Welcome!

Thank you very much for purchasing our AZ-Delivery ENC28J60 eternet module. On the following pages, we will introduce you to how to use and setup this handy device.

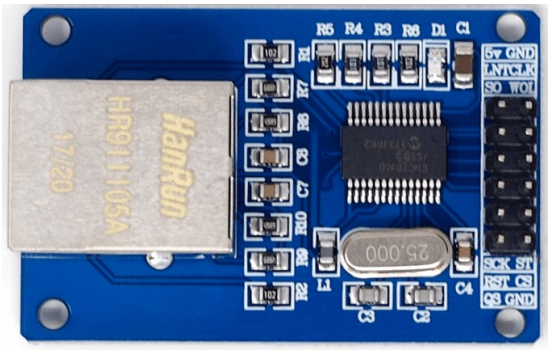
Have fun!

The network module is a low-cost, Ethernet-based, highly programmable module with an ENC28J60 microchip at its heart. The ENC28J60 chip has an SPI interface for communication with the host controller (our Arduino). Via SPI the chip only supports mode 0, with SCK (Clock) IDLE Low, LSB First communication and CS LOW active status.

Please note that the ENC28J60 is a 3.3 Volt chip, but its inputs are 5 Volt tolerant and do not need a level shifter for the Arduino. Also the outputs with 3.3 Volt level are above the high detection mark of the Arduino and can be connected directly to our Arduino. Nevertheless it is recommended to use the outputs with a level shifter for a more stable signal transmission. However, in our manual we do not do this because of simplicity.

The controller is IEEE 802.3 compatible and supports half and full duplex at 10 Mbit with automatic polarity detection and correction. It has an internal 8 KByte send and receive buffer.

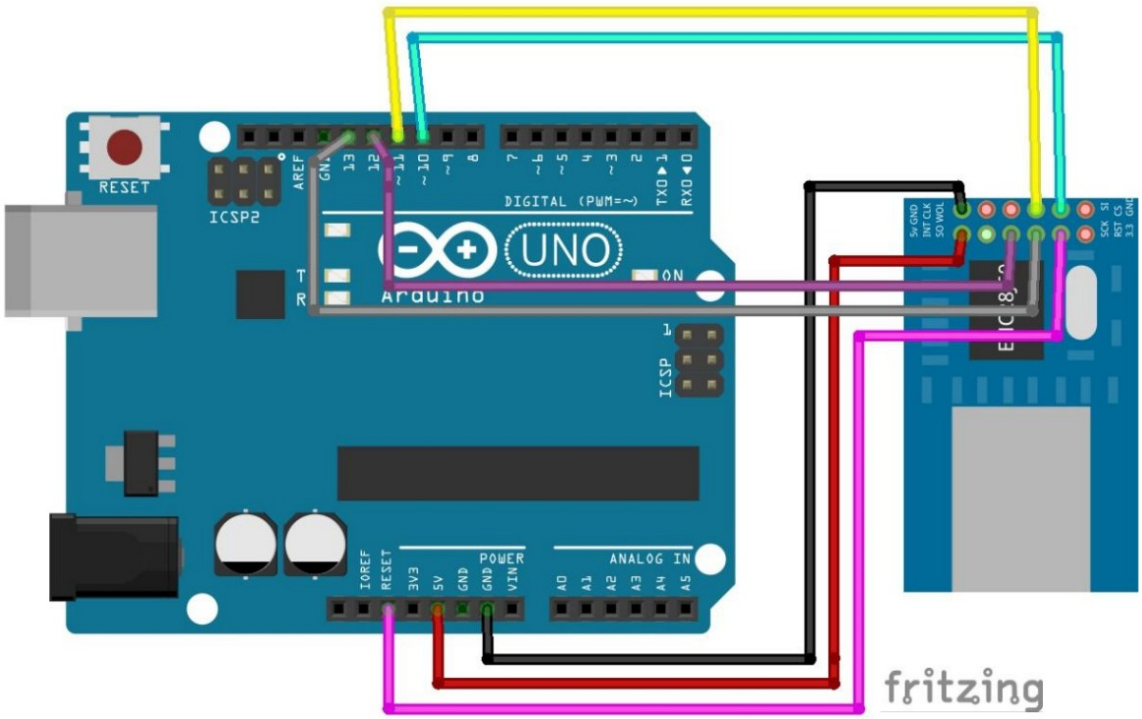
Wiring



5V (V+)	1	-	2	GND (Ground)
LNT (INT)	3	-	4	CLK (Clock out)
SO (MISO)	5	-	6	WOL (Wake on LAN)
SCK (SCLK)	7	-	8	ST (MOSI)
RST (Reset)	9	-	10	CS (Chip Select)
Q3 (3,3 Volt)	11	-	12	GND (Ground)

1. 5V - 5V power supply
2. GND - Ground
3. LNT - Interrupt (can be unconnected)
4. CLK - Clock out (can be unconnected)
5. SO - SPI BUS (Master in Slave out)
6. WOL - With appropriate programming, spec. WOL packets switch the output (can be unconnected)
7. SCK - SPI BUS (BUS Clock - maximal 20 Mhz)
8. ST - SPI BUS (Master out Slave in)
9. RST - Chip Reset
10. CS - SPI BUS (Chip Select LOW active)
11. Q3 - Alternative voltage max. 3,3V (MUST be unconnected)
12. GND - Ground

We now connect our module with the Arduino Uno as follows:



ENC28J60 Pin	>	Arduino pin
5V	>	5V
GND	>	GND
SCK	>	Pin 13
SO	>	Pin 12 - MISO
ST	>	Pin 11 - MOSI
CS	>	Pin 10 - Modifiable by the ether.begin()
RST	>	RESET

This completes the basic connection to our Arduino. The module is also brought into a defined initial state with each reset of our Arduino via the reset line, so that no problems are to be expected here with a possible new programming of the Arduino.

Programming basics

Before we go into more detail about the programming of the module, we have to make a few things in advance about the operation of the Ethernet controller. The chip only covers layer 1 (bit transmission) and layer 2 (protection) from the OSI layer model. Information on the OSI model can also be found on Wikipedia at: https://de.wikipedia.org/wiki/OSI-model . The parameters (including the MAC address) under which our module works in these two layers can be transmitted via the SPI bus as a configuration.

Detailed information about the configuration options of the module can be found in the data sheet at:
http://ww1.microchip.com/downloads/en/devicedoc/39662c.pdf

Layer 3 and all other following layers have to be mapped on our Arduino. Since the IP protocol is layer 3 and the network traffic most used TCP and UDP protocol is layer 4 (transport layer), we need to implement a custom TCP/IP stack on the Arduino. We will do that in the next step by including an appropriate library on our Arduino.

Even if the implementation of a separate TCP/IP stack in our host controller may seem quite complex at first and there are also network modules with integrated TCP/IP stack, the module offers us a very high degree of flexibility in terms of network communication design.

For example, the free definition of the package EtherTypenfeldes several packet types can be created or processed with the module. See: https://de.wikipedia.org/wiki/Ethernet#Das_Typ-Feld_(EtherType)

In our code example, we use IP packets.

Library installation

In order not to have to completely program a complex TCP/IP stack for our module, we use the free "*EtherCard*" library for our module. The "*EtherCard*" library can be installed in the Arduino Library Manager. Alternatively, it can be downloaded directly from GitHub.

To do this, we create a new subfolder named "*EtherCard_Library*" in the folder MyDocuments/Documents/ and copy the file EtherCardmaster.zip into the newly created folder. The file is previously downloaded from *GitHub* at the URL https://github.com/njh/EtherCard . (Click on "Clone or Download" on the website, then select Download Zip)

We then start our Arduino IDE and bind under Sketch > Include Library > Add ZIP. Library and insert our previously downloaded ZIP file.

Restart the Arduino IDE to activate the new "*EtherCard*" library.

First programming:

We now add the following Arduino code to our IDE:

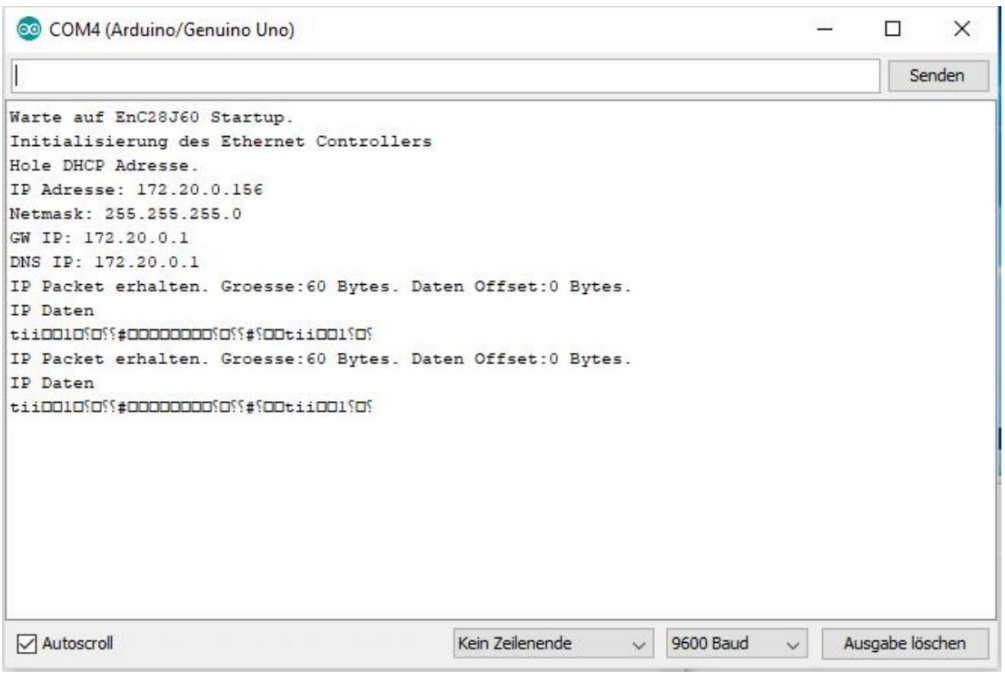
```
#include <EtherCard.h>
#define SS    10     // Slave Select Pin Nummer
uint8_t Ethernet::buffer[700]; // Packet Buffer Greetings is 512 Byte

// The Hardware MAC address is defined here..
byte mymac[] = { 0x74, 0x69, 0x69, 0x2D, 0x30, 0x31 };

static BufferFiller bfill; // used as cursor while filling the buffer
void setup() {
    Serial.begin(9600); // Open serial cutting atlas
    while(!Serial) { /* Wait for serial port */ }
    Serial.println("Waiting for EnC28J60 Startup.");
    delay(6000);
    Serial.println("Initialization of the Ethernet controller");
    if(ether.begin(sizeof Ethernet::buffer, mymac, SS) == 0) {
        Serial.println("Error: EnC28J60 not initialized.");
        while(true);
    }
    Serial.println("DHCP Address.");
    if(ether.dhcpSetup()) {
        ether.printIp("IP Adress: ", ether.myip);
        ether.printIp("Netmask: ", ether.netmask);
        ether.printIp("GW IP: ", ether.gwip);
        ether.printIp("DNS IP: ", ether.dnsip);
    }
    else {
        ether.printIp("Get DHCP address failed.");
        while(true);
    }
}
```

```
void loop() {
    word len = ether.packetReceive();  // Paket Listener
    word pos = ether.packetLoop(len);
    if(len) {
        Serial.print("Receive IP Packet. Size:");
        Serial.print(len);
        Serial.print(" Bytes. Data Offset:");
        Serial.print(pos);
        Serial.println(" Bytes. IP data:");
        for(int x = 0; x < len; x++) {
            char StrC = Ethernet::buffer[x];
            Serial.print(StrC);
        }
        Serial.println("");
    }
}
```

After uploading our program on the Arduino, we open the serial monitor in our IDE and get the following output if everything worked fine:



Our module has been assigned the IP address 172.20.0.156 by the DHCP server next to a gateway and a DNS server. (The addresses may differ in your network). We can now ping our module from a PC:

```
Ping wird ausgeführt für 172.20.0.156 mit 32 Bytes Daten:
Antwort von 172.20.0.156: Bytes=32 Zeit<1ms TTL=64
Antwort von 172.20.0.156: Bytes=32 Zeit<1ms TTL=64
Antwort von 172.20.0.156: Bytes=32 Zeit<1ms TTL=64
Antwort von 172.20.0.156: Bytes=32 Zeit<1ms TTL=64
Antwort von 172.20.0.156: Bytes=32 Zeit<1ms TTL=64

Ping-Statistik für 172.20.0.156:
    Pakete: Gesendet = 5, Empfangen = 5, Verloren = 0
    (0% Verlust),
Ca. Zeitangaben in Millisek.:
    Minimum = 0ms, Maximum = 0ms, Mittelwert = 0ms
```

We receive an answer from our module. Also in the serial console we see the ping as a data package in raw format:

```
IP Packet erhalten. Groesse:74 Bytes. Daten Offset:0 Bytes.
IP Daten
"□Z□□tii□□1□E<□+@@□□Q□□□
OK□□□abcdefghijklmnopqrstuvwabcdefghi
```

Interesting in this context is that a ping packet can receive data. Windows uses the repetitive string a-w as data in an ICMP ping packet.

You've done it, you can now use and program your module for your projects.

Now it is time to learn and make the Projects on your own. You can do that with the help of many example scripts and other tutorials, which you can find on the internet.

If you are looking for the high quality products for Arduino and Raspberry Pi, AZ-Delivery Vertriebs GmbH is the right company to get them from. You will be provided with numerous application examples, full installation guides, eBooks, libraries and assistance from our technical experts.

https://az-delivery.de

Have Fun!

Impressum

https://az-delivery.de/pages/about-us