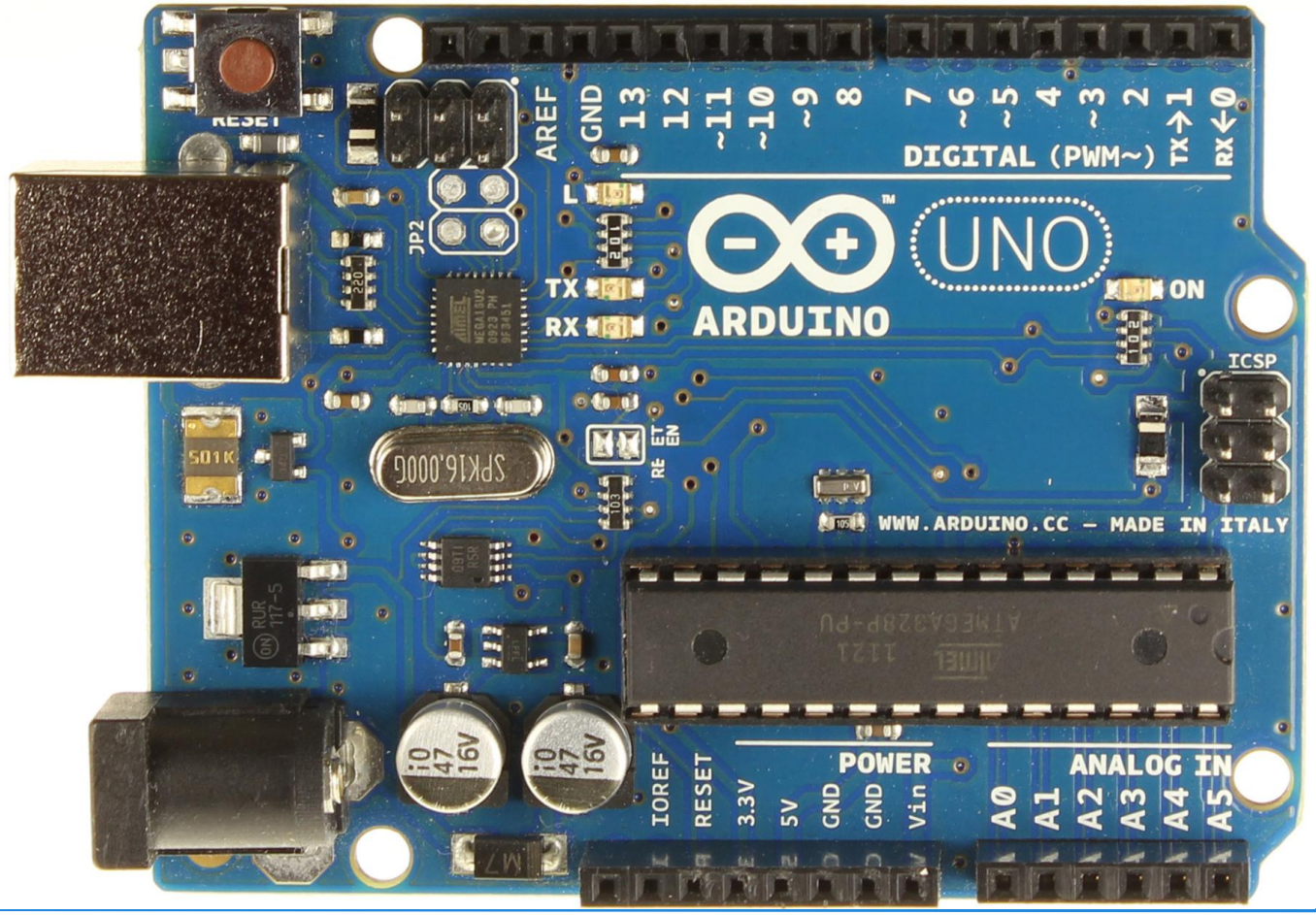
# Curso avanzado sobre Arduino: Ethernet

ElCacharreo.com

# Arduino Avanzado: Presente



# Arduino Avanzado: Presente


José Antonio Vacas Martínez

blog

javacasm@elcacharreo.com

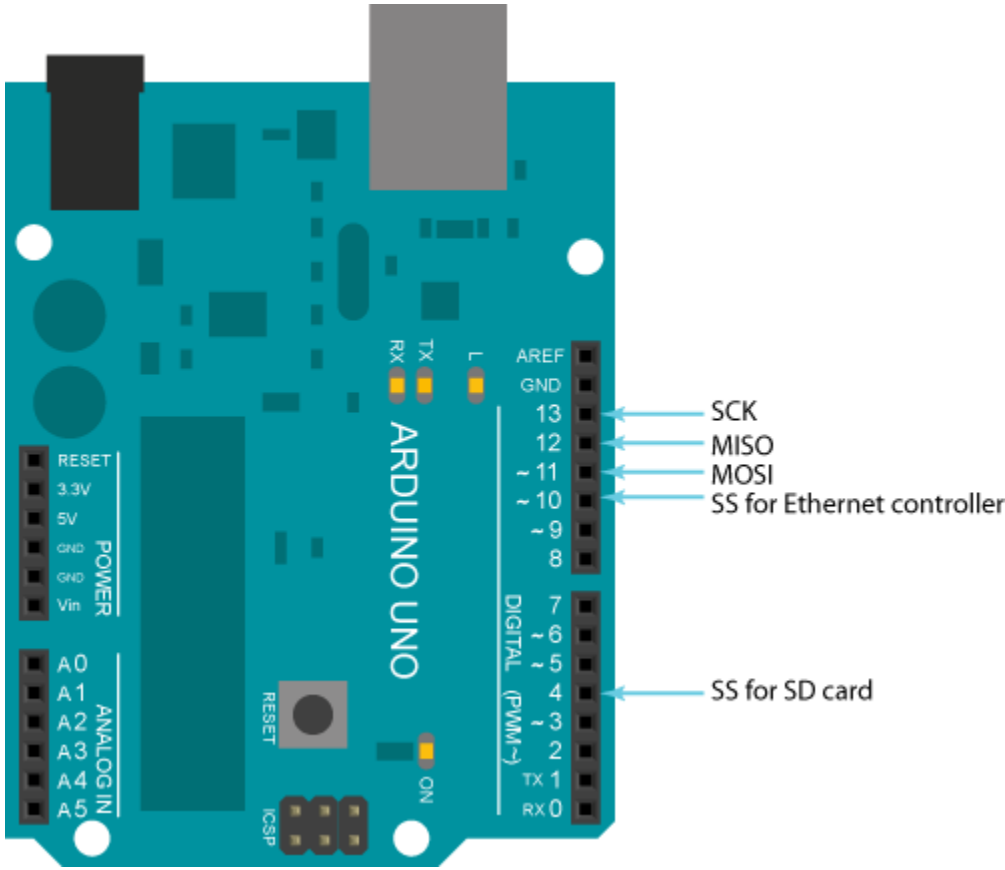
twitter
linkedin


# Arduino Avanzado: Ethernet

Puede utilizarse tanto como cliente como servidor, es decir enviado o recibiendo datos

Soporta hasta 4 conexiones simultáneas

http://www.instructables.com/id/Arduino-Ethernet-Shield-Tutorial/?ALLSTEPS



# Arduino Avanzado: Ethernet server

```
#include <Ethernet.h>
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
byte ip[] = { 10, 0, 0, 177 };    //the IP address for the shield:
byte gateway[] = { 10, 0, 0, 1 };// the router's gateway address:
byte subnet[] = { 255, 255, 0, 0 };// the subnet:

Server server = Server(23);

void setup(){
  // initialize the ethernet device
  Ethernet.begin(mac, ip, gateway, subnet);
  // start listening for clients
  server.begin();}

void loop(){
  // if an incoming client connects, there will be bytes available to read:
  Client client = server.available();
  if (client == true) {
    // read bytes from the incoming client and write them back
    // to any clients connected to the server:
    server.write(client.read());
  }
}
```

# Ejemplos Ethernet: webserver

```
void loop() {  // listen for incoming clients
  EthernetClient client = server.available();
  if (client) {
    Serial.println("new client");
    boolean currentLineIsBlank = true;     // an http request ends with a blank line
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();         Serial.write(c);
        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");
client.println();  client.println("<!DOCTYPE HTML>");  client.println("<html>");
                    // add a meta refresh tag, so the browser pulls again every 5 seconds:
          client.println("<meta http-equiv=\"refresh\" content=\"5\">");
          // 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;        }      }    }
    delay(1);     // give the web browser time to receive the data
    client.stop();     // close the connection:
    Serial.println("client disonnected");  }}
```

# Ethernet: ejemplo cliente

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

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
char server[] = "www.google.com";    // name address for Google (using DNS)
IPAddress ip(192,168,0,177);
EthernetClient client;
void setup() {
  Serial.begin(9600);   while (!Serial) {    ; // wait for serial port to connect. Needed for Leonardo only  }
  if (Ethernet.begin(mac) == 0) {    Serial.println("Failed to configure Ethernet using DHCP");
    Ethernet.begin(mac, ip);  }
  delay(1000);    Serial.println("connecting..."); // give the Ethernet shield a second to initialize:
  // if you get a connection, report back via serial:
  if (client.connect(server, 80)) {    Serial.println("connected");    // Make a HTTP request:
    client.println("GET /search?q=arduino HTTP/1.1");    client.println("Host: www.google.com");    client.println
("Connection: close");    client.println();  }
  else {        Serial.println("connection failed");  // kf you didn't get a connection to the server:}}
void loop()
{
  if (client.available()) {
    char c = client.read();
    Serial.print(c);
  }
  if (!client.connected()) {    Serial.println();    Serial.println("disconnecting.");    client.stop();
    while(true);  }}
```

# Shield: ENC28J60

enc28J60

Librería ethercard (by JeeLab)

https://github.com/jcw/ethercard/archive/master.zip

Diferencias:

- Precio
- Rendimiento
- Librerías más potente

# Shield: ENC28J60, ejemplo cliente

```
#include <EtherCard.h>
static byte mymac[] = { 0x74,0x69,0x69,0x2D,0x30,0x31 }; byte Ethernet::buffer[700];  static uint32_t timer;
char website[] PROGMEM = "www.google.com";
static void my_callback (byte status, word off, word len) {  // called when the client request is complete
  Serial.println(">>>");   Ethernet::buffer[off+300] = 0;   Serial.print((const char*) Ethernet::buffer + off);   Serial.println("...");}
void setup () {   Serial.begin(57600);   Serial.println("\n[webClient]");
  if (ether.begin(sizeof Ethernet::buffer, mymac) == 0)      Serial.println( "Failed to access Ethernet controller");
  if (!ether.dhcpSetup())     Serial.println("DHCP failed");
  ether.printIp("IP:  ", ether.myip);   ether.printIp("GW:  ", ether.gwip);     ether.printIp("DNS: ", ether.dnsip);
  if (!ether.dnsLookup(website))     Serial.println("DNS failed");
    ether.printIp("SRV: ", ether.hisip);}
void loop () {
  ether.packetLoop(ether.packetReceive());
    if (millis() > timer) {     timer = millis() + 5000;
    Serial.println();     Serial.print("<<< REQ ");
    ether.browseUrl(PSTR("/foo/"), "bar", website, my_callback);  }}
```

# Conclusiones

Gracias por vuestra atención