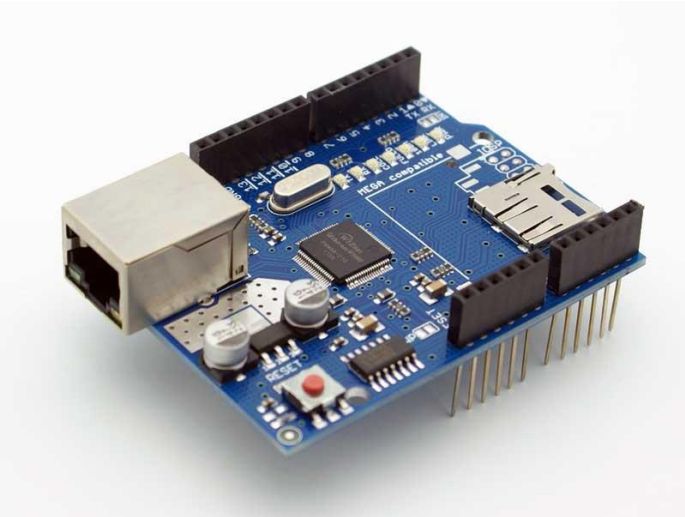
Arduino Ethernet Shield W5100 R3

1. POPIS



Arduino Ethernet Shield umožní vývojovým kitům Arduino Nano, Mega 1280/2560 nebo Duemilanove 168/328 připojení k internetu (nelze použít samostatně).

Základní charakteristika shieldu:

- Používá ethernetový čip Wiznet 5100, který umožňuje použití protokolů TCP i UDP. Pro programování je možné využít knihovnu „Ethernet library“.
- Připojení k síti přes standardní konektor RJ-45.
- Obsahuje resetovací tlačítko (reset kitu i shieldu).
- Obsahuje také slot pro MicroSD paměťovou kartu, který může být využit při přenosu dat po síti. Je kompatibilní se všemi Arduino/Genuino kity. Pro čtečku SD karet je přístupná knihovna „SD Library“.
- Obsahuje pět indikačních LED diod (TX, RX, 100M, LINK, PWR).

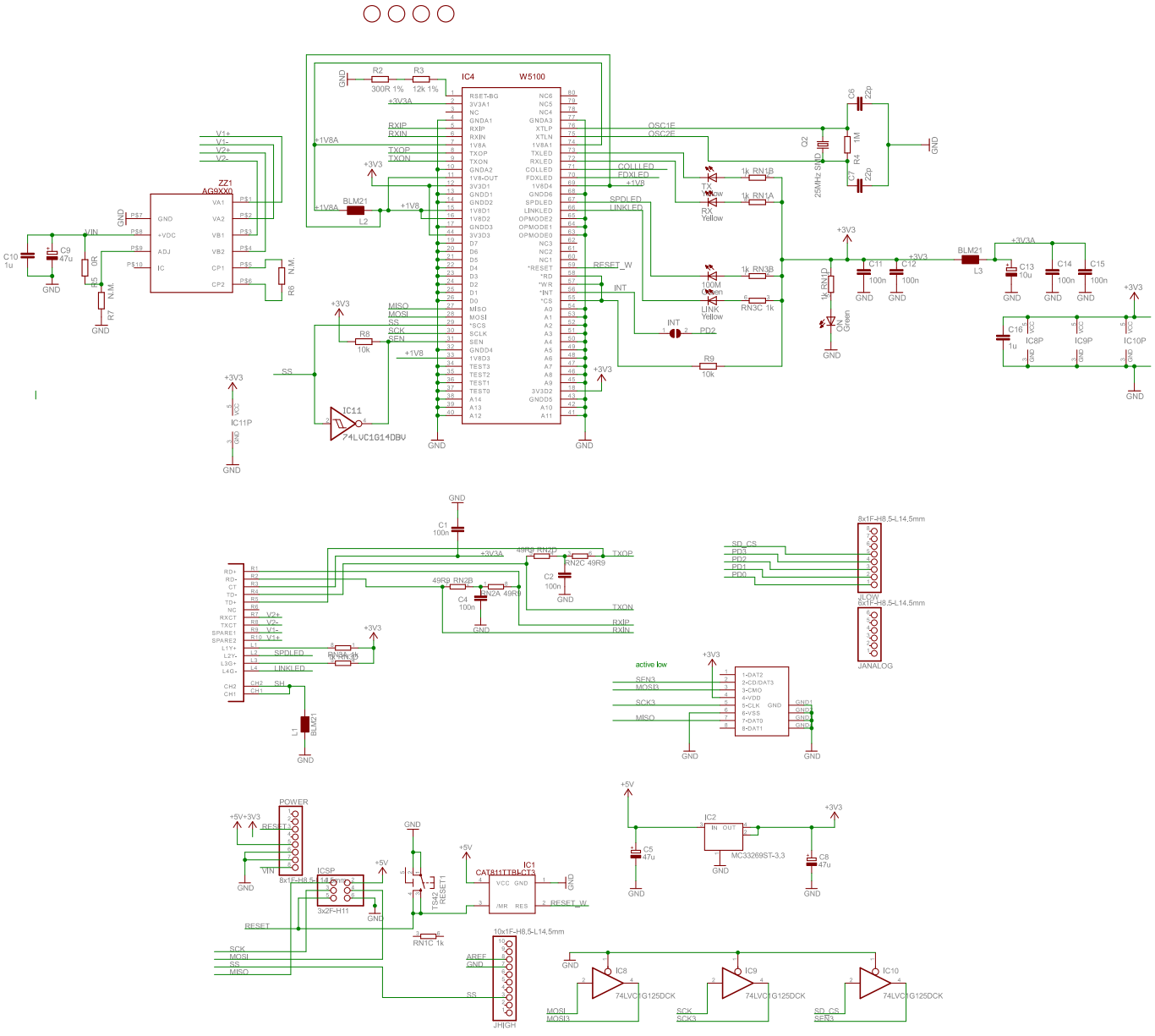
2. SPECIFIKACE ETHERNET SHIELDU

Čip	Wiznet 5100	**Rychlost připojení**	10/100 Mb
Pracovní napětí	5 V (napájeno z kitu)	**Protokoly**	TCP, UDP
Vnitřní buffer	16 KB	**Indikační LED**	5

3. ZAPOJENÍ A SCHÉMA

Tento shield nevyžaduje žádné externí zapojení, pouze jej vsuňte do vývojového kitu Arduino Uno, Mega nebo Duemilanove.

V případě, že po osazení shieldu nelze do vývojového kitu nahrát program, je příčina pravděpodobně ve špatném automatickém resetování při flashování mikroprocesoru. Řešením je zapojení (pull-down) rezistoru 10 kΩ na pin „RESET“ a pin „GND“.



4. UKÁZKA PROGRAMU – ETHERNET

Kód je obsažen ve vývojovém prostředí Arduino (Příklady -> ethernet -> dhcpaddressprinter)

```
/*
 DHCP-based IP printer

 This sketch uses the DHCP extensions to the Ethernet library
 to get an IP address via DHCP and print the address obtained.
 using an Arduino Wiznet Ethernet shield.

 Circuit:
  Ethernet shield attached to pins 10, 11, 12, 13

 created 12 April 2011
 modified 9 Apr 2012
 by Tom Igoe
 modified 02 Sept 2015
 by Arturo Guadalupi

*/

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

// Enter a MAC address for your controller below.
// Newer Ethernet shields have a MAC address printed on a sticker on the shield
byte mac[] = {
 0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x02
};

// Initialize the Ethernet client library
// with the IP address and port of the server
// that you want to connect to (port 80 is default for HTTP):
EthernetClient client;

void setup() {
 // Open serial communications and wait for port to open:
 Serial.begin(9600);
 // this check is only needed on the Leonardo:
 while (!Serial) {
  ; // wait for serial port to connect. Needed for native USB port only
 }

 // start the Ethernet connection:
 if (Ethernet.begin(mac) == 0) {
  Serial.println("Failed to configure Ethernet using DHCP");
  // no point in carrying on, so do nothing forevermore:
  for (;;)
   ;
 }
 // print your local IP address:
 printIPAddress();
}

void loop() {

 switch (Ethernet.maintain())
 {
```

```
  case 1:
    //renewed fail
    Serial.println("Error: renewed fail");
    break;

  case 2:
    //renewed success
    Serial.println("Renewed success");

    //print your local IP address:
    printIPAddress();
    break;

  case 3:
    //rebind fail
    Serial.println("Error: rebind fail");
    break;

  case 4:
    //rebind success
    Serial.println("Rebind success");

    //print your local IP address:
    printIPAddress();
    break;

  default:
    //nothing happened
    break;

 }
}

void printIPAddress()
{
 Serial.print("My IP address: ");
 for (byte thisByte = 0; thisByte < 4; thisByte++) {
  // print the value of each byte of the IP address:
  Serial.print(Ethernet.localIP()[thisByte], DEC);
  Serial.print(".");
 }

 Serial.println();
}
```

5. UKÁZKA PROGRAMU – SD KARTA

Před vložením SD karty do modulu je nutné ji zformátovat (FAT16 nebo FAT32). Kód je obsažen ve vývojovém prostředí Arduino (Příklady -> SD -> ReadWrite).

```
/*
 SD card read/write

 This example shows how to read and write data to and from an SD card file
 The circuit:
 * SD card attached to SPI bus as follows:
 ** MOSI - pin 11
 ** MISO - pin 12
```

```
 ** CLK - pin 13
 ** CS - pin 4

 created   Nov 2010
 by David A. Mellis
 modified 9 Apr 2012
 by Tom Igoe

 This example code is in the public domain.

 */

#include <SPI.h>
#include <SD.h>

File myFile;

void setup() {
 // 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.print("Initializing SD card...");

 if (!SD.begin(4)) {
   Serial.println("initialization failed!");
   return;
 }
 Serial.println("initialization done.");

 // open the file. note that only one file can be open at a time,
 // so you have to close this one before opening another.
 myFile = SD.open("test.txt", FILE_WRITE);

 // if the file opened okay, write to it:
 if (myFile) {
   Serial.print("Writing to test.txt...");
   myFile.println("testing 1, 2, 3.");
   // close the file:
   myFile.close();
   Serial.println("done.");
 } else {
   // if the file didn't open, print an error:
   Serial.println("error opening test.txt");
 }

 // re-open the file for reading:
 myFile = SD.open("test.txt");
 if (myFile) {
   Serial.println("test.txt:");

   // read from the file until there's nothing else in it:
   while (myFile.available()) {
     Serial.write(myFile.read());
   }
   // close the file:
   myFile.close();
 } else {
```

```
    // if the file didn't open, print an error:
    Serial.println("error opening test.txt");
  }
}

void loop() {
  // nothing happens after setup
}
```