MicroMod Ethernet Function Board - W5500 Hookup Guide

Introduction

Integrate your MicroMod project into an Ethernet network including Power-over-Ethernet with the SparkFun MicroMod Ethernet Function Board - W5500. This Function Board uses the W5500 Ethernet control module from WIZnet and a DC/DC converter to configure a MicroMod assembly as a connected and powered device into an Ethernet network with Power-over-Ethernet (PoE) capabilities.

SparkFun MicroMod Ethernet Function Board - W5500
COM-18708

The W5500 is a TCI/IP embedded Ethernet controller from WIZnet that uses SPI and supports up to 80 MHz speeds. We designed this Function Board to use the IEEE802.3af Alternative B power scheme which uses the spare pairs for power delivery, isolated from the data pairs. In this guide we'll highlight the capabilities of the W5500 and demonstrate how to use the MicroMod Ethernet Function Board to create an Ethernet network that can also be used for PoE.

Required Materials

Following along with this tutorial requires a few items along with the MicroMod Ethernet Function Board. Depending on what you already have, you may not need all of the items listed below.

MicroMod Processor

All MicroMod systems require a Processor board to operate. SparkFun carries a variety of Processor boards suited for different applications. Select the Processor board that best suits your Ethernet projects' needs:

SparkFun MicroMod STM32 Processor
DEV-17713

SparkFun MicroMod SAMD51 Processor
DEV-16791

SparkFun MicroMod ESP32 Processor
WRL-16781

SparkFun MicroMod Teensy Processor
DEV-16402

Note: Currently the Artemis and nRF52840 do not have built in Ethernet libraries in their Arduino cores. An external Arduino Ethernet library may work but at this time Ethernet is not supported on those Processors.

MicroMod Main Board

MicroMod Function Boards require at least one Main Board to work.

SparkFun MicroMod Main Board - Double
DEV-18576

SparkFun MicroMod Main Board - Single
DEV-18575

Peripheral Items

You'll also need a PoE power supply like a network hub or router, Ethernet cable, network hub/router or endpoint as well as a few other peripheral items to get your MicroMod Ethernet system up and running. If needed, add these items to your cart:

USB 3.1 Cable A to C - 3 Foot
CAB-14743

Reversible USB A to C Cable - 2m
CAB-15424

CAT 6 Cable - 3ft
CAB-08915

Tools

Assembling MicroMod systems requires a Phillips head screwdriver.

Pocket Screwdriver Set
TOL-12891

SparkFun Mini Screwdriver
TOL-09146

Recommended Reading

The MicroMod ecosystem is a unique way to allow users to customize their project to their needs. If you aren't familiar with the MicroMod system, click on the banner below for more information.

MicroMod

You may also want to read the tutorials below if you are not familiar with the concepts covered in them:

Serial Peripheral Interface (SPI)

SPI is commonly used to connect microcontrollers to peripherals such as sensors, shift registers, and SD cards.

What is an Arduino?

What is this 'Arduino' thing anyway? This tutorials dives into what an Arduino is and along with Arduino projects and widgets.

Installing Arduino IDE

A step-by-step guide to installing and testing the Arduino software on Windows, Mac, and Linux.

Getting Started with MicroMod

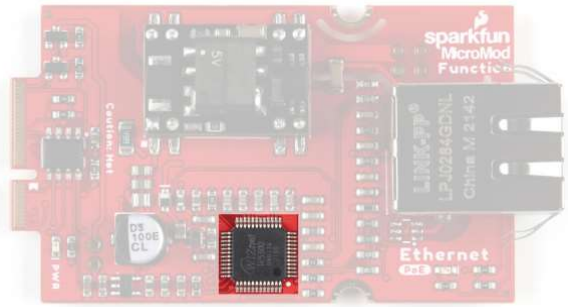
Dive into the world of MicroMod - a compact interface to connect a microcontroller to various peripherals via the M.2 Connector!

Hardware Overview

Let's take a closer look at the hardware on the Ethernet Function Board - W5500 and how it interacts with the rest of the MicroMod ecosystem.

WIZnet W5500 Ethernet Controller

The W5500 Ethernet Controller from WIZnet is a TCP/IP embedded Ethernet controller that uses SPI communication protocol to allow up to eight independent sockets operate simultaneously.



The W5500 supports the following hardwired TCP/IP protocols:

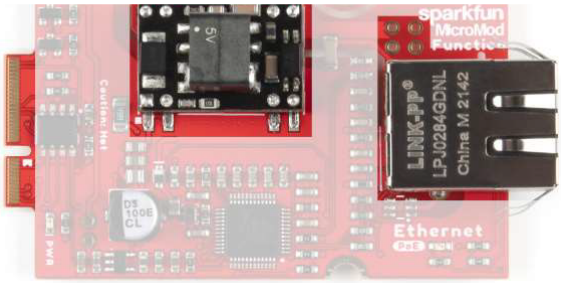
- TCP
- UDP
- ICMP
- IPv4
- ARP
- IGMP
- PPPoE

The W5500 SPI interface operates at up to 80MHz and supports fast SPI for high speed Ethernet communication. The chip also includes a Wake on Lan (WOL) operation and power down mode to help conserve power. The W5500 operates at **3.3V** but has **5V**-tolerant I/O. For detailed information on the W5500, refer to the datasheet.

The Function Board includes three solder jumpers connected to the three network mode selection pins to allow users to configure the W5500 network operation mode. By default, the board sets the W5500 to operates in 10/100Base-T with Auto-Negotiation enabled. Read on to the Solder Jumpers portion of this section or refer to the pin descriptions in the datasheet for more information on adjusting the operation mode.

Power

The Ethernet Function Board - W5500 features several power input options including PoE and USB (via the Main Board). By default, the board acts as a PoE Powered Device (PD), receiving voltage over the Ethernet connection using the IEEE802.3af Alternative B power scheme.



The Alternate B power scheme uses the spare pairs (pins 4/5 and 7/8) in the Ethernet cable for positive and negative DC voltage, keeping things simple if any troubleshooting is needed. The board includes a pair of PoE isolation jumpers that allows users to isolate these pins from the DC/DC converter input. When opened, the MicroMod assembly can receive power over USB, LiPo battery or through the DC/DC converter input PTH pins. The USB and LiPo power inputs are isolated from the PoE circuit.

DC/DC Converter Circuit

The board uses an Ag9905M Power-over-Ethernet (PoE) DC/DC converter to provide **5V** from an Ethernet connection. For complete information on the Ag9905M, refer to the datasheet.

The Ag9905 accepts an input voltage between **36V** and **57V** but a voltage of **48V** or greater is recommended on initial powerup to ensure the module functions properly. After power up, input voltage can be reduced to **36V** if needed.

The Ag9905 provides 9 Watts of power for the MicroMod system and any peripheral devices connected to it (i.e. Qwiic breakouts, etc.). The **5V** output from the converter is filtered to reduce noise for **5V** circuits on the board and is also regulated down to **3.3V**.

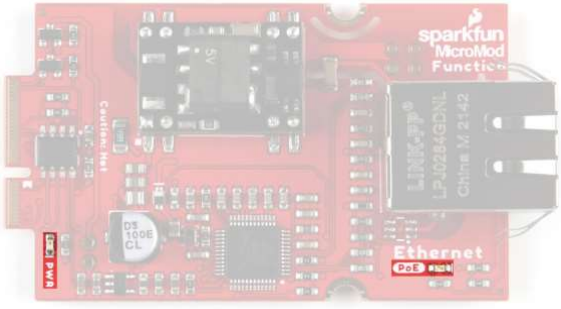
RJ45 Connector

The RJ45 connector on this function board includes embedded magnetics for PoE applications and is MagJack®-Compatible.

The Function Board uses all pairs on the RJ45 connector by default as the PoE configuration uses the spare pair for power inputs. Users who do not wish to use PoE can isolate these pins for other use by opening the PoE Isolation Jumpers. Reminder, if the PoE pairs are disconnected, power must be supplied from another source, either USB, LiPo battery or via the DC/DC converter input PTH pins on the other side of the PoE Isolation Jumpers.

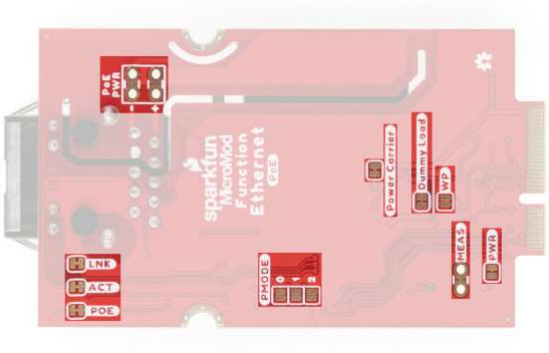
LEDs

The board has a pair of status LEDs indicating general power and PoE power as well as the pair of LEDs on the RJ45 connector for Link and Activity statuses.



Solder Jumpers

There are thirteen solder jumpers on the Function Board. The table below outlines their labels and functions:



Having trouble seeing the detail in this image? Click on it for a larger view.

Label	**Default State**	**Function**	**Notes**
PWR	CLOSED	Completes Power LED circuit.	Open to disable Power LED.
MEAS	CLOSED	Completes VCC circuit tying VCC to **5V** input.	Open to measure current drawn by the Function Board.
WP	OPEN	Pulls EEPROM Write Protect pin to **3.3V**/HIGH	Close to pull this pin to **0V**/LOW to disable write protect.
Dummy Load	CLOSED	Creates a dummy load of **100mA** on the DC/DC converter output.	Open to disable the dummy load.
Power Carrier	CLOSED	Selects power configuration (PoE or USB).	Open to disable PoE for VCC_In.
LNK	CLOSED	Completes the RJ45 Link LED circuit.	Open to disable the Link LED.
ACT	CLOSED	Completes the RJ45 Activity LED circuit.	Open to disable the Activity LED.
POE	CLOSED	Completes the PoE Status LED circuit.	Open to disable the PoE Status LED.
POE Power (Pair)	CLOSED	Ties the PoE +/- pins to DC/DC converter input.	Open both to isolate Ethernet pairs used for PoE from the DC/DC converter input.
PMODE0	OPEN	Ties PMODE0 to **3.3V**/HIGH.	Adjust this in tandem with the other two PMODE jumpers to switch network modes on the W5500.[1]
PMODE1	OPEN	Ties PMODE1 to **3.3V**/HIGH.	Adjust this in tandem with the other two PMODE jumpers to switch network modes on the W5500.[1]

PMODE2	OPEN	Ties PMODE2 to **3.3V**/HIGH.	Adjust this in tandem with the other two PMODE jumpers to switch network modes on the W5500.[1]

1: Refer to the pin description table in section 1.1 of the datasheet for a detailed overview of setting the network mode.

MicroMod Function Board Pinout

This Function Board uses the following pins on a connected Processor Board:

- 3.3V & VCC
- Power Enable
- SPI - W5500 Communication
- I^2C - EEPROM Communication
- G0 - W5500 Interrupt
- G1 - W5500 Chip Select (SPI)
- G2 - W5500 Reset

For the complete MicroMod Pinout and pins used by this function board, take a look at the tables below:

MICROMOD ETHERNET FUNCTION BOARD - W5500 PINOUT TABLE

MICROMOD GENERAL PROCESSOR PINOUT TABLE

MICROMOD GENERAL PIN DESCRIPTIONS

AUDIO	UART	GPIO/BUS	I^2C	SDIO	SPI0	Dedicated

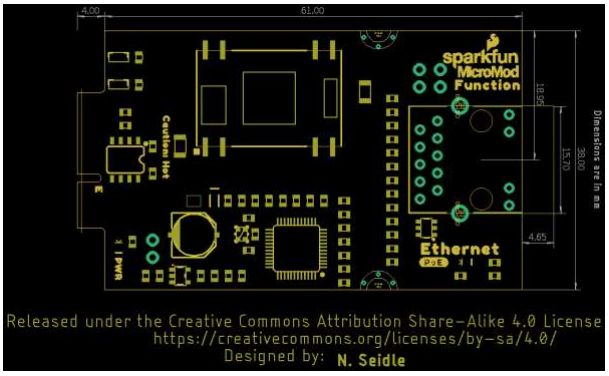
Description	Function	Bottom Pin	Top Pin	Function	Description
	(Not Connected)		**75**	GND	
	-	**74**	**73**	3.3V	Power Supply: 3.3-6V
	-	**72**	**71**	Power EN	Power Enable
	-	**70**	**69**	-	
	-	**66**	**65**	-	
	-	**64**	**63**	-	

	-	**62**	**61**	-	
	-	**60**	**59**	-	
	-	**58**	**57**	-	
	-	**56**	**55**	-	.
	-	**54**	**53**	-	
	-	**52**	**51**	ETH_RST	W5500 Reset.
	-	**50**	**49**	ETH_CS	W5500 Chip Select
	-	**48**	**47**	ETH_INT	W550 Interrupt Pin
	-	**46**	**45**	GND	
	-	**44**	**43**	-	
	-	**42**	**41**	-	
Write protection pin for the EEPROM. Pull low to enable.	EEPROM_WP	**40**	**39**	GND	
	-	**38**	**37**	-	
EEPROM I^2C address configuration.	EEPROM_A0	**36**	**35**	-	
EEPROM I^2C address configuration.	EEPROM_A1	**34**	**33**	GND	
EEPROM I^2C address configuration.	EEPROM_A2	**32**	**31**	Module Key	
	Module Key	**30**	**29**	Module Key	
	Module Key	**28**	**27**	Module Key	
	Module Key	**26**	**25**	Module Key	
	Module Key	**24**	**23**	-	
	-	**22**	**21**	I2C_SCL	I^2C - Clock signal for EEPROM
	-	**20**	**19**	I2C_SDA	I^2C - Data signal for EEPROM

	-	**18**	**17**	-	
	-	**16**	**15**	-	
	-	**14**	**13**	-	
	-	**12**	**11**	-	
	-	**10**	**9**	-	
	-	**8**	**7**	POCI	SPI Peripheral Output/Controller Input.
	-	**6**	**5**	PICO	SPI Peripheral Input/Controller Output.
	-	**4**	**3**	SCK	SPI Clock Signal
	-	**2**	**1**	GND	

Board Dimensions

The MicroMod Ethernet Function Board matches the MicroMod Function Board standard and measures 1.50" x 2.56" (38.1mm x 65.024mm).



Hardware Assembly

If you're not familiar with assembling boards using the MicroMod connection system, head over to the Getting Started with MicroMod tutorial for information on inserting and securing your MicroMod Processor and Function Boards to the Main Board:

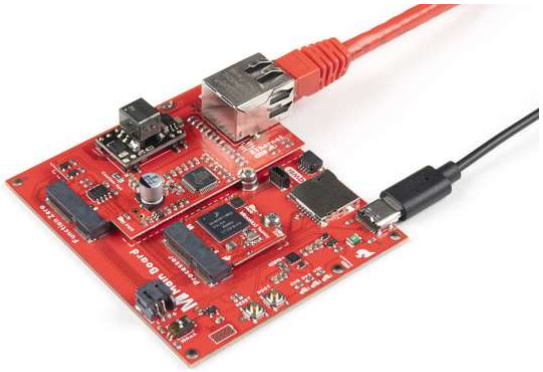
Getting Started with MicroMod

OCTOBER 21, 2020

Dive into the world of MicroMod - a compact interface to connect a microcontroller to various peripherals via the M.2 Connector!

PoE Assembly

After securing the Processor and Function Board to the Main Board, connect the MicroMod Main Board to your computer with a USB-C cable to program the Processor. Once programmed, connect the Function Board to your Ethernet hub (router, network switch, etc.) with an Ethernet cable connected to the RJ45 connector.



Software Installation

Note: The Ethernet Arduino example assumes you are using the latest version of the Arduino IDE on your desktop. If this is your first time using Arduino, please review the following tutorials.

- Installing the Arduino IDE
- Installing Board Definitions in the Arduino IDE
- Installing an Arduino Library

Processor Arduino Board Definitions and Driver

Make sure you go through the Hookup Guide for your chosen Processor Board to install the Arduino board definitions and any necessary drivers:

MicroMod ESP32 Processor Board Hookup Guide

OCTOBER 21, 2020

A short hookup guide to get started with the SparkFun MicroMod ESP32 Processor Board.

MicroMod STM32 Processor Hookup Guide

MAY 13, 2021

Get started with the MicroMod Ecosystem and the STM32 Processor Board!

MicroMod Teensy Processor Hookup Guide

JULY 1, 2021

Add the processing power and versatility of the Teensy to your MicroMod project following this guide for the SparkFun MicroMod Teensy Processor.

Note on Artemis and nRF52840 Processors: Currently the Artemis and nRF52840 do not have built in Ethernet libraries in their Arduino cores. An external Arduino Ethernet library may work but at this time Ethernet is not supported on those Processors. We will update the tutorial in the future if Ethernet support is added to these cores.

Main Board Example - Pin Connection Table

The table below helps show what pins the Function Board connects to depending on the slot it is connected to on a Main Board (Note: The Single Main Board connection is Slot 0):

AUDIO	UART	GPIO/BUS	I^2C	SDIO	SPI0	Dedicated

Function Board Pin Name	I/O Direction	Main Board's Processor Pin	
		Slot 0	Slot 1
VCC	Input	-	
3.3V	Input	-	
GND	-	-	
ETH_INT		D0	D1
CS		CS0	CS1
ETH_RST		PWM0	PWM1

Arduino Example

Now that our MicroMod system is fully assembled, we'll use a slightly modified version of the "Ethernet - Web Client" example included with Arduino to make sure everything is connected and working properly. This modified version simply updates the options for the SPI Chip Select pin to match those used on supported MicroMod Processor Boards:

```
void setup() {
  //Ethernet.init(CS); //SAMD51
  Ethernet.init(10); //Teensy
  //Ethernet.init(5); //ESP32
  //Ethernet.init(A4); //STM32
```

Note: The code defaults to use the CS pin used with the Function Board connected in Slot 0. If the board is in Slot 1, switch to the CS1 pin on your Processor.

Open the Arduino IDE and either copy the code below or navigate to the example by going to **File > Examples > Ethernet > Web Client**. Reminder, if you open the default example in Arduino, make sure to set the correct Chip Select pin for the Processor Board used as listed above. Upload the code and open the serial monitor with the baud set to **115200** and you should see the example initialize the W5500 and ping Google.com.

```
/*
  Web client
 This sketch connects to a website (http://www.google.com)
 using an Arduino Wiznet Ethernet shield.
 Circuit:
 * Ethernet shield attached to pins 10, 11, 12, 13
 created 18 Dec 2009
 by David A. Mellis
 modified 9 Apr 2012
 by Tom Igoe, based on work by Adrian McEwen
 */

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

// if you don't want to use DNS (and reduce your sketch size)
// use the numeric IP instead of the name for the server:
//IPAddress server(74,125,232,128);  // numeric IP for Google (no DNS)
char server[] = "www.google.com";    // name address for Google (using DNS)

// Set the static IP address to use if the DHCP fails to assign
IPAddress ip(192, 168, 0, 177);
IPAddress myDns(192, 168, 0, 1);

// 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;

// Variables to measure the speed
unsigned long beginMicros, endMicros;
unsigned long byteCount = 0;
bool printWebData = false;  // set to false for better speed measurement

void setup() {
  //Ethernet.init(CS); //SAMD51
  Ethernet.init(10); //Teensy
  //Ethernet.init(5); //ESP32
  //Ethernet.init(A4); //STM32

  Serial.begin(115200);

  delay(4000);

  // start the Ethernet connection:
  Serial.println("Initialize Ethernet with DHCP:");
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
```

```
    // 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(1000);
        Serial.println("Not detected");
        delay(1); // do nothing, no point running without Ethernet hardware
      }
    }
    if (Ethernet.linkStatus() == LinkOFF) {
      Serial.println("Ethernet cable is not connected.");
    }
    // try to congifure using IP address instead of DHCP:
    Ethernet.begin(mac, ip, myDns);
  } else {
    Serial.print("  DHCP assigned IP ");
    Serial.println(Ethernet.localIP());
  }
  // give the Ethernet shield a second to initialize:
  delay(1000);
  Serial.print("connecting to ");
  Serial.print(server);
  Serial.println("...");

  // if you get a connection, report back via serial:
  if (client.connect(server, 80)) {
    Serial.print("connected to ");
    Serial.println(client.remoteIP());
    // 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 {
    // if you didn't get a connection to the server:
    Serial.println("connection failed");
  }
  beginMicros = micros();
}

void loop() {
  // if there are incoming bytes available
  // from the server, read them and print them:
  int len = client.available();
  if (len > 0) {
//    byte buffer[80];
    byte buffer[512 * 4];
    if (len > sizeof(buffer)) len = sizeof(buffer);
    client.read(buffer, len);
    if (printWebData) {
      Serial.write(buffer, len); // show in the serial monitor (slows some boards)
    }
```

```
    byteCount = byteCount + len;
  }

  // if the server's disconnected, stop the client:
  if (!client.connected()) {
    endMicros = micros();
    Serial.println();
    Serial.println("disconnecting.");
    client.stop();
    Serial.print("Received ");
    Serial.print(byteCount);
    Serial.print(" bytes in ");
    float seconds = (float)(endMicros - beginMicros) / 1000000.0;
    Serial.print(seconds, 4);
    float rate = (float)byteCount / seconds / 1000.0;
    Serial.print(", rate = ");
    Serial.print(rate);
    Serial.print(" kbytes/second");
    Serial.println();

    // do nothing forevermore:
    while (true) {
      delay(1);
    }
  }
}
```

Troubleshooting

Artemis and nRF52840 Processor Support

Reminder, the Artemis and nRF52840 currently do not have built in Ethernet libraries in their Arduino cores. An external Arduino Ethernet library may work with these Processors but at this time Ethernet is not supported. We will update the tutorial in the future if Ethernet support is added to these cores.

General Troubleshooting

Not working as expected and need help?

If you need technical assistance and more information on a product that is not working as you expected, we recommend heading on over to the SparkFun Technical Assistance page for some initial troubleshooting.

SPARKFUN TECHNICAL ASSISTANCE PAGE

If you don't find what you need there, the SparkFun Forums: MicroMod are a great place to find and ask for help. If this is your first visit, you'll need to create a Forum Account to search product forums and post questions.

SPARKFUN FORUMS: MICROMOD

Resources and Going Further

That'll a wrap for this tutorial. By now your MicroMod Ethernet network should be up and running. Take a look at the resources below for more information about the MicroMod Ethernet Function Board - WIZnet W5500:

- Schematic (PDF)
- Eagle Files (ZIP)
- Board Dimensions (PNG)
- W5500 Datasheet
- Ag9900M Datasheet (Ag9905M is used)
- RJ45 Connector Datasheet
- Hardware GitHub Repository
- MicroMod Landing Page

For more information about the SparkFun MicroMod system, take a look here: