DXM Configuration Software v4 Product Manual

Original Instructions
p/n: 209933 Rev. J
18-Dec-24

Contents

Chapter Contents

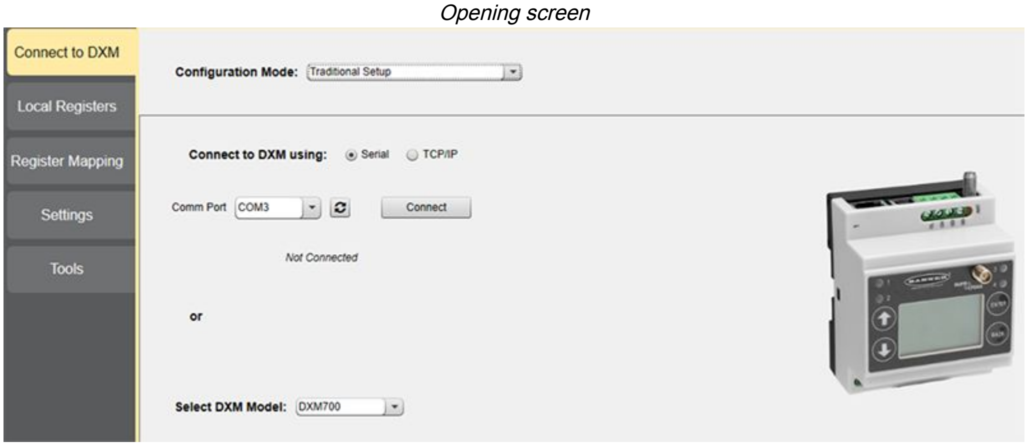
Chapter 1 Introduction and Quick Start Guide

These steps will guide you through the most common method of setting up the DXM Configuration Software and communicating with a connected DXM device. Version 4 of the configuration software supports multiple DXM device models, each including different features.

When a DXM model is connected to your computer, the software automatically detects the correct model and loads the appropriate screens. You may also manually select which model of DXM you are configuring if you intend to create a configuration file without connecting a device. This ensures that the interface and the configuration file use the correct features.

Not all screens are available for all models. To change to another model of DXM, go to the **Connect to DXM** screen and use the drop-down list to select another model. If the active configuration is incompatible with the selected model, you are prompted to either proceed and wipe out the active configuration or cancel the model change and preserve the loaded configuration.

Opening screen



Connect via USB or Ethernet. If connecting via Ethernet, set network parameters through the DXM LCD menu in the **System Cfg › Ethernet** menu. Network parameters can also be set within the configuration software. Setting parameters on the LCD menu overrides the parameters stored in the configuration file. To use the network parameters in the configuration file, reset the network parameters on the DXM LCD menu.

Since the DXM-R90x connects only via TCP, its **Connect to DXM** screen differs from the other DXM models. When the **Select DXM Model** drop-down is set to DXM-R90x, a new network discovery table is displayed. Click **Scan Network for DXMs** to detect DXM devices on the host computer's network. Discovered DXMs are listed in the network discovery table. Double-click any row entry to connect to that DXM. If the DXM's IP address is already known, the standard TCP connection option is available below the network discovery table.

Banner recommends disconnecting the COMM port through the **Device** menu before turning off the power or disconnecting the USB cable. Use **Device › Reboot** to restart the DXM if needed; the software automatically disconnects the COMM port and then reconnects it.

> **TIP:** If your connection attempts fail (the Application Status icon in the software screen's footer is red), close the configuration software and disconnect the USB cable from the computer. Reconnect the cable, launch the software, and attempt connecting again.

If you cannot connect to your DXM Controller, refer to for more information.

IMPORTANT: Any model of DXM may connect to the configuration software regardless of which device model is selected in the software. Compatibility is checked before configuration files are uploaded to the device.

DXM Configuration Software

Configure the DXM using the DXM Configuration Software. The configuration software can be used stand-alone or connected to the controller using USB or Ethernet. The software creates an XML file defined for the DXM and can be used at the website level for configuration.

The configuration software restricts the naming of registers and rules to characters a-z, A-Z, 0-9, # $ _ - ( ) space. A register name cannot end in a space. If an unacceptable name has been entered, a red ! displays.

Use the configuration software either while connected to a DXM or as a standalone configuration software.

The top-level menus are similar to other Windows programs: **File**, **Traffic**, **DXM**, and **Help**.

- Use the **File** menu to manage the loading and saving of the XML configuration file created by the configuration software.
- Use the **Traffic** menu to view data traffic on the serial bus or via UDP.
- Use the **DXM** menu to send and retrieve XML configuration files to or from a connected DXM.
- A new **Help** menu has also been added, which launches a copy of the configuration software instruction manual in the user's default PDF viewer.

For screens that contain tables with rows, click on any row to select it. Then click **Clone** or **Delete** to copy/paste or remove that row.

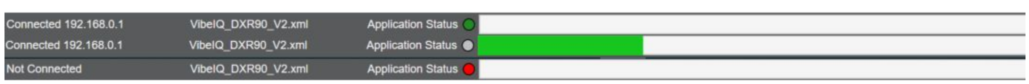
Save and Upload the Configuration File

After making any changes to the configuration, you must save the configuration files to your computer, then upload it to the device.

Changes to the XML file are not automatically saved. Save your configuration file before exiting the software and before sending the XML file to the device to avoid losing data. If you select **DXM › Send XML Configuration to DXM** before saving the configuration file, the software will prompt you to choose between saving the file or continuing without saving the file.

1. Save the XML configuration file to your hard drive by going to the **File › Save As** menu.
2. Go to the **DXM › Send XML Configuration to DXM** menu.

Status indicator bar



   - If the Application Status indicator is red, close and restart the DXM Configuration Software, unplug and re-plug in the cable and reconnect the DXM to the software.
   - If the Application Status indicator is green, the file upload is complete.
   - If the Application Status indicator is gray and the green status bar is in motion, the file transfer is in progress.

   After the file transfer is complete, the device reboots and begins running the new configuration.

Configuration File Cross Compatibility

Version 4 of the configuration software supports multiple DXM device models, each of which supports different features. As such, not all XML configuration files are cross-compatible.

For example, because the DXM700 supports a larger range of local registers than does the DXM100 or DXM150, the file in question may use registers that would be invalid for the DXM100 and DXM150 models. If this is the case, the configuration file is not converted and a warning message is displayed.

Version 4.1.7 adds support for the DXM100-A1[1] model. Version 4.8.4 adds support for the DXM1000 and DXM1200 models. Version 4.10.27 adds support for the DXM1500 and DXM-R90x1 models.

DXM100-A1 configuration files are always convertible into DXM100, DXM150, DXM700, DXM1000, DXM1200, and DXM1500 files. However, depending on which features are configured, the reverse is not always true. The following features are available to standard DXM100 models but are not supported by the DXM100-A1:

- Ethernet connectivity—Typically displayed on the **Settings › Ethernet** and **Settings › Cloud Services** screens
- Modbus TCP and CAN (**Register Mapping** screen)
- Server Port Settings (**Settings › System** screen)
- ScriptBasic Network Options and ScriptBasic RS-232 Settings (**Settings › Scripting** screen)

After selecting a DXM device model on application start-up, you can load any configuration file created for that device model or for any other model. Validation is performed to ensure that an incompatible file cannot be loaded into the configuration software or written to a connected device. Changing the device model selection on the **Model Select** screen converts the active configuration if it passes cross-compatibility validation.

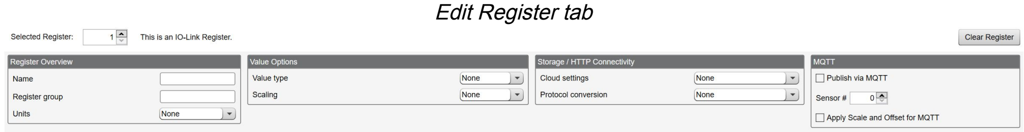
> **IMPORTANT:** Loading a cross-compatible file and saving it overwrites that file's original device model setting, reformatting it for the currently selected device model.

Configuration Instructions

Edit Registers Using the Local Registers in Use Screen

Use this screen to modify the parameters of any local registers being used.

Edit Register tab



1. Go to the **Local Registers › Local Registers in Use › Edit Register** section of the bottom of the screen.

   A list of the local registers being used displays.
2. Under the **Selected Register** box, select the register to define or modify. You may select the register by using the up/down arrows, directly entering a register number into the field, or clicking within the corresponding row in the **Local Registers in Use** table.

   Only Local Registers that have already been changed from their default configuration are displayed in the **Local Registers In Use** table.
3. Using the drop-down lists, assign a name, register group, change the units, or make other configuration changes to this register.
4. To push register values to the web server, set **Cloud Settings** to read.

   If **Cloud Settings** are set to Read, the web server only views data from the device and cannot write data to the device. If the permissions are set to Write, the web server only writes to the device and cannot read the data. If the permissions are set to Read/Write, the web server can read the data from the device and write to the device from the web.

The changes are automatically applied within the software, not the XML file. To change another register, use the up or down arrows to select another register number. To save these changes to the XML file, go to **File › Save**.

Modify Multiple Registers

Modify a range of registers from the **Local Registers › Local Registers in Use › Modify Multiple Registers** screen.

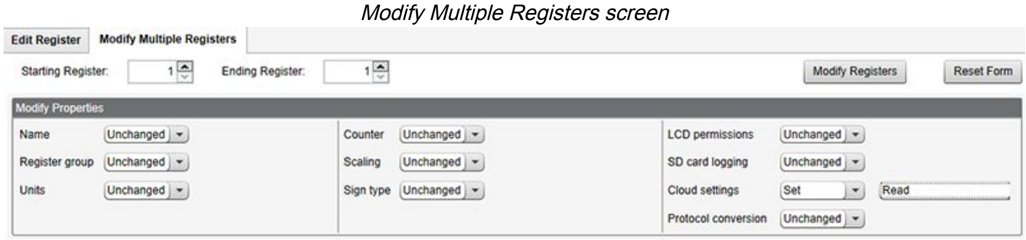
Select which parameter fields to modify. Most parameters have three selections.

- Unchanged—no changes
- Default—change to default settings
- Set—modify the parameter. Other selections will appear based on the parameter.

---

[1] Model DXM100-A1 was previously named DXM100-AG1 and may appear as -AG1 in some software and documentation. The -AG1 model is the same as the -A1 model.

Modify Multiple Registers screen



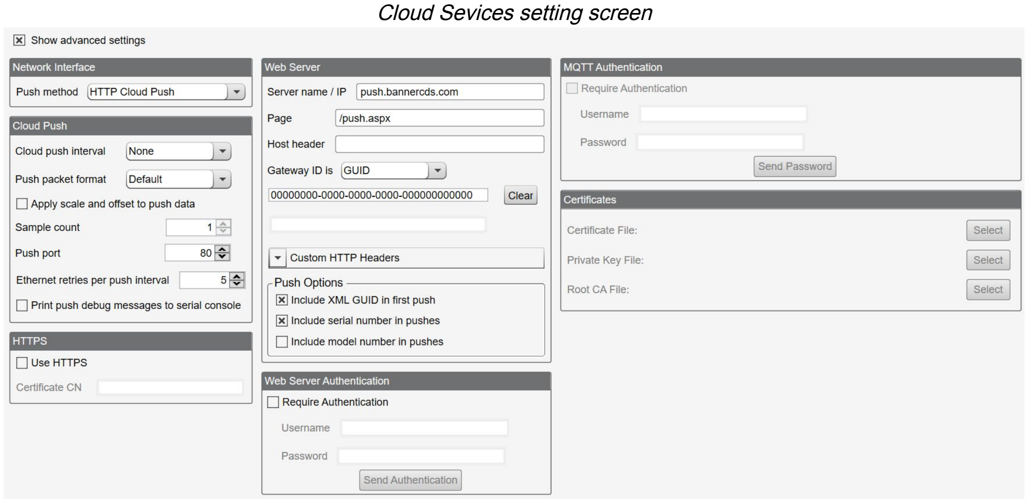
1. Enter the **Starting register** and **Ending register**.
2. Select the value to change using the drop-down list next to each value.
3. Enter the new value in the field provided.
4. To push register values to the web server, set **Cloud Settings** to **Read**.

   If the **Cloud Settings** are set to **Read**, the web server only views data from the device and cannot write data to the device. If the permissions are set to Write, the web server only writes to the device and cannot read the data. If the permissions are set to Read/Write, the web server can read the data from the device and write to the device from the web.
5. Click **Modify Registers** to save and apply the changes.

Configure the Cloud Data Services Settings

1. To configure the connection to the web server, go to the **Settings › Cloud Services** screen.

Cloud Sevices setting screen



2. Copy and paste the **Gateway ID**.

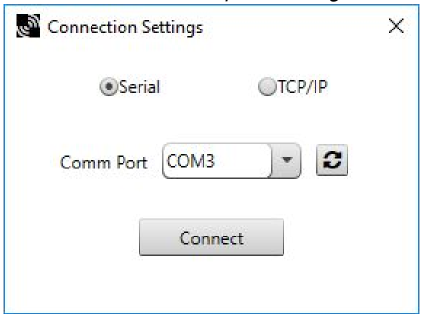
   The **Gateway ID** is the long string of numbers and letters from the Banner Cloud Data Services website.
3. Verify the **Server Name/IP** is set to `push.bannercds.com` and the **Page** is set to `/push.aspx` for sending to the website (verify Show Advanced Settings is selected).
4. Set the **Cloud Push Interval** to a value appropriate for your application.

   The **Cloud Push Interval** determines how often the device pushes the data to the web. The faster the push interval, the more data is sent to the site. Cellular plans can only push at an interval of 5 minutes or longer, while Ethernet connections can push at an interval of 1 minute or longer. The **Sample Count** specifies how many times the data is gathered within the **Cloud Push Interval**(Advanced Settings).

   For example, if the **Cloud Push Interval** is 15 minutes and the **Sample Count** is set to 3, then during each data push (every 15 minutes), 3 samples are sent to the web. This is one sample every 5 minutes.
5. Save the configuration file by going to **File › Save**.

   File names must be no more than 30 characters long, and should not contain any spaces or special characters.
6. With a USB cable connected to the device, go to the **Device › Connection Settings** menu.
7. Select the appropriate **Comm Port** and click **Connect**.

Communication ports settings



If multiple comm ports are visible, try each one until the software is able to connect to the device.

8. Go to **Device › Send Configuration to the Device** to upload the new XML file.

Creating Rules

Create a Read Rule Using Controllers with One Client Serial Port

Each read rule defines a Modbus server ID and register range to read and then store in the selected Local Registers. Use the **Register Mapping › Read Rules** screen to create a new read rule.

1. Click **Add Read Rule**.
   A new read rule is created.
2. Click the arrow next to the first rule to view the parameters.
   The list of parameters opens.
3. Type in the rule's name in the 'none' field.
4. Select the server ID of the source device.
5. Select the number of registers to read from the source device.
6. Select the starting register of the source device.
7. Select the starting register of the local/target device.
8. Set the desired parameters.

TIP: Use fewer read rules for applications that require higher response times by taking advantage of the Alternative Modbus Register groupings for DX80 Performance Nodes as shown in the DXM Wireless Controller Instruction manuals.

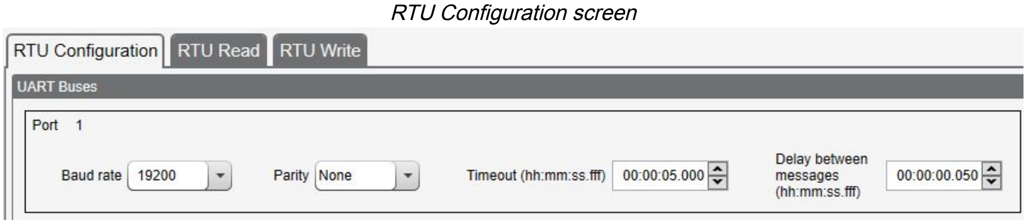
Create an RTU Read Rule Using Controllers with Multiple Client Serial Ports

Follow these steps to create a new read rule.

This example creates a read rule to read six registers (1 through 6), from Port 1 Modbus ID 4. The results are stored in the Local Registers 1 through 6.

1. Define the **Port** settings to be compatible with the connected devices.
   a. Go to the **Register Mapping › RTU › RTU Configuration** screen.

RTU Configuration screen

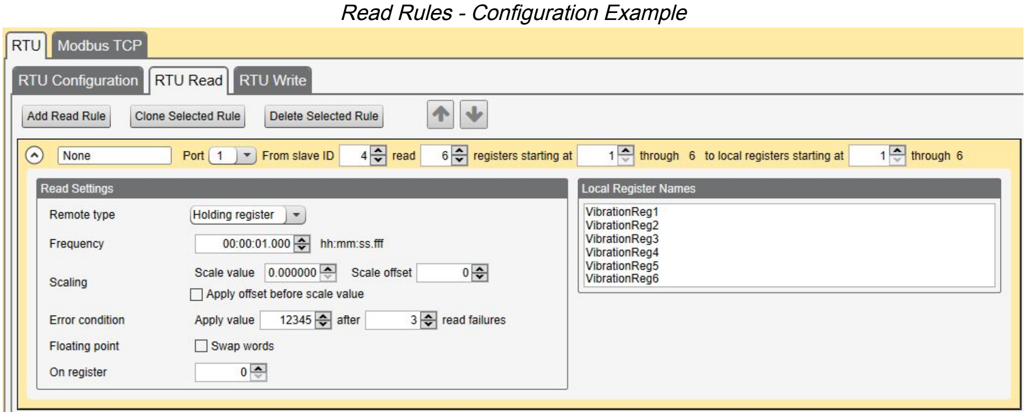


   b. Go to the **Register Mapping › RTU › RTU Configuration** screen.
   c. Modify the **Port** settings as needed.
   - Verify the **Baud Rate** and **Parity** match that of the connected Modbus server devices.

- The **Timeout** controls how long the DXM waits before determining a command failed to send. Set based on the specific application requirements.
- The **Delay between messages** defines the minimum wait time between resending another command. Set based on the specific application requirements.

2. From the **Register Mapping › RTU › RTU Read** screen, click **Add Read Rule**.
3. Click the arrow next to the name to display the parameters.
4. Name your rule.
5. Select the Port number to which the device is connected.
6. Select the Modbus ID of the device.
7. Select how many registers to read, and the beginning register.
8. Define the register type, how often to read the register, and any other appropriate parameters.
9. If necessary, select the error condition. For this example, if the read function fails after three attempts, the read rule writes 12345 to the DXM local registers. Notice the list of local register names this read rule is using.

Read Rules - Configuration Example



Baud Rate
: Defined for both the Modbus client and server

Settings include: 19200 (default), 1200, 2400, 9600, 38400, 57600, and 115200.

Delay between messages
: Applies to the Modbus client port

Sets the minimum wait time from the end of a Modbus transaction to the beginning of the next Modbus transaction.

Parity
: Defined for both the Modbus client and server

Settings include: None (default), odd, even, space, and mark

Timeout
: Applies to the Modbus client port

Covers the expected time for messages to be sent throughout the wireless network. For the DXM, the **Timeout** parameter is the maximum amount of time the DXM should wait after a request is sent until the response message is received from the Modbus server device.

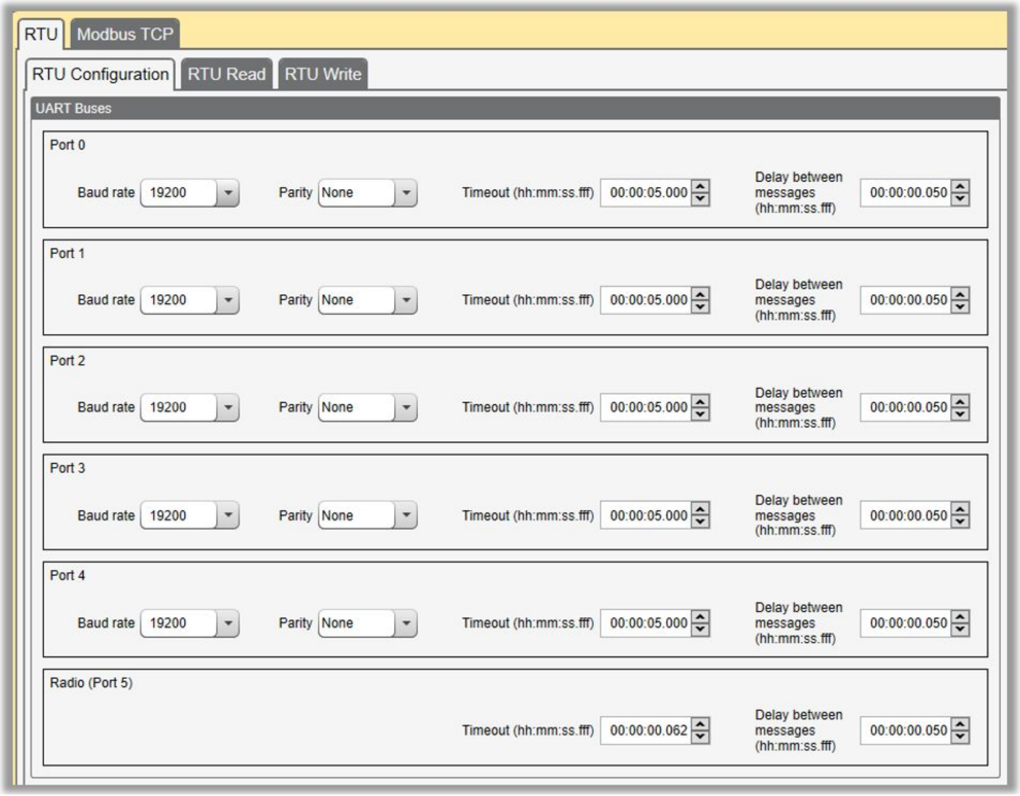
Create an RTU Read Rule for the ISM Radio of a DXM1200-B2 or -X2

Follow these steps to create a new read rule to access data from devices connected to the radio of a DXM1200-B2 or DXM1200-X2.

This example creates a read rule to read six registers (1 through 6), from the radio (port 5) at Modbus ID 1. The results are stored in the Local Registers 1 through 6.

1. Define the port settings to be compatible with the connected devices.
    a. Go to the **Register Mapping › RTU › RTU Configuration** screen.

RTU Configuration screen

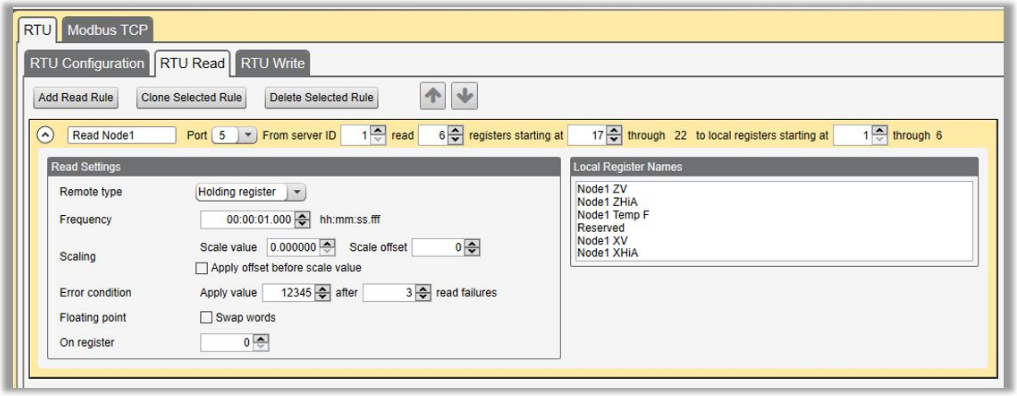


b. Go to the RTU configuration panel for **Radio (Port 5)**.
c. Modify the port settings as needed.

- **Timeout** controls how long the controller waits before determining a command has failed to send. Set the **Timeout** based on the specific application requirements. The minimum **Timeout** interval for Radio (Port 5) is 62.5 ms.
- **Delay between messages** defines the minimum wait time before sending another command. Set the delay based on the specific application requirements.

2. From the **Register Mapping › RTU › RTU Read** screen, click **Add Read Rule**.
3. Click the arrow next to the name to display the parameters.
4. Name your rule.
5. Select the port number to which the device is connected. For wireless devices, this is **Radio (Port 5)**.
6. Select the Modbus ID of the device. For wireless devices, this is **Modbus ID 1**.
7. Select how many registers to read and select the beginning register.
   Refer to the Modbus Holding Register table within the documentation for each server device.
8. Define the register type, how often to read the register, and any other appropriate parameters.
9. If necessary, select the error condition. For this example, if the read function fails after three attempts, the read rule writes 12345 to the DXM local registers. Notice the list of local register names this read rule is using.

Read Rules - Configuration Example



Create a Write Rule

The write rules write Local Register data to the defined Modbus server ID and registers. Use the **Register Mapping › Write Rules** screen to create a new write rule.

1. Click **Add Write Rule**.

   A new write rule is created.
2. Click the arrow next to the first rule to view the parameters.

   The user-defined parameters are displayed.
3. Type in the rule's name in the **None** field.
4. Select the number of registers to write to the target device.
5. Select the starting register of the local/source device.
6. Select the server ID of the target device.
7. Select the starting register of the target device.

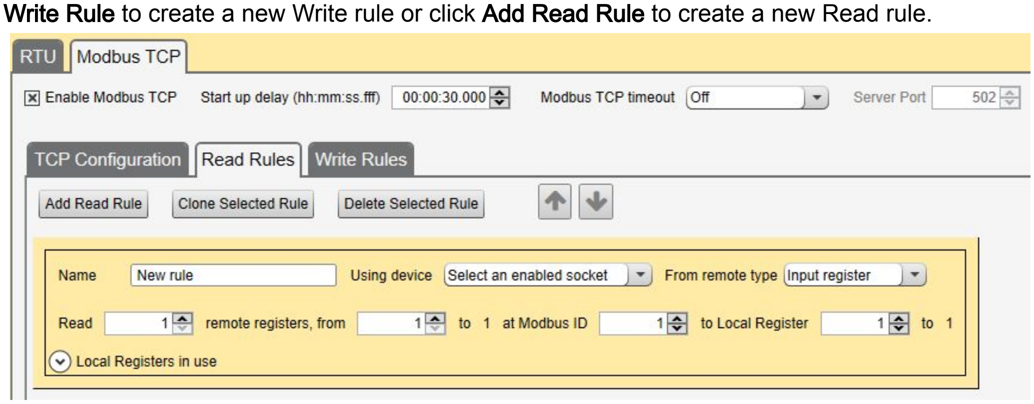
   The ending register automatically fills in based on the starting register and the number of registers selected in step 4.
8. Set the desired parameters.

> **TIP:** Use fewer read rules for applications that require higher response times by taking advantage of the Alternative Modbus Register groupings for DX80 Performance Nodes as shown in the DXM Wireless Controller Instruction manuals.

Create a Modbus TCP Write or Read Rule

Use the **Register Mapping › Modbus TCP › Write Rules** screen to define each Modbus TCP register write transaction to a selected Modbus TCP server device. Use the **Register Mapping › Modbus TCP › Read Rules** screen to define each Modbus TCP register read operation to a selected Modbus TCP server device.

1. Enable Modbus TCP.
2. Click **Add Write Rule** to create a new Write rule or click **Add Read Rule** to create a new Read rule.



3. Name your rule.

The name does not affect the rule's operation.

4. From the drop-down list, select the **Using device**. Device connections are defined on the **TCP Configuration** screen

   A new write or read rule is created. Write and read rule parameters are written in a form of a sentence to make it easier to understand the Modbus operation.

5. From the drop-down list, select the **Register type**. (This field is only available for read rules. Write rules always write to holding registers.

   Modbus TCP rules can currently be used to read from and write to Holding or Input registers. Note that prior versions of the DXM Configuration Software did not expose the **Register type** field. Modbus TCP behavior was previously restricted to holding registers.

6. Select the number of Local Registers, the starting register, Modbus ID, and the starting target register.
   - **Number of Registers**—For Write Rules, the number of internal Local Registers to write to the Modbus server. For Read Rules, the number of Modbus registers to read from the remote server device.
   - **Local Register Address**—For Write rules, the Local Register Address is the starting address of the Local Register data. For Read rules, the Local Register Address is the starting address of the Local Registers to put the read data
   - **Modbus ID**—An identifier is also known as the unit number. If there are multiple devices at the server IP address, this field uniquely identifies the server. For other DXM devices, this field will be 1. DXM devices do not respond to a unit number of 0.
   - **Remote Modbus Register Address**—For Write Rules, defines where to write the data in the server. For Read Rules, defines the starting address of the Modbus registers to read from the remote server device.

   The **Local Registers in Use** drop-down list shows all Local Register names associated to this write or read rule.

Click **Delete Last Rule** to remove the last created rule.

Create a Threshold Rule

Follow these instructions to create a new threshold rule.

1. From the **Action Rules › Thresholds** screen, click **Add Threshold Rule**.
2. Enter the **local register** you are comparing.
3. Select the mathematical function and value.
4. Select what the system should do if the register value is true or false.
5. Select any other values, such as **Hysteresis**, **On-Time**, or **Logging Options**.

Create a Register Copy Rule

Follow these instructions to create a new register copy rule.

1. From the **Local Registers › Action Rules › Register Copy** screen, click **Add Copy Rule**.
2. Enter the **Copy Register**. This is the first register for the source of the data.
3. Enter the **To Register** and **Through Register**. These are the registers for the beginning and end of the target range of registers.
   - If you are copying the contents of one register, enter the same register number into the **To Register** and **Through Register** fields.
   - If you are copying a range of registers, the range is defined by the **To Register** and **Through Register**. The **Copy Register** is the starting register of the source range.

Create a Math Rule

Use math rules to perform mathematical calculations on registers and store the results. Follow these instructions to create a new math rule.

1. From the **Action Rules › Math/Logic** screen, click **Add Math Rule**.
2. Enter a **Name** for your new math rule.
3. Select the desired **Operation** from the drop-down list.
4. Select the starting **Local Register**, And or Through, and the second (when using And) or ending (when using Through) **Local Register** to perform your mathematical calculations on.
5. Select the **Local Register** you'd like to store your results in.

Create a Control Logic Rule

Follow these instructions to create a new control logic rule.

1. From the **Action Rules › Control Logic** screen, click **Add Control Rule**.
2. **Name** your new rule.
3. Select the logical operator from the **Gate** drop-down list.
4. From the drop-down lists, select the Input 1, Input 2, Enable, and Clock options.
5. Select the **Output** options.

Create a Basic Trend Rule

Follow these instructions to create a basic trend rule.

1. From the **Action Rules › Trending** screen, click **Add Trend Rule**.
2. In the **Name** field, name your Trend Rule.
3. Define the **Register to Track**.
4. Define the **Sample Interval** (hours:minutes:seconds) you'd like to collect the data.
5. Select **Filters** and your **Filter Window**.
6. Enter the Local Registers to copy the **Average**, **Minimum**, and **Maximum** values into.

Create a Tracker Rule

Follow these instructions to create a new tracker rule.

1. From the **Action Rules › Trackers** screen, click **Add Tracker Rule**.
2. **Name** your new tracker rule.
3. Select the **Register to Track** from the list.
4. Select the **Transition to Track** from the drop-down list.
   - Rising edge—Transitioning from a zero to a non-zero value.
   - On time in ms—Length of time the register is on (non-zero), in milliseconds.
   - Off time in ms—Length of time the regsiter is off (zero), in milliseconds.
5. Select the **Result Register** to store the count in.

Create a Decoder Rule

Follow these instructions to create a new decoder rule.

1. From the **Local Registers › Action Rules › Decoders** screen, click **Add Decoder Rule**.
2. **Name** your new decoder rule.
3. Set the **Copy from Source Register** to define the source of the bits to be copied.
4. Set **Number of bits** and **Starting at bit index** to specify the subset of the source register's data to copy.
5. Set the **Destination Register** to define where the specified bits are copied.
6. Set the **Starting at bit index** (located near the Destination Register field) to specify which bit of the destination register the lowest-order bit of the copied data should occupy.

Processing the Rules

Rules and functions are evaluated by the DXM in a specific order.

1. The read rules are executed first, beginning with the first rule defined and continuing in the order the rules were entered into the DXM Configuration Software.
2. After the read rules are executed, the write rules are processed, in the same order.
3. After the write rules are processed, the DXM Configuration Software starts over with processing the read rules.

The read/write rules take time to complete, not because they require processing power, but because each rule has a lengthy overall communication time relative to the processor execution cycle time. So, in parallel, the action rules are also solved.

Each action rule is processed in order, similar to the read/write rules. The groups of action rules are solved in this specific order:

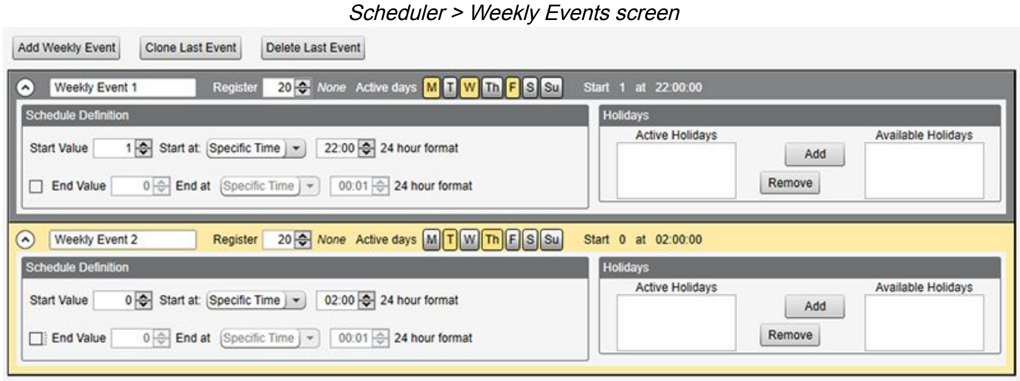
1. Calculate (Math) rules are next, continually processed
2. Copy rules, processed only when a change of state is detected on the source register
3. Threshold rules, continually processed
4. Control Logic, continually processed
5. Trending, continually processed
6. Trackers, continually processed

Creating Events and Schedules

Create a Weekly Event

Use the **Tools › Scheduler › Weekly Events** screen to define weekly events.

Scheduler > Weekly Events screen



1. Click **Add Weekly Event**.

   A new schedule rule is created.
2. Click on the arrow to the left of the new rule to expand the parameters into view.

   The user-defined parameters are displayed.
3. Name your new rule.
4. Enter the local register.
5. Select the days of the week this rule applies to.
6. Enter the starting value for the local register.
7. Use the drop-down list to select the type of Start at time: a specific time or a relative time.
8. Enter the starting time.
9. Enter the end time and end value for the local register.

Register updates can be changed up to two times per day for each rule. Each rule can be set for any number of days in the week by clicking the buttons M, T, W, Th, F, S, or Su.

If two register changes are defined for a day, define the start time to be before the end time. Select **End Value** to enable the second event in a 24-hour period. To span across two days (crossing the midnight boundary), set the start value in the first day, without selecting **End Value**. Use the next day to create the final register state.

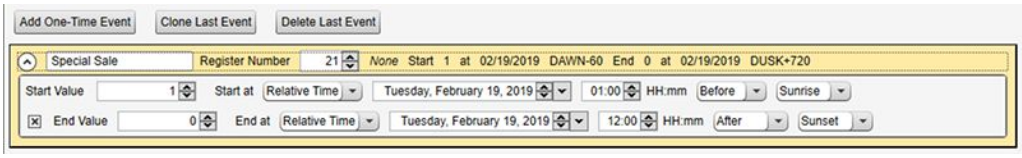
Start and end times can be specified relative to sunrise and sunset, or set to a specific time within a 24-hour period. When using sunrise or sunset times, set the GPS coordinates on the device so it can calculate sunrise and sunset.

Create a One-Time Event

Define one-time events to update registers at any time within a calendar year.

Similar to Weekly events, the times can be specific or relative to sunrise or sunset. Define one-time events using the **Tools › Scheduler › One Time Events** screen.

Scheduler > One Time Events screen

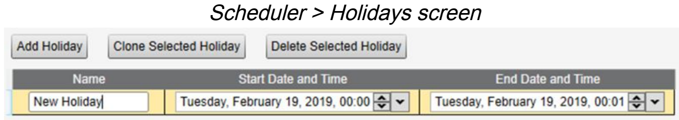


1. Click on **Add One Time Event**.
   A new one-time event is created.
2. Click on the arrow to expand the parameters into view.
   The user-defined parameters are displayed.
3. Name your one-time event by clicking on the name link and entering a name.
4. Enter the local register.
5. Enter the starting time, date, and starting value for the local register.
6. Enter the ending time, date, and ending value for the local register.

Create a Holiday Event

Use the **Tools › Scheduler › Holidays** screen to create date and/or time ranges that interrupt weekly events.

Scheduler > Holidays screen



1. Click on **Add Holiday**.
   A new rule is created.
2. Enter a name your new holiday rule.
3. Select the start date and time for the new holiday.
4. Select the stop date and time for the new holiday.

Create a Dynamic Schedule

Use the **Dynamic Update** screen to create a scheduler update file. The scheduler update file is a configuration XML file that contains only scheduler data definitions. When you upload the scheduler update file to a DXM that is running a defined schedule, the scheduler update file replaces the running schedule.

To use the Dynamic scheduler feature, follow these steps.

1. Create the main XML configuration file with the entire configuration of the DXM, including default scheduler settings.
2. Go to **Tools › Scheduler › Dynamic Update** and select **Enable dynamic scheduler**.
3. Go to **Device › Save** to save the XML configuration file.
4. Load the main XML file into the DXM.
   The DXM is running the main configuration file and will accept scheduler update files.
5. Modify the schedule settings, then click **Save scheduler update file**.
   The scheduler update file is created.
6. To load the scheduler update file, click **Load scheduler update file**.
7. To send the scheduler update file to the DXM, click **Send scheduler update to device**.
   The new schedule defined in the scheduler update file replaces the existing operating schedule.

Enable dynamic scheduler—Enables scheduler update files to be accepted. This must be selected in main XML configuration file loaded into the DXM before it will accept scheduler update files.

Get scheduler update from device—Downloads the scheduler update file from the DXM.

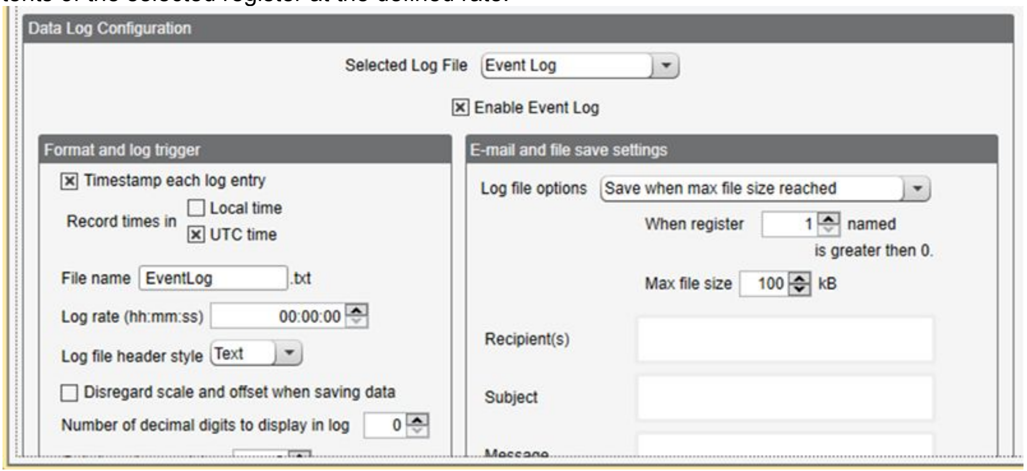
Send scheduler update from device—Uploads a scheduler update file to the DXM, replacing the existing schedule running on the DXM.

Save Scheduler update file—Saves a scheduler update file to the PC. The file only contains the scheduler configuration data.

Load scheduler update file—Loads a scheduler update file from the PC to the DXM Configuration Software software.

Create an Event Log or Data Log

Use the **Event Log** to track data only when the contents of the specified registers reach the defined value. The **Data Logs** record the contents of the selected register at the defined rate.



1. Select **Enable Log** to enable the log file creation. The register data to be saved in the log is defined under the **Local Registers** tab.
2. Select **Timestamp each log entry** to date/time stamp each log entry. Select whether you want the time stamps to be in local time or UTC time.

   The local time is determined from the device's time zone selection under on the **Settings › General** screen.
3. Name your Event or Data Log.

   A date/time stamp is added to the end of the user-defined **File Name** then stored on the Micro SD card. New files will be created with a new date/time stamp.
4. Select whether or not you want to **Disregard scale and offset when saving data** for the register or apply the scale and offset defined on the **Local Register Configuration** screen.

   Local Registers 1 through 845 are integer registers and are always stored as integers. Floating point Local Registers 1001 through 2000 are stored as floating point numbers.
5. Define the **Number of decimal points to display in log** (floating point data log only).
6. For cyclical log files (log1-log3), specify the **Log Rate**.
7. Set the **Maximum File Size** before the DXM creates a new file.
8. Select from the following **Log file options**.
   - Save when max file size reached—Save the log file to the internal DXM SD card when the defined Max file size is reached. The user must retrieve the file.
   - Save when the register value is non-zero—Save the log file to the internal DXM SD card when the defined Local Register is set to a non-zero value. The Local Register can be controlled by Action Rules, ScriptBasic, Scheduler, or Website. The user must retrieve the file.
   - Save Daily—The log file will be saved daily at 00:00 UTC time. The user must retrieve the file.

What is a Scheduled Push?

A scheduled push uses the DXM's **Scheduler** function to force a data push at a specific time. Use the DXM Configuration Software software to create, save, and upload the configuration file to the DXM.

Create a Scheduled Push

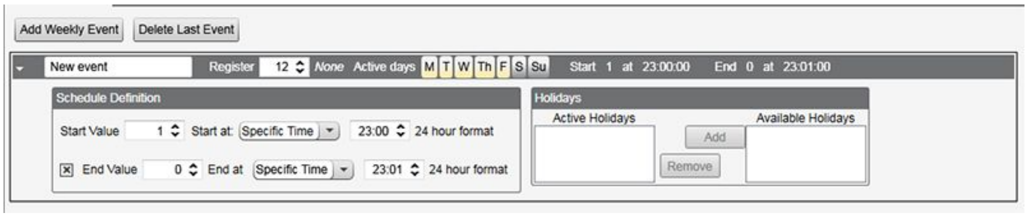
These instructions assume you have installed the latest version of the DXM Configuration Software and have launched it on your Windows-based PC.

The following example creates an automatic data push that runs at 11pm, Monday through Friday.

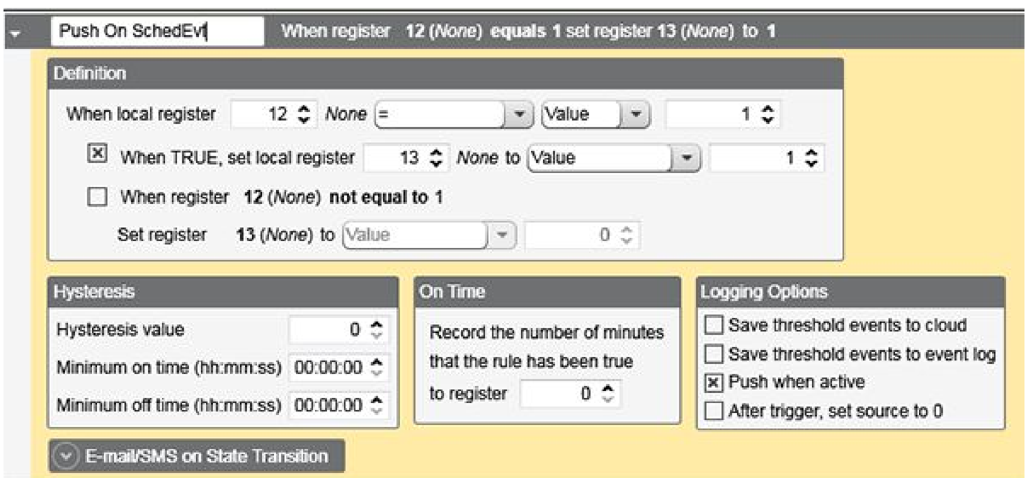
1. Define all local registers to push to the webserver.
   a. Go to the **Local Registers › Local Registers in Use** screen.
   b. Click the register number to display that register's parameters.

   c. Set **Cloud settings** to Read.
   d. Set **LCD Permissions** to Read to display the local register to the DXM's LCD.

2. Define the scheduled event.
   a. Go to the **Scheduler › Weekly Events** screen.
   b. Click **Add Weekly Event**.
   c. Click the arrow next to the new event to view all parameters.
   d. Enter a name for the event.
   e. Select the local register. For this example, we are using local register 12.
   f. Click on the days of the week that this local register will be changed. For this example, we have selected Monday through Friday.
   g. Select the start value and the specific time you want this event to occur. For this example, we have selected the start value of 1, to occur at 23:00 hours (11 pm).
   h. Select the end value and specific time you want this event to occur. For this example, the register value returns to zero at 23:01 (11:01 pm), one minute later.



3. Create an Action rule to push data to the webserver.
   a. Go to the **Local Registers › Action Rules › Thresholds** screen.
   b. Click **Add Threshold Rule**.
   c. Click the arrow next to the new rule to view all parameters.
   d. Enter a name for the rule.
   e. Fill in the parameters. For our example, we are setting local register 13 to 1 when local register 12 is 1.
   f. Select **Push when active**.



   When the value of register 12 is 1, the DXM pushes the defined data set to the webserver.

The Scheduler creates the timed event that occurs Monday through Friday. At the scheduled time and day, the value of local register 12 is set to 1 for one minute. The Action rule watches local register 12, and when the value is 1, the action rule creates a push event to the webserver.

Configure the DXM to Access the Webserver

Before the DXM can read or write data to the webserver, you must define or confirm several parameters.

1. Go to the **Settings › System** screen and set the **Device Time** and time zone.
   The device time can be verified on the DXM LCD.
2. Select whether or not the DXM should use daylight saving time (DST).
3. On the **Settings › Cloud Services** screen, set the **Cloud push interval** to none.
   This allows the action rule to push data.

4. Under the **Web Server** section, verify the **Site ID** is accurate. This Site ID is unique for every device and is created by the website.

Save and Upload the Configuration File

After making any changes to the configuration, you must save the configuration files to your computer, then upload it to the device.

Changes to the XML file are not automatically saved. Save your configuration file before exiting the software and before sending the XML file to the device to avoid losing data. If you select **DXM › Send XML Configuration to DXM** before saving the configuration file, the software will prompt you to choose between saving the file or continuing without saving the file.

1. Save the XML configuration file to your hard drive by going to the **File › Save As** menu.
2. Go to the **DXM › Send XML Configuration to DXM** menu.

Status indicator bar

   - If the Application Status indicator is red, close and restart the DXM Configuration Software, unplug and re-plug in the cable and reconnect the DXM to the software.
   - If the Application Status indicator is green, the file upload is complete.
   - If the Application Status indicator is gray and the green status bar is in motion, the file transfer is in progress.

   After the file transfer is complete, the device reboots and begins running the new configuration.

Verifying Communication Between a Modbus Sensor and MultiHop Radio or DXM

Follow these steps to monitor a Modbus Banner Engineering sensor connection and/or operation status when the sensor is connected to a MultiHop radio or a DXM Controller with an embedded MultiHop module.

Required equipment includes:

- Wireless DXM Controller client with a MultiHop radio module
- Wireless DXM Controller servers and/or MultiHop server radios
- Modbus sensor such as the Temperature/Humidity Modbus Sensor model M12FTH3Q, or an SDI-12 sensor
- Windows-based PC running the DXM Configuration Software v3 (downloaded from the Banner website)

To confirm an active radio communication between the sensor and radio, define **Read Rules** and **Action Rules**. Use two local registers to monitor each Modbus RTU sensor. Use an optional third register to monitor how long the sensor was not communicating with the radio.

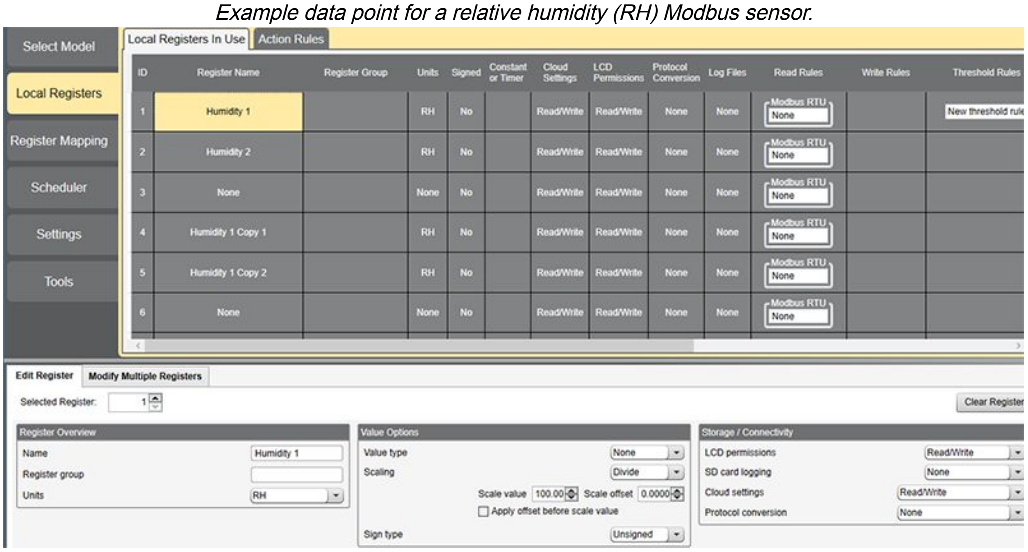
1. Connect to the DXM Controller client radio using serial or TCP/IP.
2. Define the **Local Registers**.
3. Define the **Read Rule**.
4. Define the **Threshold/Action Rule**.
5. Repeat these steps for each Modbus sensor and MultiHop server radio you'd like to track.

Define the Local Registers to Verify the Connection Between a Sensor and MultiHop Radio

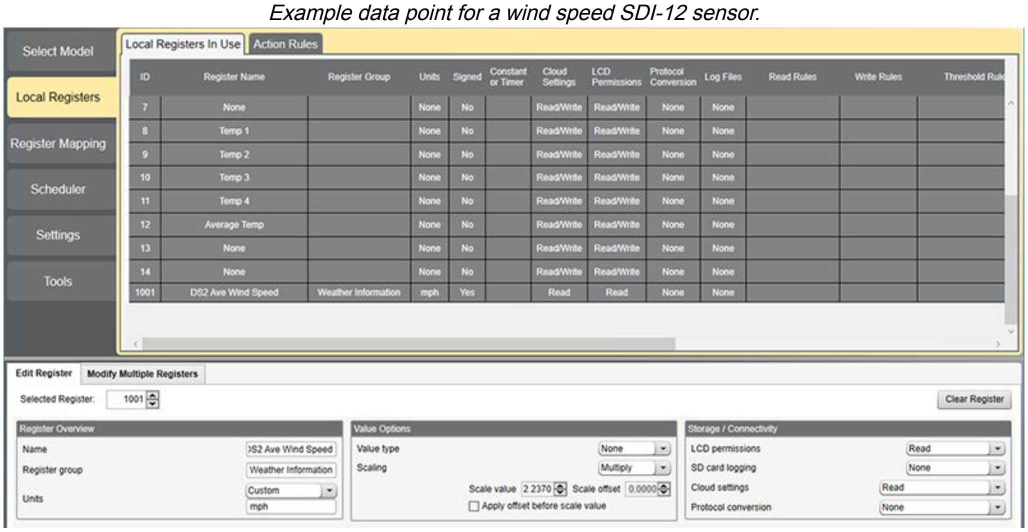
Define the local registers used to verify the connection between a Modbus sensor and a MultiHop radio.

1. Go to the **Local Registers › Local Registers in Use** screen.
2. Define a register to hold a data point.

Example data point for a relative humidity (RH) Modbus sensor.

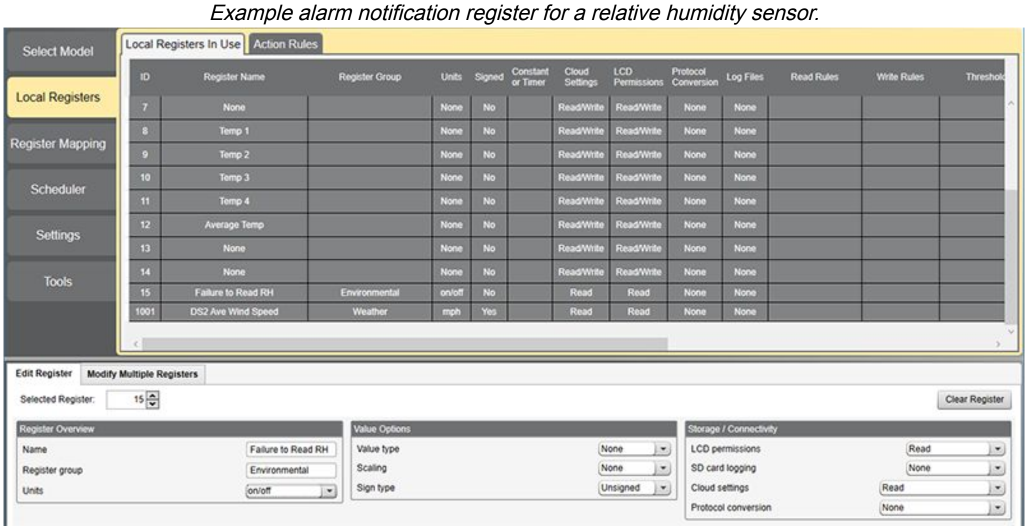


Example data point for a wind speed SDI-12 sensor.

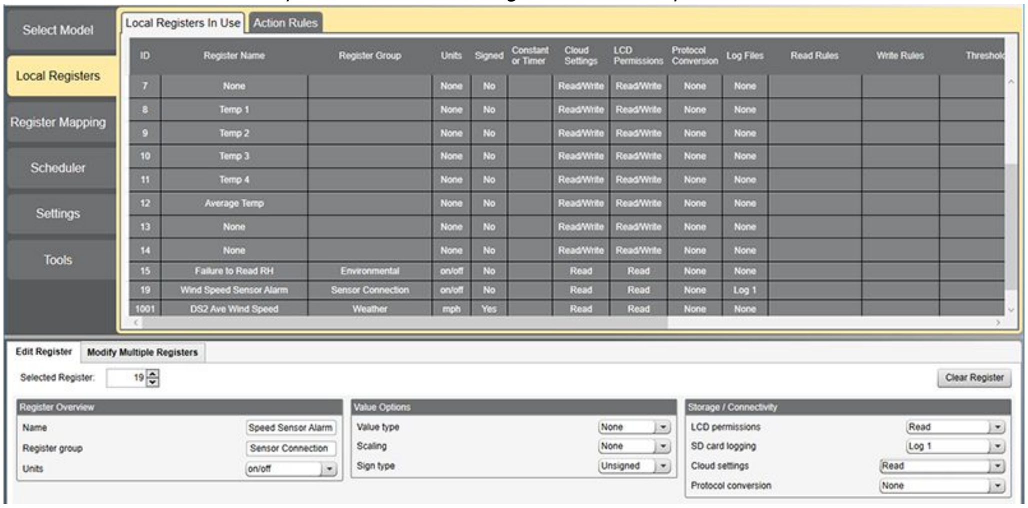


3. Define a register to be used as an alarm notification register when the MultiHop radio cannot communicate with the sensor.

Example alarm notification register for a relative humidity sensor.

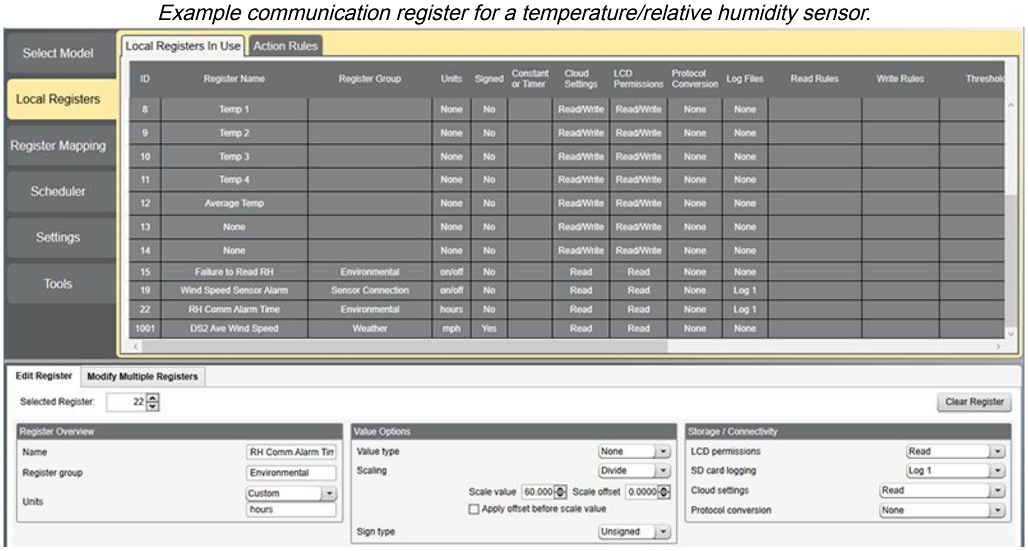


Example alarm notification register for a wind speed sensor.



4. Define a register to be used to track how long the Modbus sensor was not communicating with the client radio.

Example communication register for a temperature/relative humidity sensor.

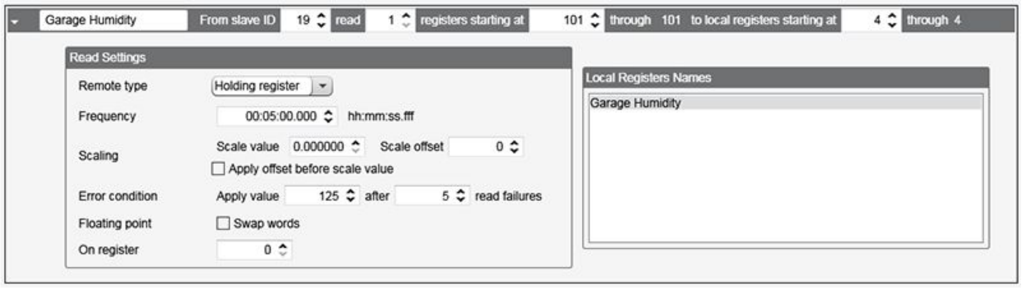


Create a Read Rule to Define Reading the Sensor Register

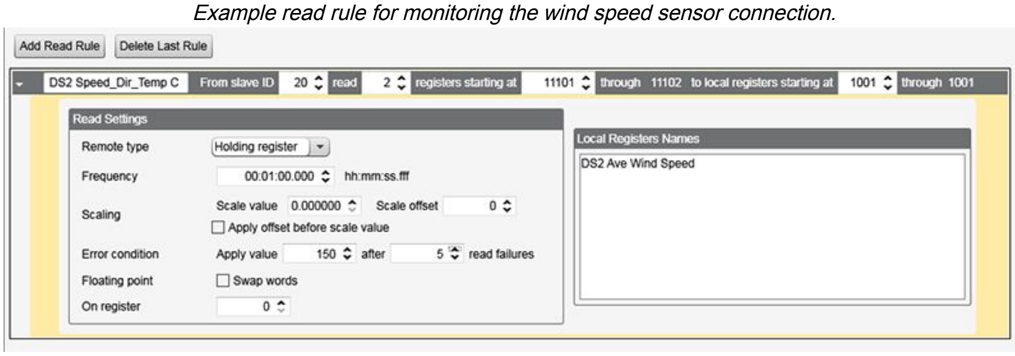
Create a Read Rule to define how often to read the sensor register and what to do if the communication attempt fails.

1. Go to the **Register Mapping › RTU › RTU Read** screen.
2. Click **Add Read Rule** to create a Read Rule.
3. Name the Read Rule and define from which server ID this register is being read, how many registers are being read, and the starting register.
4. Define how often to read this register (**Frequency**).
5. Define what value should be written to the register (**Apply value**) after the number of failed read attempts (**read failures**).

Example read rule for monitoring the relative humidity sensor connection..



Example read rule for monitoring the wind speed sensor connection.



For the relative humidity example, local register 4 (Garage Humidity) will be populated with the value from Modbus ID 19, register 101. The DXMclient radio attempts to communicate with the Modbus sensor (Server ID 19) every 5 minutes. After five consecutive unsuccessful attempts, the value of 125 is placed in the local register 19 (Failure to Read Garage RH).

For the wind speed example, local register 1001 (DS2 Wind Speed) will be populated with the value from Modbus ID 20, registers 11101 and 11102. The DXM client radio attempts to communicate with the Modbus sensor (Server ID 20) every 1 minute. After five consecutive unsuccessful attempts, the value of 150 is placed in the local register 19 (Failure to Read DS2 Wind Sensor).

> **NOTE:** You must place the SDI-12 sensor results in the 32-Bit Floating Point Register set in the DXM Controller. When using an SDI-12 sensor and an SDI-12 enabled MultiHop radio, when the DXM client or remote radios cannot communicate with the sensor, a value of 65535 is entered into the Results Register for that sensor in the Local Registers.

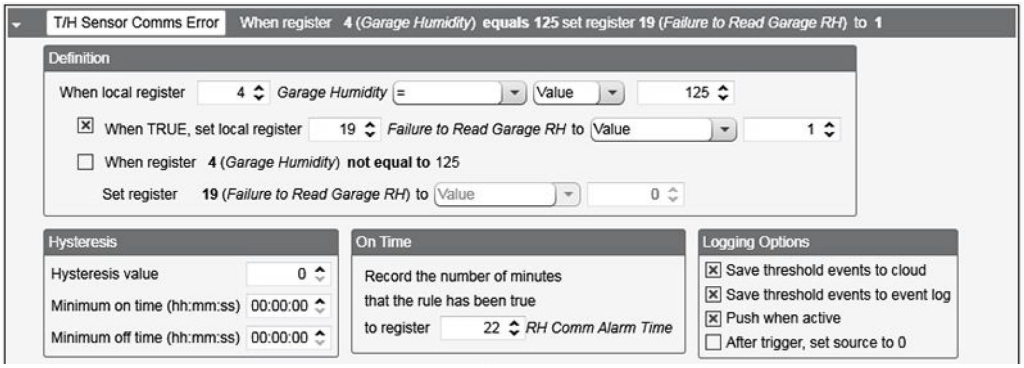
Select an alarm value that makes sense for the potential values of the application, but won't adversely affect graphing or charting the data point for analysis. For the RH example, the normal value for this local register is between 0 and 100. Therefore, 125 is not too excessive but is different enough to be a trigger value. For the wind speed example, 150 is a good choice.

Create a Threshold Rule to Verify Communication Between a Sensor and MultiHop Radio

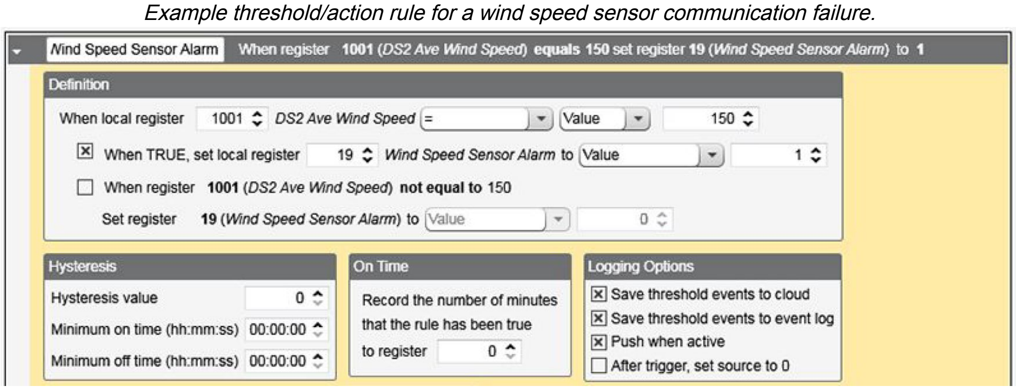
Create an action rule to define the behavior of the system when the communication fails.

1. Go to the **Local Registers › Action Rules › Thresholds** screen.
2. Click **Add Threshold Rule**.
3. Define a **Threshold Rule** so that when the local register Failure to Read value equals the error value (125 for relative humidity, 150 for wind speed), a value of 1 is entered into the Communication Alarm register.

Example threshold/action rule for a humidity sensor communication failure.



Example threshold/action rule for a wind speed sensor communication failure.



For the relative humidity example, when this register's value equals 1, local register 22 tracks how long this Modbus sensor was not able to be reached. The alarm is sent to the web server service, and the event is logged in the Events Log on the DXM.

Using Action Rules to Control External Sensors

Action rules allow for simple logic functions and simple manipulation of local register data. The processing of an action rule is autonomous from other local register functions.

- Threshold rules create event-driven conditions, such as events to the cloud or local logs
- Register Copy rules copy the contents of one register to another
- Math/Logic rules deal with 32-bit register logic with results from 0 to 4,294,967,295
- Control Logic rules are binary rules, with the results being either 0 or 1
- Trending rules find average, minimum, and maximum values
- Tracker rules monitor a Local Register value and store the result of a function in another register

The DXM can control external sensors using the timer/counter feature and Action Rules. In this example, the DXM controls a sensor using a cyclical loop with a timer. To configure this application, you must

1. Define the Local Registers
2. Define the Read/Write Rules
3. Define the Action Rules

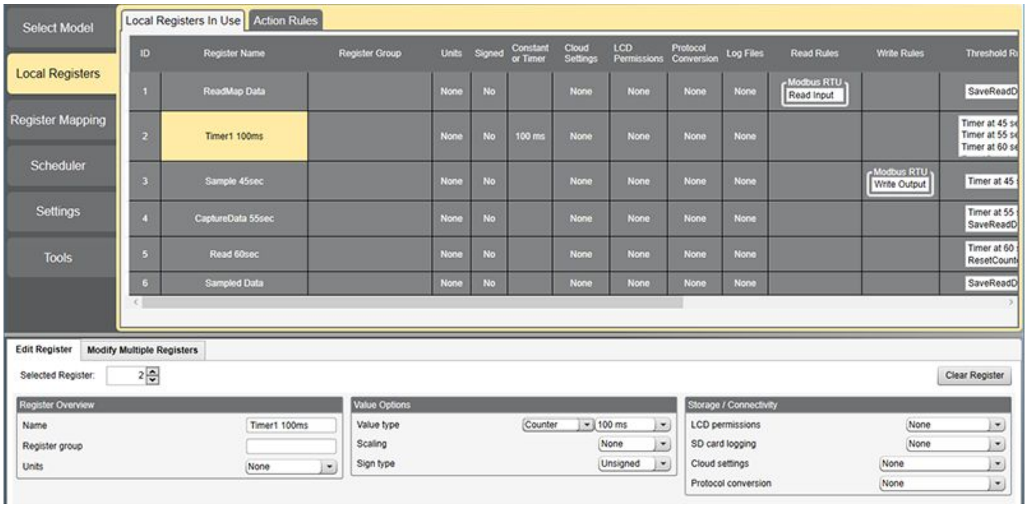
One of the local registers is defined as a timer, which starts counting at zero. When 45 seconds has passed, the output is turned on to supply power to the sensor. At 55 seconds, the data is captured from the I/O board and saved. At 65 seconds, the output is turned off and the counter reset.

Define the Local Registers

Use the DXM Configuration Software to define the local registers needed to control the external sensor.

This task assumes you have downloaded and installed the latest version of the DXM Configuration Software onto a Windows-based PC.

1. Go to the **Local Registers › Local Registers in Use** screen.
2. Name local register 1 to contain the read data from the DXM I/O board.
3. Define local register 2 to be a 100 ms timer/counter.



4. Name local register 3 to hold the result of Action Rule 1 (when the timer/counter reaches 45 seconds, this register value is 1).
5. Name local registers 4 and 5 to hold the results of Action Rules 2 and 3.
6. Name local register 6 as the saved sensor data.
7. Set the LCD permission to Read on all registers so that the register values display on the LCD.

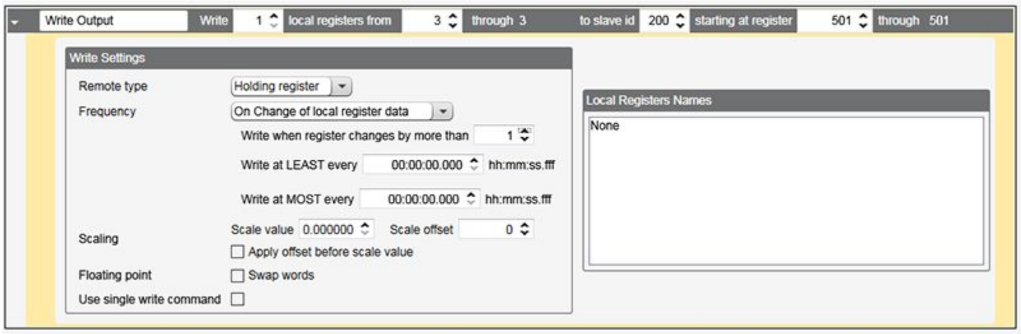
After this configuration file is saved and uploaded to the DXM, the register values display on the LCD under the **Register** menu.

Create the Read and Write Rules

Use Read/Write maps to write local register values to other Modbus devices or read Modbus registers from other Modbus devices.

For our example, the DXM I/O board is another Modbus device (server ID 200) to read to or write from.

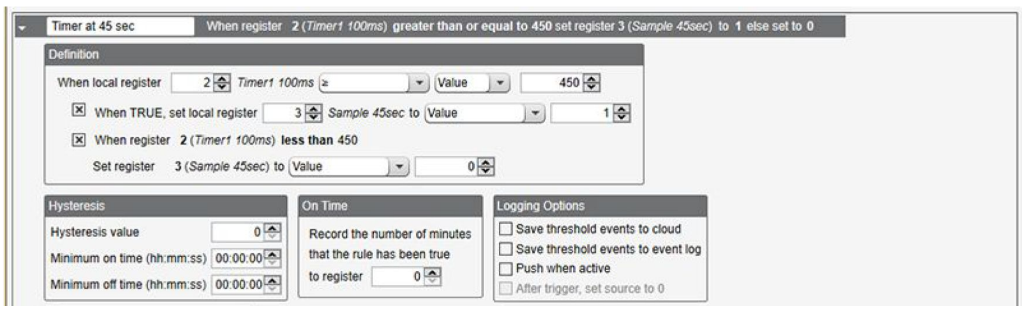
1. Go to the **Register Mapping › RTU › RTU Read** screen. Click **Add Read Rule**.
   A new read rule is created.
2. Click the arrow next to the new read rule to display the parameters.
3. Name your read rule.
4. Enter the **From server ID** and the number of registers to read from.
   For this example, we want to read from server ID 200, the DXM100 I/O board (the DXM700 I/O board is server ID 203).
5. Enter the frequency you'd like to read from this register.
   For our example, we'd like to read the selected register every 1 second.
6. Go to the **RTU Write** screen and click **Add Write Rule**.
   A new write rule is created.
7. Click the arrow next to the new write rule to display the parameters.
8. Name your write rule.
9. Enter the parameters to write from the local register to the target register.
   For our example, we have to write from local register 3 to a holding register on server ID 200, register 501. This writes a local register to an output on the DXM I/O board (server ID 200) to control the power to the external sensor. We want to write at a **Frequency** of **On Change of local register data** so that the output register 501 only changes when the local register value changes from 0 to 1.



Create the Action Rules

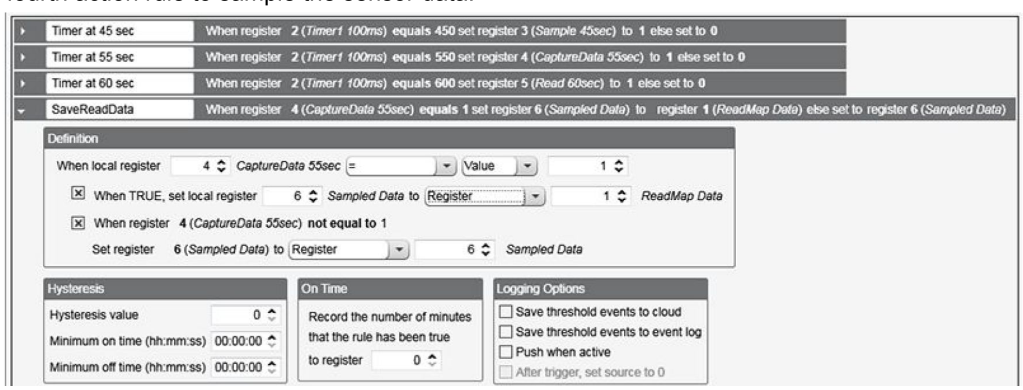
Five Action Rules define the logic statements required to complete the application. The first three action rules define when the timer is greater than 45 seconds, greater than 55 seconds, and greater than 60 seconds. The final two rules capture the read data and reset the timer back to zero to restart the process.

1. Go to the **Local Registers › Action Rules › Thresholds** screen and click **Add Threshold Rule**.
2. Name the first action rule (checks the timer to see if it has reached 45 seconds or greater) and enter the necessary parameters.



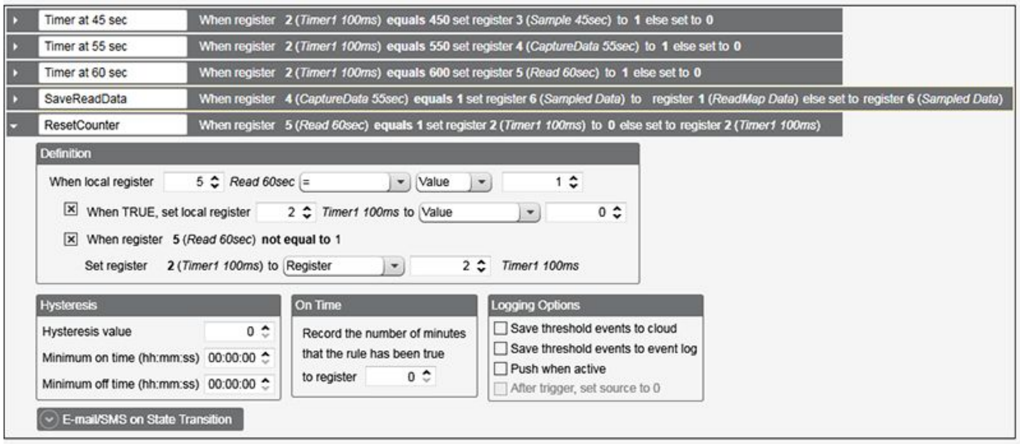
For this example, we want to check when local register 2 (the timer/counter) is greater than or equal to 45 seconds. When true, it sets local register 3 to 1, otherwise, set local register 3 to zero.

3. Create the second and third action rules, which also check the timer count (local register 2) and change the values of local register 4 (CaptureData 55sec) and local register 5 (Read 60sec), respectively.
4. Create the fourth action rule to sample the sensor data.



This action rules detects when local register 4 (timer at 55 seconds) is 1. The value of local register 6 is set to the value of local register 1 (ReadMap Data). When the value of local register 4 is not 1, local register 6 remains unchanged.

5. Create action rule 5 to reset the counter value and begin the count again.



When the timer/counter register has reached 60 seconds (local register 5 is 1), set the value of the timer register back to 0. Otherwise, do not change the value.

Save and Upload the Configuration File

After making any changes to the configuration, you must save the configuration files to your computer, then upload it to the device.

Changes to the XML file are not automatically saved. Save your configuration file before exiting the software and before sending the XML file to the device to avoid losing data. If you select **DXM › Send XML Configuration to DXM** before saving the configuration file, the software will prompt you to choose between saving the file or continuing without saving the file.

1. Save the XML configuration file to your hard drive by going to the **File › Save As** menu.
2. Go to the **DXM › Send XML Configuration to DXM** menu.

Status indicator bar

- If the Application Status indicator is red, close and restart the DXM Configuration Software, unplug and re-plug in the cable and reconnect the DXM to the software.
- If the Application Status indicator is green, the file upload is complete.
- If the Application Status indicator is gray and the green status bar is in motion, the file transfer is in progress.

After the file transfer is complete, the device reboots and begins running the new configuration.

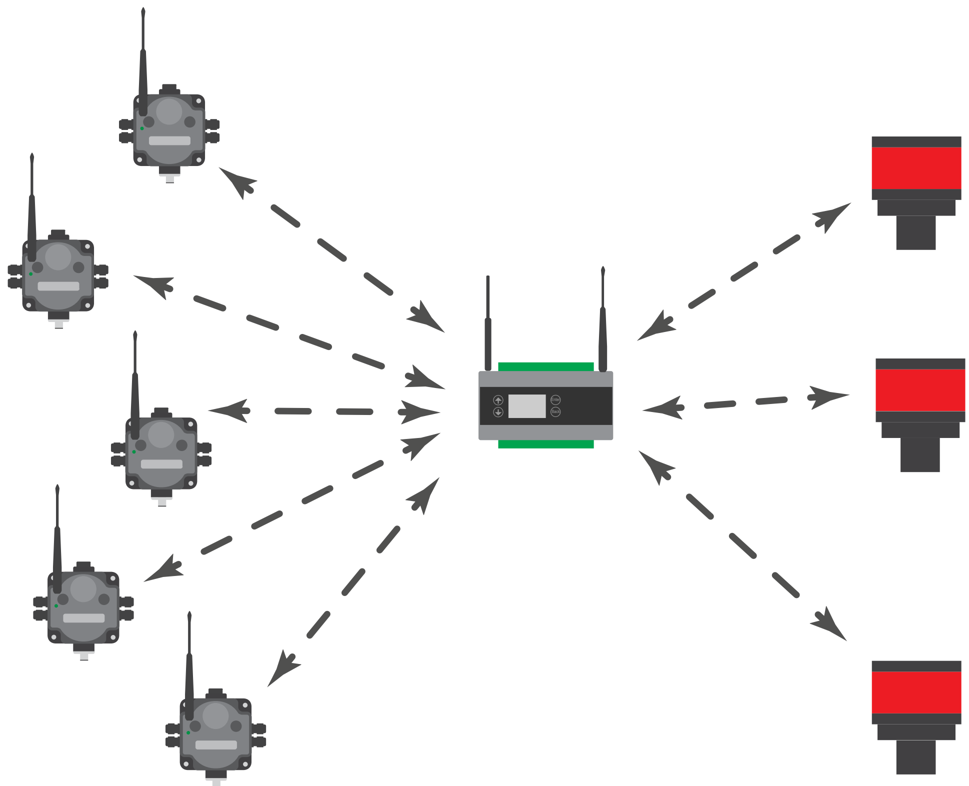
Map Many Inputs to One Output Using Action Rules

Use the DXM and Action Rules to read multiple inputs, logically OR the values, then write the result to an output or outputs. There are unlimited variations that can be accomplished using Action Rules and Read/Write maps.

This example does not explain how to use the DXM Configuration Software or discuss details of the DXM configuration. For help using the DXM Configuration Software, refer to the DXM Configuration Software Instruction Manual (p/n 158447). For help setting up the various operations of the DXM, refer to:

- DXM100-Bx Wireless Controller Instruction Manual (p/n 190037)
- DXM150-Bx Wireless Controller Instruction Manual (p/n 190038)
- DXM700-Bx Wireless Controller Instruction Manual (p/n 207894)
- DXM1200-Bx Wireless Controller Instruction Manual (p/n 216539)

The example application is saved in the configuration file `ManytoOne.xml`. The applications reads a single discrete input on five different DX80 Nodes. If any input is activated, three different TL70 wireless stack light outputs are turned on. The discrete input is connected to the wireless Nodes on input 1 and the output on the stack lights are on the first output (Modbus register 9 on the Wireless TL70 Node)



1. Define the DXM Local Registers.
2. Define the DXM's Action Rules.
3. Create the register Read and Write Rules.
4. Save the configuration file and upload the file to the DXM.

Define the Local Registers to Map Many Inputs to One Output

The Local Registers are the global storage area. Data is read from or written to the Local Registers.

1. Go to the **Local Registers › Local Registers in Use** screen.
2. Define the first five Local Registers as the Node 1 through 5 switch registers by naming them and setting **LCD permissions** to Read.
3. Define Local Register 6 to be the output data transmitted to the Wireless TL70 Nodes by naming it and setting **LCD permissions** to Read.
4. Define Local Registers 7 through 9 for the output data to be sent to the TL70 Nodes by naming them and setting **LCD permissions** to Read.

   This example uses TL70 OUT A through OUT C.

The local registers are set up for the Action Rules and Read/Write Rules.

ID	Register Name	Register Group	Units	Signed	Constant or Timer	Cloud Settings	LCD Permissions	Protocol Conversion	Log Files	Read Rules	Write Rules
1	N1 Input 1		None	No		None	Read	None	None		
2	N2 Input 1		None	No		None	Read	None	None		
3	N3 Input 1		None	No		None	Read	None	None		
4	N4 Input 1		None	No		None	Read	None	None		
5	N5 Input 1		None	No		None	Read	None	None		
6	Logic OR		None	No		None	Read	None	None		
7	N6 Out 1		None	No		None	Read	None	None		
8	N7 Out 1		None	No		None	Read	None	None		
9	N8 Out 1		None	No		None	Read	None	None		

Create the Action Rule to Map Many Inputs to One Output

Create an Action Rule that logically ORs Local Registers 1 through 5 and writes the ORed value to Local Register 6, the results register for the light outputs.

1. Go to the **Local Registers › Action Rules › Math/Logic** screen and click **Add Math Rule**.
2. Name the math rule.
3. From the drop-down lists, select Logical OR (**Operation**), Local Registers 1 through 5, and store the result in Local Register 6.

4. Go to the **Local Registers › Action Rules › Register Copy** screen and click **Add Copy Rule**.
5. Name the copy rule.

   For this example, the copy rule is named CopyOut.
6. From the drop-down list, select the Local Register to out from.

   Our example is copying from Local Register 6.
7. From the drop-down lists, select the Local Register(s) to copy to.

   Our example is copying to Local Registers 7 through 9. The names of the target registers are listed.

Create the Read and Write Rules to Map Many Inputs to One Output

Define the rules to read the switch input values from the internal DX80 Gateway (Modbus Server 1) and write the values to Local Registers.

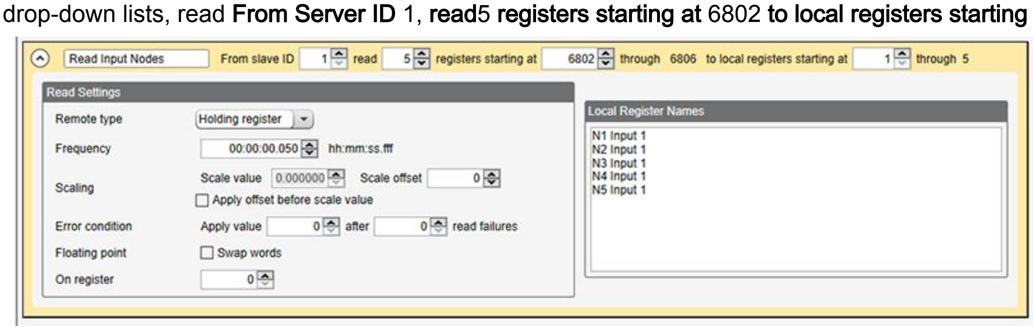
The DXM's internal ISM radio is Modbus Server ID 1.

The four DX80 Gateway registers to read for Nodes 1 through 5, input 1, are 6802 through 6806. This register location organizes the Modbus registers by each input.

Define the Write Rule to write values to the Wireless TL70 Node registers. For this example application, the three Wireless TL70s are Nodes 5, 6, and 7. Because the DX80 Gateway handles all Modbus registers, write to the Gateway Modbus registers for Nodes 5 through 7, register 9, at addresses 8002–8004.

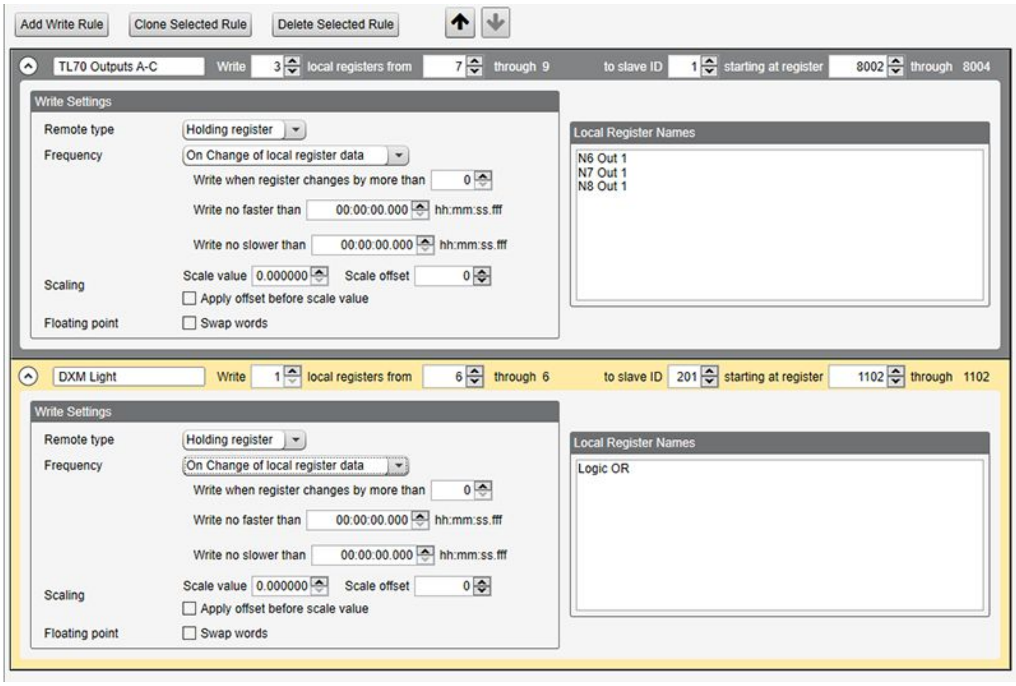
See *Alternate Modbus Register Organization in the DX80 Host Controller Systems Manual* (p/n 132114) for more information about alternate Modbus registers.

1. Go to the **Register Mapping › RTU › RTU Read** screen and click **Add Read Rule**.
2. Click the arrow to display the read rule parameters.
3. Name your new rule.
4. From the drop-down lists, read **From Server ID** 1, **read**5 **registers starting at** 6802 **to local registers starting at** 1.



5. Set the **Frequency** to 0.50 seconds.

6. Go to the **Register Mapping › RTU › RTU Write** screen and click **Add Write Rule**.
7. Click the arrow to display the write rule parameters.
8. Name the write rule and select the following parameters. Select a write **Frequency** of On change of local register data.



9. Create a second write rule, similar to the first one, that takes the logical OR value from Local Register 6 and writes it to DXM LED 1.

   For more information on the user programmable indicator LEDs, see the DXM100/1000-Bx Instruction Manual (p/n 190037).

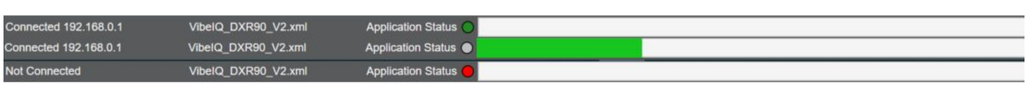
Save and Upload the Configuration File

After making any changes to the configuration, you must save the configuration files to your computer, then upload it to the device.

Changes to the XML file are not automatically saved. Save your configuration file before exiting the software and before sending the XML file to the device to avoid losing data. If you select **DXM › Send XML Configuration to DXM** before saving the configuration file, the software will prompt you to choose between saving the file or continuing without saving the file.

1. Save the XML configuration file to your hard drive by going to the **File › Save As** menu.
2. Go to the **DXM › Send XML Configuration to DXM** menu.

Status indicator bar



   - If the Application Status indicator is red, close and restart the DXM Configuration Software, unplug and re-plug in the cable and reconnect the DXM to the software.
   - If the Application Status indicator is green, the file upload is complete.
   - If the Application Status indicator is gray and the green status bar is in motion, the file transfer is in progress.

   After the file transfer is complete, the device reboots and begins running the new configuration.

Enabling Flash State Recognition on a TL70 Wireless Tower Light

These instructions describe how to enable the TL70 Wireless Tower Light to recognize a 0.8 Hz to 6 Hz flashing state produced on the input. This feature allows each light segment to have two separate states that can be recognized and tracked for reporting or triggering rules or functions within the DXM100 Wireless Controller.

To enable the flash state recognition, change the Report Type and I/O Configuration using the DX80 User Configuration Software. The TL70 must be bound to either the DXM100 or a DX80 Gateway.

> **IMPORTANT:** If using the Machine Monitoring / OEE Solutions Guide, this Tech Note DOES NOT need to be followed as the script loaded into the file will automatically do this for any TL70 Wireless Tower Light bound to the DXM.

Equipment used:

- DXM100 OR DX80 Performance Series Gateway (either 900 MHz or 2.4 GHz)
- TL70DXNx TL70 Wireless Tower Light (900 MHz or 2.4 GHz to match the DXM100 or Gateway) with RF Firmware version 6.4 or higher

To configure for flashing recognition, follow these steps:

1. Download the DX80 User Configuration Software from the Banner website and install on a computer.
2. Apply power to the DXM or DX80 Gateway and the TL70 Wireless Tower Light.
3. Follow binding procedures listed on the TL70 datasheet to bind the Node to the Gateway.
4. Using a USB cable, connect the Gateway to the computer with the configuration software installed.
5. Connect the software to the Gateway by going to the **Device › Connection Settings** menu.
6. Select **Serial** if using a DX80 Gateway or **Serial DXM** if using the DXM100. Select the COM port the USB cable is plugged into and click **Connect**. If you are unsure which COM port and multiple appear, attempt to connect to each one of them until you are successful.
7. Go to the **Configuration** screen and select **Nodes Currently in the System** from the drop-down list. Click **Get devices in system**.
8. Use the arrow next to the Node(s) that appear below the Gateway bar to expand the Node's parameters.
9. Read the current parameters for each input of the Node. Click **GET** next to that input. This takes less than one minute for each **GET**.
10. For each input (1 through 6) that needs to recognize flashing states, use the arrow next to that input to expand its parameters.
    a. Under **I/O configuration**, change the **Report Type** to Analog
    b. Under **Serial options** change the **IO configuration** to 24
11. Click **SEND** next to that updated input to send the new parameters to the device.
12. Repeat steps 9 through 11 for each input and Node that needs to recognize a flashing input.

Typical TL70 Wireless Tower Lights only indicate an ON or OFF state via a 1 or 0 when reading the input status for each light module. With the flashing input updates as described are applied, the unit displays four different states when reading the input status:

- 0 - OFF state
- 1 - ON state
- 2 - FLASHING OFF State
- 3 - FLASHING ON State

The input status 2 and 3 rotate back and forth as long as the light recognizes an input that is transitioning between ON and OFF at a frequency of at least 0.8 Hz and no faster than 6 Hz. If the input is transitioning slower than 0.8 Hz or faster than 6 Hz it may not display the input FLASHING status correctly.

Chapter Contents

Chapter 2 Software Screens

The following sections explain the function of each screen.

Local Registers Screen

The main storage elements for the DXM are its Local Registers. The Local Registers store 4-byte values that result from register mapping, action rules, or ScriptBasic.

Since a Modbus register is only 16 bits, all transactions with Modbus devices use the lower 2 bytes (16 bits) of the Local Registers.

The Local Register characteristics are defined on the **Local Registers** screen of the DXM Configuration Software.

Description of Local Registers	Local Registers		
	DXM100, DXM100-A1, and DXM150	DXM700, DXM1000, DXM1200, DXM1500, DXMR90-X1, DXMR90-X1E, DXMR90-4M, and R70ER	DXMR90-4K and DXMR110-8K
IO-Link registers	N/A	N/A	1–12000
32-bit unsigned integer-based registers	1–845	1–845 and 5001–7000	12001–12845 and 17001–19000
Special function registers that can be reset registers	846–850	846–850	12846–12850
Non-volatile registers with limited write capability for permanent data storage. Refer to the DXM Instruction Manual for more information.	851–900	851–900 and 7001–8000	12851–12900 and 19001–20000
32-bit IEEE 754 floating point registers. Floating point values require two Local Registers to store a value. Floating point Local Registers are referenced on the odd-numbered register addresses, 1001, 1003, 1005. When using Action Rules/Read Rules always reference the odd-numbered register addresses.	1001–1900	1001–5000	13001–17000

Local Register Parameters

The following parameters may appear on multiple register configuration screens.

Select the register to define.

Register Overview

Name
Set a name for the register. This parameter is only used for the display on the DXM LCD and for the BannerCDS cloud services website. As soon as you name a register, the register appears in the **Local Registers in Use** list.

Register group
Register Groups can be used to easily sort related registers on the Local Registers In Use table. Enter a tag to create a register group.

Units
Select the unit information for the Local Register data. This parameter field is only used for display on the DXM LCD and for the Banner Cloud Data Services cloud services website. Select the Custom field to create user-defined units.

Value Options

Value type
Defines the behavior of the register's value. Use the drop-down list of select:

None—Register has a value of 0 by default and can be modified by any source that has permission to set register values.

Constant—Forces the local register to be set to the specified value. Use to compare values when using action rules. Constant values cannot be overwritten by any source that would normally have permissions to modify register values.
Counter—The value increments with the specified frequency (every 100 ms or every 1 second). Write to the Local Register to start the timer at a specific value.

Scaling
The scaling parameter changes the viewing of Local Register data. Scaling is performed in two steps: applying a scale (one of five functions: add, subtract, multiple, divide, or modulo (remainder in division)), then adding an offset value. The step order can be changed to add offset first and then apply the scale by selecting the "Apply offset before scale value" checkbox.

Sign type
Supported for integer register ranges only. Set to unchanged, default, or set (Unsigned, Signed 32-bit, or Signed 16-bit).

Select **Unsigned** data format to show Local Register data as a positive integer.

Select **Signed 32-bit** format to treat the data as a two's complement value, displaying value as positive or negative.

Select **Signed 16-bit** format for specialized applications, such as ensuring compatibility with certain Banner Engineering Solutions Guides.

Storage/HTTP Connectivity

LCD permissions
Defines whether the Local Register contents will display on the DXM's Registers menu. Use the drop-down list to select from the following options:

None—The Local Register will not display on the DXM.
Read—The Local Register will display on the DXM.
Read/Write—The DXM can read and write the Local Register value from the LCD.

SD card logging
Defines the onboard micro SD card storage of this local register. Up to three different log files can be stored on the micro SD card, each with different logging definitions. Select from the following options:

None—No logging selected
Log 1—Save this Local Register data to the Data Log 1.
Log 2—Save this Local Register data to the Data Log 2.
Log 3—Save this Local Register data to the Data Log 3.

Cloud settings
Defines the register's interaction with the Banner Cloud Data Services website (https://bannercds.com). Use the drop-down list to select from:

None—The web server cannot access this register.
Read—The web server can read the value of this register.
Write—The web server can write a new value to this register.
Read/Write—The web server can both read and write the value of this register.
Push at Boot Time—Push register value to the web server whenever the DXM boots up.
Push at UTC 00:00—Push register value to the web server at midnight UTC.

Protocol conversion
Defines this Local Register as either an input or output for EtherNet/IP™. (Modbus TCP has access to all Local Registers and does not need to be defined.)

None—This register is not exposed to the host PLC over EtherNet/IP.
EIP Originator › DXM—For EtherNet/IP, data from the host PLC is written to the DXM Local Register.
EIP DXM › Originator—For EtherNet/IP, data from the DXM Local Register is sent to the host PLC.

MQTT

Publish via MQTT
Denotes whether the register is included in MQTT publishes (see **Settings › MQTT** for the configuration).

Sensor #
This register publishes with other registers in the same MQTT push group.

Define the push groups on the MQTT settings screen. Each push group can have unique destinations and publish rates.

Apply Scale and Offset for MQTT
Indicates whether calculations defined in the Scaling parameter (if any) should be applied when publishing this register's value to MQTT. Values are uploaded as integers; floating point values are truncated to the integer during the upload process.

MQTT is only supported for DXM700, DXM1000, and DXM1200 models. The four related parameters (Apply Scale/ Offset, push group, Publish when value changes, and Push to MQTT) are only visible when one of these models is selected.

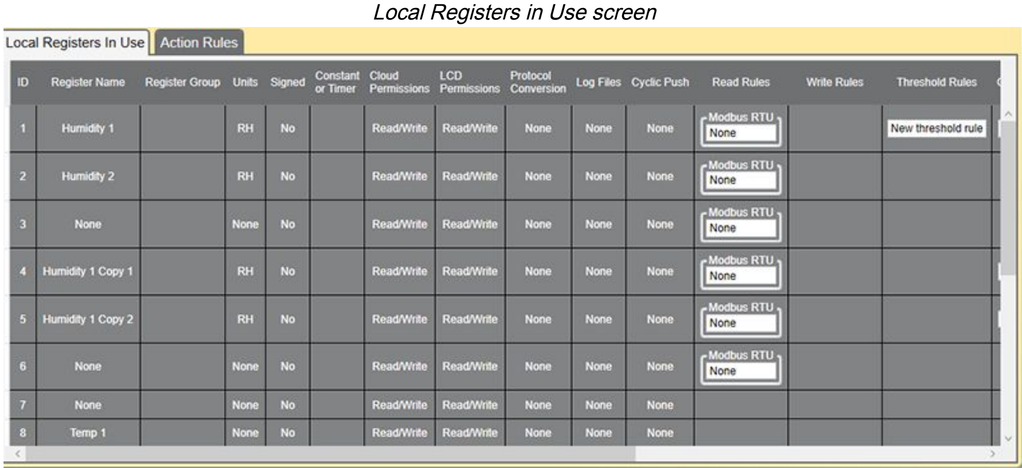
To modify MQTT parameters for multiple registers at once, use the **Batch MQTT Configuration** screen. This screen is visible only when a supported DXM model is selected.

Publish when value changes
If selected, this register publishes when its value changes in addition to the normally scheduled MQTT publishes.

Local Registers in Use

Use the **Local Registers in Use** screen to view a list of defined registers or to make changes to registers and show where the register is used in the defined configuration.

Local Registers in Use screen



ID	Register Name	Register Group	Units	Signed	Constant or Timer	Cloud Permissions	LCD Permissions	Protocol Conversion	Log Files	Cyclic Push	Read Rules	Write Rules	Threshold Rules
1	Humidity 1		RH	No		Read/Write	Read/Write	None	None	None	Modbus RTU None		New threshold rule
2	Humidity 2		RH	No		Read/Write	Read/Write	None	None	None	Modbus RTU None		
3	None		None	No		Read/Write	Read/Write	None	None	None	Modbus RTU None		
4	Humidity 1 Copy 1		RH	No		Read/Write	Read/Write	None	None	None	Modbus RTU None		
5	Humidity 1 Copy 2		RH	No		Read/Write	Read/Write	None	None	None	Modbus RTU None		
6	None		None	No		Read/Write	Read/Write	None	None	None	Modbus RTU None		
7	None		None	No		Read/Write	Read/Write	None	None	None			
8	Temp 1		None	No		Read/Write	Read/Write	None	None	None			

The lower section of this screen shows the Register Panel, a one-stop tool for configuring local register behavior. This panel includes the **Edit Register**, **Modify Multiple Registers**, and **Batch MQTT Configuration** tabs.

The **Edit Register** tab presents all parameters for a single register. See "Edit Registers Using the Local Registers in Use Screen" on page 6.

The **Modify Multiple Registers** tab allows contiguous groups of registers to be modified in a single operation. See "Modify Multiple Registers" on page 6.

Use the **Batch MQTT Configuration** tab to configure your MQTT functions for contiguous groups of registers.

The Register Panel appears automatically any time a local register-related tab is selected. Click and drag the gray bar along the panel's top margin to resize it. The Register Panel is set to its maximum height by default.

To quickly modify a register seen on the **Local Registers In Use** table, click on any cell in the register's row. That register will be loaded into the **Register Panel**, and the table will be automatically updated as you make changes.

Action Rules Screen

Action rules allow for simple logic functions and simple manipulation of local register data. The processing of an action rule is autonomous from other local register functions.

- Threshold rules create event-driven conditions, such as events to the cloud or local logs
- Register Copy rules copy the contents of one register to another
- Math/Logic rules deal with 32-bit register logic with results from 0 to 4,294,967,295
- Control Logic rules are binary rules, with the results being either 0 or 1
- Trending rules find average, minimum, and maximum values
- Tracker rules monitor a Local Register value and store the result of a function in another register

Read rules are executed first, beginning with the first rule defined. The read rules execute in the order they were entered into the DXM Configuration Software. After all read rules execute, the write rules are processed, in order. After the write rules are processed, the system begins again with the read rules.

Processing the read/write rules takes a long time to complete, not because they take a lot of processing power, but because each rule has a lengthy overall communication time relative to the processor execution cycle time. So, in parallel, action rules are solved. Each action rule is processed in order, similar to the read/write rules. The groups of action rules are solved in this specific order:

1. Calculate (Math), continually processed
2. Copy rules, processed only when a change of state is detected on the source register
3. Threshold rules, continually processed
4. Control Logic, continually processed
5. Trending, continually processed
6. Trackers, continually processed

> **IMPORTANT:** Define all action rules to use integer-based Local Registers or floating point Local Registers. Do not mix the two Local Register types within an Action Rule. To move integer-based Local Registers 1–850 to floating point Local Registers, use the Register Copy rules.

All Action Rules screens have up/down arrows near the top of the screen that scroll through the rules. All Action Rules also have **Clone Selected Rule** or **Delete Selected Rule** buttons to quickly copy/paste or delete the selected rule.

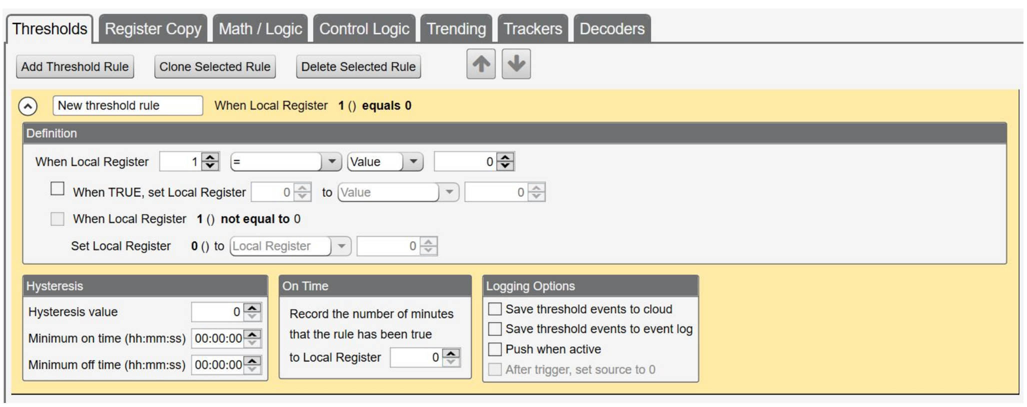
Threshold Rules

A **Threshold Rule** triggers event messages sent to the cloud (event-driven push), triggers events to be stored to the local event log and creates a standard push message to the cloud with all defined registers being sent.

A Threshold rule creates a simple IF-THEN-ELSE comparison for a register to determine its value and set another register to indicate if the rule is true or false (you must select a true option to also set a false option). The definition section of the threshold rule sets the comparison and values. The definitions of the threshold rules can further be defined by the optional parameters, Hysteresis, On Time, and Logging options.

If selected, the false option (else condition) is applied. This creates an IF-THEN-ELSE structure. To remove the ELSE condition un-check the false option or set the result register to itself.

> **IMPORTANT:** Read Rules can scale incoming Modbus register values to make Threshold rules easier to understand. If Read Rule scaling is not used, the values stored in Local Registers are the raw Modbus values from the server device. Display scaling and offset are not applied for these comparisons.



The Threshold rule is written as a sentence. Select the Local Register, operator, and value from the drop-down lists. Then define the value of the Local Register when the statement is true and not true.

Value/Register—When Value is selected, provide a fixed number in the following entry box. When Register is selected, provide a Local Register address in the following entry box. The contents of the Local Register will be used in the Threshold rule. When Same as Source is selected (else statement), the contents of the Local Register defined in the If comparison is used as the result.

Hysteresis—Optional parameter that is enabled only when values are non-zero. 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. Minimum On Time and Minimum Off Time are time-based parameters that govern how long a statement must be true or false to activate the output register.

On-Time—Stores in a Local Register how long (in minutes) the Threshold rule has been true. The On-Time value is shown in the Local Register in minutes but maintains the internal counter in seconds. This results in the accuracy of seconds, not rounding of the value stored in the Local Register.

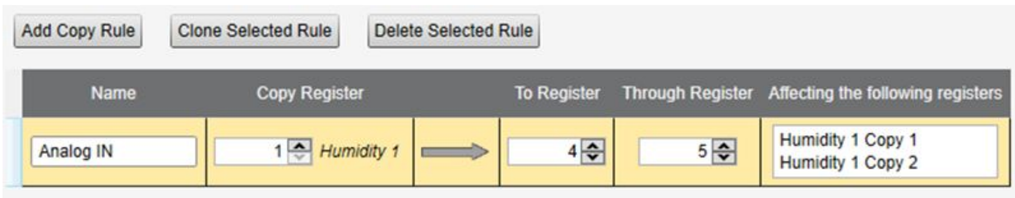
Logging Options—Threshold rules create events. These events can cause certain actions to occur, such as:
- Save threshold events to the cloud—An immediate push with the threshold register states is sent to the web server when this rule becomes active.
- Save threshold events to event log—Go to **Settings › Logging** to set up the event log.
- Push when active—Pushes the data to the cloud when the threshold rule becomes active. All local registers with the Cloud Reporting parameter ON will push to the cloud. (**Local Registers › Local Register Configuration**)
- After trigger, set source to 0—After a Push is sent as a result of the Threshold rule, the source Local Register is set to zero to indicate the Push data packet was sent.

Register Copy Rules

Use the **Register Copy** screen to copy a local register into another local register or range of registers.

> **IMPORTANT:** To move integer-based Local Registers 1–850 to floating point Local Registers, use the **Register Copy** rules. **Register Copy** automatically converts the integer value to a floating point value.

> **IMPORTANT:** Creating multiple copy rules using the same starting register will not be consistent. Instead create a daisy-chaining of registers. For example, copy register 1 to register 2, then copy register 2 to register 5, then copy register 5 to register 10.



Add Copy Rule—Click to create a new copy rule.

Name—Name your rule.

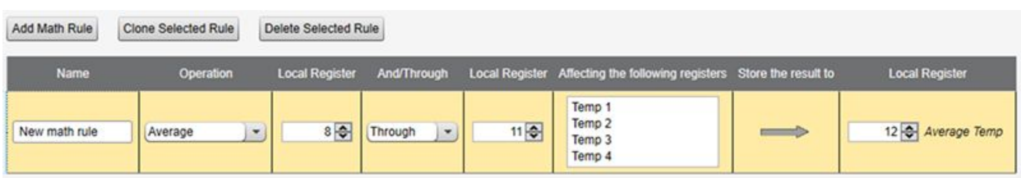
Copy Register—The starting register of the data source range of registers.

To Register—The starting register of the data target range of registers

Through Register—The ending register of the data target range of registers.

Math and Logic Rules

Math/Logic rules deal with 32-bit register logic with results from 0 to 4,294,967,295. In contrast, **Control Logic** rules are binary rules, with the results being either 0 or 1.



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. The Local Registers are unsigned 32-bit integers. All math and logic functions can operate on all 32-bits. The Local Register operations are:

- **Add**—Adds two local registers and stores the result in a Local Register.
- **Average**—Averages the values of multiple registers and stores the result.
- **Divide**—Divides local register 1 by local register 2 and stores the result. Dividing by zero results in a zero.
- **Logic NOT**—Performs a bit-wise one's complement on a Local Register and stores the result.
- **Logic OR, AND, NOR, NAND, XOR**—Performs bit-wise logic function on multiple registers and stores the result. Bit-wise logic functions operate on all 32-bits of the Local Registers. To perform logic function on a simple 0 or 1 in registers, use the Control Logic rules.
- **Multiply**—Multiplies one local register by another and stores the result.
- **Sum**—Adds multiple contiguous local registers and stores the result in a Local Register.
- **Subtract**—Subtracts Local Register 2 from Local Register 1 and stores the result. For negative numbers, the results are in two's complement form.

Control Logic Rules

Control Logic rules are binary rules, with the results being either 0 or 1. In contrast, the Math Logic rules deal with 32-bit register logic with results from 0 to 4,294,967,295.

Some Control Logic rules create memory elements using the DXM Local Registers as inputs and outputs. A value of zero in a Local Register is defined as 0 in the Control Logic rule, a non-zero value in a Local Register is defined as a 1 value. Control

Logic rules are evaluated when an input value changes state. Memory elements include S/R Flip/Flop, D Flip/Flop, T Flip/Flop and J-K Flip/Flop.

Control logic also works for floating point registers and virtual registers. In the case of floating point values, anything over 0.5 is considered logical 1.

Control Logic

Control Logic			
Function	**In 1**	**In 2**	**Q out**
AND	0	0	0
	1	0	0
	0	1	0
	1	1	1
OR	0	0	0
	1	0	1
	0	1	1
	1	1	1
XOR	0	0	0
	1	0	1
	0	1	1
	1	1	0
NOT	0	-	1
	1	-	0
NAND	0	0	1
	1	0	1
	0	1	1
	1	1	0
NOR	0	0	1
	1	0	0
	0	1	0
	1	1	0

NOTE: * Output Qn is an inverted output from the Q output and represents optional inputs or outputs. Leave the optional inputs or outputs connected to 0 when not used.

J-K Flip-Flop

If J and K are different, the output Q takes the value of J. The **Enable**, **Clock**, and **Qn** connections are not required for operation.

J-K Flip-Flop					
***Enable**	**J**	**K**	*** Clock**	**Q**	**Qn ***
0	X	X	X	No change	No change
1	X	X	0 or 1 or ↓	No change	No change
1	0	0	↑	No change	No change
1	1	0	↑	1	0
1	0	1	↑	0	1
1	1	1	↑	No change	No change

If an input is zero it is false (0), and if a value is non-zero (anything but zero) it is considered true (1). All but one requires two inputs; the NOT rule only has one input. Some of the Control Logic rules provide basic non-memory logic functions: AND, OR, XOR, NOT, NAND and NOR. These basic logic functions are based on zero and non-zero values and result in a 0 or 1.

Optionally, the rules can have a **Clock** input and an **Enable** input. The **Clock** input determines the time (0-1 rising edge) the rule is evaluated. The **Enable** input enables or disables the logic function.

- AND—Output is high (1) only if both inputs are non-zero
- OR—Output is high (1) if either input is non-zero
- XOR—Output is high (1) if either, but not both, inputs are non-zero
- NOT—Input is inverted to the output
- NAND—Output is low (0) if both inputs are high (> 0)
- NOR—Output is high (1) if both inputs are zero (0)

Set-Reset (S/R) Flip-Flop

The Set/Reset Flip-Flop, or S/R Latch, is the most basic memory device. The S input sets the output to 1, and the R input sets the output to 0. Selecting **Insert input** turns the inputs to low active, so a 0 on the S input set the Q output to 1. The truth table below is defined without the invert flag.

Set/Reset (S/R) Flip-Flop			
S	R	Q	Qn *
1	0	1	0
0	0	No change	No change
0	1	0	1
1	1	Invalid [1]	

Delay (D) Flip-Flop

The Delay (D) Flip-Flop takes the D input to the outputs at the next rising edge of the clock. All other states of clock have no effect on the outputs. The **Enable** input is optional. A **Clock** input and a **D** input changing at the same time results in an undetermined state.

Invert Inputs—Typically a input value of "1" will be evaluated as true (active), selecting "Invert inputs" will result in a "0" value to be considered true(active).

FlipFlop is R dominated—xxx

Table references:

- X = doesn't matter what the value is.
- <up arrow>, <down arrow> = signify a rising or falling edge of the clock input.
- * = optional input or output

Delay (D) Flip-Flop				
*Enable	D	Clock	Q	Qn *
0	X	X	No change	No change
1	X	0 or 1 or ↓	No change	No change
1	0	↑	0	1
1	1	↑	1	0

Toggle (T) Flip-Flop

The Toggle (T) Flip-Flop toggles the state of the output if the input is 1. The **Enable** input is optional. A **Clock** input and a **T** input changing at the same time results in an undetermined state.

[1] The Set input and Reset input active at the same time is an illegal condition and should not be used.

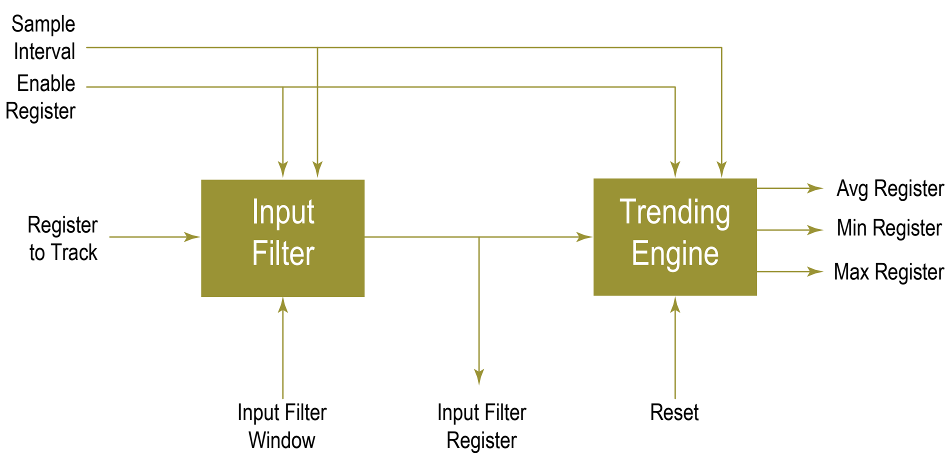
Toggle (T) Flip-Flop				
*Enable	T	Clock	Q	Qn *
0	X	X	No change	No change
1	X	0 or 1 or ↓	No change	No change
1	0	↑	0	1
1	1	↑	(toggle)	(toggle)

Trending and Filtering Rules

Use **Trending** rules to find the average, minimum, and maximum values for a specific register at a user-specified time interval. The Trend function collects data when the device boots up and accumulates the average, minimum, and maximum values for a register.

Local Registers locations to define include: Register to Track, Filter Register, Average Value Register, Minimum Value Register, Maximum Value Register, and Enable Register. When not using the Local Register definitions, leave them set to zero.

The filter and tracking is applied according to this flowchart.



The Trending rule has an optional filter input, and the filter output is the input to the Trending operation. The filter result is used before applying the Trending rule when you define the **Filter** parameter. The other Trending parameters are listed. The Average, Minimum, and Maximum register fields are optional and can be left set to zero when using the Filter output register.

Register to Track—Local Register to provide the input to the filter and Trending rule.

Sample Interval—Defines the time interval of the samples to collect from the Local Register.

Filter—Defines the specific filter algorithm to apply to the front end of the Trending rule.

Cumulative Average—Sums the samples and divides them by the number of samples.
Exponential Moving Average—A moving average filter that applies weighting factors that decrease exponentially. The weighting for each older sample decreases exponentially, never reaching zero.
Lulu Smoother—Takes the minimum and maximum values of mini sequences of sample points. The mini-sequence length is defined by the filter window parameter.
Median Average—Takes a least three sample points, sorts the sample points numerically, and selects the middle value as the result. A new sample point entered into the filter removes the oldest sample point.
Recursive Filter —Created from a percentage of the current sample plus a percentage of past samples. The number of samples is determined by the **Filter Window** parameter.
Simple Moving Average—Takes the number of samples defined by the **Filter Window** parameter and averages the samples for the result. When a new sample is taken, it becomes the latest sample point, and the oldest sample point is taken out of the filter averaging.

Weighted Moving Average—A moving average filter that applies a weighting factor to the data based on when the data was captured. More recent data has a higher weighted factor.

Filter Window—Defines the number of samples for a filter algorithm. For example, if the sample interval is 5 minutes and the Filter Window is 100, this will create a 100-point moving average over a 500-minute time period.

Filter Register—The Local Register that stores the output of the first stage filter. This is not required to connect the filter front-end to the Trending rule processing.

Average Value Register—The Local Register number to store the average value of the Trending rule. The average is calculated from the beginning of the power-up or from the Reset Trending time. (Hourly, Daily) Defining this Local Register is optional and can be left at zero.

Minimum Value Register—The Local Register number to store the Minimum value of the Trending rule. The Minimum is captured from the beginning of power-up or from the Reset Trending time. (Hourly, Daily) Defining this Local Register is optional and can be left at zero.

Maximum Value Register—The Local Register number to store the Maximum value of the Trending rule. The Maximum is captured from the beginning of power-up or from the Reset Trending time. (Hourly, Daily) Defining this Local Register is optional and can be left at zero.

Reset Trend Data—Use to reset the Trend data accumulation. (Hourly, Daily, or not at all)

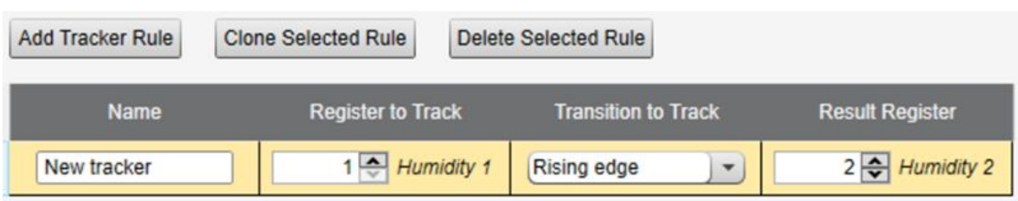
Enable Register—The Enable Register is an optional register and can be left set to zero. The Enable Register defines the address of a Local Register that can be used to turn on and off the trending function. Set the value of the Local Register defined to a 1 to turn on trending; set the value to zero to turn off trending. When disabled, the current data is held until the rule is enabled again.

Tracker Rules

Tracker rules monitor a Local Register and store the result of a function in another Local Register.

The possible functions are:

- Count the number of register transitions from 0 to non-zero value (rising edge). The speed at which it can count depends on how much work the DXM is performing. Typical tracker rates should be around 1 to 2 times per second.
- Track time in milliseconds the Local Register is in the non-zero state
- Track time in milliseconds the Local Register is in the zero state



Decoder Rules

Use the Decoders screen to copy a subset of the bits contained in a local register into another local register.

Add Decoder Rule
: Click to create a new decoder rule.

Name
: Name your rule.

Copy from Source Register
: The register to copy data from.

Number of bits
: The total number of bits (minimum of 1, maximum of 32) to copy from the source register.

The **Number of bits** field auto-decrements when either of the **Starting at bit index** fields is increased to ensure that the number of bits copied does not overflow the maximum bit index of 31.

Starting at bit index
The specified number of bits will be copied from the source register ascending from the specified index (minimum of 0, maximum of 31). Starting at bit 0 will capture the lowest-order portion of the register's value, while starting at index 31 will copy only the highest-order bit.

Destination Register
The register to copy data to.

Starting at bit index
The lowest-order bit of the data copied from the source register will occupy this bit index in the destination register.

On the DXM, these rules are executed by performing a bit shift, applying a mask, and writing the result to the destination register.

Register Mapping

Use the **Register Mapping** screens to configure read rules and write rules for Modbus RTU, Modbus TCP, and CAN/J1939.

Read and write rules allow the user to program the ability to read or write information from internal or external Modbus servers to/from the local registers. On the DXM, use the read/write rules to access the Modbus registers of the LCD display, I/O base board, and the internal ISM radio.

The **Register Mapping** screens handle all direct register mapping. DXM Configuration Software allows the user to create Write/Read rules that in turn create Modbus messages to external devices. How the user enters rules affects how a Modbus message is formed. If the user creates three individual read or write rules, those rules create three individual Modbus messages that will be sent out of the RS-485 client port. If the user creates one read or write rule that spans multiple registers, the result is one Modbus message.

Click on the arrow of the read or write rules to show all the parameters for that rule.

RTU

Use the **Register Mapping › RTU** screen to define **RTU Read** and **RTU Write** rules. The **Read Rules** or **Write Rules** interact with the Local Registers to exchange data with external Modbus devices.

For screens that contain tables with rows, click on any row to select it. Then click **Clone** or **Delete** to copy/paste or remove that row.

When DXM-R90x is the selected DXM model, an additional **RTU Configuration** screen is displayed before **RTU Read** and **RTU Write**. This screen allows each of the DXM-R90x's five UART buses to be configured separately with distinct baud, parity, timeout, and message delay settings.

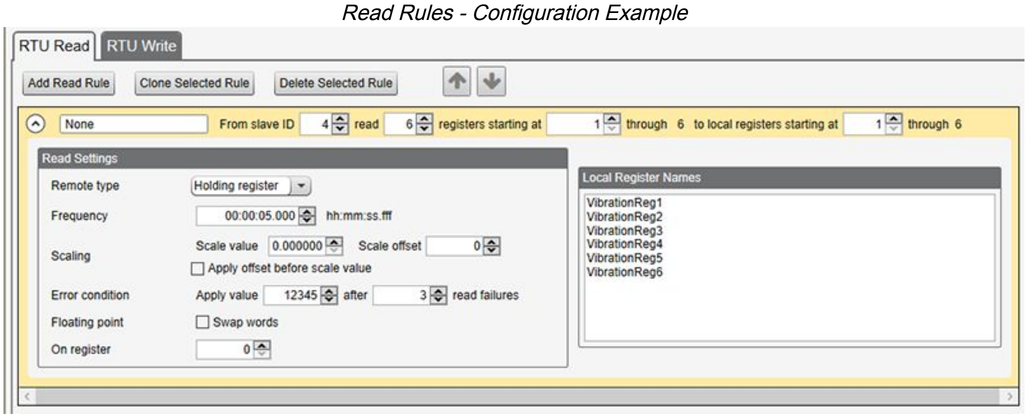
When a model DXMR90-X1, DXMR90-4M, or DXM1200-X2 is the selected DXM model, an additional RTU Configuration screen is displayed before **RTU Read** and **RTU Write**. This screen allows each of the UART buses of that device to be configured separately with distinct baud, parity, timeout, and message delay settings.

Create an RTU Read Rule

Follow these steps to create a new read rule.

This example screen shows a read rule created to read six registers (address 1 through 6), from Modbus ID 4. The results are stored in the Local Registers 1 through 6.

Read Rules - Configuration Example



1. From the **Register Mapping › RTU › RTU Read** screen, click **Add Read Rule**.
2. Click the arrow next to the name to display the parameters.

3. Name your rule.
4. Select the device ID.
5. Select how many registers to read, and the beginning register.
6. Define the register type, how often to read the register, and any other appropriate parameters.
7. If necessary, select the error condition. For this example, if the read function fails after three attempts, the read rule writes 12345 to the DXM local registers. Notice the list of local register names this read rule is using.

Read Rules

Each read rule defines a Modbus ID and register range to read and then store in the Local Registers. The Local Register names shown are the registers that are being used by the read rule.

When a model DXMR90-X1, DXMR90-4M, or DXM1200-X2 is the selected DXM model, each RTU rule created on the **RTU Read** and **RTU Write** screens contain a **Port** drop-down field that specifies which of the UART buses the rule should use for that device.

> **IMPORTANT:** Setting Read Rules or Write Rules to fast rates may cause other processes, ScriptBasic, LCD display, USB port accesses or API calls to be delayed or rejected.

The user defines parameters that can be applied to each read rule.

Remote Type—Select the register type from the drop-down list.

Read Holding register–16-bit register access, Modbus function 3; all Banner wireless devices use Holding Registers
Read Coil–Single bit access, Modbus function 1
Input Register–16-bit register access to read holding registers, Modbus function 4
Read Discrete Input–Single bit access, Modbus function 2

Frequency—Defines the cyclical rate at which the register is read. The maximum cyclic poll rate is controlled by the maximum polling rate set in the **Settings › General** screen, the **Modbus Client Communications** section. Changing that value causes the software to adjust all polling rates. For example, setting the maximum rate to 1 second causes the software to adjust all rules with a frequency of less than 1 second up to 1 second.

Offset and Scaling—Scale and Offset are applied to the Modbus data before writing to the Local Registers. The Scale value is multiplied followed by adding of the offset value. Dividing can be accomplished by using scale values less than 1. Subtracting the offset value can be accomplished by using a negative offset value. The result is held in the Local Register.
Apply Offset before Scale Value—The execution order for scaling values is changed to first add the offset value then multiply scale value. The result is held in the Local Register.

Error Conditions—Applies a default value to the Local Register after a user-defined number of Modbus register read fails.

Floating Point - Swap Words—A floating point value is a 32-bit value requiring two consecutive Modbus register reads or writes. The DXM expects the most significant part of the floating point value to be first (lowest address) followed by the least significant part. If a Modbus server device sends the least significant part first, select **Swap Words** to align the words correctly.

On Register—The On Register is the address of a Local Register used to enable the Read rule. If the On Register is zero, it is not used. If the On Register is specified, the data value of the Local Register controls the executions of the Read rule. When the value of the Local Register is zero, the Read Rule is idle. When the value of the Local Register is non-zero, it will continuously execute the Read Rule as fast as it can. This speed is governed the number of Read/Write rules defined and by the Modbus Client Communications parameter Maximum Polling Rate (default 50 ms). This parameter is located on the **Settings › General** screen. The Frequency interval is not followed when using the **On Register** feature.

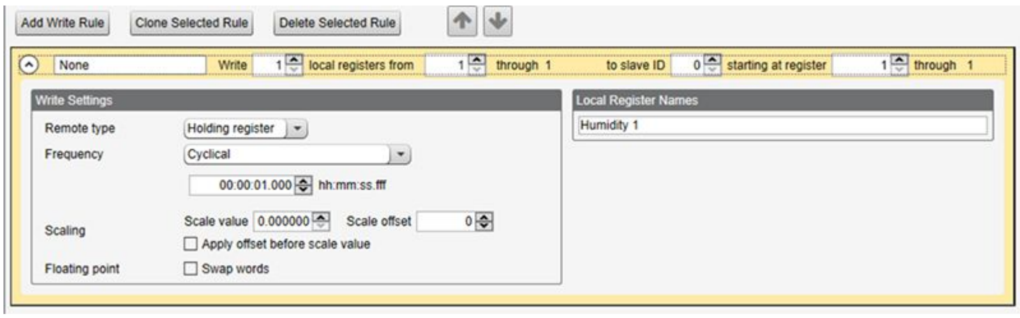
Server IDs

DXM ISM Gateway radio: 1
DXM I/O Board: 200
DXM LCD: 201
MultiHop Remote server radios: 11 through 60

Write Rules

The write rules write Local Register data to the defined Modbus ID and registers. The Local Register names shown are the registers used by the write rule.

> **IMPORTANT:** Setting Read Rules or Write Rules to fast rates may cause other processes, ScriptBasic, LCD display, USB port accesses or API calls to be delayed or rejected.



The user-defined parameters that can be applied to each Write rule are:

Remote Type—Defines the type of Modbus functions to use

Write Holding register, Modbus function 16; all Banner wireless devices use Holding Registers
Write Multiple Coil, Modbus function 15
Write Single Holding register, Modbus function 6

Frequency—Defines how often to write the Local Register to the Modbus device in one of two ways:

Cyclical—Causes a Modbus write based on a timing interval, as fast as possible or a specified time interval.
On Change of Local Register Data—Allows the user to specify certain criteria when to write to a Modbus server device. For example, the write occurs if the Local Register changes by a user-specified amount. If the user wants to write the Local Register to the Modbus server at a minimum interval, use the **write AT LEAST** time setting. Use the **write AT MOST** time interval to minimize the write cycles for Local Registers that change frequently.

The maximum cyclic poll rate is controlled by the maximum polling rate set in the **Settings › General** screen, the **Modbus Client Communications** section. Changing that value causes the configuration software to adjust all polling rates. For example, setting the maximum rate to 1 second causes the software to adjust all rules with a frequency of less than 1 second up to 1 second.

Offset and Scaling—Scale and Offset are applied to the Local Register data as it is sent in the Modbus write command. Local Register data is not changed in the process. The Scale value is multiplied followed by adding of the offset value. Dividing can be accomplished by using scale values less than 1. Subtracting the offset value can be accomplished by using a negative offset value. The result is held in the Local Register. **Apply Offset before Scale Value**—The execution order for scaling values is changed to first add the offset value then multiply scale value. The result is held in the Local Register.

Please note that when using **Offset and Scaling** within an **RTU Read Rule**, the decimals are truncated. For example, if a raw value is 99 and the scale value is 0.1, then a value of 9 arrives in the local register instead of 9.9.

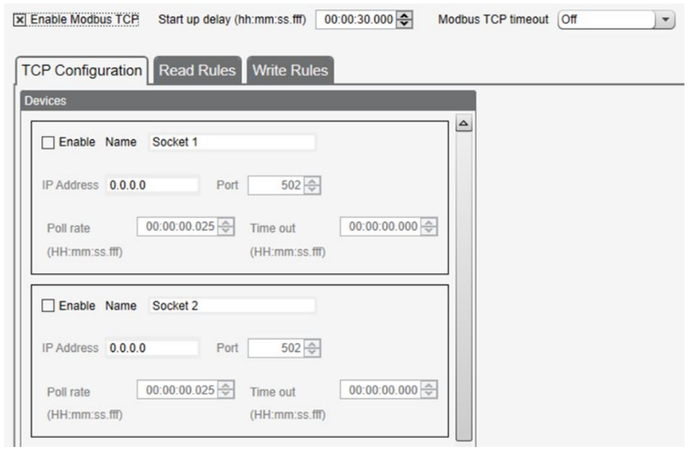
Floating Point - Swap Words—A floating point value is a 32-bit value requiring two consecutive Modbus register reads or writes. The DXM expects the most significant part of the floating point value to be first (lowest address) followed by the least significant part. If a Modbus server device sends the least significant part first, select **Swap Words** to align the words correctly.

Modbus TCP and Configuration

Use the **Register Mapping › Modbus TCP** screen to define Ethernet communication rules for the DXM to access remote register data from Modbus TCP servers. First define devices under the **TCP Configuration** tab, then define the Read/Write rules for the devices.

At the top of the window is an **Enable Modbus TCP** check box. Select this box to globally enable Modbus over TCP. The Start up delay defines the amount of time between the DXM power on and the execution of the Modbus TCP Read/Write rules.

The **Modbus TCP** screen defines



To use Modbus TCP, follow these steps:

1. Select **Enable Modbus TCP**.
2. Enable a **Socket** connection.
3. Configure the settings (IP Address, Port, Poll rate, and Time out).
4. Create and enable a rule.
5. Link the rule to an enabled Socket.

Modbus/TCP Parameters

The Modbus TCP interface parameters are:

Enable Modbus TCP
Globally enables or disables Modbus TCP

Start up Delay
Defines the time delay between boot up and the execution of the rules

Modbus TCP Timeout
Denotes the amount of time to wait before closing an inactive TCP connection

Ethernet Socket Parameters

The individual Ethernet socket parameters are:

Enable
Must be selected to open the socket connection to the Modbus TCP server

IP Address
Defines the IP address of the Modbus TCP server. The format is xxx.xxx.xxx.xxx

Port
Defines the port to be used for the Modbus TPC protocol. The default is 502.

Poll Rate
Defined how often to execute the write and read rules defined for this socket connection

Time out
How long the processor waits for an acknowledgement from the Modbus server before going on to the next rule

If Modbus TCP is disabled, this section is not copied into the final XML configuration file. If the Modbus TCP rules do not point to an enabled device, the rules are not written into the final XML configuration file.

CAN / J1939

CAN/J1939 is a wired communications protocol standard maintained by the Society of Automotive Engineers (SAE). Commonly used in off-road and heavy equipment, J1939 is a higher level protocol that is built upon the Controller Area Network (CAN) physical layer developed and originally published by Robert Bosch GmbH in 1991.

All J1939 messages are sent using the 29 bit message identifier, also known as the extended frame format, described in CAN 2.0B.

Typical J1939 data frames consist of the 29 bit identifier, 8 byte payload, a 15 bit CRC, and 20 additional bits used as flags, delimiters, or simply reserved for future use. Therefore a typical CAN packet is approximately 29 + 8 × 8 + 15 + 20 = 128 bits long. At the standard rate of 250 kbits/sec, a J1939 packet is a little over 500 µs long. Actual duration may vary because the protocol defines that "bit stuffing" shall occur any time there are 5 bits in a row of identical polarity.

There is no centralized network coordination function in a J1939 network. Every node (known as an Electronic Control Unit, or EC) in a J1939 network "claims" a unique address for itself during network commissioning (after power-up). All ECs on the bus can be producers, consumers, or both producers and consumers of data. When an EC has data that it wants to write to the bus, it begins transmitting while simultaneously monitoring the channel for collisions. Collisions can be detected immediately because each bit is sent using a dominant/recessive scheme and any EC that sees a dominant bit on the channel when it transmitted recessive bit will know to immediately cease the transmission. This is known as non-destructive bus arbitration.

Note that a minimum of two nodes must be used to initialize communication on a CAN bus. Since a transmitted message must be acknowledged in the ACK bit by a receiver, the transmitting controller sends out an error flag if the message is not properly ACKed.

Messages in J1939 are organized by a field in the 29-bit header known as the Parameter Group Number or PGN. The PGN may be thought of as an index into a large data table. The J1939 standard defines what the data means in every such table entry, but also leaves many PGNs to be defined in a proprietary manner.

For example, the PGN 65262 is defined to contain Engine Temperature. Within the 8 bytes sent with this PGN, the J1939 specification also defines that byte 1 is for Engine Coolant Temperature, byte 2 contains Engine Fuel Temperature 1, bytes 3-4 encode Engine Oil Temperature 1, bytes 5-6 encode Engine Turbocharger Oil Temperature, byte 7 represents Engine Intercooler Temperature, and byte 8 is for Engine Intercooler Thermostat Opening.

Theory of Operation

The Banner DXM functions as a protocol converter between its internal local register space and J1939 PGNs.

Users must configure the DXM to select the J1939 PGNs to be monitored, which is known as creating a J1939 Read Rule. Similarly, the DXM can be configured to transmit messages if desired, which is known as creating a J1939 Write Rule. Read Rule and Write Rules map out the link between local registers and PGNs.

Data contained in DXM local registers may be accessed using the scripting engine in the DXM.

At startup, the DXM attempts to claim a unique J1939 address on the CAN bus it is connected to. It does so by sending its desired address to the PGN 0xEE00 and waiting for a reply. The DXM broadcasts the contents of its NAME field, which is specified in the DXM Configuration Software using the J1939 Name fields. All sub-fields are currently user selectable except the Manufacturer ID.

The DXM allows two separate methods for specifying the address to claim on a J1939 network. Either one may be used; the choice of the method can be determined by whichever one is more convenient. In the first method, a list of acceptable addresses is specified. The DXM attempts to claim the first (leftmost) address, and if denied, will proceed to the next and the next. In the second method, a range of acceptable addresses is specified and the DXM attempts to claim the lowest, then increment that choice by 1 until it is able to claim one or reaches the end of the list.

No further J1939 functions (transmitting or receiving) begin until the DXM claims its address. In the DXM Configuration Software, go to the **Register Mapping › J1939** screen and click **Read device J1939 address and name** to view the current DXM device address. If it returns –1, it was unsuccessful in claiming an address.

After the DXM has claimed an address, it passively listens to all J1939 traffic. Any messages that contain PGNs for which there is a matching read rule are decoded and the data is copied into the desired local registers. Any PGNs in the J1939 Write Rules area are transmitted onto the bus at the interval chosen. Each time a J1939 message is about to be transmitted, the DXM populates the J1939 data fields with the contents of the specified local registers, then writes the message onto the bus.

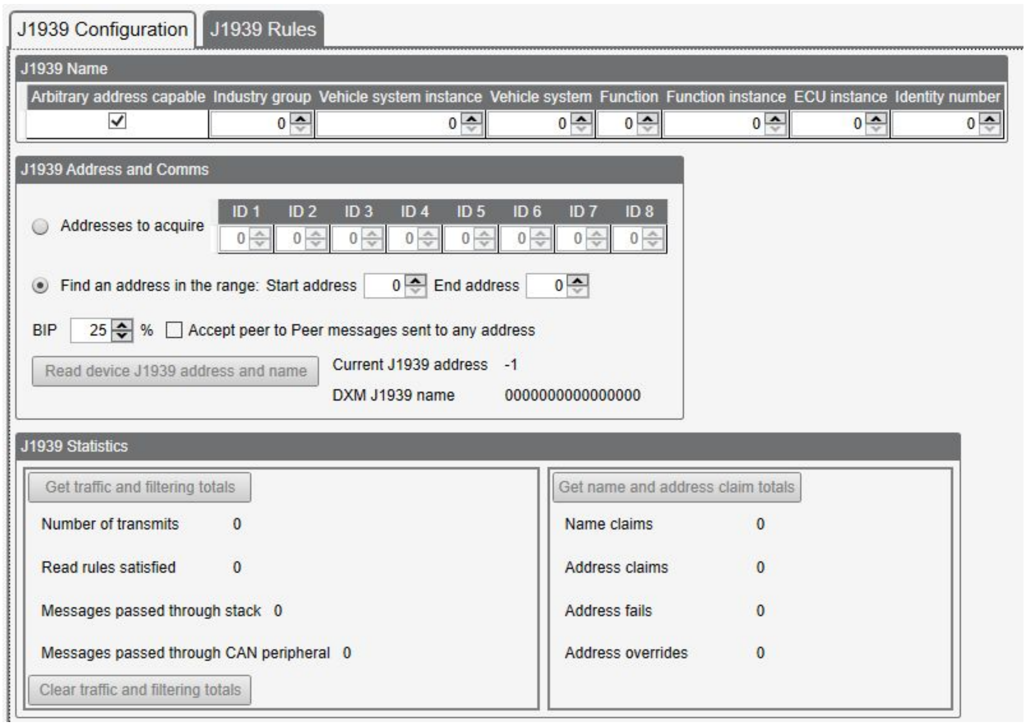
The acceptance criteria for J1939 Read Rules are:

1. The CAN protocol version must pass. This should always be the case in a J1939 network, but may have to be revisited if this device is used for other CAN bus applications.
2. The 16 bits that make up the arriving PGN are examined. Specifically the PGN is considered to be the concatenation of the 8 bit PDU Format and the 8 bit PDU specific fields.
    a. If the PGN ≥ 0xF000, then the message destination is global and the PGN is left unaltered as it is handed to the next layer of the internal stack. The implication of this is that read rules for PGNs in this range should have the PGN in the rule set to exactly what the device documentation says.
    b. If the PGN < 0xF000, then it is a peer-to-peer message and the PGN is considered to be the 8-bit PDU format bits followed by 8 bits of 0. Put another way, PGN = (PGN & 0xFF00), where & represents a bitwise logical AND operation.
        i. Read rules for PGNs in this range should have the PGN in the rule set to mask off the lower 8 bits to zero.

ii. In this case, the lower 8 bits (PDU specific bits) are the destination address. Destination address = PGN & 0x00FF.
iii. If the address claimed by the DXM does not match the message's intended destination address, the message is discarded. To prevent messages from being discarded this way, select **Accept peer to peer messages sent to any address** in the DXM Configuration Software.

Start-Up Guide for the DXM

Verify your DXM is powered up and connected to a computer that has the DXM Configuration Software installed.



1. On the DXM: Make the wired connection to the bus. Using the labels on the housing decal, CH (pin 16) is for CANH and CL (pin 15) is for CANL.
2. On your computer: Launch the DXM Configuration Software and navigate to the **Register Mapping › J1939** screen.
3. Configure the **J1939 Name** for the DXM.

   All sub-fields of the NAME area are user-configurable except the Manufacturer Code, which is permanently set to 758 (decimal) as granted to Banner Engineering by SAE.
4. Specify the preferred addresses for the DXM to claim using one of two methods:
   - Create an explicit list of addresses in the **ID1** through **ID8** fields. The DXM attempts to claim the address under **ID1** first. If that address is already taken, it attempts to claim the address under **ID2** and continues until it claims an address.
   - List the **Start address** and the **End address**. The DXM attempts to claim the **Start address** first and increment sequentially until it successfully claims or reaches the **End address**.
5. **BIP** may be left at 25%. This field is the "babbling idiot protection" and establishes is the maximum fraction of the J1939 channel bandwidth that the internal stack allows the DXM to consume with data transmissions. It is a safeguard to protect the user from specifying too many or too frequent J1939 transmissions that could cause bus collisions and performance issues.
6. Select **Accept Peer to Peer messages sent to any Address** to allow the DXM to listen in on peer-to-peer messages that aren't directed to its own bus address.

   Even though most messages on the J1939 bus are sent to a broadcast address, some peer-to-peer messaging is possible.

Create a J1939 Read Rule

Create a J1939 read rule to configure the J1939 to monitor for desired Parameter Group Numbers (PGNs) and capture their data.

These instructions assume the DXM is powered up and connected to a computer running the DXM Configuration Software.

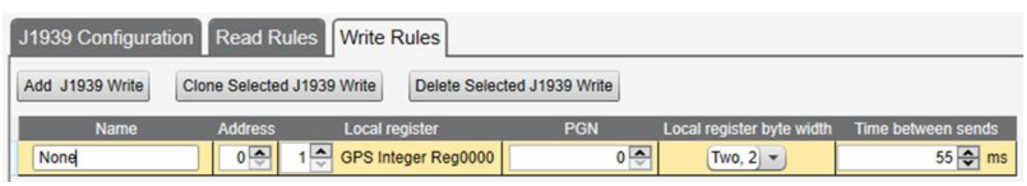
1. From the **Register Mapping › J1939** screen, click **Add J1939 Read**.
2. Enter a name for the rule.
   This name does not affect the actual behavior of the system.
3. Enter a **Source Address**.
   - If the bus address of the EC sending the PGN is known, enter that address as the **Source Address**.
   - To accept messages from any source, enter the wildcard value of 255 as the **Source Address**.

   Every J1939 message includes a **Source Address**, which is the bus address of the device that sent the message. The DXM filters incoming messages based on this address.
4. Enter a **Local Register**.
   The **Local Register** is the base address in local register memory space that the data should be sent to. When creating multiple read rules, choose to base addresses spaced so that the same local register does not get written to by multiple read rules.
5. Enter a **PGN**.
   This is the parameter group number (PGN) of the J1939 message that this rule captures.
   - If PGN ≥ 0xF000 (61440), the PGN is a broadcast rule. The message arrives and passes through the stack as long as the PGN matches the read rule.
   - If PGN < 0xF000 (61440), it is a peer-to-peer message. To be received, the lower 8 bits of the PGN must match the bus address claimed by the DXM. (This check can be overridden by selecting **Accept peer to peer messages sent to any address**.) Because the lower 8 bits of the PGN (also know as the PDU Specific) are used as an address, the PGN in the read rule should have its lower 8 bits zeroed out to make the matching work.
6. Enter the **Local register byte width**.
   Local registers are 32 bits, so it is possible to pack 1, 2, or 4 bytes of J1939 data into each local register. For J1939 messages with 8 bytes of data (the majority), packing means the data consumes 8, 4, or 2 local registers respectively depending on the choice.

Create a J1939 Write Rule

Create a J1939 write rule to transmit local register data in a J1939 message at a periodic interval.

These instructions assume the DXM is powered up and connected to a computer running the DXM Configuration Software.



1. From the **Register Mapping › J1939** screen, click **Add J1939 Write**.
2. Enter a name for the rule.
   This name does not affect the actual behavior of the system.
3. Enter a destination **Address** for the message.
   The broadcast address is 255.
4. Enter the **Local register**.
   The **Local register** is the data source's base address in local register memory space.
5. Enter the **PGN**.
   Only PGNs in the range 0xFF00 (65280) to 0xFFFF (65535) are able to be specified by the manufacturer. Use caution when assigning data to PGNs outside of this range, because the data must adhere to the formatting specified by the Society of Automotive Engineers or risk being misinterpreted by other ECs on the bus.
6. Enter the **Local register byte width**.

Local registers are 32 bits wide, so it is possible to retrieve 1, 2, or 4 bytes of J1939 data from each local register. Because most J1939 messages have 8 bytes of data, the data is sourced from 8, 4, or 2 local registers, respectively, depending on the choice.

Example: Receive a PGN Using Broadcast Communication

A manufacturer's datasheet for an EC states it transmits its information on PGN 0xFF01 (65281).

Because this PGN is in the range 0xFF00–0xFFFF (65280–65535), the data is manufacturer-specific and not constrained by a definition controlled by the SAE.

1. On the **Register Mapping › J1939** screen, click **Add J1939 Read** to create a new read rule.
2. Enter a **Name**, for example, `Proprietary_msg1`.
3. Enter the **Source address** (the sender's bus address) or enter `255` to accept data from any address on the bus.
4. Enter the **Local register** (the base address for the data).
5. Enter a **PGN** of `65281`.
6. Enter the **Local register byte width** (1, 2, or 4), which stores the bytes per local 32-bit register.

The following example J1939 message shows the relation between byte order on the bus and in the local register space:

	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
J1939 Data	0x01	0x23	0x45	0x67	0x89	0xAB	0xCD	0xEF
Reg. Address	Base	Base +1	Base +2	Base +3	Base +4	Base +5	Base +6	Base +7
1 byte/reg	0xEF	0xCD	0xAB	0x89	0x67	0x45	0x23	0x01
2 bytes/reg	0xEFCD	0xAB89	0x6745	0x2301	(n/a)	(n/a)	(n/a)	(n/a)
4 bytes/reg	0xEFCDAB89	0x67452301	(n/a)	(n/a)	(n/a)	(n/a)	(n/a)	(n/a)

Example: Receive a PGN Sent Using Peer-to-Peer Communication

A manufacturer's datasheet for a sensor says it transmits its information on PGN 0xCBF4 (52212).

Because this PGN is in the range 0x0000–0xEF00, (0–61184) this is meant to be a peer-to-peer message. The lower 8 bits of the PGN contain the destination address.

1. On the **Register Mapping › J1939** screen, click **Add J1939 Read** to create a new read rule.
2. Enter a **Name**, for example, `p2p_msg2`.
3. Enter the **Source address** (the sender's bus address) or enter `255` to accept data from any address on the bus.
4. Enter the **Local register** (the base address for the data).
5. Enter a **PGN** of `51968` or `0xCB00`. Note that the lower 8 bits have been masked.
6. Enter the **Local register byte width** (1, 2, or 4) bytes per local register.
7. Enter the Address claim. As a peer-to-peer message, this PGN is intended for the device at address 0xF4 (244) only. There are two ways to allow the DXM to read this message:
   - Have the DXM claim address 0xF4 (244) on the bus; or
   - Select **Accept peer to peer messages sent to any address**

Example: Monitor Diagnostic Troubleshooting Codes

The J1939 standard specifies a method of transmitting diagnostic trouble codes. Active trouble codes are sent using PGN 65226 or 0xFECA (consult the J1939 documentation for other related PGNs, such as Previously Active Trouble Codes).

1. On the **Register Mapping › J1939** screen, click **Add J1939 Read** to create a new read rule.
2. Enter a **Name**, for example, `active_DTCs`.
3. Enter the **Source address** or `255` to accept data from any address on the bus.
4. Enter the **Local register** (the base address for the data).
5. Enter a **PGN** of `65226` or `0xFECA`.
6. Enter the **Local register byte width** (1, 2, or 4) bytes per local register.

Each diagnostic trouble code consumes 8 bytes, but there may be more than one active code from a given EC. Such longer messages are sent using the Transport Protocol feature of J1939, which the DXM supports. It is important that the user reserve enough local register space for the messages in case there are more than the standard 8 bytes.

Example: Transmit a PGN Using Broadcast Communication

To transmit manufacturer-specific data on PGN 65296 or 0xFF10, follow these steps.

1. On the **Register Mapping › J1939** screen, click **Add J1939 Write** to create a new write rule.
2. Enter a **Name**, for example, `tx_msg2`.
3. Enter the **Address** (the destination address for the message, for example 49d=0x31).
4. Enter the **Local Register** (the base address to retrieve the data from).
5. Enter a **PGN** of `61184`.
6. Enter the **Local register byte width** (1, 2, or 4) per local register.
7. Enter a **Time between sends** of `1500` ms.

If this message is observed as a standard CAN bus message, the 29-bit arbitration ID appears as follows: 0x1CEF3116.

- The leading 1C is the priority, which has been set to the lowest, or 8.
- EF is the higher 8 bits of the PGN.
- 31 is the destination address.
- 16 is the address that this DXM has claimed on the bus.

Example: Transmit a PGN Using Peer-to-Peer Communication

To transmit manufacturer-specific data on PGN 61184 or 0xEF00, follow these steps.

1. On the **Register Mapping › J1939** screen, click **Add J1939 Write** to create a new write rule.
2. Enter a **Name**, for example, `tx_msg2`.
3. Enter the **Address**.
4. Enter the **Local register** (the base address from which to retrieve the data).
5. Enter a **PGN** of `65296`.
6. Enter the **Local register byte width** (1, 2, or 4) per local register.
7. Enter a **Time between sends** of `1500` ms.

Send Diagnostic Commands

The DXM firmware supports some diagnostic commands that can be entered through the DXM Configuration Software.

Use the buttons on the **J1939 Configuration** screen to send the following commands:

- Read device J1939 address and name
- Get traffic and filtering totals
- Clear traffic and filtering totals
- Get name and address claim totals

To manually send these commands, follow these instructions.

1. Go to the **Device › View Serial Activity** menu.

   A **Traffic on port COMxx** window opens.
2. In the single-line text box at the bottom of the window, type a command. Click **SEND**.

   The resulting serial traffic, including response, will be shown. The successful execution of CMD00xx always results in a response of RSP00xx, followed immediately by one or more parameters (xx indicates the Command Number).

Command (case sensitive, no spaces)	Command Name and Meaning	Sample Response	Fields in Response (after RSP00xx, where xx = confirmation of command number
CMD0080	Address claimed	RSP008022	Decimal value of Address claimed by the DXM. For this example, 22.
CMD0081	NAME field	RSP00815A5ADA5EFFFFB4DA	ASCII string of the DXM's NAME field in hex. The NAME is eight bytes long, so the field will contain 16 hexadecimal digits.

Continued on page 48

Continued from page 47

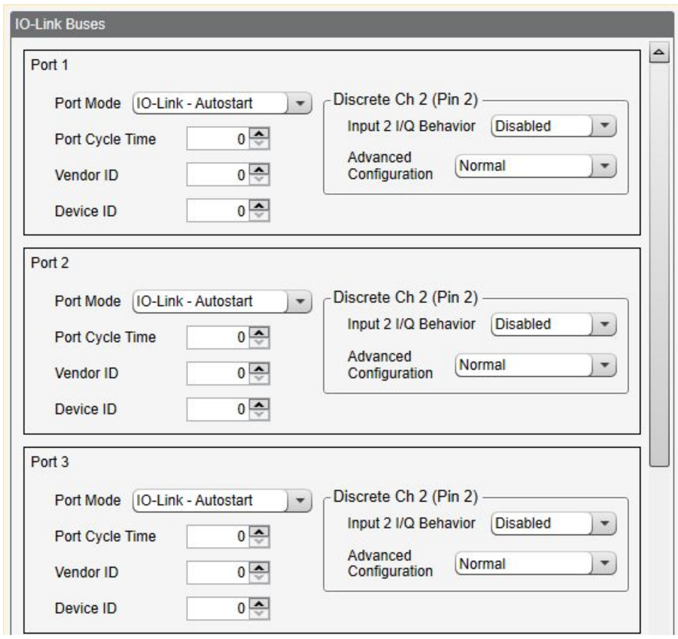
Command (case sensitive, no spaces)	Command Name and Meaning	Sample Response	Fields in Response (after RSP00xx, where xx = confirmation of command number
CMD0082	Traffic and filtering	RSP00820,2,3,4	Totalizing counter. Number of Transmits (0), number of Read Rules satisfied (2), number of messages passed through stack (3), and number of messages passed through CAN peripheral of micro (4)
CMD0083	Statistics	RSP00830,1,2,3	Statistics on name and address claiming: Name, Address, Address fails, overrides
CMD0084	Clear traffic totalizer	RSP00840	Clears the totalizer values (traffic counting) that are accessed by CMD0082

IO-Link Screen

IO-Link behavior is supported by the **DXMR90-4K** and **DXMR110-8K** models only.

For the DXMR90-4K model, the **IO-Link** screen is found under **Register Mapping**. For the DXMR110-8K, the **IO-Link** screen appears directly on the left side navigation because the **DXMR110-8K** doesn't support any other register mapping.

IO-Link screen



IO-Link Port/Operating Modes

The operating mode can be configured for any port on the IO-Link master. The following modes can be used:

Deactivated
: Use deactivated mode for any unused IO-link master ports if a device is not connected.

IO-Link Manual
: The IO-Link master only connects IO-Link devices that have a certain vendor ID and device ID (1: IOL_MANUAL).
: This mode is also required to for the Backup and Restore or Restore function.

IO-Link Autostart
: The IO-Link master connects to every connected IO-Link device (2: IOL_AUTOSTART).

Digital Input
: The IO-Link port functions as a standard digital input (3: DI_C/Q).

Digital Output
: The IO-Link port functions as a standard digital output (4: DO_C/Q).

Settings

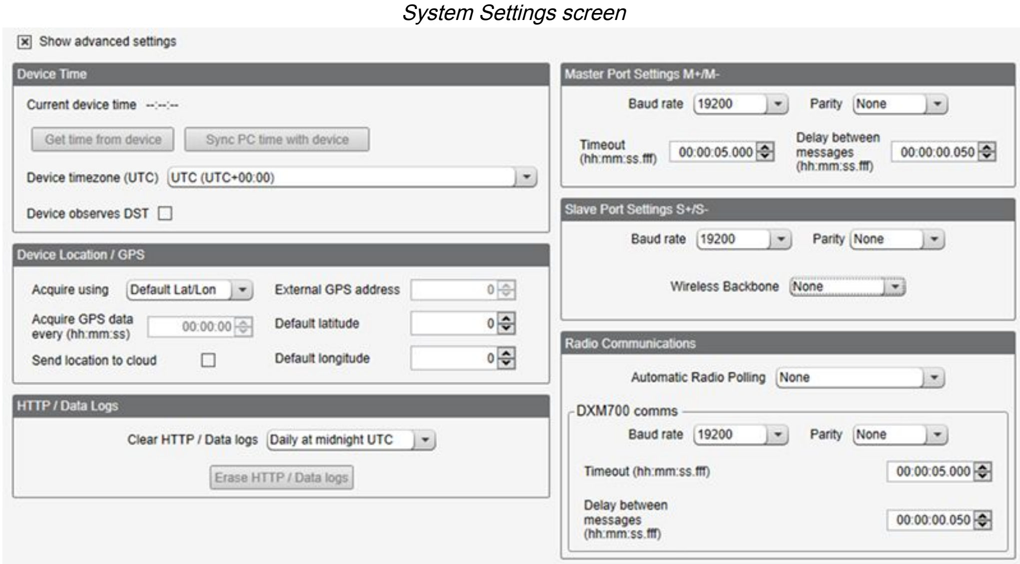
Use the **Settings** screen to define general parameters, cloud services settings, logging parameters, Script Basic settings, network settings, administration parameters, notes, and I/O board parameters.

In the upper right of the **Settings** screen is a **Show advanced settings** checkbox. By default, this box is not selected, hiding advanced configuration features. Screens with advanced configuration settings are: **System**, **Cloud Services**, **Ethernet**, and **Scripting**.

System Settings

Use **Settings › System** to define the time zone, Modbus Client serial interface, Modbus Server serial interface parameters, and to manually enter your GPS coordinates.

System Settings screen



Device Time
: Sets the time zone offset and displays the current time on the device or sync time with the PC.

Device Location/GPS
: Defines the latitude and longitude for the device. The GPS data can be entered manually for fixed assets or by defining an external GPS unit connected to the client Modbus RS-485 bus. Select **Send Location to Cloud** to report this information to the Web site.

HTTP / Data Logs
: Define when and how often to clear the HTTP and data logs.

Client/ServerPort Settings
: Defines the Modbus client and server serial port settings on the device (main RS-485 port).

The default communications settings are 19200 baud with no parity for both the client and server ports, and 5s timeout with 5ms delay between messages for the client port. When changing the communications settings on the DXM, any devices attached to the bus also need to be changed, including the internal ISM radio.

The ISM radio communication parameters should be changed first using the User Configuration Software for DX80/ Performance radios or the MultiHop Configuration Software for MultiHop radios.

Select the **Wireless Backbone** type from the drop-down list.

None—No wireless backbone behavior.
Modbus—Enables Modbus server port to come through a MultiHop HE5 module plugged into the processor board.
Ethernet—Allows DXM to act as either a client or server device when a wireless ethernet backbone module is plugged into the processor board. Selecting Ethernet causes two additional parameters to display: IP address and Subnet mask. These parameters apply only to the DXM's wireless Ethernet backbone configuration, not to its hardwired Ethernet connection. While a DXM-R90x is connected and Ethernet wireless backbone mode is selected, select either **Set DXM as Ethernet Client** or **Set DXM as Ethernet Server** to assign that DXM the respective role within the wireless ethernet backbone network.

Radio Communications

Automatic Radio Polling continually polls the embedded DX80 ISM radio for data changes. Using automatic polling is optimal for data throughput from the radio, but monopolizes the bandwidth on the client RS-485 bus. If other resources are expected to share the client RS-485 bus, it may be better to access radio data using Register Mapping. Using **Automatic Radio Polling** allocates DXM Local Registers 1–768 for the DX80 ISM radio devices (the Gateway and Nodes 1–47).

The processor organizes the data in DXM **Local Registers** based on the user selections defined below.

None
: Disables automatic radio polling.
: The user must use Register Mapping or ScriptBasic to transfer register data with the ISM radio.

Group by Device - Inputs only
: Only input data from the radio is collected and stored in the DXM Local Registers. The data is organized (grouped) by the device.
: The first 16 registers (1–16) are the Gateway registers. Node 1 uses Local Registers 17–32, Node 2 uses Local Registers 33–48 and so on.
: See the table for more details.

Group by Device - Inputs/Outputs
: Input data from the radio is collected and stored in the DXM Local Registers. Local Registers defined as the outputs for devices are automatically written to the ISM radio when the contents change. The data is organized (grouped) by the device.
: The first 16 registers (1–16) are the Gateway registers. Node 1 uses Local Registers 17–32, Node 2 uses Local Registers 33–48 and so on.
: See the table for more details.

Group by In/Out - Inputs only
: Only input data from the radio is collected and stored in the DXM Local Registers. The data is organized (grouped) by each input.
: For example, all input 1 data is stored in the first 48 Local Registers (1–48), Gateway, Node 1–47). The next 48 registers (49–96) store input 2 data for all devices.
: Local Registers 1–384 store inputs 1–8, Local Registers 385–768 store outputs 1–8.
: See the table for more details.

Group by In/Out - Inputs/Outputs
: Input data from the radio is collected and stored in the DXM Local Registers. Local Registers defined as the outputs for devices are automatically written to the ISM radio when the contents change. The data is organized (grouped) by each input.
: For example, all input 1 data is stored in the first 48 Local Registers (1–48), Gateway, Node 1–47). The next 48 registers (49–96) store input 2 data for all devices.
: Local Registers 1–384 store inputs 1–8, Local Registers 385–768 store outputs 1–8.
: See the table for more details.

DXM Local Registers organized by Device (GW=0, Nodes 1–47) where DXM Local Register address = (Device# × 16) + DevReg#

Device Type	Device Reg		Gateway	Node 1	Node 2	Node 3	Node 4	...	Node 47
Input 1	1	→	1	17	33	49	65		753
Input 2	2	→	2	18	34	50	66		754
Input 3	3	→	3	19	35	51	67	...	755
Input 4	4	→	4	20	36	52	68		756
Input 5	5	→	5	21	37	53	69		757
Input 6	6	→	6	22	38	54	70		758
Input 7	7	→	7	23	39	55	71		759
Input 8	8	→	8	24	40	56	72		760
Output 1	9	→	9	25	41	57	73		761
Output 2	10	→	10	26	42	58	74		762
Output 3	11	→	11	27	43	59	75	...	763
Output 4	12	→	12	28	44	60	76		764
Output 5	13	→	13	29	45	61	77		765
Output 6	14	→	14	30	46	62	78		766
Output 7	15	→	15	31	47	63	79		767
Output 8	16	→	16	32	48	64	80		768

DXM Local Registers organized by Input/Output (GW=0, Nodes1–47) where DXM Local Register address = ((DevReg# - 1) × 48) + (Device# + 1)

Device Type	Device Reg		Gateway	Node 1	Node 2	Node 3	Node 4	...	Node 47
Input 1	1	→	1	2	3	4	5		48
Input 2	2	→	49	50	51	52	53		96
Input 3	3	→	97	98	99	100	101	...	144
Input 4	4	→	145	146	147	148	149		192

Continued on page 51

Continued from page 50

Device Type	Device Reg		Gateway	Node 1	Node 2	Node 3	Node 4	...	Node 47
Input 5	5	→	193	194	195	196	197		240
Input 6	6	→	241	242	243	244	245		288
Input 7	7	→	289	290	291	292	293		336
Input 8	8	→	337	338	339	340	341		384
Output 1	9	→	385	386	387	388	389		432
Output 2	10	→	433	434	435	436	437		480
Output 3	11	→	481	482	483	484	485	...	528
Output 4	12	→	529	530	531	532	533		576
Output 5	13	→	577	578	579	580	581		624
Output 6	14	→	625	626	627	628	629		672
Output 7	15	→	673	674	675	676	677		720
Output 8	16	→	721	722	723	724	725		768

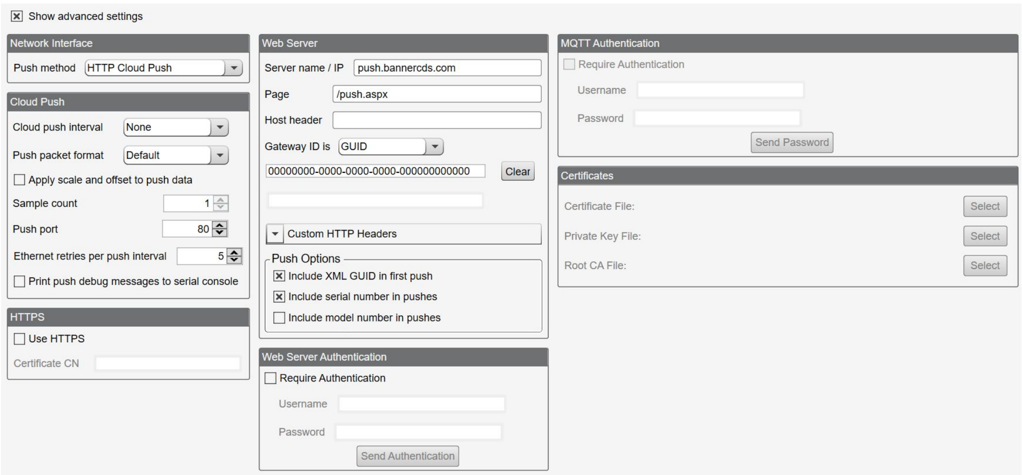
Depending on the model being configured, the Radio Communication settings may change.

Timeout
Set the communication timeout in hh:mm:ss.fff.

Delay Between Messages
Set the delay between the messages in hh:mm:ss.fff.

Cloud Services

Use **Cloud Services** to define the parameters to send register data to the website.



For instructions on how to push data to an Amazon Web Services (AWS) endpoint, refer to the technical note *Pushing to an Amazon Web Services (AWS) IoT Endpoint* (p/n b_5933667)

Select **Show advanced settings** if it isn't already selected to view all configuration parameters.

Cloud Services Network Interface

The DXM uses the **Network Interface** settings to establish the push method and whether you use an Ethernet or Cellular connection.

Select the **Push method**. Selecting **HTTP Cloud Push** or **MQTT** activates the additional parameter fields necessary to configure those options.

Push method
HTTP Cloud Push—Select to activate the Push interface drop-down list.

MQTT - Sparkplug B—Select to activate parameters relating to the Sparkplug B framework for MQTT publishing. The **MQTT - Sparkplug B** tab is visible after the selection is made.

MQTT—Select to activate the MQTT parameters. The **MQTT** tab is visible after the selection is made.

Push interface
Select **Ethernet** or **Cell** from the drop-down list. The **Cellular** tabs become visible after the selection is made. The **Ethernet** tab is always visible when a DXM model other than DXM100-A1 is selected.

See "Ethernet Settings" on page 55 and "Cellular" on page 54.

Cloud Push

Apply scale and offset to push data
When selected, the DXM applies scales and offsets to local register values before pushing them to the cloud. Do not select this parameter when pushing data to the BannerCDS website (bannercds.com).

Values are uploaded as integers; floating point values are truncated to the integer during the upload process.

Cloud Push Interval
Defines the time interval for cyclical data pushes sent to a webserver or host system. Setting the **Cloud Push Interval** to zero disables cyclical push messages. When using pushes created by a threshold rule, the **Cloud Push Interval** is ignored. Cloud pushes created from ScriptBasic are independent from these settings.

For example, with a **Cloud Push Interval** set to 1 hour and the sample count set to 10, the DXM saves the register data every 6 minutes and pushes all data to the web server every hour.

Ethernet Retries per push interval
Used when pushing HTTP data to a webserver using Ethernet and defines the number of attempts made to send HTTP data to the web server before it will save the message into a local file to be pushed up at the next push interval. (default 5)

Push packet format
Default—Set to **Default** when pushing to the BannerCDS website (bannercds.com).

JSON—Set to **JSON** when pushing to third-party cloud services such as MQTT.

Push Port
Defines the HTTP push data port on the web server.

The factory default setting is port 80. Set the port to 443 when you are using HTTPS.

Sample count
Defines how many times between the **Cloud Push Interval** to save data. The sample points are saved, then sent at the next **Cloud Push Interval**.

Cloud Services HTTPS

Selecting **HTTPS** indicates to the DXM to use TLS (Transport Layer Security) as a sub-layer under regular HTTP application layering. HTTPS encrypts and decrypts user data to and from the web server.

The webserver is required to carry a certificate. Select **Use HTTPS** to enable TLS services. Not all servers/communication networks support TLS.

Enter a **Certificate CN**. A Certificate CN is required to be associated with one or more domain names, called common name (CN). A single name Certificate CN is typically www.yoursite.com. A wildcard certificate includes single level sub-domains. The certificate CN is a wildcard certificate; *.bannercds.com. If nothing is entered into this field, the BannerCDS CN will be used; *.bannercds.com.

Cloud Services Web Server

Define all the web server settings in the **Web Server** section of this screen.

Custom HTTP Headers
Users can create up to five **Custom HTTP Headers** to provide custom information to the web server. The custom headers are sent with every push to the web server. **Custom HTTP Headers** are text fields the user can define that will be included in the HTTP header when the data is pushed to the web server.

Gateway ID
From the **Gateway ID** drop-down list, select either **GUID** or **User Defined Text**.

For custom web servers the Site ID string can be changed from a GUID (Global Unique IDentifier number) to a user-defined string of numbers or letters. The user-defined text is a unique site string defined by the web server when a site is created. Copy the string from the website into this field.

Host Header
A **Host Header** allows for multiple domains to reside at the same server address. Similar to how many people can share one phone number, dialing one number connects you to the phone but to talk to someone specific you need another piece of information. This is where the **Host Header** field is used. An IP address can get you to a server and the **Host Header** field allows for multiple domains residing at that server.

Not all servers require this field. When left blank, the intended target of the push message is the server name. The BannerCDS website does not require this field.

Page
Enter the **Page** that directs incoming data at the web server (/push.aspx)

Push Options
Include XML GUID in first push—Select to include the configuration file GUID (Global Unique IDentifier) in the first initial push to the web server. The web server can then verify that the XML configuration file that is loaded into the DXM is the same as what is loaded into the web server.

Include serial number in pushes—Select to include the serial number in your data pushes.

Include model number in pushes—Select to include the model number in your data pushes.

Include cell connection quality in pushes—Select to include the quality of the cell connection in your data pushes.

Omit push failures in logs—Select to not report push failures in the logs.

Server Name/IP
Webserver address used by the DXM when pushing data to the cloud.

Enter the domain name, `push.bannercds.net`.

The DXM defaults to using a public DNS (domain name server) to resolve domain names. To use a specific DNS, enter the IP address under the **Settings › Network** tab.

Web Server Authentication

Web Server Authentication defines a username and password to be sent to the webserver with every push data set to validate the sending device before storing any data in the database. If the webserver is expecting login credentials, the DXM must be programmed with the username and password. This only provides login credentials for authentication to the server; this does not secure the data payload.

The DXM must be connected to the computer. Select **Require Authentication**, then enter a username and password. Manually cycle power to the DXM after the username and password are written to it. The credentials cannot be read from the DXM.

IMPORTANT: After **Require Authentication** is selected and the username and password are sent to the DXM to be stored, go to **File › Save** to save the XML configuration file. Then send the new XML configuration file to the DXM. If the configuration file loaded into the DXM does not have **Require Authentication** selected, the username and password will not be sent.

Cloud Services MQTT

According to Amazon, "AWS IoT Core lets you connect IoT devices to the AWS cloud without the need to provision or manage servers. AWS IoT Core can support billions of devices and trillions of messages and can process and route those messages to AWS endpoints and to other devices reliably and securely. With AWS IoT Core, your applications can keep track of and communicate with all your devices, all the time, even when they aren't connected." (https://aws.amazon.com/iot-core/)

Please note that AWS IoT Core is only one possible vendor with which an MQTT-enabled DXM can be integrated.

To configure your system for MQTT:

1. Connect to one of the following models: DXM700, DXM1000, DXM1200-xx, DXM1500, DXMR90-X1E, DXMR110-8K, or R70ER model.
2. On the **Settings › Cloud Services** screen, select **MQTT** from the **Push method** drop-down list.
3. In the **Certificates** section, select and upload a **Certificate File**, a **Private Key File**, and a **Root CA File** to allow DXM700s to push to MQTT.

Note that these controls upload certificate files to the connected DXM700's microSD card and do not affect the DXM's actual configuration file.

MQTT

Host—The URL of the host you'll be publishing data to

ID—String that identifies this DXM when it publishes data

Port—The port to publish to; defaults to 8883 for an SSL connection

Print debug messages to serial console - When selected, the publish behavior is viewable using the Serial Console in the Traffic menu while the DXM is connected

Certificates

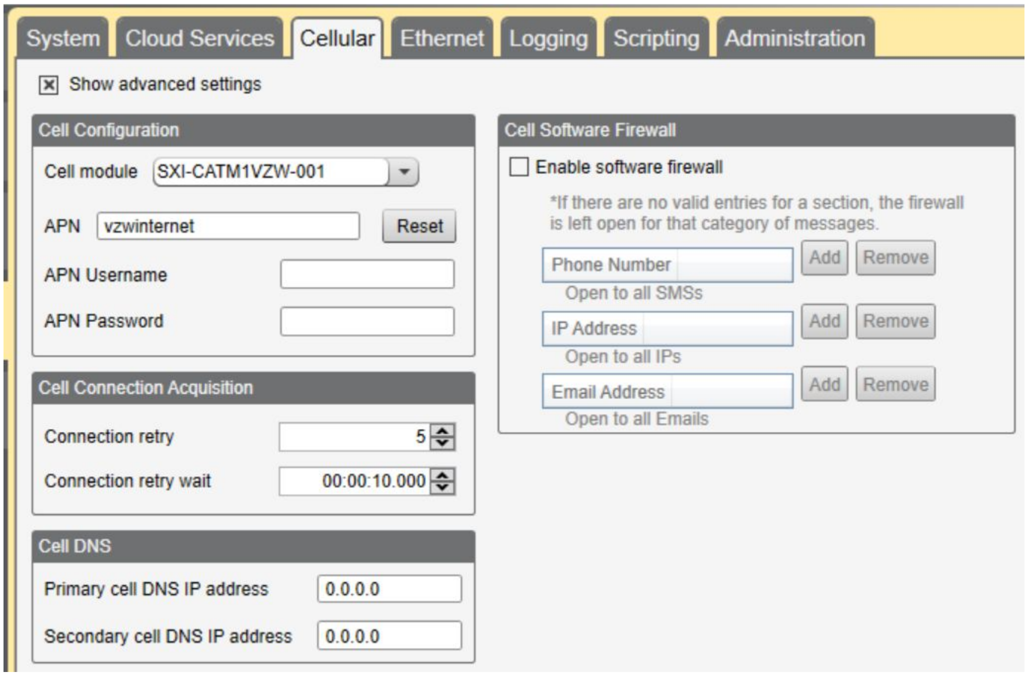
Certificate File and Private Key File—The Certificate File (.cert.pem extension) and Private Key File (.private.key extension) can be downloaded as the first step of certificate creation on AWS IoT Core (Secure > Certificates > Create).

Root CA File—The standard Root CA file (.pem extension) is entitled Amazon Root CA 1 and can be found under the CA certificates for server authentication section of the following web page: https://docs.aws.amazon.com/iot/latest/developerguide/server-authentication.html

See "MQTT Settings" on page 55.

Cellular

The **Cellular** settings screen displays only when **Cell** is selected as the push interface on the **Settings › Cloud Services** screen.



Cell Configuration

Select the DXM's cellular modem from the **Cell module** drop-down list.

- **SXI-LTE-001**—Banner Verizon 4G LTE Cat. 1 cellular modem for USA only (obsolete)
- **SXI-CATM1VZW-001**—Banner Verizon LTE Cat. M1 cellular modem for USA only
- **SXI-CATM1ATT-001**—Banner AT&T LTE CAT-M1 cellular modem for North America only (roams to Canada and Mexico on AT&T partner networks)
- **SXI-GSM-001**—Banner 3G HSPA (GSM) cellular modem for global use (obsolete)

APN, **APN Username**, and **APN Password** are user-defined fields. The APN setting for the LTE cellular modem using a Banner Cloud Data Services wireless plan is "vzwinternet". The default APN for the selected Cell module will automatically be set unless the user enters a custom APN.

Cell Connection Acquisition

These parameters the interaction with the cellular network when trying to establish a connection.

Connection Retry—Defines how many attempts are to be made to create a connection with the wireless network.

Connection Retry Wait—When a cellular connection fails, the Connection Retry Wait parameter defines how long the system waits until it attempts to create another cellular connection.

Cell DNS

The DXM defaults to using a public DNS service to resolve domain names. Enter the IP address of a primary and secondary DNS server to select a particular DNS server.

The DXM defaults to using the carrier DNS service. To redirect the DXS requests to a different DNS service, enter a primary and secondary cell DNS.

Ethernet Settings

Use the **Ethernet** screen to configure the DXM's Ethernet connection for Cloud Services, Mail, Modbus TCP, and UDP console functionality.

Current Device IP Settings

The current IP address (when connected via Ethernet) can be read from the device or by using the LCD menu on the DXM Controller. Click **Get Settings From Device** when the device is connected via the USB port.

IP Address

Select a static IP address or select the automatic assignment of an IP address by selecting DHCP from the drop-down menu. If Static IP is selected, enter the IP and subnet addresses.

You can also enter static IP addresses using the DXM's LCD menu system. Entering IP Addresses using the menu system overrides the IP addresses in the XML configuration files. Clear the IP addresses in the menu system to use the IP address in the XML configuration file.

Profinet

Select **Enable Profinet** if needed. When Profinet is enabled, all other Ethernet settings are disabled. Profinet is available for the DXM700, DXM1000, DXM1200, and DXM1500 models.

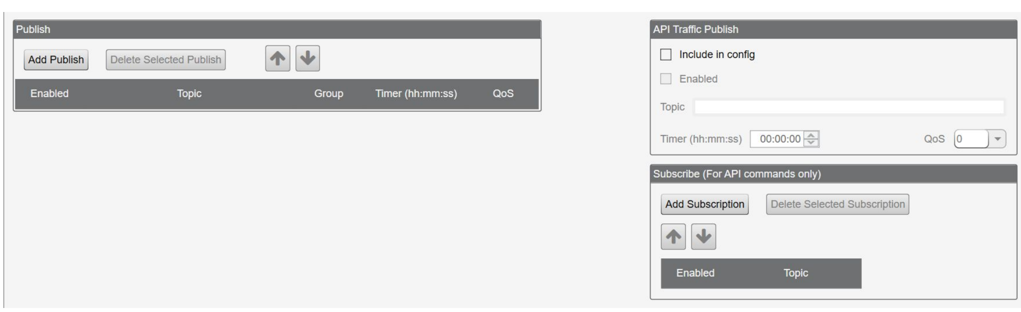
UDP Console

Select **Enable UDP console** to output DXM status messages via UDP. These messages can be viewed using the UDP console utility found in the **Device** menu.

PROFINET® is a registered trademark of PROFIBUS Nutzerorganisation e.V.

MQTT Settings

This screen is only visible when a supported DXM model is connected and MQTT is selected as the push method.



Publish

To publish data via MQTT, at least one publish rule must be defined.

Enabled—Enables the publish rule

Topic—Denotes the topic to publish data to

Group—Defines a push group for this publish rule; when this rule publishes, data from all local registers that are configured with this push group publish. Each **Local Register** belongs to one of the 32 Publish Groups (numbered 1 through 32 and configurable within the MQTT section of the **Local Registers › Local Registers in Use › Edit Register** screen). A given register will be pushed with this specified group only if **Push to MQTT** is selected. If the configuration software opens an existing configuration file within which a local register is set as a member of multiple MQTT push groups, it will instead be parsed as a member of the largest push group specified for that register, omitting any additional group memberships denoted in the file.

Timer (hh:mm)—Rate at which data associated with this rule publishes. The maximum setting is 23 hours 59 minutes (23:59).

QoS—Defines the delivery guarantee made for every message published to this topic. 0 - The message attempts to send only once; 1 - The message is re-transmitted until it receives a receipt acknowledgment.

Subscribe

The DXM is capable of subscribing to MQTT topics published to by other sources.

Enabled—Select to enable the subscription

Topic—Denotes the topic to subscribe to

API Traffic Publish

Include in config—When selected, API Traffic Publish settings are included in the configuration regardless of whether they're enabled or disabled

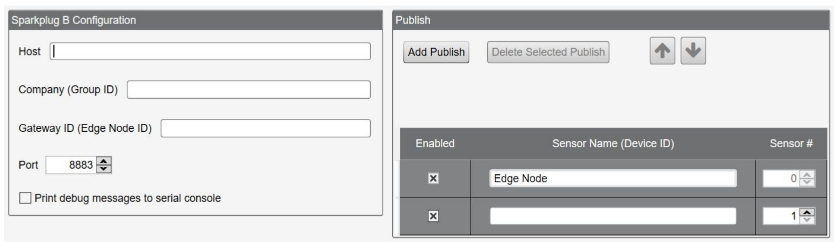
Enabled—Enables API Traffic publish behavior

Topic—Destination topic to publish API Traffic to

Timer (hh:mm)—Rate at which to publish API Traffic data. The maximum setting is 23 hours 59 minutes (23:59).

MQTT - Sparkplug B Settings

This screen is only visible when a supported DXM model is connected and MQTT - Sparkplug B is selected as the push method.



Sparkplug B Configuration

Host—The URL of the host you'll be publishing data to

Company (Group ID)—The Group ID to send data to (per the Sparkplug B spec). When publishing data to BannerCDS, set this field to the numerical ID of the company your gateway belongs to on the BannerCDS website.

Gateway ID (Edge Node ID)—The Group ID to send data to (per the Sparkplug B spec). When publishing data to BannerCDS, set this field to the numerical ID of your gateway on the BannerCDS website.

Port—The port to publish to; defaults to 8883 for an SSL connection

Include register numbers in names—When selected, each register's name appends the register number within the published data. This is required when publishing data to BannerCDS but can be deselected when publishing to other destinations.

Print debug messages to serial console—When selected, the publish behavior is viewable using the Serial Console in the Traffic menu while the DXM is connected.

Publish

When publishing using Sparkplug B, an Edge Node is defined and enabled by default for gateway-level register data. To publish data in groups representing individual sensors, create additional publish rules.

Enabled—Enables the publish rule

Sensor Name—Denotes the Device ID to publish data to. When publishing data to BannerCDS, this name appears in the sensor list belonging to this gateway on the BannerCDS website.

Sensor #—Defines a sensor # for this publish rule. When this rule publishes, data from all local registers configured with this sensor # are published. Each Local Register belongs to one of the 33 sensors (numbered 0 through 32 and configurable within the MQTT section of the **Local Registers › Local Registers in Use › Edit Registers** screen). A specified register publishes with this specified group only if **Publish via MQTT** is selected.

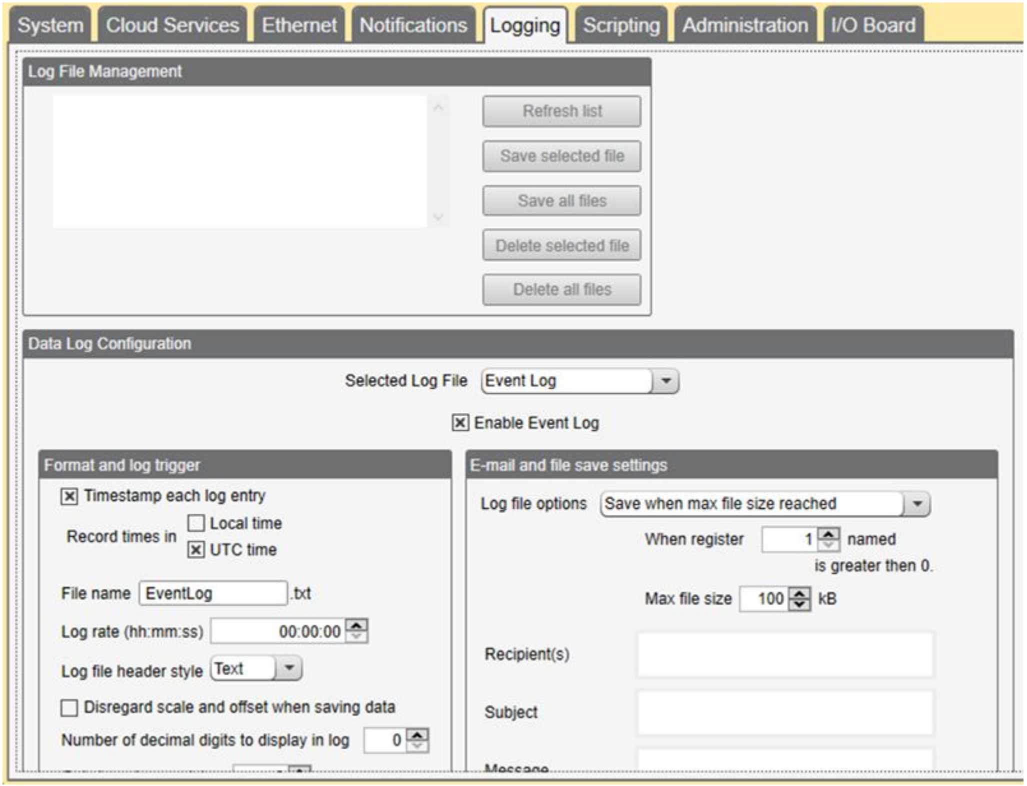
Notifications

To receive email and/or SMS text messages related to events, please use the Banner Cloud Data Services messaging feature within the webserver. Information about Banner Cloud Data Services can be found on our website.

Logging

Use **Settings › Logging** to define the local logging setup for the on-board SD card of the DXM. Up to three cyclical logs and one event log can be defined for threshold events. Click on the arrow to the left to expand the log configuration parameters.

The string length limit for each line in the Log file is 1,000 characters. Lines typically start with the time stamp followed by the values of each saved register. Discrete values (0/1) are short, but analog values could be up to 10 characters each. Taking an average of 6 characters for each value stored, the practical limit will be about 160 registers per log file.



The data log configuration parameters define the log file name, size, and what to do when the size is exceeded.

View, save, or delete files stored on the micro SD card using the **Log file Management** section. The DXM needs to be connected to a PC for these features to operate.

An HTTP Log file is created for a data packet pushed to the web server. Successful pushes cause the HTTP log to be time-stamped and placed into a dated folder under the "_sxi" directory. Typically, these logs are deleted daily.

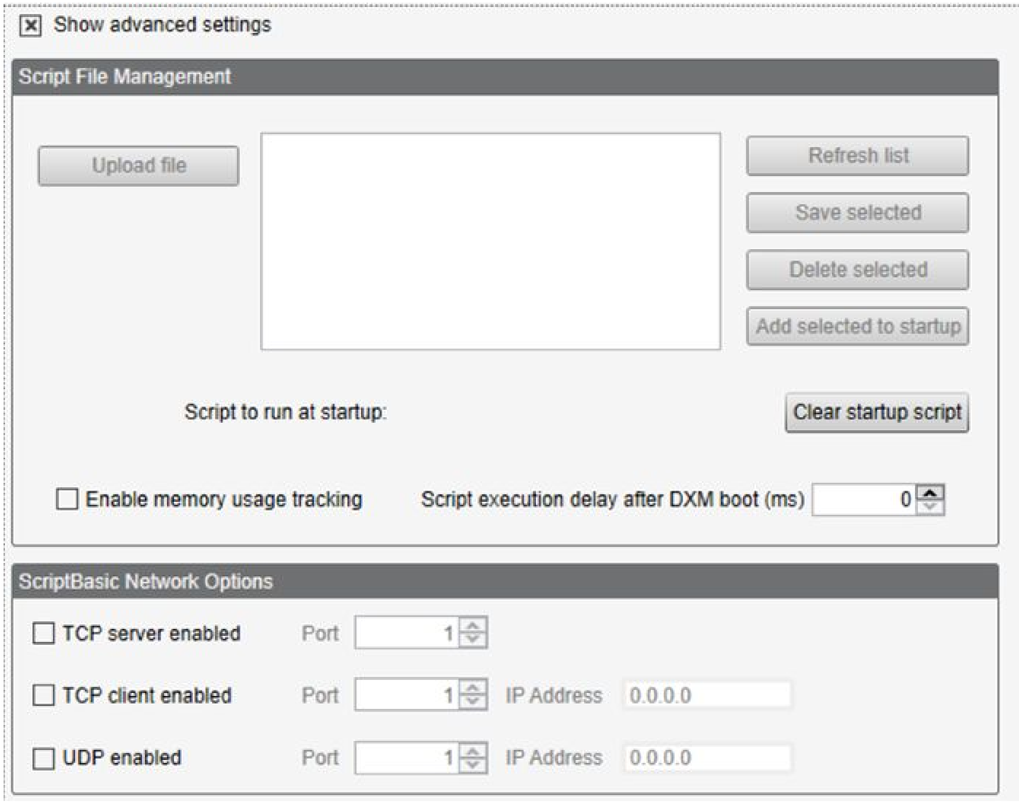
Transmissions that fail to make it to the web server accumulate in the HTTP Log file, retrying the failed attempts at the next cyclic interval. After 10 failed attempts, the HTTP Log file is time-stamped and saved on the SD card in a separate directory ("_sav") and must be retrieved manually.

Scripting

The DXM can run one ScriptBasic program. Use the **Scripting** screen to save, load, or delete script files on the DXM.

Files other than ScriptBasic files can be uploaded to the DXM. Files are placed in the root directory and can only be accessed by a ScriptBasic program.

Select **Show advanced settings** in the upper right of the screen to display the **ScriptBasic Network Options**.



Select a ScriptBasic file from the current ScriptBasic files window and select **Add Selected to Startup Scripts** to define which program should be run at boot time. Save the XML configuration file before loading to the device. A reboot is required to start a new script program.

To upload files:

1. Click **Upload**. A pop-up window appears.
2. Select the pull-down menu to select the file types: .sb, .esb or *.*.

Enable memory usage tracking
: Enables an internal memory watcher program that tracks the memory allocation for a ScriptBasic program. The results are displayed on the DXM's LCD, under **System Info › Script**.

Script execution delay after DXM boot (ms)
: Indicates the time (in ms) that the DXM should wait before it begins running the designated startup script after the DXM powers up.

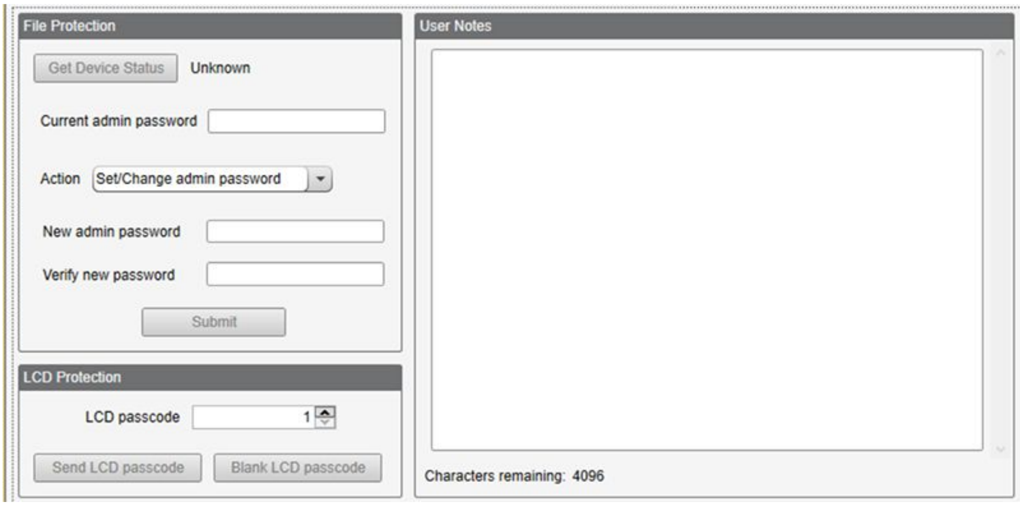
ScriptBasic Network Options
: **TCP server enabled**—ScriptBasic can operate as a server watching for Ethernet packets. Use the FILEIN command in ScriptBasic Default port is 8845.
: **TCP Client enabled**—ScriptBasic can send Ethernet packets when this is enabled. Enter IP address of destination. Default port is 8845, port is selectable.
: **UDP enabled**—Enable ScriptBasic to send/receive UDP data traffic. Enter IP address of destination.

Administration

The DXM manages two passwords that are defined on the **Settings › Adminstration** screen. Defining a **File Protection** password requires a password to be entered before a configuration file or ScriptBasic program file is loaded to the DXM. The LCD protection password requires the user to enter a password on the DXM LCD to unlock the LCD display of the DXM.

File Protection—Setting a password requires that a user enters a password before loading a configuration file or ScriptBasic file. The password is stored in the DXM and requires the DXM to be connected to the computer via USB port or Ethernet to set the password.

LCD Protection—The LCD passcode locks the DXM LCD display menu from operation. A valid passcode must contain 1–9 numbers with the first number being non-zero. The user has access to the full menu system when the passcode has been entered using the DXM LCD menu. The user can re-lock the display by selecting the **Display Lock** menu or the DXM will automatically re-lock the display after 15 minutes with no activity. If the DXM reboots, the device prompts the user for the passcode.



1. With the device connected, click on **Get Device Status**.
2. The device reports back if it is locked or unlocked.
3. To change, set, or clear a password, select the appropriate action, fill in the required fields, then click **Submit**.

Notes

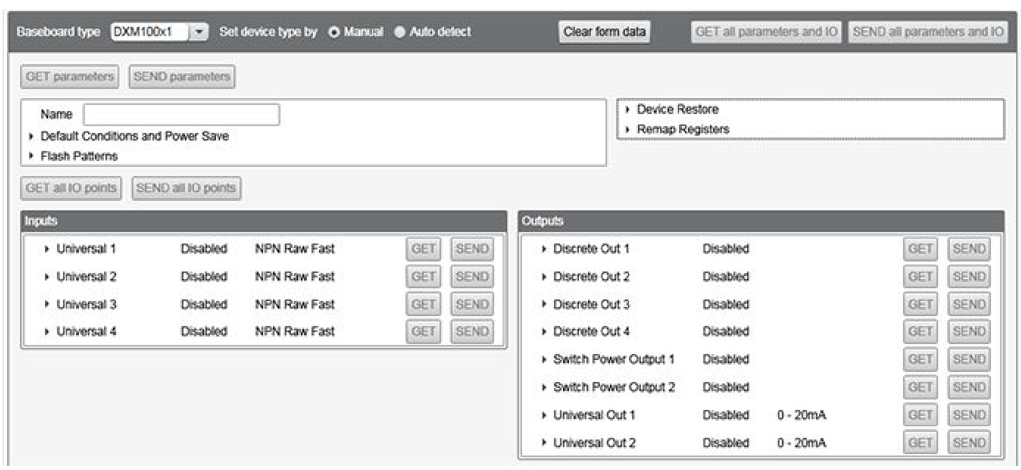
Use the **Notes** screen to enter informational text about the configuration file or notes specific to the application.

The **Notes** screen can save up to 4096 characters of alphanumeric characters and some special characters (~ ! @ # $ ^ ( ) _ - = + ? / }] { [ ). The notes are stored in the beginning of the XML file.

IO Board

The DXM I/O boards come in various combinations of inputs and outputs. The selection of the base board type can be set to **Manual** or **Auto detect**. When modifying parameters, always read (GET) the parameters first, edit, and then write (SEND) the new parameters back to the I/O board.

The I/O Board settings screen only displays when the selected **DXM Model** is DXM100, DXM100-A1, or DXM150.



Depending upon the amount of parameter data to see, click on one of the following buttons to read the current parameter settings:

GET all parameters and IO
: Reads all the I/O board device level parameter data plus the input and output parameter settings.

GET parameters
: Reads only the I/O board device parameters. Device level parameters effect the I/O board operation in general and are not specific to I/O points.

GET all IO points
: Reads the parameters for all the inputs and outputs on the I/O board.

GET
: Reads parameters for each individual I/O point, from one input or output.

Based on the number of changes made to the parameter data select one of the following buttons to write the data to the DXM I/O board:

SEND all parameters and IO
Writes all the I/O board device level parameter data plus the input and output parameter settings.

SEND parameters
Writes only the I/O board device parameters. Device level parameters effect the I/O board operation in general and are not specific to I/O points.

SEND all IO points
Writes the parameters for all the inputs and outputs on the I/O board.

SEND
Writes parameters for each individual I/O point, from one input or output.

Name

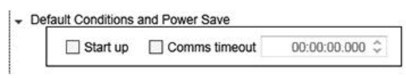
Assign a name to each device to it easier to track the device in the configuration software.

The **Name** must be less than 18 characters and contain only the standard alpha-numeric characters, the following standard ASCII characters: *, +, -, /, <, >, or a space.

After making any changes, click **SEND parameters** to write the changes. The status bar at the bottom of the screen indicates when the process is complete.

Default Conditions

Default conditions are the conditions under which outputs are sent to their defined default state. These conditions apply to all outputs.



Selecting **Start Up** sets this radio's outputs to their default values when this radio is powered up.

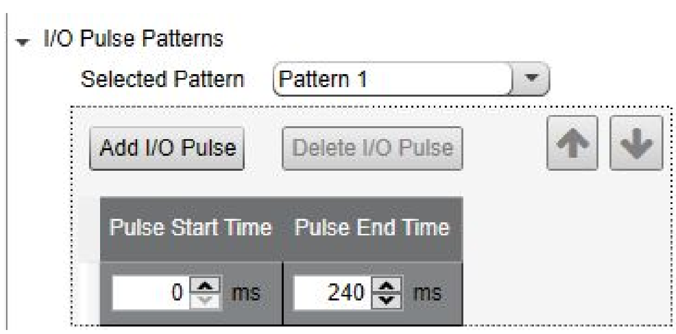
A **Comms Timeout** refers to the communication between the host system and the selected radio (which may be any radio within the network). Selecting Communication Timeout sets this radio's outputs to their selected default values when the host system has not communicated with this radio within the time specified. Set the communication timeout in seconds.

After making any changes, click **SEND parameters** to write the changes. The status bar at the bottom of the screen indicates when the process is complete.

After the radio problems are resolved with the parent radio or communication with the host system has resumed, the host system is responsible for writing the operating output value.

IO Pulse Patterns

Setting the I/O pulse pattern establishes an on and off pattern that can be used for a discrete output or switch power.

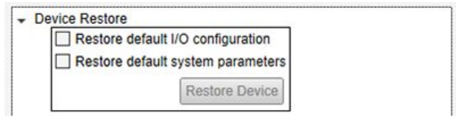


Define **I/O Pulse Patterns** by selecting specific timeslots to turn the output on or off. Although the pulse pattern was originally designed to turn on and off an indicator light, the pulse pattern can be set for any discrete output or switch power. Users may configure up to four different patterns per radio device. In the example shown, Pattern 1 is configured to turn on for the first 240 milliseconds within a 1280 millisecond range.

After making any changes, click **SEND parameters** to write the changes. The status bar at the bottom of the screen indicates when the process is complete.

Device Restore

Use these commands to restore default settings to the selected device.

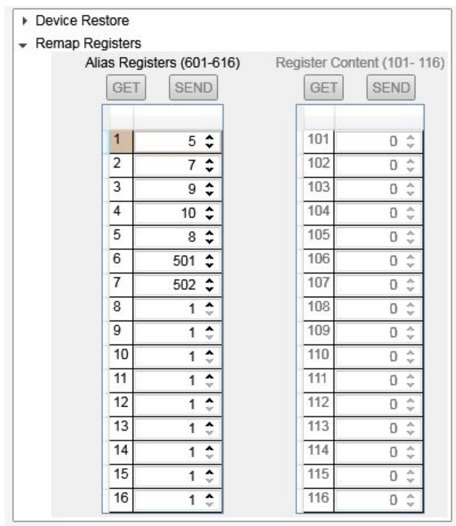


To restore default device parameters:

1. At the top of the screen, select a MultiHop device.
2. Click on **GET All Parameters** to get that device's I/O configuration.
3. In the **Device Restore** section, select the appropriate checkboxes.
    - **Restore Default I/O Configuration**. Restores default I/O configuration parameters to the values indicated in the device's datasheet, such as sample and report rates.
    - **Restore Default System Parameters**. Restore default system parameters, such as the radio configuration and binding code.
4. Click on **Restore Device** to restore the selected default values to your device.

Remap Registers or Register Aliasing

Use the **Remap Registers** section of the **Configure Device** screen to map registers to contiguous register locations to optimize Modbus read/write functions.



1. Verify the desired MultiHop Radio **Address** is selected in **Configuration Address**.
2. Click the arrow next to **Remap Registers**.
3. Fill in the source registers you would like to alias. The **Alias Registers** rows 1 through 16 are the user-defined entries of addresses of the registers to alias (rearrange). This alias table is stored in the MultiHop radio register addresses 601 through 616. In the example, source registers addresses 5, 7, 9, 10, 8, 501, and 502 are entered into the table.

The aliased **Register Contents** are in registers 101 through 116. For this example, when a host system reads Modbus registers 101 through 107 of the MultiHop radio, the register contents come from register 5, 7, 9, 10, 8, 501 and 502.

Inputs

The following parameters are used to configure the inputs.

The input parameters vary, depending on the enabled and available input types.

Input Configuration

Select **Enable** to turn on this input.

Sample Rate	The sample interval, or rate, defines how often the Sure Cross device samples the input. For battery-powered applications, setting a slower rate extends the battery life. Set the sample rate/interval in hours:minutes:seconds.

Universal Configuration

After enabling the input, configure the universal input type.

Count on falling values
Select to count on a falling transition.

Count on rising values
Select to count on a rising transition.

Input Type
Select the input type from the drop-down list.
NPN
PNP
0-20mA
0-10V
Thermistor
Potentiometer
Bridge
NPN Raw Fast

Preset Value (Counter)
Enter a number in the selection box and press the Set Value button to write a preset counter value to the register.

Analog Configuration

Enable push on change of state
Enables push registers for this input. When the discrete input changes state, the register value is pushed to the client radio if this register is configured to be a push register. For analog inputs, use the threshold and hysteresis parameters to define "on" and "off" points.

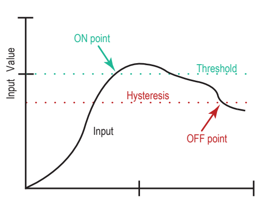
Enable register full scale
Turning Fullscale ON sets the entire register range of 0 through 65535 to represent the selected minimum through maximum input values. With Fullscale turned on, a register value of 0 represents the selected minimum value in microamps (for current inputs). A register value of 65535 represents the selected maximum value in microamps. For example, a register value of 0 is 0 and the register value of 65535 represents 20 mA (or 20,000 microamps). With Fullscale turned OFF, the register value represents unit-specific input readings. For units of current (mA), register values are stored as microAmps. Voltage values are stored as millivolts. A sensor reading of 15.53 mA is stored as 15530.

Signal Conditioning

Delta
The delta parameter defines the change required between sample points of an analog input before the analog input reports a new value. To turn off this option, set the Delta value to 0. To use the delta function, the push registers must be defined.

Hysteresis and Threshold (Analog)
Threshold and hysteresis work together to establish the ON and OFF points of an analog input. The threshold defines a trigger point or reporting threshold (ON point) for a sensor input. When the input value is higher than the threshold, the input is ON. Hysteresis defines how far below the threshold the analog input is required to be before the input is considered OFF. A typical hysteresis value is 10% to 20% of the unit's range.

In the example shown, the input is considered on at 15 mA. To consider the input off at 13 mA, set the hysteresis to 2 mA. The input will be considered off when the value is 2 mA less than the threshold.



Median Filter
When the median filter is turned on, three samples are taken for each analog sensor reading. The high and low values are discarded and the middle value is used as the analog value. Set to zero (0) to turn off the median filter. Set to one (1) to turn on the median filter.

Sample High and Sample Low
For analog inputs, the sample high parameter defines the number of consecutive samples the input signal must be above the threshold before a signal is considered active. Sample low defines the number of consecutive samples the input signal must be below the threshold minus hysteresis before a signal is considered deactivated. The sample high and sample low parameters are used to avoid unwanted input transitions.

Tau Filter
Set to zero (0) to turn off the tau filter. Set to 1 (weakest filter) through 6 (strongest filter) to turn on the tau filter. (In the DX80 products, the Low Pass Filter is a combination of the median filter and the tau filter.)

Temperature Settings

Enable full scale (Temperature)
Turning Fullscale OFF sets the register range of 0x8000 (–32767) through 0x7FFF (+32768) to represent the range of input values. With Fullscale turned OFF, a register value of 1450 represents 72.5 degrees (register values = temperature × 20). With Fullscale turned ON, users can specify the register minimum and maximum range of values. These min/max values are represented in the register as 0 (min) and 65535 (max).

Temperature resolution
Select high to store temperatures values in the registers as the measured temperature × 20. Set to low to store temperature values in tenths of a degree (measured temp × 10).

For example, if the measured temperature is 20.5 degrees, turning temperature scaling to high stores the temperature value as 410 while use low resolution stores the temperature as 205.

Units
Select either Celsius or Fahrenheit for your temperature readings.

Switch Power Settings

Configure the following parameters for the switch power settings (inputs).

Enabled
Associates I/O switch power functions to a specific input. Do not use these parameters to configure continuous, or device-level, switch power. Select one of the available switch power (SP) checkboxes to link that switch power to the input you're currently configuring.

Voltage and Warm-Up
Select the desired voltage and warm-up time. The voltage setting establishes the voltage of the switch power. The warmup time is the length of time the switch power must be on before the device can sample the input.

Extended Input Read Settings
Use the Extended Input Read parameter to link multiple input sampling times together when all devices are powered by the same Switch Power. For battery-powered devices, this uses less energy and prolongs battery life. Define the Sample Rate, Warm-up time, and Voltage parameters for the first input. These parameters for follow-on linked inputs are ignored. Click on the **Extended Input Read** arrow to select the additional inputs to read when the switch power is active.

For example, set the Sample Rate, Warm-Up time, and Voltage for Universal Input 1. Select SP1 to power Universal Input 1. By selecting Extended Input Read for 2 and 3, the device will also read inputs 2 and 3 when it reads input 1.

Outputs

The following parameters are used to configure the outputs, including the switch power outputs.

The output parameters vary, depending on the enabled and available output types. To begin configuring your outputs, **Enable** the output.

Discrete Output

Default Output Value
Select the default output value. When the selected default condition occurs and Hold Last State parameter is set to OFF, this output is set to the selected default output value (e.g. out of sync, communication timeout, start up).

Hold Last State
Retains its last value during the selected default condition (out of sync, communication timeout, start up).

Use I/O Pulse Pattern
To use a programmed pulse pattern, select **Use I/O pulse pattern**, then select the appropriate pattern from the drop-down list. Define the patterns in the Device Parameters section of this screen.

Switch Power Output

When linking a switch power output to a specific input, select the **Enable** checkbox and set the **Enable default state** to OFF. Use the settings for the specific input to link the switch power output and set the voltage and warm-up time.

For continuous switch power, set the voltage on this screen and set the default state to ON. Verify the default "start-up" conditions are set in the device parameters screens.

Enable default state
When enabled, this switch power output remains on during the selected default condition (e.g. out of sync, communication timeout, start up). When disabled, the switch power cycles off during the selected default condition.

Hold Last Voltage
When set, the switch power output retains its last value during the selected default condition (e.g. out of sync, communication timeout, start up).

Use I/O Pulse Pattern
To use a programmed pulse pattern, select **Use I/O pulse pattern**, then select the appropriate pattern from the drop-down list. Define the patterns in the Device Parameters section of this screen.

Voltage
To set a voltage for the switch power output, select a value. When configured for continuous voltage output, this switch power output no longer cycles on, warms up the sensors, then cycles back down. Because the output voltage remains constant, continuous voltage is typically used with solar power installations.

Universal Out

After enabling the output, configure the universal output type.

Output is
Select the output type from the drop-down list.

0-20mA
0-10V

Enable register full scale

Turning Fullscale ON sets the entire register range of 0 through 65535 to represent the selected minimum through maximum input values. With Fullscale turned on, a register value of 0 represents the selected minimum value in microamps (for current inputs). A register value of 65535 represents the selected maximum value in microamps. For example, a register value of 0 is 0 and the register value of 65535 represents 20 mA (or 20,000 microamps). With Fullscale turned OFF, the register value represents unit-specific input readings. For units of current (mA), register values are stored as microAmps. Voltage values are stored as millivolts. A sensor reading of 15.53 mA is stored as 15530.

Hold last state (output)

Retains its last value during the selected default condition (out of sync, communication timeout, start up, etc).

H-Bridge Configuration

Enable this output and the H-Bridge configuration to configure its parameters.

Cap warmup time

Similar to the switch power warm-up time, the h-bridge capacitor warm-up time (hours:minutes:seconds) is the time allotted to charge the capacitor used to activate the h-bridge and latching solenoid.

Switch time

Time (hours:minutes:seconds) needed to activate the h-bridge and latching solenoid.

Use I/O Pulse Pattern

To use a programmed pulse pattern, select **Use I/O pulse pattern**, then select the appropriate pattern from the drop-down list. Define the patterns in the Device Parameters section of this screen.

Voltage

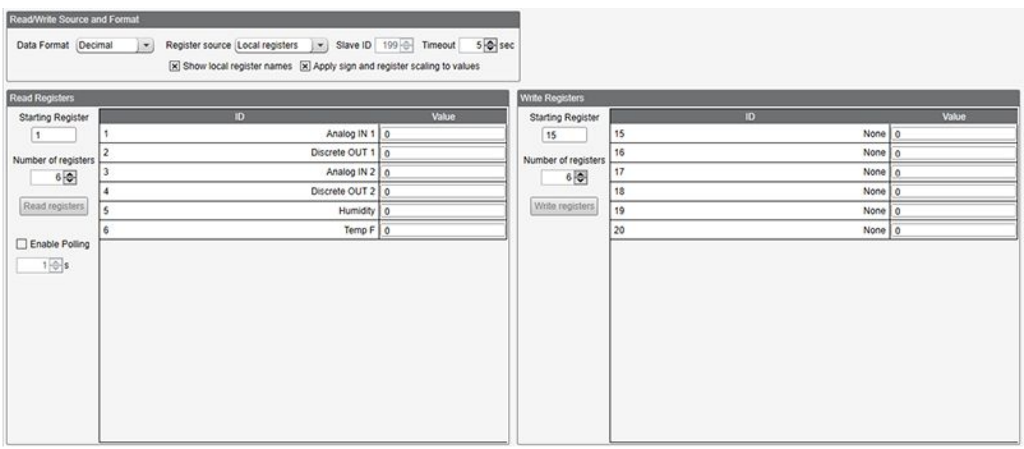
Voltage required to activate the h-bridge and latching solenoid.

Tools

The **Tools** screen provides utilities for reading and writing register values interactively, adding registers to the DXM's display, exporting protocol conversion configurations, and updating the DXM.

Register View

The Register View is a Modbus utility to help debug configurations and view register data for devices connected to the DXM.



Follow these steps to select the registers to view.

1. Connect to the DXM using USB or Ethernet.
2. From the **Register source** drop-down list, select between Local Registers (DXM), ISM registers (Gateway), remote device, I/O registers (available only when the selected model is DXM100, DXM100-A1, DXM150, DXM1000, and DXM1500), display registers, or DXM700 outputs (available only when the selected model is the DXM700). Enter a Modbus ID when accessing remote devices. If the selected DXM model is a DXM-R90x, a **Port** drop-down list displays when the Remote device is selected as the Register source. This drop-down list specifies which UART bus to use when accessing remote devices.
3. Select the **Starting Register** to display and the **Number of registers**.

Read Registers

To read the contents of a specific register or range of registers, select the starting register and the number of registers to read from.

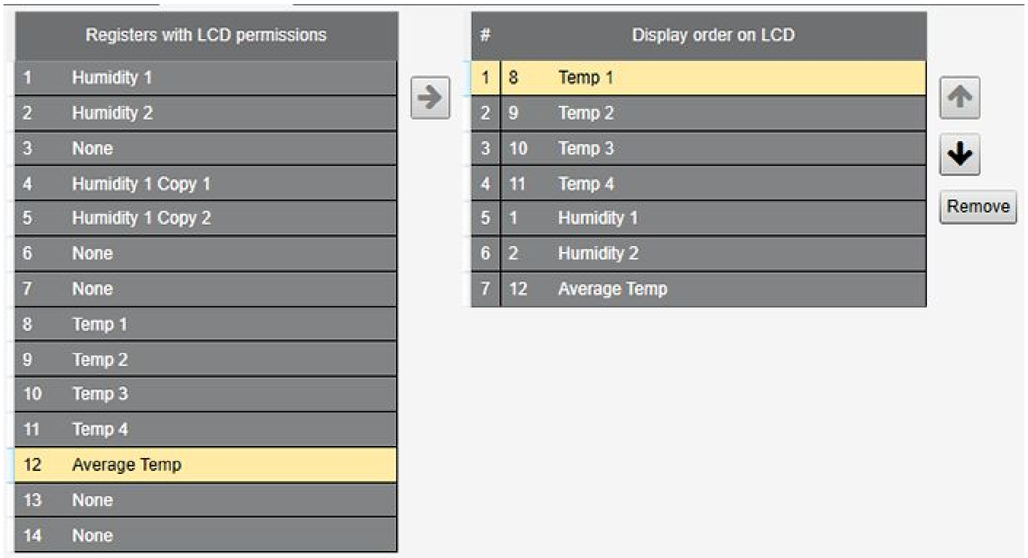
Click **Read Registers** to read the data once. The register values display in the **Read Registers** section. Select **Enable Polling**, specify a polling rate, and click **Read Registers** to create a constant polling loop.

Write Registers

To write values to a specific register or range of registers, select the starting register and the number of registers to write to. Enter the value to write to these registers and click **Write Registers** to send these defined values to the selected registers.

DXM Display

Use the **Local Registers › Local Registers in Use** screen to define which registers can be displayed under the Register menu of the DXM. Use the **DXM Display** screen to adjust the display order of the **Registers with LCD Permissions**. If the display order is not specified, the registers display in numerical order.



The Local Registers defined to be displayed on the LCD menu are shown in the left column.

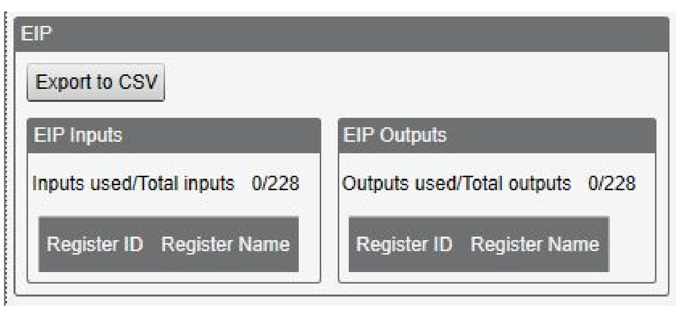
1. Select the Local Register by clicking next to the **Registers with LCD Permissions** number.
2. Click the arrow button to populate the **Registers with LCD Permissions** in the **Display order** column.
3. After defining the Local Registers to display, adjust the order using the up and down arrows to the right of the **Display order** column.

If there are more Local Registers defined with LCD permissions than there are in the **Display order** column, the remaining registers display in the order of the Local Register number. A Local Register can only be shown once on the LCD.

Protocol Conversion Overview

The **Protocol Conversion Overview** screen displays a list of the EitherNet/IP (EIP) Inputs and Outputs associated with DXM registers.

Define the Local Registers to be EtherNet/IP registers on the **Local Registers › Local Register Configuration** screen.



Scheduler

Use the **Scheduler** screens to create a calendar schedule for local register changes, including defining the days of the week, start time, stop time, and register values.

Schedules are stored in the XML configuration file, which is loaded to the DXM. Reboot the DXM to activate a new schedule.

If power is cycled to the DXM in the middle of a schedule, the DXM looks at all events scheduled that day and processes the last event before the current time.

For screens that contain tables with rows, click on any row to select it. Then click **Clone** or **Delete** to copy/paste or remove that row.

Reprogram

Use the configuration software to update the DXM's firmware. Previous firmware must be version 2.01 or later. For earlier firmware versions, contact Banner Engineering support.

Click **Get Device Information** to have the DXM read and display its serial number, model number, version, and firmware numbers for both the processor board and the I/O board.

By default, **Verify firmware compatibility before configuring DXM** is selected any time the configuration software launches. Deselect it to allow uploads of a configuration file without first verifying that the connected DXM's firmware supports the features. This field is not saved and will always be selected when the configuration software launches.

To reprogram the DXM, follow these instructions:

1. Apply power to the DXM,
2. Connect the DXM to your PC using USB or Ethernet.
3. On the menu bar, go to **Device › Connection Settings** and select the appropriate COMM port or enter the IP address.
4. From the **Tools › Reprogram** screen, click **Select upgrade file**.
5. Browse to the firmware HEX file location and select the file.

   Depending upon the connection, USB or Ethernet, the programming takes from 5 to 15 minutes to complete.

Chapter Contents

Chapter 3 Additional Information

Verifying Communication Between a MultiHop Radio and DXM Controller

Follow these steps to monitor the communications connection between a DXM acting as a client radio and the MultiHop server radios in a wireless network.

Required equipment includes:

- Wireless DXM Controller client with a MultiHop radio module
- Wireless DXM Controller servers and/or MultiHop server radios
- Windows-based PC running the DXM Configuration Software v4 (downloaded from the Banner website)

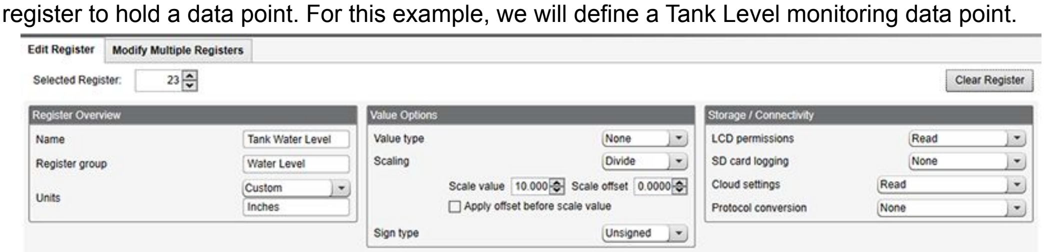
To confirm the radio communications connection between the client and server radios, define Read Rules and Action Rules. Use two local registers to monitor each MultiHop radio. Use an optional third register to monitor how long the server radio was not communicating with the client radio.

1. Connect to the DXM Controller with the MultiHop client radio using serial or TCP/IP.
2. Define the Read Rule.
3. Define the Threshold/Action Rule.
4. Repeat these steps for each MultiHop server radio you'd like to monitor.

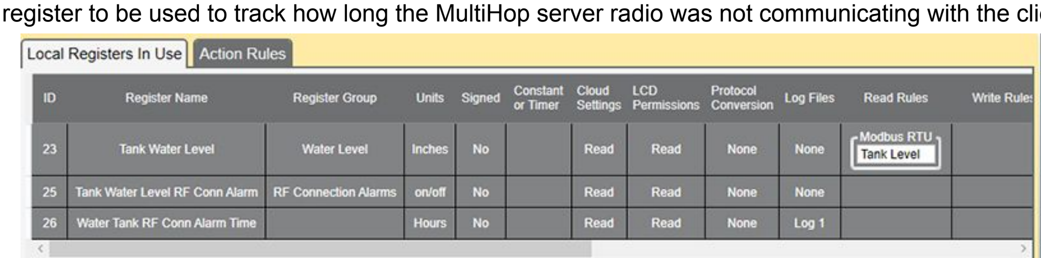
Define Local Registers to Verify Communication Between a MultiHop Radio and DXM

Define the local registers used to verify the connection between a DXM Controller with the MultiHop client radio and a MultiHop server radio.

1. Go to the **Local Registers › Local Registers in Use** screen.
2. Define a register to hold a data point. For this example, we will define a Tank Level monitoring data point.



3. Define a register to be used as an alarm notification register when the MultiHop client radio cannot communicate with the MultiHop server radios.
4. Define a register to be used to track how long the MultiHop server radio was not communicating with the client radio.

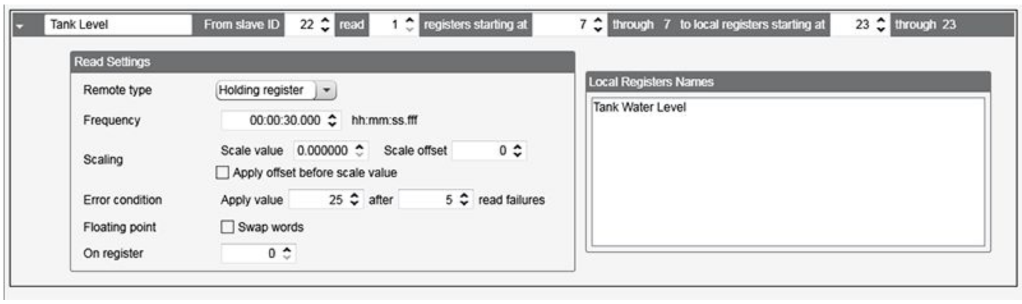


Create a Read Rule to Verify Communication Between a MultiHop Radio and DXM

Create a Read Rule to define how often to read the sensor register and what to do if the communication attempt fails.

1. Go to the **Register Mapping › RTU › RTU Read** screen.
2. Click **Add New Rule** to create a Read Rule.
3. Name the Read Rule and define from which server ID this register is being read, how many registers are being read, and the starting register.

   For the Tank Level example, we are reading one register (register 7) from server ID 22.
4. Define how often to read this register (**Frequency**).
5. Define what value should be written to the register (**Apply value**) after the number of failed read attempts (**read failures**).



   Select an alarm value that makes sense for the potential values of the application, but won't adversely affect graphing or charting the data point for analysis. For this example, we will use an alarm value of 25, because the likely values for this application will range from 0 to 20. The alarm value of 25 will be written to local register 23 after five read failures.

Create a Threshold Rule to Verify Communication Between a MultiHop Radio and DXM

Create an action rule to define the behavior of the system when the communication fails.

1. Go to the **Local Registers › Action Rules › Thresholds** screen.
2. Click **Add Threshold Rule**.
3. Define a **Threshold Rule** so that when the local register Failure to Read value equals the error value (25 for the tank level register), a value of 1 is entered into the Communication Alarm register.

   For the tank level example, when this register's value equals 1, local register 26 tracks how long this remote MultiHop server radio was not able to be reached. The alarm is sent to the web server service, and the event is logged in the Events Log on the DXM.

Chapter Contents

Chapter 4 Troubleshooting

Manually Install the DXM Driver

If the DXM fails to connect over USB, first power down the DXM and unplug the USB cable from the PC, then follow these steps.

1. Verify the Windows update is configured to download device drivers.
    a. On your Windows-based PC, go to the **Start** menu and select **Devices and Printers**.
    b. Right-click the name of your computer, then select **Device installation settings**.
    c. Select **Yes, do this automatically (recommended)**, then click **Save Changes**. If you're prompted for an administrator password or confirmation, type the password or provide confirmation. If Yes is already selected, click **Cancel** to close the dialog box.
2. Apply power to the DXM and plug the USB cable back into the computer.

    The computer should recognize the DXM. If not, go to step 3.
3. If the DXM is still unrecognized, follow these steps.
    a. Open Windows File Explorer and browse to either: C:\Program Files (x86)\Banner Wireless\DXM Configuration Platform\Drivers or C:\Program Files\Banner Wireless\DXM Configuration Platform\Drivers.
    b. Copy the files banner2001_cdc.cat and banner2001_cdc.inf into c:\Windows\inf.
    c. From the **Start** menu, go to **Control Panel › Hardware and Sound › Device and Printers › Device Manager**. The DXM Controller should appear as an unknown device.
    d. Right-click on the DXM icon and select **Properties**.
    e. Click the **Driver** tab.
    f. Click **Update Driver**.
    g. Click **Browse my Computer** for the driver software.
    h. Click **Let me pick from a list of device drivers on my computer**.
    i. Select AT91 CDC class driver and click **Next**. The process is complete. If AT91 is not present, go to the next step.
    j. Select **Show All Devices** and click **Next**.
    k. The system drivers display. Under Manufacturer, search for Banner Engineering Corp.
    l. Select the DXM Wireless Controller and click **Next**.
    m. A warning message appears stating that installing this device driver is not recommended. Click **Yes**.
    n. If the Device Manager lists a com port with the DXM Wireless Controller title, the driver successfully installed.

Update the Software

The software version displays in the lower right corner. If there is a red exclamation point in front of the version number, you are not running the most recent version of the software.

Follow these steps to update your software.

1. Click on the red exclamation point.

    An **Update Application** pop-up box appears, listing the newest version available.
2. Click on the version number link.

    The newest version of the software downloads to the download directory listed in your browser.
3. Close the software program. If you leave the software running while updating, you will need to reboot the computer.
4. If you have downloaded a ZIP file, unzip the file and double-click the EXE to install the software. If you have downloaded an EXE file, double-click the file to install the software.

Reset the Controller if the Password is Lost

If the device is locked and the password is lost, go to the DXM to clear the password. Clearing the password from the controller also erases the configuration file and all ScriptBasic programs. To clear the password:

1. Remove the power from the device.
2. Set DIP switch 4 to the ON position.
3. Disconnect the Ethernet cable.
4. Apply power to the device.
5. Immediately after powering up, press and hold the reset button on the processor board.

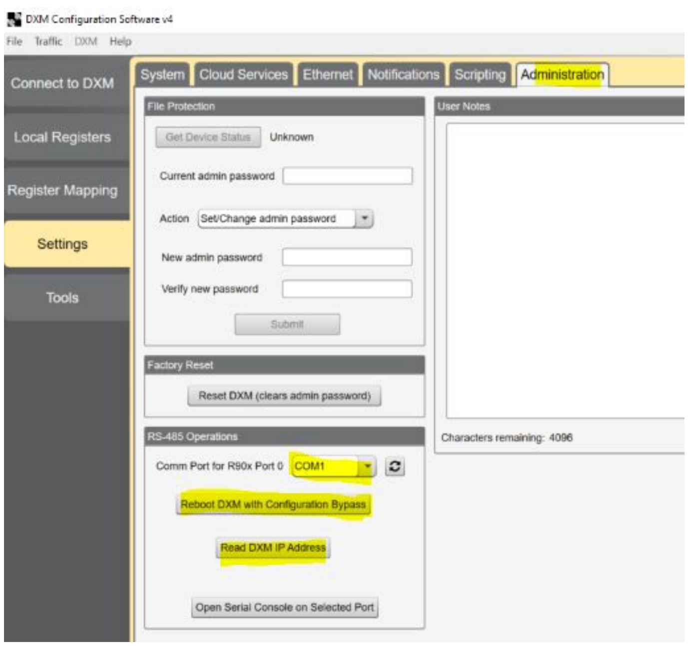
   After 10 to 15 seconds, all LEDs on the processor board flash quickly for a second. All files are erased and the password is cleared.
6. Remove the power from the device.
7. Set DIP switch 4 to the OFF position.
8. Reconnect the Ethernet cable.
9. Apply power to the device.

   This erases all files from the device.
10. Reload your configuration files.

Recover an IP Address with a DXMR90

Follow these steps to recover an IP address using a DXMR90.

The BWA-UCT-900 cable is required for these steps.



1. Power the DXM using the BWA-UCT-900 cable and plug the USB into your computer.
2. Launch the DXM Configuration Software.
3. Go to the **Settings › Administration** screen.
4. At the bottom left of the screen, select which COMM Port the DXM is connected to.
5. Click **Read DXM IP Address**. The IP Address of the DXMR90 should display.
   - If you are able to connect to the DXM at the IP Address displayed, stop here and connect to the DXM as usual.

- If the IP Address was not returned or cannot be connected to at the displayed address, then continue to the next step.

6. Click **Reboot DXM with Configuration Bypass**.
7. You may be prompted to reboot the controller at this point. If you are, click yes and then cycle power to the DXM while leaving the USB plugged into the computer. The DXM should now be set at the default IP Address and Subnet (192.168.0.1 and 255.255.255.0).
8. Confirm the IP address by clicking **Read DXM IP Address**.
9. Navigate to **Connect to DXM** and connect at the 192.168.0.1 address.
10. Edit the XML file to have the desired IP Address (**Settings › Ethernet**).
11. Send the XML file to the DXM.
    After the DXM reboots, the controller will have the IP Address assigned in the XML file.

Chapter Contents

Chapter 5 Product Support and Maintenance

Banner Engineering Corp. Software Copyright Notice

https://www.bannerengineering.com/us/en/company/terms-and-conditions.html

Disclaimer of Warranties. This software is provided "AS-IS." To the maximum extent permitted by applicable law, Banner, it affiliates, and its channel partners disclaim all warranties, expressed or implied, including any warranty that the software is fit for a particular purpose, title, merchantability, data loss, non-interference with or non-infringement of any intellectual property rights, or the accuracy, reliability, quality or content in or linked to the services. Banner and its affiliates and channel partners do not warrant that the services are secure, free from bugs, viruses, interruption, errors, theft or destruction. If the exclusions for implied warranties do not apply to you, any implied warranties are limited to 60 days from the date of first use of this software.

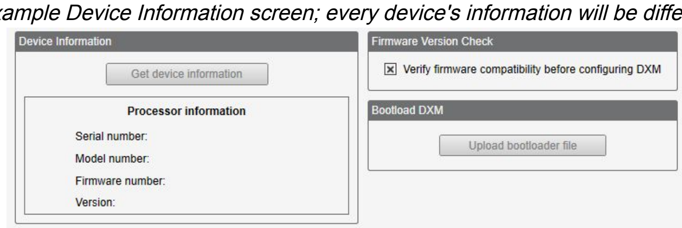
Limitation of Liability and Indemnity. Banner, its affiliates and channel partners are not liable for indirect, special, incidental, punitive or consequential damages, damages relating to corruption, security, loss or theft of data, viruses, spyware, loss of business, revenue, profits, or investment, or use of software or hardware that does not meet Banner minimum systems requirements. The above limitations apply even if Banner and its affiliates and channel partners have been advised of the possibility of such damages. This Agreement sets forth the entire liability of Banner, its affiliates and your exclusive remedy with respect to the software use.

Update Your DXMR90 and DXMR110 Processor Firmware Using the Configuration Software

Follow these steps to update your DXMR90 and DXMR110 processor firmware using the DXM Configuration Software.

1. Using the DXM Configuration Software version 4 or later, connect to the DXM via Ethernet.
   File loads to the DXM will take several minutes.
2. On the DXM Configuration Software, go to **Tools › Reprogram › Get Device Information** to verify the current firmware version.
   You must load a different version with the same firmware number for the bootloader to operate. Download firmware files from the Banner website.

   Example Device Information screen; every device's information will be different

3. Under **Tools › Reprogram**, click **Upload bootloader file** to select the firmware file to program.
4. Select the .HEX file provided to bootload the device.
   This is a large file, so it may take 10-15 minutes to upload.
5. After the file load is completed, reboot the device by selecting **DXM › Reboot DXM**.
   Upon reboot, the device will begin to bootload. There will be a solid green light on for 6 to 7 minutes. Do not be alarmed if the device appears to not do anything. After 6 to 7 minutes, an amber LED nearest the power connector flashes for 2 to 3 minutes. After the boatload process is finished, the device returns to normal operation.
6. DO NOT disconnect the power during the 6 to 7 minutes after the device cycles the power.

To verify the firmware has been updated, go to **Tools › Reprogram › Get Device Information** and verify the new versions are listed.

Updating the DXM Processor Firmware

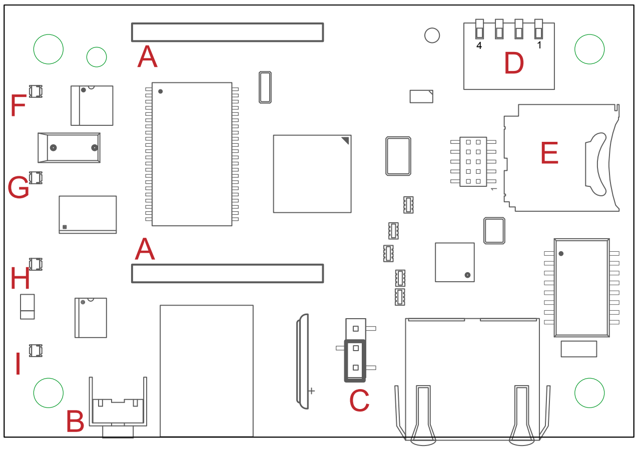
There are two different update procedures, depending on the DXM firmware version of your device.

Update Your DXM Processor Firmware (Prior to Version 2.0)

To update DXM Processor firmware prior to version 2.0, use the SAM-BA program from MicroChip/Atmel. Following these instructions to update the DXM100 or DXM150 processor firmware.

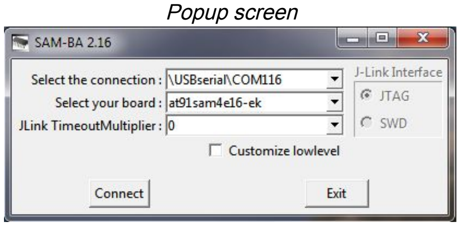
1. Download the SAM-BA software from http://www.microchip.com/developmenttools/productdetails.aspx?partno=atmel+sam-ba+in-system+programmer.
2. Install the SAM-BA program.
3. Set the processor board jumper (jumper C, shown below in the "boot load off" position).

Processor board



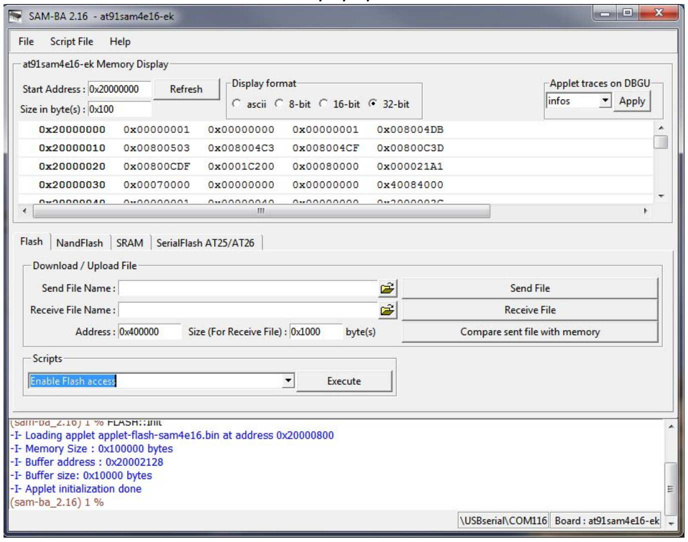
   a. Disconnect the DXM Controller from its power supply.
   b. Open the hardware cover.
   c. Using your fingers or tweezers, move the jumper to the "boot load on" position (jumper on the top two pins).
   d. Connect the DXM back to its power supply.
   e. The lower left LED on the I/O base board is solid when power is turned on. After the LED begins flashing, remove power.
   f. Move the jumper back to its original position.
   g. Replace the hardware cover.
   h. Connect the DXM back to its power supply.
4. Launch the SAM-BA program. Select the COM port and correct board. Click **CONNECT**.

Popup screen



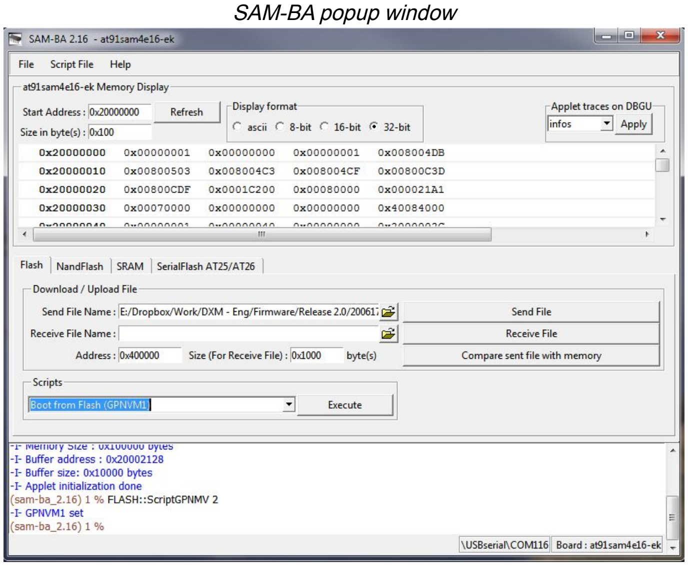
   The SAM-BA program attempts to automatically detect the COM port and the correct device.
5. On the **SCRIPTS** pull-down m] enu select **ENABLE FLASH ACCESS**. Click **EXECUTE**.

SAM-BA popup window



6. In the **SCRIPTS** pull-down menu, select **BOOT FROM FLASH (GPNVM1)**. Click **EXECUTE**. Click **EXECUTE** again if the message indicates it failed.
7. In the **Flash** tab, click on the folder icon for the **Send File Name** field. Select the boot load file (must be a *.bin file) and click **SEND FILE**. The file is: DXM PROCESSOR FIRMWARE V2.02 or go to the software section of the Wireless Reference Library on www.bannerengineering.com.

SAM-BA popup window



The load process takes a few seconds.
8. After the load is complete, the program asks if you want to lock the flash region. Click **NO**.
9. Close the SAM-BA bootloader program.
10. Cycle the power to the DXM.

The new code should now be running and the LEDs should be on.

Updating Your DXM Processor Firmware (Version 2 or Later)

DXMs with processor firmware version 2.0 or later have a built-in boot loader program to update the firmware. Use the configuration software version 3 or later, the BannerCDS webserver, or manually write the files on the SD card to update the firmware.

The new firmware file loads into the **BOOT** directory of the SD card on the DXM. The configuration software or BannerCDS website handles the reprogramming process automatically. During the programming process, the internal LEDs on the processor board indicate the status of the programming.

Update process overview

Reprogramming Step	Approximate time required	Description
Loading new firmware file (*.HEX)	Configuration software: 2 minutes over Ethernet or 15 minutes over USB BannerCDS: 2 minutes over Ethernet or 5 minutes over Cellular	Send the new firmware image to the DXM. After the new image is on the device, the controller resets. LED3 is red during the loading process.
Verify the contents of the new firmware file	1 minute	When the DXM finds a file that should be installed, LED4 (amber) flashes at about a 1 second rate while the contents of the file are validated.
New firmware file is valid		After validation successfully completes, LED4 is on (amber).
New firmware file is being loaded	2 minutes; do not remove power to the DXM during the programming process.	LED3 (red) blinks approximately once per second. LED3 continues to blink during the application programming process.
Finished		After programming has completed, the DXM resets and begins running the new firmware

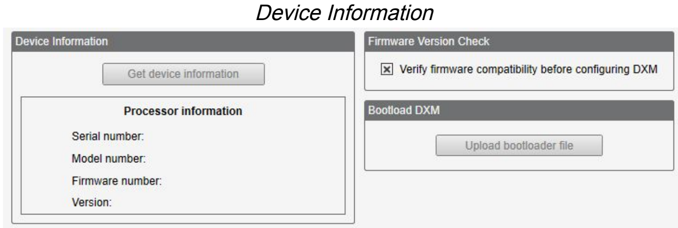
The firmware file names follow an 8.3 filename convention. The first 5 characters are the firmware part number in hexadecimal; the last 3 characters of the part number are the major/minor version number. For example, if `30FA9052.hex` is the firmware programming file, 200617 decimal (30FA9 hex) is the firmware part number and 0.5.2 (0502) is the decoded version number.

Update the DXM Processor Firmware Using the DXM Configuration Software

To update your processor firmware using the DXM Configuration Software, follow these instructions.

1. Using the DXM Configuration Software version 3 or later, connect to the DXM via USB[(1)] or Ethernet.
   File loads to the DXM will take about 15 minutes using USB or approximately 2 minutes using Ethernet.
2. On the configuration software, go to **Settings › General › Device Information** to verify the current firmware version.
   You must load a different version with the same firmware number for the boot loader to operate. Download firmware files from the Banner website.

Device Information



3. Under **Settings › Reprogram**, click **Select upgrade file** to select the firmware file to program.

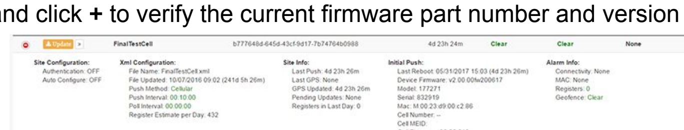
After the file load is completed, the DXM restarts and loads the new firmware file. It takes about 2 minutes to complete the programming process. The device reboots when finished. Verify the firmware has been updated, under **Settings › General › Device Information**.

Update Your Processor Firmware Using the BannerCDS Website

To update your processor firmware (version 2.0 or later) using the DXM website, follow these instructions.

To use the website to update the firmware file, first configure the DXM to push data to the website.

1. Go to **Dashboard › Sites** and click **+** to verify the current firmware part number and version on the DXM.

   Data collected from the DXM is displayed.
2. From the main **Dashboard › Sites** screen, click on **Update**.
   A popup box appears.

(1) While the file download is in process over a USB connection, do not use other applications on the PC. After the DXM reboots for a firmware update, the USB port may be unresponsive. Clear the connection by disconnecting the USB cable and restarting the DXM Configuration Software.

3. Set the **Communications Type** to **Push Reply**, and set the **Update Type** to **Firmware file**.
4. Choose the appropriate **Upload File** (*.HEX) and click **Queue**. Click **Close**.

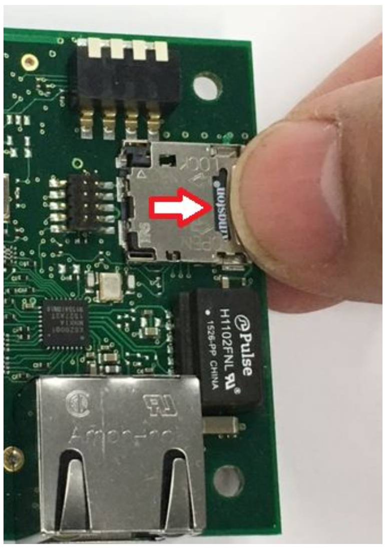
   At the next scheduled push interval, the DXM retrieves the new firmware file. The new firmware file must be the same part number of firmware that is currently in the DXM.

Update Your Processor Firmware Manually

To manually update your processor firmware (version 2.0 or later) using SD card, follow these instructions.

The firmware file can manually be put on the SD card in the BOOT directory (must have version 2.0 or later on the DXM).

1. Disconnect the DXM from its power supply.
2. Remove the micro SD card from the DXM.




   a. Open the cover housing to the DXM.
   b. Use your fingernail to slide the top metal portion of SD card holder.
   c. The metal cover hinges upward, allowing access to remove the SD card.
   d. Press down on the SD cover and slide back into position to close the SD card holder.
3. Insert the micro SD card into an SD card reader to access the data from a PC.
4. Load the new firmware file (*.hex) into the BOOT directory of the micro SD card.
5. Re-insert the micro SD card into the DXM by sliding the card into the holder.
6. Reconnect the DXM to its power supply.

   The automatic boot process should begin. If the boot process does not begin, verify the firmware file is correct and it is a different version than what is currently installed on the device.

Document title: Updating Your DXM Processor Firmware
Part number: b_4474194
Revision: C
Original Instructions

DXM Configuration Software v4 Release Notes

The following updates are included in the DXM Configuration Software v4 (content ID b_4496867). These release notes are content ID b_4498817.

Date	Version	Release Notes
16 Jan 2019	4.0.3	Initial release of the version 4 software. Note that the official name of all configuration tools changed to "configuration software". Reserve "tools" to represent hardware.

Continued on page 77

Continued from page 76

Date	Version	Release Notes
18 Feb 2019	4.0.11	Changed application name from "DXM Configuration Tool" to "DXM Configuration Software". If you are upgrading from v4.0.3 to v4.0.11, you may need to manually delete the "DXM Configuration Tool v4" shortcut from your desktop. A new shortcut named "DXM Configuration Software v4" will be created when you update. Added "Clone" and "Delete" functionality to Action Rules, Register Mapping, and Scheduler screens. Click any table row on one of these screens, then click the appropriate button to either duplicate or remove that row. Register View improvements: Bug fixes for register names and read/write operations; "Register source" options are now updated based on the selected Device Model. Updated labels throughout application for consistency and clarity.
1 Mar 2019	4.0.14	Users can now convert DXM700 configurations to DXM100/DXM150 configurations via the Model Select screen Cloud permissions and Cyclic push local register parameters have been combined into a single parameter called Cloud settings Various UI improvements and bugfixes
22 Mar 2019	4.0.18	Action rules and Register Mapping rules can now be reordered within their respective tables using the arrow buttons on the table headers. Updated validation for configuration file load. Any configuration file can now be loaded (regardless of which DXM device model it targets) if its Register IDs are valid for the currently-selected device model in the software.
10 Apr 2019	4.0.21	The Edit Register panel now includes a "Clear Register" button, which resets the selected register to its default parameters. Added warnings when configuring unusually fast Modbus RTU messaging rates. Assorted bug fixes for read/write rules, display registers, and I/O Board settings.
15 May 2019	4.1.7	Added support for DXM100-A1 devices; UI improvements for various controls
22 Jul 2019	4.2.5	Fixed localization issue on Modbus Timeout setting. Bugfixes and improvements for I/O Board settings. Fixes for Register Viewer handling of virtual registers
15 Apr 2020	4.8.4	Added Simple Setup Mode to perform automatic network discovery and start pushing data to the cloud Added support for DXM1000 and DXM1200 devices Added support for AWS IoT cloud platform
1 Dec 2020	4.10.9	Added new **Solutions Guide mode**, which allows for easy configuration of Solutions Guide sensors to push to the cloud Added support for PROFINET Software is now able to load and update outdated configuration files to the latest format Added **Signed 16-bit Integer** register type Added support for additional sensor models in **Simple Setup mode** Improved in-application status notifications and error messages Renamed **DXM-AG1** device model to **DXM-A1**
7 Jun 2021	4.10.28	Converted the MQTT to AWS IoT Core; Improved AWS IoT Core interface Added DXM1500 and DXM-R90x support Device comms and configuration file bug fixes The Push method drop-down was replaced by a pair of radio buttons (denoting HTTP Cloud Push and AWS IoT Core)
15 Jun 2021	4.10.31	**General** • Set RTU read/write rule frequency min value to 10ms • Removed lower-bound timer restrictions on MQTT publish rules **DXM-R90x** • Increased DXM discovery ping timeout • Fixed issue in which incorrect UART bus was being considered for establishing port-wise minimum frequency for RTU rules
21 Jun 2021	4.10.33	**DXM-R90x** • Set default RTU time between messages and read/write rule frequencies to 50ms • Updated XML element format for RTU Device settings in configuration files

Continued on page 78

Continued from page 77

Date	Version	Release Notes
3 Dec 2021	4.10.46	**General** • Increased default RTU Read and Write rule frequencies from 50ms to 1s (see DXMXML v1.2.52) • When Advanced Settings checkbox is unselected, hidden parameters are now reverted to values needed for Banner CDS HTTP push • Added MQTT Publisher QoS parameter **DXM-R90x** • Improved DXM autodetect process with nonstandard subnet mask handling and associated logging • Ports [0, 4] are now available for RTU Read/Write rules • Updated RTU read/write interface layouts • Added Server Port 0 parameters to Settings tab • Added bootloading and Reboot With Configuration Bypass behavior
4 Apr 2022	4.12.0	**General** • Increased default RTU Read and Write rule frequencies from 50ms to 1s (see DXMXML v1.2.52) • When the Advanced Settings checkbox is unselected, hidden parameters are now reverted to values needed for Banner CDS HTTP push • Added MQTT Publisher QoS parameter **DXM-R90x** • Improved DXM autodetect process with nonstandard subnet mask handling and associated logging • Ports [0, 4] are now available for RTU Read/Write rules • Updated RTU read/write interface layouts • Added Server Port 0 parameters to the Settings tab • Added bootloading and Reboot With Configuration Bypass behavior
25 Jul 2022	4.13.5	Added support for DXM-R904K device model Config file is now automatically loaded into the configuration software Additional error checking on script upload behavior UDP Listener now accepts a port number, mirroring TCP/IP device connection behavior Added MultiHop device addressing option to ISM Register View operations Improved RS-485 comms behavior on DXM-R90x
13 Sep 2022	4.13.8	Added support for EIP Endianness Swap setting
2 Nov 2022	4.13.10	Added PROFINET control for DXMR90-4K; improved error checking on DXMR90 hexfile uploads
Feb 2023	4.15.0	Added support for DXMR110-8K device model Added Decoder action rules for copying partial contents of registers Modbus TCP read/write rules can now specify between Remote and Input register types Fixed Register View LED register mappings for DXMR90x
11 May 2023	4.16.2	Fixed backward compatibility issue with XML file generation Corrected device model auto-detect behavior for certain DXM models Removed Simple Setup and Solutions Guide configuration modes Removed deprecated Email Notification settings SMS Notification settings are now supported only on DXM100, DXM100-A1, and DXM150 models using the SXI-LTE-001 cellular module
12 Dec 2023	4.16.4	Added support for DXMR90-4M device model
31 Jan 2024	4.17.7	Added support for the DXM1200-B2 and DXM1200-X2 device models
7 Feb 2024	4.18.0	UI improvements for DXM1200B2 and DXM1200X2-specific parameters Added reference images to Register View for LCD board behaviors on DXM1200 devices Fixed bug that incorrectly hid device connection controls when switching the selected DXM model
13 Feb 2024	4.18.1	Bug fixes for device settings on DXM100, DXM100A1, DXM150, DXMR904K, and DXMR90x models

Continued on page 79

Continued from page 78

Date	Version	Release Notes
21 Mar 2024	4.18.8	Updated device detection behavior and default UART Bus configuration for DXM1200B2 and DXM1200X2 device models Bug fixes for both AWS IoT Core and Sparkplug B MQTT configuration behavior MQTT encryption and certificate use can now be enabled individually Improved in-application messaging regarding certificate files and authentication credentials, which are configured separately from the main device configuration when a DXM is connected
19 Dec 2024	4.19.2	Added support for DXMR90-X1E device model Modbus RTU rule limits are now enforced on a per-port basis for DXM models with multiple ports Clarified MQTT nomenclature throughout application

Contact Us

Banner Engineering Corp. headquarters is located at: 9714 Tenth Avenue North | Plymouth, MN 55441, USA | Phone: + 1 888 373 6767

For worldwide locations and local representatives, visit www.bannerengineering.com.

LinkedIn

X (formerly Twitter)

Facebook