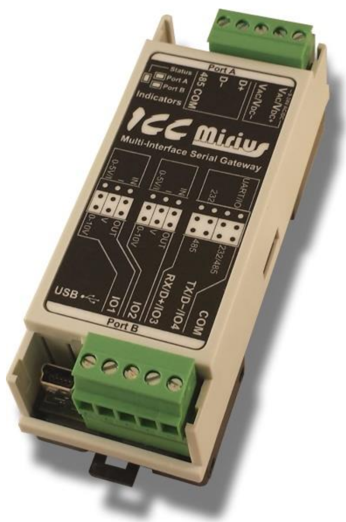
INDUSTRIAL CONTROL COMMUNICATIONS, INC.

Instruction Manual

Mirius
Multi-Interface Serial Gateway



December 6, 2024

Mirius User's Manual

Printed in U.S.A.

NOTICE TO USERS

Industrial Control Communications, Inc. reserves the right to make changes and improvements to its products without providing notice.

Industrial Control Communications, Inc. shall not be liable for technical or editorial omissions or mistakes in this manual, nor shall it be liable for incidental or consequential damages resulting from the use of information contained in this manual.

INDUSTRIAL CONTROL COMMUNICATIONS, INC.'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE-SUPPORT DEVICES OR SYSTEMS. Life-support devices or systems are devices or systems intended to sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling and user's manual, can be reasonably expected to result in significant injury.

No complex software or hardware system is perfect. Bugs may always be present in a system of any size. In order to prevent danger to life or property, it is the responsibility of the system designer to incorporate redundant protective mechanisms appropriate to the risk involved.

This user's manual may not cover all of the variations of interface applications, nor may it provide information on every possible contingency concerning installation, programming, operation, or maintenance.

The contents of this user's manual shall not become a part of or modify any prior agreement, commitment, or relationship between the customer and Industrial Control Communications, Inc. The sales contract contains the entire obligation of Industrial Control Communications, Inc. The warranty contained in the contract between the parties is the sole warranty of Industrial Control Communications, Inc., and any statements contained herein do not create new warranties or modify the existing warranty.

Any electrical or mechanical modifications to this equipment without prior written consent of Industrial Control Communications, Inc. will void all warranties and may void any UL/cUL listing or other safety certifications. Unauthorized modifications may also result in equipment damage or personal injury.

Usage Precautions

Operating Environment

- Please use the interface only when the ambient temperature of the environment into which the unit is installed is within the following specified temperature limits:

  Operation: -10 ~ +50°C (+14 ~ +122°F)
  Storage: -40 ~ +85°C (-40 ~ +185°F)

- Avoid installation locations that may be subjected to large shocks or vibrations.
- Avoid installation locations that may be subjected to rapid changes in temperature or humidity.

Installation and Wiring

- Proper ground connections are vital for both safety and signal reliability reasons. Ensure that all electrical equipment is properly grounded.
- Route all communication cables separate from high-voltage or noise-emitting cabling (such as ASD input/output power wiring).

Compliance Statements

CE Compliance Statement

This device complies with the following European Directives:

- 2014/30/EU Electromagnetic Compatibility (EMC)
- 2014/45/EU Low Voltage (LVD)
- 2011/65/EC on the Restriction of Hazardous Substance (RoHS2)

This device conforms to the following Standards:

- EN 55032:2015/AC:2016-07 (EMC Emissions)
- EN 55035:2017 (EMC Immunity)
- EN 62368-1:2014+A11:2017 (Safety)
- EN 50581:1010 (RoHS)

The technical documentation required to demonstrate that the products meet the requirements of the aforementioned directives has been compiled and is available for inspection by the relevant enforcement authorities.

FCC Compliance Statement

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

NOTE 1: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

NOTE 2: Any changes or modifications to this unit not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

ISED Compliance Statement

This Class A digital apparatus complies with Canadian ICES-003.

Cet appareil numérique de la classe [A] est conforme à la norme NMB-003 du Canada.

CAN ICES-3 (A)/NMB-3(A)

TABLE OF CONTENTS

1. Introduction

Congratulations on your purchase of the ICC Mirius Gateway. This gateway allows information to be transferred seamlessly between various serial networks as well as physical I/O. The gateway supports one RS-485 port and one multi-interface port providing two dedicated I/O terminals and two jumper-selectable terminals for I/O or serial communication via RS-485, RS-232, or UART. The gateway also hosts a USB interface for updating and configuring the gateway via a PC.

Before using the gateway, please familiarize yourself with the product and be sure to thoroughly read the instructions and precautions contained in this manual. In addition, please make sure that this instruction manual is delivered to the end user of the gateway, and keep this instruction manual in a safe place for future reference or unit inspection.

For the latest information, support software and firmware releases, please visit http://www.iccdesigns.com.

Before continuing, please take a moment to ensure that you have received all materials shipped with your kit. These items are:

- Mirius Gateway in DIN rail mountable plastic housing
- Documentation CD-ROM
- Two wall suspension elements
- USB cable

Note that different gateway firmware versions may provide varying levels of support for the various protocols. For optimal performance, always ensure that you are using the latest version of the ICC Configuration Studio and included firmware.

This manual will primarily be concerned with the gateway's hardware specifications, installation, wiring, configuration and operational characteristics.

To maximize the abilities of your new gateway, a working familiarity with this manual will be required. This manual has been prepared for the gateway installer, user, and maintenance personnel. With this in mind, use this manual to develop a system familiarity before attempting to install or operate the gateway.

2. Features

Supported Protocols

The gateway provides support for a variety of serial based fieldbus protocols. Refer to section 9 for detailed information on each specific supported driver.

Supported Baud Rates

The gateway supports the following baud rates on its RS-485 and multi-interface ports:

- 300
- 600
- 1200
- 2400
- 4800
- 9600
- 19200
- 38400
- 57600
- 76800
- 115200

Note that not all protocols support every baud rate listed above. Refer to section 9 for more information.

Supported I/O

The gateway has two dedicated universal I/O terminals and two shared general-purpose I/O terminals. Each I/O channel can be configured as an analog input, pulse output, digital input, digital output, or pulse input. Refer to section 10 for detailed information on each channel's I/O capabilities.

Power Options

The gateway accepts 9 - 24V power and supports both AC and DC supplies. The power input is full-wave rectified and electrically isolated. Power may also be provided via USB, for convenience when configuring the unit at a workstation.

Isolation

Each port is galvanically isolated from one another and from the power supply, providing three independent power domains. This eliminates any potential ground loop or induced ground noise issues between communication equipment and power supplies.

Field-Upgradeable

As new firmware becomes available, the gateway can be upgraded in the field by the end-user. Refer to section 8.3 for more information.

USB Interface

The gateway can be connected to a PC via a USB mini type-B cable. This simultaneously supplies power while providing the ability to configure the gateway, monitor data, and update firmware on the device using the ICC Configuration Studio. Refer to section 8.3 for more information.

USB Virtual COM Port Interface

The gateway can be configured to enumerate as a USB virtual COM port, allowing a PC to directly communicate to the gateway using any supported serial protocol, tunnel through the gateway to communicate on the connected RS-485 bus, or capture network traffic on the RS-485 port without impacting communications. Refer to section 8.3.5 for more information.

Network Configuration Parameters

Dynamic runtime parameters located in the gateway's internal database allow quick, on-the-fly configuration of the gateway's Port A RS-485 port via communications or USB, allowing a user to select the desired protocol, baud rate, parity, address, etc.

Persistent User Parameters

General purpose, non-volatile parameters located in the gateway's internal database are available for custom calibration data, serial numbers, device IDs, etc.

User-Configurable Network Timeouts

The gateway can be configured to perform a specific set of actions when network communications are lost. This allows each address in the database to have its own unique "fail-safe" condition in the event of network interruption (support for this feature varies depending on the protocol). Refer to section 8.3.10.2 for more information.

PLC-Style Database Manipulation Operations

A variety of database logic operations are included which provide PLC-style manipulation of database values. Categories such as logical, arithmetic and filtering operations allow for autonomous control over value modification and data movement within the database. High-level signal conditioning is also realizable via the construction of compound formulas derived from the elemental building block operations provided. Refer to section 8.3.10.3 for more information.

Flexible Mounting Capabilities

The gateway includes all hardware for panel/wall and DIN-rail mounting capabilities. Refer to section 6.1 for more information.

3. Gateway Concepts

The Mirius provides simultaneous support for many different communication protocols and physical I/O signals, allowing complex interchanges of data on otherwise unsupported networks. The Mirius is configured using the *ICC Configuration Studio*.

The Mirius supports many unique integration features, making it especially well-suited for OEMs looking to provide a complete and flexible communication solution to a customer. The gateway's Port A RS-485 port may be configured to support multiple protocols simultaneously, however, only one protocol can be active at a time. The network characteristics of the RS-485 port, such as protocol, baud rate, address, etc., can be configured dynamically, at run time, by modifying special Network Configuration Parameters located in the gateway's internal database. These parameters can be mapped to protocol objects, allowing the Mirius's RS-485 port configuration to be done entirely from the OEM's device (i.e. via keypad, touchscreen, or other means). Alternatively, these parameters can be configured by the end user via USB using the light-weight *Network Parameter Utility*.

The heart of the Mirius is its internal database. The database is an 8KB, byte-wise addressable data array. At the end of the 8KB address space, the database also includes 32 bytes of Network Configuration Parameters, 32 bytes of Protocol-Specific Configuration Parameters, and 64 bytes of Persistent User Parameters. This gives a total size of 8320 bytes for the entire database, referred to as DB_{Size} in the protocol driver manuals. The database allows data to be routed from any supported network, including I/O data, to any other supported network.

The other fundamental aspect of the Mirius is the concept of a configurable "service object". A service object is used for any master/client protocol to describe what service (read or write) is to be requested on the network. The Mirius will cycle through the defined service objects in a round-robin fashion; however, the device does implement a "write first" approach. This means that the Mirius will perform any outstanding write services before resuming its round-robin, read request cycle.

Additionally, the database and service objects provide the added benefit of "data mirroring", whereby current copies of data values (populated by a service object) are maintained locally within the device itself. This greatly reduces the request-to-response latency times on the various networks, as requests (read or write) can be entirely serviced locally, thereby eliminating the time required to execute a secondary transaction on a different network.

In order to facilitate the free scaling and conversion of native data values, a user-configurable "multiplier" and "data type" exist for some network configurations. All network values are scaled by a multiplier prior to being stored into the database or after being retrieved from the database. The data type is used to determine how many bytes are allocated for the value in the database, whether the value should be treated as signed or unsigned, and whether the value should be interpreted as an integer or a floating point number upon retrieval.

The Mirius also provides powerful diagnostics and data-monitoring features that allow the user to view port statistics, monitor and edit database values in real time, view the status of service objects, and even capture network traffic without interrupting normal communication. All this is performed via the *ICC Configuration Studio* when connected via USB to a PC.

4. Precautions and Specifications

Rotating shafts and electrical equipment can be hazardous. Installation, operation, and maintenance of the gateway shall be performed by **Qualified Personnel** only.

Qualified Personnel shall be:

- Familiar with the construction and function of the gateway, the equipment being driven, and the hazards involved.
- Trained and authorized to safely clear faults, ground and tag circuits, energize and de-energize circuits in accordance with established safety practices.
- Trained in the proper care and use of protective equipment in accordance with established safety practices.

Installation of the gateway should conform to all applicable **National Electrical Code** (NEC) *Requirements For Electrical Installation*s, all regulations of the **Occupational Safety and Health Administration**, and any other applicable national, regional, or industry codes and standards.

DO NOT install, operate, perform maintenance, or dispose of this equipment until you have read and understood all of the following product warnings and user directions. Failure to do so may result in equipment damage, operator injury, or death.

4.1 Installation Precautions

- Avoid installation in areas where vibration, heat, humidity, dust, metal particles, or high levels of electrical noise (EMI) are present.
- Do not install the gateway where it may be exposed to flammable chemicals or gasses, water, solvents, or other fluids.
- Where applicable, always ground the gateway to prevent electrical shock to personnel and to help reduce electrical noise.

  Note: *Conduit is not an acceptable ground.*

- Follow all warnings and precautions and do not exceed equipment ratings.

4.2 Maintenance Precautions

- **Do Not** attempt to disassemble, modify, or repair the gateway. Contact your ICC sales representative for repair or service information.
- If the gateway should emit smoke or an unusual odor or sound, turn the power off immediately.
- The system should be inspected periodically for damaged or improperly functioning parts, cleanliness, and to determine that all connectors are tightened securely.

4.3 Inspection

Upon receipt, perform the following checks:

- Inspect the unit for shipping damage.
- Check for loose, broken, damaged or missing parts.

Report any discrepancies to your ICC sales representative.

4.4 Maintenance and Inspection Procedure

Preventive maintenance and inspection is required to maintain the gateway in its optimal condition, and to ensure a long operational lifetime. Depending on usage and operating conditions, perform a periodic inspection once every three to six months.

Inspection Points

- Check that there are no defects in any attached wire terminal crimp points. Visually check that the crimp points are not scarred by overheating.
- Visually check all wiring and cables for damage. Replace as necessary.
- Clean off any accumulated dust and dirt.
- If use of the interface is discontinued for extended periods of time, apply power at least once every two years and confirm that the unit still functions properly.
- Do not perform hi-pot tests on the interface, as they may damage the unit.

Please pay close attention to all periodic inspection points and maintain a good operating environment.

4.5 Storage

- Store the device in a well-ventilated location (in its shipping carton, if possible).
- Avoid storage locations with extreme temperatures, high humidity, dust, or metal particles.

4.6 Warranty

This gateway is covered under warranty by ICC, Inc. for a period of 3 years from the date of shipment from the factory. For further warranty or service information, please contact Industrial Control Communications, Inc. or your local distributor.

4.7 Disposal

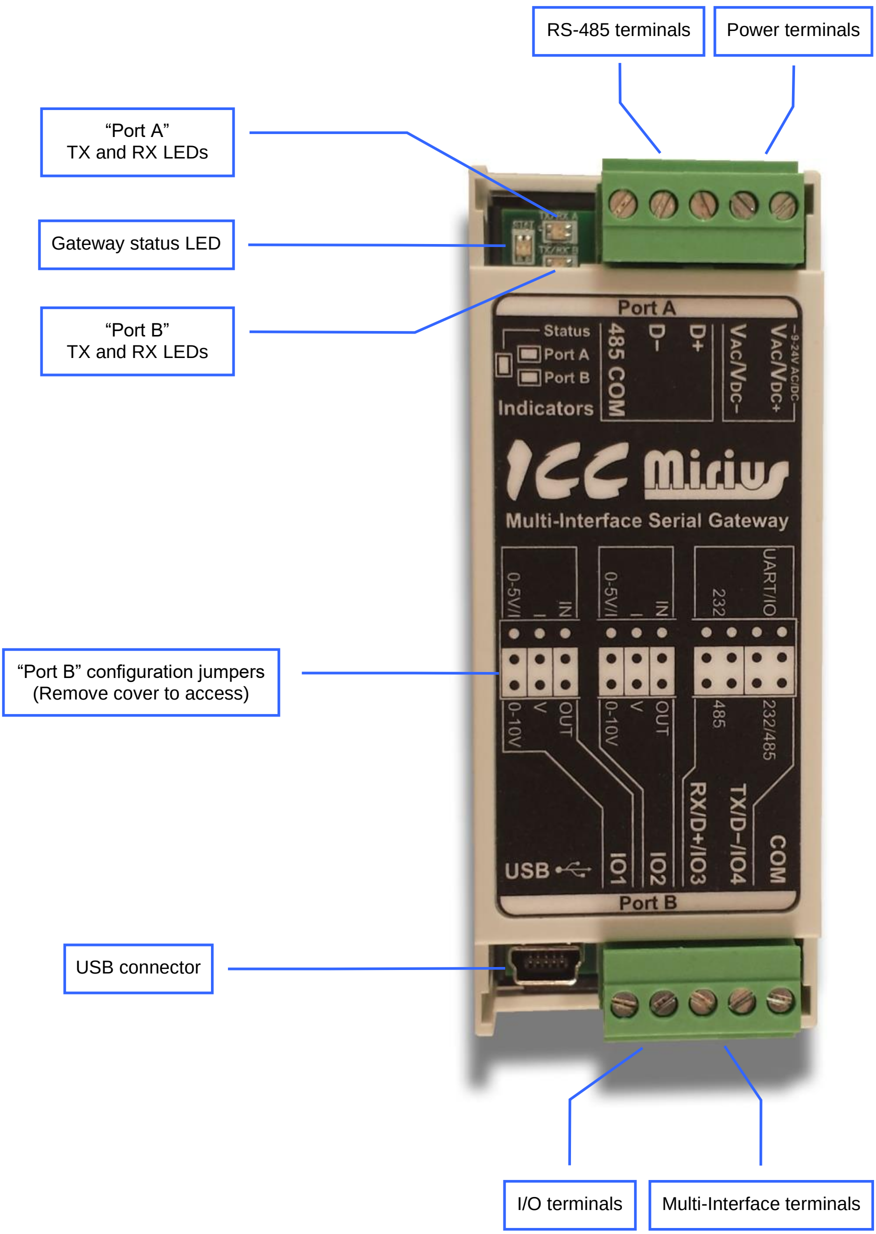
- Contact the local or state environmental agency in your area for details on the proper disposal of electrical components and packaging.
- Do not dispose of the unit via incineration.

4.8 Environmental Specifications

Item	Specification
Operating Environment	Indoors, less than 1000m above sea level, do not expose to direct sunlight or corrosive / explosive gasses
Operating Temperature	-10 ~ +50°C (+14 ~ +122°F)
Storage Temperature	-40 ~ +85°C (-40 ~ +185°F)
Relative Humidity	20% ~ 90% (without condensation)
Vibration	5.9m/s^2 {0.6G} or less (10 ~ 55Hz)
Grounding	Isolated, 3 separate power domains
Cooling Method	Self-cooled

This device is lead-free / RoHS-compliant. Lead Free

5. Gateway Overview



Gateway Overview

5.1 Power Supply Electrical Interface

When the gateway is not plugged into a PC via the USB cable, it must be powered by an external power source. Ensure that the power supply adheres to the following specifications:

Voltage rating .... 9 - 24V AC/DC
Minimum Power rating............ 1.8W
Minimum Current rating.......... 200mA (@9VDC), 75mA (@24VDC)

- Unit to be supplied by LPS or Class 2 power source.
- Typical current consumption of the Mirius when powered from a 24VDC supply is approximately 30 - 70mA, depending on network and I/O load.
- ICC offers an optional 120VAC/9VDC power supply that can be used to power the gateway from a standard wall outlet.
- The power supply must be connected to the gateway’s “Port A” terminal block on the terminals labeled V_{AC}/V_{DC+} and V_{AC}/V_{DC-} as highlighted in Figure 1.

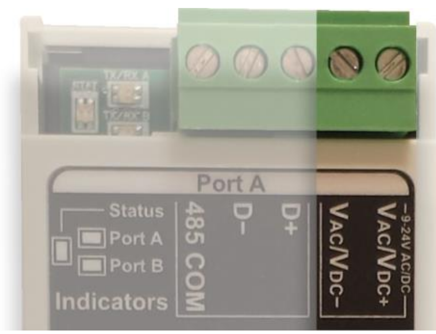


Figure 1: Port A Terminal Block Power Supply Connections

5.2 RS-485 Electrical Interface

The following describes the RS-485 specifications for the gateway's Port A and Port B (when configured for RS-485 mode) serial ports:

Isolated.................................... Yes (1.5 kV)
Unit Loads .... 1/8 Unit Load
Maximum Devices 256
Internal Biasing Resistors....... None
Internal EOL Termination .... None
Failsafe Receiver..................... Full-failsafe (open, shorted, terminated and undriven)
Differential Output Max........... 5.0V (no load)
Differential Output Min............ 2.4V (terminated and fully loaded)

6. Installation

The gateway's installation procedure will vary slightly depending on the mounting method used.

6.1 Mounting the Gateway

The gateway may be mounted on a DIN rail, panel, or wall. A minimum clearance of 6 inches is required for the ventilation slits on both Port A and Port B ends when installing the gateway. The following describes the method for the two mounting options.

6.1.1 DIN Rail Mounting

The base of the gateway's enclosure is designed to quickly and simply lock onto standard DIN rails, complying with the DIN EN 60715 TH35 standard. The release tab located at the bottom allows the gateway to easily be removed from a DIN rail.

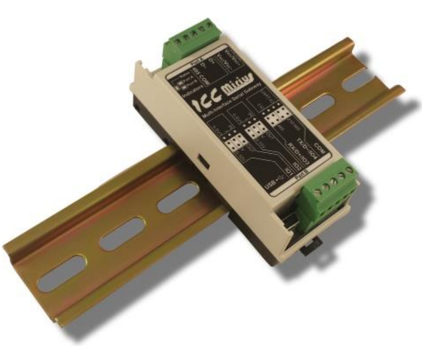


Figure 2: DIN Rail Mounting

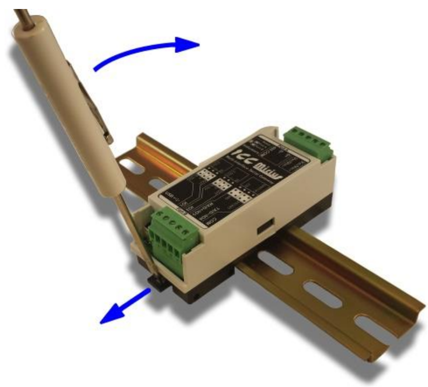


Figure 3: DIN Rail Release

6.1.2 Panel / Wall Mounting

For panel or wall mounting, two wall suspension elements are included with the gateway. Each wall suspension element inserts into the base of the gateway's enclosure into two holes located on the upper-left and lower-right sides.

Figure 4: Wall Suspension Elements

Figure 5: Wall Suspension Elements Installed

6.2 Wiring Connections

Note that in order to power the unit, a power supply must also be installed. Refer to section 5.1 for more information.

1. Mount the unit via the desired method (refer to section 6.1).
2. Connect the various networks and I/O signals to their respective plugs/terminal blocks. Ensure that any wires are fully seated into their respective terminal blocks, and route the network cables such that they are located well away from any electrical noise sources, such as adjustable-speed drive input power or motor wiring. Also take care to route all cables away from any sharp edges or positions where they may be pinched.
3. Take a moment to verify that the gateway and all network cables have sufficient clearance from electrical noise sources such as drives, motors, or power-carrying electrical wiring.
4. Connect the power supply to the gateway's "Port A" terminal block on the terminals labeled V_{AC}/V_{DC+} and V_{AC}/V_{DC-}.

6.2.1 RS-485 Wiring

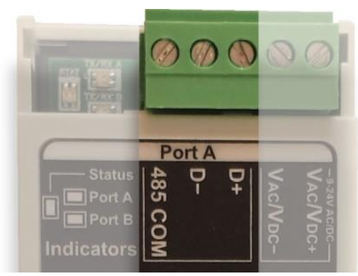


Figure 6: Port A RS-485 Terminal Block Connections

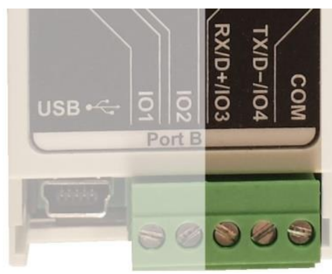


Figure 7: Port B RS-485 Terminal Block Connections

Table 1: RS-485 Terminal Definitions

Port A Terminal	Port B Terminal	Definition
D+	D+	RS-485 non-inverting data signal
D−	D−	RS-485 inverting data signal
485 COM	COM	RS-485 common-mode reference

6.2.2 RS-232 Wiring

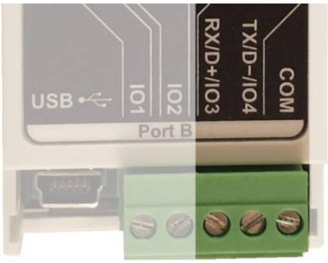


Figure 8: Port B RS-232 Terminal Block Connections

Table 2: RS-232 Terminal Definitions

Port B Terminal	Definition
RX	RS-232 receive data (to gateway from DTE)
TX	RS-232 transmit data (from gateway to DTE)
COM	RS-232 common ground

6.2.3 UART Wiring

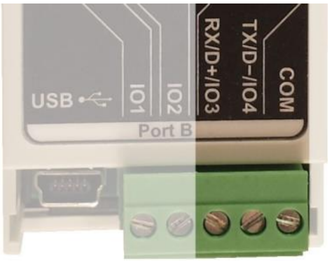


Figure 9: Port B UART Terminal Block Connections

Table 3: UART Terminal Definitions

Port B Terminal	Definition
RX	3.3V (5V Compatible) TTL receive data
TX	3.3V TTL transmit data
COM	Common ground

6.2.4 I/O Wiring

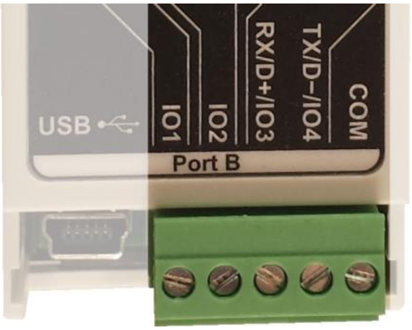


Figure 10: Port B I/O Terminal Block Connections

Table 4: I/O Terminal Definitions

I/O Terminal	Definition
IO1	Dedicated Universal I/O terminals
IO2	
IO3	Shared General-Purpose I/O terminals
IO4	
COM	I/O circuit common ground

For details on specific I/O applications, refer to section 10.

6.3 Isolation and Grounding

Grounding is of particular importance for reliable, stable operation. Communication system characteristics may vary from system to system, depending on the system environment and grounding method used.

The gateway features galvanic isolation which separates its ground references into three power domains. Both Port A and Port B are isolated from one another and from the power supply, as shown in Figure 11.

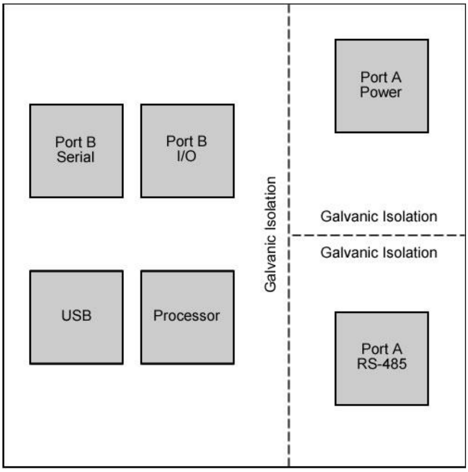


Figure 11: Isolation Diagram

Isolation provides separation of electrical signals and grounds. This eliminates common problems such as ground loops, induced ground noise, and high potential ground differences between devices that can cause communication errors and even damage to the devices. However, because the gateway's RS-485 ground references are isolated, it is required that a third signal be included in addition to the non-inverting and inverting signals for the common mode reference.

Please be sure to consider the following general points for making proper ground connections:

Grounding method checkpoints

1. Make all ground connections such that no ground current flows through the case or heatsink of a connected electrical device.
2. Do not make connections to unstable grounds (paint-coated screw heads, grounds that are subjected to inductive noise, etc.)
3. Ensure that all communication and I/O signals are referenced to the proper, corresponding ground connections on all devices.
4. Keep all logical ground connections for communications and I/O electrically separated from shield and protective ground connections.

7. LED Indicators

The gateway contains several different LED indicators, each of which conveys important information about the status of the unit and connected networks. These LEDs and their functions are summarized here.

7.1 Gateway Status

The gateway has one dichromatic LED to indicate the status of the device. This status LED can be in one of the following four states:

Startup: The LED flashes a green/red sequence on startup.

Configuration Mode: The LED flashes green rapidly while the device is in configuration mode. Refer to the Run Mode parameter in section for more information.

Normal Operation: The LED is software-controlled as defined by the Status LED Control setting for the device. Refer to section 8.3.2.2 for more information on the status LED settings.

Default
The LED is green and reflects one of the following two possible indications:

Solid green........................ the device is operating normally without USB connection

Blinking green .... the device is operating normally with USB connection

Port Activity
The LED flashes both green and red to indicate the communication activity of the selected port:

Green............................... the port has transmitted bytes since the last LED cycle

Red .... the port has received bytes since the last LED cycle

Database Value
The status LED is controlled by a value located in the device's internal database. Refer to Table 5 in section 8.3.2.2 for details.

Error Indication: If an error is detected, the status LED is red and blinks an error code. The number of sequential blinks (followed by 2 seconds of OFF time) indicates the nature of the error:

Error Code	Meaning
1...5	For internal use: contact ICC for assistance
6	USB to Serial Pass-Through mode
7	Invalid or corrupt configuration
8...10	For internal use: contact ICC for assistance

7.2 Port A Network Status

The gateway has one dichromatic LED for its Port A port to indicate the status of the RS-485 network.

Green (TX) .... Lights when the gateway is transmitting data on the RS-485 port.

Red (RX) .... Lights when the gateway is receiving data on the RS-485 port. Note that this does not indicate the validity of the data with respect to a particular protocol: only that data exists and is being detected.

7.3 Port B Status

The gateway has one dichromatic LED for its Port B port. The function of this indicator depends on the jumper configuration for the multi-interface port.

7.3.1 Port B Network Status

When the multi-interface port is configured for communications, the LED indicates the status of the RS-485, RS-232, or UART network.

Green (TX) .... Lights when the gateway is transmitting data on the port.

Red (RX) .... Lights when the gateway is receiving data on the port. Note that this does not indicate the validity of the data with respect to a particular protocol: only that data exists and is being detected.

7.3.2 Port B I/O Status

When the multi-interface port is configured for I/O, the LED indicates the current state of each I/O channel.

Green .... Lights when IO4 is logic low (0V). Off when IO4 is logic high (3.3V, 5V).

Red........................ Lights when IO3 is logic low (0V). Off when IO3 is logic high (3.3V, 5V).

8. Configuration Concepts

8.1 Overview

The gateway can be configured by a PC via a USB mini type-B cable. This connection provides power to the device, so there is no need for any external power supply while the gateway is attached to the PC.

Configuration of the gateway's Port B electrical characteristics for I/O and communications is performed via hardware jumpers. All other configuration is performed via USB with the Windows®-based *ICC Configuration Studio*. The *ICC Configuration Studio* allows both end users and OEM customers to define and map their communication data and physical I/O to the network(s) of their choice.

While the entire configuration of the gateway can be performed using only the *ICC Configuration Studio*, various run-time parameters also exist that allow the dynamic configuration of the Port A RS-485 network characteristics. An end user or integrator can configure these network parameters via USB using the *Network Parameter Utility* software. Alternatively, these parameters can be configured via communications or I/O signals through the gateway's Port B interface. This, for example, allows the gateway's RS-485 port settings to be configured entirely from an OEM's device via its own keypad, HMI, etc.

The *Network Parameter Utility* is also easily customizable for OEMs, allowing them to insert their own logos, product names, themes, etc. without any interaction from ICC. In this way, if the OEM wishes to allow their end-users to configure their device's network characteristics via USB, then the end-user perceives the software as being cohesive with the manufacturer's overall product.

8.2 Port B Jumper Configuration

The configuration jumpers for Port B allow the user to configure the proper electrical characteristics of the I/O and serial communication interfaces found on that port. The configuration jumpers are located on the gateway's control board and can be accessed by removing the top cover of the enclosure.

8.2.1 Removing the Cover

To access the Port B configuration jumpers, the top cover of the enclosure must be removed. To remove the cover, perform the following steps.

1. Remove the pluggable terminal blocks from both Port A and Port B.
2. The top piece of the enclosure is secured by two retention tabs on the left and right sides of the gateway. Use a flat-tip screwdriver to release the narrower retention tab on the left side of the gateway by inserting it into the release slot and gently prying the screwdriver up and out.
3. Continue lifting the left side of the top enclosure piece until it separates from the bottom.
4. The configuration jumpers are now accessible.

Figure 12: Removing the Cover

8.2.2 Jumper Settings

The jumpers are grouped into three functions. The dedicated universal I/O terminals, IO1 and IO2, each have three jumpers to configure the type of signal that will be connected to the terminal. The multi-interface terminals, RX/D+/IO3 and TX/D−/IO4, consist of two pairs of jumpers to select the communication interface or to configure the terminals for I/O.

8.2.2.1 I/O Jumpers (IO1 & IO2)

Figure 13: IO1 Jumpers

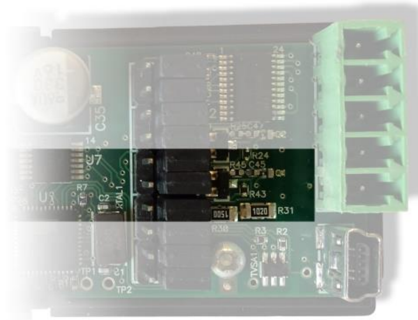


Figure 14: IO2 Jumpers

IN or OUT Jumper

This jumper selects whether the terminal is configured as an input (IN) or an output (OUT).

I or V Jumper

This jumper selects whether a current (I) or a voltage (V) signal will be applied to the terminal. If the terminal is configured as an output, this jumper must be set to voltage (V).

0-5V/I or 0-10V Jumper

This jumper selects the voltage level of the applied voltage signal. Select the 0-5V/I position for voltages between 0 and 5 volts or if current will be applied. Select the 0-10V position for voltages between 0 and 10 volts. If the terminal is configured as an output, this jumper position is irrelevant.

8.2.2.2 I/O Applications and Settings (IO1 & IO2)

Please refer to section 10.3 for information and jumper settings for specific I/O applications using the dedicated universal I/O terminals.

8.2.2.3 Multi-Interface Jumpers (RX/D+/IO3 & TX/D–/IO4)

Figure 15: Multi-Interface Jumpers

<u>UART/IO or 232/485 Jumpers</u>

These jumpers configure whether or not the multi-interface terminals are routed through the RS-232/485 transceiver. For RS-232 or RS-485 communications, set these jumpers to 232/485. For TTL UART communications or I/O functionality, set these jumpers to UART/IO to bypass the transceiver.

Note that these jumpers should always be moved in pairs. Undefined behavior may occur if both jumpers are not set to the same position.

<u>232 or 485 Jumpers</u>

These jumpers select whether RS-232 (232) or RS-485 (485) communications shall be used. When the multi-interface terminals are configured for UART/IO, it is recommended that these jumpers be removed entirely or each placed on a single post with the other end hung off, such that no electrical connection is made between any two posts.

Note that these jumpers should always be moved in pairs. Undefined behavior may occur if both jumpers are not set to the same position.

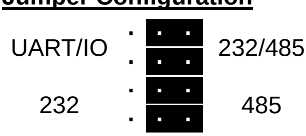
8.2.2.4 I/O Applications and Settings (IO3 & IO4)

Please refer to section 10.2 for information and jumper settings for the shared general-purpose I/O terminals.

8.2.2.5 Serial Communication Applications and Settings (RX/D+ & TX/D–)

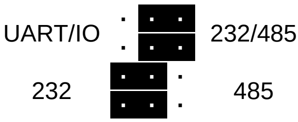
8.2.2.5.1 RS-485

<u>Jumper Configuration</u>



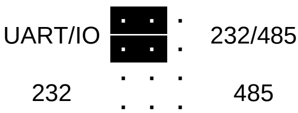
8.2.2.5.2 RS-232

Jumper Configuration



8.2.2.5.3 UART

Jumper Configuration



Note that each 232 or 485 selection jumper should be removed or placed on a single post such that no electrical connection is made between any two posts.

8.3 ICC Configuration Studio

This section will provide only a brief introduction to the configuration concepts of the *ICC Configuration Studio*. For more detailed information on how to install and use the Configuration Studio, refer to the separately-available training resources.

Creating a Device Configuration

A device can be added to the **Project** panel for configuration by first selecting the **Device Configurations** list heading and then:

- Double-clicking on it in the **Available Devices** panel.
- Right-clicking on it in the **Available Devices** panel and choosing **Add** from the context-sensitive menu.
- Hitting the <ENTER> key on the keyboard when the device is selected in the **Available Devices** panel.
- Dragging it from the **Available Devices** panel into the **Project** panel.
- Selecting it and selecting **Add Selected Device** from the **Edit** menu.
- Selecting it and clicking the **Add** button in the toolbar.

The device will then be added to the list of **Device Configurations**.

Going Online with a Device

All connected devices are automatically added to the **Discovered Devices** panel. This panel is shown by selecting the **Online Devices** list heading in the **Project** panel. To go online with a device:

- Double-click on it in the **Discovered Devices** panel.
- Right-click on it in the **Discovered Devices** panel and choose **Go Online** from the context-sensitive menu.

- Hit the <ENTER> key on the keyboard when the device is selected in the **Discovered Devices** panel.
- Drag it from the **Discovered Devices** panel into the **Project** panel.
- Select it and select **Go Online with Device** from the **Edit** menu.
- Select it and click the **Go Online** button in the toolbar.

When the studio goes online with a device, its configuration is automatically read. While the studio is online with a device, it will appear in green text in the **Discovered Devices** panel. The studio may be online with multiple devices simultaneously.

Uploading a Device's Configuration into a Project

The current configuration of an online device can be uploaded into the **Project** panel by selecting a device under the **Online Devices** list heading and then:

- Right-clicking on it and choosing **Upload Configuration** from the context-sensitive menu.
- Dragging it from the **Online Devices** heading into the **Device Configurations** heading.
- Selecting it and selecting **Upload Configuration to Project** from the **Device** menu.
- Selecting it and clicking the **Upload Configuration** button in the toolbar.

The device's configuration will then be added to the list of **Device Configurations**. Once the configuration is uploaded into the project, it may be modified.

Removing a Device Configuration from a Project

A configuration can be removed from a project by:

- Selecting the device in the **Project** panel and dragging it. A trash can icon will appear at the bottom of the **Project** panel, and dragging and dropping the device in the trash will remove it from the project.
- Hitting the <DELETE> key on the keyboard when the device is selected in the **Project** panel.
- Right-clicking on the device in the **Project** panel and choosing **Remove** from the context-sensitive menu.
- Selecting **Remove Selected Item** from the **Edit** menu when the device is selected.
- Clicking on the **Remove** button in the toolbar when the device is selected.

Going Offline with a Device

To go offline with a device:

- Select the device in the **Project** panel and drag it. A trash can icon will appear at the bottom of the **Project** panel, and dragging and dropping the device in the trash will go offline with it.

- Hit the <DELETE> key on the keyboard when the device is selected in the **Project** panel.
- Right-click on the device in the **Project** panel and choose **Go Offline** from the context-sensitive menu.
- Select **Go Offline with Device** from the **Edit** menu when the device is selected.
- Click on the **Go Offline** button in the toolbar when the device is selected.

Importing a Configuration from a Project File

An existing project file can be imported into the currently-active project. Click **File…Import Project,** and then select the desired *.icsproj file. The contents of the imported file will be merged with the active project.

Downloading a Configuration to a Device

To download a configuration to an online device, first select the device under the **Device Configurations** heading in the **Project** panel, and then navigate to **Device…Download Configuration to Device**. If the studio is currently online with only one compatible device, then the configuration will be downloaded to the online device. Otherwise, a device selection prompt is displayed to select which device to download the configuration to.

Updating Firmware

The studio automatically manages firmware updates when going online with a device and downloading a configuration to a device. Do not power off the device once the update is in progress as this may corrupt the firmware and/or the configuration.

Resetting an Online Device

To reset an online device, first select the device in the **Project** panel and then navigate to **Device…Reset Device**.

Interacting with the Database

To interact with a device’s database, select the device in the **Project** panel and then select the **Database** panel. If the **Database** panel is not visible, it can be enabled via **View…Database**. When an online device is selected, data values are updated from the device in real-time, and values can be edited by double-clicking the desired location in the database.

Diagnostics

To monitor the status of service objects, select the device in the **Project** panel and then select the **Diagnostics** panel. If the **Diagnostics** panel is not visible, it can be enabled via **View…Diagnostics**. When an online device is selected, diagnostics information is updated from the device in real-time. Individual diagnostics entries can be selected by clicking on them in the list, and multiple entries can be selected by either <CTRL>+clicking on them (to select them individually) or <SHIFT>+clicking on them (to select a range of entries). Counter values of all currently-selected diagnostics entries can be reset by clicking the **Reset Selected Counters** button.

General Configuration Process

To configure a device, add the desired protocols for the various ports, configure the communication settings (baud rate, parity, address, timeout, and scan rate/response delay etc.), and configure any objects associated with the respective protocols. Any changes will take effect once the configuration is downloaded to a device.

Note that numeric values can be entered not only in decimal but also in hexadecimal by including “0x” before the hexadecimal number.

8.3.1 General Object Editing Activities

The following editing activities apply for all types of configuration objects and project elements.

Adding an Object

To add an object, click on an item (protocol driver or Node, for example) in the **Project** panel. Any available objects for that item will be listed in the **Available Objects** panel (the panel title depends on the currently-selected item). An object can then be added to the item by:

- Double-clicking on it.
- Right-clicking on it and choosing **Add** from the context-sensitive menu.
- Hitting the <ENTER> key on the keyboard when the object is selected.
- Dragging it into the **Project** panel.
- Selecting it and selecting **Add Selected Device** from the **Edit** menu.
- Selecting it and clicking the **Add** button in the toolbar.

The object’s configurable fields can then be populated with valid values (where applicable).

Viewing an Object

In the **Project** panel, select a parent object to display a summary of all its child objects. For example, selecting a protocol driver will display the driver’s configuration in the **Summary** panel and list of current objects in the **Object List** panel.

Updating an Object

To update an object, select the object in the **Project** panel and make any required changes in the **Settings** panel.

Deleting an Object

An object can be deleted by performing one of the following actions:

- Selecting the object in the **Project** panel and dragging it. A trash can icon will appear at the bottom of the **Project** panel, and dragging the object to the trash will then delete it from the project.

- Hitting the <DELETE> key on the keyboard when the object is selected in the **Project** panel.
- Right-clicking on the object in the **Project** panel and choosing **Remove** from the context-sensitive menu.
- Selecting **Remove Selected Item** from the **Edit** menu when the object is selected.
- Clicking on the **Remove** button in the toolbar when the object is selected.

Note that this action cannot be undone. Deleting an object will also delete all of its child objects.

Copying and Pasting an Object

To copy an object, first click on an item in the **Project** panel. An object can then be copied by:

- Right-clicking on it and choosing **Copy** from the context-sensitive menu.
- Pressing the <CTRL+C> keys on the keyboard.
- Holding the <CTRL> key and dragging the item to the desired location in the **Project** panel.
- Dragging the item to a new location under a different parent object in the **Project** panel.
- Selecting **Copy Selected Item** from the **Edit** menu.
- Clicking on the **Copy** button in the toolbar.

To paste an object, first click on an item at the desired location in the **Project** panel. An object can then be pasted by:

- Right-clicking on it and choosing **Paste** from the context-sensitive menu.
- Pressing the <CTRL+V> keys on the keyboard.
- Dropping an item onto the desired location in the **Project** panel after holding the <CTRL> key and dragging the item.
- Dropping an item onto a new location under a different parent object in the **Project** panel after dragging the item.
- Selecting **Paste Item** from the **Edit** menu.
- Clicking on the **Paste** button in the toolbar.

After pasting an object, the object’s configurable fields can then be modified with valid values (where applicable).

Note that the studio allows you to copy and paste items between different locations, including different devices. This is useful for copying partial configurations from one device to another.

Reordering Objects

Objects can be reordered in the **Project** panel by dragging the item to the desired location. If the item is dragged outside of the items in the project tree, it will be moved to the end.

8.3.2 Device Settings

The following fields can be configured for a device. To view or edit device settings, click on the device in the **Project** panel. The settings are then available in the **Settings** panel.

Device Description

Each device added to a project can be individually tagged with a unique description string of up to 32 characters in length. This allows the devices within a project or an automation system to be clearly identifiable with their location or functional purpose.

Product ID

Defines a 16-bit, hexadecimal product ID for the device. This sets the value of the Product ID network configuration parameter and can be used to uniquely identify different OEM products or configurations.

Database Endianness Selection

Select the desired endianness for how data will be stored in the device's internal database for multi-byte data types. For more information on database endianness, refer to Appendix A: Database Endianness.

Write Triggering

Specifies how write requests for master/client protocols will be triggered when values are written to the device's database. Select **Change of Value** to trigger write requests only when the database value changes. Select **Database Write** to trigger write requests anytime the database location is written, regardless of value.

Default Network Protocol

Select the network protocol which will be activated by default after the configuration is downloaded to the device. If only one network protocol is configured, or the default protocol is irrelevant (due to dynamic configuration during startup), set this to **Automatic**.

Auto Run

Check this to allow the device to enter run mode automatically. If this is unchecked, the device will stay in configuration mode until the Run Mode configuration parameter is set to run. Note that the network driver is not started until the device enters run mode.

8.3.2.1 Configuration Locking Settings

The configuration locking settings allows a user to lock the device's configuration for reading and writing. When the device's configuration is locked, a user must enter the correct credentials in the ICC Configuration Studio to view or modify the configuration on the device.

Enable Lock

Check this to enable configuration locking on the device. After a configuration which enables this setting has been downloaded to the device, the device's configuration will be locked.

User Name

Enter the user name required to unlock the configuration on the device for reading and writing.

Password

Enter the password required to unlock the configuration on the device for reading and writing.

8.3.2.2 Status LED Settings

The device's status LED is software-controlled and can be configured to indicate a wide range of information to the user, providing useful insight into the operation of the device. These settings define the behavior of the device's status LED.

Status LED Control

Selects the method for controlling the device's status LED. Regardless of the option selected, upon startup, the status LED will flash the green, red, green, red startup sequence. Additionally, if an internal error occurs, the status LED will always flash red indicating the error code.

Default

This option is the default behavior of the status LED. The LED is solid green when power is applied to the device. The LED flashes green when a USB connection has been established between the device and a PC.

Port Activity

This option allows the selected port's TX and RX activity to be indicated by the status LED. For each LED cycle, if the port has transmitted any bytes since the last cycle, the status LED will light green for half of the cycle. If the port has received any bytes since the last cycle, the status LED will light red for half of the cycle.

Database Value

This option configures the status LED to be fully controlled by a value located in the device's internal database. This enables the status LED to be directly controlled via communications or by internal logic applied to data stored in the device's database. Table 5 lists the supported LED states.

Table 5: Status LED States

Value	LED State
0	Off
1	Green On
2	Red On
3	Green Flashing
4	Red Flashing
5…255	Off

Port

Selects the port for which the TX and RX activity will be indicated by the status LED.

Note that this option only applies when the Port Activity option is selected for the Status LED Control setting.

Database Address

Defines the address in the device's database used to control the status LED.

Note that this option only applies when the Database Value option is selected for the Status LED Control setting.

8.3.3 Port A RS-485 Port Settings

The RS-485 port supports various protocol drivers and allows multiple drivers to be configured on the port in a single configuration. However, only one protocol driver may be active at run time. The default protocol is designated via the Default Network Protocol device setting and, optionally, via the protocol network configuration parameter (refer to section for more details).

8.3.4 Port B Multi-Interface Port Settings

The multi-interface port supports multiple, jumper-selectable interfaces which include RS-485, RS-232, and UART communications. While the protocol drivers that can be configured on the port are the same as Port A, only one protocol driver can be selected for the multi-interface port.

8.3.5 USB Virtual COM Port Settings

The device can be configured to enumerate as a USB virtual COM port, providing direct serial communications between the device and a PC through the USB connection. The COM port can be used for various tasks, depending on the selected mode. This section details the different functions of the virtual COM port.

Mode

Select the desired mode for how the USB virtual COM port will be used. The available options are detailed below.

Serial Pass-Through

Select this option to cause the device to behave as a USB to serial converter. Any data sent to the USB virtual COM port will be sent on the physical serial port and any data received by the physical serial port will be received from the USB virtual COM port. Note that while the device is in this mode all other functionality of the device is disabled, regardless of other configuration settings.

Serial Redirect

Select this option to redirect communications from the selected serial port to the USB virtual COM port. By selecting this option, the device will communicate with the PC over the virtual COM port using the settings configured on the associated serial port. This allows the device to communicate with the PC using any of the supported serial port protocols. Note that the physical serial port is disabled when the device is configured in this mode.

Serial Sniffer

Select this option to sniff the received and transmitted packets on the selected serial port and output the data to the virtual COM port. When this mode is selected, the device will attempt to output every packet that the protocol driver configured on the serial port receives and transmits.

Because the sniffer operates independently from the physical serial port (so as not to impact communications), there may be times when the sniffer cannot output a received or transmitted packet due to the USB connection being unable to output characters faster than they are exchanged on the physical serial port. When this occurs, the sniffer will output the characters "ERR: Sniffer Packet Overflow" or "ERR: Sniffer Buffer Overflow". Additionally, the sniffer is able to detect receive errors on the serial port such as parity, overrun, and framing errors. If a receive error occurs on one or more characters of a packet, the sniffer will output the characters “ERR: Receive Error”.

Note that because the serial sniffer mode captures packets at the protocol driver level, a protocol must be configured on the selected serial port to output data to the USB virtual COM port. For convenience, there is a special “USB Serial Sniffer Settings” protocol selection to configure the serial port for sniffing only.

Serial Port

Select the desired serial port to target for use with the USB virtual COM port.

Sniffer Output Format

Select the desired output format of the serial sniffer data. The formatted data option outputs the captured data as ASCII text characters and includes annotations for whether the packet was received or transmitted, as well as a relative timestamp of when the packet was received or transmitted. The raw data option outputs the captured data as unmodified, binary characters.

8.3.6 USB Serial Capture Window

The USB Serial Capture Window allows you to connect to a device's USB Virtual COM port to view and save network packets captured by the device. The device's USB Virtual COM port must be configured for Serial Sniffer mode and the Sniffer Output Format must be set to Formatted Data.

When connected, the capture window will display the device's most recent received and transmitted packets. All packets captured during the duration of the session may be saved once the session has ended, even though they all may not be displayed in the window. The status bar at the bottom of the window tracks the duration of the connection as well as the total number of packets the device has received and transmitted.

To open the USB Serial Capture Window, select **USB Serial Capture Window…** from the **Tools** menu.

Capturing Packets

To begin capturing packets, the device must first be configured with the appropriate USB Virtual COM port settings as described above. Once configured, the device will appear in the **COM Port** selection box. Select the desired device from this drop down and connect to the device. To connect to the device, perform one of the following actions:

- Select **Connect** from the **Connection** menu.
- Click on the **Connect** button in the toolbar.

Note that connecting to a device will clear the capture log automatically.

Clearing the Capture Log

All captured data may be cleared at any time while connected to a device or after disconnecting from a device. This will also reset the connection time duration and all counters. To reset all captured data, perform one of the following actions:

- Select **Clear Log** from the **Edit** menu.
- Click on the **Clear Log** button in the toolbar.
- Hit the <DELETE> key on the keyboard.
- Right click on the capture output and select **Clear Log**.

Pausing the Display

While capturing, the output window will display only the most recent packets. Therefore, as new packets are captured and displayed in the window, old packets are removed from the display. At any time during capturing, the display updating may be paused so that no packets are added or removed. To pause the display, perform one of the following actions:

- Select **Pause Display** from the **Display** menu.
- Click on the **Pause Display** button in the toolbar.
- Right click on the capture output and select **Pause Display**.

Note that even though the display does not update when paused, packets are still being captured in the background.

Ending a Capture Session

The capture session is ended by disconnecting from the selected device. To disconnect from the device, perform one of the following actions:

- Select **Disconnect** from the **Connection** menu.
- Click on the **Disconnect** button in the toolbar.

Saving the Captured Data

Once a capture session has ended, the entire captured data may be saved. The data can be saved either as a Wireshark capture file or as a plain text document.

Wireshark Capture File

The captured data can be saved as a file which can be opened, decoded, and analyzed by Wireshark. Wireshark is a free network protocol analyzer and is available at http://www.wireshark.org/.

Any protocol capture may be viewed with Wireshark. However, Wireshark currently only supports decoding BACnet MS/TP packets and has limited support for Modbus RTU.

To save the captured data as a Wireshark capture file, perform one of the following actions:

- Select **Save As Wireshark Capture…** from the **File** menu.
- Click on the **Save As Wireshark Capture…** button in the toolbar.
- Hit the <CTRL+S> keys on the keyboard.

Text Document

The captured data can also be saved as a plain text document. To save the captured data as a text document, perform one of the following actions:

- Select **Save As Text…** from the **File** menu.
- Click on the **Save As Text…** button in the toolbar.
- Hit the <CTRL+SHIFT+S> keys on the keyboard.

8.3.7 Port Diagnostics

The Port Diagnostics window provides real-time monitoring of the number of bytes sent and received by the selected port. In addition, the Port Diagnostics also displays receive errors detected by the port.

8.3.8 Batch Update Mode

The ICC Configuration Studio supports a batch update mode for quickly updating firmware, and optionally, the configuration on all discovered devices without user interaction. While in batch update mode, the studio will automatically go online with a device, update the firmware, update the configuration if a matching configuration is found in the project, and then go offline with the device. It will do this for all discovered devices while in this mode. For each discovered device, the studio creates a log entry in the batch update log detailing the actions performed on the device.

Entering Batch Update Mode from within the Studio

To start batch update mode when the studio is open, select **Start Batch Update Mode** from the **Tools** menu. After the studio has entered batch update mode, pressing the ESC key will exit batch update mode. If any devices were discovered while in batch update mode, the studio will display a prompt to view the batch update log.

Launching the Studio in Batch Update Mode

The batch update mode can also be started when the studio is launched by using the "-b" or "-B" command line switch, and optionally, specifying a project file path to load. For example, the command line options "-b MyProject.icsproj" will load the project titled "MyProject" and start batch update mode. When batch update mode is entered using this method, the user cannot exit batch update mode using the ESC key.

Note that the command line options can also be used with a custom shortcut by appending them to the executable path in the **Target** field of the shortcut. This would allow a user to double click on the shortcut to launch the studio in batch update mode.

Viewing the Batch Update Log

After the studio has updated a device while in batch update mode, a log is available that can be accessed by selecting **Open Batch Update Log** from the **Help** menu. The log details the actions that the studio performed on discovered devices during the last batch update session.

At the end of the log, the studio records statistics for the batch update session. The statistics include the following information:

Devices Discovered

The total number of devices discovered while in batch update mode.

Successful

The total number of devices that were updated successfully.

Failed

The total number of devices that the studio failed to update.

Not Updated

The total number of devices that were not updated. This can occur if a device is already up to date, or if a device has limited network connectivity and cannot be updated.

Firmware Updated

The total number of firmware updates performed.

Configuration Updated

The total number of configuration updates performed.

Errors

The total number of devices that encountered an error while being updated. Note that this does not necessarily imply that the device failed to update.

8.3.9 I/O Settings

8.3.9.1 Overview

I/O terminals are available for applications in which the device is expected to interact with physical process signals. While a variety of different capabilities (analog input, digital output, pulse input, etc.) exist, note that not all capabilities are available on all I/O terminals. Refer to section 10 for a summary of the hardware capabilities of each IO terminal.

8.3.9.1.1 Analog Input

The analog input I/O type samples analog voltage levels between 0 and VCC on the input. A 10-bit ADC is used to encode the voltage levels to a numeric value between 0 and 1023. The raw ADC value may be scaled before being stored into the database using a "y = mx + b" -style function, where y is the resultant database value, x is the raw ADC value, m is the multiplier and b is the offset.

Database Address

Defines the database location where the (post-modified) sampled analog input value resides. The configuration studio will not allow entry of a database address that will cause the value to run past the end of the database. The highest valid database address, therefore, depends on the designated "Data Type".

Data Type

Specifies how the value will be stored in the database. This defines how many bytes will be allocated, whether the value should be treated as signed or unsigned, and whether the value should be interpreted as an integer or a floating point number. Select the desired data type from this dropdown menu.

Multiplier

The amount that the raw sampled analog input value is scaled by prior to being stored into the database (i.e. “m” in the “y = mx + b” equation).

Offset

The amount that is added to the scaled analog input value prior to being stored into the database (i.e. “b” in the “y = mx + b” equation).

8.3.9.1.2 Digital Input

The digital input I/O type samples a high or low voltage level on the input. A sampled high level produces a value of “1” for each bit selected in the bitmask when the polarity is set to “Normal”, or a value of “0” when the polarity is set to “Reverse”. A sampled low level produces a value of “0” for each bit selected in the bitmask when the polarity is set to “Normal” or a value of “1” when the polarity is set to “Reverse”.

Database Address

Defines the database address where the digital input’s value bit(s) reside.

Data Type

Fixed at “8-Bit Unsigned”

Bitmask

Specifies which bit(s) in the byte designated by the “Database Address” that the digital input maps to. It is possible to map a single digital input to multiple bits within the designated database location. All bits designated by the bitmask will correspond to the most-recently sampled value of the GPIO: they will either match, or be the inversion of the GPIO’s physical state, depending on the “Polarity” setting.

Polarity

Indicates the relationship between the physical state of the GPIO and the logical value(s) stored in the bits designated by the “Bitmask”. If the designated polarity setting is “Normal”, then the database bits will match the physical state of the GPIO (“1” when the input is sampled as “high”, and “0” when the input is sampled as “low”.) This relationship is reversed when the designated polarity setting is “Reverse”.

Contact Style

Select the type of signal that will be applied to input. Select dry if the signal is switched between open circuit and a connection to ground. Select wet if the signal always has voltage applied to it.

8.3.9.1.3 Digital Output

The digital output I/O type generates a high or low voltage level on the output. When all bits selected in the bitmask are “1”, a high level is produced when the polarity is set to “Normal”, and a low level is produced when the polarity is set to “Reverse”. Otherwise (when one or more bits are “0”), a low level is produced when the polarity is set to “Normal”, and a high level is produced when the polarity is set to “Reverse”.

Database Address

Defines the database address where the digital output’s value bit(s) reside.

Data Type

Fixed at “8-Bit Unsigned”

Bitmask

Specifies which bit(s) in the byte designated by the “Database Address” that map to the digital output. It is possible to map multiple bits within the designated database location to a single digital output. All bits designated by the bitmask will dictate the driven value of the GPIO: if all bits are “1” then the output value will be high, otherwise the output value will be low (assuming “Normal” polarity.)

Polarity

Indicates the relationship between the physical state of the GPIO and the logical value(s) stored in the bits designated by the “Bitmask”. If the designated polarity setting is “Normal”, then the GPIO will be high when all bits in the bitmask are “1”. This relationship is reversed when the designated polarity setting is “Reverse”.

8.3.9.1.4 Pulse Input

The pulse input I/O type is an interrupt-driven input which can be configured in combination with any other I/O type. The pulse input supports two modes, pulse counting and pulse frequency detection.

For pulse counting applications, an interrupt-driven pulse count is sampled at the same rate as other I/O inputs and accumulated in a database location. The count can be triggered to increment on rising edge transitions, falling edge transitions, or both depending on the selected mode. The pulse counter also features a debounce time to ignore pulses shorter than a specified duration.

For pulse frequency detection applications, an interrupt-driven pulse count is sampled at a periodic rate of approximately 2.5 seconds. The resulting pulse count and time duration are used to calculate a frequency. This frequency is periodically copied into a database location at the same rate as other I/O inputs. Frequencies from 1 Hz to 4 kHz can be detected. The frequency detection mode also features a low-pass frequency filter to ignore frequencies higher than a specified rate.

Database Address

Defines the database location where the accumulated pulse count or frequency value resides. The configuration studio will not allow entry of a database address that will cause the value to run past the end of the database. The highest valid database address, therefore, depends on the designated “Data Type”.

Data Type

Specifies how the data will be stored in the database. This defines how many bytes will be allocated for the value and whether the value should be interpreted as an integer or a floating point number. Select the desired data type from this dropdown menu.

Multiplier

The amount that the pulse count or frequency value is scaled by prior to being stored into the database (i.e. “m” in the “y = mx + b” equation).

Offset

The amount that is added to the scaled frequency value prior to being stored into the database (i.e. “b” in the “y = mx + b” equation).

Note that this setting is not available for pulse counting modes.

Mode

Selects the detection mode of the pulse input. Select between counting pulses for “Any Edge” (both rising & falling edges), “Rising Edge”, or “Falling Edge” and detecting pulse frequency.

Debounce Time

Specifies the amount of time (in milliseconds) to allow pulses to settle when transitioning from one state to another.

Filter Frequency

Specifies the cutoff frequency for a software controlled low-pass filter. Pulses with a frequency approaching this limit will be attenuated. Set this value to 0 to disable frequency filtering.

Contact Style

Select the type of signal that will be applied to input. Select dry if the signal is switched between open circuit and a connection to ground. Select wet if the signal always has voltage applied to it.

Note that this setting has no effect if the pulse input is used in conjunction with another I/O type. The other I/O type fully defines the function of the I/O signal.

8.3.9.1.5 Pulse Output

The pulse output I/O type can generate a 0.06 Hz - 6 MHz pulse waveform on the output with a duty cycle between 0.00% and 100.00%. Two modulation modes are available: pulse width modulation and frequency modulation.

In pulse width modulation mode, a duty cycle between 0.00% and 100.00% corresponds to a raw value between 0 and 10,000. The frequency is fixed in this mode to a user-definable value between 0.06 Hz and 6 MHz. However, note that frequencies above 600 KHz lose duty cycle precision as the frequency increases. For example, at 6 MHz, there are only 10 distinct duty cycle states.

In frequency modulation mode, a frequency between 0.06 Hz and 6 MHz corresponds to a raw value between 6 and 600,000,000. A value of 0 may be written to disable the output. In this mode, the duty cycle is fixed at 50%.

The database value may be scaled using an "x = (y - b) / m" inverse slope/intercept -style function to produce the raw value used for the duty cycle or frequency, where y is the database value, x is the raw value, m is the multiplier and b is the offset.

Database Address

Defines the database location where the (pre-modified) duty cycle or frequency value resides. The configuration studio will not allow entry of a database address that will cause the value to run past the end of the database. The highest valid database address, therefore, depends on the designated "Data Type".

Data Type

Specifies how the value will be stored in the database. This defines how many bytes will be allocated, whether the value should be treated as signed or unsigned, and whether the value should be interpreted as an integer or a floating point number. Select the desired data type from this dropdown menu.

Multiplier

The amount that the database value and offset difference is divided by to produce the raw duty cycle or frequency value (i.e. "m" in the "x = (y - b) / m" equation).

Offset

The amount that is subtracted from the database value prior to being scaled by the multiplier to produce the raw duty cycle or frequency value (i.e. "b" in the "x = (y - b) / m" equation).

Modulation Mode

Selects whether the pulse width or frequency of the pulse waveform will be modulated in relation to the database value. When frequency modulation is selected, the duty cycle is fixed at 50%.

PWM Frequency

Specifies the frequency of the PWM waveform (0.06 Hz - 6 MHz). This option is only available when pulse width modulation is selected as the "Modulation Mode".

8.3.10 Internal Logic Settings

8.3.10.1 Initial Persistent Values

8.3.10.1.1 Overview

The Mirius can be configured to write initial values to persistent user parameters using initializer objects. Persistent memory is initialized only once after a configuration has been downloaded to the device. This mechanism is useful for providing initial factory values for parameters mapped to the device's persistent memory. For more information on the Mirius's persistent user parameters, refer to Section 8.6.

8.3.10.1.2 Initializer Object Configuration

An initializer object is used to provide an initial value for parameters mapped to the persistent memory locations in the device's database. When persistent memory is initialized, the initializer objects are parsed and the designated 8-bit, 16-bit, or 32-bit value is written to the corresponding persistent database address(es). To add an initializer object to a device, select the device in the **Project** panel, then add **Internal Logic…Initial Persistent Values…Initializer Object**. The following paragraphs describe the configurable fields of an initializer object:

Database Address

Enter the starting database address in the persistent memory block where the first data element of this initializer object will begin. The maximum allowable database address depends on the designated Data Type.

Data Type

Specifies how the initializer value will be stored in the database. This defines how many bytes will be allocated, whether the value should be treated as signed or unsigned, and whether the value should be interpreted as an integer or a floating point number. Select the desired data type from this dropdown menu.

Value

Enter the value that each database address encompassed by this initializer object will be written to when the persistent memory is initialized.

Length

Enter the number of data elements for this initializer object. The total number of database bytes modified by an initializer object is determined by the Length multiplied by the number of bytes in the selected Data Type (1, 2 or 4 for 8-bit, 16-bit and 32-bit, respectively).

8.3.10.2 Fail-safe Values

8.3.10.2.1 Overview

The device can be configured to perform a specific set of actions when network communications are lost. This allows each address in the database to have its own unique "fail-safe" condition in the event of network interruption. Support for this feature varies depending on the protocol: refer to the specific protocol's driver manual for further information.

Note that this timeout feature is only used with slave/server protocols: this is not the same as the Timeout time used for service objects in master/client protocols.

There are two separate elements that comprise the timeout configuration:

- The timeout time
- Timeout Object configuration

8.3.10.2.2 Timeout Time

The timeout time is the maximum number of milliseconds for a break in network communications before a timeout will be triggered. This timeout setting is configured at the protocol level as part of a driver's configuration, and used by the protocol drivers themselves to determine abnormal loss-of-communications conditions. These conditions then trigger device-wide timeout processing events. If it is not desired to have a certain protocol trigger timeout processing events, then the protocol's timeout time may be set to 0 (the default value) to disable this feature.

For some protocols, the timeout time is set by the master device (PLC, scanner, etc.), and a timeout time setting is therefore not provided in the Configuration Studio's driver configuration. Additionally, not all protocols support timeout detection: refer to the specific protocol's driver manual for more information.

8.3.10.2.3 Timeout Object Configuration

A timeout object is used by the device as part of the timeout processing to set certain addresses of the database to "fail-safe" values. When a timeout event is triggered by a protocol, the timeout objects are parsed and the designated 8-bit, 16-bit, or 32-bit value is written to the corresponding database address(es). To add a timeout object to a device, select the device in the **Project** panel, then add **Internal Logic…Fail-safe Values…Timeout Object**. The following paragraphs describe the configurable fields of a timeout object:

Database Address

Enter the starting address in the database where the first data element of this timeout object will begin. The maximum allowable database address depends on the designated Data Type.

Data Type

Specifies how the timeout value will be stored in the database. This defines how many bytes will be allocated, whether the value should be treated as signed or unsigned, and whether the value should be interpreted as an integer or a floating point number. Select the desired data type from this dropdown menu.

Value

Enter the "fail-safe" timeout value that each database address encompassed by this timeout object will be automatically written with upon processing a timeout event triggered by a protocol.

Length

Enter the number of data elements for this timeout object. The total number of database bytes modified by a timeout object is determined by the Length multiplied by the number of bytes in the selected Data Type (1, 2 or 4 for 8-bit, 16-bit and 32-bit, respectively).

8.3.10.3 Database Logic

8.3.10.3.1 Overview

A variety of database logic operations are included which provide PLC-style manipulation of database values. Categories such as logical, arithmetic and filtering operations allow for autonomous control over value modification and data movement within the database. High-level signal conditioning is also realizable via the construction of compound formulas derived from the elemental building block operations provided. To add database logic operations to a device, select the device in the **Project** panel, then add **Internal Logic…Database Logic**.

Database logic operations are executed in sequential order, according to the ordinal position in which the operations are listed in the **Project** panel under the **Database Logic** heading.

Some notes of interest for the database logic operations are as follows:

All Database Logic Operations

- All inputs to an operation may either be a value located in the internal database or a constant value.
- A floating-point "Multiplier" field is available on each database-sourced input and on the output which allows the inputs to be scaled prior to operation execution, and the result to be scaled after operation execution. The input is multiplied by the input multiplier, and the result is divided by the output multiplier.
- All operations can be dynamically enabled/disabled using an optional "Enable Trigger" element (refer to section 8.3.10.3.3 for more information on Enable Trigger behavior.)
- The outputs of all operations must be stored in the internal database.
- The number of bytes taken from the database (for non-constant inputs) is determined by the corresponding "Data Type" selection, starting at the designated "Database Address".
- The number of bytes written to the database (for outputs) is determined by the corresponding "Data Type" selection, starting at the designated "Database Address".

Logical Operations

- *Not*, *And*, *Or*, and *Exclusive Or* operations can be performed on either a bitwise or logical basis, depending on the selection of the "Operation Type". When a logical operation type is chosen, non-zero input values are considered to be "true" and zero input values are considered to be "false". The output value of the logical operation will then be written to the database as "1" for true and "0" for false.
- The *Copy* operation outputs the input value.
- The *Bit Copy* operation outputs the value of a single bit from the input database location to a single bit in the output database location. No other bits in the output database location are modified by this operation.
- The *Indirect Copy* operation outputs the value at the database location specified by the input source to the database location specified by the output destination. This operation can be used to access different database locations dynamically. It could also be used to create reusable database logic subroutines by selecting a different input and output location for the subroutine during each execution cycle.
- The *Shift* operation outputs the input value bit-shifted by the shift amount.
- The *Compare* operation outputs a "1" if the comparison evaluates to true, otherwise it outputs a "0".
- The *Flag Test & Set* operation tests if the bit flags specified in the input mask are set in the input value and sets the bit flags specified in the output mask in the output value. This operation can test for ALL flags set/cleared or ANY flags set/cleared. If the flag test evaluates as true, all bit flags specified in the output mask in the output value are set, otherwise the flags are cleared. Only the bits specified in the output mask in the output value are modified by this operation.
- The *Value Change Detection* operation outputs a "1" if a change is detected in the input value between the last execution cycle and the current execution cycle, otherwise it outputs a "0".
- The *Mutiplexer* operation outputs one of its two inputs, depending on the selection. If *Selection* is zero, *Input 1* is output. If *Selection* is non-zero, *Input 2* is output.
- The *Byte Reverse* operation reverses the byte order of the input value and outputs the result.

Arithmetic Operations

- The *Add* operation calculates the expression *[Input 1] + [Input 2]*.
- The *Subtract* operation calculates the expression *[Input 1] – [Input 2]*.
- The *Multiply* operation calculates the expression *[Input 1] × [Input 2]*.
- The *Divide* operation calculates the expression *[Input 1] / [Input 2]*.
- The *Modulo* operation calculates the expression *[Input 1] mod [Input 2]*.
- The *Exponential* operation calculates the expression $[Input\ 1]^{Exponent}$. "Input 1" can be a database value, a constant value, or e (exponential function).
- The *Nth Root* operation calculates the expression $\sqrt[Degree]{Input\ 1}$.

- The *Logarithm* operation calculates the expression log_{Base}*(Input 1)*. "Base" can be a database value, a constant value or e (natural logarithm).
- The *Random* operation outputs a random number between *Input 1* and *Input 2*. Note that the operation is limited to producing only 32,768 unique values.
- The *Divide*, *Exponential*, *Nth Root* and *Logarithm* operations output an integer-rounded value when an integer data type is used.

<u>Trigonometric Operations</u>

- The *Sine* operation calculates the expression *sin(Input 1)*, where *Input1* is in radians.
- The *Cosine* operation calculates the expression *cos(Input 1)*, where *Input1* is in radians.
- The *Tangent* operation calculates the expression *tan(Input 1)*, where *Input1* is in radians.
- The *Arc Sine* operation calculates the expression sin^{-1}*(Input 1)*, where the output is in radians.
- The *Arc Cosine* operation calculates the expression cos^{-1}*(Input 1)*, where the output is in radians.
- The *Arc Tangent* operation calculates the expression tan^{-1}*(Input 1)*, where the output is in radians.

<u>Filtering Operations</u>

- The *Debounce Filter* and *Hysteresis Filter* operations are functionally identical with the single exception that the *Debounce Filter* does not use a "Value Tolerance" (it is fixed at 0).
- In order for the output of the *Debounce Filter* or *Hysteresis Filter* to change (i.e. reflect the input value), "Input 1" must first change to a value outside of the "Value Tolerance" range and then remain within the "Value Tolerance" range of the new value for the entire "Stable Time".

8.3.10.3.2 Database Logic Settings

<u>Scan Rate</u>

Defines the scan cycle time in milliseconds (50ms minimum) of the database logic processing task. All operations are evaluated for execution in sequential order at this frequency. Note that this does <u>not</u> necessarily mean that each operation is guaranteed to execute every scan cycle: only that it will be evaluated as to whether or not it should execute. Namely, if an "Enable Trigger" element is added to an operation, then the trigger must evaluate to "true" for the operation to execute during that scan cycle. Refer to section 8.3.10.3.3 for more information on Enable Trigger behavior.

8.3.10.3.3 Enable Trigger

Each database logic operation can optionally include an "Enable Trigger" element, which provides dynamic conditional execution capabilities. By default (i.e. if an enable trigger element is not added to the operation), each operation is automatically triggered to execute every scan cycle. If it is desired for an operation to execute conditionally, however, then an enable trigger element can be added to it. The enable trigger element defines an “Enable Value”, which specifies a byte-size trigger value that can reside at any location in the internal database. When implemented, the enable value is evaluated every scan cycle: if this value is non-zero (or zero when the “Inverted” Trigger Option is used), the operation will execute.

The enable value itself can be modified by any communication driver currently running on the device, which enables networked devices to dynamically control the execution of database logic operations. The enable value can also be the output result of other database logic operations. While the output of any database operation can be used for this purpose, such a scenario may most typically use the output of a “compare” operation in order to control whether or not other operations should execute (e.g. execute a certain operation only when some process variable is greater than a certain value, etc.) Allowing the conditional execution of database logic operations to be based on data values obtained via communications or as a result of other database logic operations enables the construction of flexible, hierarchical and dynamic data evaluation and manipulation engines.

Enable Value Database Address

Enter the database address which specifies the byte-size trigger value.

8.3.10.3.3.1 Trigger Options

The enable trigger can perform basic logic on the enable value to determine if an operation should execute using a variety of trigger options. These setting determine what logic should be applied to the enable value when evaluating whether or not the operation should execute.

Inverted

Specifies whether the enable logic should be inverted. This applies to both the evaluation of whether or not the operation should execute as well as resetting the enable value when the auto reset option is used.

Auto Reset

Allows the enable value to be automatically reset upon completion of the operation. The actual value written to the enable value depends on the other trigger options selected. If no options are selected, a value of 0 is written to the enable value. If the inverted option is used, a value of 1 is written to the enable value. If the bitmask option is used, each bit selected in the bitmask is written to a 0 (or a 1 if the inverted option is used) in the enable value.

Bitmask

If this option is used, it selects which bits in the enable value to evaluate. Every selected bit in the enable value must be 1 (or 0 when the inverted option is used) for the operation to execute.

8.3.11 Service Objects and Diagnostics Objects

A service object is used by the device to make requests on a network when a master/client protocol is enabled. Each service object defines the services (read and/or write) that should be performed on a range of network objects of a common type. The data from read requests is mirrored in the database starting at a user-defined address (if a read function is enabled). When a value within that address range in the database changes, a write request is generated on the network (if a write function is enabled). Specific service object configuration depends on the protocol selected: refer to the specific protocol's driver manual for further details.

Master/client drivers commonly also provide the ability to debug configured service objects while the driver is running by way of optional diagnostics objects. Where supported, diagnostics objects can be added to each service object, and a database address can be designated at which to store the status information. The diagnostics object is a 16-byte structure containing elements such as a transmission counter, receive counter, receive error counter, current status, and the last error of the defined service object. This information is detailed in Appendix B: Diagnostics Objects. Because the diagnostics object resides in the database alongside the service object's process data, it can also be accessed over any supported network by mapping appropriate network elements to the corresponding database addresses.

Alternatively, the diagnostics objects can be viewed within the Configuration Studio by selecting a device in the **Project** panel and then clicking on the **Diagnostics** tab. Diagnostics objects are automatically added to the **Diagnostics** panel, and are disseminated and displayed in plain text for easy interpretation. For online devices, diagnostics objects are updated in real-time and all counters can be reset by selecting one or more entries in the list and clicking the **Reset Selected Counters** button.

8.4 Network Parameter Utility

The Network Parameter Utility (NPU) software is an end-user application intended to quickly and easily configure the device's network settings in the field via USB. The communication settings for the Port A RS-485 port on the device are dependent upon the system in which the device is being installed. The NPU is designed to configure only these network communication settings. Refer to the separate *NPU OEM Manual* for more details and information.

Features

- Light-weight, portable application - no installation necessary, simply unzip and run the exe.
- Includes all necessary USB drivers and firmware files.
- Automatically discovers and connects to the device via USB.
- Allows configuration of standard, field-configurable network settings such as protocol, baud rate, parity, address, and other protocol-specific settings.

- Detects outdated firmware on the connected device and allows the user to update the firmware on the device to the latest version.
- Only shows those protocols and options actually configured on the connected device.
- Self-documented - each setting provides a useful tooltip to the user that describes the setting.
- Supports custom OEM product IDs, allowing the same device to be used in many different applications and with various OEM products while being presented to the user as a unique product from the OEM.
- Includes the ability for OEM's to create Device Update Files (DUF) to provide to the end user to easily update devices in the field.
- Fully customizable, including OEM branding, custom theme and styling, and disabling or hiding settings on an application-wide or protocol-specific basis.

8.5 Network Configuration Parameters

The Mirius has a bank of internal parameters which allow dynamic configuration of its Port A RS-485 port properties. These configuration parameters are mapped into the device's internal database and can be modified via USB and also via communication or I/O signals through the Port B multi-interface port. This flexibility allows the end user to perform tasks such as changing the protocol, address, baud rate, and parity of the RS-485 port while isolating them from the object mappings and other settings already configured on the device. The configuration parameters are internally mapped to the device's database and are summarized in Table 6.

Note

- All configuration parameters are non-volatile. These parameters retain their values across power cycles and reboots.
- All configuration parameters are 16-bit unsigned values (consuming 2 bytes in the internal database) unless otherwise noted.
- Configuration parameter range checking is not performed when the parameter is modified: range checking is performed by each specific driver during initialization, and invalid settings will result in the device transitioning to the error state with an indication of "invalid configuration parameters".
- Upon startup, a write request will be issued to all Service Objects that have their Write Function enabled and are mapped to the configuration parameters' database locations, thereby synchronizing the host's parameters with the PicoPort's configuration parameters.
- The Status Code parameter will have a value of 65535 (Network Communication Error) when a network timeout has occurred. The Status Code will revert to a value of 0 (Normal) when network communications is restored. This feature is only available in conjunction with the active slave/server protocol configured on the device's network port. Refer to the specific protocol's driver manual for further information on fail-safe timeout support.
- When the Run Mode parameter has a value of 1 (Configuration Mode), the device's status LED will flash green rapidly, indicating parameters are being synchronized with the host.

Table 6: Mirius Configuration Parameters

Database Address	Parameter	Notes
8192	Product ID	OEM-configurable Product ID (*read-only*) Default = 0x2101
8194	Firmware Version	Value = Firmware version * 1000 (*read-only*) (for example, 2300 = V2.300)
8196	Status Code	*Read-only* 0 = Normal 6 = USB to Serial Pass-Through Mode 7 = Invalid configuration parameters 65535 (0xFFFF) = Network Communication Error All other values = For internal use only (contact ICC)
8198	Run Mode	0 = Startup (*read-only value*) 1 = Configuration Mode (*read-only value*) 2 = Running (*read/write value*) 3 = Error (*read-only value*) 65534 (0xFFFE) = Factory Reset[1] (*write-only value*) 65535 (0xFFFF) = Reset (*write-only value*)
8200	Protocol	0 = Disabled 1 = Modbus RTU Master 2 = Modbus RTU Slave 3 = BACnet MS/TP Server 4 = BACnet MS/TP Client 5…11 = Reserved 12 = Metasys N2 Slave 13 = Toshiba ASD Master 14 = Sullair Master 15 = Modbus RTU Sniffer 16 = MSA Chillgard Monitor 17 = Metasys N2 Master 18 = Siemens FLN Slave 19 = TCS Basys Master 20 = DMX-512 Master 21 = DMX-512 Slave 22 = M-Bus Master 23 = AO Smith AIN Slave 24 = AO Smith PDNP Master 25…26 = Reserved 27 = Siemens FLN Master 28 = Toshiba Computer Link Master 29 = Generic Serial Master 30 = Generic Serial Slave 31 = Macurco Modbus Monitor 32 = Modbus RTU Firewall Router 33 = BACnet MS/TP Client/Server 253 = Host - Network Pass-Through
8202	Address	Protocol-specific

Database Address	Parameter	Notes	
8204	Baud Rate	Value = Baud rate / 100 (for example, 96 = 9600 baud)	
8206	Parity	0 = No parity, 1 stop bit 1 = Odd parity, 1 stop bit	2 = Even parity, 1 stop bit 3 = No parity, 2 stop bits
8208	Timeout	Value in ms	
8210	Scan Rate / Response Delay	Value in ms	
8212	Number of Retries	Protocol-specific	
8214…8223	Reserved	Reserved for future use	
8224…8255	Protocol-Specific Parameters	Defined by the currently-selected protocol	

1. The Factory Reset command resets the device and restores the network configuration parameters to the default values defined in the device's configuration file.

As shown in Table 6, shared network configuration parameters numbered 8202 and higher may have different valid ranges (or may be ignored altogether) depending on the selected protocol. There is also a range of parameters whose use is protocol-specific: their meaning and adjustment ranges are unique only to the currently-selected driver. Refer to the following tables for the settings available for each driver.

Table 7: Modbus RTU Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	0…3
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 8: Modbus RTU Slave Parameters

Database Address	Parameter	Notes
8202	Address	1…247
8204	Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	0…3
8208	Timeout	0…65535 (0…65.535s)
8210	Response Delay	0…65535 (0…65.535s)
8212	Number of Retries	Ignored
8224	Word Order Override Enable	Enables the word order override for all register mapping objects 0 = Disabled 1 = Enabled
8226	Word Order	0 = Little endian word order 1 = Big endian word order

Table 9: BACnet MS/TP Server Parameters

Database Address	Parameter	Notes
8202	Address	0…127
8204	Baud Rate	96…1152 (9600…115200 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	APDU Timeout	0…65535 (0…65.535s)
8210	Scan Rate / Response Delay	Ignored
8212	Number of APDU Retries	0…10
8224…8239	Device Name	16-character device name as per the BACnet specification
8240	Device Instance	0…4194302 (32-bit unsigned value)
8244	Max Master	Address…127
8246	UTC Offset	-840…720 (in minutes) Ignored if real-time clock functionality is disabled
8248	Daylight Saving	0 = Off 1 = On Ignored if real-time clock functionality is disabled
8250	Max Info Frames	1…100
8252	Fail-safe Timeout	0…65535 (0…65.535s)

Table 10: BACnet MS/TP Client Parameters

Database Address	Parameter	Notes
8202	Address	0…127
8204	Baud Rate	96…1152 (9600…115200 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	APDU Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of APDU Retries	0…10
8224…8239	Device Name	16-character device name as per the BACnet specification
8240	Device Instance	0…4194302 (32-bit unsigned value)
8244	Max Master	Address…127
8246	UTC Offset	-840…720 (in minutes) Ignored if real-time clock functionality is disabled
8248	Daylight Saving	0 = Off 1 = On Ignored if real-time clock functionality is disabled
8250	Max Info Frames	1…100

Table 11: Metasys N2 Slave Parameters

Database Address	Parameter	Notes
8202	Address	1…255
8204	Baud Rate	Ignored (fixed at 9600 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	Timeout	0…65535 (0…65.535s)
8210	Response Delay	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 12: Toshiba ASD Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	0…3
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 13: Sullair Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored (fixed at 0)
8204	Baud Rate	Ignored (fixed at 9600 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	Timeout	Ignored (fixed at 500ms)
8210	Scan Rate	Ignored (fixed at 0)
8212	Number of Retries	Ignored

Table 14: Modbus RTU Sniffer Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	0…3
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate / Response Delay	Ignored
8212	Number of Retries	Ignored

Table 15: MSA Chillgard Monitor Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	Ignored (fixed at 19200 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	Timeout	Ignored (fixed at 4.5s)
8210	Scan Rate / Response Delay	Ignored
8212	Number of Retries	Ignored

Table 16: Metasys N2 Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	Ignored (fixed at 9600 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 17: Siemens FLN Slave Parameters

Database Address	Parameter	Notes
8202	Address	1…98
8204	Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	Timeout	0…65535 (0…65.535s)
8210	Response Delay	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 18: TCS Basys Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 19: DMX-512 Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	Ignored (fixed at 250kbaud)
8206	Parity	Ignored (fixed at no parity / 2 stop bits)
8208	Timeout	Ignored
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 20: DMX-512 Slave Parameters

Database Address	Parameter	Notes
8202	Address	1…512
8204	Baud Rate	Ignored (fixed at 250kbaud)
8206	Parity	Ignored (fixed at no parity / 2 stop bits)
8208	Timeout	0…65535 (0…65.535s)
8210	Response Delay	Ignored
8212	Number of Retries	Ignored

Table 21: M-Bus Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	3…384 (300…38400 baud)
8206	Parity	Ignored (fixed at even parity / 1 stop bit)
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 22: AO Smith AIN Slave Parameters

Database Address	Parameter	Notes
8202	Address	Ignored (fixed at 16)
8204	Baud Rate	192…384 (19200…38400 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	Timeout	Ignored
8210	Response Delay	Ignored
8212	Number of Retries	Ignored

Table 23: AO Smith PDNP Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	Ignored (fixed at 19200 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 24 : Siemens FLN Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 25: Toshiba Computer Link Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	3…1152 (300…115200 baud)
8206	Parity	0…3
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 26: Generic Serial Master Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	3…1152 (300…115200 baud)
8206	Parity	0…3
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 27: Generic Serial Slave Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	3…1152 (300…115200 baud)
8206	Parity	0…3
8208	Timeout	0…65535 (0…65.535s)
8210	Response Delay	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 28 : Macurco Modbus Monitor Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	0…3
8208	Timeout	0…65535 (0…65.535s)
8210	Scan Rate / Response Delay	Ignored
8212	Number of Retries	Ignored

Table 29: Host - Network Pass-Through Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Minimum Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	Ignored
8208	Timeout	Ignored
8210	Response Delay	Ignored
8212	Number of Retries	Ignored

Table 30: Modbus RTU Firewall Router Parameters

Database Address	Parameter	Notes
8202	Address	Ignored
8204	Baud Rate	24…1152 (2400…115200 baud)
8206	Parity	0…3
8208	Timeout	Ignored
8210	Response Delay	0…65535 (0…65.535s)
8212	Number of Retries	Ignored

Table 31: BACnet MS/TP Client/Server Parameters

Database Address	Parameter	Notes
8202	Address	0…127
8204	Baud Rate	96…1152 (9600…115200 baud)
8206	Parity	Ignored (fixed at no parity / 1 stop bit)
8208	APDU Timeout	0…65535 (0…65.535s)
8210	Scan Rate	0…65535 (0…65.535s)
8210	Scan Rate / Response Delay	Ignored
8212	Number of APDU Retries	0…10
8224…8239	Device Name	16-character device name as per the BACnet specification
8240	Device Instance	0…4194302 (32-bit unsigned value)
8244	Max Master	Address…127
8246	UTC Offset	-840…720 (in minutes) Ignored if real-time clock functionality is disabled
8248	Daylight Saving	0 = Off 1 = On Ignored if real-time clock functionality is disabled
8250	Max Info Frames	1…100
8252	Fail-safe Timeout	0…65535 (0…65.535s)

8.6 Persistent User Parameters

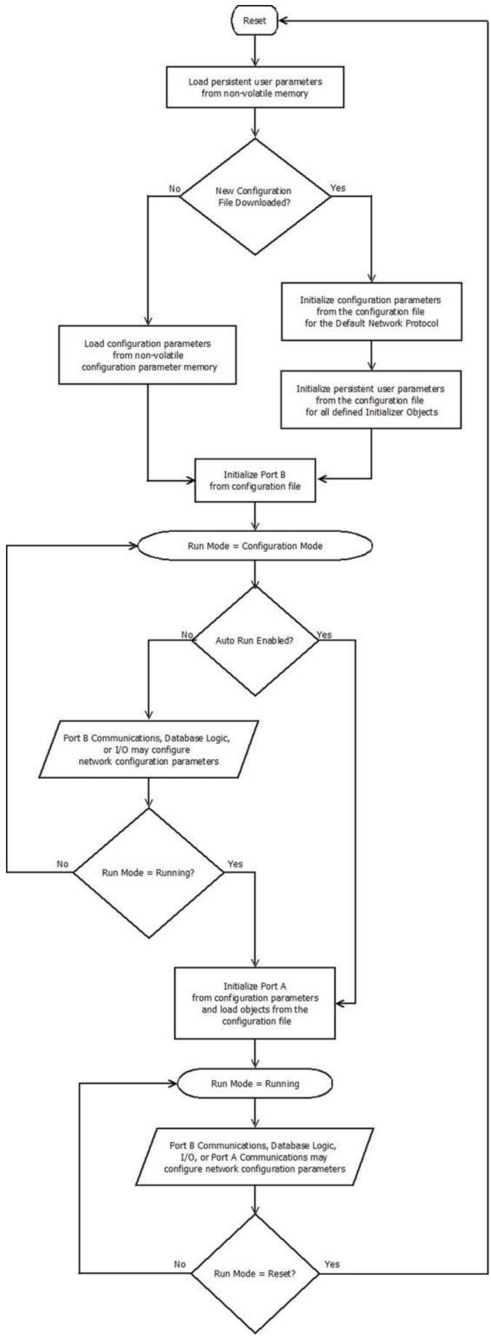
The Mirius has a block of general-purpose, persistent memory mapped into its internal database. This is useful for data which must persist across power cycles and reboots, such as calibration data or custom serial numbers. The persistent memory block consists of 64 bytes and begins at database address 8256. Aside from their persistence, these database locations behave like any other database location. Any object may be mapped to the persistent memory database locations using any data type.

Care must be taken, however, when selecting data to be mapped into the persistent memory block. Because this memory is backed by internal flash storage, it is susceptible to write cycle limitations of flash technology. Therefore, it is important to only map data which changes infrequently to these locations.

To ease OEM programming, the Mirius supports the definition of initial values for the persistent memory locations as part of the configuration of the device. This allows the Mirius to be fully programmed by an OEM by simply downloading a single configuration file. For more information on initializing persistent values, refer to Section 8.3.10.1.

8.7 Initialization Overview

Because the Mirius is intended for a wide array of applications, a variety of configuration options are provided. Configuration Studio settings such as Auto Run and the Default Network Protocol are used in conjunction with the configuration parameters to define the behavior of the device during initialization. This flowchart details the initialization steps that the Mirius performs during startup.



8.8 I/O and Database Logic Scan Rate

The Configuration Studio provides a configurable “scan rate” parameter for all GPIO and database logic. Refer to sections 8.3.9 and 8.3.10.3. These settings are found in the **I/O Settings** panel when “I/O” is selected in the **Project** panel, and in the **Database Logic Settings** panel when “Database Logic” is selected in the **Project** panel. While the general behavior of these scan rate settings are similar and therefore will be jointly discussed here, note that the I/O processing and database logic processing are performed in separate threads in the Mirius’s realtime operating system (RTOS), and therefore are unrelated to, and independent of, each other in practice.

The GPIO scan rate applies regardless of whether a GPIO pin is configured as a digital input, digital output, analog input or pulse output. In both the I/O and database logic cases, all processing is performed “in bulk”, after which a delay of “scan rate” number of ms is inserted before a subsequent processing activity is performed. This cycle (process / wait / process, etc.) is then performed forever (refer to Figure 16).

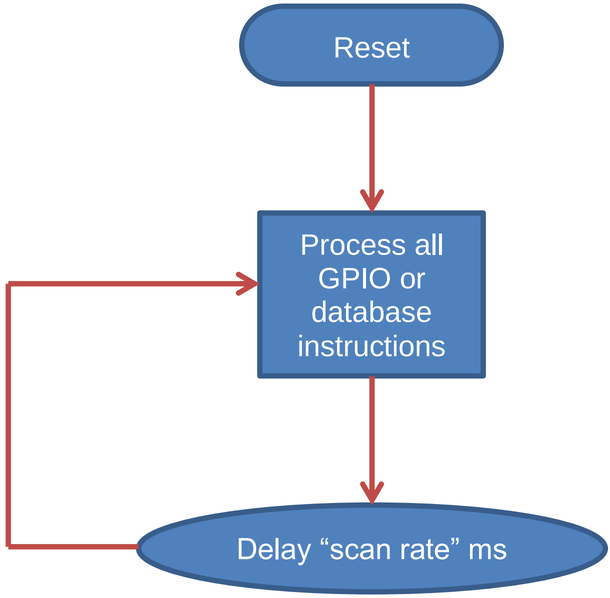


Figure 16: GPIO / Database Logic Scan Cycle

9. Serial Drivers

The gateway supports a variety of serial drivers on its RS-485 and multi-interface ports. For a list of supported protocols, refer to the *Mirius Supported Drivers and I/O List*. For detailed information on each protocol, refer to the specific protocol's driver manual.

10. I/O Functions and Specifications

The gateway has two dedicated universal I/O terminals, IO1 and IO2, and two shared general-purpose I/O terminals, IO3 and IO4. The shared I/O terminals are only available for use when Port B serial communications is disabled.

10.1 IO1 & IO2: Dedicated Universal I/O

The dedicated universal I/O terminals support analog input, digital input, digital output, pulse input, and pulse output functions.

When configured as an output, the dedicated universal I/O terminals provide an open-drain style output which performs low-side/sink switching.

Each channel has three configuration jumpers for selection of signal direction, type, and range. The settings for each of these jumpers are shown for each specific I/O application in section 10.3 below.

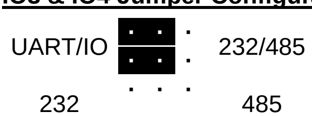
10.2 IO3 & IO4: Shared General-Purpose I/O

The general purpose I/O terminals, IO3 and IO4, are shared with Port B serial communications and may only be used when communications is disabled. These terminals support digital input, digital output, pulse input, and pulse output functions.

When used as an output, the general purpose I/O terminals provide a 3.3V TTL push/pull (CMOS) style output. These outputs are suitable for voltage signaling applications only and should not be used directly to drive loads of any kind.

The configuration jumpers for the shared IO3 and IO4 terminals select the signaling type that will be present on the terminals. There is only a single jumper configuration that applies to both IO3 and IO4 terminals, regardless of the configured I/O function. This jumper configuration is shown below.

<u>IO3 & IO4 Jumper Configuration</u>



Note that each 232 or 485 selection jumper should be removed or placed on a single post such that no electrical connection is made between any two posts.

10.3 Supported I/O Functions

The I/O function assigned to each terminal is software configurable using the *ICC Configuration Studio*, see section 8.3.9 for details.

10.3.1 Analog Input (IO1 and IO2 only)

When configured as an analog input, the dedicated universal I/O terminals (IO1 and IO2) can be used to measure either voltage (0 - 5V, 0 - 10V) or current (0 - 20mA). The analog inputs have 10-bit resolution and support user-configurable multiplier and offset scaling.

This I/O function is not supported on the general purpose I/O terminals (IO3 and IO4).

10.3.1.1 Voltage Measurement (0 - 5V)

Typical Applications

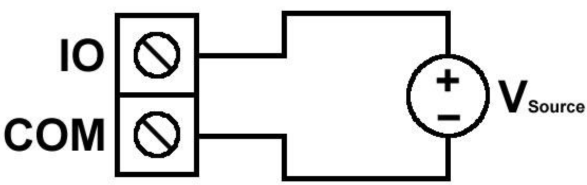
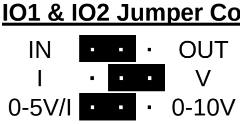


Figure 17: Voltage Measurement Circuit

Specifications

Technical Specifications for IO1 and IO2	
Input Range	0 - 5 V
Input Impedance	34 kΩ
Resolution	10 bits
ADC Type	Successive Approximation Register (SAR)
Quantization Error	± 5 mV max
Accuracy	± 0.75%
Sampling Rate	20 Hz max (User Configurable)
Scaling	Multiplier, Offset (User Configurable)
Raw Value Range	0 - 1023

IO1 & IO2 Jumper Configuration



10.3.1.2 Voltage Measurement (0 - 10V)

Typical Applications

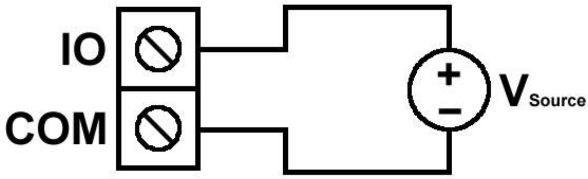


Figure 18: Voltage Measurement Circuit

Specifications

Technical Specifications for IO1 and IO2	
Input Range	0 - 10 V
Input Impedance	68 kΩ
Resolution	10 bits
ADC Type	Successive Approximation Register (SAR)
Quantization Error	± 10 mV max
Accuracy	± 1%
Sampling Rate	20 Hz max (User Configurable)
Scaling	Multiplier, Offset (User Configurable)
Raw Value Range	0 - 1023

IO1 & IO2 Jumper Configuration

IN · · · OUT
I · · · V
0-5V/I · · · 0-10V

10.3.1.3 Current Measurement (0 - 20mA)

Typical Applications

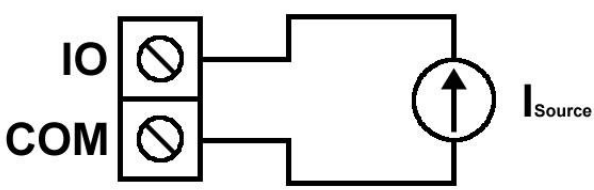
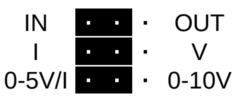


Figure 19: Current Measurement Circuit

Specifications

Technical Specifications for IO1 and IO2	
Input Range	0 - 20 mA
Input Impedance	250 Ω
Resolution	10 bits
ADC Type	Successive Approximation Register (SAR)
Quantization Error	± 0.02 mA max
Accuracy	± 1%
Sampling Rate	20 Hz max (User Configurable)
Scaling	Multiplier, Offset (User Configurable)
Raw Value Range	0 - 1023

IO1 & IO2 Jumper Configuration



10.3.2 Digital Input

As a digital input, the I/O terminals support both dry-contact style and wet-contact style applications. The contact style is software-configurable and dictates whether or not internal biasing will be applied to the I/O terminal.

For dry-contact applications, the device detects when a connection is made between its I/O terminal and COM terminal. These applications do not require external voltage to be applied.

The dedicated universal I/O terminals (IO1 and IO2) are compatible with several wet-contact style applications, supporting applied voltages of 3.3V, 5V, and 10V. A single jumper setting selects between 3.3V/5V levels and 10V levels.

The general purpose I/O terminals (IO3 and IO4) support 3.3V and 5V wet-contact style applications only.

10.3.2.1 Dry-Contact (Open / GND)

Typical Applications

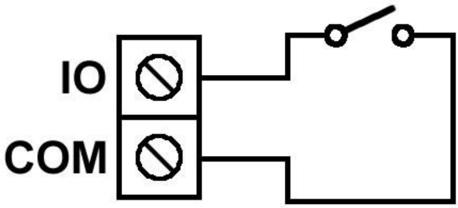


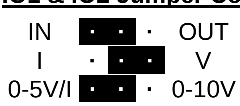
Figure 20: Dry Contact Circuit

Specifications

Technical Specifications for IO1 and IO2	
Logic Low Input	GND (COM)
Logic High Input	Open Circuit
Internal Bias Voltage	3.3 V
Bias Resistance	17 kΩ typical
Contact Current	0.3mA max
Sampling Rate	20 Hz max (User Configurable)
Polarity	Normal, Reversed (User Configurable)
Logic Low Raw Value	0
Logic High Raw Value	1

Technical Specifications for IO3 and IO4	
Logic Low Input	GND (COM)
Logic High Input	Open Circuit
Internal Bias Voltage	3.3 V
Bias Resistance	24 kΩ
Contact Current	0.15mA
Sampling Rate	20 Hz max (User Configurable)
Polarity	Normal, Reversed (User Configurable)
Logic Low Raw Value	0
Logic High Raw Value	1

IO1 & IO2 Jumper Configuration



IO3 & IO4 Jumper Configuration

Refer to section 10.2 for IO3 and IO4 jumper settings.

10.3.2.2 Wet-Contact (0V / 3.3V, 0V / 5V)

Typical Applications

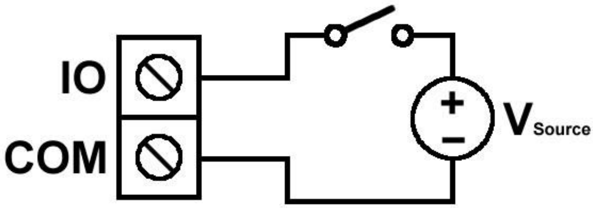


Figure 21: Wet Contact Switch Circuit (IO1 & IO2 Only)

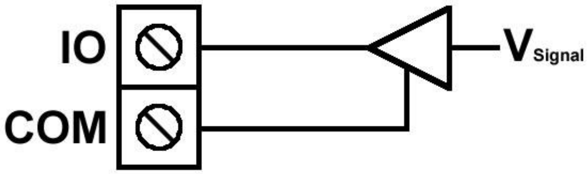


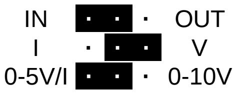
Figure 22: Wet Contact Signal Circuit

Specifications

Technical Specifications for IO1 and IO2	
Logic Low Input	0.4 V max
Logic High Input	1.8 V min
Sampling Rate	20 Hz max (User Configurable)
Polarity	Normal, Reversed (User Configurable)
Logic Low Raw Value	0
Logic High Raw Value	1

Technical Specifications for IO3 and IO4	
Logic Low Input	0.5 V max
Logic High Input	1.9 V min
Internal Bias Voltage	3.3 V
Bias Resistance	24 kΩ
Sampling Rate	20 Hz max (User Configurable)
Polarity	Normal, Reversed (User Configurable)
Logic Low Raw Value	0
Logic High Raw Value	1

IO1 & IO2 Jumper Configuration



IO3 & IO4 Jumper Configuration

Refer to section 10.2 for IO3 and IO4 jumper settings.

10.3.2.3 Wet-Contact (0V / 10V) (IO1 and IO2 only)

Typical Applications

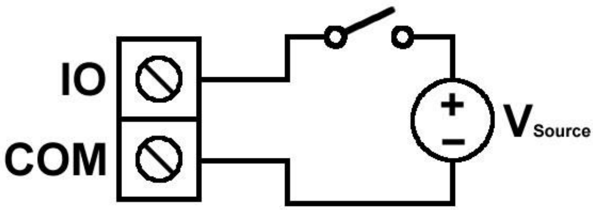


Figure 23: Wet Contact Switch Circuit

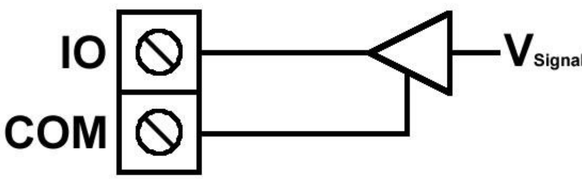
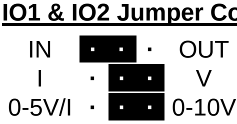


Figure 24: Wet Contact Signal Circuit

Specifications

Technical Specifications for IO1 and IO2	
Logic Low Input	0.8 V max
Logic High Input	3.6 V min
Sampling Rate	20 Hz max (User Configurable)
Polarity	Normal, Reversed (User Configurable)
Logic Low Raw Value	0
Logic High Raw Value	1

IO1 & IO2 Jumper Configuration



10.3.3 Digital Output

The dedicated universal I/O terminals (IO1 and IO2) expose an open-drain style output to the I/O terminal. This solid-state switch performs low-side switching, also known as sink switching, connecting the I/O terminal to the COM terminal. This output type can be used to switch loads of up to 250 mA on and off and supports external biasing of up to 24 VDC.

The general purpose I/O terminals (IO3 and IO4) expose a push/pull style output to the I/O terminal. This CMOS-driven output switches between applying 3.3V and 0V to the output terminal, with respect to the COM terminal.

Typical Applications

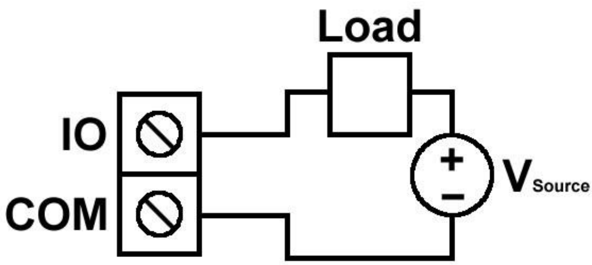


Figure 25: Open Drain Load Circuit (IO1 & IO2 Only)

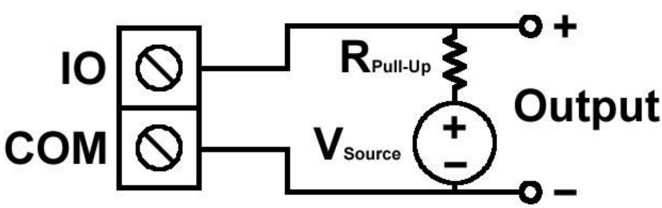


Figure 26: Open Drain Output Circuit (IO1 & IO2 Only)

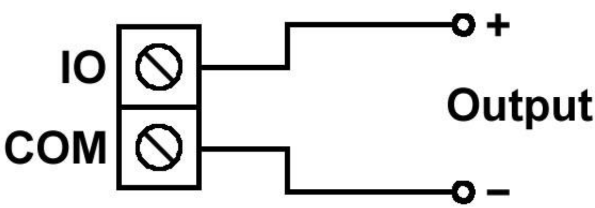


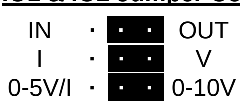
Figure 27: Push/Pull Output Circuit (IO3 & IO4 Only)

Specifications

Technical Specifications for IO1 and IO2	
Output Style	Open Drain
Switching	Low-Side/Sink
Output Impedance	0.4 mΩ max
External Bias Voltage	24 VDC max
Max Load	250 mA
Power Dissipation	500 mW max
Logic Low Output	Open Circuit
Logic High Output	Sink to GND (COM)
Polarity	Normal, Reversed (User Configurable)
Logic Low Raw Value	0
Logic High Raw Value	1

Technical Specifications for IO3 and IO4	
Output Style	Push/Pull (3.3V CMOS)
Max Load	4 mA
Logic Low Output	0 V
Logic High Output	3.3 V
Polarity	Normal, Reversed (User Configurable)
Logic Low Raw Value	0
Logic High Raw Value	1

IO1 & IO2 Jumper Configuration



IO3 & IO4 Jumper Configuration

Refer to section 10.2 for IO3 and IO4 jumper settings.

10.3.4 Pulse Input

The pulse input is an interrupt-driven I/O type that can be used in conjunction with any other I/O type on any I/O terminal. The device supports both pulse counting and frequency detection applications.

For pulse counting applications, the device can detect rising edges, falling edges, or both. A configurable debounce time is also available to provide filtering and allow the signal to settle for a predetermined amount of time when changing state.

When configured for frequency detection applications, the device samples the signal for approximately 2.5 seconds, beginning on a rising edge and ending on a subsequent rising edge. The number pulses detected during the sample period is then divided by the sample time to produce a measured frequency. The device also supports a software-controlled low-pass filter, allowing unwanted high frequency signals to be ignored. It is recommended that the filter frequency be set to at least twice the maximum frequency to be measured to eliminate the possibility of aliasing.

All I/O terminals are compatible with dry-contact applications. The device detects a pulse when its I/O terminal is connected to its COM terminal. These applications do not require external voltage to be applied.

The dedicated universal I/O terminals (IO1 and IO2) are compatible with several wet-contact style applications, supporting applied voltages of 3.3V, 5V, and 10V. A single jumper setting selects between 3.3V/5V levels and 10V levels.

The general purpose I/O terminals (IO3 and IO4) support 3.3V and 5V wet-contact style applications only.

10.3.4.1 Pulse Counter

10.3.4.1.1 Dry-Contact (Open / GND)

Typical Applications

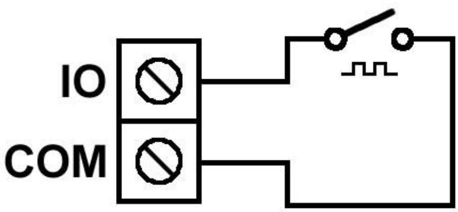


Figure 28: Dry Contact Pulse Circuit

Specifications

Technical Specifications for IO1 and IO2	
Logic Low Input	GND (COM)
Logic High Input	Open Circuit
Internal Bias Voltage	3.3 V
Bias Resistance	17 kΩ typical
Contact Current	0.3mA max
Minimum Pulse Width	100 µs
Detection Modes	Any Edge, Rising Edge, Falling Edge
Filtering	Debounce Time (User Configurable)
Scaling	Multiplier (User Configurable)

Technical Specifications for IO3 and IO4	
Logic Low Input	GND (COM)
Logic High Input	Open Circuit
Internal Bias Voltage	3.3 V
Bias Resistance	24 kΩ
Contact Current	0.15mA max
Minimum Pulse Width	100 μs
Detection Modes	Any Edge, Rising Edge, Falling Edge
Filtering	Debounce Time (User Configurable)
Scaling	Multiplier (User Configurable)

IO1 & IO2 Jumper Configuration

IN · · · OUT
I · · · V
0-5V/I · · · 0-10V

IO3 & IO4 Jumper Configuration

Refer to section 10.2 for IO3 and IO4 jumper settings.

10.3.4.1.2 Wet-Contact (0V / 3.3V, 0V / 5V)

Typical Applications

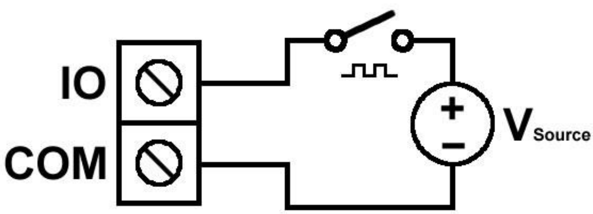


Figure 29: Wet Contact Pulse Switch Circuit (IO1 & IO2 Only)

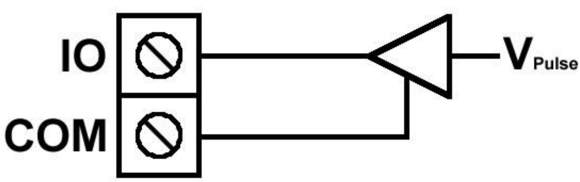


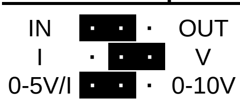
Figure 30: Wet Contact Pulse Signal Circuit

Specifications

Technical Specifications for IO1 and IO2	
Logic Low Input	0.4 V max
Logic High Input	1.8 V min
Minimum Pulse Width	100 μs
Detection Modes	Any Edge, Rising Edge, Falling Edge
Filtering	Debounce Time (User Configurable)
Scaling	Multiplier (User Configurable)

Technical Specifications for IO3 and IO4	
Logic Low Input	0.5 V max
Logic High Input	1.9 V min
Internal Bias Voltage	3.3 V
Bias Resistance	24 kΩ
Minimum Pulse Width	100 μs
Detection Modes	Any Edge, Rising Edge, Falling Edge
Filtering	Debounce Time (User Configurable)
Scaling	Multiplier (User Configurable)

IO1 & IO2 Jumper Configuration



IO3 & IO4 Jumper Configuration

Refer to section 10.2 for IO3 and IO4 jumper settings.

10.3.4.1.3 Wet-Contact (0V / 10V) (IO1 and IO2 only)

Typical Applications

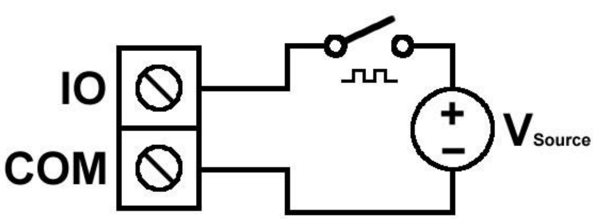


Figure 31: Wet Contact Pulse Switch Circuit

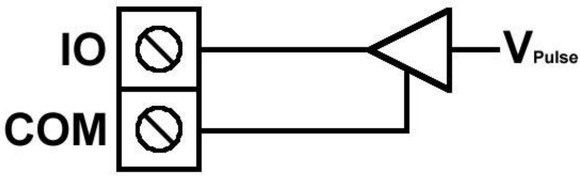


Figure 32: Wet Contact Pulse Signal Circuit

<u>Specifications</u>

Technical Specifications for IO1 and IO2	
Logic Low Input	0.8 V max
Logic High Input	3.6 V min
Minimum Pulse Width	100 µs
Detection Modes	Any Edge, Rising Edge, Falling Edge
Filtering	Debounce Time (User Configurable)
Scaling	Multiplier (User Configurable)

<u>IO1 & IO2 Jumper Configuration</u>

IN ▪ ▪ ▪ OUT
I ▪ ▪ ▪ V
0-5V/I ▪ ▪ ▪ 0-10V

10.3.4.2 Frequency Detection

10.3.4.2.1 Dry-Contact (Open / GND)

<u>Typical Applications</u>

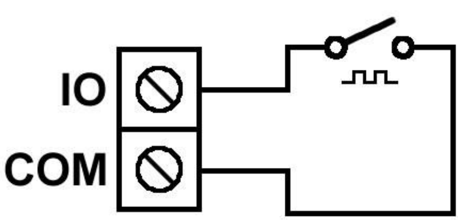


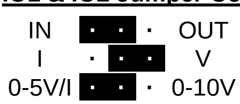
Figure 33: Dry Contact Pulse Circuit

Specifications

Technical Specifications for IO1 and IO2	
Logic Low Input	GND (COM)
Logic High Input	Open Circuit
Internal Bias Voltage	3.3 V
Bias Resistance	17 kΩ typical
Contact Current	0.3mA max
Frequency Range	1 Hz - 4 kHz
Sample Period	2.5 s
Filtering	Low-Pass Frequency Filter (User Configurable)
Scaling	Multiplier, Offset (User Configurable)

Technical Specifications for IO3 and IO4	
Logic Low Input	GND (COM)
Logic High Input	Open Circuit
Internal Bias Voltage	3.3 V
Bias Resistance	24 kΩ
Contact Current	0.15mA max
Frequency Range	1 Hz - 4 kHz
Sample Period	2.5 s
Filtering	Low-Pass Frequency Filter (User Configurable)
Scaling	Multiplier, Offset (User Configurable)

IO1 & IO2 Jumper Configuration



IO3 & IO4 Jumper Configuration

Refer to section 10.2 for IO3 and IO4 jumper settings.

10.3.4.2.2 Wet-Contact (0V / 3.3V, 0V / 5V)

Typical Applications

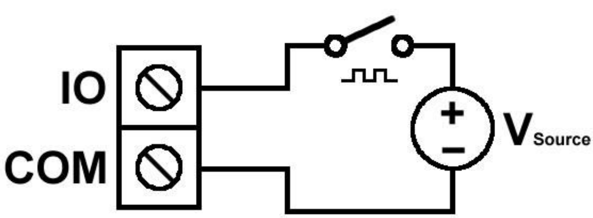


Figure 34: Wet Contact Pulse Switch Circuit (IO1 & IO2 Only)

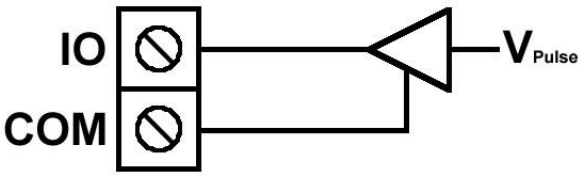


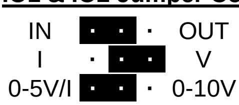
Figure 35: Wet Contact Pulse Signal Circuit

Specifications

Technical Specifications for IO1 and IO2	
Logic Low Input	0.4 V max
Logic High Input	1.8 V min
Frequency Range	1 Hz - 4 kHz
Sample Period	2.5 s
Filtering	Low-Pass Frequency Filter (User Configurable)
Scaling	Multiplier, Offset (User Configurable)

Technical Specifications for IO3 and IO4	
Logic Low Input	0.5 V max
Logic High Input	1.9 V min
Internal Bias Voltage	3.3 V
Bias Resistance	24 kΩ
Frequency Range	1 Hz - 4 kHz
Sample Period	2.5 s
Filtering	Low-Pass Frequency Filter (User Configurable)
Scaling	Multiplier, Offset (User Configurable)

IO1 & IO2 Jumper Configuration



IO3 & IO4 Jumper Configuration

Refer to section 10.2 for IO3 and IO4 jumper settings.

10.3.4.2.3 Wet-Contact (0V / 10V) (IO1 and IO2 only)

Typical Applications

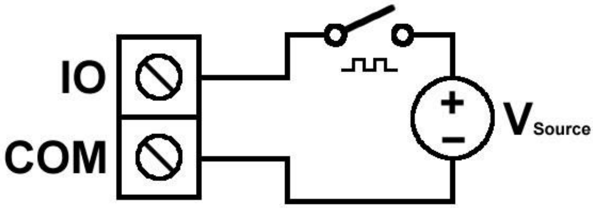


Figure 36: Wet Contact Pulse Switch Circuit

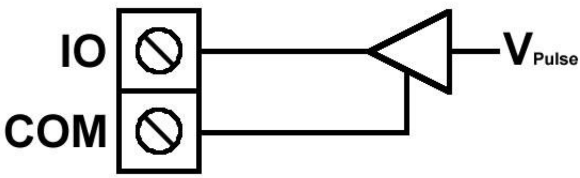


Figure 37: Wet Contact Pulse Signal Circuit

Specifications

Technical Specifications for IO1 and IO2	
Logic Low Input	0.8 V max
Logic High Input	3.6 V min
Frequency Range	1 Hz - 4 kHz
Sample Period	2.5 s
Filtering	Low-Pass Frequency Filter (User Configurable)
Scaling	Multiplier, Offset (User Configurable)

IO1 & IO2 Jumper Configuration

IN ▪ ▪ · OUT
I · ▪ ▪ V
0-5V/I · ▪ ▪ 0-10V

10.3.5 Pulse Output

The pulse output functionality allows the device to output pulse waveforms on the I/O terminals. Two types of modulation modes are supported.

Pulse width modulation outputs a constant frequency signal with a varying duty cycle. The duty cycle determines the width of each pulse. For analog output applications, this PWM signal may be filtered using a simple resistor and capacitor network to produce a varying DC voltage signal.

Frequency modulation outputs a fixed, 50 percent duty cycle signal with a varying frequency. A wide range of frequencies are possible, although higher frequencies may be limited by capacitive effects of the external circuitry connected to the output terminal.

The dedicated universal I/O terminals (IO1 and IO2) expose an open-drain style output to the I/O terminal. Each pulse performs low-side switching, also known as sink switching, connecting the I/O terminal to the COM terminal.

The general purpose I/O terminals (IO3 and IO4) expose a push/pull style output to the I/O terminal. For each pulse, 3.3V is applied on the I/O terminal, with respect to the COM terminal.

10.3.5.1 Pulse Width Modulation

Typical Applications

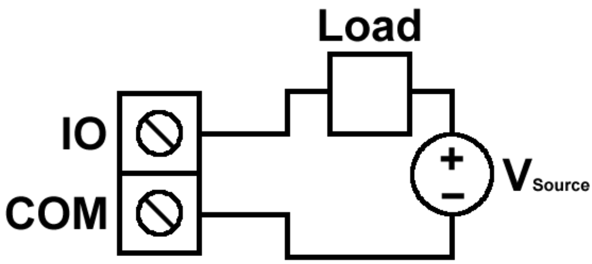


Figure 38: Open Drain Load Circuit (IO1 & IO2 Only)

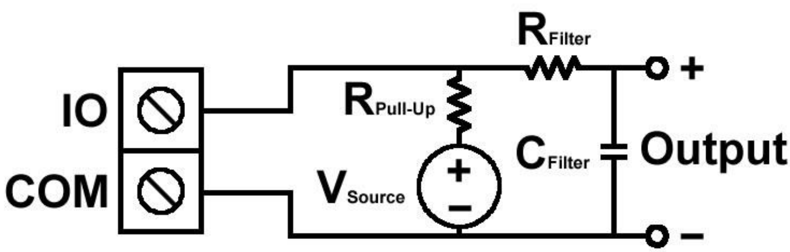


Figure 39: Open Drain Analog Output Circuit (IO1 & IO2 Only)

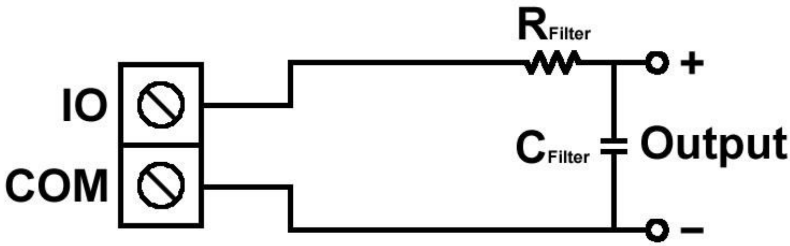


Figure 40: Push/Pull Analog Output Circuit (IO3 & IO4 Only)

Specifications

Technical Specifications for IO1 and IO2	
Output Style	Open Drain
Switching	Low-Side/Sink
Output Impedance	0.4 mΩ max
External Bias Voltage	24 VDC max
Max Load	250 mA
Power Dissipation	500 mW max
Duty Cycle	0 - 100 %
Frequency	0.06 Hz - 6 MHz[1] (User Configurable)
Scaling	Multiplier, Offset (User Configurable)
0% Output	Open Circuit
100% Output	Sink to GND (COM)
Raw Value Range	0 - 10,000 (0.00% - 100.00%) 6 – 600,000,000 (0.06Hz – 6MHz)

Technical Specifications for IO3 and IO4	
Output Style	Push/Pull (3.3V CMOS)
Max Load	4 mA
Duty Cycle	0 - 100 %
Frequency	0.06 Hz - 6 MHz[1] (User Configurable)
Scaling	Multiplier, Offset (User Configurable)
0% Output	0 V
100% Output	3.3 V
Raw Value Range	0 - 10,000 (0.00% - 100.00%) 6 – 600,000,000 (0.06Hz – 6MHz)

[1] Maximum achievable frequency may be limited by external components due to capacitive charging/discharging effects.

IO1 & IO2 Jumper Configuration

IN	OUT
I	V
0-5V/I	0-10V

IO3 & IO4 Jumper Configuration

Refer to section 10.2 for IO3 and IO4 jumper settings.

10.3.5.2 Frequency Modulation

Typical Applications

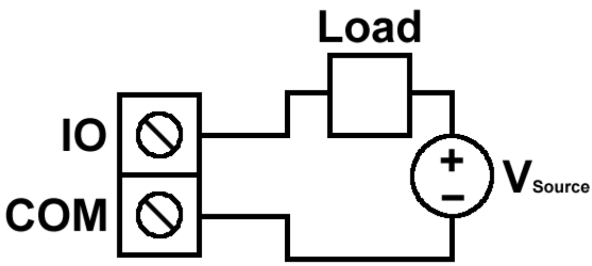


Figure 41: Open Drain Load Circuit (IO1 & IO2 Only)

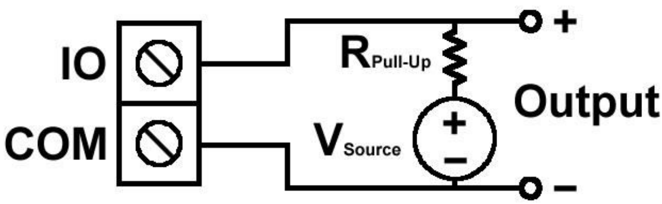


Figure 42: Open Drain Output Circuit (IO1 & IO2 Only)

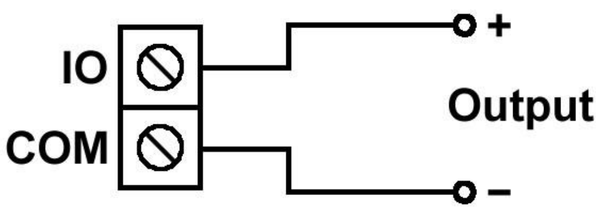


Figure 43: Push/Pull Output Circuit (IO3 & IO4 Only)

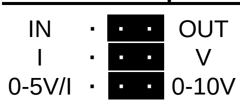
Specifications

Technical Specifications for IO1 and IO2	
Output Style	Open Drain
Switching	Low-Side/Sink
Output Impedance	0.4 mΩ max
External Bias Voltage	24 VDC max
Max Load	250 mA
Power Dissipation	500 mW max
Duty Cycle	50%
Frequency	0.06 Hz - 6 MHz[1]
Scaling	Multiplier, Offset (User Configurable)
0 Hz Output	Open Circuit
Raw Value Range	6 - 600,000,000 (0.06 Hz - 6 MHz)

Technical Specifications for IO3 and IO4	
Output Style	Push/Pull (3.3V CMOS)
Max Load	4 mA
Duty Cycle	50%
Frequency	0.06 Hz - 6 MHz[1]
Scaling	Multiplier, Offset (User Configurable)
0 Hz Output	0 V
Raw Value Range	6 - 600,000,000 (0.06 Hz - 6 MHz)

[1] Maximum achievable frequency may be limited by external components due to capacitive charging/discharging effects.

IO1 & IO2 Jumper Configuration



IO3 & IO4 Jumper Configuration

Refer to section 10.2 for IO3 and IO4 jumper settings.

11. Troubleshooting

Although by no means exhaustive, the following table provides possible causes behind some of the most common errors experienced when using the gateway.

Problem	Symptom	Solution
The gateway will not turn on	All LEDs are off and the gateway shows no activity	• Confirm that power is connected to the correct inputs on the Port A terminal block. • If powering over USB, ensure the USB cable from the kit is used, and that the computer is able to supply sufficient power as a USB host device. • If firmware was being updated, it may have been corrupted. Unplug and reconnect the USB cable and run the Configuration Studio. Follow the Configuration Studio instructions to restore the firmware.
No communication between the RS-485 network and the gateway	The gateway's RS-485 TX and RX LEDs are blinking slowly, sporadically, or not at all	• Check connections and orientation of wiring between the network and the gateway. • Confirm that the protocol, baud rate, parity, and address settings on the RS-485 port match your network configuration.
No communication between the RS-485 network and the gateway	The gateway's RS-485 RX LED is solid ON	• The RS-485 signal wires are reversed. • Ensure that a network reference/ground wire is in place.
Firmware-generated error	The module status LED is flashing red: the number of times the LED flashes indicates an error code	• 6 flashes indicate the gateway is in USB to Serial Pass-Through mode. All other functionality of the device is disabled. • 7 flashes indicate one or more Network Configuration Parameters are invalid. • Any other number of flashes indicates an internal device error. Record the blink sequence and contact ICC for further assistance.

Problem	Symptom	Solution
The device will not communicate to the Configuration Studio via USB	The USB cable is plugged into both the PC and the device, but the module status LED is not flashing green: the Configuration Studio may indicate a communication error	• Unplug and reconnect the USB cable. • Try a different USB cable. • Try a different USB port on the computer. • Reinstall the Configuration Studio. • Reinstall the USB device drivers (contact ICC for assistance).

12. Appendix A: Database Endianness

A key feature of the Millennium Series gateways is the ability to change the byte order storage scheme for data in the database between big endian and little endian. The database endianness is the convention used to store multi-byte data to or retrieve multi-byte data from the database. The selected endianness affects the end-to-end consistency of multi-byte data between the two networks on the gateway.

To better understand how this byte-ordering scheme works, the following explains how the gateway stores and retrieves multi-byte data to and from the database. Data is stored into the database starting at the low address and filled to higher addresses. The endianness determines whether the most-significant or least-significant bytes are stored first.

Let's look at some examples that demonstrate this. Figure 44 shows how the hex value 0x12345678 is stored into the database using a big endian byte order. Since the hex value 12 is the most significant byte, it is stored at address "a", the lowest address.

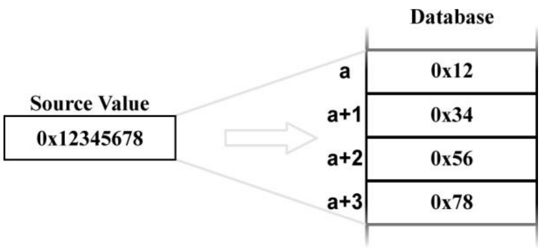


Figure 44: Big Endian Storage

Figure 45 demonstrates how the hex value 0x12345678 is stored into the database using a little endian byte order. Since the hex value 78 is the least significant byte, it is stored at the lowest address.

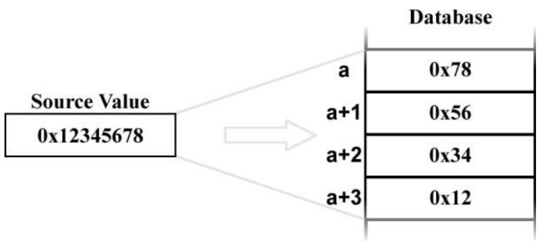


Figure 45: Little Endian Storage

Similarly, data is retrieved from the database starting at the low address. The endianness decides whether the first byte is interpreted as the least-significant byte or the most-significant byte of the multi-byte number.

Here are some examples that demonstrate this. Figure 46 shows how the hex value 0x12345678 is retrieved from the database using a big endian byte order. Since the hex value 12 is at address "a", the lowest address, it is the most significant byte.

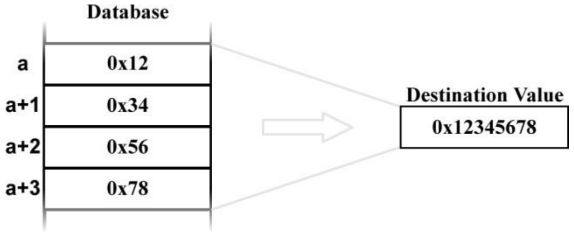


Figure 46: Big Endian Retrieval

Figure 47 demonstrates how the hex value 0x12345678 is retrieved from the database using a little endian byte order. Since the hex value 78 is at the lowest address, it is the least significant byte.

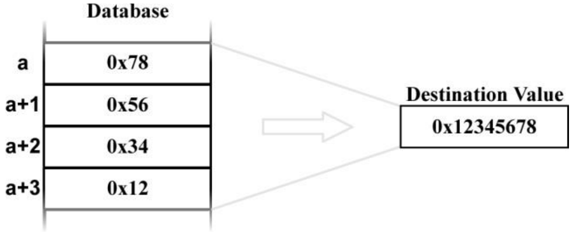


Figure 47: Little Endian Retrieval

The above examples illustrate the data movement to and from the gateway's internal database. This idea helps explain the data movement, as a whole, from one port to the other on the gateway between two different networks. Because networks vary in the manner that they exchange data, endianness selection must be part of the gateway's configuration in order to ensure coherent multi-byte data exchange. There are two data exchange methods used by the supported networks of the gateway.

The first method is used in those networks that define a byte order for how to interpret multi-byte data within an array of bytes. PROFIBUS, for example, defines a big-endian order for multi-byte data, while DeviceNet defines a little-endian order for multi-byte data. These networks exchange I/O data by means of a "bag of bytes" approach, whereas the gateway need not concern itself with where individual values are delimited within the array of bytes itself (as this is determined by the sending or receiving nodes on the networks). The bytes are simply stored into the database in the order they were received. Gateway endianness selection therefore has no effect on data storage or retrieval with a “bag of bytes” protocol driver.

The other method is that used by networks that exchange data by means of an “object value” system, whereas data is exchanged by addressing a certain object to read or write data. Modbus for example, uses registers, while BACnet uses objects such as analog values to exchange data. When multi-byte values are received by the gateway, the bytes must be stored into the database in the order defined by the endianness selected. Likewise, when retrieving multi-byte values from the database for the gateway to transmit, the endianness selected will determine how the data is reconstructed when read from the database.

The selection of the correct byte ordering is crucial for coherent interaction between these two types of networks on the gateway. The following presents examples of how the database endianness affects end-to-end communication between networks and when each byte-ordering scheme should be used.

12.1 Modbus - PROFIBUS Example

This example shows the interaction between a network using an object value method (Modbus) and one using a bag of bytes method (PROFIBUS) to exchange data. The gateway reads holding registers 1 and 2 from the Modbus network, stores the data into the database, and then sends the 4 bytes of input data onto the PROFIBUS network. Figure 48 shows this data movement for the gateway’s database configured as big endian. Because the PROFIBUS specification defines multi-byte values within the byte array to be interpreted as big endian, it is recommended that the database be configured for big-endian byte order when using PROFIBUS. In the example, holding register 1 has a value of 0x1234 and holding register 2 has a value of 0x5678. When the PROFIBUS device receiving the input data from the gateway recombines the two pairs of 2-byte values, the resulting data is 0x1234 and 0x5678, thus successfully receiving the correct values for holding registers 1 and 2.

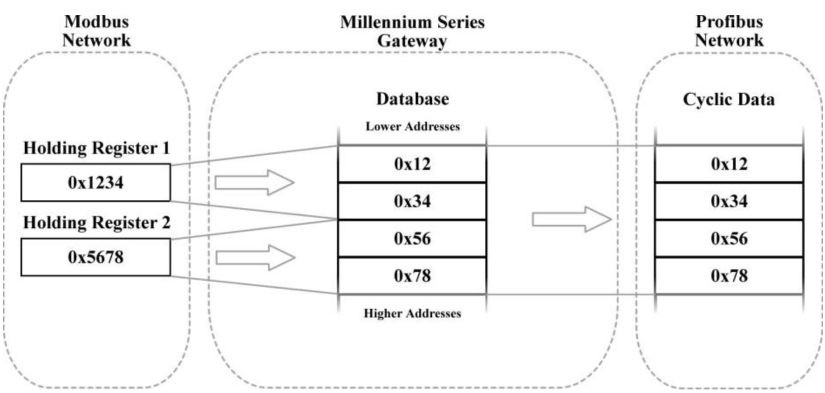


Figure 48: Modbus - PROFIBUS Big Endian

In contrast, Figure 49 shows the effects of configuring the database for little-endian byte order. Holding registers 1 and 2 again have values of 0x1234 and 0x5678, respectively. However, when the PROFIBUS device receiving the input data from the gateway interprets these values, the resulting pairs of 2-byte values become 0x3412 and 0x7856, thus receiving incorrect values for holding registers 1 and 2. Note that in both examples, the PROFIBUS network data is always identical, byte-for-byte, to the gateway's database. For this reason it is important to configure gateways that use a bag-of-bytes style network, such as the PBDP-1000, to use the same endianness as defined for that network.

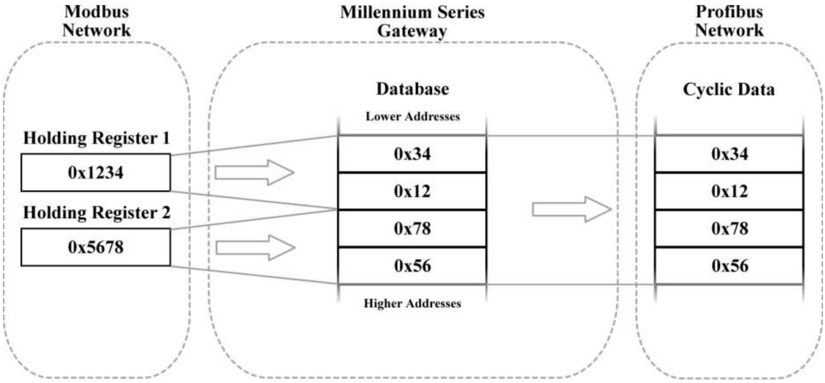


Figure 49: Modbus - PROFIBUS Little Endian

12.2 Modbus - DeviceNet Example

This example shows the interaction between a network using an object value method (Modbus) and one using a bag of bytes method (DeviceNet) to exchange data. The gateway reads holding registers 1 and 2 from the Modbus network, stores the data into the database, and then sends the 4 bytes of input data onto the DeviceNet network. Figure 50 shows this data movement for the gateway's database configured as little endian. Because the DeviceNet specification defines multi-byte values within the byte array to be interpreted as little endian, it is recommended that the database be configured for little-endian byte order when using DeviceNet. In the example, holding register 1 has a value of 0x1234 and holding register 2 has a value of 0x5678. When the DeviceNet device receiving the input data from the gateway recombines the two pairs of 2-byte values, the resulting data is 0x1234 and 0x5678, thus successfully receiving the correct values for holding registers 1 and 2.

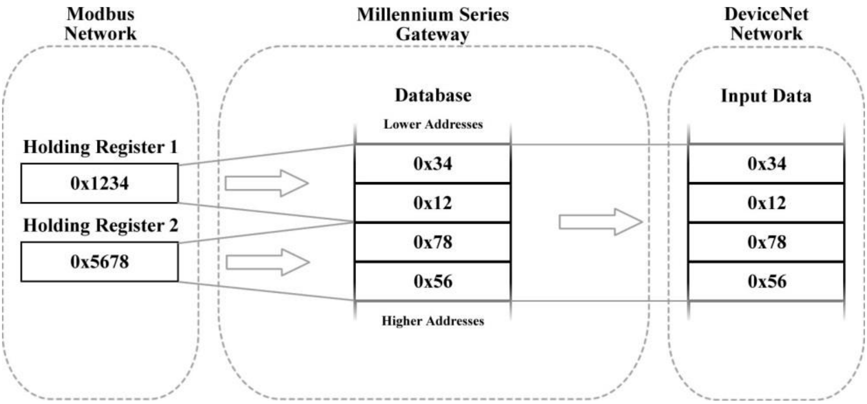


Figure 50: Modbus - DeviceNet Little Endian

In contrast, Figure 51 shows the effects of configuring the database for big-endian byte order. Holding registers 1 and 2 again have values of 0x1234 and 0x5678, respectively. However, when the DeviceNet device receiving the input data from the gateway interprets these values, the resulting pairs of 2-byte values become 0x3412 and 0x7856, thus receiving incorrect values for holding registers 1 and 2. Note that in both examples, the DeviceNet network data is always identical, byte-for-byte, to the gateway's database. For this reason it is important to configure gateways that use a bag-of-bytes style network, such as the DNET-1000, to use the same endianness as defined for that network.

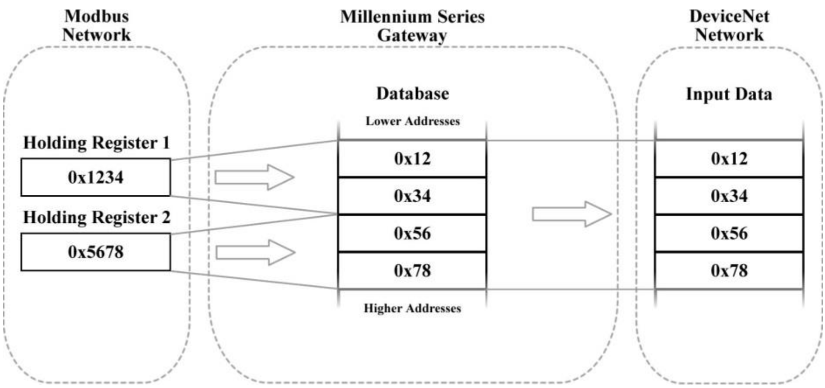


Figure 51: Modbus - DeviceNet Big Endian

12.3 BACnet - DeviceNet Example

This example is quite similar to the previous one as data is exchanged between an object-value style network (BACnet) and a bag-of-bytes style network (DeviceNet). The key difference is that in this example, BACnet Analog Value 0 is a 32-bit value, as opposed to two 16-bit Modbus registers. Here, the gateway reads analog value 0 from the BACnet network, stores the data into the database, and sends the input data onto the DeviceNet network. Figure 52 demonstrates the data flow from the BACnet network to the DeviceNet network through a gateway configured to use a little endian database. Because the DeviceNet specification defines multi-byte values within the byte array to be interpreted as little endian, it is recommended that the database be configured for little-endian byte order when using DeviceNet. In the example, analog value 0 has a value of 0x12345678. When the DeviceNet device receiving the input data from the gateway interprets the 4 bytes, the resulting 4-byte value will be 0x12345678, thus successfully receiving the original value of the BACnet analog value object.

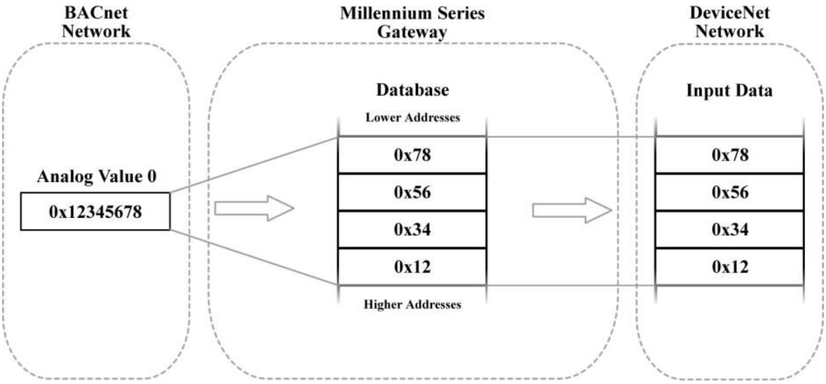


Figure 52: BACnet - DeviceNet Little Endian

Conversely, Figure 53 illustrates the consequences of configuring the database for big-endian byte order using this scenario. Once again, Analog Value 0 has a value of 0x12345678. But now, when the DeviceNet device interprets the 4 bytes of input data sent by the gateway, the resulting 4-byte value is 0x78563412, thus receiving an incorrect value for Analog Value 0. Note that in this example as well, the DeviceNet byte array is identical, byte-for-byte to the database. This example, in conjunction with the previous, demonstrates the dependence on the bag-of-bytes style networks for correct database endianness selection.

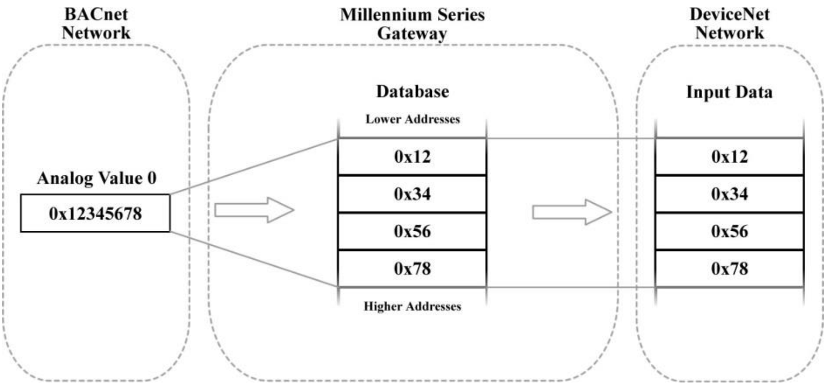


Figure 53: BACnet - DeviceNet Big Endian

12.4 BACnet - Modbus Analog Element Example

This example exhibits two networks that both use an object value scheme to exchange data. In this scenario, the database endianness is irrelevant if the data types are the same for both networks. This example shows communication between a BACnet network and a Modbus network using two 16-bit analog value BACnet objects and two 16-bit Modbus holding registers. As shown in Figure 54, the values from the BACnet network are stored into the database with big-endian byte ordering. Figure 55 shows the values from the BACnet network being stored into the database with little-endian byte ordering. Regardless of the byte-ordering scheme used, the two holding registers on the Modbus network receive the same values. Notice that in both cases, analog values 1 and 2 have values of 0x1234 and 0x5678, respectively, while holding registers 1 and 2 also have values of 0x1234 and 0x5678, respectively. The only difference between the two cases is how the data is being stored internally on the gateway itself.

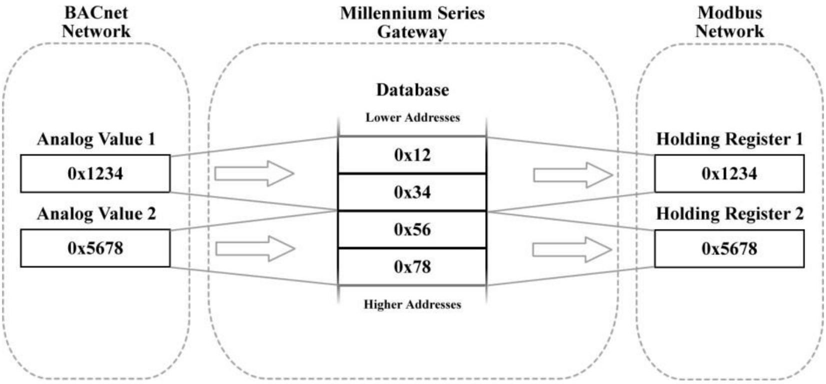


Figure 54: BACnet - Modbus (Analog Objects & Registers) Big Endian

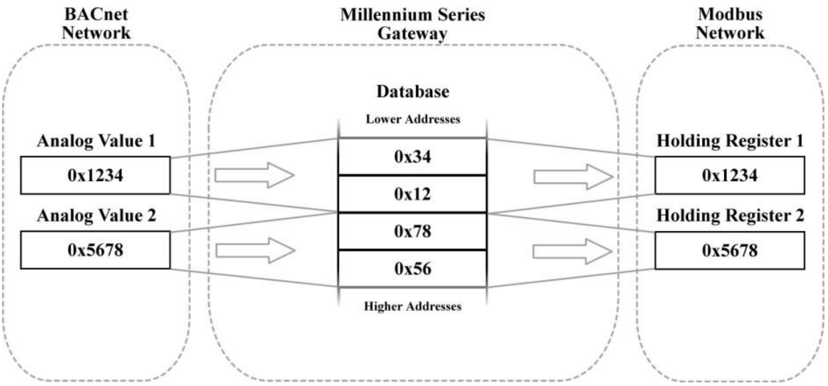


Figure 55: BACnet - Modbus (Analog Objects & Registers) Little Endian

12.5 BACnet - Modbus Binary Element Example

This example also contains two networks that both employ an object value method for exchanging data, but unlike the previous example, the database endianness does affect the end-to-end alignment of the data. In this example, communication is taking place between a BACnet network and a Modbus network using single-bit data elements. The BACnet side is using binary values 1 through 32, while the Modbus side is using coil status 1 through 32. The byte ordering of the database is significant because of the manner in which Modbus coils are mapped in the gateway. Coils (and input statuses) are mapped to registers, not addresses (refer to the Modbus driver documentation for more information). Since registers are 16-bit entities, the byte order of the registers (and by association, the coils), is affected by the endianness configured for the database. BACnet binary objects, however, are mapped on a byte-wise basis into the database.

When the database is configured for a little-endian byte order, binary values 1…8 corresponds to coils 1…8, binary values 9…16 corresponds to coils 9…16, and so on. This can be seen in Figure 56. Notice that the least significant bytes of the registers that the coils map to are placed in the lower memory addresses in the database. Because Modbus discretes are mapped to registers in a bit-wise little-endian fashion, it is recommended that the database be little endian in this scenario so that bit-wise data will align between networks.

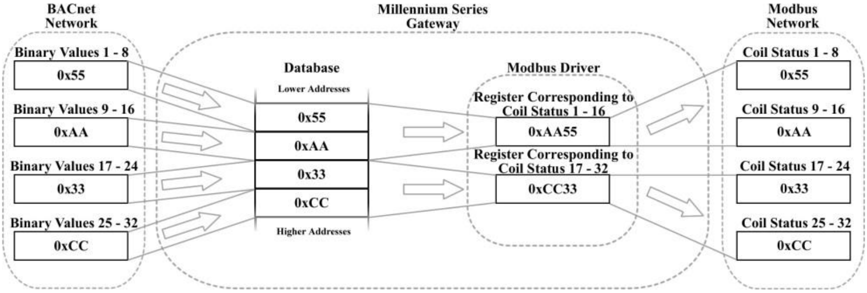


Figure 56: BACnet - Modbus (Binary Objects & Discretes) Little Endian

However, when the database is configured for a big-endian byte order, binary values 1…8 correspond to coils 9…16, binary values 9…16 correspond to coils 1…8, and so on. This can be seen in Figure 57. Since the most significant bytes of the Modbus registers that the coils map to are now mapped to lower addresses, the alignment between the two networks' bit-wise data is byte swapped. While this alignment can still be used, it is much more intuitive when the database is configured to be little endian.

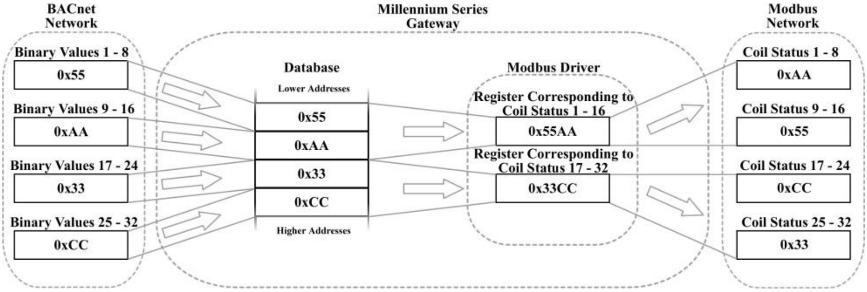


Figure 57: BACnet - Modbus (Binary Objects & Discretes) Big Endian

13. Appendix B: Diagnostics Objects

This section details the information that is enabled by adding a diagnostics object to a service object. Figure 58 diagrams the structure of this status information. Because this 16-byte structure resides in the database at a user-designated location, it can be accessed from any supported network or protocol in order to continuously determine the health and performance of the corresponding service object.

Byte Offset	0	4	8	12	13	14	16
Status Info	TX Counter	RX Counter	RX Error Counter	Current Status	Last Error	Reserved	

Figure 58: Diagnostics Object Format

TX Counter

A 32-bit counter that increments when the driver transmits a packet.

RX Counter

A 32-bit counter that increments when the driver receives a valid packet.

RX Error Counter

A 32-bit counter that increments when the gateway receives an error response packet, or when an error occurs upon reception of a packet.

Current Status

Indicates the status of the most-recently received packet. This field is updated each time the "RX Counter" or "RX Error Counter" increments. Refer to Table 32 for a list of supported codes.

Last Error

Indicates the last reception error that occurred. This field is updated each time the "RX Error Counter" increments. Refer to Table 32 for a list of supported codes.

Reserved

These two bytes are reserved for future use.

Table 32: Status / Error Codes

Status / Error Code (Hex)	Description
0x00	No Error
0xF0	Invalid Data Address
0xF1	Data Error
0xF2	Write To Read-Only
0xF3	Read From Write-Only
0xF4	Target Busy
0xF5	Target Error
0xF6	Cannot Execute
0xF7	Mode Error
0xF8	Other Error
0xF9	Memory Error
0xFA	Receive Error
0xFB	Invalid Function
0xFC	Invalid Packet
0xFD	Security Error
0xFE	Checksum Error
0xFF	Timeout Error

14. Appendix C: BACnet PICS

BACnet Protocol Implementation Conformance Statement (PICS)

Date:	December 11, 2020
Vendor Name:	ICC, Inc.
Product Name:	Mirius Multi-Interface Serial Gateway
Product Model Number:	Mirius
Application Software Version:	V4.300
Firmware Revision:	V4.300
BACnet Protocol Revision:	12

Product Description:

The Mirius is a multiprotocol, multi-interface serial communications gateway. This product supports native BACnet, connecting directly to the MS/TP LAN using baud rates of 9600, 19200, 38400, 57600, 76800, and 115200. The device can be configured as a BACnet Client or as a BACnet Server.

BACnet Standard Device Profile (Annex L):

☐ BACnet Operator Workstation (B-OWS)
☐ BACnet Building Controller (B-BC)
☐ BACnet Advanced Application Controller (B-AAC)
☒ BACnet Application Specific Controller (B-ASC)
☐ BACnet Smart Sensor (B-SS)
☐ BACnet Smart Actuator (B-SA)

BACnet Interoperability Building Blocks Supported (Annex K):

☒ Data Sharing – ReadProperty-A (DS-RP-A)
☒ Data Sharing – ReadProperty-B (DS-RP-B)
☒ Data Sharing – ReadPropertyMultiple-B (DS-RPM-B)
☒ Data Sharing – WriteProperty-A (DS-WP-A)
☒ Data Sharing – WriteProperty-B (DS-WP-B)
☒ Data Sharing – WritePropertyMultiple-B (DS-WPM-B)
☒ Data Sharing – COV-B (DS-COV-B)
☒ Device Management – Dynamic Device Binding-A (DM-DDB-A)
☒ Device Management – Dynamic Device Binding-B (DM-DDB-B)
☒ Device Management – Dynamic Object Binding-B (DM-DOB-B)
☒ Device Management – DeviceCommunicationControl-B (DM-DCC-B)
☒ Device Management – ReinitializeDevice-B (DM-RD-B)
☒ Device Management – TimeSynchronization-B (DM-TS-B)*
☒ Device Management – UTCTimeSynchronization-B (DM-UTC-B)*

* Available only when Real-time Clock Settings are enabled

Segmentation Capability:

☐ Able to transmit segmented messages Window Size ________
☐ Able to receive segmented messages Window Size ________

Standard Object Types Supported:

Property	Object Type									
	Device	BI	BO	BV	AI	AO	AV	MSI	MSO	MSV
Object Identifier	W	R	R	R	R	R	R	R	R	R
Object Name	W	R	R	R	R	R	R	R	R	R
Object Type	R	R	R	R	R	R	R	R	R	R
System Status	R									
Vendor Name	R									
Vendor Identifier	R									
Model Name	R									
Firmware Revision	R									
Application Software Version	R									
Protocol Version	R									
Protocol Revision	R									
Protocol Services Supported	R									
Protocol Object Types Supported	R									
Object List	R									
Max APDU Length Accepted	R									
Segmentation Supported	R									
Local Time*	W									
Local Date*	W									
UTC Offset*	W (-840…720)									
Daylight Savings Status*	W									
APDU Timeout	W (10…65535)									
Number Of APDU Retries	W (0…10)									
Max Master	W (1…127)									
Max Info Frames	W (1…100)									
Device Address Binding	R									
Database Revision	R									
Active COV Subscriptions	R									
Present Value		R	W	W	R	W	W	R	W	W
Status Flags		R	R	R	R	R	R	R	R	R
Event State		R	R	R	R	R	R	R	R	R
Reliability		R	R	R	R	R	R	R	R	R
Out Of Service		R	R	R	R	R	R	R	R	R
Number Of States								R	R	R
Units					R	R	R			
Priority Array			R	R		R	R		R	R

Relinquish Default			R	R		R	R		R	R
COV Increment					W	W	W			
Polarity		W	W							
Inactive Text		R	R	R						
Active Text		R	R	R						

R – Readable using BACnet services
W – Readable and writable using BACnet services

***** Available only when Real-time Clock Settings are enabled

Data Link Layer Options:

☐ BACnet IP, (Annex J)
☐ BACnet IP, (Annex J), Foreign Device
☐ ISO 8802-3, Ethernet (Clause 7)
☐ ANSI/ATA 878.1, 2.5 Mb. ARCNET (Clause 8)
☐ ANSI/ATA 878.1, RS-485 ARCNET (Clause 8), baud rate(s) ______
☒ MS/TP master (Clause 9), baud rate(s): 9600, 19200, 38400, 57600, 76800, 115200
☐ MS/TP slave (Clause 9), baud rate(s): ______
☐ Point-To-Point, EIA 232 (Clause 10), baud rate(s): ______
☐ Point-To-Point, modem, (Clause 10), baud rate(s): ______
☐ LonTalk, (Clause 11), medium: ______
☐ Other: ______

Device Address Binding:

Is static device binding supported? (This is currently for two-way communication with MS/TP slaves and certain other devices.) ☒ Yes ☐ No

Networking Options:

☐ Router, Clause 6 - List all routing configurations
☐ Annex H, BACnet Tunneling Router over IP
☐ BACnet/IP Broadcast Management Device (BBMD)
Does the BBMD support registrations by Foreign Devices? ☐ Yes ☐ No

Network Security Options:

☒ Non-secure Device - is capable of operating without BACnet Network Security
☐ Secure Device - is capable of using BACnet Network Security (NS-SD BIBB)
☐ Multiple Application-Specific Keys:
☐ Supports encryption (NS-ED BIBB)
☐ Key Server (NS-KS BIBB)

Character Sets Supported:

Indicating support for multiple character sets does not imply that they can all be supported simultaneously.

☒ ISO 10646 (UTF-8)	☐ IBM™/Microsoft™ DBCS	☐ JIS X 0208
☐ ISO 10646 (UCS-4)	☐ ISO 10646 (UCS-2)	☐ ISO 8859-1

If this product is a communication gateway, describe the types of non-BACnet equipment/networks(s) that the gateway supports:

Refer to protocol-specific manuals for other supported protocols.

INDUSTRIAL CONTROL COMMUNICATIONS, INC.

230 Horizon Drive, Suite 100
Verona, WI USA 53593
Tel: [608] 831-1255 Fax: [608] 831-2045

http://www.iccdesigns.com

Printed in U.S.A