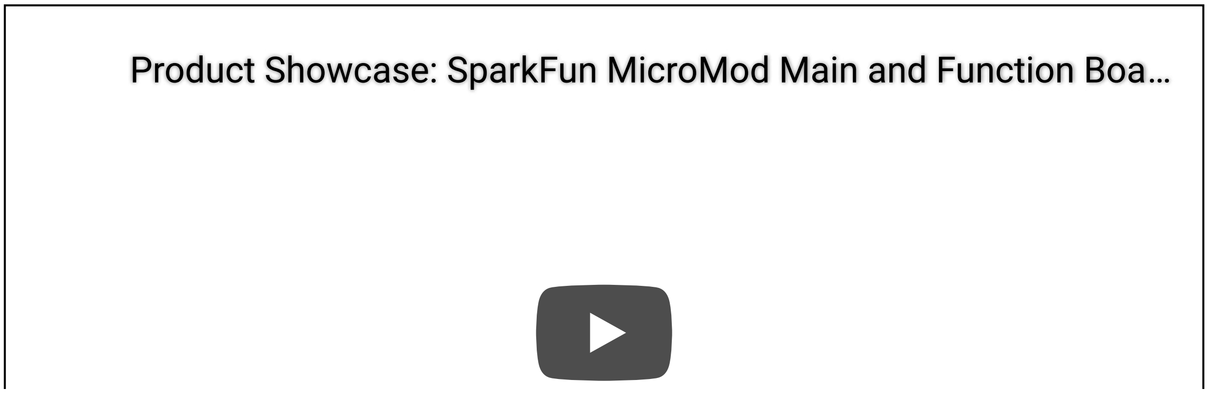
MicroMod WiFi Function Board - DA16200 Hookup Guide

Introduction

Want to add all the IoT functionality of the DA16200 to the adaptability of the MicroMod system? Then you're in luck! The SparkFun MicroMod DA16200 Function Board adds a fully integrated WiFi module with a 40MHz crystal oscillator, 32.768KHz RTC clock, RF Lumped RF filter, 4MB flash memory, and an on-board chip antenna to any MicroMod project. With the addition of JTAG connectors for deep dive programming, you've got everything you need to get your MicroMod setup ready for your next IoT project.

SparkFun MicroMod WiFi Function Board - DA16200
◉ WRL-18594



Product Showcase: SparkFun MicroMod Main and Function Boa...

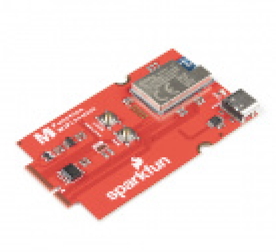
Required Materials

To follow along with this tutorial, you will need the following materials at a minimum. You may not need everything though depending on what you have. Add it to your cart, read through the guide, and adjust the cart as necessary.

SparkFun MicroMod Main Board - Single
◉ DEV-18575

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

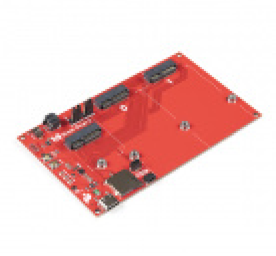
SparkFun MicroMod Artemis Processor
◉ DEV-16401



SparkFun MicroMod WiFi Function Board - DA16200
◉ WRL-18594

MicroMod Main Board

To hold the processor board and function board, you will need one Main board. Depending on your application, you may choose to have one or two additional function boards.

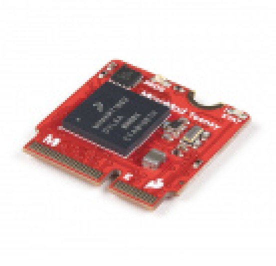


SparkFun MicroMod Main Board - Single
◉ DEV-18575

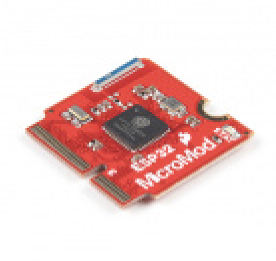
SparkFun MicroMod Main Board - Double
◉ DEV-18576

MicroMod Processor Board

There are a variety of MicroMod Processor Boards available to choose from.



SparkFun MicroMod Teensy Processor
◉ DEV-16402



SparkFun MicroMod ESP32 Processor
○ WRL-16781

SparkFun MicroMod nRF52840 Processor
◉ WRL-16984

SparkFun MicroMod RP2040 Processor
○ DEV-17720

Tools

You will need a screw driver to secure the Processor and Function boards.

SparkFun Mini Screwdriver
◉ TOL-09146

Suggested Reading

If you aren’t familiar with the following concepts, we recommend checking out these tutorials before continuing.

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!

MicroMod Artemis Processor Board Hookup Guide

Get started with the Artemis MicroMod Processor Board in this tutorial!

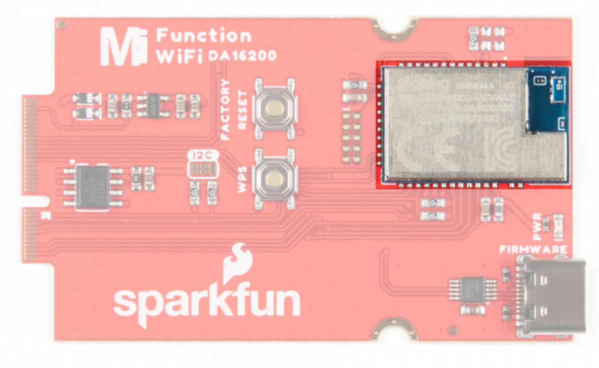
Designing with MicroMod

This tutorial will walk you through the specs of the MicroMod processor and carrier board as well as the basics of incorporating the MicroMod form factor into your own PCB designs!

Hardware Overview

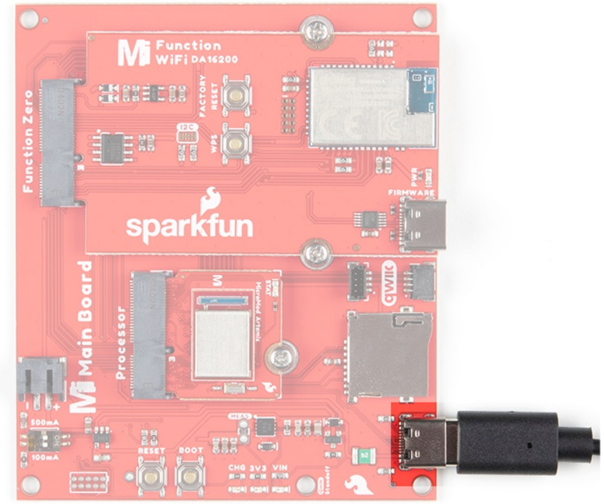
DA16200

The MicroMod WiFi Function Board includes the DA16200 module from Dialog with AT command firmware. This chip is a fully integrated Wi-Fi® module with ultra-low power consumption, 40 MHz crystal oscillator, 32.768 KHz RTC clock, RF Lumped RF filter, 4 M-byte flash memory, and an onboard chip antenna. For more information, refer to the datasheet.



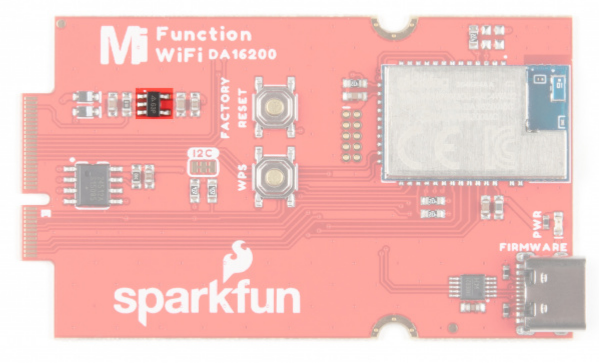
Power

To power the board, you will need to apply power to a SparkFun Main Board; either a Single or a Double. Power applied will connect to the Function Board's VIN pin, which will be regulated down for the rest of the board with the AP2112 3.3V/600mA voltage regulator.



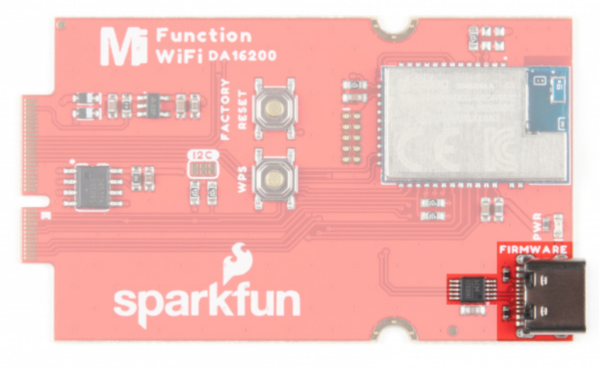
AP2112

Operating voltage for the DA16200 is **3.3 V**. The AP2112 can handle a voltage range of 2.5-6V with a 600mA output and ensures appropriate voltage to the WiFi unit.



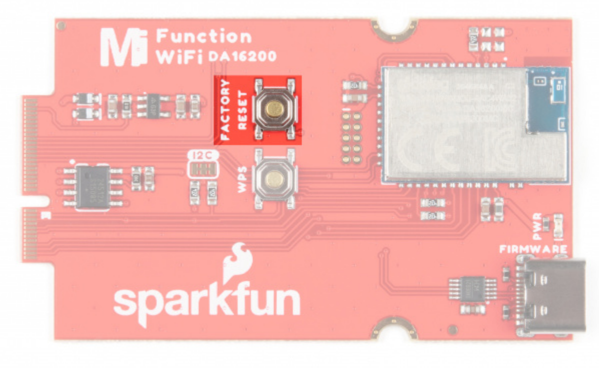
CH340

The board is populated with the CH340E USB-to-Serial converter to update firmware on the DA16200 through its USB Type C connector. You will need a Main Board (for power) and a second USB cable to update the firmware.



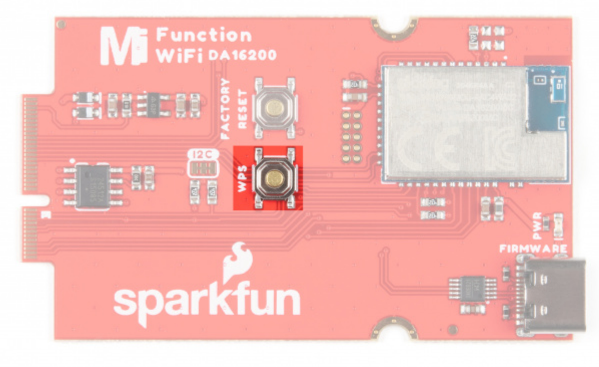
Reset Button

The reset button allows users to reset the program running on the Artemis module without unplugging the board.



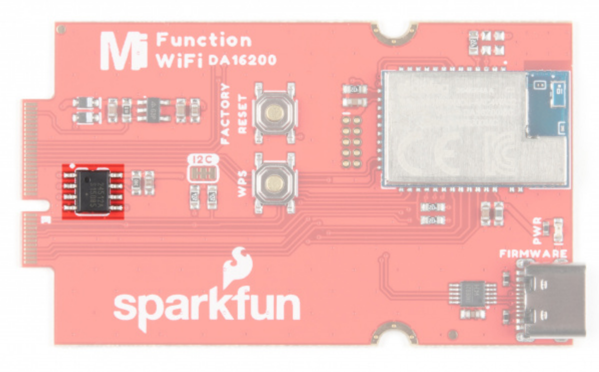
WPS Button

The Wi-Fi Protected Setup Button allows you to easily and quickly connect to a WiFi network.



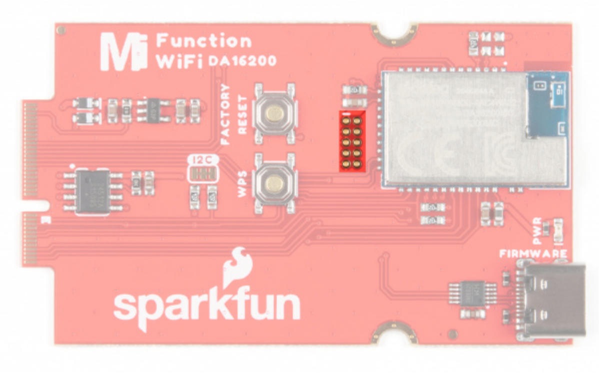
EEPROM

The board includes an I^{2}C EEPROM. To enable writing to the EEPROM, pull WP low, either through the EEPROM_WP pin or by closing JP3.



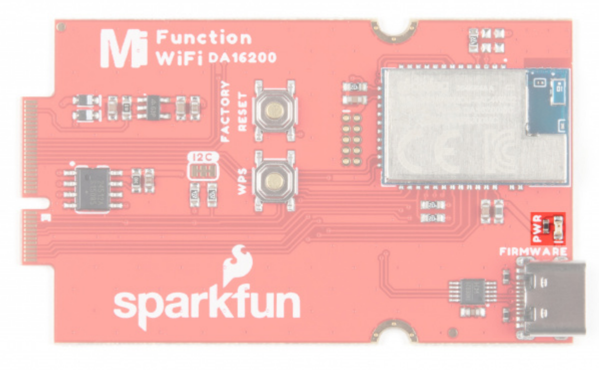
JTAG

An unpopulated JTAG footprint is available for more advanced users who need breakpoint level debugging. We recommend checking out our JTAG section for the compatible male header and a compatible JTAG programmer and debugger.



LED

There is only one LED available which is the Power LED. The LED lights up to indicate availability for the DA16200 from the 3.3V voltage regulator. You can disable it by cutting the jumper on the back of the board (see Jumpers below).

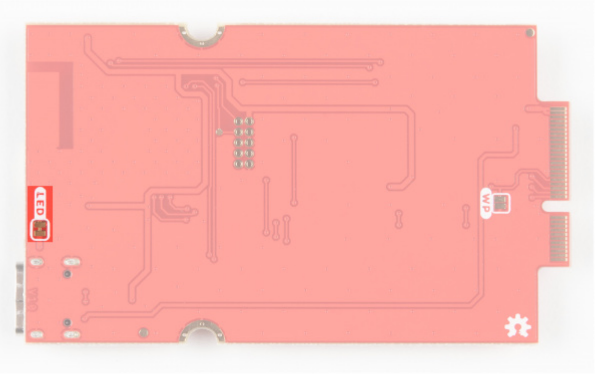


Jumpers

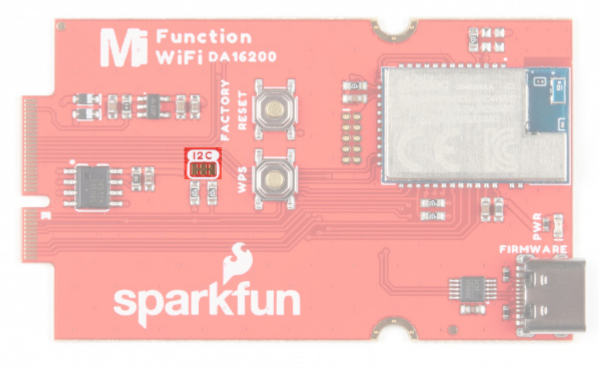
Note: If this is your first time working with jumpers, check out the How to Work with Jumper Pads and PCB Traces tutorial for more information.

The following jumpers are included to configure the board.

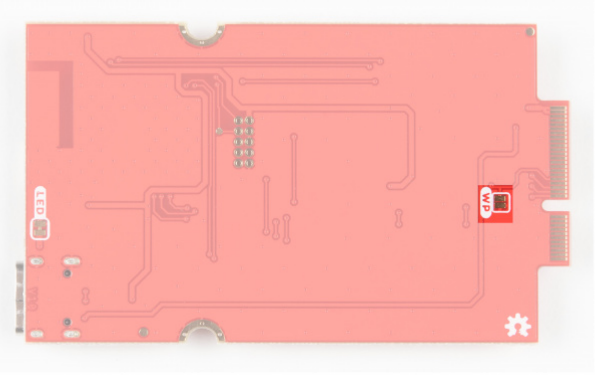
- **PWR** - By default, the jumper with the label `PWR` is closed. This jumper connects the 3.3V line and LED. Cutting this jumper will disable the LED.



- **I^2C Pull-up Resistors** - By default, this three way jumper labeled $\mathtt{I^2C}$ is closed and connects two pull-up resistors to the I^2C data lines. If you have many devices on your I^2C data lines, then you may consider cutting these two jumpers.

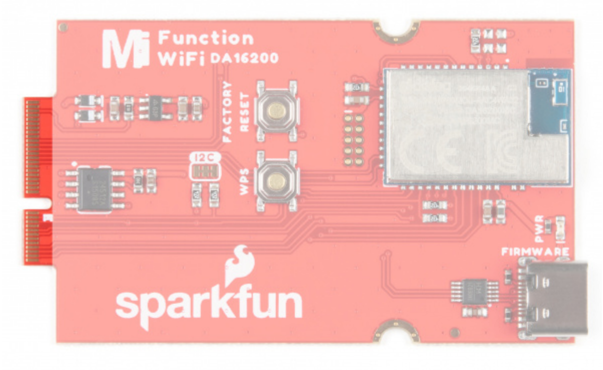


- **WP** - By default, write protect for the EEPROM is on. To disable write protection, close this jumper.



Hardware Pinout

The M.2 connector on the MicroMod WiFi Function Board - DA16200 is routed according to the pinout chart below:



Depending on your window size, you may need to use the horizontal scroll bar at the bottom of the table to view the additional pin functions. Note that the M.2 connector pins on opposing sides are offset from each other as indicated by the bottom pins where it says (Not Connected)*. There is no connection to pins that have a "-" under the primary function.

MICROMOD WIFI FUNCTION BOARD - DA16200 PINOUT TABLE

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

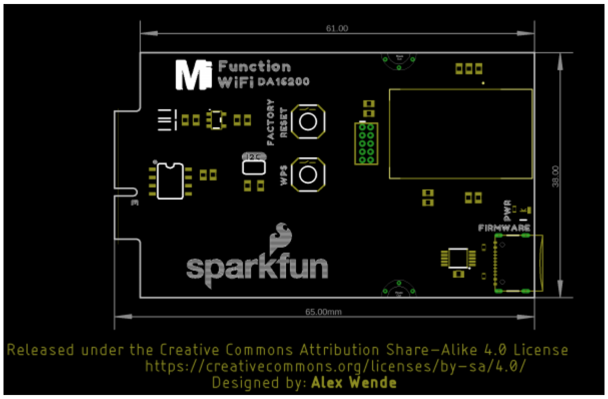
Description	Function	Bottom Pin	Top Pin	Function	Description
	(Not Connected)		**75**	GND	
	-	**74**	**73**	3.3V	Power Supply: 3.3-6V
	-	**72**	**71**	Power EN	Power Supply: 3.3-6V
	-	**70**	**69**	-	
	-	**66**	**65**	-	
	-	**64**	**63**	-	
	-	**62**	**61**	-	
	-	**60**	**59**	-	

	-	**58**	**57**	RTC_WAKE1	This pin is an input pin for receiving an external event signal from an external device like a sensor. The RTC block detects an external event signal via this pin and wakes up DA16200 from Sleep mode 2 or Sleep mode 3.
	-	**56**	**55**	RTC_WAKE2	This pin is an input pin for receiving an external event signal from an external device like a sensor. The RTC block detects an external event signal via this pin and wakes up DA16200 from Sleep mode 2 or Sleep mode 3.
	-	**54**	**53**	RTC_PWR_KEY	Enable to kick off a predefined power-up sequence and eventually all the necessary power is supplied to all the sub-blocks including the main digital block in DA16200. When disabled, all blocks are powered off and this mode is defined as Sleep mode 1.
	-	**52**	**51**	GPIOA11	
	-	**50**	**49**	CS	
	-	**48**	**47**	GPIOA10	
	-	**46**	**45**	GND	
	-	**44**	**43**	-	
	-	**42**	**41**	-	

Controls the write protection pin for the EEPROM. Pull low to enable.	EEPROM_WP	**40**	**39**	GND	
	-	**38**	**37**	-	
Controls EEPROM's I^{2}C address configuration.	EEPROM_A0	**36**	**35**	-	
Controls EEPROM's I^{2}C address configuration.	EEPROM_A1	**34**	**33**	GND	
Controls EEPROM's I^{2}C 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^{2}C - Clock signal for EEPROM
	-	**20**	**19**	I2C_SDA	I^{2}C - Data signal for EEPROM
	-	**18**	**17**	-	
	-	**16**	**15**	U1_RXI	
	-	**14**	**13**	U1_TXO	
	-	**12**	**11**	-	
	-	**10**	**9**	-	
	-	**8**	**7**	CIPO	
	-	**6**	**5**	COPI	
	-	**4**	**3**	SCK	
	-	**2**	**1**	GND	

Board Dimensions

The board uses the standard MicroMod Function Board size which measures about 1.50"x2.56".


61.00
Function
WiFi DA16200
FACTORY
RESET
WPS
I2C
PWR
FIRMWARE
sparkfun
38.00
65.00mm
Released under the Creative Commons Attribution Share-Alike 4.0 License
https://creativecommons.org/licenses/by-sa/4.0/
Designed by: Alex Wende

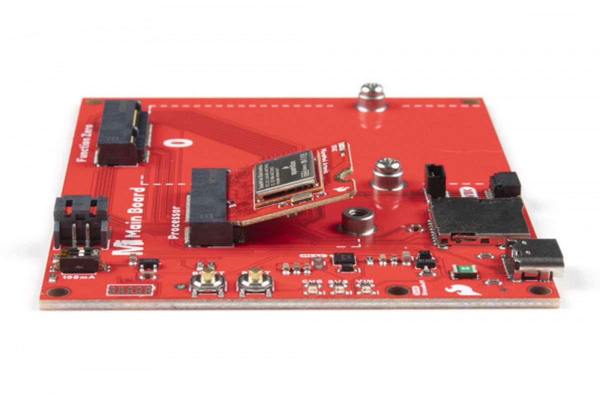
Hardware Hookup

Adding a Processor Board to the Main Board

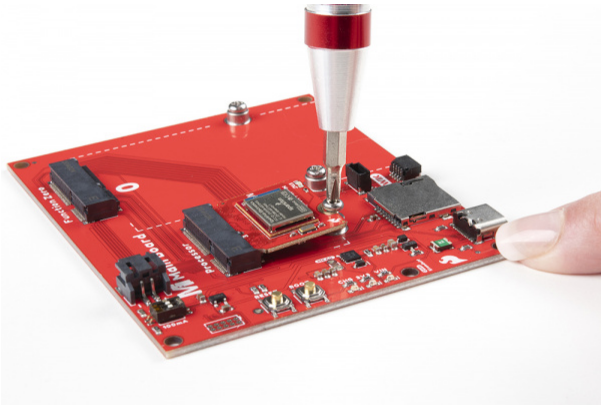
Align the top key of the MicroMod Artemis Processor Board to the *Processor* screw terminal of your Main Board and angle the board into the socket. Insert the board at an angle into the M.2 connector.

> **Note:** There is no way to insert the processor backward since the key prevents it from mating with the M.2 connector and as an extra safeguard to prevent inserting a processor that matches the key, the mounting screw is offset so you will not be able to secure an improperly connected processor board.

The Processor Board will stick up at an angle, as seen here:

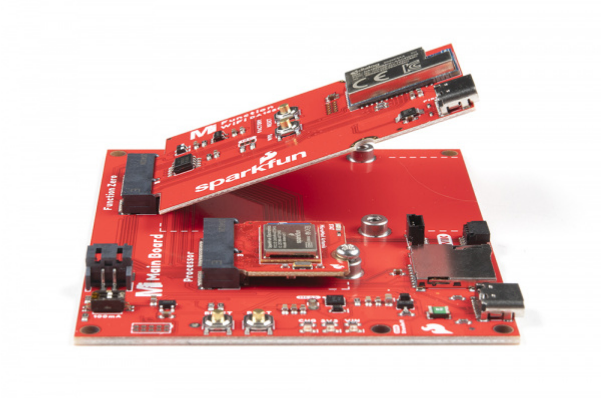


Once the board is in the socket, gently push the MicroMod Processor Board down and tighten the screw with a Phillip's head.

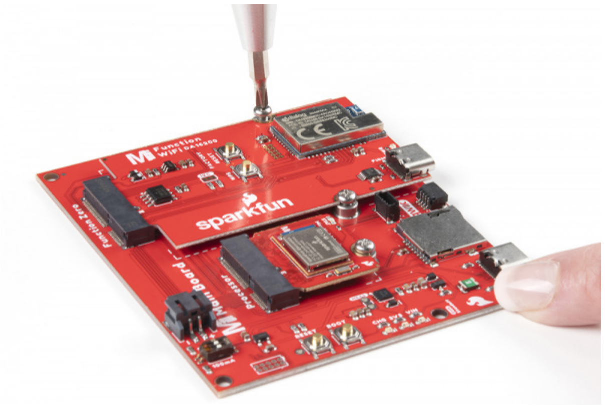


Adding a Function Board to the Main Board

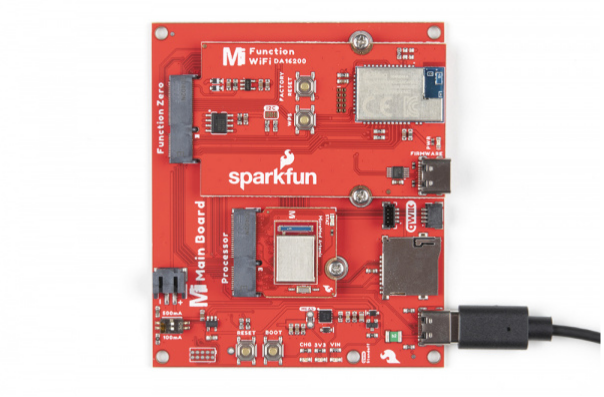
As with the Processor Board, align the top key of the MicroMod WiFi Function Board to the screw terminal of your Main Board on the *Function Board* side and angle the board into the socket. Insert the board at an angle into the M.2 connector.



Once the board is in the socket, gently push the MicroMod Function Board down and tighten the screws with a Phillip's head.

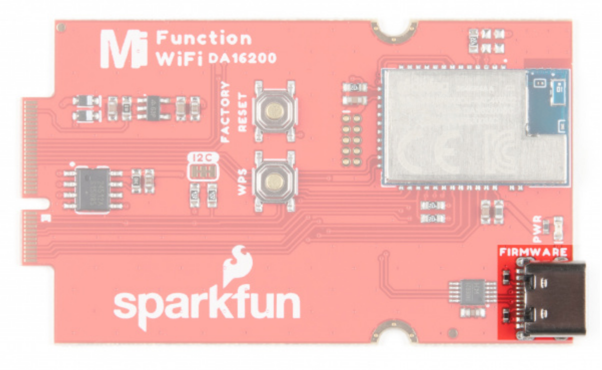


Once both boards are secure, your assembled MicroMod system should look similar to the image below!



DA16200 Firmware Update

To update the firmware you will need to connect the USB C cable to the MicroMod WiFi Function Board (DA16200) to a computer's COM port. An additional Main Board with a second USB C cable is also needed to power the Main Board and MicroMod WiFi Function Board.



Software Installation

Note: This 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 our tutorial on installing the Arduino IDE.

Arduino Board Definitions

We'll assume that you installed the necessary board files for your Processor Board. In this case, we used the MicroMod Artemis Processor Board.

MicroMod Artemis Processor Board Hookup Guide
OCTOBER 21, 2020
Get started with the Artemis MicroMod Processor Board in this tutorial!

CH340 Driver

You may need to install CH340 driver for the USB-to-serial converter. Check out our section on "How to Install CH340 Drivers" for help with the installation.

DOWNLOAD WINDOWS CH340 DRIVER (ZIP)

DOWNLOAD MAC OSX CH340 DRIVER (ZIP)

> **Note:** If applicable, make sure you are using the proper driver files for your CPU architecture. This is usually indicated by a folder or file name with "*x86*" for 32-bit processors or "*x64*" for 64-bit processors.

Main Board Example - Pin Connection Table

For the following examples, we are using the Artemis Processor Board. For DA16200 specific pins, here is the mapping to use in Arduino sketches:

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

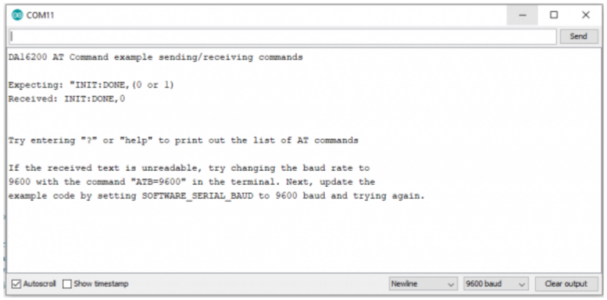
Function Board Pin Name	I/O Direction	Main Board's Processor Pin	
		Slot 0	Slot 1
VCC	Input	-	
3.3V	Input	-	
GND	-	-	
RTC_WAKE1	Input	G2	G7
RTC_WAKE2	Output	G1	G6
RTC_PWR_KEY	Input	G0	G5
GPIOA11		PWM0	PWM1
CS		CS0	CS1
GPIOA10		D0	D1

Example 1: Basic Connection

Copy and paste the code below into a fresh Arduino sketch.

```
/******************************************************************************
  Example_01 Basic Connection

  Connect to the DA16200 Module
  Print help commands

  Development environment specifics:
  IDE: Arduino 1.8.13
  Hardware Platform: MicroMod Artemis Processor, Single MicroMod Main Board

  Hardware Connections:
  Connect the MicroMod Processor Board to the Processor M2 Connector of the Main Board
  Connect the MicroMod DA16200 Function Board to the Function M2 Connector of the Main Board

  G0      --> RTC_PWR_KEY
  4       --> PWR_EN
  3.3V    --> 3.3V
  GND     --> GND

  This program is distributed in the hope that it will be useful,
  but WITHOUT ANY WARRANTY; without even the implied warranty of
  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
  GNU General Public License for more details.

  You should have received a copy of the GNU General Public License
  along with this program.  If not, see <http://www.gnu.org/licenses/>.
******************************************************************************/
#define PWR_EN 4
#define RTC_PWR_KEY 16

void setup() {
  pinMode(PWR_EN,OUTPUT);
  digitalWrite(PWR_EN,LOW);

  pinMode(RTC_PWR_KEY,OUTPUT);
  digitalWrite(RTC_PWR_KEY,LOW);

  Serial.begin(9600);
  Serial1.begin(115200); //Set SoftwareSerial baud

  while(!Serial){

  }
  delay(1000);



  //Enable DA16200 Module RTC power block
  pinMode(PWR_EN,OUTPUT);
  digitalWrite(PWR_EN,LOW);

  pinMode(RTC_PWR_KEY,OUTPUT);
  digitalWrite(RTC_PWR_KEY,LOW);
  Serial.println("DA16200 AT Command example sending/receiving commands\n");
  while(!Serial){

  }

  pinMode(PWR_EN,INPUT);

  pinMode(RTC_PWR_KEY,OUTPUT);
  digitalWrite(RTC_PWR_KEY,HIGH);

  delay(2000);


  //Listen for ready message ("+INIT:DONE")
  byte count = 0;
  String msg = "";

  while(count<20)
  {
    delay(100);
    while(Serial1.available())
    {
      msg += char(Serial1.read());
    }
    if(msg.length() > 5) break;
    count++;
    delay(100);
  }
  msg = msg.substring(3,msg.length());

  if(msg.length()>5)
  {
    Serial.println("Expecting: \"INIT:DONE,(0 or 1)");
    Serial.println("Received: " + msg);
  }
  else
  {
    Serial.println("Failed to receive initialization message\n");

    Serial.println("Make sure the baud rate for Serial1 matches the baud rate\n" \
                   "saved to the DA16200. You can also perform a factory reset by\n" \
                   "pressing and holding the GPIOA7 button for ~5s, which will\n" \
                   "reset the baud rate back to 115200");
  }
    Serial.println("\nTry entering \"?\" or \"help\" to print out the list of AT commands\n" \
                 "\nIf the received text is unreadable, try changing the baud rate to\n" \
                 "9600 with the command \"ATB=9600\" in the terminal. Next, update the\n" \
                 "example code by setting SOFTWARE_SERIAL_BAUD to 9600 baud and trying again.\n"
);
}

void loop() {
    while(Serial.available())
    {
      Serial1.print(char(Serial.read()));
    }

    while(Serial1.available())
    {
      Serial.print(char(Serial1.read()));
    }
}
```

Set your Board and Serial Port, and then upload the sketch to your Arduino. Then open the serial monitor. Make sure your baud rate is set to 9600. You'll begin to see output.



```
DA16200 AT Command example sending/receiving commands

Expecting: "INIT:DONE,(0 or 1)
Received: INIT:DONE,0


Try entering "?" or "help" to print out the list of AT commands

If the received text is unreadable, try changing the baud rate to
9600 with the command "ATB=9600" in the terminal. Next, update the
example code by setting SOFTWARE_SERIAL_BAUD to 9600 baud and trying again.
```

☑ Autoscroll □ Show timestamp Newline 9600 baud

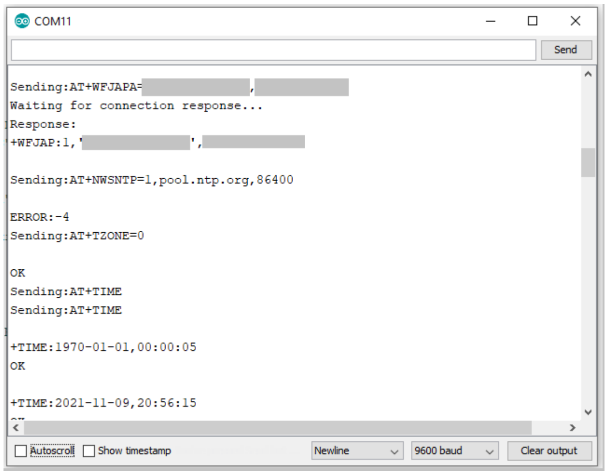
Example 2: Connecting to WiFi

Let's check out the WiFi with a simple example to grab the time.

Copy and paste the code below into a fresh Arduino sketch.

```
/******************************************************************************
  Example_02 WiFi Communcation

  Connect WiFi using the provided network credentials
  Talk to NTP server to set the current date/time
  Update the time to the correct time zone
  Print the current time approx. once every second

  Development environment specifics:
  IDE: Arduino 1.8.13
  Hardware Platform: MicroMod Artemis Processor, Single MicroMod Main Board

  Hardware Connections:
  Connect the MicroMod Processor Board to the Processor M2 Connector of the Main Board
  Connect the MicroMod DA16200 Function Board to the Function M2 Connector of the Main Board

  G0      --> RTC_PWR_KEY
  4       --> PWR_EN
  3.3V    --> 3.3V
  GND     --> GND

  This program is distributed in the hope that it will be useful,
  but WITHOUT ANY WARRANTY; without even the implied warranty of
  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
  GNU General Public License for more details.

  You should have received a copy of the GNU General Public License
  along with this program.  If not, see <http://www.gnu.org/licenses/>.
******************************************************************************/
#define PWR_EN 4
#define RTC_PWR_KEY 16

#define SOFTWARE_SERIAL_BAUD 115200
//#define SOFTWARE_SERIAL_BAUD 9600

String wifiSSID = "SSID";
String wifiPass = "PASSWORD";

int timezoneOffset = 0; //The hours offset from UTC (Mountain time is -6 for daylight savings, and -7 for standard)

void setup() {
  Serial.begin(9600);
  Serial1.begin(SOFTWARE_SERIAL_BAUD); //Set SoftwareSerial baud

  //Enable DA16200 Module RTC power block
  pinMode(RTC_PWR_KEY,OUTPUT);
  digitalWrite(RTC_PWR_KEY,HIGH);

  Serial.println("DA16200 AT Command Example: Connecting to WiFi\n");

  //Listen for ready message ("+INIT:DONE")
  byte count = 0;
  String msg = "";
  while(count<20)
  {
    while(Serial1.available())
    {
      msg += char(Serial1.read());
    }
    if(msg.length() > 5) break;
    count++;
    delay(100);
  }
  msg = msg.substring(3,msg.length()); //Remove NULL,CR,LF characters from response

  if(msg.length()>5)
  {
    Serial.println("Expecting: \"INIT:DONE,(0 or 1)");
    Serial.println("Received: " + msg);
  }
  else
  {
    Serial.println("Failed to receive initialization message.\n" \
                   "Make sure you're using the correct baud rate.\n");
    while(1);
  }

  //Configure module for STA mode
  Serial.println("Sending:AT+WFMODE=0");
  Serial1.println("AT+WFMODE=0");

  //Wait for "OK" response
  while(1)
  {
    msg = "";
    while(Serial1.available())
    {
      msg += char(Serial1.read());
      delay(1);
    }
    Serial.print(msg);
    if(msg.length() > 1) break;
  }

  //Apply a software reset to finish changing the mode
  Serial.println("Sending:AT+RESTART");
  Serial1.println("AT+RESTART");

  //Wait for "OK" response
  while(1)
  {
    msg = "";
    while(Serial1.available())
    {
      msg += char(Serial1.read());
      delay(1);
    }
    Serial.print(msg);
    if(msg.length() > 1) break;
  }

  //Listen for ready message ("+INIT:DONE") after the reset is finished
  count = 0;
  msg = "";
  while(count<20)
  {
    while(Serial1.available())
    {
      msg += char(Serial1.read());
    }
    if(msg.length() > 5) break;
    count++;
    delay(100);
  }

  Serial.println(count);
  Serial.println(msg);
  msg = msg.substring(3,msg.length()); //Remove NULL,CR,LF characters from response

  if(msg.length()>5)
  {
    Serial.println("Expecting: \"INIT:DONE,(0 or 1)");
    Serial.println("Received: " + msg);
  }
  else
  {
    Serial.println("Failed to receive initialization message.\n" \
                   "Continuing anyway...\n");
  }

  //Connect to WiFi using the provided credentials
  Serial.println("Sending:AT+WFJAPA=" + wifiSSID + "," + wifiPass);
  Serial1.println("AT+WFJAPA=" + wifiSSID + "," + wifiPass);

  Serial.println("Waiting for connection response...");
  while(1)
  {
    msg = "";
    while(Serial1.available())
    {
      msg += char(Serial1.read());
      delay(1);
    }

    if(msg.length() > 10)
    {
      Serial.print("Response:");
      Serial.println(msg);
      break;
    }
  }

  msg = msg.substring(3,msg.length()); //Remove NULL,CR,LF characters from response

  //If connection to AP is successful, response will be WFJAP:1,SSID,IP_ADDRESS, or WJAP:0 if fa
iled
  if(msg.startsWith("WFJAP:1"))
  {
      //Talk to NTP server to get the current time, along with how often to get time sync
      Serial.println("Sending:AT+NWSNTP=1,pool.ntp.org,86400");
      Serial1.println("AT+NWSNTP=1,pool.ntp.org,86400");

      //Wait for "OK" response
      while(1)
      {
        String msg = "";
        while(Serial1.available())
        {
          msg += char(Serial1.read());
          delay(1);
        }
        Serial.print(msg);
        if(msg.length() > 1) break;
      }

      //Provides the correct UTC offset for the current time
      Serial.println("Sending:AT+TZONE="+String(timezoneOffset*3600));
      Serial1.println("AT+TZONE="+String(timezoneOffset*3600));

      //Wait for "OK" response
      while(1)
      {
        String msg = "";
        while(Serial1.available())
        {
          msg += char(Serial1.read());
          delay(1);
        }
        Serial.print(msg);
        if(msg.length() > 1) break;
      }
  }
  else
  {
    Serial.println("Connection unsucessful :(\n\n" \
                   "Make sure the WiFi credentials are correct, and the module is in the station 
mode");
    while(1);
  }
}

void loop() {
  //Get the current time
  Serial.println("Sending:AT+TIME");
  Serial1.println("AT+TIME");

  while(Serial1.available())
  {
    Serial.print(char(Serial1.read()));
    delay(1);
  }

  delay(1000);
}
```

Set your Board and Serial Port, and then upload the sketch to your Arduino. Then open the serial monitor. Make sure your baud rate is set to 9600. You'll begin to see output.



```
Sending:AT+WFJAPA=[redacted],[redacted]
Waiting for connection response...
Response:
+WFJAP:1,'[redacted]',[redacted]

Sending:AT+NWSNTP=1,pool.ntp.org,86400

ERROR:-4
Sending:AT+TZONE=0

OK
Sending:AT+TIME
Sending:AT+TIME

+TIME:1970-01-01,00:00:05
OK

+TIME:2021-11-09,20:56:15
```

□ Autoscroll □ Show timestamp Newline 9600 baud

Firmware Update

If for any reason you want to re-flash the AT Firmware from Dialog Semiconductor, you can download it below:

DOWNLOAD AT FIRMWARE

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

Now that you've successfully got your MicroMod ESP Function Board up and running, it's time to incorporate it into your own project! For more information, check out the resources below:

- Schematic (PDF)
- Eagle Files (ZIP)
- Board Dimensions (PNG)
- CH340 Drivers
  - Windows
  - Max OSX
- DA16200 Datasheet (PDF)
- DA16200 AT Command User Manual
- Download AT Firmware
- AP2112 Datasheet (PDF)
- GitHub Hardware Repo
- Dialog's Resources for DA16200

Looking for more inspiration? Check out these other tutorials related to MicroMod.

MicroMod Artemis Processor Board Hookup Guide

Get started with the Artemis MicroMod Processor Board in this tutorial!

MicroMod Weather Carrier Board Hookup Guide

A quick guide to help to create your own MicroMod weather station using the MicroMod Weather Carrier Board and Processor of your choice.

SparkFun MicroMod Input and Display Carrier Board Hookup Guide

A short Hookup Guide to get started with the SparkFun MicroMod Input and Display Carrier Board

MicroMod Asset Tracker Carrier Board Hookup Guide

Get started with the SparkFun MicroMod Asset Tracker Carrier Board following this Hookup Guide. The Asset Tracker uses the u-blox SARA-R510M8S LTE-M / NB-IoT module to provide a host of data communication options.