INDUSTRIAL CONTROL COMMUNICATIONS, INC.

EtherNet/IP Client Driver Manual

October 30, 2014

TABLE OF CONTENTS

1 EtherNet/IP Client

1.1 Overview

This driver supports the EtherNet/IP client protocol, including the CSP variant. This driver can be used to access information on EtherNet/IP and CSP server devices which support class 1 I/O messaging and class 3 explicit messaging. The EtherNet/IP client driver uses service objects to produce and consume data and to issue explicit read and write requests to the server device. EtherNet/IP service objects make use of an additional driver construct called a connection object in order to target server devices. A connection object can be thought of as a communication channel or "pipe" which is created between the driver and the server device, independent of the underlying service objects that may later make use of that communication channel to transfer service object requests.

A connection object defines a connection to a specific endpoint (IP address). For each connection object, however, up to two underlying EtherNet/IP client connections can be simultaneously established: one connection is reserved for class 1 I/O messaging and the other connection is reserved for class 3 explicit messaging. The class 3 explicit messaging connection is capable of managing any number of service objects targeting the remote endpoint defined by the connection object (the service objects are serviced sequentially). Several different types of class 3 requests are available to match the capabilities of a variety of EtherNet/IP and CSP server devices.

Ethernet-enabled Allen-Bradley legacy PLCs (such as the PLC-5/E, SLC-5/05 and MicroLogix platforms) use a protocol called CSP (Client Server Protocol) to communicate via Ethernet. The variant of CSP used by these PLCs is also known as "PCCC" (Programmable Controller Communication Commands) and "AB Ethernet". The driver supports CSP via the "Typed Read" and "Typed Write" services for direct connectivity to these PLCs, as well as for connectivity to server devices that have traditionally connected to these PLCs.

Only class 3 connections can be used when connecting to CSP server devices, as these devices do not support class 1 data transfers. Additionally, not all EtherNet/IP server devices support both class 1 and class 3 connections: please confirm the capabilities of the targeted server device when configuring the driver's service objects.

In this section, the terms "O→T" and "T→O" are EtherNet/IP specification nomenclature which stands for "Originator to Target" and "Target to Originator", respectively. Typically, the originator is the client and the target is the server.

Some notes of interest are:

- The driver supports the EtherNet/IP protocol (release 1.0), administered by the Open DeviceNet Vendor Association (ODVA).
- This product has been self-tested by ICC, Inc. and found to comply with ODVA EtherNet/IP Conformance Test Software Version A-6.
- The identity object's product type code is 12 (Communications Adapter).
- Each connection object can simultaneously support class 1 (I/O) and class 3 (explicit) connections. Each class 3 connection can support multiple service objects of various types.

- Class 1 and class 3 connections operate independently of each other.
- The driver supports up to 4 connection objects and up to 64 total service objects.
- Both automatic and manual triggering mechanisms are available for each service object.
- Class 1 connections do not support configuration data.
- Data Table Read, Data Table Write, Typed Read, and Typed Write services can use the slot number to target the (Logix, PLC5, SLC5/05 etc.) controller in the rack.
- The driver supports Produced Tag connections to cyclically access data on a -Logix PLC.
- The driver only supports multicast transport in the T→O direction.
- If a class 1 connection's consuming half (O→T) times out, then the producing half (T→O) will also time-out and will stop producing.
- For class 1 connections, if the **API** (actual packet interval) returned by the server is within ±10ms of the client's configured **RPI** setting, then the connection is successful. This ±10ms restriction is enforced in order to notify the user (by way of an unsuccessful connection) that the server's data rate capabilities "substantially" differ from the configured client settings. The connections will always produce and consume data according to the **API**.

1.2 Supported Class 3 Services

The driver supports the following class 3 (explicit messaging) services through its various service objects:

Get Attribute Single

Located in the *CIP Generic Service Object*. Returns the contents of the specified attribute.

Set Attribute Single

Located in the *CIP Generic Service Object*. Modifies the contents of the specified attribute.

Data Table Read

Located in the *Data Table R/W Service Object*. Reads data associated with a tag name.

Data Table Write

Located in the *Data Table R/W Service Object*. Writes data associated with a tag name.

Typed Read

Located in the *Typed R/W Service Object*. Reads a block of data from a file number and offset.

Typed Write

Located in the *Typed R/W Service Object*. Writes a block of data to a file number and offset.

1.3 Client Settings

Explicit Messaging Scan Rate

This field does not apply to class 1 and produced tag service objects. This is the time in milliseconds the driver will wait between sending class 3 explicit messaging requests. This is a useful feature to reduce overall network utilization, or for certain devices or infrastructure components (such as radio modems) that may not be capable of sustaining the maximum packet rates that the driver is capable of producing. The start time for this delay is taken with respect to the moment at which the driver is capable of sending the next packet (due to either reception or timeout of the previous request). If no additional time is required, setting this field to 0 instructs the driver to send its next request packet as soon as possible.

1.4 Connection Object Settings

Name

A unique name used for identifying the connection object. Enter a string of up to 16 characters in length.

IP Address

Defines the IP address of the server device to be targeted by the connection object. All connection object IP address settings must be unique.

1.5 Service Object Settings

The driver uses service objects to describe what services the driver should perform. Due to the wide variety of EtherNet/IP and CSP server devices that are available on the market, a variety of different service object types are available. Each type provides a different mechanism for accessing information on a server device.

- When class 1 (I/O) service objects are in use, the driver will produce and consume data from/to the database in a cyclic fashion.
- When produced tag service objects are in use, the driver will only consume data to the database in a cyclic fashion.
- When explicit messaging service objects are in use, the driver will generate an explicit read or write request. For each service object, the driver will continually read the targeted elements defined within the service object from the designated server, storing the values in the database (if the read function is enabled). When data in the database changes where the service object elements are mapped, a write request is generated to the designated server notifying it of the changed values (if the write function is enabled).

In all cases, each service object is activated according to its designated triggering mechanism (refer to section 1.6). Also note that class 1 service objects and produced tag service objects are both types of class 1 (I/O) messages, and that only one class 1 message type is supported per connection object.

1.5.1 Class 1 Service Object Settings

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Slot

Defines the destination device slot number in a rack for routing purposes. This field is optional: a setting of “N/A” (Not Applicable) specifies that a slot number entry will not be included during connection establishment. Devices typically require a setting of “N/A” for successful connection establishment, but please refer to each server device’s documentation for guidance regarding appropriate slot number setting or omission.

Produced Database Address

Specifies the database address to which the first produced byte of I/O data is mapped. The configuration studio will not allow entry of a starting database address that will cause the produced data array to run past the end of the database. The highest valid database address, therefore, will depend on the number of bytes to be produced. The produced data array cannot overlap the consumed data array.

Produced Bytes

Specifies the number of bytes (0…509) to produce (send to the server device).

Consumed Database Address

Specifies the database address to which the first consumed byte of I/O data is mapped. The configuration studio will not allow entry of a starting database address that will cause the consumed data array to run past the end of the database. The highest valid database address, therefore, will depend on the number of bytes to be consumed. The produced data array cannot overlap the consumed data array.

Consumed Bytes

Specifies the number of bytes (0…509) to consume (receive from the server device).

Configuration Connection Point

Specifies the assembly instance number (0…65535) for the configuration connection included in the initial “forward open” request to the server. While this assembly instance number is required, no additional configuration data is included in the “forward open” request. Some EtherNet/IP servers ignore this field.

Produced Connection Point

Specifies the assembly instance number (0…65535) for the produced data connection (O→T direction).

Consumed Connection Point

Specifies the assembly instance number (0…65535) for the consumed data connection (T→O direction).

Enable Run/Idle Header

Check this checkbox to include the optional run/idle header in the produced data packets. Most EtherNet/IP server devices expect the run/idle header to be present in the data packets they receive.

RPI

Specifies the Requested Packet Interval in units of milliseconds. The minimum supported RPI is 10ms.

Multiplier

Specifies the connection timeout multiplier. The connection's timeout time is determined by multiplying the RPI value by the connection timeout multiplier.

1.5.2 CIP Generic Service Object Settings

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Slot

Defines the destination device slot number in a rack for routing purposes. This field is optional: a setting of "N/A" (Not Applicable) specifies that a slot number entry will not be included during connection establishment. Devices typically require a setting of "N/A" for successful connection establishment, but please refer to each server device's documentation for guidance regarding appropriate slot number setting or omission.

Database Address

Defines the database address where the first byte of this service object's data value is mapped. The configuration studio will not allow entry of a starting database address that will cause the data to run past the end of the database. The highest valid database address, therefore, will depend on the configured number of bytes.

Number of Bytes

Specifies the number of bytes (1…240) that are to be accessed by this service object. By appropriately setting this field, access to any supported CIP data type (INT, DWORD, REAL, arrays etc.) is possible.

Class

Specifies the class code (1…65535) of the targeted CIP object.

Instance

Specifies the instance number (1…65535) of the targeted CIP object.

Attribute

Specifies the attribute number (1…65535) of the targeted CIP object.

Get Attribute Single

Check this checkbox to enable reading via the "get attribute single" service. If reading is enabled, then the service object will read from the server unless a pending write exists. Refer to section 1.6 for further information regarding the triggering of read services.

Set Attribute Single

Check this checkbox to enable writing via the "set attribute single" service. If writing is enabled, then when values encompassed by this service object change in the database, these changes will be written down to the targeted server. Refer to section 1.6 for further information regarding the triggering of write services.

1.5.3 Data Table R/W Service Object Settings

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Slot

Defines the destination device slot number in a rack for routing purposes. This field is optional: a setting of "N/A" (Not Applicable) specifies that a slot number entry will not be included during connection establishment. Devices typically require a setting of "N/A" for successful connection establishment, but please refer to each server device's documentation for guidance regarding appropriate slot number setting or omission.

Database Address

Defines the database address where the first element of this service object's data is mapped. The configuration studio will not allow entry of a starting database address that will cause the data to run past the end of the database. The highest valid database address, therefore, will depend on the configured *Database Data Type* and *Number of Elements*.

Number of Elements

Specifies the number of elements that are to be accessed by this service object. The cumulative total number of bytes, calculated by multiplying the *Number of Elements* times the size of the designated *Element Data Type* (1 for 8-bit, 2 for 16-bit and 4 for 32-bit data types) must be 1…240.

Database Data Type

Specifies how the value will be stored in the database for each *Tag* element in this service object. 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 when converted to the *Element Data Type* for transmission over the network. Select the desired data type from this dropdown menu. For most situations, it is strongly recommended to match the *Database Data Type* and the *Element Data Type*.

Element Data Type

Specifies how the value will be interpreted for each *Tag* element in this service object. This defines the size of each element, 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 when converted from the *Element Data Type* to the *Database Data Type*. Select the desired data type from this dropdown menu. For most situations, it is strongly recommended to match the *Database Data Type* and the *Element Data Type*.

Multiplier

The amount that associated network values are scaled by prior to being stored into the database or after being retrieved from the database. Upon retrieval from the database, raw data is multiplied by the multiplier to produce a network value (to be sent to the device). Similarly, network values (read from the device) are divided by the multiplier before being stored into the database.

Tag

Specifies the tag name on the target device. Enter a string from 1 to 40 characters in length. Note that tag names may only contain alphanumeric characters and the underscore character.

Offset

Specifies the *Tag* offset (0…65535). When accessing an individual tag, set this field to 0. When accessing an array tag, this field's setting designates the starting array index to access.

Data Table Read

Check this checkbox to enable reading via the "data table read" service. If reading is enabled, then the service object will read from the server unless a pending write exists. Refer to section 1.6 for further information regarding the triggering of read services.

Data Table Write

Check this checkbox to enable writing via the "data table write" service. If writing is enabled, then when values encompassed by this service object change in the database, these changes will be written down to the targeted server. Refer to section 1.6 for further information regarding the triggering of write services.

1.5.4 Typed R/W Service Object Settings

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Slot

Defines the destination device slot number in a rack for routing purposes. This field is optional: a setting of "N/A" (Not Applicable) specifies that a slot number entry will not be included during connection establishment. Devices typically require a numeric slot setting for successful connection establishment, but please refer to each server device's documentation for guidance regarding appropriate slot number setting or omission. For example, when targeting an Allen-Bradley processor (such as a -Logix, PLC-5/E, or SLC-5/05 platform), the *Slot* is typically set to 0, as this is commonly where the processor resides in the rack.

Database Address

Defines the database address where the first element of this service object's data is mapped. The configuration studio will not allow entry of a starting database address that will cause the data to run past the end of the database. The highest valid database address, therefore, will depend on the configured *Database Data Type* and *Number of Elements*.

Number of Elements
Specifies the number of elements (1…120) that are to be accessed by this service object.

Database Data Type
Specifies how the value will be stored in the database for each *File* element in this service object. 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 when converted to the *Element Data Type* for transmission over the network. Select the desired data type from this dropdown menu. For most situations, it is strongly recommended to match the *Database Data Type* and the *Element Data Type*.

Element Data Type
Specifies how the value will be interpreted for each *File* element in this service object. This defines the size of each element, 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 when converted from the *Element Data Type* to the *Database Data Type*. Select the desired data type from this dropdown menu. For most situations, it is strongly recommended to match the *Database Data Type* and the *Element Data Type*.

File Number
Specifies the file number (0…65535) on the target device.

Offset
Designates the offset of the starting element (0…65535) to access within the file.

Multiplier
The amount that associated network values are scaled by prior to being stored into the database or after being retrieved from the database. Upon retrieval from the database, raw data is multiplied by the multiplier to produce a network value (to be sent to the device). Similarly, network values (read from the device) are divided by the multiplier before being stored into the database.

Typed Read
Check this checkbox to enable reading via the "typed read" service. If reading is enabled, then the service object will read from the server unless a pending write exists. Refer to section 1.6 for further information regarding the triggering of read services.

Typed Write
Check this checkbox to enable writing via the "typed write" service. If writing is enabled, then when values encompassed by this service object change in the database, these changes will be written down to the targeted server. Refer to section 1.6 for further information regarding the triggering of write services.

1.5.5 Produced Tag Service Object Settings
Produced tag messages are a unidirectional form of class 1 implicit messaging that is used only to consume data from a -Logix PLC produced tag. Prior to using produced tag messaging, it is suggested that users read the whitepaper titled "*Accessing Data on a Logix PLC with the ICC*

ETH-1000 Ethernet Gateway", which can be found in the documents section at http://www.iccdesigns.com.

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Slot

Defines the destination device slot number in a rack for routing purposes. This field is optional: a setting of "N/A" (Not Applicable) specifies that a slot number entry will not be included during connection establishment. Devices typically require a numeric slot setting for successful connection establishment, but please refer to each server device's documentation for guidance regarding appropriate slot number setting or omission. For example, when targeting an Allen-Bradley –Logix processor, the *Slot* is typically set to 0, as this is commonly where the processor resides in the rack.

Produced Tag

Specifies the produced tag name on the target device. Enter a string from 1 to 40 characters in length. Note that tag names may only contain alphanumeric characters and the underscore character.

Consumed Database Address

Specifies the database address to which the first consumed byte of data is mapped. The configuration studio will not allow entry of a starting database address that will cause the consumed data array to run past the end of the database. The highest valid database address, therefore, will depend on the number of bytes to be consumed.

Consumed Bytes

Specifies the number of bytes (0…509) to consume (receive from the server device).

RPI

Specifies the Requested Packet Interval in units of milliseconds. The minimum supported RPI is 10ms.

Multiplier

Specifies the connection timeout multiplier. The connection's timeout time is determined by multiplying the RPI value by the connection timeout multiplier.

1.6 Manual Trigger

1.6.1 Overview

Every service object is triggered to execute periodically by some form of stimulus. This stimulus may take the form of a packet interval timer (for class 1 connections), or expiration of the driver scan rate cycle timer (for class 3 connections). By default, the driver automatically manages each service object's production/consumption or reading/writing (according to whether the service object uses a class 1 or class 3 connection).

However, there may be situations where external (manual) control of this trigger stimulus is desirable in order to achieve a certain degree of control over a service object's behavior (the conditions in which connections are established or when data is exchanged with the remote device). This can be accomplished by adding an optional Manual Trigger element to the service object. Once added, a manual trigger can act as the service object's execution stimulus by manipulating a single bit in the internal database. Trigger bits can be manipulated either by actively injecting data into the database from a remote client via any supported server protocol, or by new data values being actively read into the database via service objects associated with a client protocol.

1.6.2 Manual Trigger Settings

Trigger Address

Specifies the database address that contains the byte-size *Trigger Bit* structure.

Trigger Bit

Specifies which bit in the byte designated by the *Trigger Address* is to be used as the trigger bit. Only one bit may be selected in the *Trigger Bit* structure, and each bit at a given *Trigger Address* may only be associated with one service object (in other words, there is a unique "one service object to one trigger bit" association). This mechanism allows up to 8 service object trigger bits to be simultaneously assigned to any given database address.

1.6.3 Behavior

While the definition of what constitutes a trigger bit is consistent for all service objects, the triggering mechanism behavior is different for class 1 and class 3 connections:

For class 1 connections

- When auto trigger is used, the class 1 connection will always be enabled. This means that the I/O connection will be persistent and production and/or consumption will continuously occur at the designated RPI interval.
- When manual trigger is used, the class 1 connection will be enabled only when the designated trigger bit is a "1": if the trigger bit becomes a "0", then the client will close the class 1 connection to the server. If the trigger bit transitions back to a value of "1", the client will once again reestablish the class 1 connection to the server and resume regular I/O data production and/or consumption.

For class 3 connections

- When auto trigger is used, cyclic expiration of the connection object's *Scan Rate* timer causes the service object to unconditionally trigger. The service object's specific behavior then depends on the configuration of the "read" and "write" function checkboxes: if reading is enabled, the service object will read from the remote server unless writing is enabled and a pending write exists. If writing is enabled, then when values encompassed by the service object change in the database, these changes will be written down to the targeted server. Regardless, the service object will be guaranteed to either read or write when triggered.
- When manual trigger is used, then only reading or writing may be enabled (not both). This essentially turns the service object into a read-only or write-only service object. In this mode, cyclic expiration of the connection object's *Scan Rate* timer causes the service object

to only evaluate whether or not it should trigger. This evaluation is performed by inspecting the state of the trigger bit and reacting to it as a "one-shot", meaning that when an external source sets the trigger bit to "1", then the corresponding service object request (read or write) is unconditionally generated (regardless of whether or not the service object data has changed). Once the request has been completed (either successfully or unsuccessfully), the driver will update the database with the current data (if the service object is read-only), update the diagnostics object data structure (if implemented) to indicate the success or failure of the transaction, and then automatically clear the trigger bit. In this way, a remote device can both trigger a read/write class 3 action, as well as be notified of the completion and resulting status by appropriately mapping the trigger bit, service object data, and diagnostics object structure into a block of data that is accessible via another network (such as a read/write service object controlled by a client driver on another network). If the service object evaluates its trigger bit to be "0", then it takes no further action.

Note that because the internal database is initialized to "0" values after every boot cycle, all defined manual trigger bits will cause their respective service objects to be disabled until explicitly enabled from an external source.

1.7 Diagnostics Object

Each service object can optionally include a diagnostics object for debugging and diagnostics.

Diagnostics Database Address

Enter the database address at which to store the diagnostics information.

INDUSTRIAL CONTROL COMMUNICATIONS, INC.

1600 Aspen Commons, Suite 210
Middleton, WI USA 53562-4720
Tel: [608] 831-1255   Fax: [608] 831-2045

http://www.iccdesigns.com

Printed in U.S.A