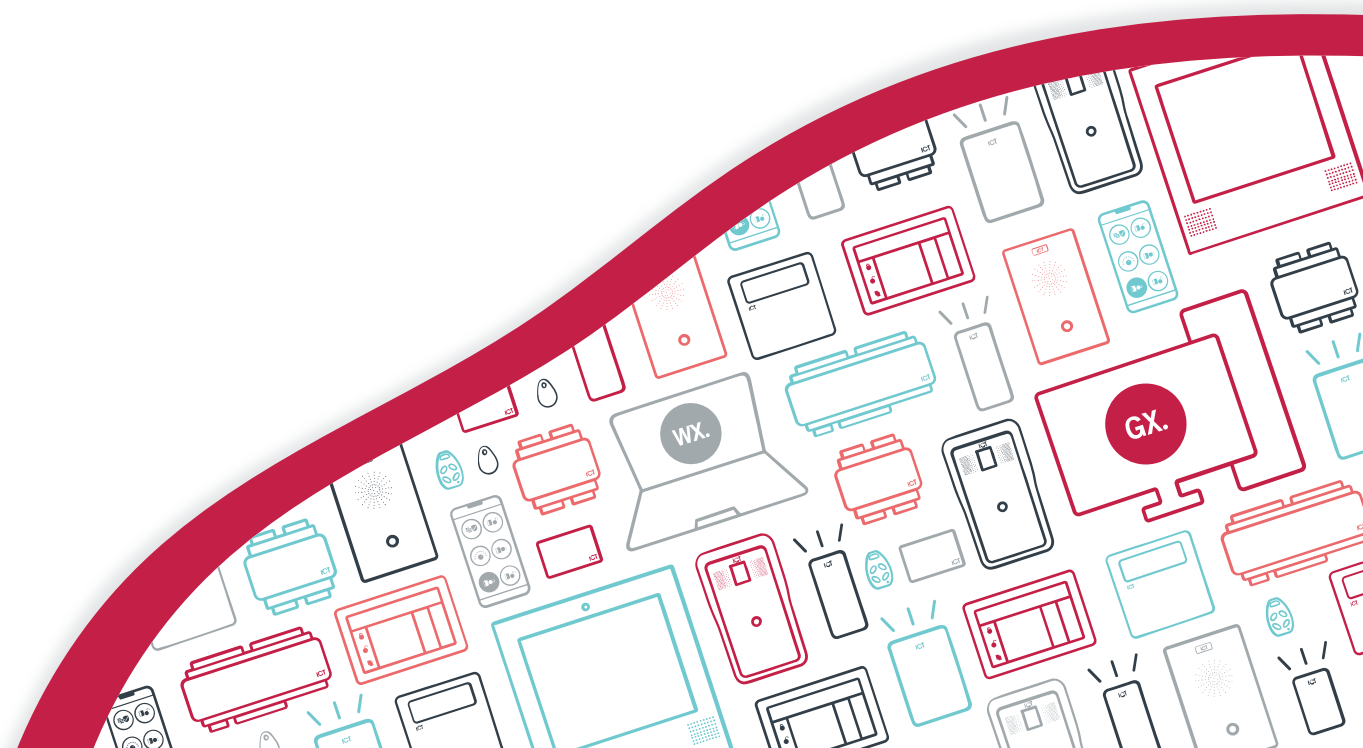
AN-353

Protege GX Modbus Client Integration

Application Note



Last Published: 18-Apr-23 01:33 PM

Contents

Introduction

Modbus is an industry standard communication protocol that is widely used for building automation. Modbus devices communicate using a client-server architecture in which only one device (the client) can initiate transactions (called queries). The other devices (servers) respond by supplying the requested data to the client, or by taking the action requested in the query.

Protege GX offers two methods of integrating with Modbus systems:

Modbus Server Integration	Modbus Client Integration
One or more Protege GX controllers act as Modbus **servers**.	One Protege GX controller acts as a Modbus **client**.
The Modbus client reads and writes Protege GX inputs and outputs.	The controller reads and writes digital inputs (inputs), coils (outputs) and registers (data values) on connected Modbus server devices.
Communication over ethernet (TCP/IP).	Communication over RS-485.
Documented in Application Note 023: Protege GX Modbus Server Integration.	Documented in Application Note 353: Protege GX Modbus Client Integration.

This application note covers the **Modbus client integration**, allowing a Protege GX controller to monitor and control building automation devices such as thermostats, humidity controls and light sensors.

Previously the Modbus client was referred to as the "master" and server devices were referred to as "slave" devices. This terminology has been deprecated by the Modbus Organization (see this press release).

Prerequisites

The following must be installed and operational:

Component	Firmware Version	Notes
Protege GX Controller	2.08.1355 or higher	The controller must have an available RS-485 reader port. (i.e. not being used for door control).

Supported Modbus Devices

This integration can be used to monitor and control Modbus server devices that meet the following requirements:

- The server device must have an RS-485 port, or be able to connect to an RS-485 interface module.
- The server device must communicate using the Modbus RTU protocol.

It is the integrator's responsibility to validate any devices that will be integrated with Protege GX.

Capabilities

Each Protege GX controller has the following capabilities:

- Scan up to 32 Modbus server devices connected via RS-485.
- Read/Write a maximum of 16 items (digital inputs, coils and registers) per server device, of which up to 8 can be registers.

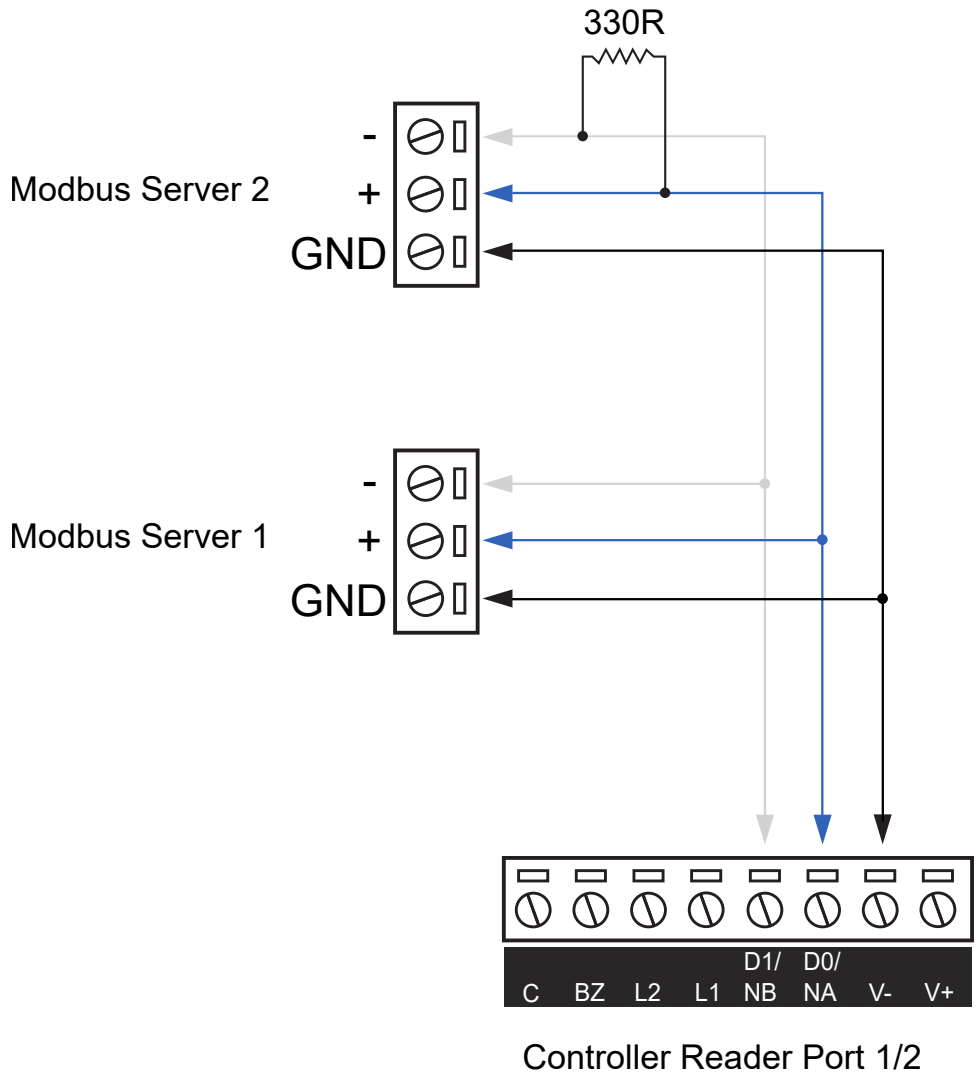
The following table identifies which Modbus function codes are supported by the integration:

Function Type			Function	Function Code	Supported
Data Access	Bit Access	Physical Discrete Inputs	Read Discrete Inputs	2	✓
		Physical and Virtual Coils/Outputs	Read Coils	1	✓
			Write Single Coil	5	✓
			Write Multiple Coils	15	✓
	16-Bit Access	Physical Input Registers	Read Input Registers	4	✓
		Internal Registers or Physical Output Registers	Read Multiple Holding Registers	3	✓
			Write Single Holding Register	6	✓
			Write Multiple Holding Registers	16	✓
			Read/Write Multiple Registers	23	✓
			Mask Write Register	22	✗
			Read FIFO Queue	24	✗
	File Record Access		Read File Record	20	✗
			Write File Record	21	✗
Diagnostics			Read Exception Status	7	✗
			Diagnostic	8	✗
			Get Com Event Counter	11	✗
			Get Com Event Log	12	✗
			Report Server ID	17	✗
			Read Device Identification	43	✗
Other			Encapsulated Interface Transport	43	✗

Physical Connections

In this integration, the controller communicates with the Modbus server devices via one of its RS-485 reader ports. If you are using an RS-485 interface module, connect it to the server device as per the installation manual.

Prior to connecting each device to the network, ensure that it has a unique server address in the range 1-32.



Controller Reader Port 1/2

A termination resistor should be placed at the end of the line. This is a 330 ohm resistor wired across the NA (+) and NB (-) terminals of the last connected device.

Programming Steps

This section covers the programming required to set up this integration. In summary:

1. Enable the Modbus protocol on the controller's RS-485 reader port.
2. Create data values to represent the registers.
3. Create outputs to represent the digital inputs and coils.
4. Create analog expanders to represent the server devices. Enter commands to map the data values and outputs to Modbus registers, digital inputs and coils.
5. Complete any further programming necessary for monitoring and controlling the Modbus devices.

Enabling the Modbus Protocol

The controller communicates with the Modbus server network using one of its onboard reader ports. First, enable the controller's onboard reader expander if you have not already.

1. In Protege GX, navigate to **Sites | Controllers**.
2. In the **Configuration** tab, set **Register as reader expander** to an available reader expander address.
3. Navigate to **Expanders | Reader expanders** and add a reader expander record.
4. Set the **Physical address** to the address used above.

Then you must enable Modbus communications on the reader port that the Modbus network is connected to.

1. In **Expanders | Reader expanders**, select the controller's onboard reader expander.
2. Expand the **Commands** section and enter one of the following commands:
   - If the Modbus network is connected to reader port 1:
     ```
     PortOne = 17
     ```
   - If the Modbus network is connected to reader port 2:
     ```
     PortTwo = 17
     ```
3. Save the changes.
4. Wait for the changes to be downloaded to the controller, then right click on the reader expander record and click **Update module**.

Adding the Data Values

One data value record is required for each Modbus register that is being monitored or controlled.

1. Navigate to **Automation | Data values**.
2. Click **Add** to add a new data value and give it a descriptive **Name** (e.g. Fridge A Temperature Control).
3. If the controller will write to this register (e.g. control the temperature), you may wish to set a **Preset value**. Enter a value, then enable one or both of the following:
   - **Preset power up**: The register will be set to the preset when the controller first powers up.
   - **Preset value**: The register will be set to the preset whenever any programming is downloaded to the controller.

   These options are not used for registers that the controller is only reading from.
4. Save the data value.
5. Note the **Database ID** of the new data value record.

If you wish to display the data values on a status page or floor plan you will need to create a variable record for each one in **Automation | Variables**.

Adding the Outputs

One virtual output record is required for each digital input or coil that is being monitored and each coil that is being controlled.

1. Navigate to **Expanders | Output expanders**.
2. From the toolbar, select the **Controller** that is used for the Modbus integration.
3. Click **Add** to add a new output expander.
4. Enable the **Virtual module** option.
5. Set the **Physical address** to a value above existing physical expanders (e.g. 32).
6. Click **Save**.
7. In the popup window, disable **Add trouble inputs** then click **Add now**.
8. Navigate to **Programming | Outputs** and locate the expander's new outputs.
9. Give each output a descriptive name (e.g. Fridge 1 On/Off Status) and note the **Database ID** of the new record.
10. If necessary, create more output expanders to support additional outputs.

Adding the Analog Expanders

Each Modbus server device in the network is represented by an analog expander record in Protege GX.

1. Navigate to **Expanders | Analog expanders**.
2. From the toolbar, select the **Controller** that is used for the Modbus integration.
3. Click **Add** to create a new analog expander. Give it a descriptive name (e.g. Fridge 1).
4. Enable the **Virtual module** option. This will prevent the controller from generating health status messages for this module.
5. Set the **Physical address** to the Modbus server address of the connected device.
6. Click **Save**.
7. In the popup window, set the **Outputs** to 0 and disable **Add trouble inputs**. Then click **Add now**.

Mapping the Registers, Digital Inputs and Coils

The following commands are used to map each component to the relevant record in Protege GX. Enter each command on a new line. You can map up to 16 records per device, of which up to 8 can be registers.

- To read a register, enter:

  `MbsReg = R,x,y`

  where **x** is the address of the register on the Modbus server device and **y** is the Database ID of the data value to map it to.
- To read and write to a register, enter:

  `MbsReg = RW,x,y`

  where **x** is the address of the register on the Modbus server device and **y** is the Database ID of the data value to map it to.
- To read a coil or digital value, enter:

  `MbsBit = R,x,y`

where **x** is the address of the coil or digital input on the Modbus server device and **y** is the Database ID of the output to map it to.

- To read and write to a coil or digital value, enter:

  `MbsBit = RW,x,y`

  where **x** is the address of the coil or digital input on the Modbus server device and **y** is the Database ID of the output to map it to.

As soon as at least one command has been entered and saved the controller will begin to scan for a Modbus device with the matching address.

Some examples of commands are:

- Read register 3 and store it as data value 16: `MbsReg = R,3,16`
- Read and write register 15 and store it as data value 17: `MbsReg = RW,15,17`
- Read digital input 5 and store it as output 433: `MbsBit = R,5,433`
- Read and write coil 1 and store it as output 434: `MbsBit = RW,1,434`

Downloading Data Values to the Controller

Because the integration is programmed using commands, the data values are not automatically downloaded to the controller. Some additional configuration is required to assign the data values to the controller so that they will be downloaded by the software.

The first four data values on each server device can be assigned to the analog expander record.

1. Select the relevant analog expander.
2. In the **Channel 1** tab, assign the first data value as the **Channel 1 data value**.
3. Repeat for **Channel 2**, **Channel 3** and **Channel 4**.
4. Click **Save**.

The remaining four data values can be downloaded to the controller using a value compare programmable function. This function is only used to associate the data values with the controller and does not need to be started or run.

1. Navigate **Automation | Programmable functions**.
2. Select the relevant **Controller** in the toolbar.
3. Add a new programmable function with a name that describes its purpose (e.g. Function to Download Data Values for Modbus Integration).
4. Set the **Type** to Value compare.
5. In the **Value compare** tab, select the **Output to enable this function**. This can be any output on the controller which will control this integration. You may wish to use a spare virtual output for this purpose.
6. Use the **Analog input data variable register**, **Set point data value**, **Hysteresis data value** and **Hysteresis time data value** fields to select the four remaining data values.
7. Click **Save**.

Once these steps have been completed all of the data values will be downloaded to the correct controller and can be used to display and control the registers on the Modbus server devices.

The **Controller** column in the data values list may display the incorrect controller. This is a display issue only and does not affect what is downloaded.

Further Programming

Protege GX provides a variety of programming options for monitoring and controlling the Modbus registers, digital inputs and coils.

Manual Monitoring and Control

To monitor outputs, add them to a **status page** or **floor plan**. If you have write control, you can right click on the output and select **Activate**, **Deactivate** or **Activate timed**.

To monitor a data value, you will need to create a variable:

1. Navigate to **Automation | Variables**.
2. Add a new variable with an appropriate name.
3. Set the **Scale** and **Offset** to convert the data provided by the Modbus register into a human-readable number. This will depend on the server device.
4. Set the **Data value** which this variable will display.
5. Click **Save**.

Now you can add this variable to a floor plan as normal. If you have write control, you can right click on the variable and set the value.

Automated Monitoring and Control

A number of programmable functions are available to automatically monitor and control devices. These are available in **Automation | Programmable functions**.

- **Logic control**: Controls an output or output group based on the state of one or two triggering outputs. Several logical operations are available.
- **Ripple output**: Activates a series of outputs in a ripple pattern based on a single triggering output. This can be used to stage large current devices and multiple lighting circuits.
- **Input follows output**: Controls an input based on the state of an output. Can be used to activate alarms based on an output state.
- **Value compare**: Compares two data values and activates outputs based on their relative quantities. For example, this can be used to control lighting circuits based on daylight sensor inputs.
- **Register counter**: Increments or decrements a data value based on the state of an input.
- **Average**: Calculates the average of up to 8 input data values and writes it to an output data value. For example, this can be used to take an average of multiple temperature sensors.
- **Variable output compare**: Compares a single input data value with a series of 'fixed point' data values. When the input value reaches each fixed point an output data value is updated with a known quantity.
- **Floor temping**: Manages an air tempering system with single-stage heating and cooling.
- **Roof top heat pack**: Manages an HVAC system with up to 4 stages of heating and cooling and two stages of dehumidification.

For more information about programming and using these functions, see the Protege GX Operator Reference Manual.

Designers & manufacturers of integrated electronic access control, security and automation products.
Designed & manufactured by Integrated Control Technology Ltd.