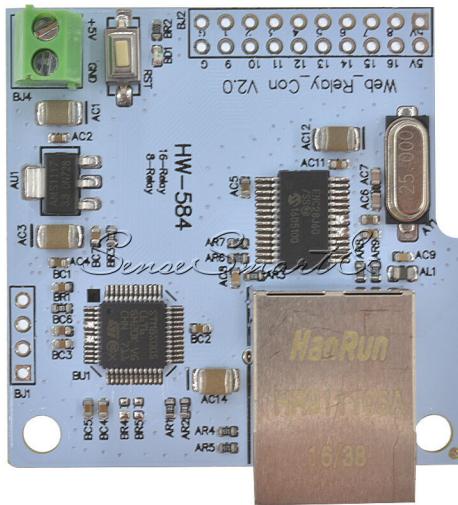


Network Module Manual

Replacement Firmware for the
Web_Relay_Con V2.0 HW-584
February 21, 2021

Most Recent Major Update Code Rev 20210123 1257

Introduction



Did you buy one (or more) of these Network Modules and then find disappointment in the software on the board?

- All of the modules have the same MAC address. That's a problem if you want more than one on your network. And the supplier does not give you a way to change the MAC.
- If you change the IP Address the device returns to its default IP Address when it power cycles. That makes it pretty much useless even if you only put one on your network - unless you're OK with it always having IP Address 192.168.1.4.

I was disappointed enough that I decided to reprogram the device to provide a web server interface that lets you change the IP Address, Gateway (Default Router) Address, Netmask, Port number (a REAL port number), and MAC Address. I also added the ability for the device to remember all these settings through a power cycle. Any Output settings you make are also saved through a power cycle (Outputs typically being Relay controls).

NOTE 1: The software provided in this project only works with the “Web_Relays_Con V2.0 HW-584” which is based on the STM8S-005 or STM8S-105 processor and ENC28J60 ethernet controller. I haven’t tried it with any other version of the hardware. I think the V.1 FC-160 is based on a Nuvoton processor and this code and the tools are incompatible.

NOTE 2: I am not in any way associated with the manufacturer of this device. I only wrote code to run on it for my own hobby purposes, and I am making it available for other hobbyists.

NOTE 3: If you’re looking to buy these modules the best source I’ve found is eBay. Best search term is “ENC28J60 Network Module”. I’ve also seen them on Amazon, Banggood, and Aliexpress. In some cases they show photos of both the V.1 and V2.0 versions, but don’t provide a way of specifying which one you want. You may need to communicate with the seller to be sure you’ll get the V2.0 device.

Thank You!

Many thanks to Carlos Ladeira for his help with user interface ideas, many hours of testing, and his patience during development of the MQTT version of this code. And many thanks to Jevdeni Kiski for his guidance and code contributions in developing the Home Assistant interface and new Configuration interface. We’ve ended up with a much better project as a result.

Document License

Copyright (C) 2020 Michael Nielson.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Table of Contents

Introduction.....	1
Thank You!	2
Document License	2
Table of Contents.....	3
Change Log.....	4
Screen Shots and Usage.....	9
Notes on Feature Settings	12
Notes on Individual IO Settings.....	14
Notes on the MAC Address	16
Notes on REST Commands	18
Notes on MQTT	20
Notes on Link Error Statistics.....	27
Functional Limitations	31
Programming the Module	33
Alternative Way to Force Defaults or Downgrade Firmware.....	44
Alternative Way to Set Initial IP Address.....	47
Display Values vs Pin Logic Levels	50
Network Module Schematic	52
Pinouts.....	54
Notes on Interfacing to Relay Modules	55
Notes on Inputs	61
Hardware Design to Maintain Relay States Through a Power Loss or Reboot.....	62
Adding DS18B20 Temperature Sensors	68
Developers: Setting Up a Development Environment.....	70
Developers: Location of EEPROM Variables	72
Developers: Notes on Debug Bytes	74
Developers: Notes on Configuration Debug and pin_control Bytes	78
Developers: Using the UART	80
Code Credits.....	83
Documentation License Note.....	86

Change Log

June 20, 2020 – Initial Release

August 2, 2020 – Added descriptions of “8 Output / 8 Input” and “16 Input” configurations.

August 6, 2020 – Added a note to make sure people know they don’t need to set up a development environment unless they need to change the code.

August 20, 2020 – Added a note on editing the .stp file

November 16, 2020 – Major change: Added MQTT support

Other changes:

- Changed the GUI to have a Configuration page in place of the former Address Settings page
- Moved the device Name input field to the new Configuration page
- Moved the Relay Invert control to the new Configuration page
- Added an Input Invert control
- Added the ability to select whether all relays retain their state, are all forced off, or are all forced on when a power cycle occurs. This is in the “Config” settings added to the Configuration page.
- Added the ability to select whether the Ethernet interface operates in Half or Full Duplex.
- Added the ability to configure settings for MQTT Broker IP Address and Port.
- Added the ability to provide options MQTT ID and Password information.
- Added MQTT connection status indicators in the Configuration page.
- Fixed several corner case bugs in the web server code that were causing anomalous behavior in browsers.

November 19, 2020 – Minor edits to this document:

- Corrected the screen shot for the MQTT IO Control page
- Changed text to indicate that Full Duplex worked with some unmanaged switches, but not all.
- Added that you can use “all” to turn all relays on or off with MQTT.

November 21, 2020 – Minor change to code and this document:

- Fix to the “short form” Input pin state information – it was not showing correctly if the Invert function was on.
- Added tables showing how the displayed pin states relate to the physical pin voltages.
- Added an error count statistics display to the MQTT version of the code.

November 26, 2020 – Document changes only:

- Rearranged some sections in the document.
- Added EEPROM bit field definitions.
- Added tables showing relationship of browser and MQTT fields to pin logic levels.

November 30, 2020

Code changes:

- o Added the /98 REST command
- o Corrected typo in the HELP pages that mis-stated the REST command numbers.

Document changes:

- o Added section describing all the REST commands.

December 2, 2020

Code changes:

- o Fixed issue regarding browsers on multiple IP addresses.
- o Fixed issue regarding browser interference on page changes.

Document changes:

- o Added section to describe functional limitations (like number of browser sessions, browser interference).
- o Added section on an alternative method for entering the initial IP Address.

December 4, 2020

Code changes:

- o Code change made to reduce or eliminate relay state changes during reboot.

Document changes:

- o Added section on alternative hardware design methods for maintaining relays states during power loss.

December 18, 2020 Code Revision 20201218 2202

Code changes:

- o Fixed bug in RXERIF diagnostic counter (MQTT builds only).
- o Significant rewrite of timing functions around the MQTT code to improve the rate at which MQTT commands can be executed.
- o Added Independent Watchdog (hardware watchdog) to restart the module should it hang.
- o Removed the Error Statistics button from the MQTT builds.
- o Added EEPROM lock-out except when intentionally making EEPROM changes.

Document changes:

- o Relocated the MQTT Error Statistic description

December 21, 2020

MAJOR CHANGE: Changed all MQTT commands to match the Home Assistant standard.

Code changes:

- Fixed timing issue in MQTT command processing
- Changed all MQTT commands to match the Home Assistant standard.
- Simplified the appearance of the browser interface to make code space available for continued MQTT development.

Document changes:

- Updated MQTT command list

December 30, 2020 Code Revision 20201230 0411

Code changes:

- Added window sizing to improve appearance on small devices (iPhone etc).
- Added Home Assistant Auto Discovery including a Config setting to enable Auto Discovery.
- Changed reset button routine to prevent hardware watchdog from firing while button is pressed.

Document changes:

- Added description of Auto Discovery and associated Config setting

January 23, 2021 Code Revision 20210123 1257

MAJOR CODE UPDATE. This code update will retain the majority of your settings from previous releases like your IP addresses, Port numbers, and MAC address.

However, you need to re-enter settings associated with the IO pins (Input/Output, Invert, after Boot State, etc). This will be readily apparent in the Browser GUI. Note that the default is for all pins to be Input pins.

Code changes:

- Updated GUI to improve the way IP Addresses and MAC numbers are entered.
- Updated GUI and code to provide a single build for Browser and MQTT configurations.
- Updated GUI and code to allow each pin to be individually configured as an input or output, and to allow each pin to be individually configured for Invert and Power On state.
- “Help” no longer fits in the Flash on the module, so a link was added to get to documentation on the GitHub site.
- Added a checkbox based “Features” field to replace the former “Config” bytes.
- Improved the Home Assistant Auto Discovery feature to better control population of the Home Assistant device management screen when Pin configurations are changed.

Document changes:

- Describe above code changes
- Changed Screen Shots

- Replaced Config Settings section with Feature Settings section
- Added section on Individual IO Settings
- Updated REST commands
- Eliminated section “Notes on Compiling Different Configurations”
- Updated section “Location of EEPROM Variables”
- Added section “Alternative Way to Force Defaults”
- Updated section “Programming the Module” to reflect use of a single release for all configurations.

January 26, 2021 Code Revision 20210126 0355

Code changes:

- Repaired Debug Statistics functionality, HOWEVER that functionality is still not enabled due to lack of space and continuing work on other features and bug fixes.

Document changes:

- None.

January 26, 2021 Code Revision 20210126 0527

Code changes:

- Fixed two bugs where a integer-to-hex conversion was being made that should have been integer-to-decimal.

Document changes:

- None.

January 27, 2021 Code Revision 20210127 1112

Code changes:

- Fixed bug that was allowing blank fields in the IP Address, Gateway, Netmask, and Port fields.
- Fixed bug that was preventing operation of the REST 00-31 commands.
- Fixed bug that was causing the REST 98 and 99 commands to output the wrong values.

Document changes:

- None.

February 8, 2021 Code Revision 20210208 0523

Code changes:

- Added DS18B20 Temperature Sensor interface and display functions

Document changes:

- Documented the DS18B20 interface

February 20, 2021 Code Revision 20210220 2350

Code changes:

- Added support for UART debug output messages on IO 11 in Developer builds
- Re-enabled the former MQTT Error Statistics page renamed as Link Error Statistics

Document changes:

- Documented support for UART debug output messages on IO 11
- Documented Link Error Statistics
- Added description of Debug bytes for developers
- Changed some section titles to clarify which ones are for Developers, and reorganized the document to collocate Developer sections.
- Moved the Change Log after the Table of Contents
- Updated information on where to find .stp and .sx files on GitHub

February 21, 2021 Code Revision 20210221 1826

Code changes:

- Bug fix in POST parsing routine

Document changes:

- None

Screen Shots and Usage

IO Control

Name: NetworkModule-182

SET

Output #1	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #2	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #3	<input checked="" type="radio"/> ON <input type="radio"/> OFF
Output #4	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #5	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #6	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #7	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #8	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #9	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #10	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #11	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #12	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #13	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #14	<input type="radio"/> ON <input checked="" type="radio"/> OFF
Output #15	<input type="radio"/> ON <input checked="" type="radio"/> OFF

Device name display. The name can be changed on the Configuration page.

Color coded quick reference for Input and Output states

Relays are turned on and off with radio buttons

Changes take effect when you click on Save

You can Undo your changes before you save

Save Undo All

Refresh Configuration

Page Refresh and Navigation to Configuration page

Temperature Sensors

1 022.3°C
2 022.5°C
3 -----°C
4 -----°C
5 -----°C

If DS18B20 mode is enabled on the Configuration page Temperature Sensor data appears here.

Screen Shots and Usage

Configuration

Name	NetworkModule-182	Enter a device name of your choosing so that you can easily tell which Network Module you are connected to	
IP Address	192.168.1.182	Change IP Address, Gateway Address, Netmask, Port Number, MAC Address	
Gateway	192.168.1.1		
Netmask	255.255.255.0		
Port	8080		
MAC Address	c2:4d:69:6b:65:02		
Features	<input type="checkbox"/> Full Duplex <input type="checkbox"/> HA Auto <input checked="" type="checkbox"/> MQTT <input checked="" type="checkbox"/> DS18B20	Features can be enabled with check boxes: Full/Half Duplex Ethernet Home Assistant Autodiscovery MQTT, DS18B20	
MQTT Server	192.168.1.106		
MQTT Port	1883		
MQTT Username		MQTT Server settings, Username, and Password	
MQTT Password		MQTT Status indicators to show connection progress	
MQTT Status	██████		
IO	Type	Invert Boot state	Configure each IO as: Input/Output/Disabled – do this first and Save, then configure Invert and Boot State
#1	output	<input type="checkbox"/> off	
#2	output	<input checked="" type="checkbox"/> off	The IOControl Display shows "ON" or "OFF". The Invert setting is used to invert the condition on the physical pin.
#3	output	<input type="checkbox"/> off	
#4	output	<input type="checkbox"/> off	
#5	output	<input type="checkbox"/> off	
#6	output	<input checked="" type="checkbox"/> off	
#7	output	<input type="checkbox"/> off	
#8	output	<input type="checkbox"/> off	
#9	output	<input type="checkbox"/> off	
#10	output	<input type="checkbox"/> off	
#11	output	<input type="checkbox"/> off	
#12	output	<input type="checkbox"/> off	
#13	output	<input type="checkbox"/> off	
#14	output	<input type="checkbox"/> off	
#15	output	<input type="checkbox"/> off	
#16	disabled	<input type="checkbox"/>	The Boot state setting lets you control how the Output state is handled when the device reboots. Off – The Output will go to the OFF state (which can be high or low depending on the Invert setting). On – The Output will go to the ON state. Retain – The Output will go to the state it was in prior to reboot.
Save Undo All			Changes take effect when you click Save You can Undo changes before you save them
Code Revision 20210207 1543			
Help Wiki			
Reboot			
Refresh IO Control			

Screen Shots and Usage

1111110111111011

HTML output page with a short form representation of the Relay states. Useful to app writers.

Relay states are Relay 1 on the left, Relay 16 on the right.

This display will look the same with the /98 and /99 command.

Notes on Feature Settings

The Features checkboxes on the Configuration page let you modify operation of the code as follows:

Features	<input type="checkbox"/> Full Duplex <input type="checkbox"/> HA Auto <input checked="" type="checkbox"/> MQTT <input checked="" type="checkbox"/> DS18B20
----------	---

Full Duplex Setting:

The Full Duplex checkbox determines the Half / Full Duplex Ethernet communication method. **The default setting is Half Duplex because that is the most reliable setting for the ENC28J60 Ethernet chip.** Since all the Ethernet transactions that will occur with the module are small and infrequent there is no real performance advantage to using Full Duplex.

During test it was found that Cisco business level switches exhibited Half Duplex timing that the ENC28J60 cannot handle, the symptom being a device disconnect (and automatic recovery) every few hours. This isn't the fault of the Cisco switch, rather it appears to be the fault of errata in the ENC28J60. During test it was determined that we could get around this issue when connected to the Cisco switch by enabling Full Duplex mode in the ENC28J60. While there was concern that this would not work (due to chip spec notes and online discussion over the years), it seemed to run error free. There may be other switches which show the same issue. Again, the reason Full Duplex works may be the very low messaging rate used with the Network Module which eliminates the need for flow control.

Note 1: The spec for the chip indicates that Full/Half Duplex auto-negotiation DOES NOT work. However, experimentation showed that both Full and Half Duplex worked with some unmanaged switches, but not with others. Problems were always running Half Duplex only with the Cisco 1G managed switch. No problem was seen running Half Duplex with a Cisco 10/100 managed switch.

Note 2: If you choose to use the Full Duplex setting note that the spec says the Switch port the device is connected to MUST be manually configured for Full Duplex operation ... even though our testing did not always show that to be the case. Of course we had a limited number of switches and this might be an issue on some other switch.

Note3: Feel free to experiment with this setting at your own risk to see what works best in your network configuration. I recommend you use Half Duplex and only try Full Duplex if you have issues.

HA Auto:

The Home Assistant Auto Discovery setting enables the Network Module to send MQTT Auto Discovery Publish messages to your Home Assistant server. Checking this setting will automatically enable MQTT (and the MQTT checkbox will automatically be set). Do not enable this setting unless you are operating in an MQTT environment with Home Assistant.

MQTT:

Checking this box will enable the MQTT interface. HA Auto DOES NOT need to be enabled with MQTT. This will allow you to operate with MQTT servers without using Home Assistant, OR it will enable you to use Home Assistant without Auto Discovery.

DS18B20:

Checking this box will cause IO 16 (Pin 16) to be disabled for use as an Input / Output pin and will enable operation of the DS18B20 Temperature Sensor interface on IO 16 (Pin 16). You can attach up to 5 DS18B20 temperature sensors to pin 16, and the temperatures sensed by those devices will be displayed on the IOControl page. See the section “Adding DS18B20 Temperature Sensors”.

Notes on Individual IO Settings

The Individual IO Settings drop down and check boxes on the Configuration page lets you modify functionality of each IO as follows:

IO	Type	Invert Boot state
#1	input	<input checked="" type="checkbox"/>
#2	input	<input type="checkbox"/>
#3	input	<input type="checkbox"/>
#4	input	<input type="checkbox"/>
#5	input	<input type="checkbox"/>
#6	input	<input type="checkbox"/>
#7	input	<input type="checkbox"/>
#8	input	<input type="checkbox"/>
#9	input	<input type="checkbox"/>
#10	input	<input type="checkbox"/>
#11	output	<input type="checkbox"/> off
#12	output	<input type="checkbox"/> on
#13	output	<input type="checkbox"/> retain
#14	output	<input checked="" type="checkbox"/> off
#15	output	<input type="checkbox"/> off
#16	output	<input type="checkbox"/> off

Type Drop Down Settings:

Each IO has a “Type” drop down box that lets you configure each IO as follows:

Input – Self explanatory. Sets the IO as an input.

Output – Self explanatory. Sets the IO as an output.

Disabled – In Browser applications a Disabled Input or Output will not appear in the IO Control page. REST commands will also not affect a Disable Output. In Home Assistant applications a Disabled Input or Output will result in a Config message with an empty payload, resulting in that IO being deleted from the Home Assistant configuration.

IMPORTANT: Set the Input/Output/Disabled setting, THEN Save, THEN make other setting changes.

VERY IMPORTANT: Be sure you understand your hardware design. You must avoid setting an IO as an Output if the associated pin is tied to VCC or Ground, as that is likely to damage the output driver on the processor. If your hardware design can provide high levels of input current on a pin make sure that pin is defined as an Input.

Invert Settings:

Each IO has an “Invert” checkbox.

Effect on Inputs:

If not checked, a low voltage on the Input pin will display as OFF in the IOControl page and will be reported as OFF to MQTT clients.

If checked, a low voltage in the Input pin will display as ON in the IOControl page and will be reported as ON to MQTT clients.

Effect on Outputs:

If not checked, an OFF indication in the IOControl page or MQTT Client will result in a low voltage on the Output pin.

If checked, an OFF indication in the IOControl page or MQTT Client will result in a high voltage on the Output pin.

Since some devices connected to Output pins may be in an ON state with a low voltage, and others may be in an OFF state with a low voltage you will need to figure out what the Invert setting should be for your specific design. It is really a simple matter of connecting your peripheral device, setting ON or OFF in the Browser, then checking the Invert box as needed so that an ON state in the Browser matches an ON condition in your peripheral.

Boot State Settings:

Each IO has a “Boot State” dropdown box. The effect of each setting is as follows:

Off: After boot the state of the Output is OFF.

On: After boot the state of the Output is ON.

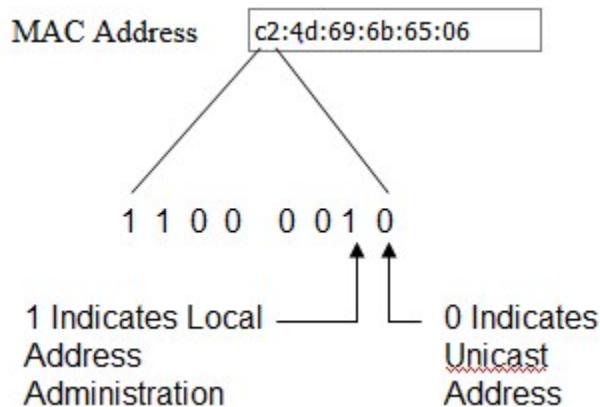
Retain: After boot the state of the Output is the same as it was before boot.

Notes on the MAC Address

When new the Network Modules all have the same MAC address. This obviously doesn't work when you try to put more than one on a network.

A MAC address is only used within your network. Your router(s) and switch(es) use the MAC address as the means of uniquely addressing all the hardware in your network. The MAC address does not appear outside your network so it only needs to be unique to YOUR network, not to the entire world. This being the case, you only need to make sure that any MAC address you put in the Network Module does not conflict with any other hardware in your local network.

The default MAC address value in the code provided is just a random value with the exception that it has the two least significant bits of the most significant octet arranged to make it a "Unicast" and "Locally Administered Address (LAA)" as illustrated here. You MUST make sure you use a LAA and Unicast address.



All other bits and octets in the MAC address (including those in the most significant octet) can be anything you want as long as you set the two bits above as shown.

Despite this being a LAA MAC address there is still some very remote possibility the MAC you pick will conflict with some other hardware you have on your network. You can search on Google to find methods of finding all MAC addresses on your network – the method you choose will depend on your level of expertise. Generally this is not required, and if you suspect a conflict you may just find it easier to try a different MAC address on the Network Module. Maybe make the middle fours octets something you fancy.

A good reference for MAC address explanations is here:
https://en.wikipedia.org/wiki/MAC_address

If you are installing multiple devices on your network I suggest that you just change the values in the least significant octet and leave the others as-is. I advise you add a label to your Network Module with the MAC you programmed into it.

Notes on REST Commands

A REST (Representational State Transfer) type of interface has been implemented to enable access to Input/Output states and other functions without the use of the browser. This is to enable development of external programs to operate the Network Module without use of the full GUI. If Help pages are enabled in the build you are using some of this information is available there, but Flash memory space is limited and the Help pages are one of the first places that are reduced to make memory available for additional code. A complete list of the REST commands is provided here.

For all commands enter an http request as follows:

`http://IP:Port/xx`

where

- IP = the Network Module IP Address, for example 192.168.1.4
- Port = the Network Module Port number, for example 8080 (Port number may be omitted if the device is set to Port 80)
- xx = one of the codes below

00 = IO 1 OFF	08 = IO 5 OFF	16 = IO 9 OFF	24 = IO 13 OFF
01 = IO 1 ON	09 = IO 5 ON	17 = IO 9 ON	25 = IO 13 ON
02 = IO 2 OFF	10 = IO 6 OFF	18 = IO 10 OFF	26 = IO 14 OFF
03 = IO 2 ON	11 = IO 6 ON	19 = IO 10 ON	27 = IO 14 ON
04 = IO 3 OFF	12 = IO 7 OFF	20 = IO 11 OFF	28 = IO 15 OFF
05 = IO 3 ON	13 = IO 7 ON	21 = IO 11 ON	29 = IO 15 ON
06 = IO 4 OFF	14 = IO 8 OFF	22 = IO 12 OFF	30 = IO 16 OFF
07 = IO 4 ON	15 = IO 8 ON	23 = IO 12 ON	31 = IO 16 ON

55 = All Outputs ON

56 = All Outputs OFF

60 = Show IOControl Page

61 = Show Configuration Page

65 = Flash LED

91 = Reboot

98 = Show Short Form IO Settings (without HTML formatting)

99 = Same as /98

Note1: Output control commands (00 to 31) only work for IO defined as an Output. If the IO referenced in the command is an Input no action is taken.

Note 2: Commands 98 and 99 provide the IO states as 16 alphanumeric characters WITHOUT any HTML formatting. Both commands produce the same result for

backward compatibility. This command may be useful to some external applications that automate interaction with the Network Module using URL style commands.

Notes on MQTT

This is not a tutorial on MQTT as there are a lot of great resources online to bring you up to speed if you are just getting started with this protocol. Here I am only including notes on the tools and methods used in test of the MQTT functionality on the Network Module.

Configuration Settings:

First a discussion of the Configuration page parameters associated with MQTT

MQTT Server	192.168.1.106
MQTT Port	1883
MQTT Username	[redacted]
MQTT Password	[redacted]
MQTT Status	

At this date the MQTT Broker Server must be specified in the form of an IP Address (as opposed to a URL). In future versions I may be able to allow use of a URL for the Broker Server.

The MQTT Port is self explanatory. The MQTT default value of 1883 automatically appears, but you can enter any port number you have assigned to MQTT on your Broker Server.

Security:

The MQTT Username and MQTT Password are optional and only required if you've set up your Broker to require them.

IMPORTANT: SSL/TSL are NOT implemented due to memory restrictions. I haven't found any implementations that are small enough to fit in the flash space available on the Network Module, so I don't expect this to ever be possible.

If security is really important due to the need to access MQTT on the device from the internet I suggest using access to your internal MQTT Broker Server in a secure way, then letting the Broker Server pass all messages on your internal network. You don't have to do it this way, but this is a suggestion for improving security rather than just exposing the device to the internet.

Tools and test methods:

Tools used in development and test of the MQTT functionality:

- The Mosquitto Broker was used on a Windows 10 laptop and worked very well with the Network Module.
- Chrome with MQTTLens was used at various points to provide a manual MQTT subscribe and publish interface.
- NodeRed was used to drive automated MQTT messages.
- Carlos Ladeira used Home Assistant in concert with NodeRed and the Mosquitto Broker in a Linux environment to perform extensive long run testing.

You aren't restricted to the above. Any tools and interfaces that are MQTT compliant should work just as well. But if you are just getting started I can recommend the above as a good place to start.

MQTT Status Indicators:

MQTT Status 

The MQTT Status indicators show connection progress with the MQTT Server and Broker:

- Box 1 - Indicates that the MQTT Connection process has started.
- Box 2 - Indicates a successful ARP reply from the Server.
- Box 3 - Indicates a successful TCP connection with the Server.
- Box 4 - Indicates the MQTT Broker has responded (Connect phase)
- Box 5 - Indicates initial communication with the Broker has completed successfully (initial subscribe and publish messages completed).

Once all 5 boxes are green the Network Module is connected to the Broker and normal MQTT communications can proceed.

Reminder regarding the Output “Boot State” setting:

A reminder regarding the Output Boot State setting for the Outputs on the Configuration page: If you select ‘Retain’ the Output states are written to the EEPROM every time an Output changes state. If you anticipate a lot of Output state changes you may wear out the EEPROM with too many changes to the Output states.

Client Publish and Subscribe messaging:

The Client **Publishes** the following messages to control outputs:

NetworkModule/<devicename>/output/xx/set	Payload: "ON" or "OFF"
NetworkModule/<devicename>/output/all/set	Payload: "ON" or "OFF"
NetworkModule/<devicename>/state-req	Payload: none

Where "xx" is the IO number of the Output

The Client **Subscribes** to the following:

NetworkModule/<devicename>/availability

This Subscribe enables the Client to receive the Network Module online / offline messages.

NetworkModule/<devicename>/input/+

NetworkModule/<devicename>/output/+

These two Subscribes enable the Client to receive changes in Input and Output states. The "+" causes the broker to send the client the IO state messages without reflecting the client's own "set" commands back to the client (reduces traffic).

NetworkModule/<devicename>/temp/+

This allows the client to receive the Temperature Sensor data produced by the Network Module if DS18B20 mode is enabled and Temperature Sensors are attached to the Network Module.

NetworkModule/<devicename>/state

This allows the client to receive the responses to the state-req Publish commands that the Client sends.

The Network Module **Publishes** the following when an IO state change occurs:

NetworkModule/<devicename>/input/xx Payload: "ON" or "OFF"

NetworkModule/<devicename>/output/xx Payload: "ON" or "OFF"

Where "xx" is the IO number of the Input or Output

The Network Module **Publishes** the following every 30 seconds if DS18B20 mode is enabled and Temperature Sensors are attached to the Network Module::

NetworkModule/<devicename>/temp/xx Payload: Temperature in degrees Celsius in the format " 000.0" or "-000.0"

Where "xx" is the Temperature Sensor name from 01 to 05

The Network Module **Publishes** the following in response to receiving a state-req Publish message:

NetworkModule/<devicename>/state Payload: see below

The payload consists of two bytes with the bits organized as follows:

First byte:

- Bit 7 = IO 16 state (1 = ON, 0 = OFF)
- Bit 6 = IO 15 state
- Bit 5 = IO 14 state
- Bit 4 = IO 13 state
- Bit 3 = IO 12 state
- Bit 2 = IO 11 state
- Bit 1 = IO 10 state
- Bit 0 = IO 9 state

Second byte:

- Bit 7 = IO 8 state
- Bit 6 = IO 7 state
- Bit 5 = IO 6 state
- Bit 4 = IO 5 state
- Bit 3 = IO 4 state
- Bit 2 = IO 3 state
- Bit 1 = IO 2 state
- Bit 0 = IO 1 state

Note that Disabled IO will still have either an ON or OFF state – dependent on the last state seen for that IO. The user application is responsible for knowing which IO are Enabled and Disabled when using the above response.

The Network Module **Subscribes** to the following topics when it connects to the Broker:

NetworkModule/<devicename>/output/+/set

This Subscribe enables the Network Module to receive output Publish commands from Clients. The + as used above causes the broker to send the module the "set" commands, but won't reflect the modules own IO state messages back to the module (reduces traffic).

NetworkModule/<devicename>/state-req

This Subscribe enables the Network Module to receive the state-req Publish command from Clients

When the Network Module connects to the Broker it will establish a "last will" message of "offline" with the will topic "NetworkModule/<devicename>/availability"

When the Network Module connects to the Broker it will Publish to the following topics:

NetworkModule/<devicename>/availability Payload: online

NetworkModule/<devicename>/input/xx Payload: "ON" or "OFF"

NetworkModule/<devicename>/output/xx Payload: "ON" or "OFF"

Where “xx” is the IO number for the Input or Output.
No Publish occurs for a Disabled IO.

NetworkModule/<devicename>/temp/xx Payload: Temperature in degrees Celsius in the format " 000.0" or "-000.0"
Where “xx” is the Temperature Sensor name from 01 to 05

Home Assistant Auto Discovery Publish messaging:

If the HA Auto checkbox is set to enable the Network Module **Publishes** the following messages at boot time:

For each Output:

homeassistant/switch/<macaddress>/xx/config Payload: see below
homeassistant/binary_sensor/<macaddress>/xx/config Payload: empty

The “binary_sensor” message with an empty payload makes sure that Home Assistant will delete any prior configuration on this IO where the IO was previously defined as an Input.

homeassistant/sensor/<macaddress>/yy/config Payload: empty
This only occurs if IO 16 is defined as an Output. The “sensor” message with an empty payload makes sure that Home Assistant will delete any prior sensor configuration on IO 16 where IO 16 was previously defined for use as a Temperature Sensor connection.

For each Input:

homeassistant/ binary_sensor /<macaddress>/xx/config Payload: see below
homeassistant/switch/<macaddress>/xx/config Payload: empty

The “switch” message with an empty payload makes sure that Home Assistant will delete any prior configuration on this IO where the IO was previously defined as an Output.

homeassistant/sensor/<macaddress>/yy/config Payload: empty
This only occurs if IO 16 is defined as an Input. The “sensor” message with an empty payload makes sure that Home Assistant will delete any prior sensor configuration on IO 16 where IO 16 was previously defined for use as a Temperature Sensor connection.

For each Disabled IO:

homeassistant/ binary_sensor /<macaddress>/xx/config Payload: empty
homeassistant/switch/<macaddress>/xx/config Payload: empty
These “empty payload” messages sure that Home Assistant will delete any prior configuration on this IO.

homeassistant/sensor/<macaddress>/yy/config Payload: empty
This only occurs if DS18B20 mode is Disabled. The “sensor” message with an empty payload makes sure that Home Assistant will delete any

prior sensor configuration on IO 16 where IO 16 was previously defined for use as a Temperature Sensor connection.

In the above topics <macaddress> is the MAC address of the Network Module. The “xx” is the IO number. The “yy” is the Temperature Sensor number. Outputs are defined as “switch” topics, Inputs are defined as “binary_sensor” topics, and Temperature Sensors are defined as “sensor” topics.

Where a Home Assistant Auto Discovery Payload is not empty it takes this form for the “switch” topics:

```
{  
  "uniq_id": "<macaddress>_output_01",  
  "name": "<devicename> output 01",  
  "~": "NetworkModule/<devicename>",  
  "avty_t": "~/availability",  
  "stat_t": "~/output/01",  
  "cmd_t": "~/output/01/set",  
  "dev": {  
    "ids": ["NetworkModule_<macaddress>"],  
    "mdl": "HW-584",  
    "mf": "NetworkModule",  
    "name": "<devicename>",  
    "sw": "<code_revision>"  
  }  
}
```

The above example is for an Output on IO 01.

<macaddress> is replaced with the MAC address of the Network Module as entered on the Configuration page..

<devicename> is replaced with the Name of the Network Module as entered on the Configuration page.

<code_revision> is replaced with the code revision for the firmware programmed into the Network Module.

Where a Home Assistant Auto Discovery Payload is not empty it takes this form for the “binary-sensor” topics:

```
{  
  "uniq_id": "<macaddress>_input_01",  
  "name": "<devicename> input 01",  
  "~": "NetworkModule/<devicename>",  
  "avty_t": "~/availability",  
  "stat_t": "~/input/01",  
  "cmd_t": null  
}
```

```

"dev": {
  "ids": ["NetworkModule_<macaddress>"],
  "mdl": "HW-584",
  "mf": "NetworkModule",
  "name": "<devicename>",
  "sw": "<code_revision>"
}
}

```

The above example is for Input on IO 01.

<macaddress> is replaced with the MAC address of the Network Module as entered on the Configuration page..

<devicename> is replaced with the Name of the Network Module as entered on the Configuration page.

<code_revision> is replaced with the code revision for the firmware programmed into the Network Module.

Where a Home Assistant Auto Discovery Payload is not empty it takes this form for the “sensor” topics:

```

{
  "uniq_id": "<macaddress>_temp_01",
  "name": "<devicename> temp 16",
  "~": "NetworkModule/<devicename>",
  "avty_t": "~/availability",
  "stat_t": "~/temp/01",
  "unit_of_meas": "\xc2\xb0\x43",
  "dev": {
    "ids": ["NetworkModule_<macaddress>"],
    "mdl": "HW-584",
    "mf": "NetworkModule",
    "name": "<devicename>",
    "sw": "<code_revision>"
  }
}

```

The above example is for Temperature Sensor 01.

<macaddress> is replaced with the MAC address of the Network Module as entered on the Configuration page..

<devicename> is replaced with the Name of the Network Module as entered on the Configuration page.

<code_revision> is replaced with the code revision for the firmware programmed into the Network Module.

Notes on Link Error Statistics

Link Error Statistics are accessible only via the http command “`http://IP:Port/66`”. The statistics may be useful to you for determining if Full Duplex works better than Half Duplex in your particular network configuration. As noted in other parts of the manual the normal mode of operation is “Half Duplex”, and you should not need to change that. However, during development it was noticed that the Network Module works better with some Cisco 1 Gbit business class switches if the Network Module and the Cisco switch are manually set to Full Duplex. So, this statistics page can let you compare error statistics over several hours or days to help you decide which configuration works better for you.

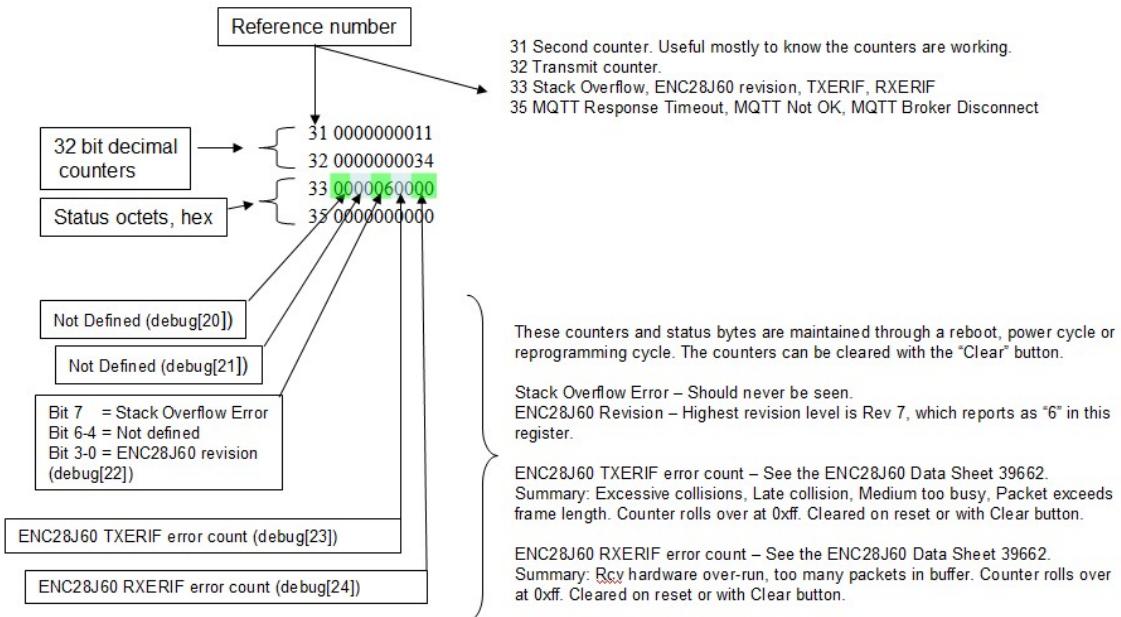
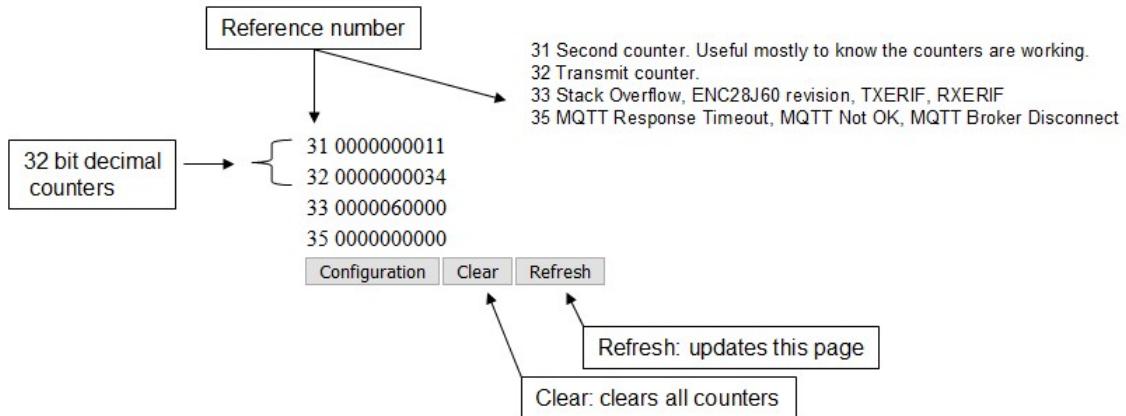
IMPORTANT: If you use “Full Duplex” the ENC28J60 specifications state that you must manually set your switch to “Full Duplex”. This is because the ENC28J60 cannot auto-negotiate Full/Half Duplex. If your switch does not allow you to manually set Full Duplex you should probably just leave the Network Module at Half Duplex. Having said this, experimentation seems to suggest that some switches will work just fine regardless of the Full/Half duplex setting in the Network Module. So ... do your own experiments and use the “/66” command to observe results. But in general I suggest you just leave the module at its default “Half Duplex” mode.

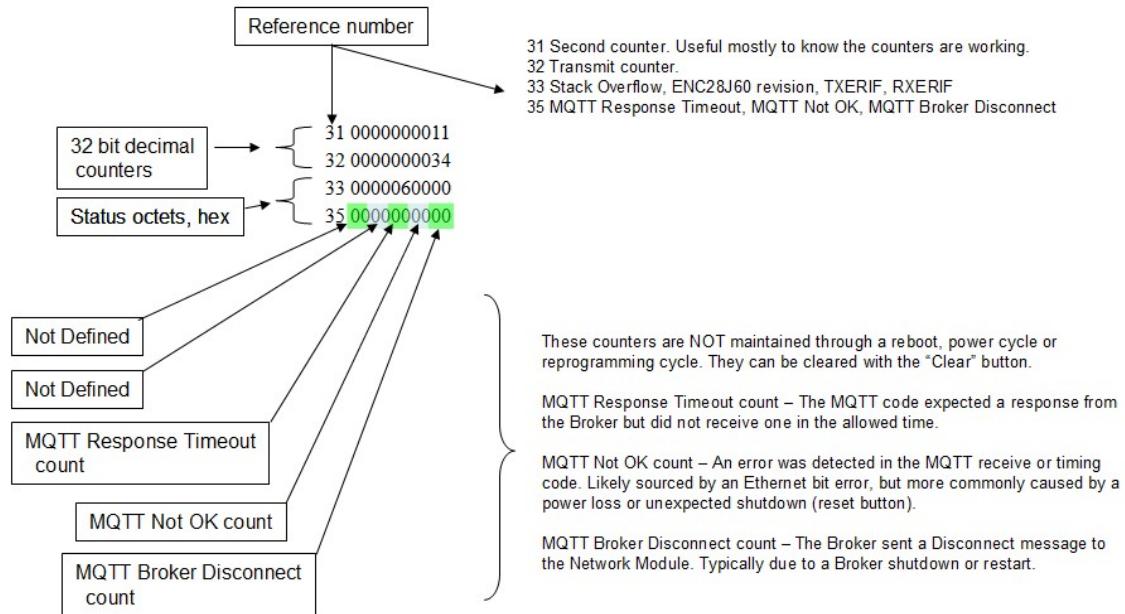
If you enter the “/66” command you’ll get a display similar to this:

```
31 0000002594
32 0000000342
33 0000060000
35 0000000100
  
```

The following explains the above:

Debug Statistics Display Schema





Seconds counter: # of seconds since boot

Transmit counter: # of transmits from the ENC28J60

Stack error: Not a counter, “1” will indicate detection of a stack overflow

ENC28J60 revision: Chip revision. This should indicate “06”, which indicates chip B7.

TXERIF: count of TRXERIF errors (see ENC28J60 documentation)

RXERIF: count of RXERIF errors (see ENC28J60 documentation)

MQTT Response Timeout count: Count of response timeouts in the MQTT code

MQTT Not OK: Count of MQTT Not OK events in the MQTT code

MQTT Broker Disconnect count: Count of MQTT Broker disconnects as detected in the MQTT code

The Error Statistics page is “semi-hidden” and enabled only by the “/66” command because it can be very confusing to the typical user.

The content of the Link Error Stats page is likely to change as the code matures. It was added to assist with testing of MQTT performance using a variety of switches. One critical finding is that the statistics are not completely consistent, likely due to the number of variables affecting the values. Still, they can be useful for relative measurements when determining if you might have unusually high error rates on your network, and/or if you might need to experiment with the Half / Full Duplex setting.

The RXERIF error indicates that the ENC28J60 experienced a receive buffer overflow condition. This likely indicates extremely high network traffic. The impact is that packets received at the Network Module may be dropped. It is not unusual to have a few of these errors over time.

The TXERIF error indicates that a transmit abort has occurred. This error can occur as a result of the following;

- 1) Excessive Ethernet packet collisions
- 2) Late collisions
- 3) Transmission was unable to occur because the medium was occupied for too long.

It is normal to see transmit or receive errors, just not "too many", which is somewhat arbitrary. On a LAN with well behaving clients and good cabling you may only see one error every few months. Communication over WiFi, WAN, or Internet will see much higher error rates. And the error rate depends on how much traffic is occurring. Some users may never see an error indication, which is great. Others may see lots of errors, which may be 'normal' or may indicate a problem depending on their specific environment.

These counters are included because they were useful during development to help determine that Full Duplex worked much better than Half Duplex on the Cisco managed switches. With Half Duplex a TXERIF error was occurring several times per day, accompanied by an MQTT disconnect (and automatic reconnect). Once Full Duplex was enabled zero errors were seen for several weeks in the Cisco configuration.

On the other hand, TXERIF errors were seen when Full Duplex was used with some unmanaged switches, and no errors when those same switches were used with Half Duplex.

Your specific configuration and conditions may require some experimentation to get the best result.

Note: The "Seconds since boot" counter is approximate. The Network Module does not have a highly accurate clock, and there will be an accumulating deviation from real time, particularly if the counter is run for a long period. But, it is close enough for this purpose.

Functional Limitations

The code space and RAM in the processor on the Network Module is extremely limited, so there are many functional limitations that you would not expect on a device without these constraints. Some of the limitations to be aware of:

Maximum number of TCP Connections:

Maximum number of TCP Connections: 4. Implications: Each browser session and MQTT connection requires a TCP Connection. If you are running a non-MQTT build of the code you could connect up to 4 browsers to the device at one time. If you are running an MQTT build of the code you could also have up to 3 browser sessions connected.

Multiple browsers connected at the same time:

Multiple browsers connected at the same time CAN interfere with each other. For instance, multiple browsers attempting to make configuration changes at the same time can cause unexpected results, particularly if Save is clicked on both browsers at the same time. I recommend you select ONE browser to make configuration changes, and the other browsers should be used for monitoring. Or at least make sure you only make configuration changes on one browser at a time.

REST command rate:

Using the REST commands with a high repetition rate may slow the response time of the Network Module to the point that the browser interface becomes unusable. A suggestion for high repetition rates is to use the MQTT interface instead as it is more efficient than the HTML interface used by REST commands. Even so, you can push enough MQTT commands that the browsers might be unusable.

MQTT SSL/TSL:

MQTT does not support SSL/TSL. There is insufficient code space to implement this functionality.

Overall Command rate:

Processing speed is very limited given the functions implemented, so I imagine it will be easy to over-run the Network Module with state change requests. The code is single threaded, so whatever function has been requested must be completed before the next can be addressed. More testing needs to be done to determine if packets are simply dropped (if too many received) or if there are cases where the module may stop functioning. So far I haven't seen a "stopped functioning" scenario. If that were to occur a power cycle may be the only recovery option.

Configuration Errors:

There are very few "warnings" in the code to keep the user from creating bad configurations. The most concerning is that if you enable "Retain" for the power

cycle output states AND you subject the device to rapid output state changes you run the risk of wearing out the EEPROM. Other situations likely only cause the device to lose contact with browsers or MQTT brokers (like mis-configuring IP addresses or Port numbers).

- IMPORTANT: “Retain” was ON by default in the early code releases. The default “Boot State” is now “OFF” to aid in preventing a user from inadvertently wearing out EEPROM. The original usage scenario was anticipated to be one in which output state changes would occur only via human interaction with a browser (therefore “infrequent changes”). Subsequent users have started implementing increasing automation, with some saying they may create many output changes per hour forever. In those “frequent output change” scenarios I strongly recommend using the OFF or ON “Boot State” setting instead of “Retain” to eliminate the EEPROM wearout concern.

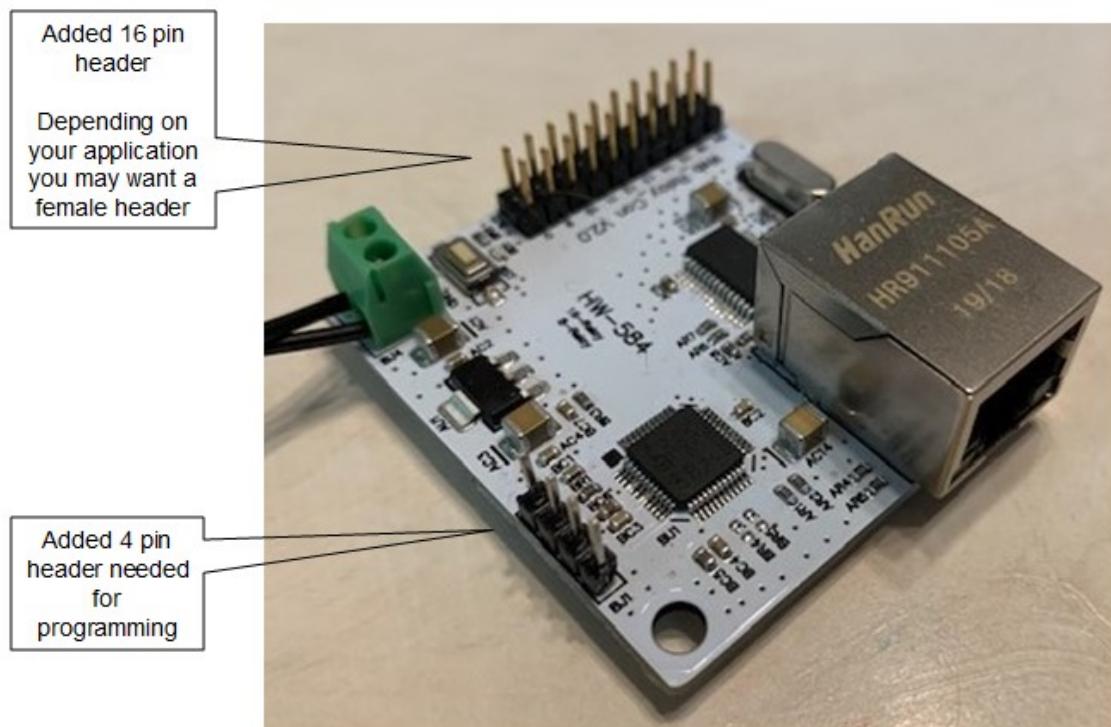
Programming the Module

Assuming you have the Web_Relays_Con V2.0 HW-584 and all you want to do is apply this firmware the following describes the process.

IMPORTANT NOTE: In the steps below you'll turn off the Read Out Protection bit on the Network Module. This will ERASE the program currently in the device. It will only work again after you successfully reprogram it. DO THIS AT YOUR OWN RISK.

Note that as of January 2021 I've gone to one release of the firmware that covers the functionality of the four previous parallel releases.

1) Prepare your Network Module: Install a 4 pin header on the board (see photo)



2) Buy the Programmer: Purchase a ST-Link V2 (see photo). If you are patient you can get one from China in about a month for about \$3.50. Or in less than a week from within the US for about \$6.00 (assuming you are in North America). Price estimates are as of June 2020. Search on Google, Amazon, eBay, etc.

The ST-Link V2 is required to reprogram the Network Module. It is a USB to SWIM interface module supported by free software from STMicroelectronics. You'll need a four wire Dupont cable if you don't already have one. Some sellers ship the module with a cable. The Dupont cable is just a simple four wire cable with female push connectors on each end (as shown in the photo below).

The ST-Link V2 modules come in several colors so pick the color you like.



3) Obtain and Install Free Software: All of my development work was on the Windows 10 OS. If you are using Linux you will have a little more homework to do on your own, but I don't think there is much difference. For Windows you'll need to download and install the following files:

en.stsw-link009.zip

You'll find the above at <https://www.st.com/en/development-tools/stsw-link009.html>

en.stvp-stm8.zip

You'll find the above at <https://www.st.com/en/development-tools/stvp-stm8.html>

You'll need to create an account at st.com to get the above software. It's free but they want an email address to contact you. When you try to download the software you'll be asked for your account credentials and given the option to create an account. By providing my email address I've gotten some invitations to online programming seminars but otherwise no spam. Not much hassle.

The stsw-link009 software is the driver to operate the ST-Link V2.

The stvp-stm8 software is a development utility and the programmer specific to the STM8 processor. When you install en.stvp-stm8 you'll get two programs:

- 1) ST Visual Develop
- 2) ST Visual Programmer (STVP)

I only used STVP even when developing the code. And if you are only reprogramming your devices STVP is the only tool you'll need.

4) Copy the Program: Now that you've installed the necessary software you need to copy the STVP Project file and the Binary file from GitHub that will be programmed into the Network Module.

NOTE: Beginning with the February 20, 2021 release you should obtain source code and executable files from the "Release" part of the GitHub web page (along the right margin of the page).

On my Windows 10 machine the project was located in the following directory:

```
C:/Users/Mike/Documents/COSMIC/FSE_Compilers/CXSTM8/NetworkModule
```

If you locate your copy of the project files in a similar Documents file location this should minimize the tinkering you have to do. And should you decide to modify the program you'll already have an appropriate directory set up.

The STVP programmer needs a ".stp" and ".sx" file pair to program the Network Module. Now that we have one code set to cover all the previous functionality you'll only need to copy the following files into the Documents directory you created above:

NetworkModule.stp - The STVP project file
NetworkModule.sx - The NetworkModule binary file

You will find these files in the "Release" section of the GitHub web page (along the right margin of the page). The above are the only files you need to copy from the GitHub project account if you only want to program your module and you are not jumping right into code modifications.

IMPORTANT1: Since the path to your "Documents" directory will be different than mine (if for no other reason than your user ID is different than "Mike), you may need to **edit the .stp file to match your directory path**. Open the .stp file with NotePad or NotePad++ and look for the following. Edit it to match the path to your .sz file.

```
30 File0=C:\Users\Mike\Documents\COSMIC\FSE_Compilers\CXSTM8\NetworkModule\NetworkModule.sxCRLF
```

IMPORTANT2: Later releases of the code have already modified the .stp file so that you should not need to edit it. If you find the following in the .stp file you only need to make sure that the .stp file and the .sx file are in the same directory:

```
30 File0=NetworkModule.sxCRLF
```

I use NotePad++ and have it set to show the CR/LF at the end of the line. If you use NotePad as your text editor you won't see that.

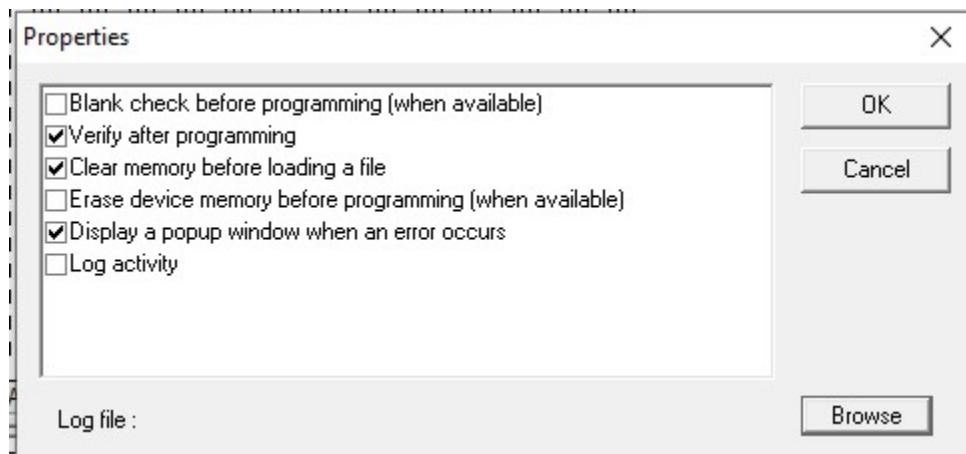
Telling STVP where your files are:

Since your User name on your Windows machine is probably not "Mike" you'll need to start STVP, click on "**Project/Open**", and browse for the .stp file that you copied to your **Documents/...** directory. Once you open the project file STVP should automatically load the .sx file from that same directory.

Setting up ST-Link Communication:

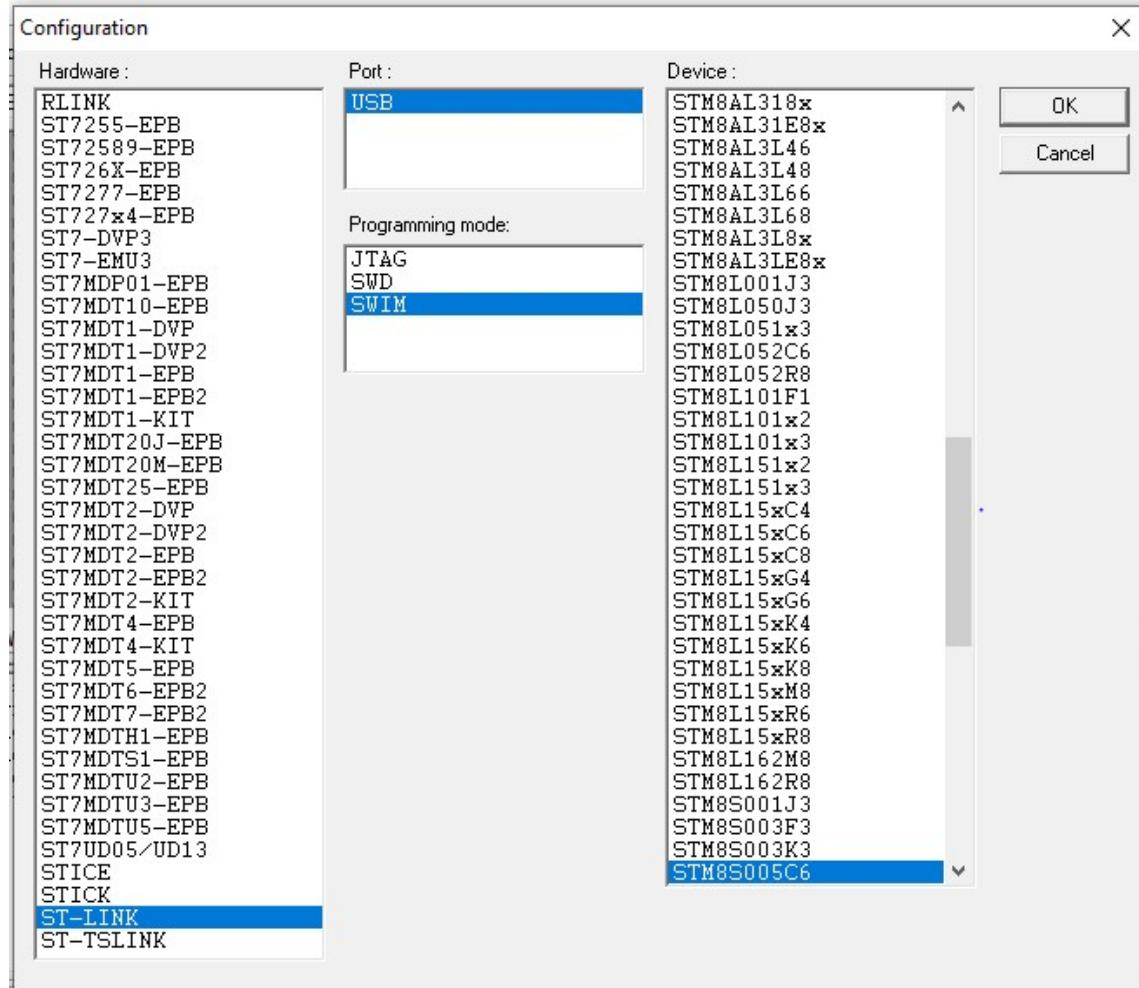
The project file contains various settings that enable the ST-Link V2 to communicate with your target board. They should already be set for you, but just in case the following is how I had them set:

Under “Edit/Preferences”:



(Continued)

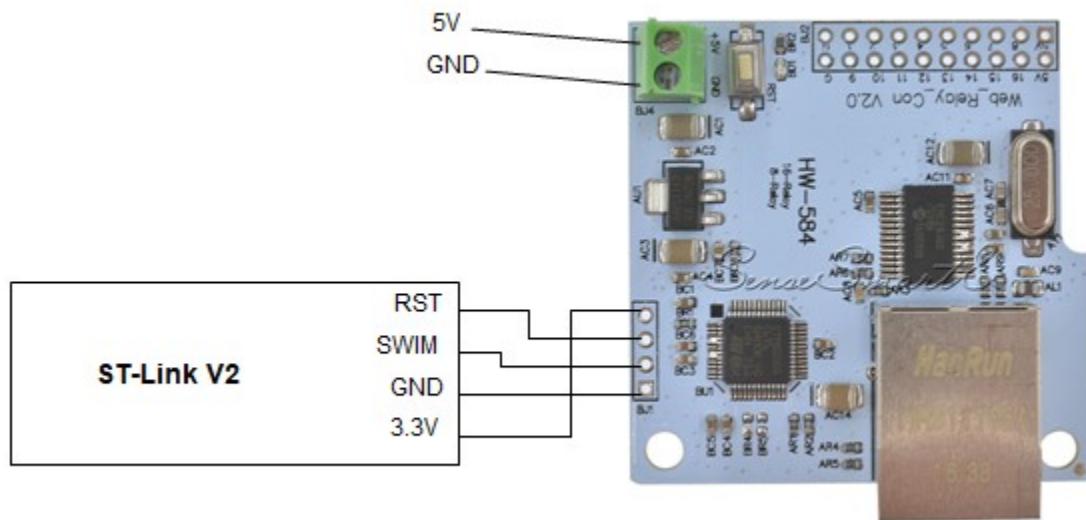
Under “Configure/Configure ST Visual Programmer”



If the above looks OK you are ready to program the Network Module.

Setting up the Hardware to allow programming:

First, attach the ST-Link V2 to your Network Module as follows:



Apply power to your Network Module. You should be using a 5V power supply connected to the power pins on the Network Module.

Plug the ST-Link V2 into your PC USB port.

If STVP is not already running, start it now.

If the NetworkModule.stp project is not already loaded, load it now (click on "Project/Open", and browse for the .stp file that you copied to your **Documents/...** directory). Give it 10 or 20 seconds to load the .sx file.

If you see “out of range” messages like the following this is NOT an error. It would have been nice if the messages were more informative, but they are just telling you that the indicated addresses are in non-programmable areas of the chip during program load. The addresses shown are typically in EEPROM and RAM.

```
> Loading file C:\Users\Mike\Documents\COSMIC\FSE_Compilers\CXSTM8\MQTT-Network
FILE : line 961: Address 0x4000 is out of range and is ignored!
FILE : line 961: Address 0x4001 is out of range and is ignored!
FILE : line 961: Address 0x4002 is out of range and is ignored!
FILE : line 961: Address 0x4003 is out of range and is ignored!
FILE : line 961: Address 0x4004 is out of range and is ignored!
FILE : line 961: Address 0x4005 is out of range and is ignored!
FILE : line 961: Address 0x4006 is out of range and is ignored!
```

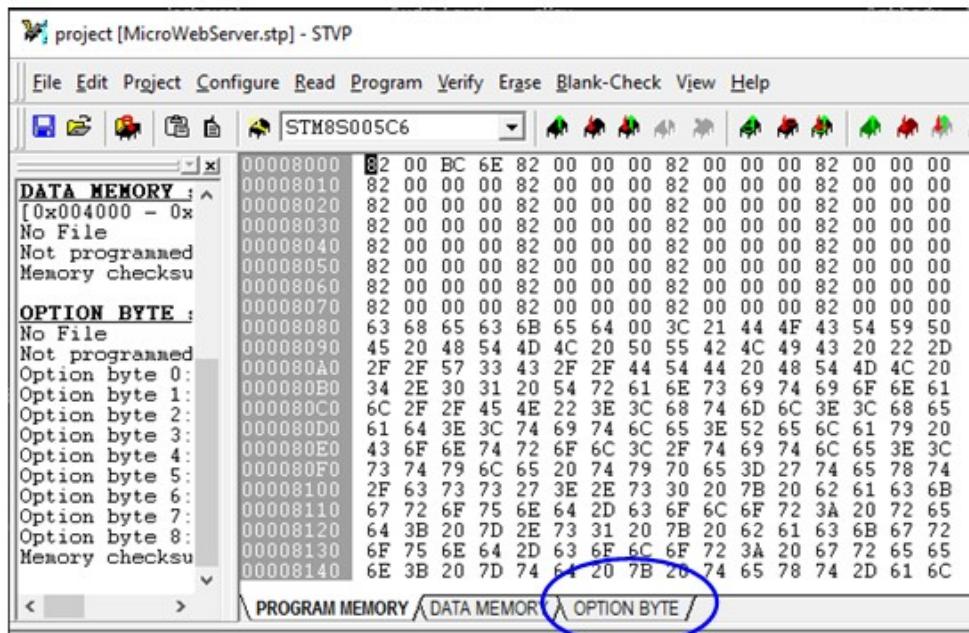
Once the program is successfully loaded in the programmer you will see a message like this (although the checksum will likely be different than what you see here).

```
FILE : line 964: Address 0xD is out of range and is ignored!
FILE : line 964: Address 0xE is out of range and is ignored!
FILE : line 964: Address 0xF is out of range and is ignored!
< File successfully loaded. File Checksum 0x297EE9
```

Clear the ROP Bit:

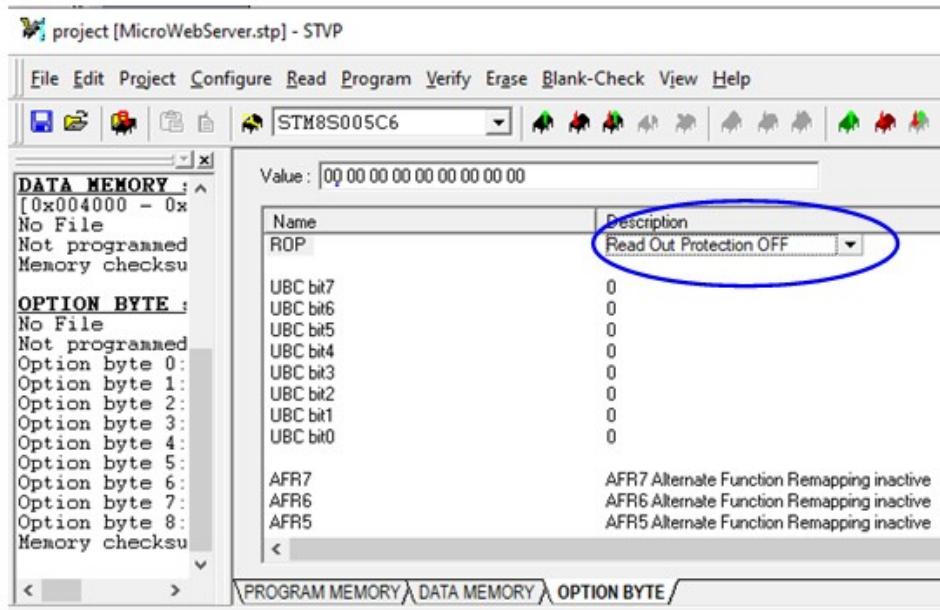
If this is the first time you are programming your Network Module you will need to clear the Read Out Protection (ROP) bit. If you don't clear the ROP any attempt to program the Network Module will give you a "This device is protected" message. How to clear the ROP bit:

In the STVP main window click on the “Option Byte” tab



(Continued)

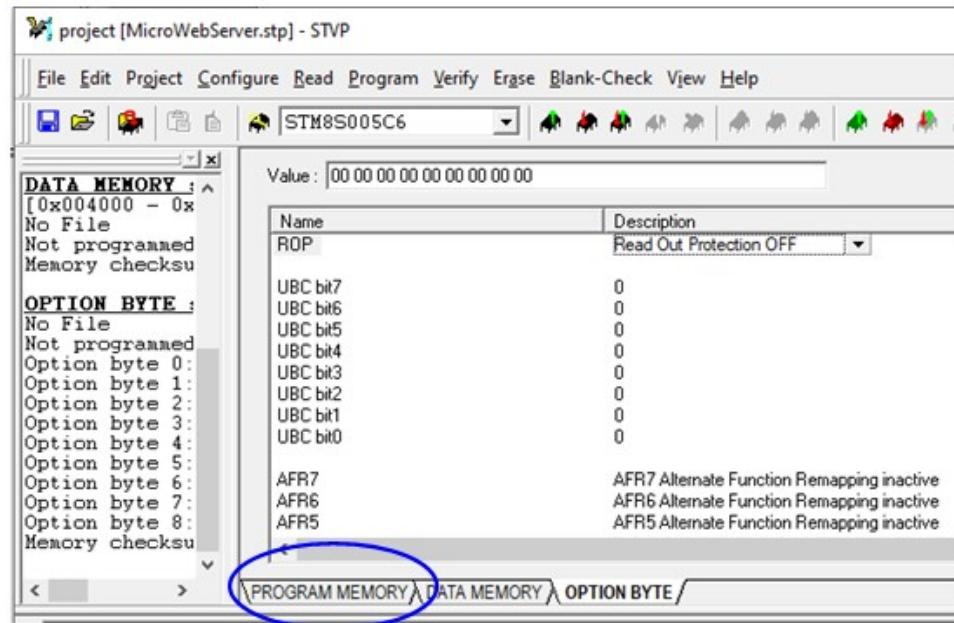
Make sure “**Read Out Protection OFF**” is selected in this drop down.



Next click on “**Program / Current Tab**”. This will clear the ROP bit and allow you to reprogram the device. **IMPORTANT: CLEARING THE ROP BIT ERASES THE CODE IN THE NETWORK MODULE.** After you clear the ROP bit you **MUST** reprogram the Network Module to make it useful again.

Programming the Device:

Select the Program Memory tab



Next select “Program / Current Tab”

If you got an error message while attempting to program the Network Module:

- a) Make sure the RST connection is in place.
- b) Make sure the power supply connected to the Network Module is providing 5V.
- c) Make sure you have good connections from the ST-Link V2 to the Network Module.
- d) You might have to unplug the ST-Link V2 from the USB port on your PC and plug it back in again.
- e) You might have to stop the STVP program, unplug and replug the ST-Link V2, then restart the STVP program.
- f) If you have 16 relays connected to your Network Module I suggest disconnecting them while reprogramming. If you have a very robust power supply it may be possible to leave them connected. The Network Module will be reset a couple of times during programming, and this may cause the relays to simultaneously turn on and off. Whether this interferes with programming depends on whether your power supply can handle the surge caused by the relay coils.

Generally I haven't had to do any of the above as I seldom saw an error. But on occasion I saw an error message that the link was not working, and the above tinkering got it working again.

(Continued)

If you see a message indicating programming success you are ready to attempt to connect to the Network Module via the Ethernet connector.

- a) Disconnect the RST wire between the ST-Link V2 and the Network Module. You can also disconnect the other wires, or leave them connected for the time being.
- b) Connect the Ethernet cable. I suggest you do this the first time without using your network. Make a direct Ethernet cable connection from the Network Module to your PC and attempt to access it at 192.168.1.4:8080. If the connection does not work check your IPV4 Ethernet settings on the PC and set it to use IP address 192.168.1.100 (not DHCP). If you don't know how to do this Google it. Here's a helpful link:

<https://stevessmarthomeguide.com/setting-up-static-ip-address-windows-10/>

While the device is directly connected to your PC you can use your browser to make address setting changes on the Network Module that are appropriate to your network. Then you can connect the device to your network, return your PC to its original Ethernet settings, and attempt to access the device.

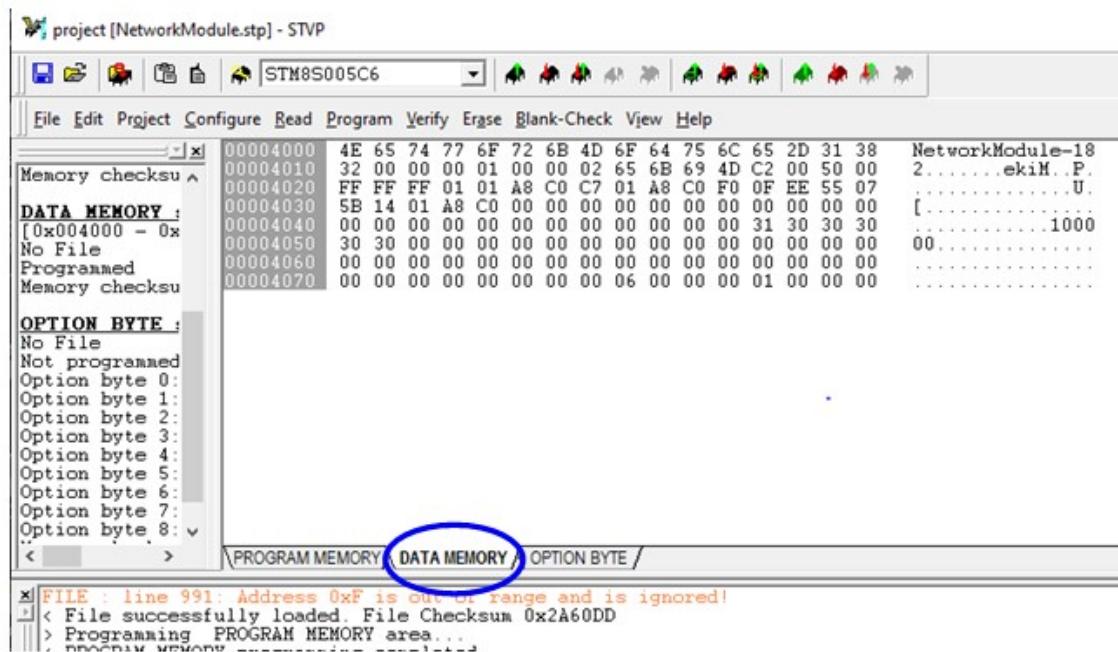
Note: See the section “Alternative Way to Set Initial IP Address”.

Alternative Way to Force Defaults or Downgrade Firmware

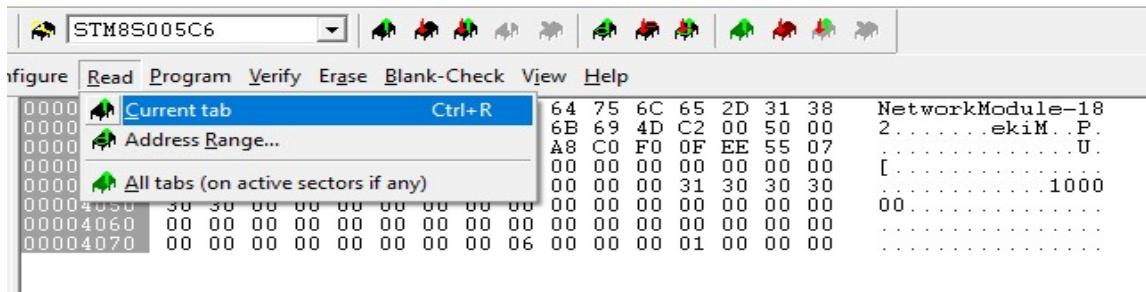
Normally all you have to do to return to “factory defaults” is press the Reset button on the Network Module board for 10 seconds. However, during development there were a couple of times where pressing the Reset button did not revive the Network Module to a point that it would operate. Admittedly this was due to “in development” code errors and may not be needed, but I’m providing it just in case.

This information is only useful if you are going to do your own development from this code. I DO NOT RECOMMEND THIS METHOD UNLESS ALL ELSE HAS FAILED TO REVIVE THE DEVICE.

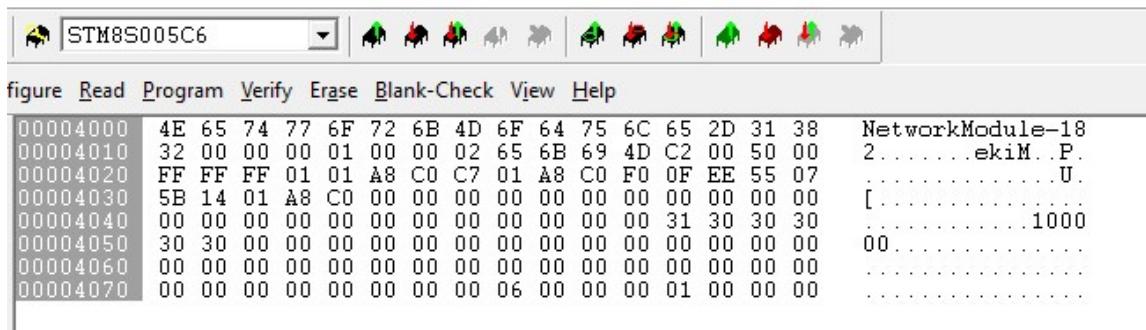
- 1) When the Network Module is connected to the STLink (including the Reset wire) you can access the EEPROM content with the “Data Memory” tab.



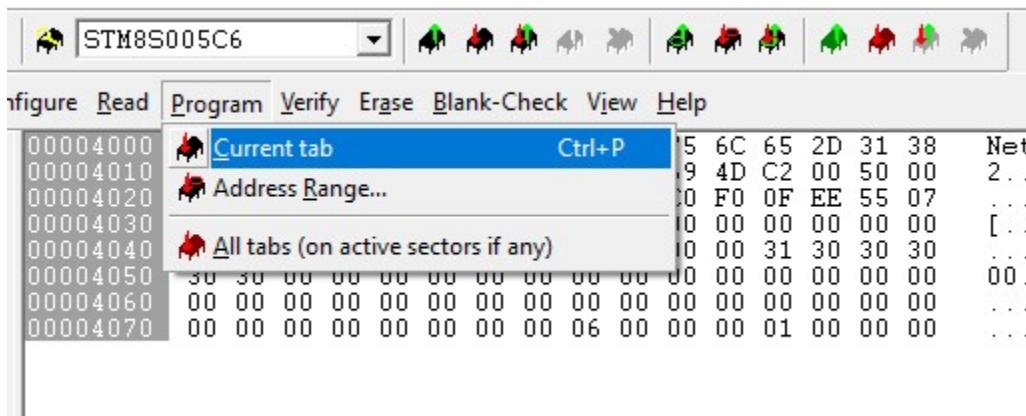
- 2a) After clicking on the Data Memory tab click on Read / Current tab to read the EEPROM contents.



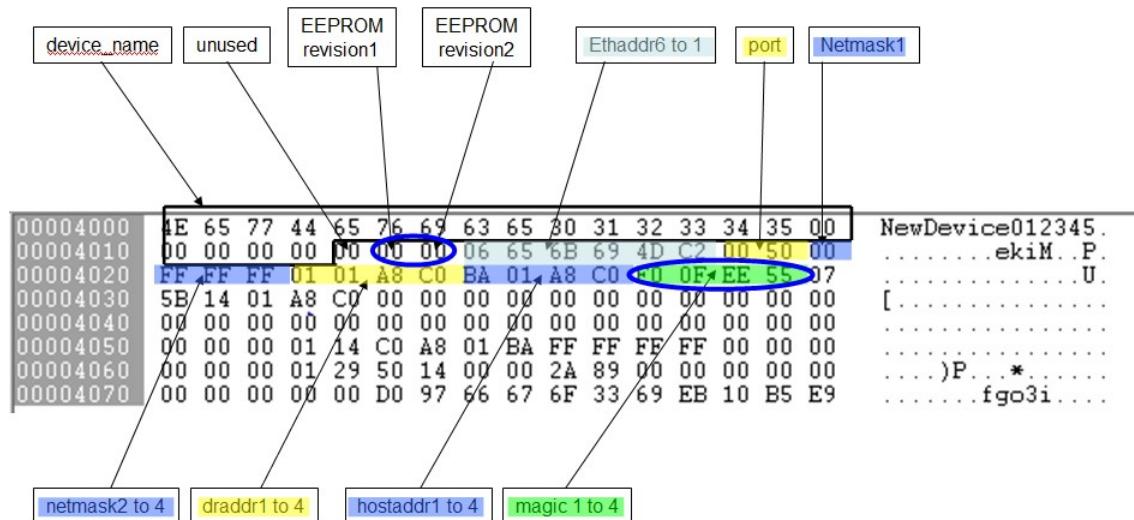
You'll get a display that looks something like this:



- 2b) You can change any value in the Data Memory one character at a time, then you can write the result to the EEPROM with the Program / Current tab selection.



- 2c) To force the EEPROM to a state where firmware must start over again set the bytes circled in blue to 0.



After committing to EEPROM reboot the device (power cycle or reconnect the reset wire, wait 2 seconds, and disconnect the reset wire).

- 2) You can now disconnect the STLink and connect the Network Module to your network. Once connected you should be able to use a browser to connect to the Network Module via the factory default address 192.168.1.4:8080.

FIRMWARE DOWNGRADE

Once you've cleared the EEPROM content as shown above you should be able to load a prior release of the firmware should you need to do so.

Alternative Way to Set Initial IP Address

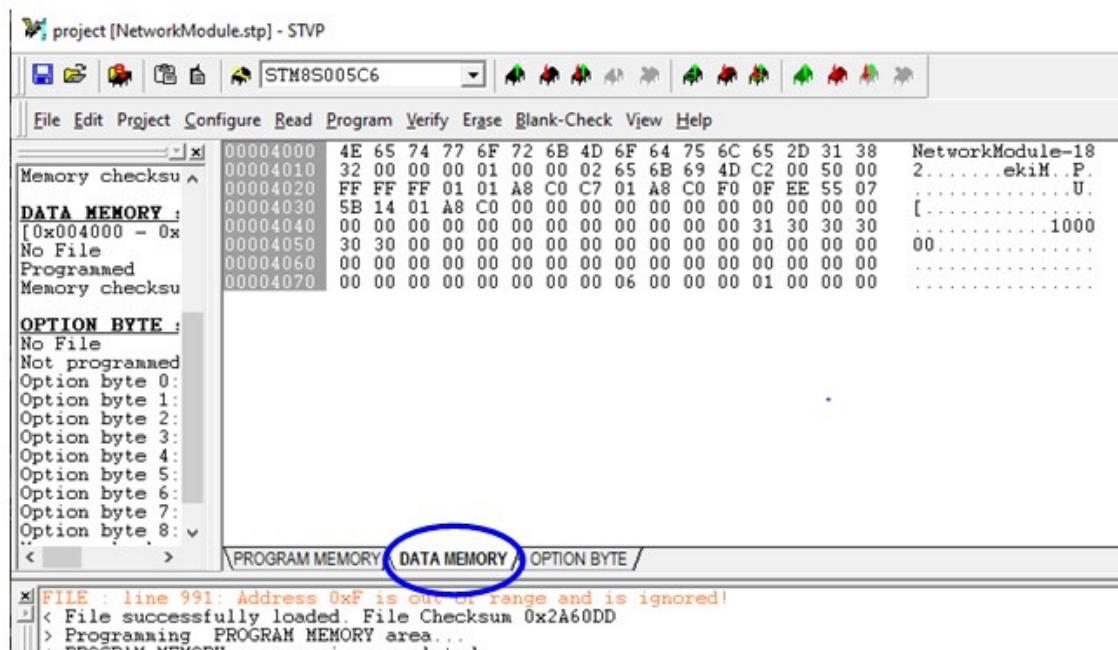
A user commented that it was cumbersome to have to set up a laptop with a fixed IP Address to program the Network Module with its first “network compatible address”. Here’s an alternative that may be useful to you.

Let’s say your network already uses 192.168.1.4, so you can’t attach the device directly to your network. Or perhaps your network uses some other variant of the 192.168.xxx.xxx address range, or even the 10.0.0.x address range. A way to work around this without needing to set up a laptop or PC for the initial Ethernet connection to the Network module as follows;

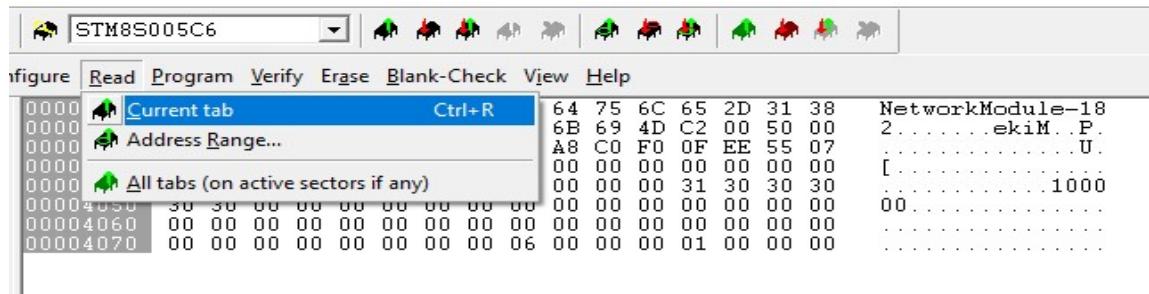
- 1) Assumption is that you successfully programmed the flash with the Network Module code. Reboot the Network Module (usually just be releasing the reset wire). Then reconnect the reset wire.
- 2) Using the STLink change the IP address in the EEPROM. Reboot the Network Module.
- 3) Attach the Network Module to your network, access the Network Module with a browser, and finish changing any settings via the Configuration menu.

Here’s an illustrated version of the above with greater detail;

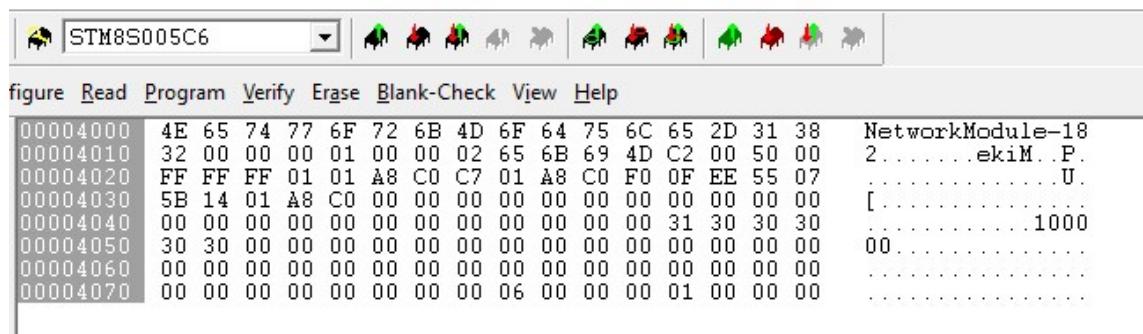
- 3) Step 1 is covered in the section “Programming the Module”
- 4) When the Network Module is connected to the STLink (including the Reset wire) you can access the EEPROM content with the “Data Memory” tab.



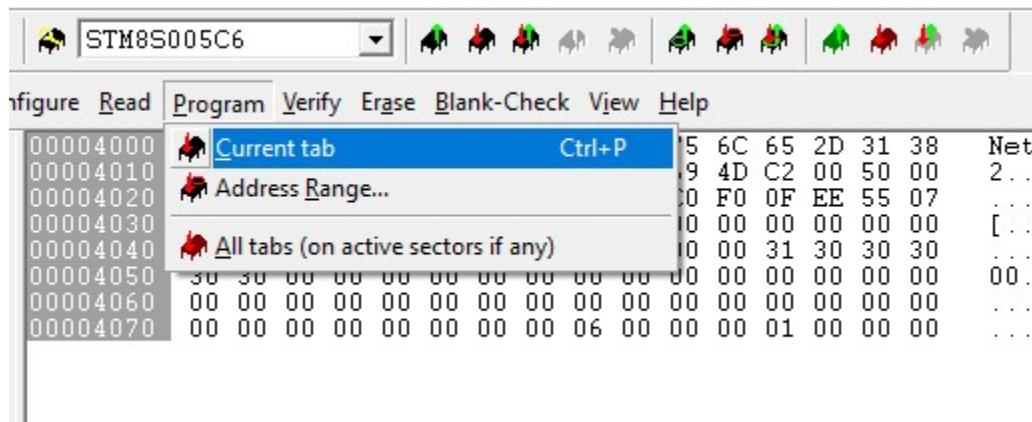
- 2a) After clicking on the Data Memory tab click on Read / Current tab to read the EEPROM contents.



You'll get a display that looks something like this:

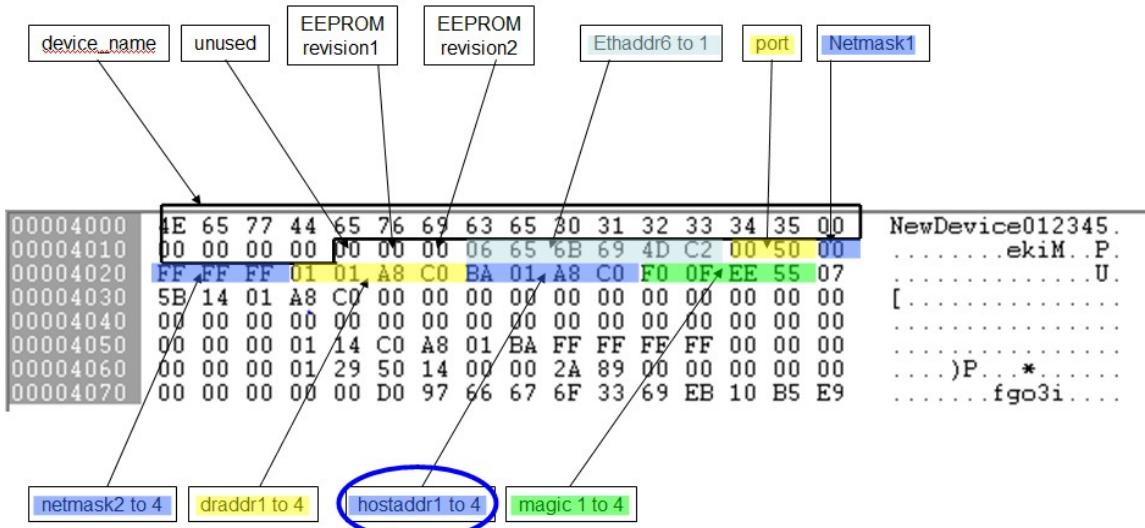


- 2b) You can change any value in the Data Memory one character at a time, then you can write the result to the EEPROM with the Program / Current tab selection.



- 2c) SO Which value do you want to change? You can change any of the values, but typically you only need to change the IP Address of the module to get it to appear on

your network. Looking at the EEPROM map the below shows where the IP Address is located.



“hostaddr1 to 4” is the Network Module IP Address. Note that it is in hex, and it is in reverse order (MSB on the right, LSB on the left). In decimal format the address shown is 192.168.1.186. Click on any character to change it, then be sure to use Program / Current tab to commit the changes to EEPROM.

After committing to EEPROM reboot the device (power cycle or reconnect the reset wire, wait 2 seconds, and disconnect the reset wire).

- 5) You can now disconnect the STLink and connect the Network Module to your network. Once connected you should be able to use a browser to connect to the Network Module and make any further changes you need in the Configuration page.

IMPORTANT NOTES:

- a) You should not use the “direct access to the EEPROM” method for anything other than the minimum needed to gain access via a browser. Usually you only need to change the Network Module IP Address (hostaddr). It’s too easy to make a mistake ... so don’t forget about the reset button if you mess it up.
- b) If you press the reset button on the Network Module it will return to the hard-coded defaults, NOT to the changes you manually put in the EEPROM. You’ll have to go through this process again to get back to a network compatible IP address.
- c) If you change the “Magic Number” it will cause a return to factory defaults on reboot.

Display Values vs Pin Logic Levels

This information may be useful to you for understanding how the values displayed in the browser or contained in the MQTT fields correspond to output and input pin voltage levels.

16 Outputs			State Values Reported in Browser	State Values reported in Short Form (/99)	Pin numbers as marked on board									Outputs						
Outputs	Invert Setting	All On			Pin16	Pin15	Pin14	Pin13	Pin12	Pin11	Pin10	Pin9	Pin8	Pin7	Pin6	Pin5	Pin4	Pin3	Pin2	Pin1
All Off	All low (L)	No Inv	All Red	0000000000000000	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
1 On	Out1 high (H)	No Inv	#1 Green	1000000000000000	L	L	L	L	L	L	L	L	L	L	L	L	L	L	H	
8 On	Out8 high (H)	No Inv	#8 Green	0000000100000000	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
All Off	All high (H)	Inv	All Red	0000000000000000	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	
All On	All low (L)	Inv	All Green	1111111100000000	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
1 On	Out1 low (L)	Inv	#1 Green	1000000000000000	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	
8 On	Out8 low (L)	Inv	#8 Green	0000000100000000	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	
H = +3V logic level			Note that Pin 1 is the					H = +3V logic level					L = 0V logic level							

8 Out / 8 In			State Values Reported in Browser	State Values reported in Short Form (/99)	Pin numbers as marked on board									Outputs						
Inputs	Invert Setting	All On			Pin16	Pin15	Pin14	Pin13	Pin12	Pin11	Pin10	Pin9	Pin8	Pin7	Pin6	Pin5	Pin4	Pin3	Pin2	Pin1
All Off	All low (L)	No Inv	All Red	xxxxxxxx00000000	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
1 On	In1 high (H)	No Inv	#1 Green	xxxxxxxx10000000	L	L	L	L	L	L	L	L	H	L	L	L	L	L	H	
8 On	In8 high (H)	No Inv	#8 Green	xxxxxxxx00000001	H	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
All Off	All high (H)	Inv	All Red	xxxxxxxx00000000	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	
All On	All low (L)	Inv	All Green	xxxxxxxx11111111	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	
1 On	In1 low (L)	Inv	#1 Green	xxxxxxxx10000000	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	
8 On	In8 low (L)	Inv	#8 Green	xxxxxxxx00000001	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	
H = +3V logic level			Note that Pin 1 is the					H = +3V logic level					L = 0V logic level							

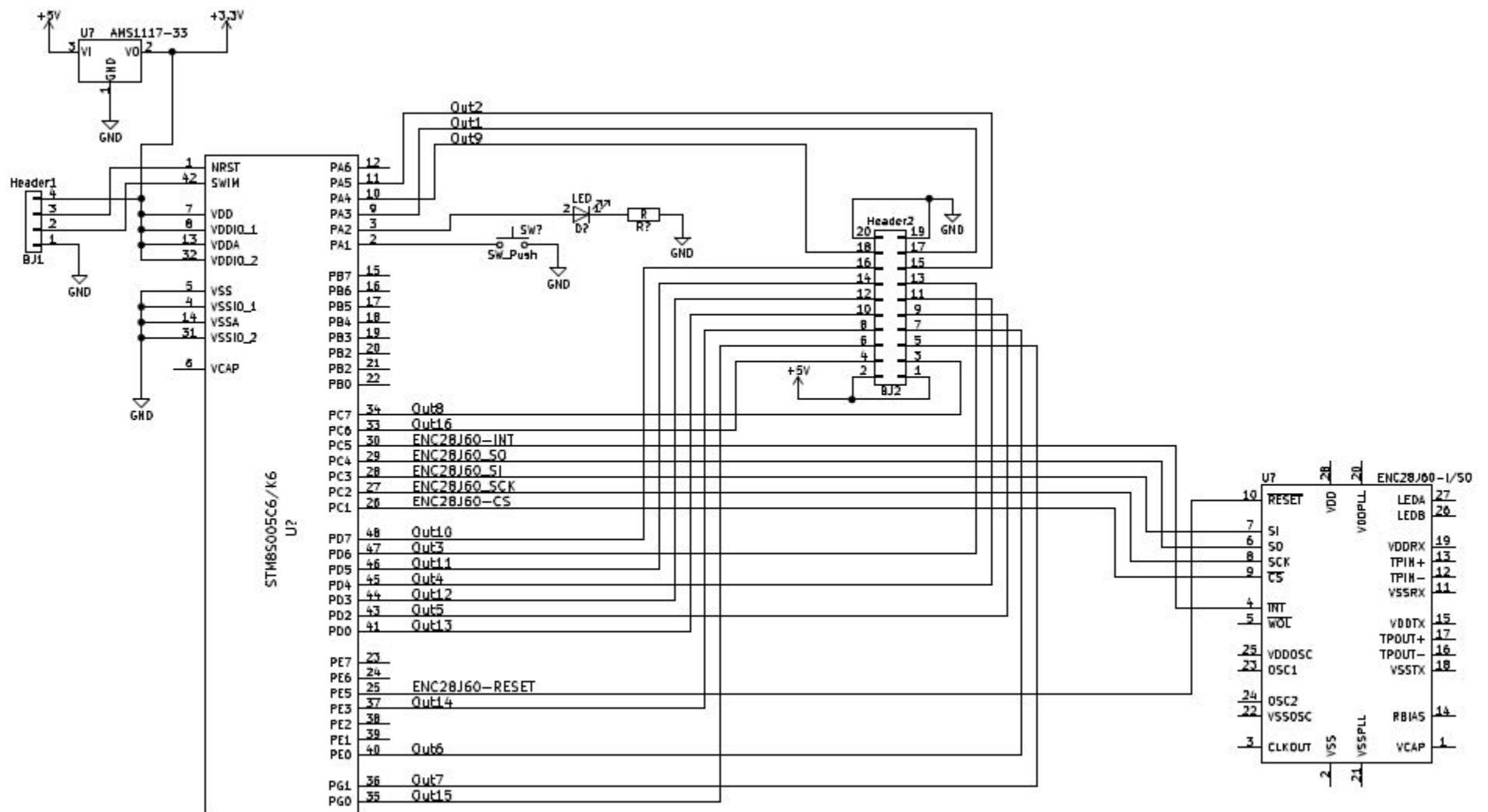
16 Inputs			State Values Reported in Browser	State Values reported in Short Form (/99)	Pin numbers as marked on board															
Inputs	Invert Setting	Outputs			Inputs								Inputs							
		Pin16	Pin15	Pin14	Pin13	Pin12	Pin11	Pin10	Pin9	Pin8	Pin7	Pin6	Pin5	Pin4	Pin3	Pin2	Pin1			
		All On	All high (H)	No Inv	All Green	1111111111111111	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L		
		All Off	All low (L)	No Inv	All Red	0000000000000000	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L		
		1 On	In1 high (H)	No Inv	#1 Green	1000000000000000	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L		
		8 On	In8 high (H)	No Inv	#8 Green	0000000100000000	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L		
		All Off	All high (H)	Inv	All Red	0000000000000000	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H		
		All On	All low (L)	Inv	All Green	1111111111111111	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	
		1 On	In1 low (L)	Inv	#1 Green	1000000000000000	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	
		8 On	In8 low (L)	Inv	#8 Green	0000000100000000	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	
H = +3V logic level			Note that Pin 1 is the left most bit, Pin 16 is the right most bit								H = +3V logic level									
L = 0V logic level											L = 0V logic level									

MQTT 8Out / 8in			State Values Reported in MQTT	State Values reported in Short Form (/99)	Pin numbers as marked on board																
Inputs	Invert Setting	Outputs			Inputs								Outputs								
		Byte 0 Inputs	Byte 1 Relays	Pin16	Pin15	Pin14	Pin13	Pin12	Pin11	Pin10	Pin9	Pin8	Pin7	Pin6	Pin5	Pin4	Pin3	Pin2	Pin1		
		All On	All high (H)	No Inv	FF	All Green	xxxxxxxx11111111	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H			
		All Off	All low (L)	No Inv	00	All Red	xxxxxxxx00000000	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L			
		1 On	In1 high (H)	No Inv	01	#1 Green	xxxxxxxx10000000	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L			
		8 On	In8 high (H)	No Inv	80	#8 Green	xxxxxxxx00000001	H L L L L L L L	H L L L L L L L	H L L L L L L L	H L L L L L L L	H L L L L L L L	H L L L L L L L	H L L L L L L L	H L L L L L L L	H L L L L L L L	H L L L L L L L	H L L L L L L L			
		All Off	All high (H)	Inv	00	All Red	xxxxxxxx00000000	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H			
		All On	All low (L)	Inv	FF	All Green	xxxxxxxx11111111	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L	L L L L L L L L			
		1 On	In1 low (L)	Inv	01	#1 Green	xxxxxxxx10000000	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H	H H H H H H H H			
		8 On	In8 low (L)	Inv	80	#8 Green	xxxxxxxx00000001	L H H H H H H H	L H H H H H H H	L H H H H H H H	L H H H H H H H	L H H H H H H H	L H H H H H H H	L H H H H H H H	L H H H H H H H	L H H H H H H H	L H H H H H H H	L H H H H H H H			
All On			H = +3V logic level								Note that Pin 1 is the left most bit, Pin 16 is the right most bit								H = +3V logic level		
All Off			L = 0V logic level								L = 0V logic level								L = 0V logic level		

Network Module Schematic

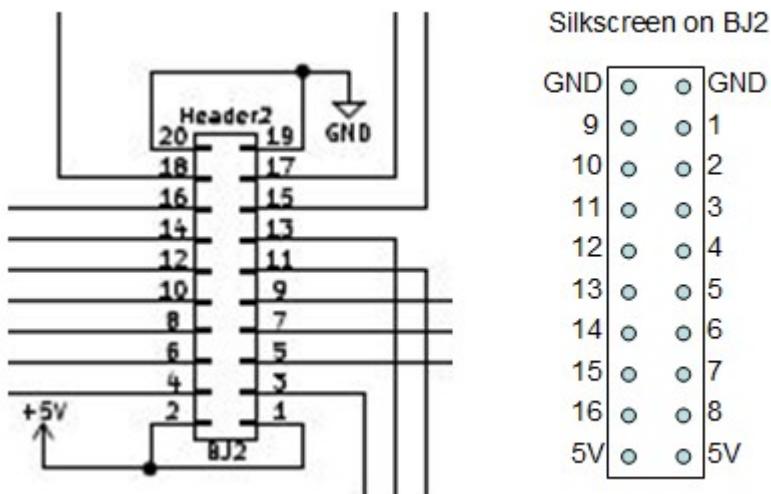
I traced out the parts of the Network Module that are pertinent to developing the new software. I did not trace ALL connections as my intention was not to reverse engineer the hardware design. My intention was only to fix the inadequate function of the software. The schematic may be useful should you decide to improve on the software I've provided. Some notes:

- There are a number of capacitors connecting power and ground. These are left out of the schematic.
- The VCAP pin on the processor was not traced.
- Unused pins or pins that did not appear to be a necessary part of the functionality were programmed to be inputs with pull-ups. These are shown as disconnected on the schematic even if there was a component attached.
 - o There are some components connected to the Port B pins. I suspect the original code used these to identify if the board was "8 port" or "16 port".
- I didn't trace out most of the pins on the ENC28J60, as I knew the design worked and did not need to do any modifications. Some notes:
 - o The SPI interface on the ENC28J60 is not connected to the SPI interface on the STM8S005. Ordinary port pins on the STM8S005 are used to "bit bang" the SPI interface. Not very fast, but this is not an Ethernet performance design so it works just fine.
 - o The -WOL pin does not appear to be connected.
 - o The CLKOUT pin is not connected.
- If you dig into the STM8S005 specification you'll find that most pins that I show simply as "port pins" can be defined for other uses. I didn't include all that information in the component drawing as it just creates confusion in this context. The Network Module uses all the pins as "port pins", so that is all I show.
- The STM8S005 operates on its internal 16MHz clock. It does not have an external crystal or clock source.



Pinouts

Schematic representation of the connection header vs silkscreen on the board;



Following are the pin definitions for the firmware configurations.

Silkscreen on BJ2

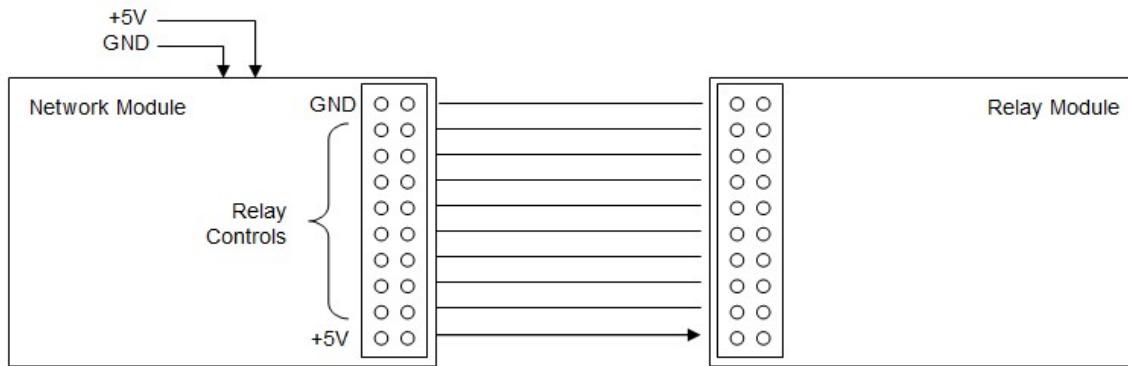
	GND	9	10	11	12	13	14	15	16	5V	
IO 9	○	○	○	○	○	○	○	○	○	○	1
IO 10		○	○	○	○	○	○	○	○		2
IO 11			○	○	○	○	○	○	○		3
IO 12				○	○	○	○	○	○		4
IO 13					○	○	○	○	○		5
IO 14						○	○	○	○		6
IO 15							○	○	○		7
IO 16								○	○		8
										○	5V

Notes on Interfacing to Relay Modules

There are two things to be cautious of when attaching relay modules to the Network Module.

Power Distribution

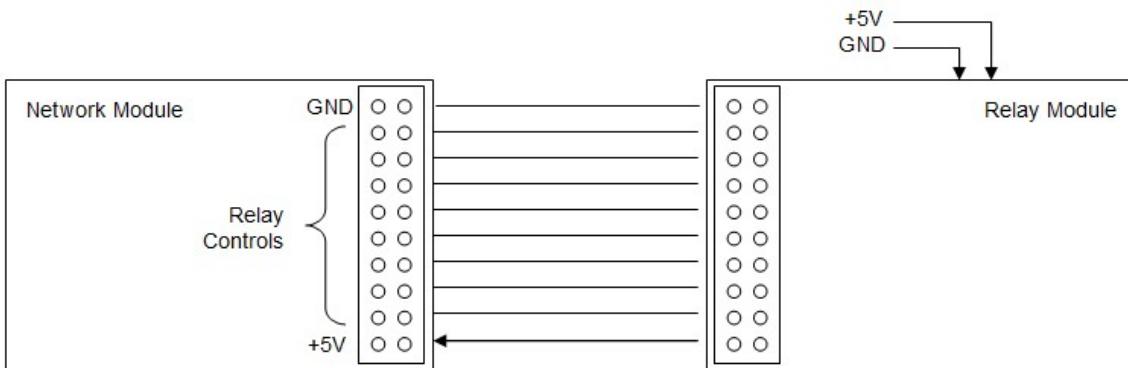
The first thing to consider is supplying power to the relay modules. The basic design of the Network Module is intended to provide +5V power to the relay modules via the pin header that also provides the relay control signals. This works well for just a few relays (up to 3 or 4). This connection method is illustrated here:



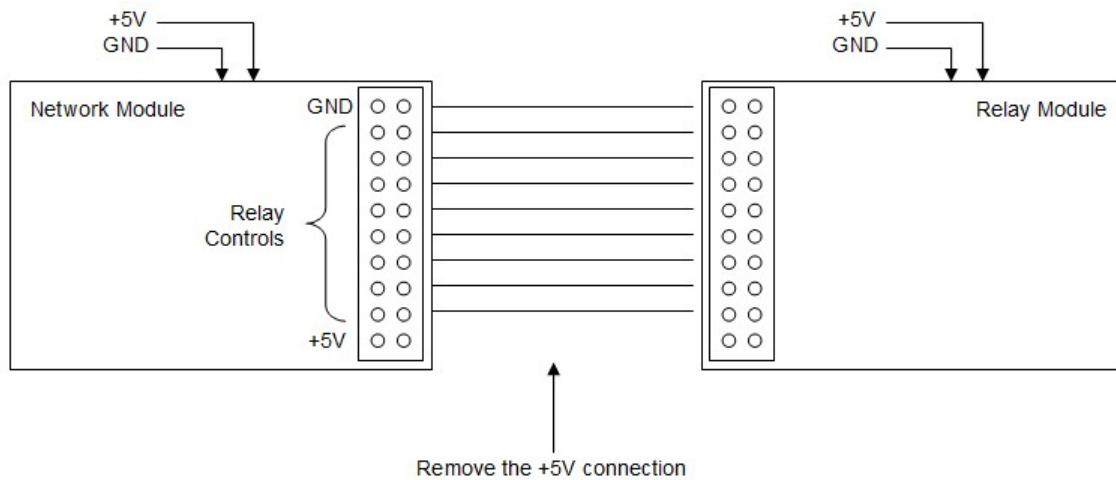
If you attach more relays you need to make sure that there is sufficient current supplied by your +5V power supply attached to the Network Module AND you need to make sure the method used to send power to the relay modules is adequate. This is particularly important if you are transferring power via a ribbon cable.

If you don't think you can provide adequate power to the relay modules via the Network Module relay header you can consider a couple of options:

- 1) Connect +5V power only at the Relays, and let the power/signal header send +5V back to the Network module.

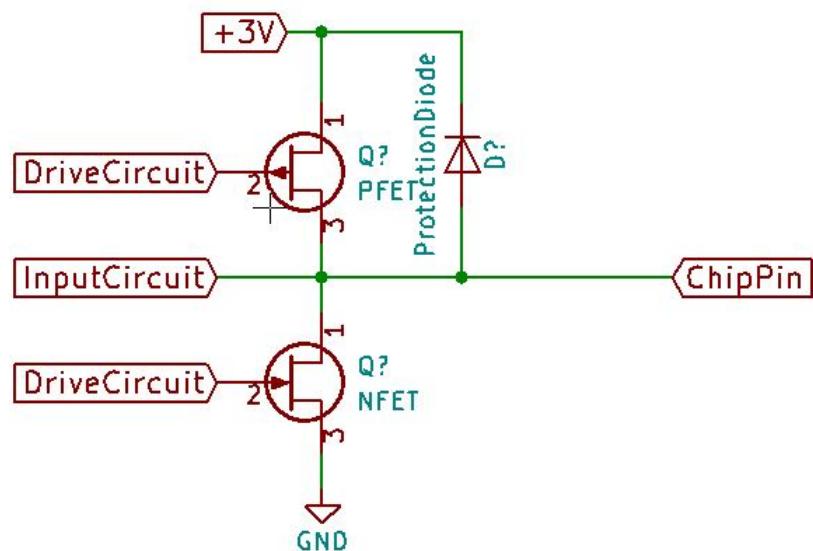


- 2) Use separate +5V power supplies on the Network Module and Relay Modules. If you do this you'll need to disconnect the +5V power connection between the headers.



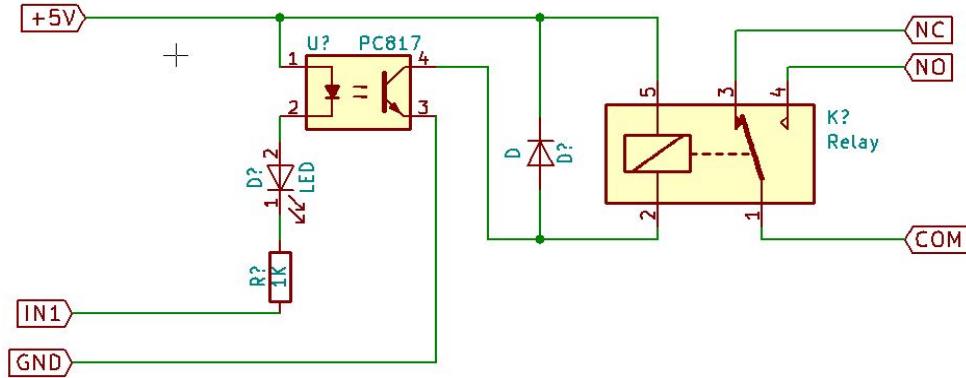
Type of Relay Module

The second consideration is the type of relay modules you attach. The SM8S processor on the Network Module operates at 3V and its outputs are connected directly to the relay control header. So, you need to avoid inadvertently causing +5V feedback from the relay modules to the 3V output pins of the processor that exceed the processor specifications (check the spec, but the short version is: Max 3.3V and/or limit to 4mA per pin, AND limit to 20mA across all pins). The reason this is a concern is because the SM8S output pins have overvoltage protection diodes that can provide a current path if a voltage higher than 3.3V appears on the pin when it is not in an active pull-down state. To visualize this here is a drawing illustrating the output pin:



Focus on the Protection Diode. There is also a protection diode to ground, but it is not a concern in this discussion so I left it out. If any of the relay modules can provide a current path from a higher voltage through the chip pin (when the pin is not pulling down) then there is the potential for damage. Knowing this let's look at typical relay module designs.

- a) **Opto-isolated relay boards:** If you use opto-isolated relay boards there should not be a concern as long as the relay boards are designed to operate at a voltage no higher than 5V. The typical design of these relay modules looks like this:

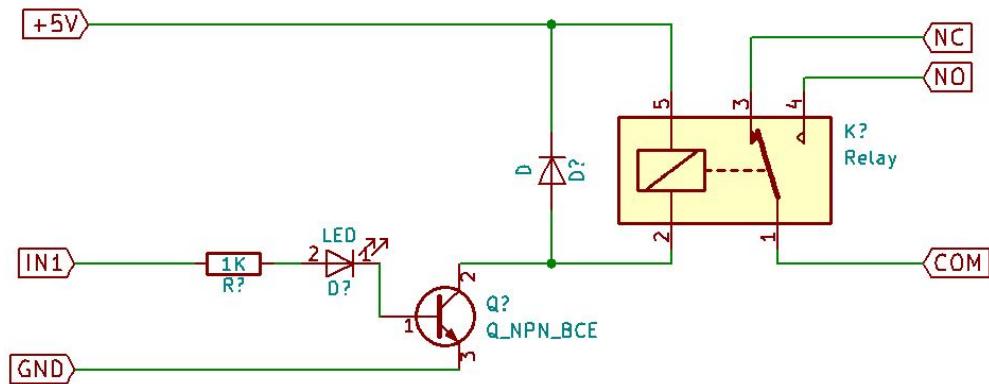


Note that in fact this relay module can provide a current path from +5V, through the photo emitter diode of the opto-isolator, through the visible LED, through the 1K resistor, then to the SM8S output pin via the “IN1” connection. But this will still work and here is why:

- The difference in voltage from the 5V supply to the SM8S output pin is $5V - 3V$. But about 0.7V is dropped across the photo emitter diode. Then another 0.7V is dropped across the LED. And about 0.3V is dropped across the protection diode in the SM8S. The result is that there is only $5 - 3 - 0.7 - 0.7 - 0.3 = 0.3V$ potential across the 1K resistor. This will result in about 300 uA of current flowing through the path. This is not enough current to damage the SM8S and not enough current to cause the relay module to operate. So while not ideal it works.
- If your relay module does not have the LED in the trigger signal path as shown in the drawing above it might still work, but you'll have to test it to verify. The difference is that the 0.7 volt drop across the LED is missing from the equation so about 1mA will flow into the output pin of the SM8S. That won't hurt the SM8S, but it might cause the opto-isolator to operate in turn preventing the relay from releasing or causing the relay to release intermittently.

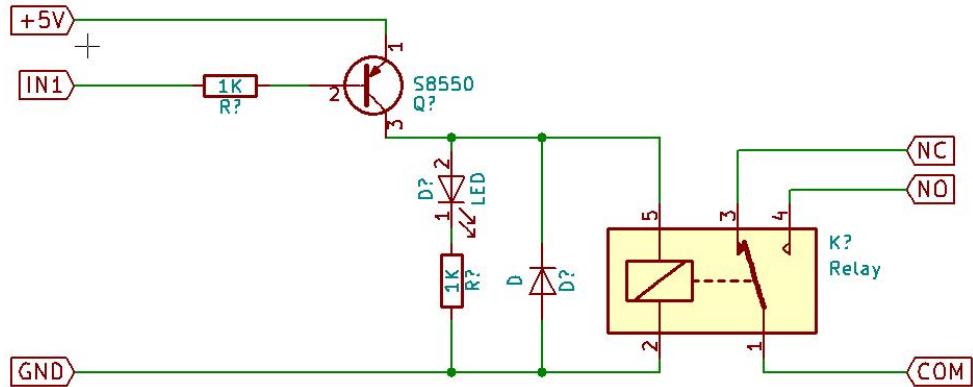
- b) **Non-isolated relay module, Active HIGH trigger signal:** Some relay modules do not have opto-isolators. If they are of a design that has an active high trigger signal then the typical design has a 1K ohm resistor feeding the base of a NPN transistor. This type of relay module should operate just fine when connected directly to the Network Module, although you'll find that the logic seems reversed and you may have to set or clear the "Invert" function in the Relay Control page of the GUI.

A typical active-high relay module circuit design:



The reason this module works with the Network Module is because it has no path from +5V back to the SM8S output pin..

- c) **Non-isolated relay module, Active LOW trigger signal:** This is another relay module design that does not have opto-isolators. This design typically has an active low trigger signal, and the typical design has a 1K ohm resistor feeding the base of a PNP transistor. A typical relay module design looks like this:



This design is problematic in that the PNP transistor is connected to 5V, and when the Network Module control signal goes to a high state a reverse current flow (also known as an injected current flow) will travel from +5V through the PNP transistor, through the 1K resistor, and into the SM8S output pin. Analyzing this path:

- The difference in voltage from the 5V power supply to the SM8S output pin is $5V - 3V$. About 0.7V is dropped across the PNP transistor, and about 0.3V is dropped across the protection diode in the SM8S. The result is that there is $5 - 3 - 0.7 - 0.3 = 1V$ potential across the 1K resistor. This will result in about 1mA of current flowing through the path. This is not enough current to damage the SM8S, but it is in the active region of the PNP transistor. This may not allow the relay to turn off – or the relay may operate intermittently. If this is the case and you are unable to get a more compatible relay module you will need to provide a voltage shifting buffer between the Network Module and the Relay Module.
- If the relay module you have places the LED in series with the PNP transistor the module may work better due to the voltage drop across the LED. However, there may still be enough current to cause the PNP transistor and the relay to operate intermittently. All you can do is give it a try.

Notes on Inputs

If you use the code configurations that provide digital inputs you'll need to be careful about the voltage you put on the input pin. The pins are directly connected to the SM8S processor. The processor operates at 3V, so you'll need to limit the high level voltage applied to the pin to 3V, or limit the current to no more than 1 mA.

Each input pin has a weak pull-up applied internal to the SM8S processor. The pull-up has a typical resistance equivalent of 60Kohm, but can range from 30Kohm to 80Kohm.

Some recommendations:

- 1) If you are using 3V logic to drive the input pin you should be able to directly connect it.
- 2) If your driver circuitry might place more than 3V on the input pin you can do one of the following:
 - a) Use open collector devices or level translators to prevent putting more than 3V on the input pin.
 - b) Use relay contacts to ground the input pin, relying on the SM8S pull-up to take the pin high. This might not be adequate if the wiring to the input pin is long or is subject to electrical interference.
 - c) Put a 1Kohm resistor between the driver logic and the input pin, but be sure the driver cannot exceed 5V. This isn't ideal, but should limit any current driven into the SM8S to an acceptable level and still achieve adequate logic levels at the SM8S input.

Hardware Design to Maintain Relay States Through a Power Loss or Reboot

A user wanted to know how to prevent relays from changing state during a power loss on the Network Module. This question is very dependent on the whether the relays remain powered up during the power loss on the Network Module, AND it is dependent on whether the control input to the relay is active low or active high. So let's explore why things happen and what you can do about it.

First of all, be aware that when the Network Module loses power its outputs go to a low level signal. I'm sure this makes sense to you: no power, no signal output. But there is the additional consideration that the overvoltage protection on the device pins (effectively a diode to VCC) will look like a pull-down when VCC on the STM8S processor goes to zero. Also be aware that when the STM8S processor powers up it defaults all IO pins to a "floating input" state. This is a function of the chip design, so it can't be changed.

Regardless of what you define as ON or OFF, this discussion here is purely from the perspective of the output signal levels on the Network Module and the input type of the relay being driven.

Scenarios and what happens:

Scenario 1:

- a) Assume relays remain powered up while the Network Module is powered down.
- b) Relays are active high inputs (a high level signal activates the relay).

What will happen:

- When the Network Module loses power and then reboots any relay that is inactive will remain inactive if Retain was enabled in Configuration.
- When the Network Module loses power and then reboots any relay that is active will go to an inactive state during Network Module power loss, then the relay will return to an active state if Retain was enabled in Configuration.
- Note: If the relays also lose power they will of course go inactive during power loss.

Scenario 2:

- a) Assume relays remain powered up while the Network Module is powered down.
- b) Relays are active low inputs (a low level signal activates the relay).

What will happen:

- When the Network Module loses power and then reboots any relay that is inactive will go to an active state because the Network Module outputs go to a low state

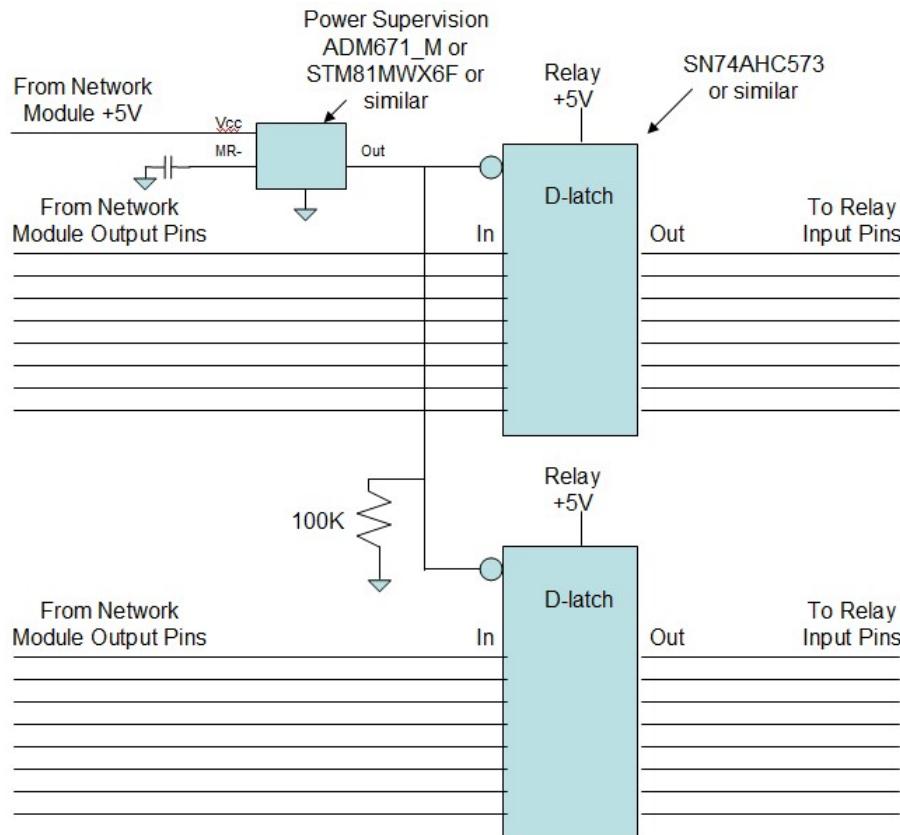
- during power loss, then the relay will return to an inactive state if Retain was enabled in Configuration.
- When the Network Module loses power and then reboots any relay that is active will remain in an active state if Retain was enabled in Configuration.
 - Note: If the relays also lose power they will of course go inactive during power loss.

So let's say you don't like the above scenarios and you want the relays to stay just the way you set them through a power loss, regardless of whether they were active or not. Well, this can only be done with hardware external to the Network Module. Here are some suggested solutions:

Option A:

- Assume the relays remain powered up while only the Network Module loses power.
- The relays can be either active low or active high inputs.

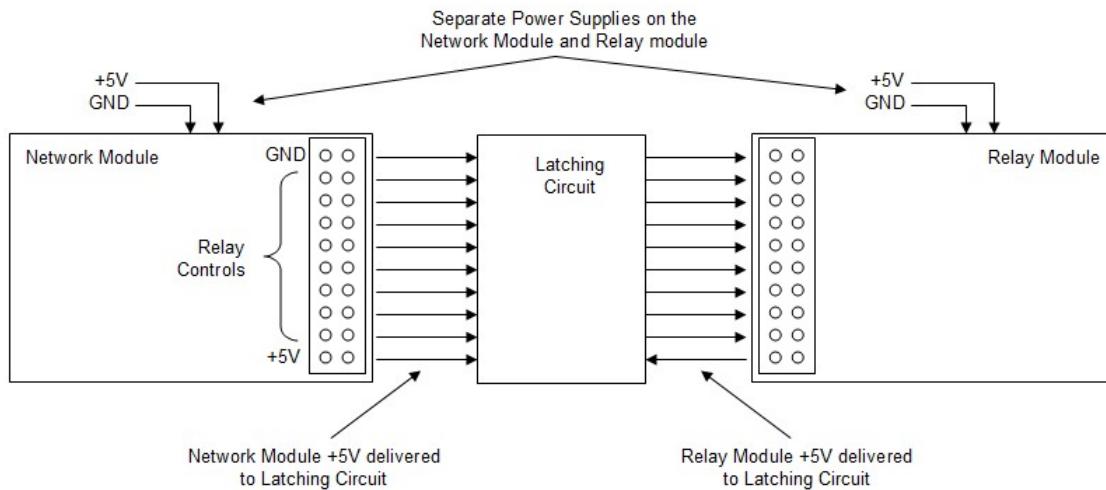
Since we know the Network Module outputs will go low during power loss we need to build hardware that will maintain the state of the relay control input during the outage. An example circuit is shown here for 16 Relays:



In the above circuit the Power Supervision device will detect when power is falling on the Network Module and will cause the D-latch devices to capture the output states of the Network Module. As long as power is maintained on the Relays and the D-latches the relays will maintain their state.

Once power returns on the Network Module the Power Supervision device will continue to keep the D-latch devices in a hold state until the STM8 processor is operating. If additional time is needed a capacitor can be added to the MR- input.

The latching circuit described above is placed between the Network Module and the Relay Module as shown here:



The downside to the above: The devices in the Latching Circuit may not all be available in DIP form – some may be surface mount. This implies that you may have to design a circuit board for this solution. That is not as onerous as you might think. There are cheap manufacturing sources in China that will make up to 10 circuit boards for almost nothing (less than \$10 USD). But you must design the board and generate gerber files.

Option B:

- Use “Latching Relays”

This option almost sounds like nirvana until you look at the details.

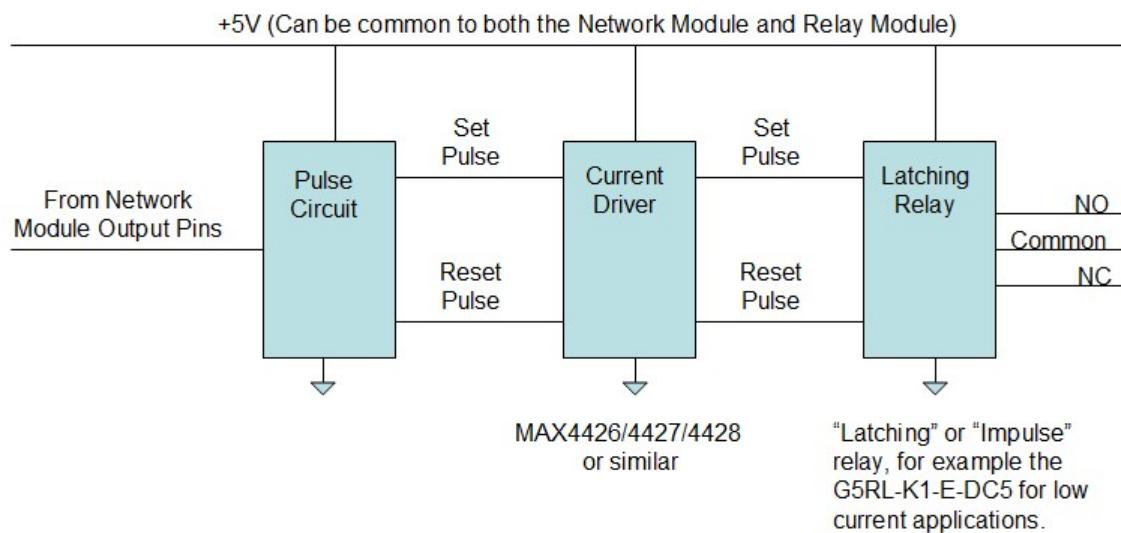
First let me caution you about “bi-stable self latching” relays that you may find on eBay. Those are meant to be provided with a pulse and they will switch and hold their state. BUT, you can’t tell what state they are in remotely, so not very useful for remote

applications. It might be possible to tap into their circuitry and find a sense point that could be fed back to one of the Network Module sense inputs. But remember these bi-stable relays are not actual latching relays as they require that power be maintained to the relay for it to retain its state (just like Option A above)..

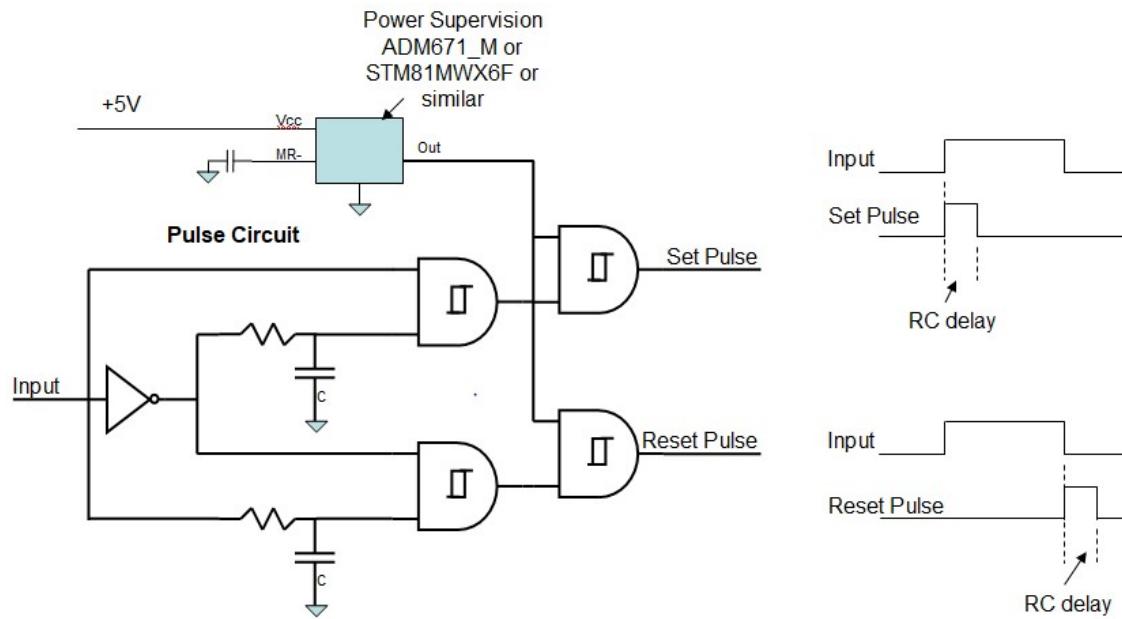
So, what about REAL latching relays? They will maintain their state even if ALL power is lost to the Network Module and the Relay Module. These relays are sometimes called “Impulse” relays because they require a pulse on a Set pin to put them in one state, and a pulse on a Reset pin to go to the other state. An example of this type of relay is the G5RL-K1-E-DC5 for low current applications.

Since the Network module only has a single state output a circuit is required to convert that signal to the pulses needed by the latching relays. And a driver is needed to supply the necessary pulse current to the relays. AND the circuit still needs to detect power loss to prevent inadvertent pulse generation when power is lost and restored.

A block diagram looks like this:



The pulse circuit looks like this:



The above needs to be repeated for each relay (except the Power Supervision can be common).

The resistor and capacitor values are determined as follows:

Need minimum pulse width of 30ms

When input is rising the output will switch at about 2.6v

When input is falling the output will switch at about 1.8v

Need about 60K and 1uF for rising signal. This will give an RC of about 60ms, with a pulse width of about 40ms.

Using 60K and 1uF for the falling signal will have a pulse width of about 60ms.

As in Option A a capacitor can be applied to the MR- input of the Power Supervision if a longer delay time is needed for the STM8 to stabilize.

Suggested part number for the Schmitt NAND gate is the SN74HC7001 and for the Inverter is the SN74LVC1G04.

After considering the Pulse Circuit, Power Supervision, and Current Drivers once again the circuit is complicated enough to require a circuit board.

Option C:

You may want to consider using a UPS to prevent power loss. That will solve most of the problems discussed here. If power loss is nearly non-existent, then the only concern is brief relay chatter in event of a reboot, which should also be nearly non-existent once the Network Module is set up.

IMPORTANT: I have not implemented any of the above power loss circuits, so there may be errors in what I've described. If you plan to go this route analyze the design carefully.

Summary:

In the end you need to consider carefully if fully retaining the relay states all the way through a power loss is really necessary. You can use the Retain setting in the Network Module to be sure the relays return to their pre-power loss state, but they may still “chatter” once or twice during reboot or power loss and recovery.

Adding DS18B20 Temperature Sensors

Code was added to allow you to use IO 16 (pin 16) for DS18B20 Temperature Sensors. You can attach up to 5 DS18B20 sensors to this pin.

In the Features section of the Configuration page you'll see a checkbox for DS18B20.

- | | |
|----------|--|
| Features | <input type="checkbox"/> Full Duplex
<input type="checkbox"/> HA Auto
<input type="checkbox"/> MQTT
<input checked="" type="checkbox"/> DS18B20 |
|----------|--|

If you check this box IO 16 will show as “Disabled”, and in fact the pin IS disabled for use as an Input/Output pin. But you can now attach up to 5 DS18B20 temperature sensors to the pin and have the temperature seen at these sensors displayed in the IOControl page. If you also check MQTT the temperature is Published on MQTT. And, if you also check HA Auto the temperature sensor will be Auto Discovered in Home Assistant.

Diagram for attaching one sensor:

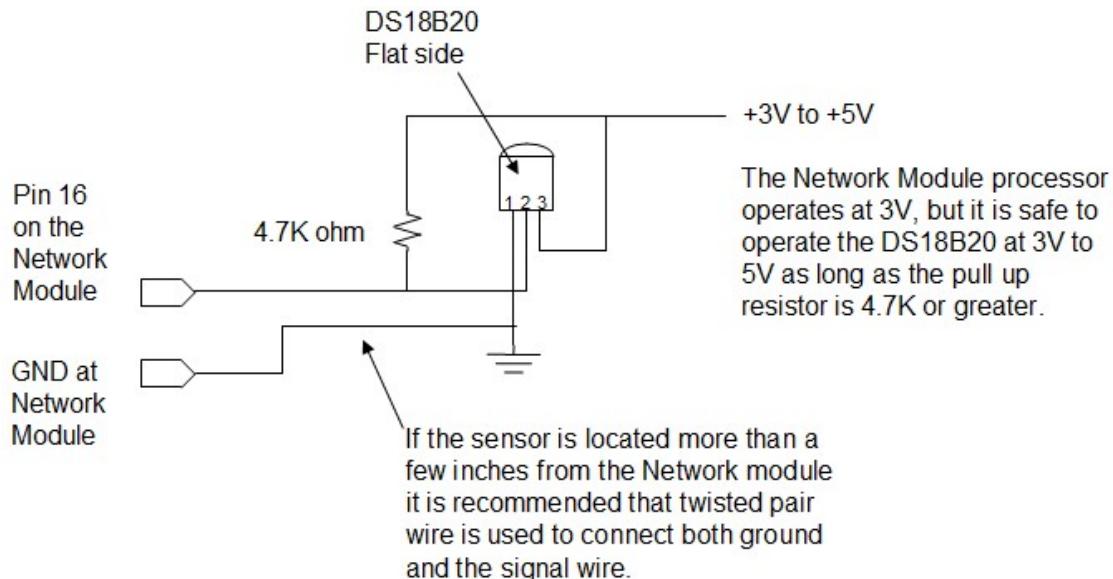
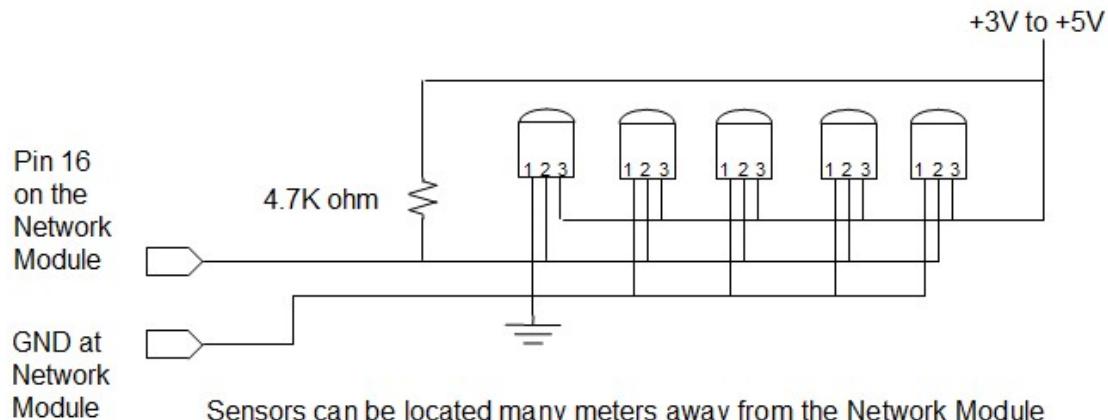


Diagram for attaching multiple sensors:



Sensors can be located many meters away from the Network Module and may be spaced many meters apart from each other. Distance has not been tested and is subject to the drive capability of the Network Module, cable type used, and electromagnetic interference conditions around the installation. See the DS18B20 specification.

On power up the temperature display may show a "filler" value of -----°C degrees for each sensor. Likewise, if one of the 5 sensors is missing it will show the filler value. As soon as communication with the sensor is established the value will be as reported by the sensor. If the sensor fails or becomes disconnected after initial communication is established the reported temperature will be -000.1°C until reboot occurs.

Note that sensors appear in the display in the order of the unique Serial Number contained within each DS18B20. Thus if a sensor fails and is replaced the order of the sensors in the display may change.

At this time temperature is only shown in Celsius.

Developers: Setting Up a Development Environment

NOTE: This information is only needed if you plan to set up your own development environment to modify and compile code. You don't need to do this if you are going to use the binaries (.stp and .sx files) already provided.

If you want to change the code for your own use I assume you have some experience with coding and the tools typically involved. I used the tools described in the previous sections for actual programming of the device, and used the Cosmic tools for the development environment. To duplicate this you'll want the following:

- 1) **Download and install the Cosmic Compiler:** Use the one that is specifically for the STM8 devices. Start at this website
<https://www.st.com/en/development-tools/cxstm8.html#product-details>
Click on Product Details and follow the link to the "partner website". From there you can download the compiler. The compiler is free. They will send you a 1-year license, but I think you can renew over and over. Note that the license is specific to the machine you install it on.
As an FYI, even though my PC is x64, the tools installed in this directory:
C:/Program Files (x86)/COSMIC/FSE_Compilers/
- 2) **Download and install the following library from st.com:**
en.stsw-stm8069.zip You'll find it at
<https://www.st.com/en/embedded-software/stsw-stm8069.html>
NOTE: I included this library in the files included with the project so you may not need this step if you copy all the files from GitHub. This is the STM8S_StdPeriph_Driver directory.
- 3) **Copy the Program:** With the above installed the next step is to copy the entire project from GitHub into your Documents directory. On my Windows 10 machine the project was located in the following directory:
C:/Users/Mike/Documents/COSMIC/FSE_Compilers/CXSTM8/NetworkModule
Of course you will likely have a different user ID.

Start the Cosmic tools by double clicking on the NetworkModule.prjsm8 file. You should be on your way.

A note about my coding style: My coding is not particularly esoteric or convoluted. I try to keep it simple to read and understand even if that is less efficient. And I put a lot of

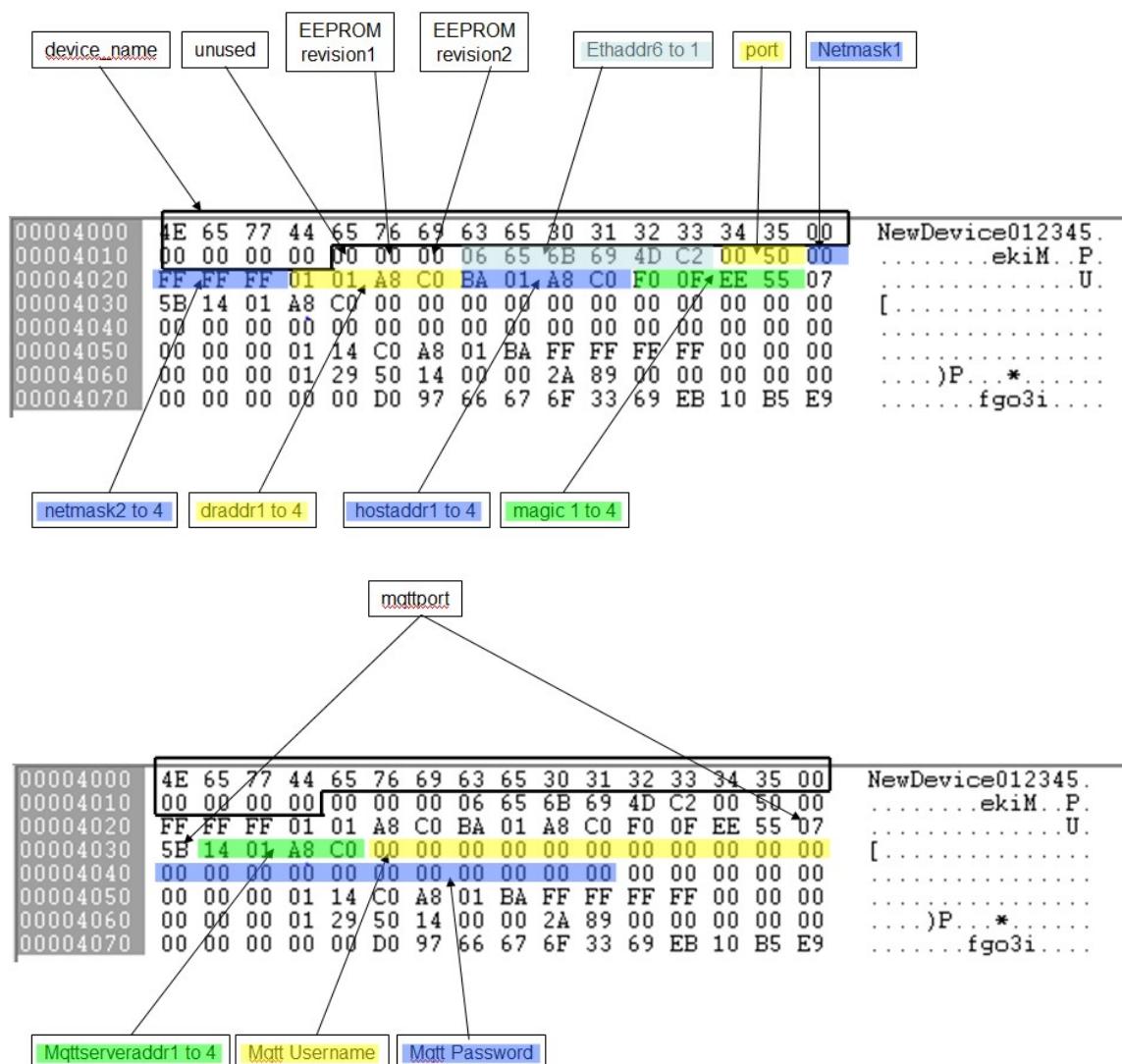
comments in, particularly if I had to do things to make the code work that didn't fully make sense to me. Sometimes that stuff happens and my intention is to come back and look at it again later. So, feel free to modify and "do it your way". I'm not proud as long as it works.

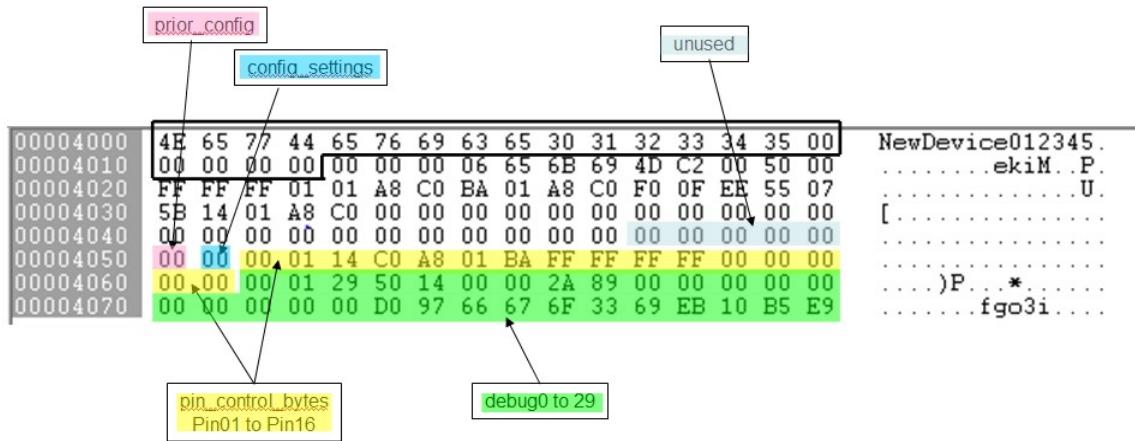
Developers: Location of EEPROM Variables

NOTE: This information is only needed if you plan to set up your own development environment to modify and compile code.

You can view the contents of the EEPROM with the STVP programmer by selecting the “Data Memory” tab and using the “Read / Current Tab” function. The displayed information has the following definitions.

Note that the data in some fields is “left to right”, a more human readable direction. For instance the device_name field. Some fields are “right to left”, for instance the hostaddr field.





The debug bytes are defined by the developer to provide non-volatile storage of any information the developer needs to debug code function. Some routines are already present in the source files to help with capture of debug information.

Developers: Notes on Debug Bytes

NOTE: This information is only needed if you plan to set up your own development environment to modify and compile code.

THIS INFORMATION MAY NOT BE ACCURATE if the number of debug[] bytes changes due to code changes. In particular the number of debug[] bytes may be reduced if more EEPROM is needed to support added functionality.

The code contains compile options for implementing various debug modes. This section describes the debug modes and how debug information is captured in EEPROM.

The general idea:

DEBUG_SUPPORT compile time options are determined by the #define DEBUG_SUPPORT statement in the uiport.h file.

When code is compiled with any DEBUG_SUPPORT option debug data is stored in RAM locations named debug[].

Each debug[] byte has a corresponding storage location in EEPROM named stored_debug[].

The data in the debug[] RAM locations is copied to EEPROM when the function update_debug_storage1() is called, or when a function requires specific debug[] bytes to be stored in EEPROM. In some functions any change in a specific debug[] byte will cause it to be stored in EEPROM.

Note that ANY time a debug[] byte is stored in EEPROM it is first compared to the byte already stored in EEPROM, and the EEPROM write only occurs if the bytes do not compare. This is to prevent excessive writes to EEPROM. Example code:

```
for (i = 0; i < NUM_DEBUG_BYTES; i++) {  
    if (stored_debug[i] != debug[i]) stored_debug[i] = debug[i];  
}
```

IMPORTANT: This “compare before write to EEPROM” concept should be used any time a write to EEPROM of any kind is to occur. This will help prevent excessive wear on the EEPROM, particularly when a coding error is made that might cause a frequent write to EEPROM.

ALSO IMPORTANT: The EEPROM must be unlocked prior to any write (see function unlock_eeprom()). After the write the EEPROM should be locked again (see function lock_eeprom()). This helps prevent coding errors from inadvertently writing EEPROM.

When code is compiled with DEBUG_SUPPORT == 1

All debug[] bytes are available to the developer to be used as needed, and they are copied to EEPROM when the function update_debug_storage1() is called.

When code is compiled with DEBUG_SUPPORT == 7, 11, OR 15

The last 10 bytes of debug[] (and stored_debug[]) are reserved for some specific debug information. The last 10 debug[] bytes are stored in EEPROM whenever the debug[] byte content changes. The debug[] bytes prior to the last 10 can be used just like the debug[] bytes are used when DEBUG_SUPPORT == 1.

!!!! Leave DEBUG_SUPPORT 11 enabled for Production Code. While this seems inconsistent with the normal use of “debug”, DEBUG_SUPPORT 11 enables the “Link Error Stats” web page. The Link Error Stats web page can be very useful to regular users that may need to diagnose whether Full Duplex is needed with their network switch(es).

In addition to the “update_debug_storage1()” function there are some specialized “copy debug[] to stored_debug[]” routines as follows:

update_debug_storage()

This function will copy the debug[] bytes to EEPROM only if debug[0] == 0x01. If the function finds that debug[0] contains 0x01 it will write 0x02 into debug[0]. This creates a “single snapshot” capability.

capture_uip_buf_transmit()

This function will capture a portion of the uip_buf when transmit data is present. The function can be modified to perform the capture when various triggers are present. There are some sample triggers in the code that are commented out.

capture_uip_buf_receive()

This function is similar to capture_uip_buf_transmit but contains triggers for capturing a portion of the uip_buf when receive data is present. There are some sample triggers in the code that are commented out.

capture_mqtt_sendbuf()

This function is similar to the other capture functions above except that it is designed to capture a portion of the data in the MQTT sendbuf.

```
#define DEBUG_SUPPORT table
```

Bit 0: Enable write to EEPROM for debug[] bytes

Bit 1: Last 10 bytes of debug[] allocated to:

- Reset Status Register counters
- TXERIF counter
- RXERIF counter
- Stack Overflow bit
- ENC28J60 revision level

Bit 2: UART enabled

Will display the above plus other developer implemented information

Bit 3: Browser page enabled for Link Error Stats. Will display:

- Seconds counter (since last boot)
- Transmit counter (since last boot)
- Octets showing Stack Overflow, ENC28J60 revision, TXERIF counter, RXERIF counter
- Octets showing MQTT Response Timeout counter, MQTT Not OK counter, MQTT Broker Disconnect counter

Bits										
7	6	5	4	3	2	1	0	Dec	Function	
0	0	0	0	0	0	0	0	0	No debug No UART No Link Error Stats browser page	
0	0	0	0	0	0	0	1	1	debug[] bytes enabled - visible only via STVP No UART No Link Error Stats browser page	
0	0	0	0	0	1	1	1	7	debug[] bytes enabled - visible only via STVP Last 10 bytes of debug[] allocated to specific debug data UART enabled No Link Error Stats browser page	
0	0	0	0	1	0	1	1	11	debug[] bytes enabled – visible only via STVP Last 10 bytes of debug[] allocated to specific debug data No UART Link Error Stats browser page enabled Set DEBUG_SUPPORT 11 for Production Code	
0	0	0	0	1	1	1	1	15	debug[] bytes enabled – visible only via STVP Last 10 bytes of debug[] allocated specific debug data UART enabled Link Error Stats browser page enabled	

“visible only via STVP” means that the STVP programmer software should be used to display the “Data Memory” in order to view these bytes. See section “Location of EEPROM Variables”.

The table below defines the debug[] bytes dependent on the DEBUG_SUPPORT setting. Note that NUM_DEBUG_BYTES defined in main.h sets the total number of debug[] bytes available. If you make code changes that use up more EEPROM you will have to adjust NUM_DEBUG_BYTES in main.h,

debug[] Bytes Defined

DEBUG_SUPPORT == 1	DEBUG_SUPPORT == 7, 11, 15				
debug[00] *	debug[00] *				
debug[01] *	debug[01] *				
debug[02] *	debug[02] *				
debug[03] *	debug[03] *				
debug[04] *	debug[04] *				
debug[05] *	debug[05] *				
debug[06] *	debug[06] *				
debug[07] *	debug[07] *				
debug[08] *	debug[08] *				
debug[09] *	debug[09] *				
debug[10] *	debug[10] *				
debug[11] *	debug[11] *				
debug[12] *	debug[12] *				
debug[13] *	debug[13] *				
debug[14] *	debug[14] *				
debug[15] *	debug[15] *				
debug[16] *	debug[16] *				
debug[17] *	debug[17] *				
debug[18] *	debug[18] *				
debug[19] *	debug[19] *				
debug[20] *	debug[20] ** } debug[20] Not used; Available debug[21] ** } debug[21] Not used; Available debug[22] ** } debug[22] Bit 7 = Stack Overflow Error Bit 6-4 = Not defined Bit 3-0 = ENC28J60 revision debug[23] ** } debug[23] ENC28J60 TXERIF counter debug[24] ** } debug[24] ENC28J60 RXERIF counter				
debug[25] *	debug[25] ** } debug[25] EMCF: EMC reset flag count debug[26] *	debug[26] ** } debug[26] SWIMF: SWIM reset flag count debug[27] *	debug[27] ** } debug[27] ILLOPF: Illegal opcode reset flag count debug[28] *	debug[28] ** } debug[28] IWDGF: Independent Watchdog reset flag count debug[29] *	debug[29] ** } debug[29] WWDGF: Window Watchdog reset flag count

* Stored in EEPROM when update_eeprom_storage1() is called

** Stored in EEPROM each time a change occurs

Developers: Notes on Configuration Debug and pin_control Bytes

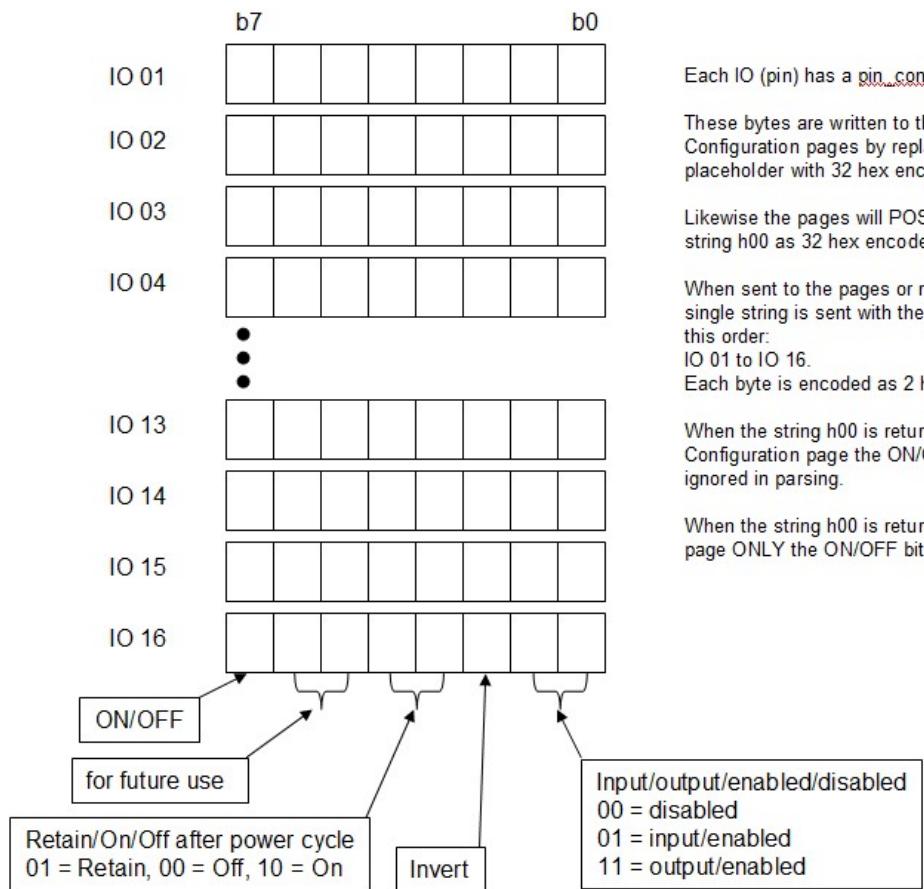
NOTE: This information is only needed if you plan to set up your own development environment to modify and compile code.

The Configuration page contains a debug feature useful for development. If you go to the Configuration page, add #d to the URL, then click Refresh a column of numbers will appear to the right of the settings as illustrated here:

IO	Type	Invert Boot state	
#1	input	<input checked="" type="checkbox"/>	5
#2	input	<input type="checkbox"/>	129
#3	input	<input type="checkbox"/>	129
#4	input	<input type="checkbox"/>	129
#5	input	<input type="checkbox"/>	129
#6	input	<input type="checkbox"/>	129
#7	input	<input type="checkbox"/>	129
#8	input	<input type="checkbox"/>	129
#9	input	<input type="checkbox"/>	129
#10	input	<input type="checkbox"/>	129
#11	output	<input type="checkbox"/>	off
#12	output	<input type="checkbox"/>	on
#13	output	<input type="checkbox"/>	retain
#14	output	<input checked="" type="checkbox"/>	off
#15	output	<input type="checkbox"/>	off
#16	disabled		128

These values are the decimal equivalents of the content of the pin_control bytes in the code. The pin_control bytes are defined as shown here:

pin_control Bytes



Each IO (pin) has a pin_control byte.

These bytes are written to the IOControl and Configuration pages by replacing the %h00 placeholder with 32 hex encoded characters.

Likewise the pages will POST all 16 bytes in the string h00 as 32 hex encoded characters.

When sent to the pages or received in a POST a single string is sent with the pin_control bytes in this order:

IO 01 to IO 16.

Each byte is encoded as 2 hex characters.

When the string h00 is returned by the Configuration page the ON/OFF bit will be ignored in parsing.

When the string h00 is returned by the IOControl page ONLY the ON/OFF bit is examined.

Developers: Using the UART

NOTE: This information is only needed if you plan to set up your own development environment to modify and compile code.

The STM8S processor includes a UART. The code can be compiled with the UART enabled so that you can add debug statements to the code. The compile option is available in the uioprt.h file (see `#define DEBUG_SUPPORT`).

When the UART is enabled it will take over IO Pin 11 and use that as the UART transmit pin. The Configuration page will show the pin as an Output, and you will not be able to change that pin configuration when the UART option is compiled. Likewise, the IOControl page will show the pin as an Output, and any attempt to change the pin state will not be successful. But the pin will continue to function as the UART output until the code is recompiled with the Debug UART option turned off.

The UART transmitter is set up as follows:

Baud rate: 115200

Data bits: 8

Parity: None

Stop bits: 1

To connect the Network Module UART Tx pin (IO 11) to a terminal or your PC you'll need a TTL-RS232 or TTL-USB converter. There are several ways to do this, but I will describe the method I used:

- a) Use a TTL-USB converter to connect the Tx pin to the USB port on a laptop.
Here's the device I used:



The image shows a red printed circuit board (PCB) with a central integrated circuit (IC) labeled 'FT232'. A small American flag icon with the text 'Free Shipping' is visible in the top left corner of the image area.

**FT232RL 3.3V 5.5V FTDI USB to TTL Serial Adapter F
Arduino Mini Port N137**

Condition: New

Bulk savings:

Buy 1 \$3.99/ea	Buy 2 \$3.79/ea	Buy 3 \$3.67/ea
-----------------	-----------------	-----------------

Quantity: 4 or more for \$3.59/ea

More than 10 available
[164 sold](#) / [See feedback](#)

Price: US \$3.99/ea

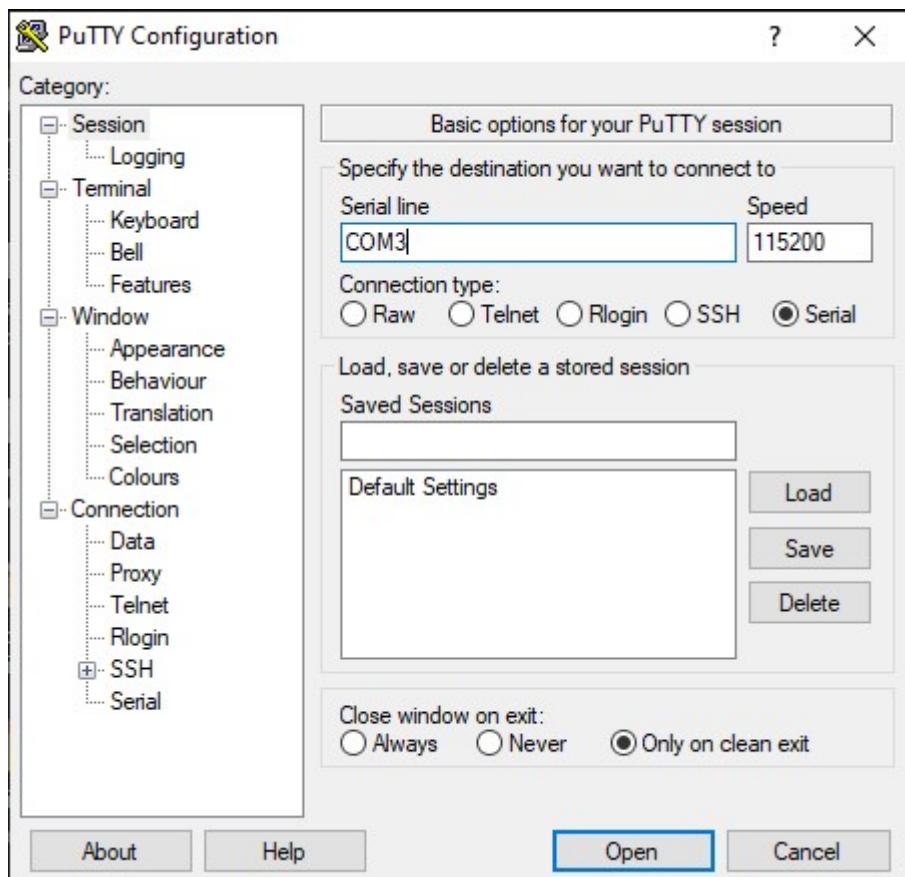
[Buy another](#)

[Add to cart](#)

[Add to Watchlist](#)

Keep in mind that the Network Module operates at 3V, so the adapter needs to be set to 3V IO on the pin connection side of the interface.

- b) My laptop was a Windows 10 OS, so I had to download the drivers for the TTL-USB interface here:
<https://ftdichip.com/drivers/>
- c) I used PuTTY on my laptop to communicate with the TTL-USB device. In my case the device showed up as COM3. PuTTY setup is simple:



- d) If everything is set up correctly you should see this in the PuTTY display when the Network Module boots:

XXXXXXXXXXXXXXXXXXXXXXXXXXXX

Add screenshot of UART banner

XXXXXXXXXXXXXXXXXXXXXXXXXXXX

Code Credits

This project borrows heavily from the work of Simon Kueppers “MicroWebServer” project available on GitHub. Extract of Simon Kueppers’ code sharing statement:

```
* Author: Simon Kueppers  
* Email: simon.kueppers@web.de  
* Homepage: http://klinkerstein.m-faq.de  
*
```

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

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/>>.

Copyright 2008 Simon Kueppers

Simon Kueppers work in turn borrows heavily from the work of Adam Dunkels uIP project, also available on GitHub and other locations. Extract of Adam Dunkels’ code sharing statement:

```
/**  
 * \file  
 * The uIP TCP/IP stack code.  
 * \author Adam Dunkels <adam@dunkels.com>  
 */  
  
/*  
 * Copyright (c) 2001-2003, Adam Dunkels.  
 * All rights reserved.  
 *  
 * Redistribution and use in source and binary forms, with or without  
 * modification, are permitted provided that the following conditions  
 * are met:  
 * 1. Redistributions of source code must retain the above copyright  
 * notice, this list of conditions and the following disclaimer.  
 */
```

- * 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * 3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.
- *
- * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
- */

With the addition of the MQTT Interface the following statement is applicable:

MIT License

Copyright(c) 2018 Liam Bindle

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,

DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

I am likewise making my code available with this statement:

- Author: Michael Nielson
- Email: nielsonm.projects@gmail.com

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

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.

See the GNU General Public License at <<http://www.gnu.org/licenses/>>.

Copyright 2020 Michael Nielson

And, finally, I want to credit Carlos Ladeira and Jevgeni Kiski for their very significant contributions to code and test.

Documentation License Note

And now for the GNU Free Documentation License (follows on the next page). If you read the short preamble the typical user will know all you really need to know.

GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.
<https://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not

explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that

translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take

reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These

may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a

translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <https://www.gnu.org/licenses/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be

used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.