Remotely Detecting Offline Device (or Device Errors)

A common question is "how can I monitor device status?" or "how can I detect if the Modbus slave device is offline?" There are two approaches to answering this question. One option is to read a register containing an error code. The other is to cause the data to default to a known "bad" value, or flag value, when a device error is detected.

Approach #1: Error code register

A special set of registers begins at #9001. These are error code registers generated automatically by the system.

Error code registers are automatically named -Sys Error, Fn #n for system errors, -TCP error, device #n for TCP device errors, and -RTU error, device #n for Modbus RTU errors. In each instance of "device", the "n" is device number as referenced on other pages (either TCP device number from the device list, or RTU slave address).

Error encoding for devices is as follows: The error register will contain a code ABCCC where A=exception codes 1..3 (4), B=error code as follows, and CCC=rule number:

Error code B indicates the following errors:
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
9 = Socket error (CCC=error code, see below)

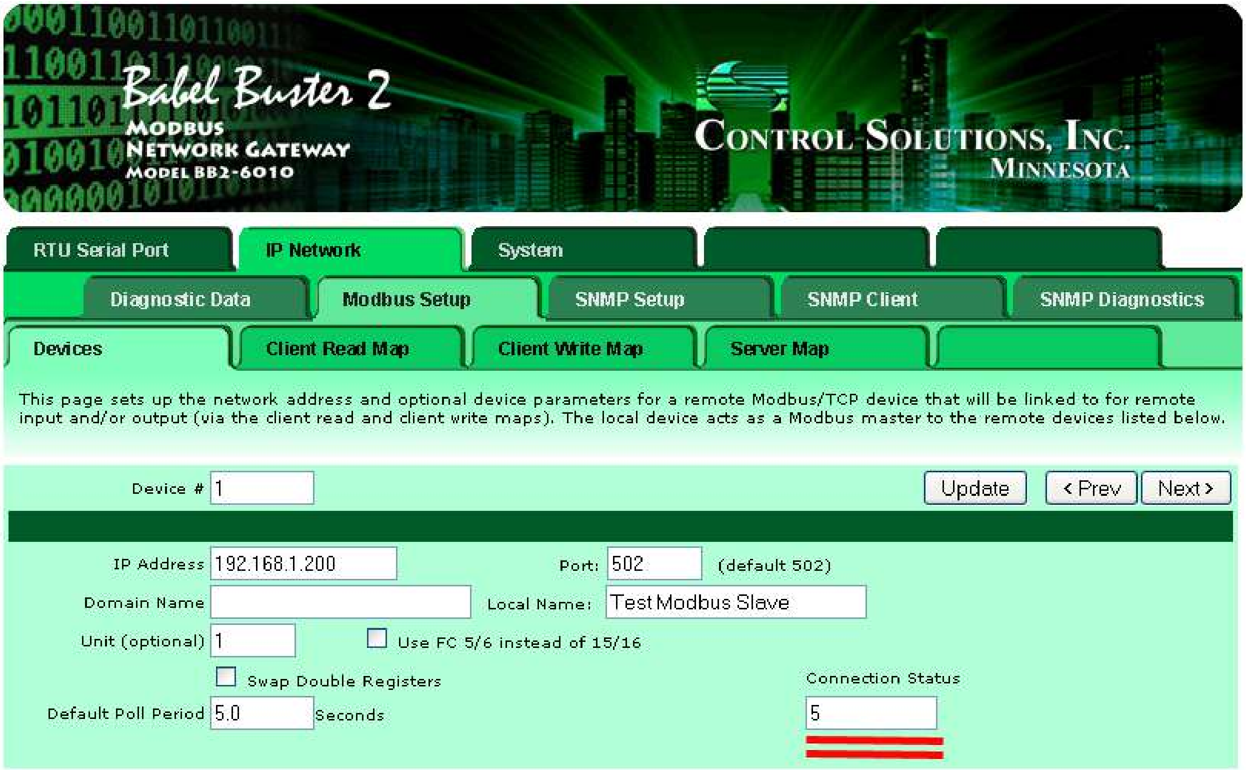
Error code A indicates the following exception codes only when B is code 2 indicating an exception code was returned:
1 = Illegal function code
2 = Illegal data address (the requested register does not exist in the device)
3 = Illegal data value
4 = other/out of range code

Some example error codes that are common: 5001 (05001) means no response from device at map rule #1. Example 2: 22005 means exception code 2 occurred on map rule #5. This generally means the register you attempted to access is not available at that remote device. Example 3: 6039 means a CRC error was received at map rule #39, and generally indicates a transmission error due to noise on the line.

Special case of socket error: Code will be >9000, and 9xxx will indicate socket error code xxx. Common socket codes include the following:

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

The following partial screen shots show the most common Modbus TCP example of device not connected, which results in device status code 5. These examples show the Babel Buster BB2-6010, but would be identical for Babel Buster SPX, and similar for other Modbus gateway products. The device page would appear as follows with the device is not connected:

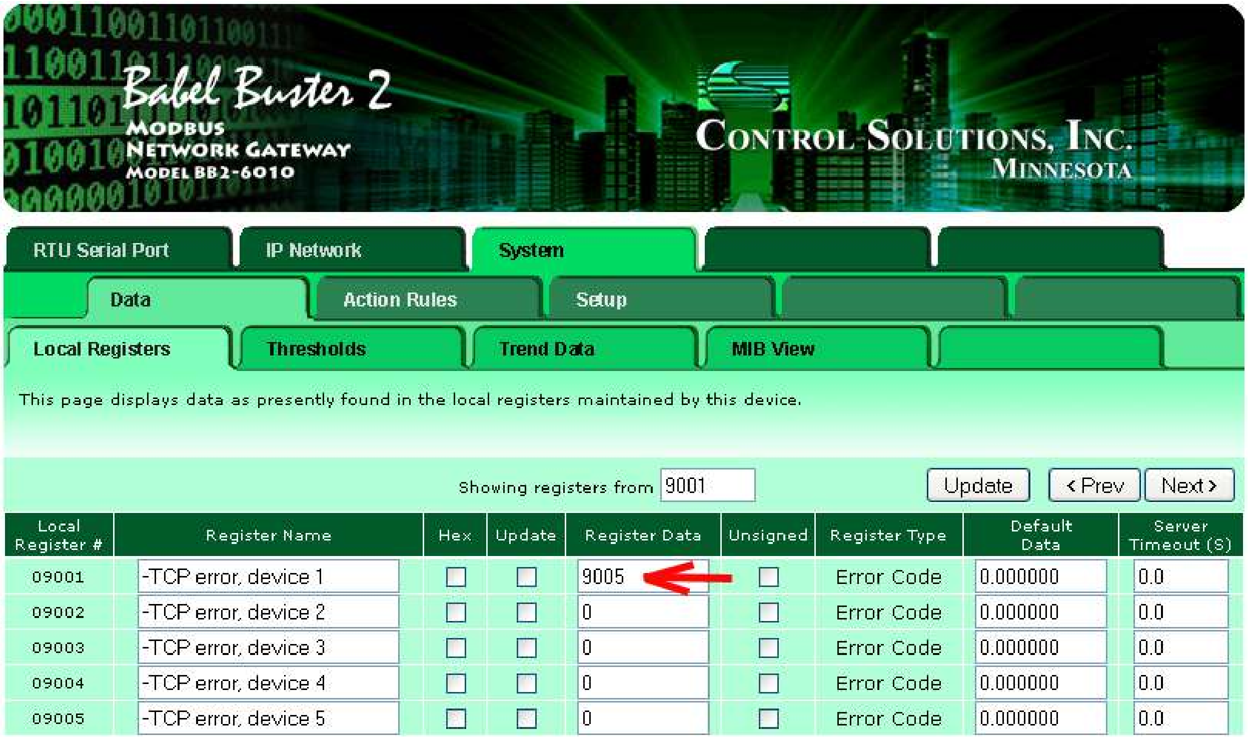


This page sets up the network address and optional device parameters for a remote Modbus/TCP device that will be linked to for remote input and/or output (via the client read and client write maps). The local device acts as a Modbus master to the remote devices listed below.

Device # 1

IP Address 192.168.1.200
Port: 502 (default 502)
Domain Name
Local Name: Test Modbus Slave
Unit (optional) 1
□ Use FC 5/6 instead of 15/16
□ Swap Double Registers
Connection Status 5
Default Poll Period 5.0 Seconds

The corresponding error code will appear in register 9001 for Modbus TCP device 1, as follows:

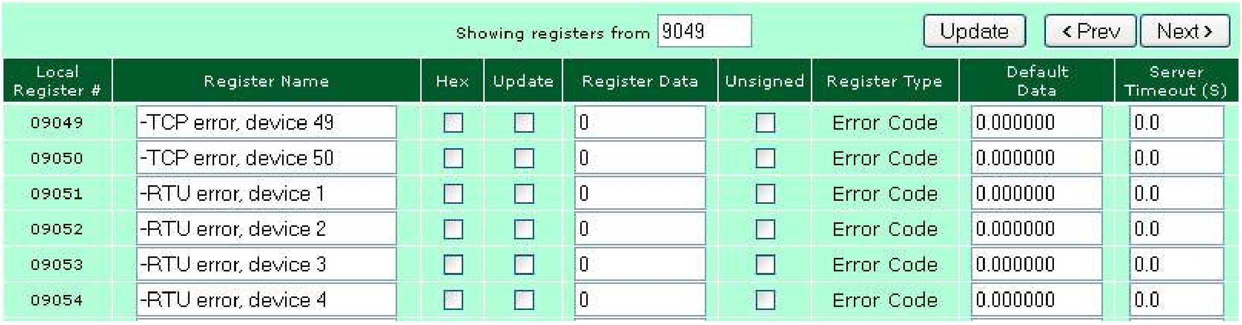


This page displays data as presently found in the local registers maintained by this device.

Showing registers from 9001

Local Register #	Register Name	Hex	Update	Register Data	Unsigned	Register Type	Default Data	Server Timeout (S)
09001	-TCP error, device 1	□	□	9005	□	Error Code	0.000000	0.0
09002	-TCP error, device 2	□	□	0	□	Error Code	0.000000	0.0
09003	-TCP error, device 3	□	□	0	□	Error Code	0.000000	0.0
09004	-TCP error, device 4	□	□	0	□	Error Code	0.000000	0.0
09005	-TCP error, device 5	□	□	0	□	Error Code	0.000000	0.0

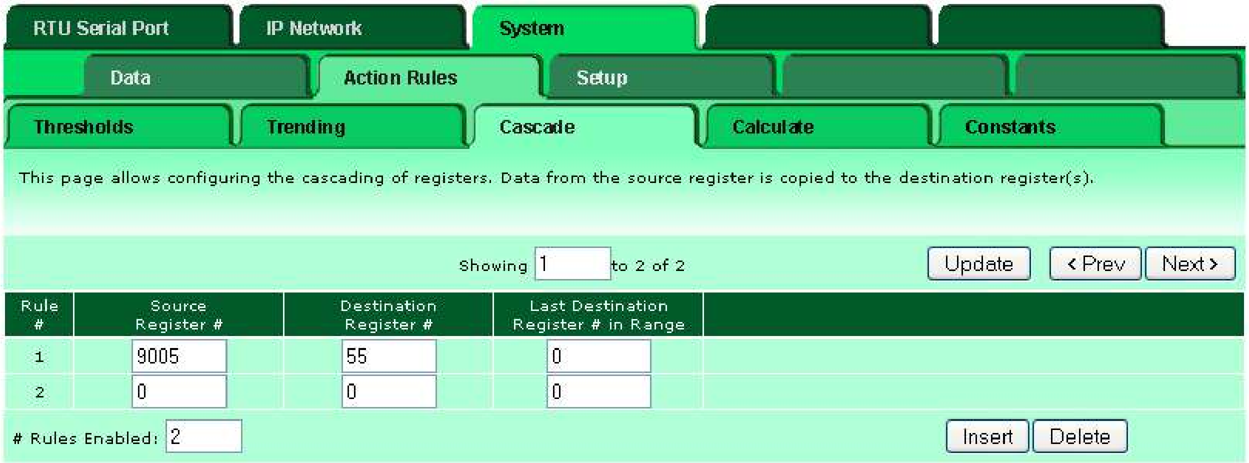
The RTU error registers immediately follow the TCP registers.



Showing registers from 9049

Local Register #	Register Name	Hex	Update	Register Data	Unsigned	Register Type	Default Data	Server Timeout (S)
09049	-TCP error, device 49	□	□	0	□	Error Code	0.000000	0.0
09050	-TCP error, device 50	□	□	0	□	Error Code	0.000000	0.0
09051	-RTU error, device 1	□	□	0	□	Error Code	0.000000	0.0
09052	-RTU error, device 2	□	□	0	□	Error Code	0.000000	0.0
09053	-RTU error, device 3	□	□	0	□	Error Code	0.000000	0.0
09054	-RTU error, device 4	□	□	0	□	Error Code	0.000000	0.0

Although you can view the 9000 series registers on the web page, you cannot access these via SNMP. To get the error codes out to SNMP, you first need to copy the value to a lower numbered register in the range of 1-999. Use a cascade rule as shown here to accomplish the copy function:



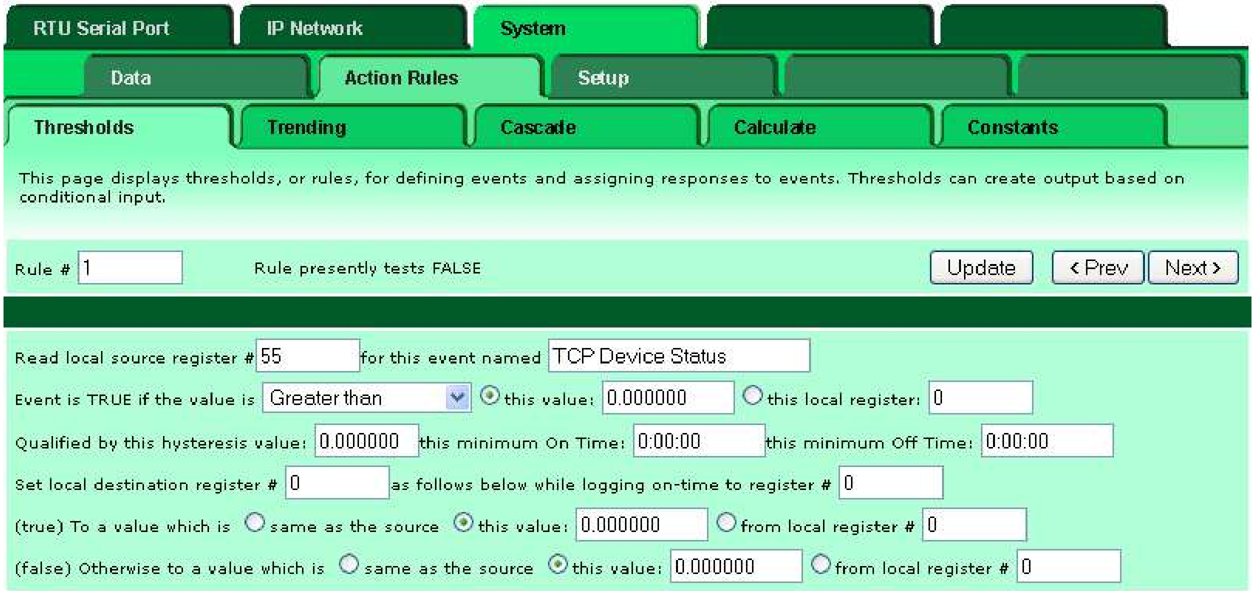
This page allows configuring the cascading of registers. Data from the source register is copied to the destination register(s).

Showing 1 to 2 of 2

Rule #	Source Register #	Destination Register #	Last Destination Register # in Range
1	9005	55	0
2	0	0	0

Rules Enabled: 2

At this point, you can use SNMP Get to retrieve the error code. You might (optionally) want to generate an SNMP trap upon device error. You can accomplish this by first creating a threshold rule that will test for any instance of a value that is not zero, as illustrated here:



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

Rule # 1 Rule presently tests FALSE

Read local source register # 55 for this event named TCP Device Status

Event is TRUE if the value is Greater than ● this value: 0.000000 ○ this local register: 0

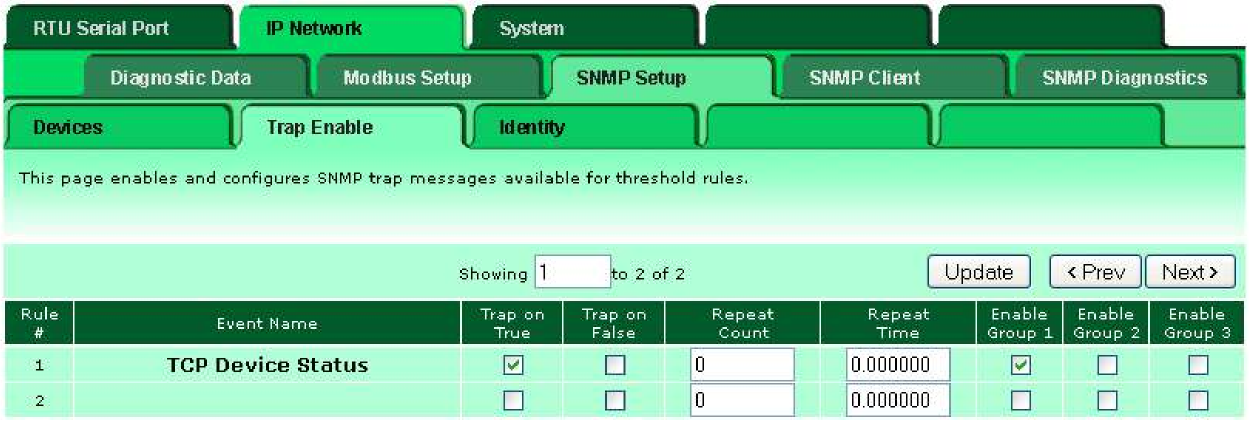
Qualified by this hysteresis value: 0.000000 this minimum On Time: 0:00:00 this minimum Off Time: 0:00:00

Set local destination register # 0 as follows below while logging on-time to register # 0

(true) To a value which is ○ same as the source ● this value: 0.000000 ○ from local register # 0

(false) Otherwise to a value which is ○ same as the source ● this value: 0.000000 ○ from local register # 0

Then go to the SNMP traps page and enable the trap for the rule just created.



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

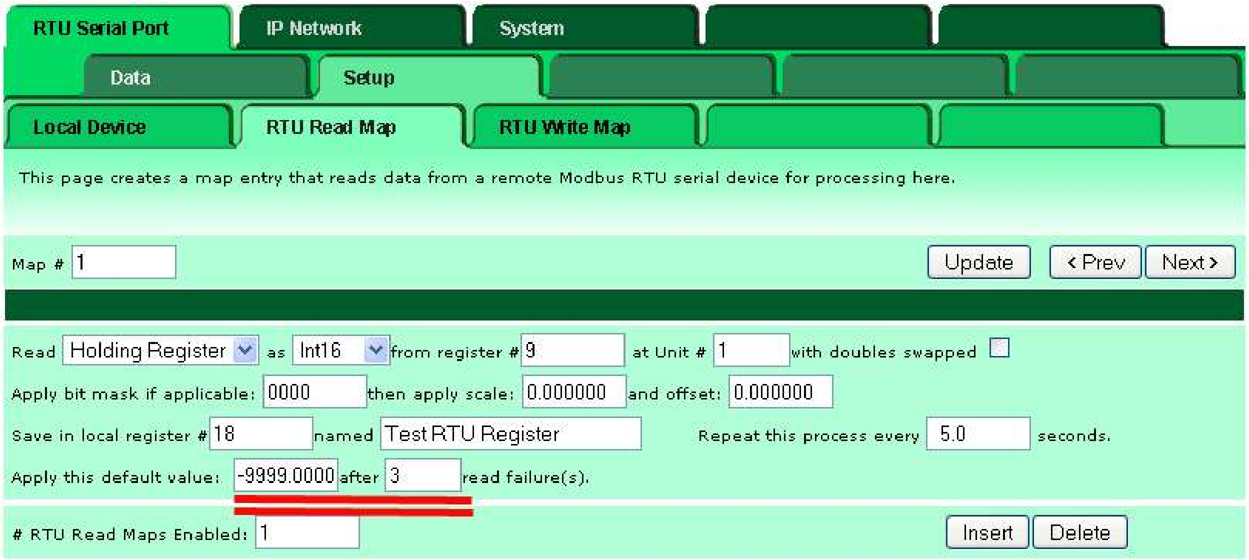
Showing 1 to 2 of 2

Rule #	Event Name	Trap on True	Trap on False	Repeat Count	Repeat Time	Enable Group 1	Enable Group 2	Enable Group 3
1	**TCP Device Status**	☑	□	0	0.000000	☑	□	□
2		□	□	0	0.000000	□	□	□

Approach #2: Default value set upon error

An example of a default value that could be used as a flag to a program processing the data would be to have a room temperature take on a value of -99 when the slave device is not responding. We can reasonably say that -99 cannot possibly be the room temperature, and therefore a fault apparently exists.

The default value is set in the read map (RTU or TCP). The local register will assume the default value after 'N' failures trying to read the remote Modbus slave. If the failure count is zero, the default is disabled. An example of the RTU read map with default enabled is shown here, and the TCP map will also allow setting of a default value:



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

Map # 1

Read Holding Register as Int16 from register # 9 at Unit # 1 with doubles swapped □

Apply bit mask if applicable: 0000 then apply scale: 0.000000 and offset: 0.000000

Save in local register # 18 named Test RTU Register Repeat this process every 5.0 seconds.

Apply this default value: -9999.0000 after 3 read failure(s).

RTU Read Maps Enabled: 1

Article ID: 12
Created On: Sat, Dec 22, 2012 at 12:08 AM
Last Updated On: Fri, Dec 12, 2014 at 11:31 PM