA: N/A	Fecha: 13/12/2021	**No. Parte: A000024**	

AG Electrónica ¿Qué vamos a innovar hoy?