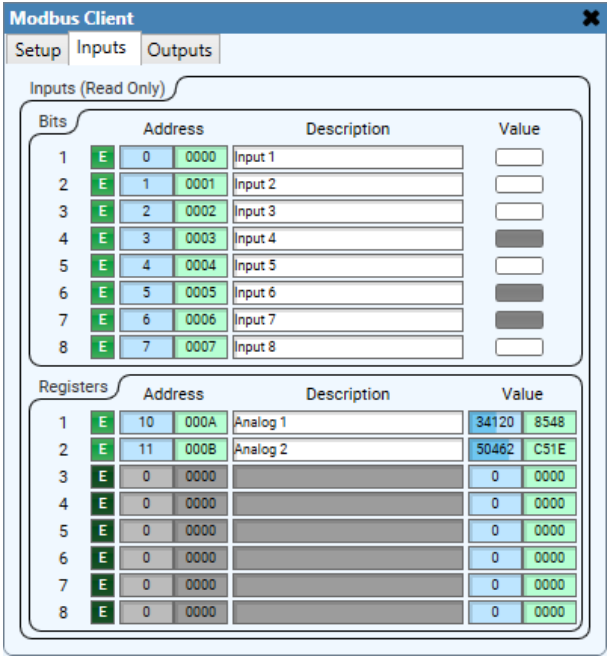
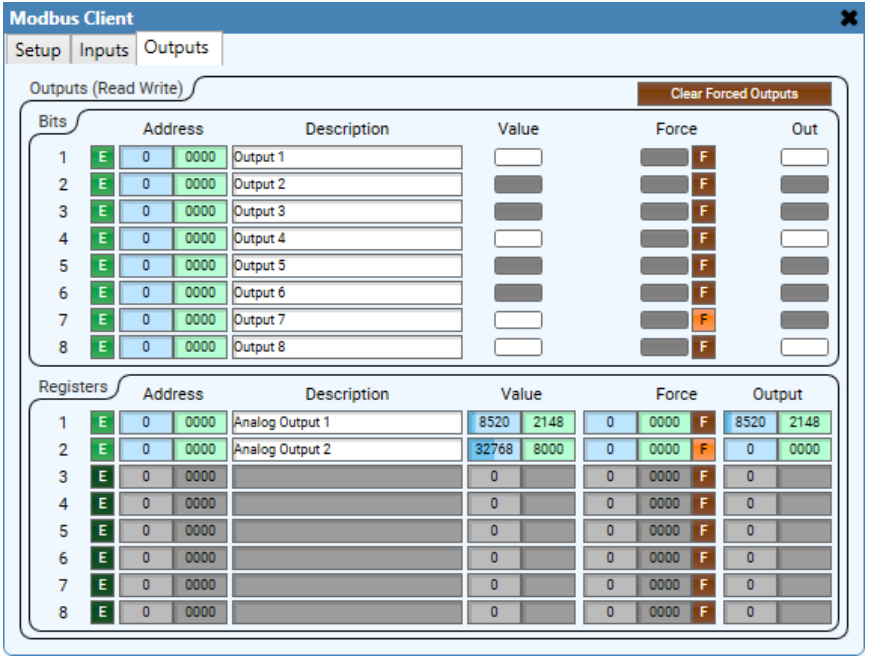
Plugin Guide

MODBUS CLIENT

Version 1.1.2
July 11, 2024





Programation.io

MODBUS CLIENT

allows Q-SYS to interface with external Modbus Server devices over standard TCP/IP communication. This allows Q-SYS to receive inputs and control outputs of automation equipment that supports the Modbus over TCP protocol. This can greatly expand the IO capabilities of a Q-SYS system by allowing both digital and analog signals to be easily integrated into a design.

The plugin offers up to 999 each of Input Bits, Input Registers, Output Bits, and Output Registers offering a staggering number of control IO points – hundreds more than a typical control system. Inputs are polled from the Modbus Server Device up to 20 times per second, configurable in 50ms increments. Outputs are updated up to 30 times per second.

Both Input and Output Bits are represented as toggle buttons for easy control pin wiring. Registers are represented as integer fields with possible values from 0 to 65,535, and as hexadecimal string fields with values from 0x0000 to 0xFFFF.

Each Input and Output in the plugin may be configured for any IO address and do not have to be sequential. Unused IO can be disabled to reduce clutter and confusion. IO may also be temporarily disabled while testing or commissioning is ongoing.

Outputs offer an “Override” feature which allows output values to be specified within the plugin UI and thus ignoring data from the control pin. When an Override is active, the provided value will be applied to the output allowing for easier system testing, commissioning, and temporary bypass.

Outputs also offer a “Sync” feature which allows them to be updated in Q-Sys whenever they are changed on the server device by another actor, such as PLC logic or by another control system.

Warning

The Modbus Client plugin is not intended to be used in manufacturing, motion, or safety applications, or any other applications where a software failure could result in damage to equipment or property, injury to persons or animals, or loss of life. The plugin is only intended to expand the IO capabilities of a Q-SYS system for control usage within an entertainment-related permanent installation in which General Purpose IO is not sufficient.

While Programation LLC attempts to mitigate software errors, we do not guarantee that our software is free from errors as outlined in the Software License Agreement. By using this plugin, you assume all risk associated with any consequences that may arise if the plugin should malfunction, including but not limited to, a malfunction of the Q-SYS system, the scripting environment, or with the intellectual property of Programation LLC.

Important Disclaimer

Due to data processing limitations within the Q-SYS control engine, information may only be processed up to 30 times per second, or about every 33ms. Further, the Modbus Client plugin can only poll and refresh input data from a Server Device up to 20 times per second or every 50ms. These refresh rates are far too slow for use in any motion or safety applications. These refresh rates may also be considered too slow for certain control applications. Careful consideration should be given when designing and implementing a system that relies on this plugin or third-party devices. These performance rates are approximately equal to the refresh rates of native Q-SYS GPIO which does suffice for most general-purpose control applications.

Properties

Input Bits

an `integer` between `0` and 999 that defines the number of input bits available for configuration

Input Registers

an `integer` between `0` and 999 that defines the number of input registers available for configuration

Output Bits

an `integer` between `0` and 999 that defines the number of output bits available for configuration

Output Registers

an `integer` between `0` and 999 that defines the number of output registers available for configuration

Is Required

a `boolean` that sets whether the target device is required for system functionality

Sync Outputs

a `boolean` that sets whether output states should be read back from the remote device to maintain synchronization. Keeping this disabled will cause output controls to be “write only” and not update in Q-Sys if the remote device changes the output value.

Registered Owner

a `string` that lists the license holder name of the plugin registration for this installation

Registered Project

a `string` that lists the licensed project name of the plugin registration for this installation

Registered Key

a `string` that lists the license key code of the plugin registration for this installation

Controls

Status

displays various status messages to aid in monitoring the connection with the Server Device. Communication errors will show as a fault. When a force is present on an output, the status will show as Compromised as a reminder that the system has been overridden.

Server Port

sets the TCP communication port number to use when connecting to the Server Device

Server Address

sets the IP address to use when connecting to the Server Device

Bus Address

sets the Modbus Bus Address (or Device ID) of the Server Device. This is useful if more than one Server exists at the same IP address. Refer to the Notes section for more details.

Polling Steps

sets the polling frequency of inputs from the Server Device in increments, or steps, of 50 milliseconds

Polling Interval

displays the time of each input poll

Input Enable

a toggle button that sets whether this Input is used and should be polled

Input Dec Address

an integer field that ranges from 0 to 65,536 that sets the memory address of the input in decimal

Input Hex Address

a string field that ranges from 0x0000 to 0xFFFF that sets the memory address of the input in hexadecimal

Controls

Input Description
a text field that provides human-readable labeling of each input

Input Bit Value
a toggle button that represents the state of the input on the Server Device

Input Register Dec Value
an integer field that ranges from 0 to 65,535 that represents the value of the register on the Server Device in decimal

Input Register Hex Value
a string field that represents the register value in hexadecimal, ranging from 0x000 to 0xFFFF

Output Enable
a toggle button that sets whether this Output is used and should send updates to the Server Device

Output Dec Address
an integer field that ranges from 0 to 65,536 that sets the memory address of the output of the Server Device in decimal

Output Hex Address
a string field that ranges from 0x0000 to 0xFFFF that sets the memory address of the output in hexadecimal

Output Description
a text field that provides human-readable labeling of each output

Output Bit Value
a toggle button that sets the desired state of the output of the Server Device

Output Bit Override Value
a toggle button that sets the value of the output in the overridden state

Controls

Output Bit Override Enable

a toggle button that overrides the output bit's value with the value specified by the Bit Override Value toggle button. When active, the output will only follow the overridden value.

Output Bit Output

a toggle button that represents the actual output state, after considering the overridden state

Output Bit Sync

a toggle button the enables the bit to be read back from the remote device to keep the current state synchronized with the plugin

Output Bit Read Only

a toggle that sets whether the output bit should be read-only within Q-Sys, causing the plugin to only show the current state of the output on the remote device

Output Register Dec Value

an integer field ranging from 0 to 65,535 that sets the desired decimal value of the output of the Server Device

Output Register Hex Value

a string field ranging from 0x0000 to 0xFFFF that indicates the hexadecimal value of the decimal value

Output Register Dec Override Value

an integer field ranging from 0 to 65,535 that sets the decimal value of the output in a overridden state

Output Register Hex Override Value

a string field ranging from 0x0000 to 0xFFFF that sets the hexadecimal value of the output in a overridden state

Output Register Override Enable

a toggle button that overrides the output register's value with the value specified by the Register Override Dec/Hex Value fields. When active, the output will only follow the overridden values.

Output Register Dec Output

an integer field ranging from 0 to 65,535 that represents the actual output value in decimal, after considering the overridden state

Output Register Hex Output

a string field ranging from 0x0000 to 0xFFFF that represents the actual output value in hexadecimal, after considering the overridden state

Output Register Sync

a toggle button the enables the register to be read back from the remote device to keep the current state synchronized with the plugin

Output Register Read Only

a toggle that sets whether the output register should be read-only within Q-Sys, causing the plugin to only show the current state of the output on the remote device

Notes

Implemented Modbus Functions

The Modbus Client plugin is compliant with the official Modbus TCP protocol and uses the methods:

- 0x01 (Read Output Coils)
- 0x02 (Read Discrete Inputs)
- 0x03 (Read Holding/Output Registers)
- 0x04 (Read Input Registers)
- 0x15 (Write Multiple Coils)
- 0x16 (Write Multiple Registers)

I/O Addressing – Supporting both Modbus TCP and JBus

Modbus TCP is a somewhat modified version of the original Modbus Serial communication protocol. Modbus TCP has certain limitations to the number of addresses available and the quantity of coils and registers that can be communicated in a single request. This is to keep a TCP message under the default MTU size. JBus, on the other hand, is more relaxed with these requirements. The most noticeable difference between the two protocols for system programmers is that Modbus TCP is 0-based in its memory addressing while JBus is 1-based.

The Modbus Client plugin is designed to not be concerned with whether the requests are 0-based or 1-based. It is up to the Q-SYS programmer to define the memory addresses for their use-case appropriately. The plugin does not apply any offset to the address numbers listed. If no I/O are addressed as 0, and a request is issued for memory address 0, then the request will fail.

Consequently, the plugin allows I/O to be addressed as high as 65,536 or 0x10000. This is to maintain compliancy with JBus, however it is an illegal data address for the Modbus protocol. Since all hexadecimal values are limited to 4 digits (2 bytes), the plugin will show a decimal address of 65,536 but a hexadecimal address of 0xFFFF.

Notes

Hexadecimal readouts of address and value fields are just for the programmer's convenience and do not have any practical effect on the data transmission. The plugin only considers decimal values for requests and transmissions. When a hexadecimal value is entered or is changed, however, the corresponding decimal field will also be updated to the correct value.

While the plugin will attempt to receive either Modbus TCP or JBus style messages without additional user configuration, it is recommended that messages stay under the default MTU size of 1500 bytes, otherwise the message may be split into multiple packets and the request may fail to be processed correctly.

Non-Sequential and Non-Contiguous Addresses

As the Modbus Client plugin allows any Input or Output to have any address, addressing may be non-sequential and non-contiguous. This allows flexibility in only creating I/O that is needed at any address. Several installations typically prefer to use logical numbering conventions for various groups of I/O which often skips over blocks of addresses. The plugin supports this without having to have a large number of unused controls.

When a poll request is made for inputs, all inputs are arranged first by bits, then by registers, in ascending order based on their Modbus Address, regardless of line number in the plugin. Contiguous regions of address numbers are then requested in a single message. Each non-contiguous break will result in a separate message being transmitted. A large number of requests may result if many non-contiguous input addresses are requested. Performance may degrade if the requests cannot be fulfilled faster than the specified polling time. For this reason, the polling time should be set to a slower interval when using a large number of I/O, depending on the performance of the Server Device.

I/O Priority

All enabled inputs are polled on every polling interval. Outputs are only transmitted when a control is changed or when the plugin connects to the Server Device. As Q-SYS only updates controls at a rate of 30 hertz, this becomes the frequency limit to updating or controlling outputs. Transmitting output write messages and input read requests are generated

Notes

asynchronously within the plugin. To prevent overwhelming the server device with requests, only one message may be in flight at a time. Messages are queued within the plugin and are only transmitted when a response is received. The queue honors output write requests first before sending poll requests from the queue. This allows outputs to be more time accurate at the minor expense of delaying input triggers.

When the Sync Outputs property is Enabled, and any outputs have their Sync button on, the outputs will be added to the polling queue on each polling interval. These will be updated after the input bits and registers. If a remote device has a force enabled on an output, that is, it is set to ignore change requests, this will cause the plugin to appear as if the outputs are not updating or changing.

Bus Address

Traditional Modbus is a serial line protocol that supported multiple devices to be connected to the same line. Therefore, the protocol includes a "Bus Address" field which allows requests to target a specific device on the communication bus. Modbus TCP offers the same functionality for cases where multiple servers share the same IP address. The default for most devices is a Bus Address of zero.

Multiple instances of the Modbus Client plugin can run in the same Q-SYS design simultaneously provided that they are connecting to different Server Devices. It is not recommended for two Client plugins to attempt to connect to the same Server Device.

Change Log

V1.0.0 – December 2, 2021

Original release

V1.0.2 – April 12, 2022

- Fixed an issue where messages would not be sent after the connection with a device was dropped and reconnected which could lead to high memory consumption causing a core crash.

- Added a safeguard to ensure the maximum amount of memory used by the plugin would not impact system stability. A status fault message will appear when this occurs.

- Lengthened the logging interval of repetitive verbose log messages from 5 seconds to 30 seconds, cutting down on log entries.

- Improved connection monitoring to report connection interruption faster (within 1 second).

- Improved status reporting when multiple read or write errors were received which caused the status control to flicker violently.

- Bug fix where changing the polling interval caused a script error and failed to update the displayed polling time in milliseconds.

Change Log

V1.0.3 – April 13, 2022

- Modified the connection timeout to be equal to the polling interval to better detect and manage dead TCP sessions

- Added a limit to the number of messages to be queued to prevent increasing memory consumption leading to a core crash. An error message will appear when this occurs and the connection to the remote device will be reset.

V1.0.4 – May 25, 2022

- Improved messaging with the remote device upon TCP connection establishment allowing for immediate controls updates upon connection. This is most noticeable when using a slower polling interval.

V1.1.0 – March 21, 2023

- Changed the “Force” terminology to “Override” to minimize confusion as to the influence over the end device

- Added new property “Sync Outputs” to enabling reading output states from the remote server device

- Added new controls “Sync” and “Read Only” on outputs, when the “Sync Outputs” property is Enabled

- Output controls in the plugin UI are now read-only and cannot be directly manipulated

- Improved the change detection method to reduce unnecessary network traffic

- Added better connection handling, timeout detection, and reconnection with the remote device

- Added a safeguard to prevent polling faster than what the network or remote server can handle

Change Log

- Added a safeguard to reconnect if the remote device stops communicating for 5 times the polling interval

- Added a default message heartbeat to keep connections alive when no input or output controls are enabled

V1.1.1 – JUNE 28, 2024

- Changed the input controls from read-only to be editable to allow for easier system commissioning without needing to rely on the connected device to change input values. Now a Q-SYS design can be fully tested from the plugin without hardware connection.

- Modified the behavior of the plugin to not throw a fault when a device does not support the Vendor or Product Code Identification function used by the plugin as a standby heartbeat mechanism.

V1.1.2 – JULY 11, 2024

- Added demo functionality for a time-limited trial of the plugin.