# Tutorial

### ****I/O defines on Mega ETH****

For Ethernet W5500 Chipset:

* D10 — SS (CS1)
* D51 — MOSI
* D50 — MISO
* D52 — SCK
* D8 — INT (select D8 with W5500-INT pin jumper, pin solder jumper at bottom layer, default: not connected)
* D7/3.3V — RST (select D7 with W5500-RST pin solder jumper at the bottom layer, default: connected to 3.3V)

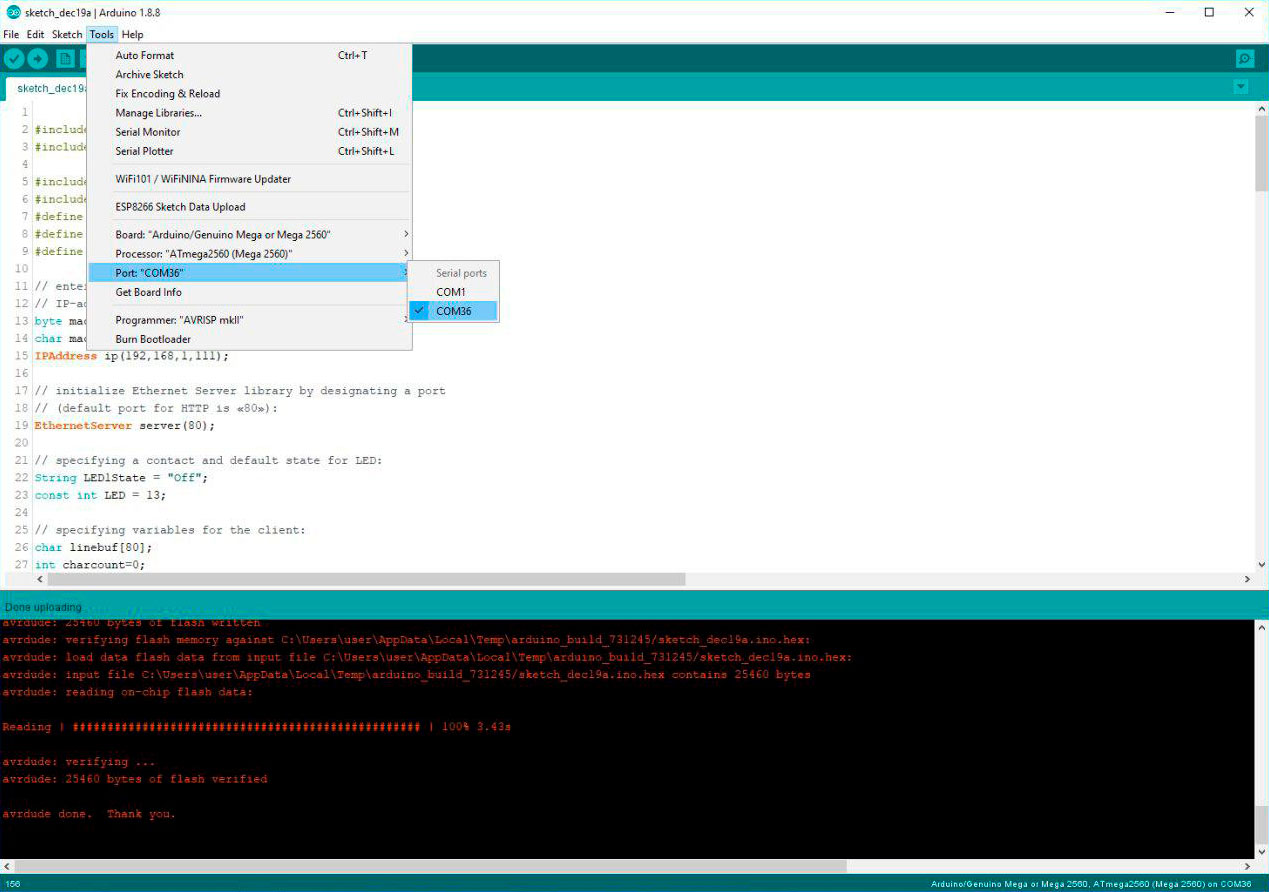
For SD card reader:

* D4 — CS(CS2)
* D9 – Card detect (pin solder jumper at bottom layer, default: not connected)
* D51 — MOSI/DI
* D50 — MISO/DO
* D52 — SCK/CLK

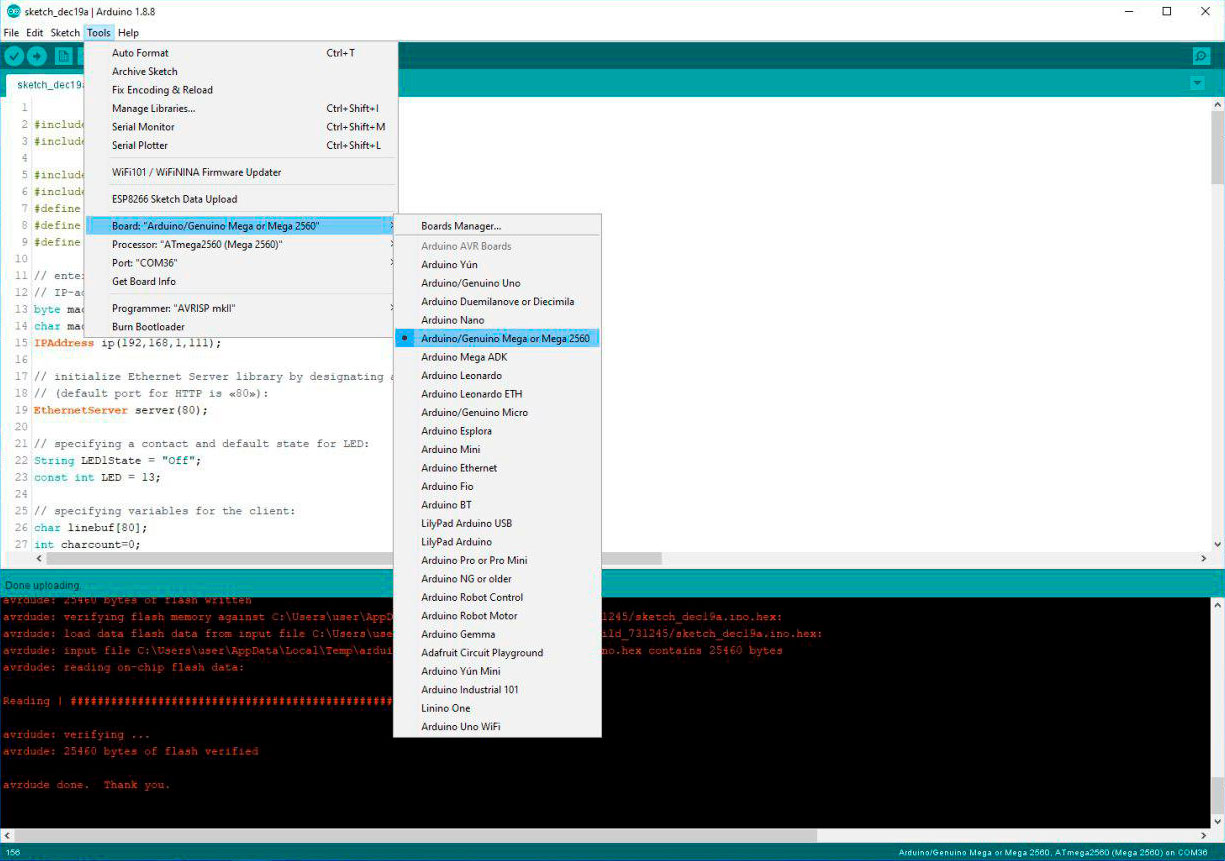
****NOTE:****Both W5500 and SD card communicate with ATmega2560 via SPI bus. Pin D10 and pin D4 are chip Selection pins for W5500 and SD slot. They cannot be used as general I/O.

### ****Steps for Arduino IDE:****

1. Select port:



1. Select board:



1. Please install the libraries:
   1. SD.h — [SD Library](https://www.arduino.cc/en/Reference/SD)
   2. SPI.h — [SPI library](https://www.arduino.cc/en/Reference/SPI)
   3. Ethernet.h/Ethernet2.h/ Ethernet3.h — Ethernet / Ethernet2 / Ethernet3 libraries

Copy below code into the sketch and then upload:

#include <Ethernet3.h>

#include <SPI.h>

#include <EEPROM.h>

#include <SD.h>

#define SS 10 //W5500 CS

#define RST 7 //W5500 RST

#define CS 4 //SD CS pin

// enter MAC-address and IP-address of your controller below;

// IP-address depends on your local network:

byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

char macstr[18];

IPAddress ip(192,168,1,111);

// initialize Ethernet Server library by designating a port

// (default port for HTTP is «80»):

EthernetServer server(80);

// specifying a contact and default state for LED:

String LED1State = "Off";

const int LED = 13;

// specifying variables for the client:

char linebuf[80];

int charcount=0;

void eeprom\_read()

{

if (EEPROM.read(1) == '#')

{

for (int i = 2; i < 6; i++)

{

mac[i] = EEPROM.read(i);

}

}

}

void eeprom\_write()

{

EEPROM.write(1, '#');

snprintf(macstr, 18, "%02x:%02x:%02x:%02x:%02x:%02x", mac[0], mac[1], mac[2], mac[3], mac[4], mac[5]);

}

void setup() {

// preparing LED-module:

pinMode(LED\_BUILTIN, OUTPUT);

digitalWrite(LED\_BUILTIN, HIGH);

pinMode(SS, OUTPUT);

pinMode(RST, OUTPUT);

pinMode(CS, OUTPUT);

digitalWrite(SS, LOW);

digitalWrite(CS, HIGH);

/\* If you want to control Reset function of W5500 Ethernet controller \*/

digitalWrite(RST,HIGH);

pinMode(LED, OUTPUT);

digitalWrite(LED, HIGH);

// opening a sequential communication with 9600 baud speed:

Serial.begin(9600);

eeprom\_read();

eeprom\_write();

// initialising Ethernet-communication and server:

Ethernet.begin(mac, ip);

server.begin();

Serial.print("server is at "); // "server at "

Serial.println(Ethernet.localIP());

Serial.println(Ethernet.macAddressReport());

}

// Displaying a webpage with a «ON/OFF» button for LED:

void dashboardPage(EthernetClient &client) {

client.println("<!DOCTYPE HTML><html><head>");

client.println("<meta name=\"viewport\" content=\"width=device-width, initial-scale=1\"></head><body>");

client.println("<h3>Arduino Web Server - <a href=\"/\">Refresh</a></h3>");

client.println("<h3>local IP<h3>");

client.println(Ethernet.localIP());

client.println("<h3>");

client.println("<h3>mac Address<h3>");

client.println(Ethernet.macAddressReport());

client.println("<h3>");

client.println("<h3>TEXT<h3>");

client.println("<section id=\"contact\"><div class=\"content\"><div id=\"form\"><form action=""id=\"contactForm\"method=\"GET\"><textarea class=\"message\"placeholder=\"Enter your message\"tabindex=4></textarea><input type=\"submit\"name=\"submit\"value=\"Send to Serial\"class=\"submit\"tabindex=5></form></div></section>");

// generating a button to control LED:

client.println("<h4>LED 13 - State: " + LED1State + "</h4>");

// if LED is off, Displaying an «ON» button:

if(LED1State == "Off"){

client.println("<a href=\"/LED13on\"><button>ON</button></a>");

}

// if LED is on, Displaying an «OFF» button:

else if(LED1State == "On"){

client.println("<a href=\"/LED13off\"><button>OFF</button></a>");

}

client.println("</body></html>");

}

void loop() {

// reading the incoming clients:

EthernetClient client = server.available();

if (client) {

//Serial.print (client.read());

//Serial.println("new client"); // "new client"

memset(linebuf,0,sizeof(linebuf));

charcount=0;

// HTTP-request is ending with blank line:

boolean currentLineIsBlank = true;

while (client.connected()) {

if (client.available()) {

char c = client.read();

// reading a HTTP-request, one symbol at a time:

linebuf[charcount]=c;

if (charcount<sizeof(linebuf)-1) charcount++;

// if you reached the end of the line (i.e. if you recieved

// symbol form a new line), it means that

// HTTP-request is completed, and you can send the answer:

if (c == '\n' && currentLineIsBlank) {

dashboardPage(client);

break;

}

if (c == '\n') {

if (strstr(linebuf,"GET /id=") > 0)Serial.println(linebuf);

if (strstr(linebuf,"GET /LED13off") > 0){

digitalWrite(LED, HIGH);

LED1State = "Off";

}

else if (strstr(linebuf,"GET /LED13on") > 0){

digitalWrite(LED, LOW);

LED1State = "On";

}

// if you recieved a symbol form a new line

currentLineIsBlank = true;

memset(linebuf,0,sizeof(linebuf));

charcount=0;

}

else if (c != '\r') {

// if you recieved any other symbol

currentLineIsBlank = false;

}

}

}

// providing a time for a borwser to recieve the data:

delay(1);

// closing the connection:

client.stop();

//Serial.println("client disonnected"); // "Client is disconnected"

}

}

### ****Result****

