



User Guide

Babel Buster 2

Models BB2-6010, SPX, SP
Modbus RTU/TCP, SNMP
Gateway
Rev. 1.1 – Mar. 2021

© 2021 Control Solutions, Inc.

BB2-6010, SPX, SP User Guide Contents

[1 Introduction](#)

- 1.1 How to Use This Guide
- 1.2 Overview of Gateway Devices
- 1.3 Important Safety Notice
- 1.4 Warranty

[2 Connecting Gateway for the First Time](#)

[3 Configuring Gateway as a Modbus RTU Master](#)

- 3.1 Modbus RTU Device Configuration
- 3.2 Modbus RTU Master Read Maps
- 3.3 Modbus RTU Master Write Maps
- 3.4 Modbus RTU Master Data Displayed Per Slave
- 3.5 Modbus RTU Errors

[4 Configuring Gateway as a Modbus TCP Client](#)

- 4.1 Modbus TCP Device Configuration
- 4.2 Modbus TCP Client Read Maps
- 4.3 Modbus TCP Client Write Maps
- 4.4 Modbus TCP Errors

[5 Configuring Gateway as a Modbus RTU Slave](#)

- 5.1 Modbus RTU Device Configuration
- 5.2 Modbus RTU Slave Register Mapping

[6 Configuring Gateway as a Modbus TCP Server](#)

- 6.1 Modbus TCP Device Configuration
- 6.2 Modbus TCP Register Mapping
- 6.3 Modbus Virtual Device Register Mapping

[7 Configuring Gateway as an SNMP Server](#)

- 7.1 Local SNMP MIB
- 7.2 SNMP Trap Destinations
- 7.3 SNMP Trap Triggers
- 7.4 SNMP Identity
- 7.5 Testing the SNMP Agent

[8 Configuring Gateway as an SNMP Client](#)

- 8.1 SNMP Device Configuration
- 8.2 SNMP Client Read Maps
- 8.3 SNMP Client Write Maps
- 8.4 SNMP Errors

[9 Configuring Action Rules](#)

- 9.1 Threshold Rules
- 9.2 Trending Rules
- 9.3 Cascade Rules
- 9.4 Calculate Rules
- 9.5 Constant Rules

[10 Trouble Shooting](#)

- 10.1 Modbus RTU Trouble Shooting
- 10.2 Modbus TCP Trouble Shooting
- 10.3 SNMP Trouble Shooting
- 10.4 Modbus Reference Information

[Appendix A Hardware Details](#)

- A.1 Wiring
- A.2 Front Panel LED Indicators
- A.3 RS-485 Line Termination and Bias
- A.4 Babel Buster SP DB-9 Connector & DIP Switch



1. Introduction

1.1 How to Use This Guide

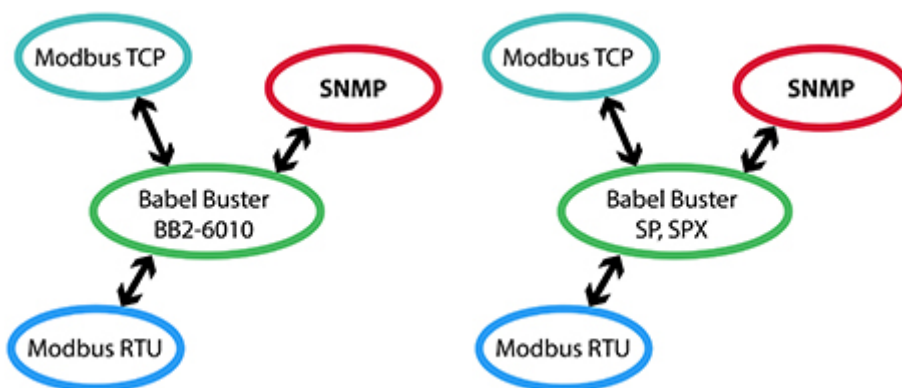
This user guide provides background information on how the gateway works, and an overview of the configuration process. There are several sections for groups of tabs found in the web interface in the gateway which is accessed by opening a web browser and browsing to the IP address of the device.

You should at least read the overview sections to gain an understanding of how the gateway functions. You can use the remaining sections as reference material to look up as needed. There is a "Quick Help" section at the bottom of each web page in the gateway which is generally sufficient for quick reference in setting up the gateway.

Throughout this user guide, screen shots taken from a Babel Buster BB2-6010 are used to illustrate. However, the same instructions apply to Babel Buster SPX and Babel Buster SP with each available screen looking nearly identical.

1.2 Overview of Gateway Devices

The Babel Buster BB2-6010, Babel Buster SPX, and Babel Buster SP are Modbus gateways used to remap registers between Modbus RTU and Modbus TCP devices, and/or between Modbus devices and SNMP. The gateway may be used as Modbus TCP client and server, and Modbus RTU master or slave, and SNMP server (or agent). In addition, the BB2-6010 and Babel Buster SPX may be used as an SNMP client (manager).



The most common application for these gateways is interfacing a Modbus device to an SNMP network, or vice versa. Another common application is remapping registers from multiple slave devices to appear as a single device on the alternate network, mapping RTU to TCP or vice versa.

If you are looking for a gateway that does not do any register mapping and simply passes Modbus commands through from RTU to TCP or vice versa, the models covered here are not the correct gateways. The same model numbers with a "-GW" suffix are used for that purpose.

1.3 Important Safety Notice

Proper system design is required for reliable and safe operation of distributed control systems incorporating any Control Solutions product. It is extremely important for the user and system designer to consider the effects of loss of power, loss of communications, and failure of components in the design of any monitoring or control application. This is especially important where the potential for property damage, personal injury, or loss of life may exist. By using ANY Control Solutions, Inc., product, the user has agreed to assume all risk and responsibility for proper system design as well as any consequence for improper system design.

1.4 Warranty

This software and documentation is provided "as is," without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of fitness or merchantability for a particular purpose. Control Solutions may make improvements and/or changes in this documentation or in the product(s) and/or the program(s) described in this documentation at any time. This product could include software bugs, technical inaccuracies, typographical errors, and the like. Changes are periodically made to the information herein; these changes may be incorporated in new editions of the software.

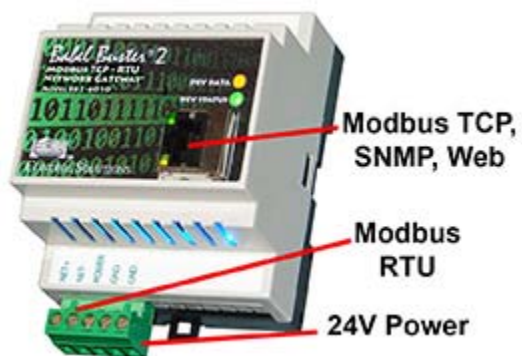


2. Connecting the Gateway for the First Time

Follow these steps to make the initial connection to the BB2-6010.

(a) Connect power.

Babel Buster BB2-6010: Apply +12 to +24VDC or 24VAC to the terminal marked "POWER", and common or ground the the terminal marked "GND".



Babel Buster SPX: Apply +12 to +24VDC or 24VAC to the terminal marked "POWER", and common or ground the the terminal marked "GND".



Babel Buster SP: Plug power adapter into power connector marked "Power".



(b) Connect a CAT5 cable between the RJ-45 jack on the gateway, and your network switch or hub. You cannot connect directly to your PC unless you use a "crossover" cable.

(c) Apply power.

BB2-6010: A blue LED inside the case should light indicating power is present. SPX: A green LED inside the case close to the RJ45 connector should light indicating power is present. SP: The LED next to the power connector should light red for a time (but will go out after bootup since this single LED doubles as communication indicator).

All models: If the link LED on the RJ45 jack is not on, check your Ethernet cable connections. Both link and activity LEDs on the RJ45 jack will be on solid for a short time during boot-up. The entire bootup process will take 1-2 minutes, during which time you will not be able to connect with a browser.

Ethernet link LED is the yellow LED integrated into the CAT5 connector on BB2-6010 and SPX, and green on SP. Ethernet activity LED is the green LED integrated into the CAT5 connector on BB2-6010 and SPX, and yellow on SP (SP is reverse of other two).

(d) The default IP address as shipped is 10.0.0.101. If your PC is not already on the 10.0.0.0 domain, you will need to add a route on your PC (XP only). Do this by opening a command prompt. First type "ipconfig" and note the IP address listed. This is your PC's IP address. Now type the command

```
route add 10.0.0.0 mask 255.255.255.0 1.2.3.4
```

but substitute your PC's IP address for 1.2.3.4.

This generally works, but if this fails, you will need to temporarily change your computer's IP address to a fixed address that starts with 10.0.0. and ends with anything but 101.




[Click here for startup and trouble shooting.](#)

[Click here for system capacity summary.](#)

[Click here for Hardware Guide](#)

Model BB2-6010
v2.37.4

Quick Help

Click any tab above to log in. If you are not already logged in, you will be asked for your user name and password. You will need these in order to log in.

To log out, simply close your browser. **IMPORTANT:** If you have made configuration changes that you want to save permanently, go to the System->Setup->Config File page and click "save". Changes made by clicking "update" are only temporary until you save changes permanently in your configuration file.

(e) Open your browser, and enter "http://10.0.0.101/" in the address window. You should see a page with the "Babel Buster BB2-6010" header shown above (or as applicable by model). From this point, you will find help on each page in the web site contained within the product.

(f) When you click on any of the page tabs such as System Setup, you will be asked for a user name and password. The default login is user name "system" with password "admin". You can also log in as "root" using password "buster". You must log in as "root" if you will be changing the IP address.

(g) To can change the IP address of the gateway, go to the Network page under System :: Setup. The following page should appear. Change the IP address, and subnet mask and gateway if applicable. Click Change IP to save the changes. The process of programming this into Flash takes around half a minute. The new IP address only takes effect following the next system restart or power cycle.

Babel Buster 2
MODBUS
NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | IP Network | **System** | | |

Data | Action Rules | **Setup** | | |

Config File | Network | User | | |

This page allows you to change this device's IP address, and select whether double registers are swapped when returned to a remote client accessing this server.

IP Address	<input type="text" value="192.168.1.88"/>	192.168.1.88	<input type="button" value="- Refresh -"/>
Subnet Mask	<input type="text" value="255.255.255.0"/>	255.255.255.0	<input type="button" value="Change IP"/>
Gateway	<input type="text" value="192.168.1.1"/>	192.168.1.1	

(h) Most changes are stored in an XML configuration file in the device's Flash file system. Only a few are stored differently, and the IP address is one of those. Normally, clicking Update on any configuration page only stores that configuration information to a temporary RAM copy of the configuration file. To make your changes other than IP address permanent, you must click Save on the Config File page (System :: Setup :: Config File).

Babel Buster 2
MODBUS
NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | IP Network | **System** | | |

Data | Action Rules | **Setup** | | |

Config File | Network | User | | |

This page allows you to manage configuration files.

Store configuration to Flash file selected from directory, or to new file if checked.

Local file directory

Create new file

Boot configuration Confirm

Upload Configuration File

No file selected.



3. Configuring Gateway as a Modbus RTU Master

The BB2-6010/SPX/SP can be a Modbus RTU master or slave. As a master you can read Modbus data from, or write Modbus data to, other Modbus slaves. The gateway will periodically poll the other Modbus devices according to register maps you set up. To read from a remote Modbus device, configure a Read Map. To write to a remote Modbus device, configure a Write Map.

Data read from a remote device is stored in a local register when received. Data written to a remote device is taken from a local register when sent. The local registers are the same collection of registers that are accessible to other masters when operating as slave, and accessible to other Modbus TCP devices as a collection of holding registers. The local registers are also accessible as SNMP MIB variables.

3.1 Modbus RTU Device Configuration

Modbus device configuration for RTU really consists of port configuration, and includes setting the slave address if the gateway is functioning as Modbus slave.

This page displays configuration parameters for the Modbus RTU serial port.

Baud Rate: 19200 Parity: None, 1 Stop Bit Update

I am the Master I am a Slave

Parameters for RTU Master:

Default Poll Rate: 5.000 Seconds

Timeout: 1.000 Seconds

Use FC 5/6 instead of 15/16 for unit numbers (slave addresses) starting at: 0

Parameters for RTU Slave:

My Address or Unit #: 1

Double registers are swapped

Select baud rate and parity from the drop down list. Click either Master or Slave buttons to select type of operation. Enter timing parameters or address as applicable. Click update to register your changes.

IMPORTANT: Set timeout to something long enough for the device. If too short, the gateway will not wait long enough for a response from the Modbus slave device, and the result will be a lot of "no response" errors from the device even though the device is perfectly functional.

If your slave/server device only supports function codes 5 and 6 for writing, check the Use FC 5/6 box. The default function codes are 15 and 16, which are most widely used. If you check the box, you should also enter a "starting at" unit # or slave address. This allows supporting both types of devices at the same time provided you assign slave addresses in two non-overlapping groups. (These settings do not apply if the gateway is the slave.)

The double register swap on this page only applies when the local device (the gateway you are configuring here) is functioning as a Modbus RTU slave.

The term "swapped" only applies to double or float formats. Modbus registers are, by definition, 16 bits of data per register. Access to 32-bit data, either 32-bit integer ("double"), or IEEE 754 floating point ("float"), is supported by the use of two consecutive registers. Modbus protocol is inherently "big endian", therefore, Modbus by the Module defaults to having the high order register first for double and float. If the low order register comes first on the device being accessed, check the "swapped" box.

If you have "swapped" turned around, you will quickly recognize it. If floating point data is reversed, a 1.0 becomes 2.2779508e-41, which simply rounds to zero. The pattern is not as predictable as the 1.0 example would suggest. A floating point 1.1 becomes negative 107609184. If 32-bit integer data is reversed, 1 becomes 65536.

3.2 Modbus RTU Master Read Maps

Getting the gateway to read registers from another Modbus device requires setting up a "Read Map" as shown here.

Read remote registers into local registers. This page creates a map entry that reads data from one or more remote Modbus RTU serial devices for processing here. Click on map number to see more detail and insert/delete rules.

Showing 1 to 2 of 2

Map #	Remote Type	Remote Register Format	Remote Register #	Remote Unit #	Scale	Local Register #	Name
1	Holding Register	Int 16-bit	5	2	0.000000	1	Test Register
2	None	Int 16-bit	0	0	0.000000	0	

Quick Help

This page only applies if the local device is configured as a Master.

Rule number simply tells you where you are in the list of register maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the number in the "Showing" box, then click Update.

Rules entered on this page only read data from remote devices. Go to the RTU Write Map to write data to those devices. The full parameter set is different for read versus write.

Rule number simply tells you where you're at on the list of register maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Showing" box, then click Update.

Rules entered on this page only read data from remote devices. Go to the RTU Write Map to write data to those devices. The full parameter set is different for read versus write.

An abbreviated version of a list of rules is shown on this page. Any of the parameters shown may be changed here and registered by clicking the Update button. To view and/or modify the complete set of parameters, click on the map number in the left most column.

For each remote register to be read, enter the register type, format, number, and remote unit (slave address). Register numbers must be standard numbers starting at 1 (not 40001). The abbreviation "Int" under Format means signed integer, while "UInt" represents unsigned integer. The Float format refers to 32-bit IEEE 754 format. Bit is a single bit, and should only be used with coil or discrete input register types (you cannot read "Bit" from a holding register - see section 10.4 for reading bits from a holding register). The "Mod10" format is unique to Schneider Electric power meters, and is supported in 2, 3, and 4-register formats.

When the remote register is read, the data will be multiplied by the scale factor and written to the local register number given. The name is optional and used only for display purposes.

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type.

Local register numbers are 1-999 for integer values, and 1001-1999 accessed as register pairs for floating

point. If you try to enter an even number above 1001, you will get an error message. All floating point register pairs start on odd boundaries. All local registers are accessed via Modbus as holding registers.

Click on the map number in the left column of the tabular read map page (above) to get the expanded view of one read map at a time (below).

This page creates a map entry that reads data from a remote Modbus RTU serial device for processing here.

Map #

Read as from register # at Unit # with low register first

Apply bit mask if applicable: then apply scale: and offset:

Save in local register # named Repeat this process every seconds.

Apply this default value: after read failure(s).

RTU Read Maps Enabled:

Rule number simply tells you where you're at on the list of register maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Map #" box, then click Update.

For each remote register to be read, enter the register type, format, number, and remote unit (slave address).

When the remote register is read, data may be manipulated before being written to the local register. If a bit mask is entered (in hexadecimal), and the remote register type is signed or unsigned (16-bit data), the mask will be bit-wise logical AND-ed with the data, and the retained bits will be right justified in the result. The result will then be multiplied by the scale factor. The offset is then added and this final result is written to the local register number given. The name is optional and used only for display purposes.

The periodic poll time determines how often the remote register will be read. This number, if nonzero, will override the default poll time given in the Devices page for the remote device being read.

The default value will be stored into the local register after the given number of read failures if the fail count is non-zero. Setting the count to zero will disable the default, and the object will retain the most recent value obtained.

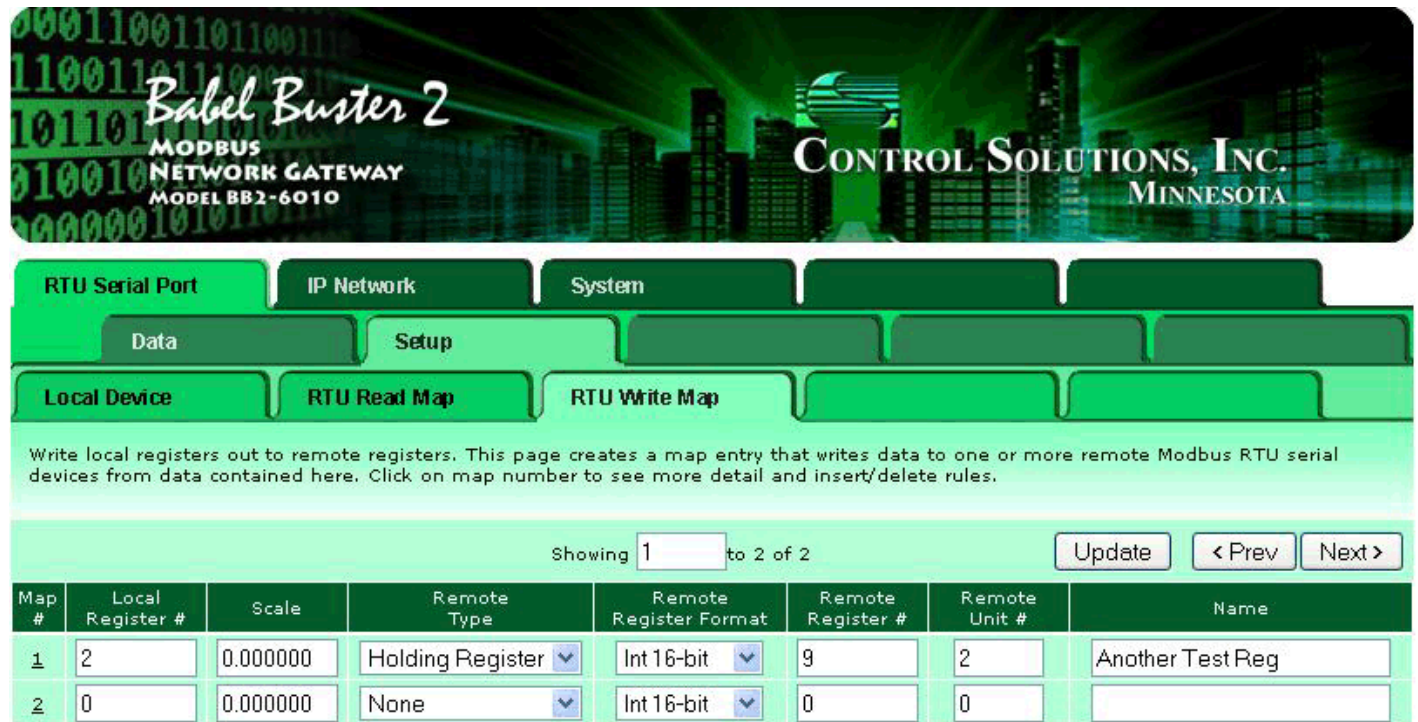
Delete will remove the rule number shown in the "Map #" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having zero for a source object or "none" for remote type.

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type. If you wish to prevent these from being displayed, reduce the number of rules enabled.

The number of rules enabled simply limits the scope of rule review so that you do not have to review a lot of unused rules. If the displayed rules are used up and you need more, increase the enabled number.

3.3 Modbus RTU Master Write Maps

Getting the gateway to write registers to another Modbus device requires setting up a "Write Map" as shown here.



The screenshot shows the Babel Buster 2 Modbus Network Gateway web interface. The page title is "Babel Buster 2 MODBUS NETWORK GATEWAY MODEL BB2-6010" and the company is "CONTROL SOLUTIONS, INC. MINNESOTA". The navigation menu includes "RTU Serial Port", "IP Network", "System", "Data", and "Setup". Under "Setup", there are tabs for "Local Device", "RTU Read Map", and "RTU Write Map". The "RTU Write Map" tab is active, displaying a table of write maps. The table has columns for Map #, Local Register #, Scale, Remote Type, Remote Register Format, Remote Register #, Remote Unit #, and Name. Two maps are listed: Map 1 with Local Register # 2, Scale 0.000000, Remote Type Holding Register, Remote Register Format Int 16-bit, Remote Register # 9, Remote Unit # 2, and Name Another Test Reg; and Map 2 with Local Register # 0, Scale 0.000000, Remote Type None, Remote Register Format Int 16-bit, Remote Register # 0, Remote Unit # 0, and Name empty. The page also shows a "Showing 1 to 2 of 2" indicator, an "Update" button, and "Prev" and "Next" navigation buttons.

Map #	Local Register #	Scale	Remote Type	Remote Register Format	Remote Register #	Remote Unit #	Name
1	2	0.000000	Holding Register	Int 16-bit	9	2	Another Test Reg
2	0	0.000000	None	Int 16-bit	0	0	

Rule number simply tells you where you're at on the list of register maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Showing" box, then click Update.

Rules entered on this page only write data to remote devices. Go to the Client Read Map to read data from those devices. The full parameter set is different for read versus write.

An abbreviated version of a list of rules is shown on this page. Any of the parameters shown may be changed here and registered by clicking the Update button. To view and/or modify the complete set of parameters, click on the map number in the left most column.

Data from the local register given will be multiplied by the scale factor before being written. For each remote register to be written, enter the register type, format, number, and remote unit (slave address).

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type.

Local register numbers are 1-999 for integer values, and 1001-1999 accessed as register pairs for floating point. If you try to enter an even number above 1001, you will get an error message. All floating point register pairs start on odd boundaries. All local registers are accessed via Modbus as holding registers.

Click on the map number in the left column of the tabular write map page (above) to get the expanded

view of one write map at a time (below).

Babel Buster 2
MODBUS NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | IP Network | System

Data | Setup

Local Device | RTU Read Map | RTU Write Map

This page creates a map entry that writes data to a remote Modbus RTU serial device from data contained here.

Map #

Read local register # named

Write remote register when local register changes by > or when seconds have elapsed with no change.

Otherwise write remote register unconditionally, applying local register data as follows:

Apply scale: and offset: Then if applicable, apply bit mask: and bit fill:

Write as to register # at Unit # with low register first

Repeat this process at least no more than every seconds.

Client Write Maps Enabled:

Rule number simply tells you where you're at on the list of register maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Map #" box, then click Update.

The local register data may be written periodically, or when it changes, or both. To send upon change (send on delta), check the first box and enter the amount by which the local register must change before being written to the remote device. To guarantee that the remote register will be written at least occasionally even if the data does not change, check the second box and enter some amount of time. This time period will be referred to as the "maximum quiet time".

Data from the local register may be manipulated before being written to the remote register. The local data is first multiplied by the scale factor. The offset is then added to it. If a bit mask is entered, and the remote register type is signed or unsigned (16-bit data), the mask will be bit-wise logical AND-ed with the data. The mask is right justified, then AND-ed with the data. The result is then left shifted back to the original position of the mask. In other words, the least significant bits of the original data will be stuffed at the position marked by the mask.

After the scaling and masking, the bit fill will be logically OR-ed into the result, but only if the mask was nonzero and was used. Both mask and fill are entered in hexadecimal.

Multiple local registers may be packed into a single remote register. To accomplish this, define two or more rules in sequence with the same remote destination. If the destination is the same, data types are 16-bit (integer or unsigned), bit masks are nonzero, and the rules are sequential, the results of all qualifying rules will be OR-ed together before being sent to the remote destination.

For the remote register to be written, enter the register type, format, number, and remote unit (slave address).

The repeat time may determine how often the remote register will be written. If send on delta and maximum quiet time are not checked above, clicking the "at least" button will establish a periodic update time. If send on delta is used and you wish to limit the network traffic in the event changes are frequent, click the "no more than" button and enter the minimum time that should elapse before another write to the remote device.

Delete will remove the rule number shown in the "Map #" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having zero for a source register or "none" for remote type.

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type. If you wish to prevent these from being displayed, reduce the number of rules enabled.

The number of rules enabled simply limits the scope of rule review so that you do not have to review a lot of unused rules. If the displayed rules are used up and you need more, increase the enabled number.

3.4 Modbus RTU Master Data Displayed Per Slave

This page displays data to and from registers in devices accessed via the Modbus RTU serial port.

RTU Unit # 2 Showing 1 to 2 of 2 Update < Prev Next >

Dir	Reg. Type	Remote Reg. #	Register Name	Local Reg. #	Hex	Update	Register Data	Time since Last update
From	Holding Reg	00005	Test Register	00001	<input type="checkbox"/>	<input type="checkbox"/>	0	3.570
To	Holding Reg	00009	Another Test Reg	00002	<input type="checkbox"/>	<input type="checkbox"/>	0	3.670

RTU Unit # 2 + Unit - Unit

The values of Modbus registers that have been read from remote RTU serial devices is displayed here. One remote unit at a time is displayed. To display a different unit, change the RTU Unit #.

Simply click the Update button to view the most recent data. Enter a new value and check the Update box if the value should be changed when you click the Update button. Check the Hex box if you wish to view or enter values in hexadecimal (not recommended for floating point).

Click Update to view the most recent data values. Click "Prev" or "Next" to scroll through the list of registers. You may also enter a number in the "Showing" box to jump directly to a given register when

Update is clicked.

3.5 Modbus RTU Errors

This page displays error codes encountered in processing reads and writes via the Modbus RTU serial port.

Showing devices from

Unit #	Reset -->	Read Error	Offending Read Map #	Reset -->	Write Error	Offending Write Map #	Reset -->	Total Messages	No Responses	CRC Errors	Exceptions
1	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0	0
2	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	12	0	0	0
3	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0	0
4	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0	0
5	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0	0
6	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0	0
7	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0	0

The first occurrence of read and write errors are shown along with the map number that was being processed when the error occurred. Check the reset box and click update to clear it and possibly show the next error if there are more than one active error conditions.

A total count of all errors is also shown. This total is the sum of errors for all maps for this device. Check the reset box and click update to reset the counts. Click Update to view the most recent data values.

Error code indications of A/B indicate the following errors with the first number:

- 1 = Transaction ID out of sync
- 2 = Exception code returned by remote device
- 3 = Function code mismatch (bad packet)
- 4 = Insufficient data (bad packet)
- 5 = No response from remote device, timed out
- 6 = CRC error in received packet

When A is code 2 indicating an exception code was returned, B indicates the exception as follows:

- 1 = Illegal function code
- 2 = Illegal data address (the requested register does not exist in the device)
- 3 = Illegal data value



4. Configuring Gateway as a Modbus TCP Client (Master)

The BB2-6010/SPX/SP can be a Modbus client or server. As a client (master) you can read Modbus data from, or write Modbus data to, other Modbus servers (slaves). The gateway will periodically poll the other Modbus devices according to register maps you set up. The Modbus server (slave) devices that you will read/write are defined on the Devices page. To read from a remote Modbus device, configure a Read Map. To write to a remote Modbus device, configure Write Map.

Data read from a remote device is stored in a local register when received. Data written to a remote device is taken from a local register when sent. The local registers are the same collection of registers that are accessible to other masters when this gateway is operating as slave, and accessible to other Modbus RTU devices as a collection of holding registers. The local registers are also accessible as SNMP MIB variables.

4.1 Modbus TCP Device Configuration

The Modbus Devices page is where you define other Modbus TCP devices that this gateway will read registers from and/or write register to.

This page sets up the network address and optional device parameters for a remote Modbus/TCP device that will be linked to for remote input and/or output (via the client read and client write maps). The local device acts as a Modbus master to the remote devices listed below.

Device # Update < Prev Next >

IP Address Port: (default 502)

Domain Name Local Name:

Unit (optional) Use FC 5/6 instead of 15/16

Low register is first for multiple registers

Default Poll Period Seconds Connection Status

Device number simply shows you where you are on the device list. Click "next" and "prev" to scroll through the list.

Remote Modbus/TCP devices to be accessed by this device are specified here. Enter the IP address of the remote device, a name to reference in other pages, a unit number, poll rate, and check "swapped" if appropriate. Then click "update".

If your slave/server device only supports function codes 5 and 6 for writing, check the Use FC 5/6 box. The default function codes are 15 and 16, which are most widely used.

You may have the system look up the IP address given a domain name. If the IP address is set to zero, an attempt will be made to find the host by name. You also have the option of specifying a non-standard port number. If port is left set to zero, 502 will be used.

The term "swapped" only applies to double or float formats. Modbus registers are, by definition, 16 bits of data per register. Access to 32-bit data, either 32-bit integer ("double"), or IEEE 754 floating point ("float"), is supported by the use of two consecutive registers. Modbus protocol is inherently "big endian", therefore, Modbus by the Module defaults to having the high order register first for double and float. If the low order register comes first on the device being accessed, check the "swapped" box.

If you have "swapped" turned around, you will quickly recognize it. If floating point data is reversed, a 1.0 becomes 2.2779508e-41, which simply rounds to zero. The pattern is not as predictable as the 1.0 example would suggest. A floating point 1.1 becomes negative 107609184. If 32-bit integer data is reversed, 1 becomes 65536.

Connection status will show a non-zero error code if there is a socket error. Possible errors include:

- 5 = Connection failed, unable to bind (usually means remote device not connected or not reachable)
- 81 = Connection in progress (means unsuccessful connect attempt, still trying)
- 95 = Network is unreachable

- 97 = Connection aborted
- 98 = Connection reset by peer
- 103 = Connection timed out
- 104 = Connection refused
- 107 = Host is unreachable

4.2 Modbus TCP Client Read Maps

Getting the gateway to read registers from another Modbus device requires setting up a "Read Map" as shown here.



The screenshot shows the Babel Buster 2 Modbus Network Gateway web interface. The page title is "Babel Buster 2 MODBUS NETWORK GATEWAY MODEL BB2-6010" and the company is "CONTROL SOLUTIONS, INC. MINNESOTA". The navigation menu includes "RTU Serial Port", "IP Network", "System", "Diagnostic Data", "Modbus Setup", "SNMP Setup", "SNMP Client", "SNMP Diagnostics", "Devices", "Client Read Map", "Client Write Map", and "Server Map". The "Client Read Map" page contains the following text: "Read remote registers into local registers. This page creates a map entry that reads data from one or more remote Modbus/TCP servers for processing here. Click on map number to see more detail and insert/delete rules." Below this is a "Showing 1 to 2 of 2" indicator, an "Update" button, and "Prev" and "Next" buttons. The main table lists the following data:

Map #	Remote Type	Remote Register Format	Remote Register #	Remote Device	Scale	Local Register #	Name
1	Holding Register	Int 16-bit	2	SPX Demo	0.000000	3	Test Reg 3
2	None	Int 16-bit	0	None	0.000000	0	

Rule number simply tells you where you're at on the list of register maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Showing" box, then click Update.

Rules entered on this page only read data from remote devices. Go to the Client Write Map to write data to those devices. The full parameter set is different for read versus write.

An abbreviated version of a list of rules is shown on this page. Any of the parameters shown may be changed here and registered by clicking the Update button. To view and/or modify the complete set of parameters, click on the map number in the left most column.

For each remote register to be read, enter the register type, format, number, and location (device). The names in the device list are defined in the Devices page. Register numbers must be standard numbers starting at 1 (not 40001). The abbreviation "Int" under Format means signed integer, while "Uint" represents unsigned integer. The Float format refers to 32-bit IEEE 754 format. Bit is a single bit, and should only be used with coil or discrete input register types (you cannot read "Bit" from a holding register - see section 10.4 for reading bits from a holding register). The "Mod10" format is unique to Schneider Electric power meters, and is supported in 2, 3, and 4-register formats.

When the remote register is read, the data will be multiplied by the scale factor and written to the local register number given. The name is optional and used only for display purposes.

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until

deleted. Unused rules at the end of the list will always show none as the type.

This page creates a map entry that reads data from a remote Modbus/TCP server for processing here.

Map #

Read as from register # at unit #

Apply bit mask if applicable: then apply scale: and offset:

Save in local register # named Repeat this process every seconds.

Apply this default value: after read failure(s).

Client Read Maps Enabled:

Rule number simply tells you where you're at on the list of register maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Map #" box, then click Update.

For each remote register to be read, enter the register type, format, number, and location (device). The names in the device list are defined in the Devices page.

When the remote register is read, data may be manipulated before being written to the local register. If a bit mask is entered (in hexadecimal), and the remote register type is signed or unsigned (16-bit data), the mask will be bit-wise logical AND-ed with the data, and the retained bits will be right justified in the result. The result will then be multiplied by the scale factor. The offset is then added and this final result is written to the local register number given. The name is optional and used only for display purposes.

The periodic poll time determines how often the remote register will be read. This number, if nonzero, will override the default poll time given in the Devices page for the remote device being read.

The default value will be stored into the local register after the given number of read failures if the fail count is non-zero. Setting the count to zero will disable the default, and the register will retain the most recent value obtained.

Delete will remove the rule number shown in the "Map #" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having zero for a source register or "none" for remote type.

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type. If you wish to prevent these from being displayed, reduce the number of rules enabled.

The number of rules enabled simply limits the scope of rule review so that you do not have to review a lot of unused rules. If the displayed rules are used up and you need more, increase the enabled number.

4.3 Modbus TCP Client Write Maps

Getting the gateway to write registers to another Modbus device requires setting up a "Write Map" as shown here.

Write local registers out to remote registers. This page creates a map entry that writes data to one or more remote Modbus/TCP servers from data contained here. Click on map number to see more detail and insert/delete rules.

Showing 1 to 2 of 2

Map #	Local Register #	Scale	Remote Type	Remote Register Format	Remote Register #	Remote Device	Name
1	6	0.000000	Holding Register	Int 16-bit	4	SPX Demo	Test Reg 4
2	0	0.000000	None	Int 16-bit	0	None	

Rule number simply tells you where you're at on the list of register maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Showing" box, then click Update.

Rules entered on this page only write data to remote devices. Go to the Client Read Map to read data from those devices. The full parameter set is different for read versus write.

An abbreviated version of a list of rules is shown on this page. Any of the parameters shown may be changed here and registered by clicking the Update button. To view and/or modify the complete set of parameters, click on the map number in the left most column.

Data from the local register given will be multiplied by the scale factor before being written. For each remote register to be written, enter the register type, format, number, and location (device). The names in the device list are defined in the Devices page. The name is optional and used only for display purposes.

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type.

Babel Buster 2
MODBUS NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | IP Network | System

Diagnostic Data | Modbus Setup | SNMP Setup | SNMP Client | SNMP Diagnostics

Devices | Client Read Map | Client Write Map | Server Map

This page creates a map entry that writes data to one or more remote Modbus/TCP servers from data contained here.

Map #

Read local register # named

Write remote register when local register changes by > or when seconds have elapsed with no change.

Otherwise write remote register unconditionally, applying local register data as follows:

Apply scale: and offset: Then if applicable, apply bit mask: and bit fill:

Write as to register # at unit #

Repeat this process at least no more than every seconds.

Client Write Maps Enabled:

Rule number simply tells you where you're at on the list of register maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Map #" box, then click Update.

The local register data may be written periodically, or when it changes, or both. To send upon change (send on delta), check the first box and enter the amount by which the local register must change (more than) before being written to the remote device. To guarantee that the remote register will be written at least occasionally even if the data does not change, check the second box and enter some amount of time. This time period will be referred to as the "maximum quiet time".

Data from the local register may be manipulated before being written to the remote register. The local data is first multiplied by the scale factor. The offset is then added to it. If a bit mask is entered, and the remote register type is signed or unsigned (16-bit data), the mask will be bit-wise logical AND-ed with the data. The mask is right justified, then AND-ed with the data. The result is then left shifted back to the original position of the mask. In other words, the least significant bits of the original data will be stuffed at the position marked by the mask.

After the scaling and masking, the bit fill will be logically OR-ed into the result, but only if the mask was nonzero and was used. Both mask and fill are entered in hexadecimal.

Multiple local registers may be packed into a single remote register. To accomplish this, define two or more rules in sequence with the same remote destination. If the destination is the same, data types are 16-bit (integer or unsigned), bit masks are nonzero, and the rules are sequential, the results of all qualifying rules will be OR-ed together before being sent to the remote destination.

For the remote register to be written, enter the register type, format, number, and location (device). The names in the device list are defined in the Devices page.

The repeat time may determine how often the remote register will be written. If send on delta and maximum quiet time are not checked above, clicking the "at least" button will establish a periodic update time. If send on delta is used and you wish to limit the network traffic in the event changes are frequent, click the "no more than" button and enter the minimum time that should elapse before another write to the remote device.

Delete will remove the rule number shown in the "Map #" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having zero for a source register or "none" for remote type.

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type. If you wish to prevent these from being displayed, reduce the number of rules enabled.

The number of rules enabled simply limits the scope of rule review so that you do not have to review a lot of unused rules. If the displayed rules are used up and you need more, increase the enabled number.

4.4 Modbus TCP Errors

Modbus TCP error counts are shown here. In addition to checking error counts here, be sure to check the TCP device status on the Modbus Setup :: Devices page as this will indicate errors due to TCP connection problems.

This page displays error codes encountered in processing Modbus Client reads and writes via the Modbus TCP connection(s).

Device	Reset -->	Read Error	Offending Read Map #	Reset -->	Write Error	Offending Write Map #	Reset -->	Total Messages	No Responses	Exceptions
1	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	77	0	0
2	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0
3	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0
4	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0
5	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0
6	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0
7	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	0	0

The first occurrence of read and write errors are shown along with the map number that was being processed when the error occurred. Check the reset box and click update to clear it and possibly show the next error if there are more than one active error conditions.

A total count of all errors is also shown. This total is the sum of errors for all maps for this device. Check

the reset box and click update to reset the counts. Click Update to view the most recent data values.

Error code indications of A/B indicate the following errors with the first number:

- 1 = Transaction ID out of sync
- 2 = Exception code returned by remote device
- 3 = Function code mismatch (bad packet)
- 4 = Inusfficient data (bad packet)
- 5 = No response from remote device, timed out
- 6 = CRC error in received packet

When A is code 2 indicating an exception code was returned, B indicates the exception as follows:

- 1 = Illegal function code
- 2 = Illegal data address (the requested register does not exist in the device)
- 3 = Illegal data value



5. Configuring Gateway as a Modbus RTU Slave

The BB2-6010/SPX/SP can be a Modbus RTU master or slave. As slave, the gateway will respond to another Modbus master and return data requested. The various objects in the gateway are accessed as holding registers, with register numbers calculated and based on object type and instance.

5.1 Modbus RTU Device Configuration

Modbus device configuration for RTU really consists of port configuration, and includes setting the slave address if the gateway is functioning as Modbus slave.

A screenshot of the Babel Buster 2 Modbus Network Gateway configuration interface. The interface has a green theme and a navigation menu at the top with tabs for "RTU Serial Port", "IP Network", and "System". Under "RTU Serial Port", there are sub-tabs for "Data" and "Setup". Below these are "Local Device", "RTU Read Map", and "RTU Write Map". The main content area shows configuration parameters for the Modbus RTU serial port. It includes a "Baud Rate" dropdown set to "19200" and a "Parity" dropdown set to "None, 1 Stop Bit". There is an "Update" button. Below this, there are two radio buttons: "I am the Master" (unselected) and "I am a Slave" (selected). Under "I am the Master", there are fields for "Default Poll Rate" (5.000) and "Timeout" (1.000), both in seconds. Under "I am a Slave", there is a field for "My Address or Unit #" (1), a checkbox for "Double registers are swapped" (unchecked), and a checkbox for "Use FC 5/6 instead of 15/16 for unit numbers (slave addresses) starting at" (0).

Select baud rate and parity from the drop down list. Click either Master or Slave buttons to select type of operation. Enter timing parameters or address as applicable. Click update to register your changes.

The double register swap on this page only applies when the local device (the gateway you are configuring here) is functioning as a Modbus RTU slave. If the Modbus master expects least significant data to be in the first (lowest numbered) register, then check the "swap" box.

The term "swapped" only applies to double or float formats. Modbus registers are, by definition, 16 bits of data per register. Access to 32-bit data, either 32-bit integer ("double"), or IEEE 754 floating

point ("float"), is supported by the use of two consecutive registers. Modbus protocol is inherently "big endian", therefore, Modbus by the Module defaults to having the high order register first for double and float. If the low order register comes first on the device being accessed, check the "swapped" box.

If you have "swapped" turned around, you will quickly recognize it. If floating point data is reversed, a 1.0 becomes 2.2779508e-41, which simply rounds to zero. The pattern is not as predictable as the 1.0 example would suggest. A floating point 1.1 becomes negative 107609184. If 32-bit integer data is reversed, 1 becomes 65536.

5.2 Modbus RTU Slave Register Mapping

The mappings shown below are used when the gateway is treated as a Modbus RTU slave. All local registers are accessed as holding registers. For floating point register pairs, the high order or most significant half of the data value is contained in the first (lowest numbered) holding register. Floating point values must be read as register pairs, and must be read on odd numbered boundaries.

Register Type	Register Number Range	Read As
Integer	1 - 999	Single 16-bit registers
Floating Point	1001-1999	Pairs of 16-bit registers (IEEE 754 floating point)

The Server Map discussed in section 6.3 allows the user to create a "virtual" Modbus device. The Server Map allows the local registers in the gateway, normally only accessed as holding registers, to be accessed as other Modbus types such as coils instead. The Server Map also allows defining a specific register set for those applications where the gateway needs to look exactly like some other existing piece of equipment to be replaced by the gateway.

Registers mapped in the Server Map are accessible from Modbus TCP in BB2-6010, SPX, and SP. Registers mapped are also accessible from Modbus RTU (when configured as slave) in BB2-6010 and SPX.



6. Configuring Gateway as a Modbus TCP Server (Slave)

The term "server" is often used to describe the Modbus TCP version of a Modbus slave. A server will provide data when a client asks for it. The concept of master/slave is less significant in Modbus TCP because any TCP device can be both master and slave at the same time, and there can be multiple "masters" on the network. That is in contrast with Modbus RTU where there can be only one master and multiple slaves, and each device must be one or the other.

The Modbus TCP server is simply a collection of registers that may contain data. The source of that data in the case of Babel Buster BB2-6010/SPX/SP can be any of several possible sources. It may be read from another Modbus device. Another Modbus device could have put it there by writing to the gateway. The data could have been received by the SNMP client or SNMP server.

6.1 Modbus TCP Device Configuration

There is really nothing to configure to get the gateway to function as a Modbus TCP server (slave) other than set the gateway's IP address. The only exception is if you want to remap the local registers using the Server Map, but this is optional.

This page allows you to change this device's IP address, and select whether double registers are swapped when returned to a remote client accessing this server.

IP Address	<input type="text" value="192.168.1.88"/>	192.168.1.88	<input type="button" value="- Refresh -"/>
Subnet Mask	<input type="text" value="255.255.255.0"/>	255.255.255.0	<input type="button" value="Change IP"/>
Gateway	<input type="text" value="192.168.1.1"/>	192.168.1.1	

To change the IP address of this device, enter the address, subnet mask, and gateway, then click "change IP". Set the IP address to 255.255.255.255 to specify that DHCP should be used to obtain an IP address upon power-up. IP address change will take effect upon next power cycle.

6.2 Modbus TCP Register Mapping

The mappings shown below are used when the gateway is treated as a Modbus TCP slave. All local registers are accessed as holding registers. For floating point register pairs, the high order or most significant half of the data value is contained in the first (lowest numbered) holding register. Floating point values must be read as register pairs, and must be read on odd numbered boundaries.

Register Type	Register Number Range	Read As
Integer	1 - 999	Single 16-bit registers
Floating Point	1001-1999	Pairs of 16-bit registers (IEEE 754 floating point)

6.3 Modbus Virtual Device Register Mapping

The Server Map allows the user to create a "virtual" Modbus device. The Server Map allows the local registers in the gateway, normally only accessed as holding registers, to be accessed as other Modbus types such as coils instead. The Server Map also allows defining a specific register set for those applications where the gateway needs to look exactly like some other existing piece of equipment to be replaced by the gateway.

Registers mapped in the Server Map are accessible from Modbus TCP in BB2-6010, SPX, and SP. Registers mapped are also accessible from Modbus RTU (when configured as slave) in BB2-6010 and SPX.



Babel Buster 2
MODBUS NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | **IP Network** | System

Diagnostic Data | **Modbus Setup** | SNMP Setup | SNMP Client | SNMP Diagnostics

Devices | Client Read Map | Client Write Map | **Server Map**

Create remote client's custom view of local registers. This page sets up the register map for the virtual Modbus/TCP server. This map is also referred to as the "user map". This allows you to remap the default server register map to match any layout you wish, including matching the map found in other equipment.

Showing 1 to 3 of 3 Update < Prev Next >

Map #	Mapped Register #	Mapped Register Format	Local Register #	Scale Factor	Offset	Bit Field	Fill	Name
1	40001	Unsigned Integer ▾	1	0.000000	0.000000	0000	0000	Modicon reg 1
2	40002	Unsigned Integer ▾	2	0.000000	0.000000	0000	0000	Modicon reg 2
3	0	None ▾	0	0.000000	0.000000	0000	0000	

Custom Registers Enabled: 3

User Map Enabled Map is Exclusive
 Swap Double Registers Zero fill null registers
 Use Modicon mapping

Insert Delete

For each register to be mapped into the custom map, enter the server address where this register should appear, the format it should be presented in, and the source of the data. Scale factor is optional. The source data will be multiplied by this to produce the data in the mapped server register. Offset is optional. This value will be added to the source data after multiplying by the scale factor.

Bit field and fill allow compiling register contents derived from multiple sources if the bit field is defined (nonzero). The source data will be limited to the number of bits represented in the bit field (which is a hexadecimal value), and shifted into the position represented by '1' bits in the field. Fill bits will be logically OR'ed into the result before being presented by the server. Consecutive server map entries that reference the same server address will all be OR'ed together and presented at that address. Duplicate map entries that reference the same server address but are not listed in consecutive order following the first instance will be skipped. No special bit field processing takes place if the bit field is set to zero. Bit fields apply to 16-bit integer or unsigned integer server registers only.

The name is optional and is used for display purposes only.

Delete will remove the rule number shown in the "Showing" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having "none" for register format.

Selecting "none" as the register format effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show "none" as the format. If you wish to prevent these from being displayed, reduce the number of rules enabled.

Enter the number of Modbus registers that should be available in your customized register mapping and check "User Map Enabled" to begin using a customized map. Check "Map is Exclusive" if access to registers outside of this map should be prohibited. If exclusive is not selected, all local registers not overlapped by the custom map will also be accessible to the remote client.

By default, double registers in Control Solutions products are "big endian" meaning the most significant bytes are in the first register and least significant bytes are in the second register. If remote clients accessing this server at this IP address expect "little endian", check the swap box. Modbus protocol by definition is "big endian" within each register, but the "endian" order of the registers for 32-bit values is less standardized.

Normally an attempt to read an undefined register will return an exception (error) code. To enable reading of large data packets without nuisance errors, you have the option of zero filling null registers. This means that reading an undefined register in between valid defined registers will simply return zero data rather than an error.

Check "Use Modicon mapping" to map 0X, 1X, 3X and 4X registers anywhere in i.CanDoIt register space. When you use Modicon mapping, the Mapped Register number should be in the following ranges:

Mapped Register #	Read (Write) as	Function codes expected
0-9999	Coil	1, 5, 15
10001-19999	Discrete Input (read only)	2
30001-39999	Input Register (read only)	4
40001-49999	Holding Register	3, 6, 16

Any of the Modicon register types may be mapped to any local register, except that coils and discrete inputs cannot map to floating point registers. When a local register is read as coil or discrete input, any nonzero value in the local register will return a set bit, and zero in the local register will return a clear bit. Local registers written as coils will be set to 0 or 1.

To use Modicon mapping, you must check the Use Modicon box, and also check User Map Enabled. It is also highly recommended that you check the Map is Exclusive box when using Modicon mapping. Remember to go to the Config File page and save your changes.

When using the Server Map, you may check the outside world's view of your device by going to the IP Network :: Diagnostic Data :: Modbus Server page, which is illustrated below.



Babel Buster 2
MODBUS NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | **IP Network** | System

Diagnostic Data | Modbus Setup | SNMP Setup | SNMP Client | SNMP Diagnostics

Modbus Server | Error Codes

This page displays data as mapped by the virtual server "user map" defined under Modbus->Setup->Server Map. This is a snap shot of what the remote client will see when the user map is enabled.

Showing 1 to 3 of 3 Update < Prev Next >

Map #	Mapped Register #	Register Name	Register Data
1	40001	Modicon reg 1	59278
2	40002	Modicon reg 2	29
3	00000		Not found

Diagnostic Info

1: 0/0
2: 0/0
3: 0/0
4: 0/0
5: 0/0



7. Configuring Gateway as an SNMP Server (Agent)

The Babel Buster BB2-6010/SPX/SP can act as an SNMP agent or server. The SNMP agent's MIB is largely preconfigured, but you do need to configure trap generation.

7.1 Local SNMP MIB

The MIB as defined for the gateway is shown on the MIB View page illustrated here. The OIDs are those you would use from an external SNMP manager to Get or Set data in the gateway's local registers.

This page displays data as presently found in the local registers maintained by this device.

Showing registers from

Local Register #	Register Name	Update	Register Data	Register Type	SNMP MIB OID
00001	Test Register	<input type="checkbox"/>	0	Integer	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.1.2.1
00002	Another Test Reg	<input type="checkbox"/>	0	Integer	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.1.2.2
00003	Test Reg 3	<input type="checkbox"/>	121	Integer	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.1.2.3
00004	Test Reg 4	<input type="checkbox"/>	0	Integer	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.1.2.4
00005		<input type="checkbox"/>	0	Integer	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.1.2.5
00006		<input type="checkbox"/>	0	Integer	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.1.2.6

The values of data registers contained within this device are displayed on this page, and optionally changed. Check the Update box if the value should be updated, and enter a new value.

Click Update to view the most recent data values. Click "Prev" or "Next" to scroll through the list of registers. You may also enter a number in the "Showing" box to jump directly to a given register when Update is clicked.

This page shows registers as organized for Modbus access, and shows the corresponding SNMP MIB OID. Values from I/O points are mirrored as integer starting in register #1 and floating point starting in register

#1001. The integer and floating point registers are found in two separate branches of the MIB.

First integer register: 1.3.6.1.4.1.3815.1.2.2.1.1.1.1.1.2.1

Second integer register: 1.3.6.1.4.1.3815.1.2.2.1.1.1.1.1.2.2 etc.

First floating point register: 1.3.6.1.4.1.3815.1.2.2.1.1.2.1.1.2.1

7.2 SNMP Trap Destinations

The SNMP Trap receivers that the gateway will send traps to are defined here.

The screenshot shows the web interface for the Babel Buster 2 Modbus Network Gateway. The main navigation bar includes tabs for RTU Serial Port, IP Network, System, Diagnostic Data, Modbus Setup, SNMP Setup, SNMP Client, and SNMP Diagnostics. The SNMP Setup page is currently selected, displaying the following configuration options:

- Device #: 1
- IP Address: 192.168.1.101
- Local Name: Monitoring System
- Domain Name: (empty field)
- Group Membership: Group 1, Group 2, Group 3

Buttons for 'Update', '< Prev', and 'Next >' are visible at the top right of the configuration area.

Device number simply shows you where you are on the device list. Click "next" and "prev" to scroll through the list.

SNMP managers that traps should be sent to are listed here. The IP address is used to send the trap message. The system name is simply used for display purposes.

You may leave the IP address set to 0.0.0.0 and enter a domain name instead. The nameservers identified as DNS1 and DNS2 in the System :: Setup :: Network page will be used to try to look up the IP address by host (domain) name.

Traps, when enabled on the Trap Enable page, are sent to one of three "groups". The IP address shown here may be a member of one or more groups. If a trap is to be sent to a particular group, all members of that group will receive the trap.

7.3 SNMP Trap Triggers

Traps are triggered by Threshold rules defined under System :: Action Rules. To generate traps, you must first set up one or more Threshold rules. Then come to the Trap Enable page to select which trap receivers the resulting traps will be sent to.

This page enables and configures SNMP trap messages available for threshold rules.

Showing 1 to 2 of 2

Rule #	Event Name	Trap on True	Trap on False	Repeat Count	Repeat Time	Enable Group 1	Enable Group 2	Enable Group 3
1	Test Reg High	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	0	0.000000	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2		<input type="checkbox"/>	<input type="checkbox"/>	0	0.000000	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Rules defined under System :: Action Rules :: Thresholds are listed here along with options for enabling SNMP trap messages triggered by these threshold rules. Check the "Trap on..." boxes to turn on traps to the groups checked on the right.

The repeat count is the number of times the same trap will be sent when triggered. This number of traps will be sent at approximately 100 millisecond intervals. The repeat time is the delay period between re-transmissions of the trap, or series of traps as determined by the repeat count. Repeat time is in seconds. Example: If repeat count is set to 3, and repeat time is set to 60 seconds, then three trap messages will be sent in a burst and this burst will be repeated once every minute.

The following information is sent in a Trap message: The local register number, that register's name, that register's data at the time the trap event occurred, the event name, the test type, the test value or threshold, and the event state. The "events" are defined by Threshold Rules under System :: Action Rules :: Thresholds.

7.4 SNMP Identity

Use this page to set the system information that will be retrieved via SNMP to identify this gateway as an SNMP agent.

The screenshot displays the web interface for the Babel Buster 2 Modbus Network Gateway. The header features the product name 'Babel Buster 2' and 'CONTROL SOLUTIONS, INC. MINNESOTA'. The navigation menu includes 'RTU Serial Port', 'IP Network', 'System', 'Diagnostic Data', 'Modbus Setup', 'SNMP Setup', 'SNMP Client', and 'SNMP Diagnostics'. The 'SNMP Setup' page contains a description: 'This page displays and edits SNMP system information.' Below this is an 'Update' button and a form with the following fields:

System Name	Unassigned Babel Buster BB2-6010
System Location	White Bear Lake, Minnesota
System Contact	www.csimn.com

7.5 Testing the SNMP Agent

You can obtain a free SNMP Browser tool called ManageEngine from Zoho Corp. by visiting www.manageengine.com. Using ManageEngine, you can perform SNMP Gets and Sets on your gateway.

Start by downloading and installing ManageEngine MibBrowser. Then, before you can effectively use it, you need to load the CSI MIB files available at csimn.com. You need to load two files from the "current MIBs" directory in the CSI MIB library. First, load the file "CSIreg.mib". Second, load the file "CSImodbus.mib". Once you do this, you will be able to expand the MIB tree as illustrated below.

The screenshot shows the ManageEngine MibBrowser Free Tool interface. On the left, a tree view displays the loaded MIB modules, with 'dataFloatRegister' selected under the path: CSI-BB2-MODBUS-MIB > csi > csiProduct > modbus > bb2-6010 > modbusObjects > dataRegs > dataRegInt > dataRegFloat > dataFloatRegister. The right pane shows the configuration for the selected object, including Host (192.168.1.87), Port (161), and Object ID (.1.3.6.1.4.1.3815.1.2.2.1.1.2.1.1.2). The Object ID field is populated with 'jects.dataRegs.dataRegFloat.dataFloatTable.dataFloatEntry.dataFloatRegister'. Below this, a log shows the execution of a GET request to 192.168.1.87:161, returning the value 1025.000000 for dataFloatRegister.1. The bottom pane displays the object's details, including its syntax (DisplayString (SIZE (0..40))), access (read-write), and description ('Register's data value.').

Assuming you have successfully loaded the MIBs, select "dataFloatRegister" to Get a floating point register (register pairs starting at 1001), or select "dataIntRegister" to Get an integer register (in the range of 1-999). The Object ID window in ManageEngine will contain only the OID for the table as a whole. You need to add an index to the end of the OID. The OID...

. iso . org . dod . internet . private . enterprises . csi . csiProduct . modbus . bb2-6010 . modbusObjects . dataRegs . dataRegFloat . dataFloatTable . dataFloatEntry . dataFloatRegister

needs to become...

. iso . org . dod . internet . private . enterprises . csi . csiProduct . modbus . bb2-6010 . modbusObjects . dataRegs . dataRegFloat . dataFloatTable . dataFloatEntry . dataFloatRegister . 1

to access the first floating point register (pair). The screen shot illustrated above shows the results of performing a Get on the first floating point OID, returning the data illustrated in the screen below as found in the BB2-6010 at the time.

Babel Buster 2
MODBUS
NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port IP Network **System**

Data Action Rules Setup

Local Registers Thresholds Trend Data MIB View

This page displays data as presently found in the local registers maintained by this device.

Showing registers from 1001 Update < Prev Next >

Local Register #	Register Name	Update	Register Data	Register Type	SNMP MIB OID
01001	Test Register	<input type="checkbox"/>	1025.000	Float	1.3.6.1.4.1.3815.1.2.2.1.1.2.1.1.2.1
01003		<input type="checkbox"/>	0.000000	Float	1.3.6.1.4.1.3815.1.2.2.1.1.2.1.1.2.2
01005		<input type="checkbox"/>	0.000000	Float	1.3.6.1.4.1.3815.1.2.2.1.1.2.1.1.2.3

You can also test the gateway's generation of SNMP traps using ManageEngine's trap viewer. Select "Trap Viewer" under the View menu. After receipt of a trap, the Trap Viewer window looks something like this:

TrapViewer

Class	Type	Source	Date	Message
Clear	v2c Trap	192.168.1.87	Fri Dec 11 09:55:431.3.6.1.2.1.1.3.0: TimeTicks: 4...

Authenticate v1/v2c traps (Community) Enable Logging Log Format

Authenticate v3 Trap Enable Mail Configure Mail

Port 162 TrapList 162:public Add Del

Community public TrapParser Load

Start Stop Show Details Delete Entry ParserEditor

Traps : 1 Inform : 0 Status : Listening for Traps

To get traps to be viewed on your PC, you need to go into the Windows firewall and allow SNMP traps. Then set up a trap destination in the gateway using your PC's IP address as illustrated here:

RTU Serial Port IP Network System

Diagnostic Data Modbus Setup **SNMP Setup** SNMP Client SNMP Diagnostics

Devices Trap Enable Identity

This page sets up the network address and group membership for a remote SNMP manager which traps will be sent to.

Device #

IP Address Local Name:

Domain Name

Group Membership Group 1 Group 2 Group 3

Enable one of your threshold rules for trap generation as shown here:

RTU Serial Port IP Network System

Diagnostic Data Modbus Setup **SNMP Setup** SNMP Client SNMP Diagnostics

Devices Trap Enable Identity

This page enables and configures SNMP trap messages available for threshold rules.

Showing to 2 of 2

Rule #	Event Name	Trap on True	Trap on False	Repeat Count	Repeat Time	Enable Group 1	Enable Group 2	Enable Group 3
1	Test Reg High	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="text" value="0"/>	<input type="text" value="0.000000"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2		<input type="checkbox"/>	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="text" value="0.000000"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

If your gateway is polling a device to obtain register data, cause your device to produce data that will result in a trap. But you can also force data values via the web interface if your gateway is not yet connected to a device. Simply go to the System :: Data :: Local Registers page, enter a value, check the Update box, and click the Update button.

Babel Buster 2
MODBUS
NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | IP Network | **System** | | |

Data | Action Rules | Setup | | |

Local Registers | **Thresholds** | Trend Data | MIB View | |

This page displays data as presently found in the local registers maintained by this device.

Showing registers from

Local Register #	Register Name	Hex	Update	Register Data	Unsigned	Register Type	Default Data	Server Timeout (s)
00001	Test Register 1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	122	<input type="checkbox"/>	Integer	0.000000	0.0

In our example, clicking Update above produced the trap shown in the Trap Viewer window above. If you right click on the trap and select Show Detail, you will see the full content of the trap as follows:

Trap Details	
TimeStamp	42 days, 18 hours, 37 minutes, 50 seconds.
Enterprise	
Generic Type	
Specific Type	
Message	.1.3.6.1.2.1.1.3.0: TimeTicks: 42 days, 18 hours, 37 minutes, 50 seconds.: .1.3.6.1.6.3.1.1.4.1.0: Object ID: .1.3.6.1.4.1.3815.1.2.2.2.0.1: .iso.org.dod.internet.private.enterprises.csi.csiProduct.modbus.bb2-6010.modbusC try.eventRegNum.1.0: INTEGER: 1: .iso.org.dod.internet.private.enterprises.csi.csiProduct.modbus.bb2-6010.modbusC try.eventRegName.1.0: Test Register 1: .iso.org.dod.internet.private.enterprises.csi.csiProduct.modbus.bb2-6010.modbusC try.eventRegData.1.0: 122.000000: .iso.org.dod.internet.private.enterprises.csi.csiProduct.modbus.bb2-6010.modbusC try.eventName.1.0: Test Reg High: .iso.org.dod.internet.private.enterprises.csi.csiProduct.modbus.bb2-6010.modbusC try.eventTestType.1.0: INTEGER: 1: .iso.org.dod.internet.private.enterprises.csi.csiProduct.modbus.bb2-6010.modbusC try.eventTestVal.1.0: 100.000000: .iso.org.dod.internet.private.enterprises.csi.csiProduct.modbus.bb2-6010.modbusC try.eventState.1.0: INTEGER: 1: .1.3.6.1.6.3.1.1.4.3.0: Object ID: .1.3.6.1.4.1.3815.1.2.2.2:
Severity	Clear
Entity	192.168.1.87
RemotePort	2960
LocalPort	162
Community	public
Node	192.168.1.87
Source	192.168.1.87
TimeReceived	Fri Dec 11 09:59:43 CST 2015
HelpURL	0-0.html

Another very useful tool for trouble shooting SNMP is Wireshark, available at no cost at www.wireshark.org. One of the more useful things Wireshark will do is simply see whether there is any network traffic actually happening. If Wireshark is not running on the same PC as ManageEngine (or other software of interest), then you will need to connect the devices of interest using an old 10BaseT hub rather than faster switch because switches are actually a type of unmanaged router that filters all traffic not addressed to the physical port, meaning Wireshark will not see traffic between other devices.

The trap received in the above example looks like this in Wireshark:

Capturing from Local Area Connection [Wireshark 1.12.2 (v1.12.2-0-g898fa22 from master-1.12)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
1965	520.420957	192.168.1.109	192.168.1.87	TCP	62	80-55310 [SYN, ACK] Seq=0 Win=8192 Len=0 MSS=1460
1966	520.431478	192.168.1.87	192.168.1.109	TCP	62	80-55310 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460
1967	520.431547	192.168.1.109	192.168.1.87	TCP	54	55310-80 [ACK] Seq=1 Ack=1 Win=65700 Len=0
1968	520.431808	192.168.1.109	192.168.1.87	HTTP	2227	POST /Forms/pgSysdata_1 HTTP/1.1 (application/javascript)
1969	520.554499	192.168.1.87	192.168.1.109	SNMP	361	snmpv2-trap 1.3.6.1.2.1.1.3.0 1.3.6.1.6.3.1.1.4.1.0: 1.3.6.1.4.1.3815.1.2.2.2.0.1 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.2.1.0: 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.3.1.0: 3132322e303030303030 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.4.1.0: 546573742052656769737465722031 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.5.1.0: 54657374205265672048696768 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.6.1.0: 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.7.1.0: 3130302e303030303030 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.8.1.0: 1.3.6.1.6.3.1.1.4.3.0: 1.3.6.1.4.1.3815.1.2.2.2 (iso.3.6.1.4.1.3815.1.2.2.2)
1970	520.617228	192.168.1.87	192.168.1.109	HTTP	204	HTTP/1.1 303 See other
1971	520.617642	192.168.1.109	192.168.1.87	TCP	54	55310-80 [FIN, ACK] Seq=2174 Ack=151 Win=0 Len=0

Frame 1969: 361 bytes on wire (2888 bits), 361 bytes captured (2888 bits) on interface 0

- Ethernet II, Src: Digiboar_30:af:ba (00:40:9d:30:af:ba), Dst: Dell_1a:23:86 (18:03:73:1a:23:86)
- Internet Protocol Version 4, Src: 192.168.1.87 (192.168.1.87), Dst: 192.168.1.109 (192.168.1.109)
- User Datagram Protocol, Src Port: 2960 (2960), Dst Port: 162 (162)
- Simple Network Management Protocol
 - version: v2c (1)
 - community: public
 - data: snmpv2-trap (7)
 - snmpv2-trap
 - request-id: 16
 - error-status: noError (0)
 - error-index: 0
 - variable-bindings: 10 items
 - 1.3.6.1.2.1.1.3.0: 369587073
 - 1.3.6.1.6.3.1.1.4.1.0: 1.3.6.1.4.1.3815.1.2.2.2.0.1 (iso.3.6.1.4.1.3815.1.2.2.2.0.1)
 - 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.2.1.0:
 - 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.3.1.0: 3132322e303030303030
 - 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.4.1.0: 546573742052656769737465722031
 - 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.5.1.0: 54657374205265672048696768
 - 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.6.1.0:
 - 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.7.1.0: 3130302e303030303030
 - 1.3.6.1.4.1.3815.1.2.2.1.2.1.1.8.1.0:
 - 1.3.6.1.6.3.1.1.4.3.0: 1.3.6.1.4.1.3815.1.2.2.2 (iso.3.6.1.4.1.3815.1.2.2.2)

```

0000  18 03 73 1a 23 86 00 40 9d 30 af ba 08 00 45 00  ..s.#..@ .O...E.
0010  01 5b ed b7 00 00 3c 11 0b c6 c0 a8 01 57 c0 a8  .[....<. ....W..
0020  01 6d 0b 90 00 a2 01 47 e5 57 30 82 01 3b 02 01  .m....G .w0.;..
0030  01 04 06 70 75 62 6c 69 63 a7 82 01 2c 02 02 00  ...publi c.....
0040  10 02 01 00 02 01 00 30 82 01 1e 30 10 06 08 2b  ....0 ...O...+
0050  06 01 02 01 01 03 00 43 04 16 07 73 81 30 1b 06  ....C ...S.O..
0060  0a 2b 06 01 06 03 01 01 04 01 00 06 0d 2b 06 01  .+..... ..+..
0070  04 01 04 67 01 02 02 02 00 01 30 16 06 11 2b 06  .n .....n .+

```

Local Area Connection: <live capture in prog... Profile: Default



8. Configuring Gateway as an SNMP Client (Manager)

The Babel Buster BB2-6010/SPX (but not SP) has the ability to be an SNMP client. In "master/slave" terms, this would be the master. Configuring the SNMP client starts with defining one or more SNMP devices that will be queried. Then, like the other possible client functions in the gateway, you set up read and write maps. A "read map" will use SNMP Get to query the device, and a "write map" will use SNMP Set to write to the device.

The SNMP Client configuration pages are illustrated below along with a summary of how to use them.

8.1 SNMP Device Configuration

The SNMP devices that will become the "slaves" or servers to be queried are defined here.

This page sets up the network address for a remote SNMP device that will be linked to local objects via the client read and client write maps. The local device acts as an SNMP client (manager) to the remote agents listed below.

Device # Update < Prev Next >

IP Address Local Name:

SNMP Version v1 v2c

SNMP Community

Default Poll Period Seconds Device Status

Device number simply shows you where you are on the device list. Click "next" and "prev" to scroll through the list.

Remote SNMP devices to be accessed by this device are specified here. Enter the IP address of the remote

device, a name to reference in other pages, and a default poll rate. Then click "update".

This gateway expects to access SNMP devices via the standard port 161.

Device status will be a number from 1 to 5 for the following application level error conditions:

- 1 = could not bind socket
- 2 = could not build SNMP packet
- 3 = problem with map, could not parse OID or bad type
- 4 = response timed out
- 5 = unable to parse data

Error codes from 80 and up are IP stack errors. Since SNMP uses UDP, which is connection-less, the only likely errors are:

- 95 = Network is unreachable
- 107 = Host is unreachable

8.2 SNMP Client Read Maps

The list of OIDs in the remote SNMP devices that are to be read is defined here. Data is read from the SNMP device and placed in a local register in the gateway.

The screenshot shows the web interface for the Babel Buster 2 Modbus Network Gateway. The page title is "Babel Buster 2 MODBUS NETWORK GATEWAY MODEL BB2-6010" and the company is "CONTROL SOLUTIONS, INC. MINNESOTA". The navigation menu includes "RTU Serial Port", "IP Network", "System", "Diagnostic Data", "Modbus Setup", "SNMP Setup", "SNMP Client", and "SNMP Diagnostics". The "SNMP Client" section is active, showing "Client Read Map" and "Client Write Map" options. Below the navigation, there is a description: "Read remote SNMP OIDs into local registers. This page creates a map entry that reads data from remote SNMP agents for processing here. Click on map number to see more detail and insert/delete rules." The interface shows "Showing 1 to 3 of 3" and "Update", "< Prev", "Next >" buttons. A table displays the current configuration:

Map #	Remote SNMP OID	Remote Device	Local Register #	Local Register Name
1	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.2.1	Test System	10	Test SNMP reg 1
2	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.2.2	Test System	11	Test SNMP reg 2
3		None	0	

Rule number simply tells you where you're at on the list of OID maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Showing" box, then click Update.

Rules entered on this page only read data from remote devices. Go to the Client Write Map to write data to those devices. The full parameter set is different for read versus write.

An abbreviated version of a list of rules is shown on this page. Any of the parameters shown may be changed here and registered by clicking the Update button. To view and/or modify the complete set of parameters, click on the map number in the left most column.

For each remote OID to be read, enter the full SNMP OID and location (device). The names in the device list are defined in the Devices page.

The register name is optional and used only for display purposes.

Entering zero (none) for local register effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type.

The screenshot shows the web interface for the Babel Buster 2 Modbus Network Gateway. The header includes the product name and logo for Control Solutions, Inc. The navigation menu is divided into several sections: RTU Serial Port, IP Network (selected), System, Diagnostic Data, Modbus Setup, SNMP Setup, SNMP Client (selected), and SNMP Diagnostics. Below the menu, there are sub-sections for Devices, Client Read Map (selected), and Client Write Map. The main content area contains a description: "This page creates a map entry that reads data from a remote SNMP agent for processing here." Below this is a form for configuring a Client Read Map. The "Map #" field is set to 1. The "Read OID" field contains "1.3.6.1.4.1.3815.1.2.2.1.1.1.1.2.1" and the "from" dropdown is set to "TestSystem". The "Then apply scale" field is 0.000000 and the "offset" field is 0.000000. The "Save in local register" field is 10, named "TestSNMP reg 1", and the "Repeat this process every" field is 10.0 seconds. The "Apply this default value" field is 0.000000 after 0 read failure(s). At the bottom, the "# Client Read Maps Enabled" field is 3, with "Insert" and "Delete" buttons.

Rule number simply tells you where you're at on the list of OID maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Map #" box, then click Update.

For each remote OID to be read, enter the full OID and location (device). The names in the device list are defined in the Devices page.

When the remote OID is read, data may be manipulated before being written to the local register. The result will be multiplied by the scale factor if any non-zero scale factor is given. The offset is then added and this final result is written to the local register number given. The name is optional and used only for display purposes.

The periodic poll time determines how often the remote OID will be read. This number, if nonzero, will override the default poll time given in the Devices page for the remote device being read.

The default value will be stored into the local register after the given number of read failures if the fail count is non-zero. Setting the count to zero will disable the default, and the register will retain the most recent value obtained.

Delete will remove the rule number shown in the "Map #" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having zero for a source register or

"none" for remote type.

Entering zero (for none) for local register effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type. If you wish to prevent these from being displayed, reduce the number of rules enabled.

The number of rules enabled simply limits the scope of rule review so that you do not have to review a lot of unused rules. If the displayed rules are used up and you need more, increase the enabled number.

8.3 SNMP Client Write Maps

The list of OIDs in the remote SNMP devices that are to be written is defined here. Data is taken from local registers in the gateway and written to the remote SNMP device.

Write local registers out to remote SNMP OIDs. This page creates a map entry that writes data remote SNMP agents from data contained here. Click on map number to see more detail and insert/delete rules.

Showing 1 to 2 of 2

Map #	Local Register #	Remote SNMP OID	Remote Data Type	Remote Device	Local Register Name
1	15	1.3.6.1.4.1.3815.1.2.2.1.1.1.2.5	Integer	Test System	Test Write 15
2	0		Undefined	None	

Rule number simply tells you where you're at on the list of OID maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Showing" box, then click Update.

Rules entered on this page only write data to remote devices. Go to the Client Read Map to read data from those devices. The full parameter set is different for read versus write.

An abbreviated version of a list of rules is shown on this page. Any of the parameters shown may be changed here and registered by clicking the Update button. To view and/or modify the complete set of parameters, click on the map number in the left most column.

Important note about data type: SNMP does not have a universally accepted representation for floating point. The most commonly used means of representing real data is scaled integers, and this method is supported by the DM2-MOD. IEEE 754 is not recognized as an SNMP standard and is not used. X.690 defines an encoding for real data, but it is inefficient and little used. A common recommendation is to use ASCII string representation of floating point data, and this method is supported by DM2-MOD (Octet String Num). Another known but application specific implementation is the ASN OPAQUE FLOAT used in netsnmp applications. This method is also supported by DM2-MOD but should be tested to confirm compatibility.

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until

deleted. Unused rules at the end of the list will always show none as the type.

The screenshot shows the web interface for the Babel Buster 2 Modbus Network Gateway. The page title is "Babel Buster 2 MODBUS NETWORK GATEWAY MODEL BB2-6010" and the company is "CONTROL SOLUTIONS, INC. MINNESOTA". The navigation menu includes: RTU Serial Port, IP Network (selected), System, Diagnostic Data, Modbus Setup, SNMP Setup, SNMP Client (selected), and SNMP Diagnostics. Below the menu are sub-tabs: Devices, Client Read Map, and Client Write Map (selected). The main content area is titled "This page creates a map entry that writes data to remote SNMP agents from data contained here." and includes a "Map #" field with the value "1", "Update", "< Prev", and "Next >" buttons. The configuration details for the selected map are:

- Read local register # 15 named Test Write 15
- Apply default value of 0.000000 at power-up and/or when 0.0 seconds have elapsed with no host update.
- Write remote OID any time local register has changed by 0.000000 or when 0.0 seconds have elapsed with no change.
- Otherwise write remote OID unconditionally. In any event, when writing remote OID, apply local register data as follows:
 - Apply scale: 0.000000 and offset: 0.000000
 - Write OID 1.3.6.1.4.1.3815.1.2.2.1.1.1.1.2.5 as Integer at Test System
 - Repeat this process at least no more than every 10.0 seconds.
- # Client Write Maps Enabled: 2

 At the bottom right are "Insert" and "Delete" buttons.

Rule number simply tells you where you're at on the list of OID maps. Click "next" and "prev" to scroll through the list. To advance directly to a specific map, enter the desired number in the "Map #" box, then click Update.

The local register data may be written periodically, or when it changes, or both. To send upon change (send on delta), check the first box and enter the amount by which the local register must change before being written to the remote device. To guarantee that the remote OID will be written at least occasionally even if the data does not change, check the second box and enter some amount of time. This time period will be referred to as the "maximum quiet time".

Data from the local register may be manipulated before being written to the remote OID. The local data is first multiplied by the scale factor. The offset is then added to it. The data is then sent to the remote SNMP agent. Enter the full OID to be written, the SNMP ASN data type to be written (select from list), and the location (device). The names in the device list are defined in the Devices page.

Important note about data type: SNMP does not have a universally accepted representation for floating point. The most commonly used means of representing real data is scaled integers, and this method is supported by DM2-MOD. IEEE 754 is not recognized as an SNMP standard and is not used. X.690 defines an encoding for real data, but it is inefficient and little used. A common recommendation is to use ASCII string representation of floating point data, and this method is supported by DM2-MOD (Octet String Num). Another known but application specific implementation is the ASN OPAQUE FLOAT used in netsnmp applications. This method is also supported by DM2-MOD but should be tested to confirm compatibility.

The repeat time may determine how often the remote OID will be written. If send on delta and maximum quiet time are not checked above, clicking the "at least" button will establish a periodic update time. If send on delta is used and you wish to limit the network traffic in the event changes are frequent, click the "no more than" button and enter the minimum time that should elapse before another write to the remote device.

Delete will remove the rule number shown in the "Map #" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having zero/none for a source register.

Selecting "none" for remote type effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show none as the type. If you wish to prevent these from being displayed, reduce the number of rules enabled.

The number of rules enabled simply limits the scope of rule review so that you do not have to review a lot of unused rules. If the displayed rules are used up and you need more, increase the enabled number.

8.4 SNMP Errors

Errors encountered by the SNMP client are listed here. There are two error lists, one for the read maps and one for the write maps.

Babel Buster 2
MODBUS
NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | **IP Network** | System

Diagnostic Data | Modbus Setup | **SNMP Setup** | SNMP Client | SNMP Diagnostics

Errors: Read Maps | Errors: Write Maps

This page displays errors, if any, reported by the remote server upon attempts to write to that device.

<< Top | Next >

Map #	Remote OID	Remote Device	Local Name	Error Code
1	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.2.5	Test System	Test Write 15	9

Reset Errors

Errors for SNMP client read maps are shown on this page. Only those maps with errors to report are listed. Refer to the code and class lists below for interpretation.

Common error codes for the SNMP client are as follows:

- 9 = No response from remote Agent (server)
- 10 = Unable to interpret data
- 11 = Reply does not match request

Other error codes are possible but improbable. Codes in the 80-120 range indicate socket errors; however, because SNMP uses UDP/IP, which is "connectionless", socket errors would indicate something internal is

seriously broken.



9. Configuring Action Rules

Action rules provide a number of "actions" as described below. For SNMP applications, one of the most important rules is the "Threshold Rule" as these become the triggers for sending SNMP traps.

9.1 Threshold Rules

Threshold rules are used to trigger actions based on events. The event occurs when a source register meets the criteria defined by the rule. The result of the event can be setting another register to a value as indicated in the rule template. The result of the event can also include triggering an SNMP trap.



RTU Serial Port IP Network **System**

Data Action Rules Setup

Thresholds Trending Cascade Calculate Constants

This page displays a summary of threshold rules currently in effect.

Showing 1 to 2 of 2

Rule #	Event Name	Source Register #	Test Criteria	Test Value	Result Register #	True Value	False Value
<u>1</u>	Test Reg High	00001	Greater than	100.0000	00000	0.000000	0.000000
<u>2</u>		00000	Unknown	0.000000	00000	0.000000	0.000000

Click on the rule number in the left column to see the entire threshold rule template, and to make changes to the rule.

Babel Buster 2
MODBUS
NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | IP Network | **System** | |

Data | **Action Rules** | Setup | |

Thresholds | Trending | Cascade | Calculate | Constants

This page displays thresholds, or rules, for defining events and assigning responses to events. Thresholds can create output based on conditional input.

Rule # Rule presently tests FALSE

Read local source register # for this event named

Event is TRUE if the value is this value: this local register:

Qualified by this hysteresis value: this minimum On Time: this minimum Off Time:

Set local destination register # as follows below while logging on-time to register #

(true) To a value which is same as the source this value: from local register #

(false) Otherwise to a value which is same as the source this value: from local register #

Rules Enabled:

Rule number tells you where you're at in the list of rules. Click "next" and "prev" to scroll through the list. To define a new rule, begin with a source register, and simply follow through the process entering criteria and selecting options as you go. The name is optional and used only for display purposes.

Select a comparison or test, and click the button for your choice of what the local register should be compared to. Then enter either the fixed value or local register number.

Qualifications are optional, and enabled only when values are nonzero. How hysteresis is applied depends on the comparison. For a test that becomes true if greater than, the test will not return to false until the local register is less than the test value by a margin of at least this hysteresis value. If a test becomes true if less than, it will not return to false until the local register is greater than the test value by a margin of at least this hysteresis value.

On time and off time, if specified, determine how long the condition must be true (on time) or false (off time) before the true or false response is actually taken.

One special test needs further explanation. The "Changed by" test is a rate of change test. If the source register has changed by the value specified as "this value" or the value contained in the local register referenced, the test is true. The minimum On Time on the following line applies. The test will be true if the source register has changed by the specified amount in this period of time. The On Time is interpreted in this special manner only for the "Changed by" test (otherwise see previous paragraph). Absolute value of the change found in the local register is tested, so the specified value (or delta) to test against should always be a positive number.

Now that you have specified what the condition is, you proceed to define the response. Start by selecting which local register the response is applied to. This will be known as your destination register. Typically this register will be linked to an output. **IMPORTANT** note for SNMP users: You do not need to apply the

result to any destination register if you simply want to report the event via a Trap. Leave the destination register set to zero and ignore the 2 lines that follow it. The result of the test will be processed as true or false by SNMP Trap processing without any destination register specified.

The first line after the destination register number is the response that will be taken when the condition is true, and the following line is the response that will be taken when the condition is false. Either the source register is copied, a fixed value is applied, or another register is used to provide the data written to the destination register.

Delete will remove the rule number shown in the "Showing" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having zero for a source register. Rule numbers work like row numbers in a spread sheet. If you insert a rule, existing rules slide down the sheet and get a new number. Likewise if you delete a rule, the rest of the rules slide up the sheet and get new rule numbers.

Entering zero as a source register effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show zero as the source register. If you wish to prevent these from being displayed, reduce the number of rules enabled.

The number of rules enabled simply limits the scope of rule review so that you do not have to review a lot of unused rules.

This page displays data as presently found in the local registers maintained by this device.

Showing 1 to 2 of 2

Rule #	Event Name	Local Register #	Register Data	Test Criteria	Test Value	Result *
1	Test Reg High	00001	118.0000	Greater than	100.0000	TRUE
2	Test Reg High	00000	0.000000	Unknown	0.000000	FALSE

To view the current status of a threshold rule, go to Thresholds under the Data tab.

9.2 Trending Rules

Trending allows tracking minimum, maximum, and average values over some period of time.

This page allows setup of min/max/average tracking of selected registers.

Showing 1 to 1 of 1 Update < Prev Next >

Tracked Register #	Register Name	Average Register	Min. Register	Max. Register	Reset	Period HH:MM:SS	Slices
1	Test Register	21	22	23	Manually	0:00:30	5

Trends Enabled: 1 Insert Delete

Fist integer register for saving results 1 First float register for saving results 1001 Auto Allocate

Select a register to be tracked, then select other unassigned registers as the destination for the results. Select when or how often the tracking should be reset.

You may automate the assignment of result registers. Select the first integer and first floating point register numbers. Then click Auto Allocate to fill in all result registers for all tracked registers.

The trend data is updated periodically at the period given. This period is subdivided into the number of slices given. A sliding window average of this number of slices is maintained, and this filtered value is processed as the trend data. Thus the minimum data value will be the lowest average of N slices over any given Period of time, and the maximum data value will be the highest average of N slices over any given Period of time. (Note: To get instant minimum and maximum to be tracked, set the slice count to 1 which results in no filtering.)

The trending may be reset automatically, either daily or hourly, or reset manually. To reset manually, go to the Trend Data page, check the reset box, and click the reset button.

Babel Buster 2
MODBUS NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | IP Network | **System** | | |

Data | Action Rules | Setup | | |

Local Registers | **Thresholds** | Trend Data | MIB View | |

This page displays data as presently found in the local registers maintained by this device.

Showing 1 to 1 of 1 Update < Prev Next >

Tracked Register #	Register Name	Average Value	Min. Value	Max. Value	Reset Now
1	Test Register	119.2501	118.0000	120.0000	<input type="checkbox"/>

Clear All

View the current trend values by going to the Trend Data page under Data.

9.3 Cascade Rules

Cascade rules allow effectively copying data from one register to another.

Babel Buster 2
MODBUS NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | IP Network | **System** | | |

Data | Action Rules | Setup | | |

Thresholds | Trending | **Cascade** | Calculate | Constants

This page allows configuring the cascading of registers. Data from the source register is copied to the destination register(s).

Showing 1 to 2 of 2 Update < Prev Next >

Rule #	Source Register #	Destination Register #	Last Destination Register # in Range
1	1	11	11
2	0	0	0

Rules Enabled: 2 Insert Delete

Enter the single destination register, or a range of destination registers, to copy the source register to. If no destination register is entered, only a single register will be copied to.

Delete will remove the rule number shown in the "Showing" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having zero for a source register.

Entering zero as a source register effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show zero as the source register. If you wish to prevent these from being displayed, reduce the number of rules enabled.

The number of rules enabled simply limits the scope of rule review so that you do not have to review a lot of unused rules.

A note about multiple cascade rules from a single source: Each register is allowed to propagate only once to avoid redundant transmission of data. Therefore, to cascade the same source to more than one destination that are not contiguous, cascade the registers in sequence. In other words, rather than cascade A to B and A to C, you should cascade A to B, then B to C.

9.4 Calculate Rules

Calculate rules allow performing basic math and logic functions using local registers as the operands, and another local register to store the result.

This page allows setting up simple calculations on register data.

Showing 1 to 2 of 2 Update < Prev Next >

Rule #	Perform Operation	Using Register #	And/Through	This Register #	Place Result in Register #
1	add	1	and	2	6
2	none	0	and	0	0

Rules Enabled: Insert Delete

Quick I

Select the applicable operation from the drop list. All but logical NOT operations require two operands. Enter register numbers as applicable.

Some operations, such as average, sum, and divide, are valid for ranges of multiple registers.

Delete will remove the rule number shown in the "Showing" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having "none" for an operation (unused rule).

Select the operation to be performed from the drop list. All but logical NOT operations require two operands. Enter register numbers as applicable.

Some operations are valid for ranges of registers. Select "and" or "thru" to select two registers or multiple registers in a range. Average, sum, and logic operations are valid for ranges of multiple registers.

Delete will remove the rule number shown in the "Showing" box. Insert will insert a new rule before the rule number shown, and is used for placing rules between existing rules. It is not necessary to use Insert to add rules to the bottom of the list or to define any rule presently having "none" for an operation (unused rule).

Selecting "none" as the operation effectively deletes the rule even though it will still appear in the list until deleted. Unused rules at the end of the list will always show "none" as the operation. If you wish to prevent these from being displayed, reduce the number of rules enabled.

The number of rules enabled simply limits the scope of rule review so that you do not have to review a lot of unused rules.

9.5 Constant Rules

Constant rules allow placing a fixed value into a local register one time at startup (or upon loading the XML configuration file).

Babel Buster 2
MODBUS
NETWORK GATEWAY
MODEL BB2-6010

CONTROL SOLUTIONS, INC.
MINNESOTA

RTU Serial Port | IP Network | **System** | |

Data | **Action Rules** | Setup | |

Thresholds | Trending | Cascade | Calculate | **Constants**

This page defines constants that will be written to the assigned local registers one time at startup.

Showing 1 to 2 of 2 Update < Prev Next >

Rule #	Value	Destination Register #
1	155.000000	7
2	0.000000	0

Rules Enabled: 2 Insert Delete

Simply enter the the value to be written and the local register number that it should be written to one time at startup, and click Update.



10. Trouble Shooting

The following discussion covers some of the most common problems encountered in configuring the gateway.

10.1 Modbus RTU Trouble Shooting

This discussion assumes you want the Babel Buster gateway to be the Modbus Master (most common use). Let's review the setup procedure for a single Modbus read map. We suggest starting with one register. Once you get that working, proceed to fill up the table.

If you are using the gateway as Modbus RTU slave, and are unable to reach the device, check the following: (a) Check your wiring; (b) Check baud rate and character format - they must be the same on both ends; (c) Check to see that the slave address set in the gateway matches what the master is using to try to connect to it. The most common problem will be failure to connect. If you are getting an exception error, then the register number you are asking for does not exist in the gateway.

For trouble shooting the gateway as Modbus RTU master: First, go to the Local Device page and make sure you have the baud rate set, and parity (if any) selected. If you do not know what baud rate your Modbus device is set to, consult that manufacturers documentation before proceeding.

Make sure the Master button is clicked. Start with a liberally slow timeout, like 0.5 second just to be rather certain you do not have timeout problems. (We have yet to see a piece of working equipment take longer than half a second to respond to a Modbus master.)

This page displays configuration parameters for the Modbus RTU serial port.

Baud Rate: 19200 Parity: None [Update]

I am the Master I am a Slave

Parameters for RTU Master:

Default Poll Rate: 2.000 Seconds

Timeout: 0.500 Seconds

Parameters for RTU Slave:

My Address or Unit #: 0

Double registers are swapped

Next, go to the RTU Read Map page (below) and click the "1" in the left column. This takes you to the expanded view of map #1.

Read remote registers into local registers. This page creates a map entry that reads data from one or more remote Modbus RTU serial devices for processing here. Click on map number to see more detail and insert/delete rules.

Showing 1 to 15 of 15 Update < Prev Next >

Map #	Remote Type	Remote Register Format	Remote Register #	Remote Unit #	Swapped	Scale	Local Register #	Name
1	none	Integer	0	0	<input type="checkbox"/>	0.00000	0	

To get started, you must enable some maps. Enter a number greater than zero in the # RTU Read Maps Enabled window at the bottom of the expanded view page (below) and click Update.

Next, select a register type, a register number, a unit # (aka slave ID or slave address), and a local register number to store the data in. If any of the red check marks shown below are "none" or zero, you will get no action even attempted. Make sure the Unit # (slave ID or slave address) matches whatever you have your Modbus device set to. If you are uncertain what address it is set to, you need to consult the manufacturer's documentation for that equipment before proceeding.

The following example shows the only non-zero entries required to successfully read holding register #1 from unit #1 and store the data in local register #1. Once these (or comparable) entries have been made, click the Update button.

This page creates a map entry that reads data from a remote Modbus RTU serial device for processing here.

Map # 1 Update < Prev Next >

Read **Holding Register** as **Integer** from register # **1** at Unit # **1** with doubles swapped

Apply bit mask if applicable: 0000 then apply scale: 0.00000 and offset: 0.00000

Save in local register # **1** named Repeat this process every **2.0** seconds.

Apply this default value: 0.00000 after **0** read failure(s).

RTU Read Maps Enable: **15** Insert Delete

At this point, you can go to the data page (below) and see if you have data showing up. If you get no data, there is a problem. The confirmation that you are probably getting no data is the "time since last update". In this example, we see 126 seconds have elapsed. We are attempting to update every 2 seconds, so obviously data retrieval is not happening.


This page displays data to and from registers in devices accessed via the Modbus RTU serial port.

Showing 1 to 10 of 10 Update < Prev Next >

Dir.	Reg. Type	Remote Reg. #	Register Name	Local Reg. #	Hex	Update	Register Data	Time since Last update
From	Holding Reg	00001		00001	<input type="checkbox"/>	<input type="checkbox"/>	0	126.720

If you are getting no data, check the Error Codes page (below). Here we see that the "No Responses" is about equal to the "Total Messages". This means we are not getting anything back from the Modbus slave. If you are certain all of the above setup is correct, the only conclusion you (or we) can come to at this point is that there is a wiring problem, or the slave is not responding or not configured correctly. Review wiring information (see Appendix A), and check the slave configuration.

This page displays error codes encountered in processing reads and writes via the Modbus RTU serial port.

Showing devices from 1  Update < Prev Next >

Unit #	Reset -->	Read Error	Offending Read Map #	Reset -->	Write Error	Offending Write Map #	Reset -->	Total Messages	No Responses	CRC Errors	Exceptions
1	<input type="checkbox"/>	5/0	1	<input type="checkbox"/>	5/0	1	<input type="checkbox"/>	70	69	0	0

The other fairly common problem is to see some number of Exceptions, and over time, this number increases. This means you have configured the gateway to ask the slave for a register that does not exist in the slave. Be sure you did not enter 40001 when you are looking for holding register 1. The gateway configuration does not use Modicon notation except on the Server Map page (and only then if you explicitly enable that option).

10.2 Modbus TCP Trouble Shooting

This discussion assumes you want the Babel Buster SP (SPX) to be the Modbus TCP Master. Let's review the setup procedure for a single Modbus read map. We suggest starting with one register. Once you get that working, proceed to fill up the table.

If you are using the gateway as Modbus TCP slave, and are unable to reach the device, check to see that the IP address of the gateway is reachable from the master. The most common problem will be failure to connect. If you are getting an exception error, then the register number you are asking for does not exist in the gateway.

For troubleshooting the gateway as TCP master: First, go to the IP Network :: Modbus Setup :: Devices page and make sure you have the IP address and port number set. Unless instructed otherwise, use port 502.

This page sets up the network address and optional device parameters for a remote Modbus/TCP device that will be linked to for remote input and/or output (via the client read and client write maps). The local device acts as a Modbus master to the remote devices listed below.

Device #

IP Address: Port: (default 502)

Domain Name: Local Name:

Unit (optional):

Swap Double Registers

Default Poll Period: Seconds

Connection Status:

Next, go to the Click Read Map page (below) and click the "1" in the left column. This takes you to the expanded view of map #1.

Read remote registers into local registers. This page creates a map entry that reads data from one or more remote Modbus/TCP servers for processing here. Click on map number to see more detail and insert/delete rules.

Showing to 1 of 1

Map #	Remote Type	Remote Register Format	Remote Register #	Remote Device	Scale	Local Register #	Name
1	None	Integer	0	None	0.00000	0	

To get started, you must enable some maps. Enter a number greater than zero in the # Client Read Maps Enabled window at the bottom of the expanded view page (below) and click Update.

Next, select a register type, a register number, a device (from list created above), and a local register number to store the data in. If any of the red check marks shown below are "none" or zero, you will get no action even attempted. Make sure the IP address matches whatever you have your Modbus device set to. If you are uncertain what IP address it is set to, you need to consult the manufacturer's documentation for that equipment before proceeding.

The following example shows the only non-zero entries required to successfully read holding register #1 from "My Modbus Device" and store the data in local register #1. Once these (or comparable) entries have been made, click the Update button.

This page creates a map entry that reads data from a remote Modbus/TCP server for processing here.

Map #

Read as from register # at

Apply bit mask if applicable: then apply scale: and offset:

Save in local register # named Repeat this process every seconds.

Apply this default value: after read failure(s).

Client Read Maps Enabled:

At this point, you can go to the data page (below) and see if you have data showing up. If you get no data, there is a problem.

This page displays data as presently found in the local registers maintained by this device.

Showing registers from

Local Register #	Register Name	Hex	Update	Register Data	Unsigned	Register Type	Default Data	Server Timeout (s)
00001	My Data Register	<input type="checkbox"/>	<input type="checkbox"/>	7438	<input type="checkbox"/>	Integer	0.000000	0.0

If you are getting no data, check the Error Codes page (below). Here we see that the "No Responses" is some number greater than total messages. Zero total messages means we never succeeded in making a TCP connection.

This page displays error codes encountered in processing Modbus Client reads and writes via the Modbus TCP connection(s).

Device	Reset -->	Read Error	Offending Read Map #	Reset -->	Write Error	Offending Write Map #	Reset -->	Total Messages	No Responses	Exceptions
1	<input type="checkbox"/>	5/0	1	<input type="checkbox"/>	0/0	0	<input type="checkbox"/>	0	3	0

The first occurrence of read and write errors are shown along with the map number that was being processed when the error occurred. Check the reset box and click update to clear it and possibly show the next error if there are more than one active error conditions.

A total count of all errors is also shown. This total is the sum of errors for all maps for this device. Check the reset box and click update to reset the counts. Click Update to view the most recent data values.

Error code indications of A/B indicate the following errors with the first number:

- 1 = Transaction ID out of sync
- 2 = Exception code returned by remote device
- 3 = Function code mismatch (bad packet)
- 4 = Insufficient data (bad packet)
- 5 = No response from remote device, timed out
- 6 = CRC error in received packet

When A is code 2 indicating an exception code was returned, B indicates the exception as follows:

- 1 = Illegal function code
- 2 = Illegal data address (the requested register does not exist in the device)
- 3 = Illegal data value

If you are getting "No Responses", check the Device page, and note the Connection Status.

The screenshot shows the 'Modbus Setup' configuration page. At the top, there are tabs for 'RTU Serial Port', 'IP Network' (selected), 'System', and 'SNMP Setup'. Below these are sub-tabs for 'Diagnostic Data', 'Modbus Setup', and 'SNMP Setup'. The 'Modbus Setup' sub-tab is active, showing fields for 'Device # 1', 'IP Address 192.168.1.141', 'Port: 502 (default 502)', 'Domain Name', 'Local Name: My Modbus Device', 'Unit (optional) 1', 'Swap Double Registers' (unchecked), and 'Default Poll Period 2.0 Seconds'. A 'Connection Status' field is circled in red, showing the value '5'. Navigation buttons 'Update', '< Prev', and 'Next >' are visible.

Connection status will show a non-zero error code if there is a socket error. The most common errors include:

- 5 = Connection failed, unable to bind (usually means remote device not connected or not reachable)
- 81 = Connection in progress (means unsuccessful connect attempt, still trying)
- 95 = Network is unreachable
- 97 = Connection aborted
- 98 = Connection reset by peer
- 103 = Connection timed out
- 104 = Connection refused
- 107 = Host is unreachable

10.3 SNMP Trouble Shooting

Testing the gateway as an SNMP agent is discussed in section 7.5. Configuring the gateway as SNMP client (manager) is discussed in section 8. An overview of what to look for in getting the SNMP client working follows.

First, go to the SNMP Client :: Devices page and make sure you have the IP address and SNMP community set. The gateway will make either v1 or v2 requests. Select the version that matches the equipment you are attempting to query.

This page sets up the network address for a remote SNMP device that will be linked to local objects via the client read and client write maps. The local device acts as an SNMP client (manager) to the remote agents listed below.

Device # 1 Update < Prev Next >

IP Address: 192.168.1.142 Local Name: Test System

SNMP Version: v1 v2c

SNMP Community: private

Default Poll Period: 10.0 Seconds Device Status: 0 Reset

The OIDs of the data points you want the gateway to Get are entered on the SNMP Client Read Map. The required parameters are available on the summary page. Click on Map number in the first column to access the expanded view of the read rule.

Read remote SNMP OIDs into local registers. This page creates a map entry that reads data from remote SNMP agents for processing here. Click on map number to see more detail and insert/delete rules.

Showing 1 to 3 of 3 Update < Prev Next >

Map #	Remote SNMP OID	Remote Device	Local Register #	Local Register Name
1	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.2.1	Test System	10	Test SNMP reg 1
2	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.2.2	Test System	11	Test SNMP reg 2
3		None	0	

The OID, device (which identifies IP address), and a local register are the only things required for reading an OID from another device.

This page creates a map entry that reads data from a remote SNMP agent for processing here.

Map # 1 Update < Prev Next >

Read OID: 1.3.6.1.4.1.3815.1.2.2.1.1.1.1.2.1 from Test System

Then apply scale: 0.000000 and offset: 0.000000

Save in local register # 10 named Test SNMP reg 1 Repeat this process every 10.0 seconds.

Apply this default value: 0.000000 after 0 read failure(s).

Client Read Maps Enabled: 3 Insert Delete

If there are errors in the Get process for Read Maps, or Set process for Write Maps, they will be listed on the respective Errors pages.

This page displays errors, if any, reported by the remote server upon attempts to write to that device.

Map #	Remote OID	Remote Device	Local Name	Error Code
1	1.3.6.1.4.1.3815.1.2.2.1.1.1.1.2.5	Test System	Test Write 15	9

Common error codes for the SNMP client are as follows:

- 9 = No response from remote Agent (server)
- 10 = Unable to interpret data
- 11 = Reply does not match request

Except for device timeout (no response), the most common error is code 10, which means the SNMP device did respond, but the gateway could not understand the data. The most common data type is integer, or some variation of integer. However, MIBs often contain character strings or octet strings which the gateway does not know how to translate into a number to place into a Modbus register. Character strings are not translated into Modbus registers by this gateway, and therefore if the OID contains a character string, you will not be able to access that from Modbus.

The other place to look for error information is on the Devices page. If device status is non-zero, this means there is a problem.

This page sets up the network address for a remote SNMP device that will be linked to local objects via the client read and client write maps. The local device acts as an SNMP client (manager) to the remote agents listed below.

Device # 1

IP Address: 192.168.1.142 Local Name: Test System

SNMP Version: v1 v2c

SNMP Community: private

Default Poll Period: 10.0 Seconds

Device Status: 4

Device status will be a number from 1 to 5 for the following application level error conditions:

- 1 = could not bind socket
- 2 = could not build SNMP packet
- 3 = problem with map, could not parse OID or bad type
- 4 = response timed out
- 5 = unable to parse data

In most cases, the code indicated on the Devices page will be redundant with those shown on the SNMP Client Errors pages.

10.4 Modbus Reference Information

Modbus Register Types

The types of registers referenced in Modbus devices include the following:

- Coil (Discrete Output)
- Discrete Input
- Input Register
- Holding Register

Whether a particular device includes all of these register types is up to the manufacturer. It is very common to find all I/O mapped to holding registers only. Coils are 1-bit registers, are used to control discrete outputs, and may be read or written. Discrete Inputs are 1-bit registers used as inputs, and may only be read. Input registers are 16-bit registers used for input, and may only be read. Holding registers are the most universal 16-bit register, may be read or written, and may be used for a variety of things including inputs, outputs, configuration data, or any requirement for "holding" data.

Modbus Function Codes

Modbus protocol defines several function codes for accessing Modbus registers. There are four different data blocks defined by Modbus, and the addresses or register numbers in each of those overlap. Therefore, a complete definition of where to find a piece of data requires both the address (or register number) and function code (or register type).

The function codes most commonly recognized by Modbus devices are indicated in the table below. This is only a subset of the codes available - several of the codes have special applications that most often do not apply.

Function Code	Register Type
1	Read Coil
2	Read Discrete Input
3	Read Holding Registers
4	Read Input Registers
5	Write Single Coil
6	Write Single Holding Register
15	Write Multiple Coils
16	Write Multiple Holding Registers

Modbus Exception (error) Codes

When a Modbus slave recognizes a packet, but determines that there is an error in the request, it will return an exception code reply instead of a data reply. The exception reply consists of the slave address or unit number, a copy of the function code with the high bit set, and an exception code. If the function code was 3, for example, the function code in the exception reply will be 0x83. The exception codes will be one of the following:

1	Illegal Function	The function code received in the query is not recognized by the slave or is not allowed by the slave.
2	Illegal Data Address	The data address (register number) received in the query is not an allowed address for the slave, i.e., the register does not exist. If multiple registers were requested, at least one was not permitted.

3	Illegal Data Value	The value contained in the query's data field is not acceptable to the slave.
4	Slave Device Failure	An unrecoverable error occurred (for BB2-3060 means corrupt packet was received).
6	Slave Device Busy	The slave is engaged in processing a long-duration command. The master should try again later.
10 (hex 0A)	Gateway Path Unavailable	Gateway could not establish communication with target device. In the case of BB2-3060, will most often mean there is no IP address configured.
11 (hex 0B)	Gateway Target Device Failed to Respond	Specialized use in conjunction with gateways, indicates no response was received from the target device. In the case of BB2-3060, means there was a TCP link failure, unable to connect to TCP target device.
17 (hex 11)	Gateway Target Device Failed to Respond	No response from slave, request timed out.

Modicon convention notation for Modbus registers

Modbus was originally developed by Gould-Modicon, which is presently Schneider Electric. The notation originally used by Modicon is still often used today, even though considered obsolete by present Modbus standards. The advantage in using the Modicon notation is that two pieces of information are included in a single number: (a) The register type; (b) The register number. A register number offset defines the type.

The types of registers referenced in Modbus devices, and supported by Babel Buster gateways, include the following:

- Coil (Discrete Output)
- Discrete Input
- Input Register
- Holding Register

Valid address ranges as originally defined for Modbus were 0 to 9999 for each of the above register types. Valid ranges allowed in the current specification are 0 to 65,535. The address range applies to each type of register, and one needs to look at the function code in the Modbus message packet to determine what register type is being referenced. The Modicon convention uses the first digit of a register reference to identify the register type.

Register types and reference ranges recognized by Babel Buster gateways are as follows:

0x = Coil = 00001-09999
 1x = Discrete Input = 10001-19999
 3x = Input Register = 30001-39999
 4x = Holding Register = 40001-49999

Translating references to addresses, reference 40001 selects the holding register at address 0000, most often referred to as holding register number 1. The reference 40001 will appear in documentation using Modicon notation, but Babel Buster gateways require specifying "holding register" and entering that register number as just "1".

If registers are 16-bits, how does one read Floating Point or 32-bit data?

Modbus protocol defines a holding register as 16 bits wide; however, there is a widely used defacto standard for reading and writing data wider than 16 bits. The most common are IEEE 754 floating point, and 32-bit integer. The convention may also be extended to double precision floating point and 64-bit integer data.

The wide data simply consists of two consecutive "registers" treated as a single wide register. Floating point in 32-bit IEEE 754 standard, and 32-bit integer data, are widely used. Although the convention of register pairs is widely recognized, agreement on whether the high order or low order register should

come first is not standardized. For this reason, many devices, including all Control Solutions gateways, support register "swapping". This means you simply check the "swapped" option (aka "High reg first" in some devices) if the other device treats wide data in the opposite order relative to Control Solutions default order.

Control Solutions Modbus products all default to placing the high order register first, or in the lower numbered register. This is known as "big endian", and is consistent with Modbus protocol which is by definition big endian.

What does notation like 40001:7 mean?

This is a commonly used notation for referencing individual bits in a register. This particular example, 40001:7, references (Modicon) register 40001, bit 7. Bits are generally numbered starting at bit 0, which is the least significant or right most bit in the field of 16 bits found in a Modbus register.

How do I read individual bits in a register?

Documentation tends to be slightly different for every Modbus device. But if your device packs multiple bits into a single holding register, the documentation will note up to 16 different items found at the same register number or address. The bits may be identified with "Bn" or "Dn" or just "bit n". Most of the time, the least significant bit will be called bit 0 and the most significant will be bit 15. It is possible you could find reference to bit 1 through bit 16, in which case just subtract one from the number to reference the table below.

You cannot read just one bit from a holding register. There is no way to do that - Modbus protocol simply does not provide that function. You must read all 16 bits, and then test the individual bit you are interested in for true or false (1 or 0). Babel Buster gateways provide an automatic way of doing that by including a "mask" in each register map or rule. Each time the register is read, the mask will be logically AND-ed with the data from the register, and the result will be right justified to yield a 1 or 0 based on whether the selected bit was 1 or 0. Babel Buster gateways provide optimization when successive read maps or rules are selecting different bits from the same register. The Modbus register will be read from the slave once, and the 16-bit data will be shared with successive maps or rules, with each map or rule selecting its bit of interest.

The bit mask shown in the expanded form of the Babel Buster RTU read map is a 4 digit hexadecimal (16 bit) value used to mask out one or more bits in a register. The selected bits will be right justified, so a single bit regardless of where positioned in the source register will be stored locally as 0 or 1. The hex bit mask values would be as follows:

```
B0/D0/bit 0 mask = 0001
B1/D1/bit 1 mask = 0002
B2/D2/bit 2 mask = 0004
B3/D3/bit 3 mask = 0008
B4/D4/bit 4 mask = 0010
B5/D5/bit 5 mask = 0020
B6/D6/bit 6 mask = 0040
B7/D7/bit 7 mask = 0080
B8/D8/bit 8 mask = 0100
B9/D9/bit 9 mask = 0200
B10/D10/bit 10 mask = 0400
B11/D11/bit 11 mask = 0800
B12/D12/bit 12 mask = 1000
B13/D13/bit 13 mask = 2000
B14/D14/bit 14 mask = 4000
B15/D15/bit 15 mask = 8000
```

Some Modbus devices also back two 8-bit values into a single 16-bit register. The two values will typically be documented as "high byte" and "low byte" or simply have "H" and "L" indicated. If you run into this scenario, the masking for bytes is as follows:

High byte mask = FF00
 Low byte mask = 00FF

When the mask value in a Babel Buster gateway is more than just one bit, the mask is still logically AND-ed with the data from the Modbus slave, and the entire resulting value is right justified to produce an integer value of less than the original bit width of the original register.

The following example will read bit 11 from holding register 22 at slave address 5, and place either 0 or 1 in local register 38.

This page creates a map entry that reads data from a remote Modbus RTU serial device for processing here.

Map # 1 Update < Prev Next >

Read Holding Register as Uint 16-bit from register # 22 at Unit # 5 with low register first

Apply bit mask if applicable: 0800 then apply scale: 0.000000 and offset: 0.000000

Save in local register # 38 named MotorRunning Repeat this process every 2.0 seconds.

Apply this default value: 0.000000 after 0 read failure(s).

RTU Read Maps Enabled: 1 Insert Delete

Deciphering Modbus Documentation

Documentation for Modbus is not well standardized. Actually there is a standard, but not well followed when it comes to documentation. You will have to do one or more of the following to decipher which register a manufacturer is really referring to:

- a) Look for the register description, such as holding register, coil, etc. If the documentation says #1, and tells you they are holding registers, then you have holding register #1. You also have user friendly documentation.
- b) Look at the numbers themselves. If you see the first register on the list having a number 40001, that really tells you register #1, and it is a holding register. This form of notation is often referred to as the old Modicon convention.
- c) Look for a definition of function codes to be used. If you see a register #1, along with notation telling you to use function codes 3 and 16, that also tells you it is holding register #1.

IMPORTANT: Register 1 is address 0. Read on...

d) Do the numbers in your documentation refer to the register number or address? Register #1 is address zero. If it is not clear whether your documentation refers to register or address, and you are not getting the expected result, try plus or minus one for register number. All Control Solutions products refer to register numbers in configuration software or web pages. However, some manufacturers document their devices showing address, not register numbers. When you have addresses, you must add one when entering that register into configuration software from Control Solutions.

Can I put 2 gateways on the same Modbus network?

You can not have more than one Master on a Modbus RTU (RS-485) network. Therefore, if the gateway is to be configured as the Master, you can only have 1 gateway. You cannot use multiple gateways to read

more points from the same Modbus slave device.

Multiple gateways configured as slaves can reside on the same Modbus RS-485 network.

If you are using RS-232 devices, you can have only two devices total, regardless of how they are configured. RS-232 is not multi-drop.

How many devices can I have on a Modbus RTU network?

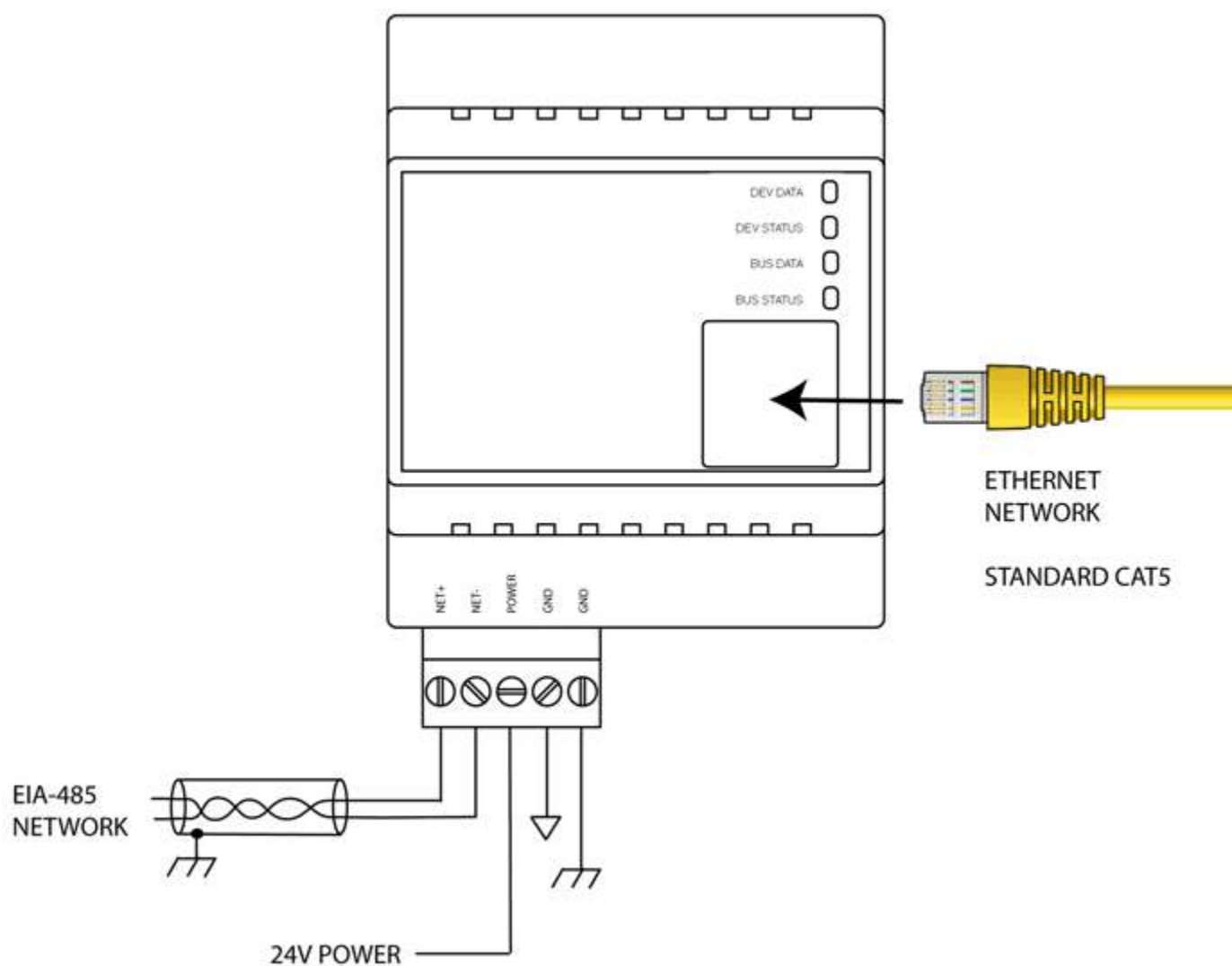
Logically you can address over 250 devices; however, the RS-485 transceivers are not capable of physically driving that many devices. Modbus protocol states that the limit is 32 devices, and most RS-485 transceivers will agree with this. Only if all devices on the network have low load transceivers can you have more than 32 devices.



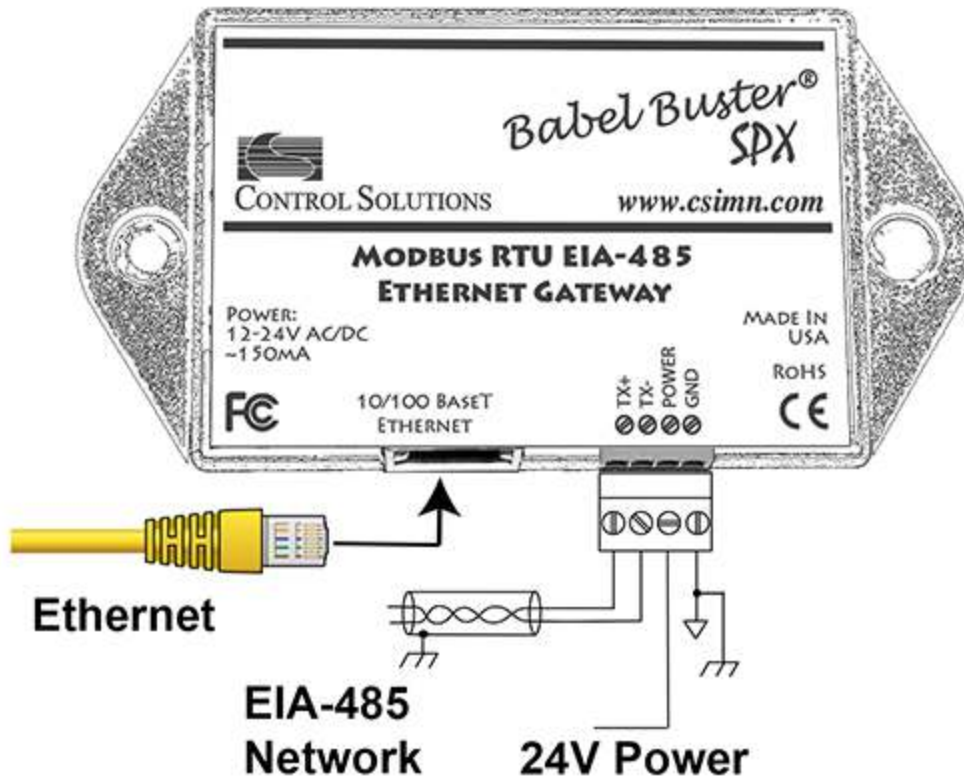
Appendix A Hardware Details

A.1 Wiring

Wiring for the Babel Buster BB2-6010 is illustrated below.



Wiring for the Babel Buster SPX is illustrated below (see section A.4 for SP).



Wire the gateway as illustrated. Follow all conventional standards for wiring of EIA-485 networks when connecting the Modbus RTU EIA-485 (RS485) network. This includes use and termination of shield, termination of the network, and grounding.

IMPORTANT: Although EIA-485 (RS485) is thought of as a 2-wire network, you **MUST** include a third conductor connected to GND or common at each device so that all devices are operating at close to the same ground potential. Proper grounding of equipment should ensure proper operation without the third conductor; however, proper grounding often cannot be relied upon. If large common mode voltages are present, you may even need to insert optically isolated repeaters between EIA-485 devices.

Use standard CAT5 cables for Ethernet connections. Use control wire as applicable for local electrical codes for connecting the 24V (AC or DC) power supply.

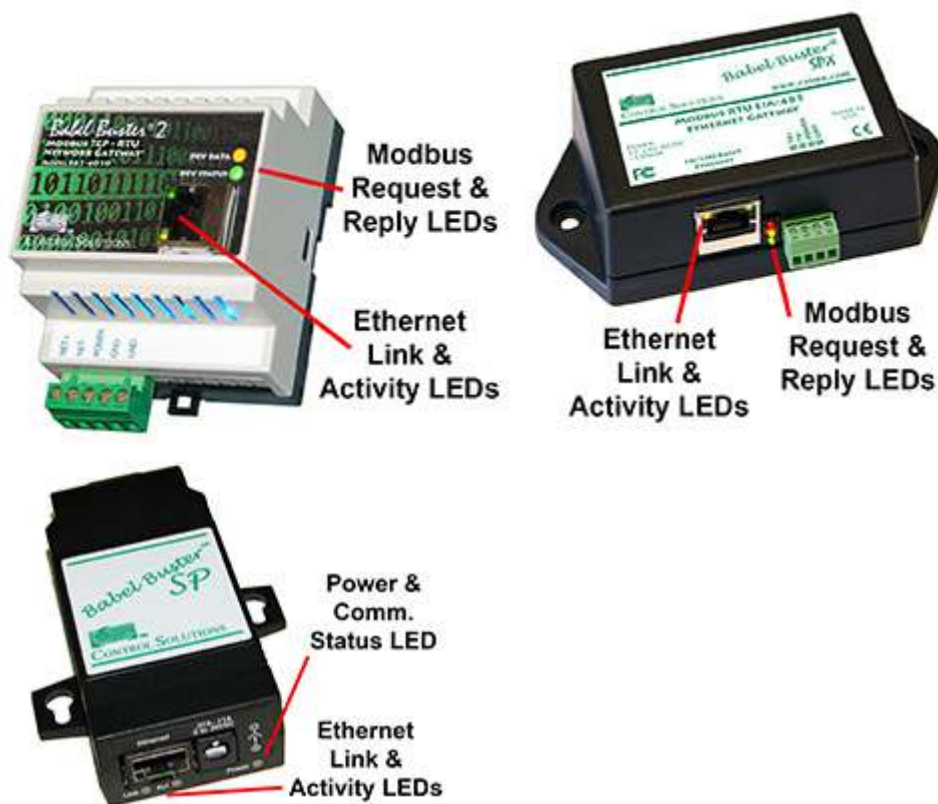
Note that in addition to connecting power supply common to a GND terminal, you must also connect a GND terminal to earth ground in order to ensure proper ESD protection.

A.2 Front Panel LED Indicators

Power-up LED behavior for BB2-6010: Following a server boot-up delay, all LEDs on front panel will turn on yellow or red for half a second, then all will turn on green for half a second. Then they will proceed to indicate as normally defined for the indicators.

Power-up LED behavior for SPX: Following a server boot-up delay, all LEDs will turn on briefly, then proceed to indicate as normally defined for the indicators.

Power-up LED behavior for SP: The single power/status LED will light up red and remain on until server boot-up.



Babel Buster BB2-6010 and SPX request/reply LEDs reflect RTU traffic while the Ethernet activity LED will indicate TCP traffic. To see TCP errors, one needs to look at the Errors page in the web UI.

Babel Buster SP: Since there is only one LED available, it simply indicates Modbus traffic. To see either TCP or RTU errors, one needs to look at the Errors pages in the web UI.

Babel Buster BB2-6010 LEDs indicate as follows (LEDs are bi-color):

DEV DATA	Flashes yellow each time a request is sent when operating as Modbus Master, or each time a request is received when operating as Modbus Slave.
DEV STATUS	<p>Operating as Modbus Master, flashes green each time a good response is received, or red when an error code is received, the request times out, or there is a flaw in the response such as CRC error.</p> <p>Operating as Modbus Slave, flashes green each time a good response is sent, or red if an exception code is sent (meaning the received request resulted in an error).</p>

Babel Buster SPX LEDs indicate as follows:

Yellow	Flashes each time a request is sent when operating as Modbus Master, or each time a request is received when operating as Modbus Slave.
Green	<p>Operating as Modbus Master, flashes each time a good response is received.</p> <p>Operating as Modbus Slave, flashes green each time a good response is sent.</p>
Red	Operating as Modbus Master, flashes when an error code is received, the request times out, or there is a flaw in the response such as CRC error.

Operating as Modbus Slave, flashes if an exception code is sent (meaning the received request resulted in an error).

Ethernet link LED is the yellow LED integrated into the CAT5 connector on BB2-6010 and SPX, and green on SP. Ethernet activity LED is the green LED integrated into the CAT5 connector on BB2-6010 and SPX, and yellow on SP (SP is reverse of other two).

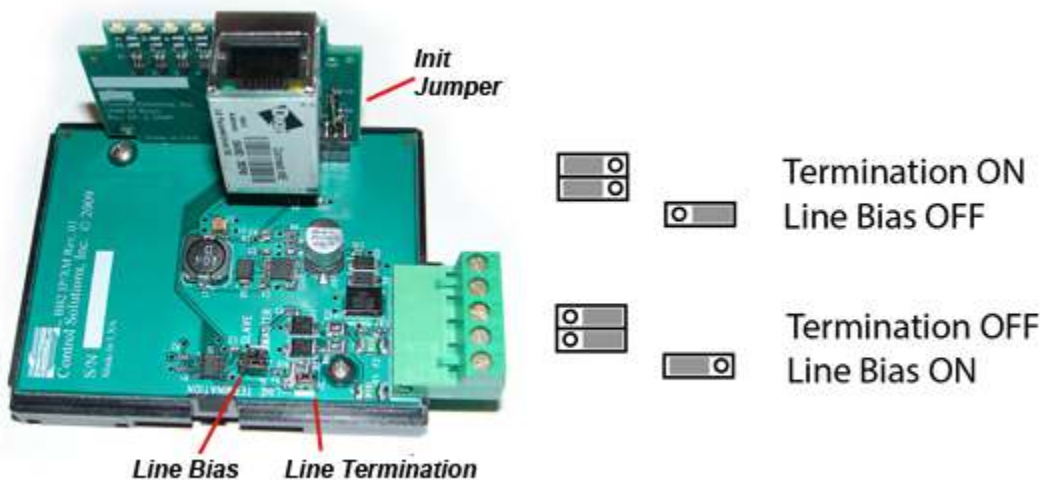
A.3 RS-485 Line Termination & Bias

Enable line termination only when this device is placed at the end of the network. Termination should only be enabled at two points on the network, and these two points must be specifically the end points.

Enable line bias when needed. Line bias should only be enabled at one point on the network, and does not have to be the end point. Line bias holds the line in a known neutral state when no devices are transmitting. Without bias, the transition from offline to online by a transmitter can look like a false start bit and cause loss of communication.

The line conditioning options are enabled when the respective shunt is moved to the position indicated by the white block next to the 3-pin header. Putting the shunt on the opposite 2 pins disables the option, and is simply a place to store the shunt.

Jumper locations for Babel Buster BB2-6010:



Jumper locations for Babel Buster SPX (see following section for SP):

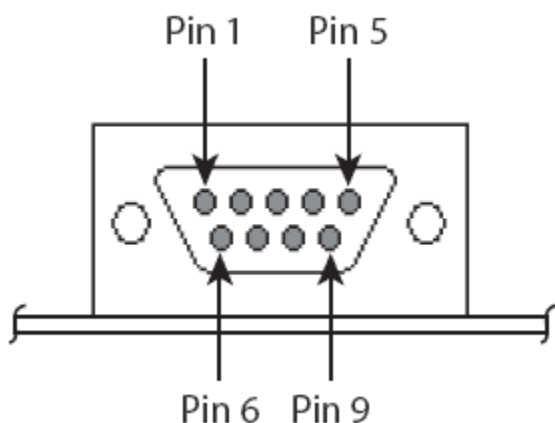


The "Init" jumper on the server module should only be used when advised by tech support. Installing this jumper prior to power-up causes the server to go into firmware update mode.

A.4 Babel Buster SP DB-9 Connector & DIP Switch

Pin connections are shown below. RS-485 (EIA-485) is by definition 2-wire. Any reference to 4-wire RS-485 is really EIA-422. Babel Buster SP supports EIA-422, but also automatically supports EIA-485 with a simple wiring configuration. Simply connect all the pluses together, and all the minuses together. Connect RXD+ to TXD+ on Babel Buster SP, and that one connection to + on your Modbus device. Do the same with the minus side. Then set the DIP switches for EIA-485 half duplex, termination ON. Although called "2-wire" you really need 3 conductors, 2 wires for signal + and - with a third conductor for ground/common.

If your Modbus device has its RS-485 terminal marked A and B, be aware that about half of the equipment out there has A and B backwards. This is due to the fact that A and B as published by EIA-485 are the reverse of A and B as defined by the RS-485 chip makers. Somewhere along the line they didn't compare notes. There is no harm done in reversing the wires, it simply will not communicate. If you are having trouble communicating, especially if the device terminals are marked A and B, try reversing them.



The above view is the male DB9 as seen looking into the Babel Buster SP. Important: This is **NOT** a PC COM port. Study the wiring table below carefully. Also keep in mind that while Pin 1 is on the left on the male connector, Pin 1 will be on the right looking at the female connector.

DB-9 Pin Assignments			
Pin	EIA-232	EIA-422/485 Full-Duplex	EIA-485 Half-Duplex
1	DCD	CTS-	Not used
2	RXD	RXD+	RXD+
3	TXD	TXD+	TXD+
4	DTR	RTS-	Not used
5	GND	GND	GND
6	DSR	RXD-	RXD-
7	RTS	RTS+	Not used
8	CTS	CTS+	Not used
9	NA	TXD-	TXD-

In addition to connecting wiring as indicated above, you need to change the DIP switch settings to select what type of port you are configuring.

