ICCP Client/Server Driver

Filename	ICCP.dll
Manufacturer	Standard IEC 60870-6
Devices	
Protocol	ICCP - TASE.2 (MMS) over TCP/IP Version 2000-08
Version	2.0.31
Last Update	12/10/2024
Platform	Win32
Dependencies	IOKit v2.00
Superblock Readings	No
Level	31300

Introduction

ICCP Client/Server Driver communicates with other ICCP Clients or Servers over Ethernet TCP/IP. Each instance of this Driver allows Clients, Servers, and Client/Server communication Profiles with other Centers.

This Driver, according to which is defined by 60870-6 standard, requires the definition of a Bilateral Table that specifies variables, parameters, and permissions for exchanging data among centers that use TASE.2 protocol. This table's format, its configuration, or maintenance is outside the scope of this standard.

For an easy interchange of this table among different software manufacturers, ICCP Client/Server Driver uses a **CSV** (*Comma-Separated Values*) file format to define a Bilateral Table for Clients or Servers. These files, called ICCPClient.blt on the Client side and ICCPServer.blt on the Server side, have the same internal structure that users must follow.

To create these files, the following possibilities are available:

- If this Driver works as an ICCP Client, users can create a declaration file for an ICCPClient.blt reading point using an Online ICCP Server
- Using a file in Excel format, provided with this Driver, and exporting it to a **CSV** format, and renaming it to ICCPClient.blt or ICCPServer.blt
- Importing an XML file in the **Sisco AXS4** format
- Editing this file manually with an ASCII editor, such as Notepad
- Editing, at run time, this Driver's operation Tags

Client or Server Tags in **E3** or **Elipse Power** can be created automatically in **Offline** mode using the Tag Browser window, if these files were already created.

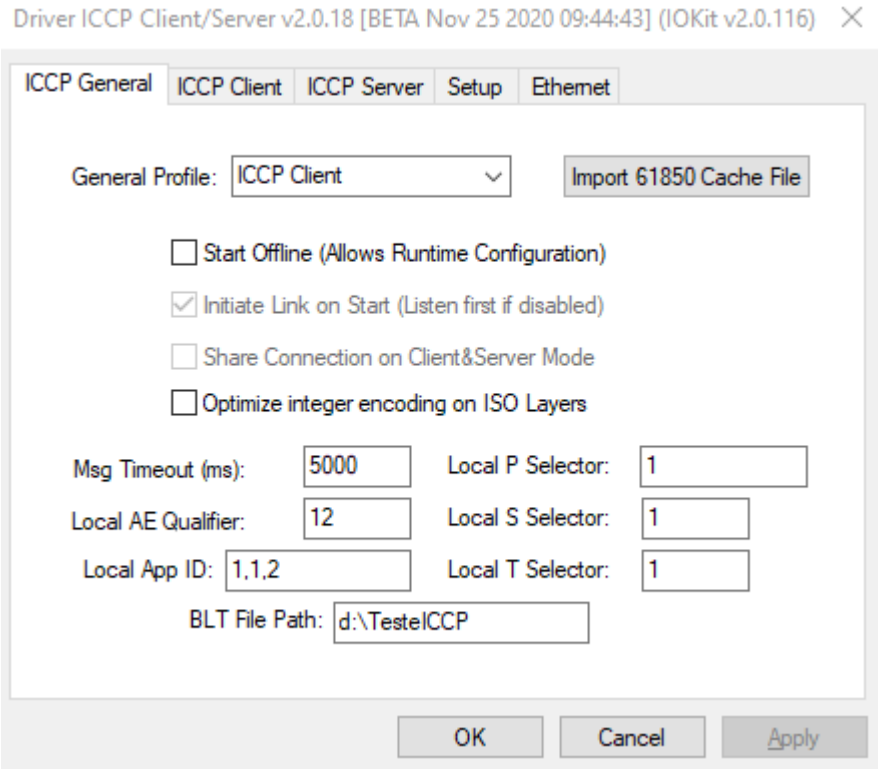
This Driver implements ICCP Blocks 1, 2, 3, and 5 (**Basic Services**, **Extended Conditions**, **Blocked Transfers**, and **SBO Device Control**, respectively).

Configuration

This section contains information about the configuration of ICCP Client/Server Driver. [P] configuration parameters are not used, and all configurations are performed on Driver's configuration window, on **ICCP General**, **ICCP Client**, and **ICCP Server** tabs, described on the next topics.

ICCP General Tab

Tab to configure local and general parameters.



ICCP General tab

The available options on this tab are described on the next table.

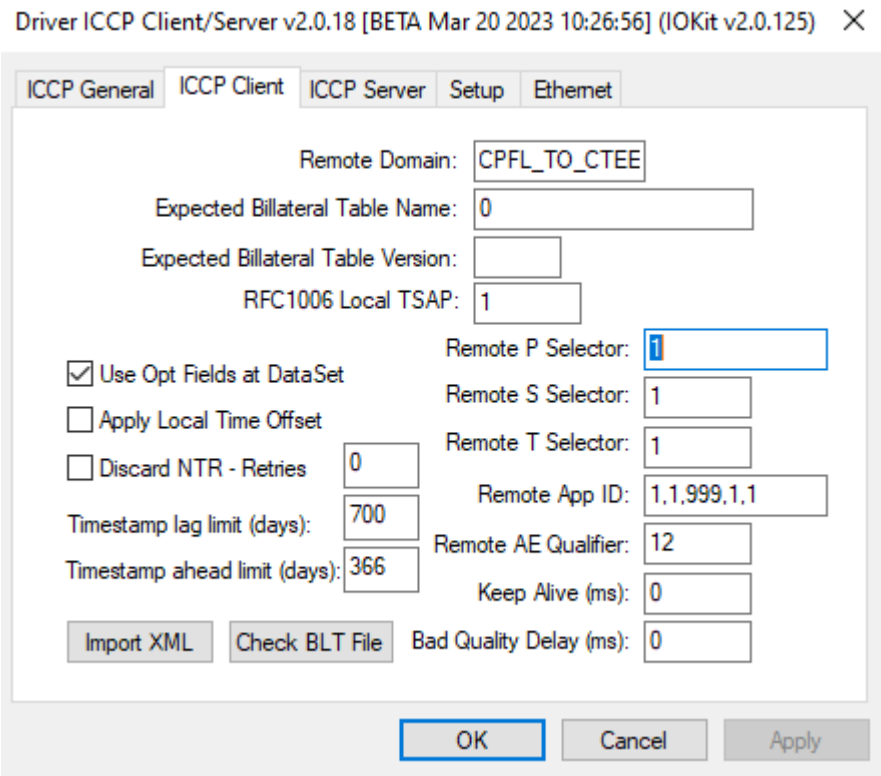
Available options on ICCP General tab

OPTION	DESCRIPTION
General Profile	Informs whether this Driver operates as an ICCP Client (collecting data from an ICCP Server, the **ICCP Client** option), as an ICCP Server (sending data to an ICCP Client, the **ICCP Server** option), or as a Server and Client at the same time (the **ICCP Client/Server** option). In this last case, select the **Initiate Link on Start** option so that this Driver starts the communication, that is, works primarily as an ICCP Client
Import 61850 Cache File	Generates a basic BLT file from **Logical Device Cache** (.ld extension) files generated by **Elipse Software**'s IEC 61850 Driver. Use this option if a local **E3** or **Elipse Power** application is reading data from an IEC 61850 device to provide data as an ICCP Server to other centers
Start Offline (Allows Runtime Configuration)	Configures this Driver to wait for configuration parameters before execution. When using this option, after loaded to memory, this Driver remains in **Idle** status, where an application can configure any parameter defined on its configuration window. This operation is performed using **IOKit**'s **Set Parameter** Tags (please check topic **Documentation of I/O Interfaces** for more information). Each parameter corresponds to a textual identifier, listed on topic **Driver Parameters**, and to a value. After all

OPTION	DESCRIPTION
	parameters are configured, this Driver can be executed by changing the **Start Offline** option to False (0, zero)
Optimize Integer Encoding on ISO Layers	Informs if integer numbers shall be compacted to the minimum possible bytes. If FALSE, the size is kept at the maximum.
Initiate Link on Start (Listen first if disabled)	In a dual Client/Server connection, where both stations are Clients and Servers at the same time, users can define in this option if an application tries to connect to a remote Server immediately after executed or if an application must wait for a remote connection to start the operation on the Client side
Share Connection on Client/Server Mode	When in Client/Server mode, leave this option selected to allow the bidirectional communication to happen over the same TCP/IP connection (socket). If the **Initiate Link on Start** option is also checked, then this Driver executes a TCP/IP connection and starts the ICCP protocol as a Client. Otherwise, it is recommended to configure this Driver to listen on a TCP/IP port, on the **Ethernet** tab, so that the remote partner can connect and start as an ICCP Client
Msg Timeout (ms)	A time-out for an application answer to **Select** and **Operate** commands
Local P Selector	Value of a local selector used by the OSI Presentation Layer (ISO/IEC 8823). Default value of this option is 1 (one)
Local S Selector	Value of a local selector used by the OSI Session Layer (ISO/IEC 8327). Default value of this option is 1 (one)
Local T Selector	Value of a local selector used by the OSI Transport Layer (ISO/IEC 8073). Default value of this option is 1 (one)
Local App ID	Identifier used by the OSI Association Layer (ISO/OSI 8650) in the **ASN.1** (Abstract Syntax Notation 1) format. Indicates a data format adopted by the **AARQ** (Association Request) function, whose default value is "1,1,999,1,1" (iso.1.999.1.1)
Local AE Qualifier	Identifier used by the OSI Association Layer as a format. Default value of this option is 12
BLT File Path	File path of a bilateral table, with a .blt extension

ICCP Client Tab

Tab to configure **Client**-side parameters.



ICCP Client tab

The available option on this tab are described on the next table.

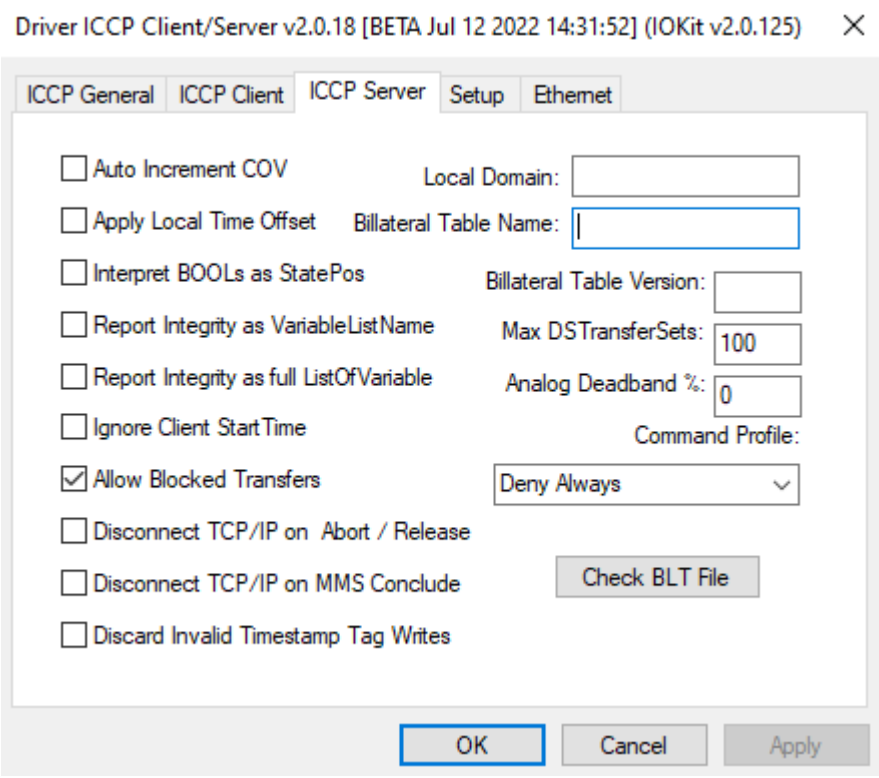
Available options on ICCP Client tab

OPTION	DESCRIPTION
Remote Domain	Informs the Domain name of a remote ICCP Server, that is, an instance name. This name is checked on both sides, Client and Server, for several operations and it must match exactly
Expected Billateral Table Name	Informs the name of an ICCP Server's Bilateral Table. If the other side, the Server, is an **E3** or **Elipse Power** application using this Driver, this name is an alias to definitions of an ICCPServer.blt file and it must be checked by the Client to ensure application's security
Expected Billateral Table Version	Informs the version of an ICCP Server's Bilateral Table. This version must be checked by the Client to ensure application's security
RFC1006 Remote TSAP	This Driver uses RFC 1006 specification as a transport layer for ISO over TCP packages. Therefore, users must inform the **TSAP** (Transport Service Access Point) used by this Driver on the remote side, the Server, to establish a connection using this protocol. Default value of this option is 1 (one)
Use Opt Fields at DataSet	IEC 60870-6 version 2000-08 standard defines optional fields that can be declared and sent with a **DataSet TransferSet** message. These fields help identifying message parameters, such as timestamp and purpose. Default value of this option is selected
Apply Local Time Offset	Considers the timestamp provided by the ICCP Server in **UTC** format, applying the local time offset
Discard NTR - Retries	Informs if the driver shall discard read responses from the server which contain null timestamps (NTR = Null Timestamp Responses). If in a Dataset read which contains datatypes with timestamps, all of them are null, all values will be discarded and a new read request will be sent, until the maximum count defined at the Retries property.

OPTION	DESCRIPTION
Timestamp lag limit (days)	Inform the max number of days that a timestamp can be in the past compared to current time. If this limit is exceeded, the timestamp will be replaced by current time and tag quality will be 64 (Uncertain).
Timestamp ahead limit (days)	Inform the max number of days that a timestamp can be in the future compared to current time. If this limit is exceeded, the timestamp will be replaced by current time and tag quality will be 64 (Uncertain).
Import XML	Creates an ICCPClient.blt file by importing an XML file in **Sisco AXS4** format
Check BLT File	Checks consistency of an ICCPClient.blt file and displays a dialog box with the results of this check
Remote P Selector	Value of a remote selector used by the OSI Presentation Layer (ISO/IEC 8823). Default value of this option is 1 (one)
Remote S Selector	Value of a remote selector used by the OSI Session Layer (ISO/IEC 8327). Default value of this option is 1 (one)
Remote T Selector	Value of a remote selector used by the OSI Transport Layer (ISO/IEC 8073). Default value of this option is 1 (one)
Remote App ID	Identifier used by the OSI Association Layer (ISO/OSI 8650) in **ASN.1** (Abstract Syntax Notation 1) format. Indicates a data format adopted by the **AARQ** (Association Request) function, whose default value is "1,1,999,1,1" (iso.1.999.1.1)
Remote AE Qualifier	Identifier used by the OSI Association Layer as a format. Default value of this option is 12
Keep Alive (ms)	Time Interval used by a Client to check if the connection is still alive, by sending an **MMS Identify** message, if no message was sent during the specified period of time
Bad Quality Delay (ms)	Delay to be considered to return bad quality to a point, after the reception of a value with bad quality from server. If this parameter is greater than zero (0), the received value will be reported immediately, but with good quality. When the delay time is expired, if the point still has bad quality, then it will be updated accordingly. If before the delay time is expired, a value with good quality is received, then the new value is reported immediately and the delay count stops.

ICCP Server Tab

Tab to configure **Server**-side parameters.



ICCP Server tab

The available options on this tab are described on the next table.

Available options on ICCP Server tab

OPTION	DESCRIPTION
Local Domain	Informs the Domain name of a local ICCP Server, that is, an instance name. This name is checked on both sides, Client and Server, for several operations and it must match exactly
Billateral Table Name	Informs the name of an ICCP Server's Bilateral Table. This name is an alias to definitions of an ICCPServer.blt file and it must be checked by the Client to ensure application's security
Billateral Table Version	Informs the version of an ICCP Server's Bilateral Table. This version must be checked by the Client to ensure application's security
Max DataSet TransferSets	For each **DataSet** to transfer, the Server must create a **TransferSet** instance, by Client's request, which is linked to this **DataSet**. This option indicates the maximum number of **TransferSets** that can be created by this Driver. Default value of this option is 100
Analog Deadband %	Informs a general analog dead band to apply to data when written by an application. Only values that exceed this dead band are copied to current values considered as events
Auto Increment COV	If this option is selected, for data types with the **COV** (Counter of Variations) property, each new value that exceeds the dead band verification forces an auto-increment of the **COV** property

OPTION	DESCRIPTION
Apply Local Time Offset	Considers the timestamp provided by the ICCP Server in **UTC** format, applying the local time offset
Interpret BOOLs as StatePos	**StateXXX** datatypes (**StateQ** and **StateQTimeTag**, among others) uses two bits to identify breakers or switch positions (**Open**, **Closed**, or **Invalid**). By using this option, users can map a Boolean type, zero or one, to the **Open** and **Closed** statuses. The value 0 (zero) maps to 1 (one, **StateOff**) and the value 1 (one) maps to 2 (two, **StateOn**)
Report Integrity VariableListName	When sending report messages, this option informs whether variable names must be informed as "VariableListName" if selected and as "ListOfVariable" otherwise. Default value in previous versions is deselected, that is "ListOfVariable"
Report Integrity as full ListOfVariable	When sending report messages, this option indicates whether users must inform the name of all items. This option turns a report less efficient, because it shrinks space for data, and it must be used only for compatibility with clients not correctly implemented
Ignore Client StartTime	When a Client sends a **Start DataSet TransferSet** request and the time interval is wrong, use this option to ignore the informed time, so that the server starts processing and transmitting data immediately
Allow Blocked Transfers	Informs if at the read response of the reserved item "Supported_Features" the Block 3 - Blocked Transfers is present.
Disconnect TCP/IP on Abort/ Release	When a Client sends a Session's **Abort** or Association's **Release** command without closing the TCP/IP channel, leave this option selected so that the Server forces a disconnection
Disconnect TCP/IP on MMS Conclude	When a client sends a MMS **Conclude** without closing the TCP/IP channel, leave this option selected so that the Server forces a disconnection
Discard Invalid Timestamp Tag Writes	Upon receiving a tag write with a new value, the driver as a Server uses the parameters Timestamp Lag Limit Days e Timestamp Ahead Limit Days defined at the client tab to check if timestamp is valid. If this option is checked, the write operation with invalid timestamp is cancelled. Otherwise, the value is accepted, but the timestamp is replaced by current time and quality is considered as uncertain (64).
Check BLT File	Checks the consistency of an ICCPServer.blt file and displays a dialog box with the results of this check

Bilateral Table

Bilateral Tables ICCPClient.blt and ICCPServer.blt must follow some rules for their processing by an ICCP Client/Server Driver. These files must be in **ASCII** format, generated by **one of the options described at the beginning of this document** and contain the sections described on the next topics.

Format

NOTE

To declare a line as a comment, use the sequence of character "\\\\" at the beginning of that line.

```
ICCPVersionYear-ICCPVersionMonth;BLTFileFormatVersion;BLTName
**DataSets**
DataSetCount
DataSetDeclaration (repeats the number of times indicated by DataSetCount)
**DataValues**
DataValueCount
DataValueDeclaration (repeats the number of times indicated by DataValueCount)
**Devices**
DeviceCount
DeviceDeclaration (repeats the number of times indicated by DeviceCount)
**Messages**
MessageCount
MessageDeclaration (repeats the number of times indicated by MessageCount)
```

Available options in Bilateral Table format

OPTION	DESCRIPTION
ICCPVersionYear	Must be equal to "2000"
ICCPVersionMonth	Must be equal to "08"
BLTFileFormatVersion	Must be equal to "2"
BLTName	Textual name of a Bilateral Table
****DataSets****	**DataSet** identifier
DataSetCount	Total number of **DataSets** that follow
DataSetDeclaration	A series of parameters separated by a colon. **DataSetName**: Unique name of a **DataSet**; **Delay**: A delay, in seconds, so that a Server starts monitoring points of this **DataSet** after starting execution; **Interval**: Time interval among Server reports, in seconds, used by the **IntervalTimeout** option in the **DSConditionsRequested** parameter; **TLE**: Time limit for execution, in seconds; **BufferTime**: Time interval to keep in memory changing events in objects before sending a report, in seconds; **IntegrityCheck**: Time value for an integrity check when the **IntegrityTimeout** condition is used, in seconds. It is only significant when the **RBE** parameter is equal to 1 (one); **DSConditionsRequested**: A byte whose five most significant bits represent a bit mask with instructions for monitoring an ICCP Server (**Bit 7**: **IntervalTimeout**. Periodic sending, **Bit 6**: **IntegrityTimeout**. Valid only if the **RBE** option is selected, **Bit 5**: **ObjectChange**, **Bit 4**: **OperatorRequest**, **Bit 3**: **OtherExternalEvent**, **Bit 2**: Not used, **Bit 1**: Not used, and **Bit 0**: Not used); **BlockData**: Enables a message compression if configured to 1 (one) or disables compression if configured to 0 (zero); **Critical**: Asks for a confirmation from ICCP Clients for each report if configured to 1 (one) and does not require confirmation if configured to 0 (zero); **RBE**: Enables a report by the exception mechanism if configured to 1 (one) or disables a report by the exception mechanism if configured to 0

OPTION	DESCRIPTION
	(zero); **ReportAllException**: If configured to 1 (one), informs a Server to report all changes on the monitored points. If configured to 0 (zero), informs a Server to report only the value of the last monitored point when the report was generated
****DataValues****	**DataValue** identifier
DataValueCount	Total number of **DataValues** that follow
DataValueDeclaration	A series of parameters separated by a colon. **DataSetIndex**: **DataSet** index (previous section) that reports this **DataValue**, starting at 1 (one). Configure as 0 (zero) if there is no **DataValues** to report; **TagName**: Unique name of this **DataValue**; **DataType**: Select one of the pre-defined ICCP data types: **Discrete**, **DiscreteQ**, **DiscreteQTimeTag**, **DiscreteQTimeTagExtended**, **DiscreteExtended**, **Real**, **RealQ**, **RealQTimeTag**, **RealQTimeTagExtended**, **RealExtended**, **State**, **StateQ**, **StateQTimeTag**, **StateQTimeTagExtended**, or **StateExtended**; **Scope**: Local (ICC, 1) or Global (VCC, 0); **SourceType**: 0 (Telemetered), 1 (Calculated), 2 (Entered), or 3 (Estimated); **AllowWrite**: Yes (1) or No (0); and **Description**: Optional description
****Devices****	Identifier of devices
DeviceCount	Total number of devices that follow
DeviceDeclaration	A series of parameters separated by a colon. **DataSetIndex**: **DataSet** index; **Name**: Name of this device; **CmdType**: CMD (C), SP_DISCRETE (D), or SP_REAL (R); **Scope**: Local (ICC, 1) or Global (VCC, 0); **TAGGABLE**: Yes (1) or No (0); **SBO**: Yes (1) or No (0); **CheckBackID**: Number expected to be returned by a Server when this device is selected and only used when **SBO** is equal to 1 (one); and **Description**: Optional description
****Messages****	Identifier of messages
MessageCount	Total number of messages that follow
MessageDeclaration	A series of parameters separated by a colon. **InfoReference**: Name of a message slot; **Size**: Maximum size; and **Description**: Optional description

Example File

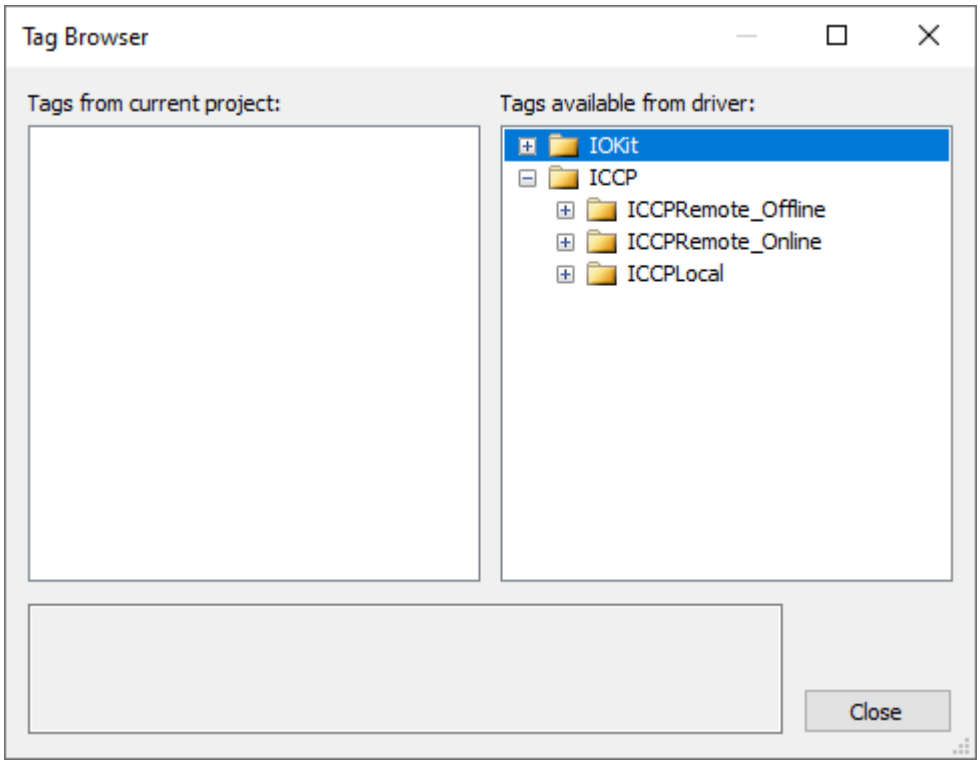
```
2000-08;2;Example
\\\\ DataSet Declaration
\\\\ **Identifier**
\\\\ Number of Items
\\\\ DataSetName, Delay, Interval,
\\\\ TLE, BufferTime, IntegrityCheck,
\\\\ DSConditionsRequested, BlockData, Critical,
\\\\ RBE, ReportAllException
**DataSets**
1
DS1;0;1;0;1;60;96;1;1;1;1
\\\\ DataValue Declaration
\\\\ **Identifier**
\\\\ Number of Items
\\\\ DataSetIndex, TagName, DataType,
\\\\ Scope:ICC=1 VCC=0,
\\\\ SourceType:TEL=0 CAL=1 ENT=2 EST=3,
\\\\ AllowWrite:YES=1 NO=0, Description
**DataValues**
10
1;CMMXU1$MX$A$phsA$cVal$mag$f;RealQTimeTagExtended;0;0;0;
1;CMMXU1$MX$A$phsA$range;RealQTimeTagExtended;0;0;0;
1;CMMXU1$MX$A$phsB$cVal$mag$f;RealQTimeTagExtended;0;0;0;
1;CMMXU1$MX$A$phsB$range;RealQTimeTagExtended;0;0;0;
1;CMMXU1$MX$A$phsC$cVal$mag$f;RealQTimeTagExtended;0;0;0;
1;CMMXU1$MX$A$phsC$range;RealQTimeTagExtended;0;0;0;
1;CVMMXU1$MX$Amp$mag$f;RealQTimeTagExtended;0;0;0;
1;CVMMXU1$MX$Amp$range;RealQTimeTagExtended;0;0;0;
1;CVMMXU1$MX$Hz$mag$f;RealQTimeTagExtended;0;0;0;
1;CVMMXU1$MX$Hz$range;RealQTimeTagExtended;0;0;0;
\\\\ Command Declaration
\\\\ **Identifier**
\\\\ Number of Items
\\\\ DataSetIndex, Name,
\\\\ CmdType: NONE=N CMD=C SP_DISCRETE=D SP_REAL=R,
\\\\ Scope:ICC=1 VCC=0, TAGGABLE: YES=1 NO=0,
\\\\ SBO: YES=1 NO=0, CheckBackID, Description
**Devices**
2
0;CMMXU1$CO;C;0;0;0;0;
0;CMSQI1$CO;C;0;0;0;0;
\\\\ Message Declaration
\\\\ **Identifier**
\\\\ Number of Items
\\\\ InfoReference, Size, Description
**Messages**
0
```

Important Notes

- An ICCP Server uses a **MaxPDU** (Maximum Number of Bytes in a Single MMS Data Unit) size equal to 32000 bytes. It is Client's responsibility to restrict the size for each **DataSet** to fit into a single PDU. If this size is exceeded, a new **DataSet** must be created.
- An ICCP Server only checks the total number of messages with events (**ReportAllChanges**) in **Block** mode. Clients must preferably use the **Block** mode to avoid messages with a size greater than the expected by the Client.
- Each Tag must appear on a single **DataSet**, but several **DataSets** are allowed.

Tag Browser

Click **Tag Browser** on a Driver's toolbar in **E3** or **Elipse Power** to open Tag Browser window, displayed on the next figure. This window allows dragging and dropping Tags found on Client and Server configurations.



Tag Browser

The **Tags available from driver** list contains the items described next.

- **IOKit**: Contains **IOKit**'s default Tags to read and write general parameters and statuses, among others. These Tags **CANNOT** be used by this Driver because host's default connection is not used. This Driver uses two internal instances of a host, one for the Client and another one for the Server
- **ICCP**: Contains the items **ICCPRemote_Offline**, which displays Tags read from an ICCPClient.blt file representing a remote ICCP Server, **ICCPRemote_Online**, which reads declarations of points from a remote ICCP Server, creates an ICCPCLient.blt file, and displays these Tags (a TCP/IP connection correctly configured must be available to read these points), and **ICCPLocal_Offline**, which displays Tags read from an ICCPServer.blt file representing a local ICCP Server

Tag Reference

This Driver only uses the *Item* and *N1* parameters of Tags. If a Tag references a Server side, the *N1* parameter must be equal to 0 (zero). If a Tag references a Client side, the *N1* parameter must be equal to 1 (one).

Server Side Tags

ITEM	OPERATION	N1	MEANING
PointName	Writing	0 (zero)	Receives the current value, timestamp, and quality of a point from an application. The **PointName** Tag must be declared in the ICCPServer.blt file
PointName:Source	Writing	0 (zero)	Changes the default source of a Point. Possible values are **0**: Telemetry, **1**: Calculated, **2**: Entered, or **3**: Estimated
DeviceName:Select	Reading and Writing	0 (zero)	Indicates whether a device is selected on the Client side, by using the value "1". An

ITEM	OPERATION	N1	MEANING
			application must write back this value "1" to confirm that it accepted the selected operation or "0" to reject this operation. This operation must be completed within the maximum time informed in the **Msg Timeout (ms)** option on the **ICCP General** tab
DeviceName:Operate	Reading and Writing	0 (zero)	Indicates whether a device is in operation on the Client side. This Tag's value is sent by the Client side. An application must send the same value back to confirm a successful operation or any other value to reject this operation. This operation must be completed within the maximum time informed in the **Msg Timeout (ms)** option on the **ICCP General** tab
DeviceName:Tag	Reading and Writing	0 (zero)	A Block with two Elements containing device's tags. Element 0 (zero) indicates the device's status and it is a bit mask, where **Bit 0**: 1 (one, Armed) or 0 (zero, Not Armed); **Bits 1 and 2**: 0 (zero, No Tag), 1 (one, CloseOnlyInhibit), or 2 (two, OpenAndCloseInhibit). Element 1 (one) indicates the reason for tagging, in **Text** format. **NOTE**: If a device is tagged with an **Armed** status, this device is set back to its **Idle** status after the time-out of the **Select** command
ServerStatus	Reading	0 (zero)	Returns the internal status on the ICCP Server side. Possible values are **0**: Starting, **1**: Reading .ld files, **2**: Retrieving Server's directory (**GetServerDirectory**), or **3**: Processing an ld file (Tag Browsing)
ServerConnection	Reading	0 (zero)	Indicates the connection status. Possible values are **1**:

ITEM	OPERATION	N1	MEANING
			Connected or **0**: Disconnected
UpdatePointDatabase	Writing	0 (zero)	Inserts or updates a point on Server's database. Receives a Block with two, three, or four Elements. Values for Elements are **0**: Point's name; **1**: Point's type, between zero and 15 (**0**: Discrete, **1**: DiscreteQ, **2**: DiscreteQTimeTag, **3**: DiscreteQTimeTagExtended, **4**: DiscreteExtended, **5**: Real, **6**: RealQ, **7**: RealQTimeTag, **8**: RealQTimeTagExtended, **9**: RealExtended, **10**: State, **11**: StateQ, **12**: StateQTimeTag, **13**: StateQTimeTagExtended, **14**: StateExtended, or **15**: Tag_Value); **2**: Scope (**0**: VCC or **1**: ICC); and **3**: Source (**0**: Telemetry, **1**: Calculated, **2**: Entered, or **3**: Estimated). Default value is 0 (zero)
UpdateDeviceDatabase	Writing	0 (zero)	Inserts a device on Server's database. It must receive a Block containing between two and six Elements. Values for Elements are **0**: Device's name as a **String**; **1**: Device's type as a **String** (**N**: None, **C**: CMD, **D**: SP_DISCRETE, or **R**: SP_REAL); **2**: Scope, optional (**0**: VCC or **1**: ICC); **3**: Tag, optional (**0**: No or **1**: Yes); **4**: SBO, optional (**0**: No or **1**: Yes); **5**: CheckBackID, optional; and **6**: Description, an optional **String**
DeleteDatabase	Writing	0 (zero)	Deletes all definitions from a Bilateral Table (BLT) file. It must receive a **String** with the name of a BLT file. This file's name and path are not modified. Users must write to this Tag only when this Driver starts in **Offline** mode, allowing to delete the current database, redefine its content, and finally use the **SaveDatabase** Tag
SaveDatabase	Writing	0 (zero)	Saves the Bilateral Table (BLT) file. It must receive a **String** with the name of the

ITEM	OPERATION	N1	MEANING
			BLT file. This file's name and path are not modified
UpdatePoint	Writing	0 (zero)	Updates a point in memory. It must receive a Block with two or three Elements. Values for Elements are **0**: Point's name; **1**: New value; and **2**: Source (**0**: Telemetry, default, **1**: Calculated, **2**: Entered, or **3**: Estimated). **NOTE**: This Tag represents the same writing operation as the **PointName** Tag, except that it allows source's information on the same operation
SelectConfirm	Writing	0 (zero)	Receives a confirmation from an application that a device was selected. It must receive a Block with two Elements. Values for Elements are **0**: Device's name; and **1**: 1 (one, a fixed numerical value). **NOTE**: This Tag represents the same writing operation as the **DeviceName:Select** Tag
OperateConfirm	Writing	0 (zero)	Receives a confirmation from an application that a device was operated. It must receive a Block with two Elements. Values for Elements are **0**: Device's name and **1**: The same value received to confirm or a different value to reject. **NOTE**: This Tag represents the same writing operation as the **DeviceName:Operate** Tag
Tagging	Writing	0 (zero)	Inserts a tag in a device, similar to the writing operation **DeviceName:Tag**. It must receive a Block with three Elements. Values for Elements are **0**: Device's name; **1**: Device's status as a bit mask whose bits 0 (zero) and 1 (one) indicate the Tag (**No Tag**, **OpenCloseInhibit**, or **CloseOnlyInhibit**) and bit 2 (two) indicates the status (**Armed** or **Idle**); and **2**: Reason for tagging in **Text** format. **NOTE**: If a device is

ITEM	OPERATION	N1	MEANING
			tagged with the **Armed** status, this device is set back to its **Idle** status

Client Side Tags

ITEM	OPERATION	N1	MEANING
PointName	Reading and Writing	1 (one)	Reads or writes the current value, timestamp, and quality of a point. The **PointName** Tag must be declared in the ICCPClient.blt file
PointName:COV	Reading	1 (one)	Reads the **COV** (Counter of Variations) property of a point, if available
PointName:Source	Reading	1 (one)	Reads the point's current source. Possible values are **0**: Telemetry, **1**: Calculated, **2**: Entered, or **3**: Estimated
GetDataValue:PointName	Reading	1 (one)	Operation which allows the validation of a tag at the server database, returning the current value if it exists. This tag shall not be used as the main way to get a tag value, because the read process is poll based and may loose events between each read.
DeviceName	Reading	1 (one)	Reads a device's status. A bit mask where Bit 0 (zero) can be **1**: Armed or **0**: Not Armed and Bits 1 (one) and 2 (two) can be **0**: No Tag, **1**: CloseOnlyInhibit, or **2**: OpenAndCloseInhibit
DeviceName:Select	Writing	1 (one)	Requests a **Select** command in a device. This request may fail if the point is not an **SBO** type or if any security checks performed by the ICCP Server fails. If the device is declared on a **DataSet**, the linked Tag with device's status (previous item) shows the configured bit 0 (zero), indicating that this device is in the **Armed** status, if operation is successful. This Tag can contain any value
DeviceName:Operate	Writing	1 (one)	Requests an **Operate** command in a device. This request may fail if the point

ITEM	OPERATION	N1	MEANING
			is not an **SBO** type and is not in the **Armed** status, or if any security checks performed by the ICCP Server fails. This Tag must contain the value to send to the ICCP Server. IEC 60870-6-802 standard defines values "0" (zero) and "1" (one) for operations with digital commands, where **0**: Trip, Open, Off, or Lower (for Transformers) and **1**: Close, On, or Raise (for Transformers)
DeviceName:Tag	Reading and Writing	1 (one)	A Block with two Elements containing device's Tags. Element 0 (zero) indicates device's status as a bit mask, where bit 0 (zero) can be **1**: Armed or **0**: Not Armed and bits 1 (one) and 2 (two) can be **0**: No Tag, **1**: CloseOnlyInhibit, or **2**: OpenAndCloseInhibit. Element 1 (one) indicates the reason for tagging in **Text** format
DataSetName:GI	Writing	1 (one)	Forces an integrity request for the informed **DataSet**, with the **DSConditions** parameter configured to **OperatorRequest**
ClientStatus	Reading	1 (one)	Returns the internal status on the ICCP Client side. Possible values are **0**: Starting, **1**: Reading .ld files, 2: Retrieving directory (**GetServerDirectory**), or **3**: Processing an ld file (Tag Browsing)
ClientConnection	Reading	1 (one)	Indicates the status of a connection. Possible values are **1**: Connected or **0**: Not connected
StartBatchDatabaseUpdate	Writing	1 (one)	Starts a process of batch update to change Client's Bilateral Table. This step is mandatory before using other Tags for edition at run time, such as **UpdateDatasetDatabase**, **UpdatePointDatabase**, and **UpdateDeviceDatabase**

ITEM	OPERATION	N1	MEANING
UpdateDatasetDatabase	Writing	1 (one)	Inserts or updates a **DataSet** on a Client's database. It must receive a Block with 11 Elements that follows the same order described on the **Dataset** section of the Bilateral Table file. Values for Elements are **0**: DataSetName, **1**: Delay, **2**: Interval, **3**: TLE, **4**: BufferTime, **5**: IntegrityCheck, **6**: DSConditionsRequested, **7**: BlockData, **8**: Critical, **9**: RBE, and **10**: ReportAllException
UpdatePointDatabase	Writing	1 (one)	Inserts or updates a point on a Client's database. It must receive a Block with two, three, four, or five Elements. Values for Elements are **0**: Point's name; **1**: Point's type, between zero and 15 (**0**: Discrete, **1**: DiscreteQ, **2**: DiscreteQTimeTag, **3**: DiscreteQTimeTagExtended, **4**: DiscreteExtended, **5**: Real, **6**: RealQ, **7**: RealQTimeTag, **8**: RealQTimeTagExtended, **9**: RealExtended, **10**: State, **11**: StateQ, **12**: StateQTimeTag, **13**: StateQTimeTagExtended, **14**: StateExtended, or **15**: Tag_Value); **2**: Scope (**0**: VCC or **1**: ICC, default); **3**: Source (**0**: Telemetry, default, **1**: Calculated, **2**: Entered, or **3**: Estimated); and **4**: DataSet index (zero if it is not linked to a **DataSet**)
UpdateDeviceDatabase	Writing	1 (one)	Inserts a device on a Client's database. It must receive a Block with a minimum of two and a maximum of six Elements. Values for Elements are **0**: Device's name, an on-demand **String**; **1**: Device's type, an on-demand **String**. Possible values are **N**: None, **C**: CMD, **D**: SP_DISCRETE, or **R**: SP_REAL; **2**: Scope, optional (**0**: VCC or **1**: ICC); **3**: Tagging, optional (**0**: No or **1**: Yes); **4**: SBO, optional (**0**: No or **1**: Yes); **5**:

ITEM	OPERATION	N1	MEANING
			CheckBackID, optional; and **6**: Description, an optional **String**
DeleteDatabase	Writing	1 (one)	Deletes all definitions from a Bilateral Table (BLT) file. It must receive a **String** with the name of the BLT file. This file's name and path are not modified. This Tag must be written only when this Driver starts in **Offline** mode, allowing to delete the current database, redefine its content, and finally use the **SaveDatabase** Tag
SaveDatabase	Writing	1 (one)	Saves a Bilateral Table (BLT) file. It must receive a **String** with the name of the BLT file. This file's name and path are not modified

Runtime Configuration

This topic contains information on how to change a Driver's configuration and members of Bilateral Tables at run time.

Driver Parameters

If the **Start Offline (Allows Runtime Configuration)** option on **ICCP General** tab is selected, users can change Driver's parameters at run time. Each option on this tab has a corresponding identifier that can be used to inform a new value, using IOKit's special Tag **Write Parameter**, defined by the address **N1.N2.N3.N4 = -1, 0, 0, 3** and that must receive an array with the identifier and its value.

Properties and their identifiers

PROPERTY	IDENTIFIER
ICCP_LOCAL_PSEL	ICCP.LocalPSel
ICCP_LOCAL_SSEL	ICCP.LocalSSel
ICCP_LOCAL_TSEL	ICCP.LocalTSel
ICCP_LOCAL_AE_QUALIFIER	ICCP.LocalAEQualifier
ICCP_LOCAL_APP_ID	ICCP.LocalAppID
ICCP_LOCAL_TSAP	ICCP.LocalTSAP
ICCP_REM_PSEL	ICCP.RemPSel
ICCP_REM_SSEL	ICCP.RemSSel
ICCP_REM_TSEL	ICCP.RemTSel
ICCP_REM_AE_QUALIFIER	ICCP.RemAEQualifier
ICCP_REM_APP_ID	ICCP.RemAppID
ICCP_REM_TSAP	ICCP.RemTSAP
ICCP_CBB	ICCP.CBB

PROPERTY	IDENTIFIER
ICCP_CLIENT_SERVICES	ICCP.ClientServices
ICCP_SERVER_SERVICES	ICCP.ServerServices
ICCP_TIMEOUT	ICCP.Timeout
ICCP_BUFFER_TIME	ICCP.BufferTime
ICCP_INTEGRITY	ICCP.Integrity
ICCP_RPT_INTERVAL	ICCP.RptInterval
ICCP_RBE	ICCP.RBE
ICCP_REBUILDDS	ICCP.RebuildDS
ICCP_DISCARDDS	ICCP.DiscardDS
ICCP_PROFILE	ICCP.Profile
ICCP_REMOTE_DOMAIN	ICCP.RemoteDomain
ICCP_LOCAL_DOMAIN	ICCP.LocalDomain
ICCP_BLTFILEPATH	ICCP.BltFilePath
ICCP_AUTO_INCR_COV	ICCP.AutoIncrCOV
ICCP_MAX_DSTS	ICCP.MaxDSTS
ICCP_SELECT_TOUT	ICCP.SelectTout
ICCP_BLTNAME	ICCP.BLTName
ICCP_BLTVERSION	ICCP.BLTVersion
ICCP_USEOPTFLDS	ICCP.UseOptFlds
ICCP_INITLINK	ICCP.InitLink
ICCP_EXP_BLTNAME	ICCP.ExpBLTName
ICCP_EXP_BLTVERSION	ICCP.ExpBLTVersion
ICCP_ANALOG_DEADBAND	ICCP.AnalogDeadBand
ICCP_START_OFFLINE	ICCP.StartOffline
ICCP_CLIENT_OFFLINE	ICCP.ClientOffline
ICCP_SERVER_OFFLINE	ICCP.ServerOffline

Example of a Driver configuration.

```
Write -1, 0, 0, 3, Array("ICCP.ClientOffline", 1)
```

Server Parameters

Users can change Bilateral Tables of ICCP Servers at run time to add or modify Tags. However, if users want to remove Tags, the Server must be set to **Offline** mode.

Adding or Changing Tags

Use calls to **UpdatePointDatabase** and **UpdateDeviceDatabase** Tags. If these Tags already exist, these parameters are updated. If these Tags do not exist, a new point is then created. At the end of this process, execute a call to the **SaveDatabase** Tag.

Removing, Adding, or Changing Tags

If a removal operation is needed, a sequence of steps must be performed, according to the options described next.

Option 1: Driver Starts in Offline Mode

1. Select the **Start Offline (Allows Runtime Configuration)** option.
2. Set the ICCP Server to **Offline** mode by writing the value "1" to the **ICCP.ServerOffline** Tag, according to the next example.

```
Write -1, 0, 0, 3, Array("ICCP.ServerOffline", 1)
```

3. Configure the **ICCP.StartOffline** Tag with the value "0". This Driver creates the Server component, because it is now in **Online** mode. However, this Server component is still in **Offline** mode, due to the previous step, according to the next example.

```
Write -1, 0, 0, 3, Array("ICCP.StartOffline", 0)
```

4. Execute calls to **UpdatePointDatabase** and **UpdateDeviceDatabase** Tags. As the database is empty, all items must be inserted.
5. Execute a call to the **SaveDatabase** Tag.
6. Set the ICCP Server back to **Online** mode by writing the value "0" to the **ICCP.ServerOffline** Tag, according to the next example.

```
Write -1, 0, 0, 3, Array("ICCP.ServerOffline", 0)
```

Option 2: Driver Starts in Online Mode

1. Deselect the **Start Offline (Allows Runtime Configuration)** option. This configuration disconnects all communication channels, both Clients and Servers, according to the next example.

```
Write -1, 0, 0, 3, Array("ICCP.StartOffline", 1)
```

2. Repeat the next steps of the procedure to **start in Offline mode**.

Client Parameters

Users can change Bilateral Tables of ICCP Clients, at run time, to add or change Tags, for example, to avoid stopping and restarting a Driver to apply these changes.

On the **Client** side, this operation can be performed while the Client is in **Offline** mode or during its normal communication.

During communication, users can change **DataSet** parameters, add, change, and remove **DataPoints** and devices. In **Offline** mode, users can create a completely new Bilateral Table.

Changes in Offline Mode

This operation can be performed when starting this Driver, by configuring the **Start Offline (Allows Runtime Configuration)** option. In this case, execute the steps described next.

1. Select the **Start Offline (Allows Runtime Configuration)** option.

2. Set the ICCP Client to **Offline** mode by writing the value "1" to the **ICCP.ClientOffline** Tag, according to the next example.

```
Write -1, 0, 0, 3, Array("ICCP.ClientOffline", 1)
```

3. Configure the **ICCP.StartOffline** Tag with the value "0". This Driver creates the Client component, because it is now in **Online** mode. However, this Client component is still in **Offline** mode, due to the previous step, according to the next example.

```
Write -1, 0, 0, 3, Array("ICCP.StartOffline", 0)
```

4. Write to the **StartBatchDatabaseUpdate** Tag. In the next example, this Tag was created dynamically before the call.

```
Set StartBatch = AddObject("IOTag", False, "StartBatch")
StartBatch.ParamItem = "StartBatchDatabaseUpdate"
StartBatch.N1 = 1
StartBatch.Activate
StartBatch.WriteEx 1
```

5. Add **DataSets**, **DataPoints**, and devices by using the **UpdateDatasetDatabase**, **UpdatePointDatabase**, and **UpdateDeviceDatabase** Tags. A single call must be performed for each component to create, according to the next example.

```
'In this case, there is a folder called SYS
'with a Block Tag with five Elements
'named UpdatePoint.
Set UpdtPoint = Item("SYS").Item("UpdatePoint")
UpdtPoint.WriteEx Array("TAGNAME_AAA",13,0,0,1)
UpdtPoint.WriteEx Array("TAGNAME_BBB",13,0,0,1)
UpdtPoint.WriteEx Array("TAGNAME_CCC",13,0,0,1)
```

6. Execute a call to the **SaveDatabase** Tag.

```
Set SaveDB = Item("SYS").Item("SaveDatabase")
SaveDB.WriteEx "BLT_Name"
```

7. Set the Client back to **Online** mode.

```
Write -1, 0, 0, 3, Array("ICCP.ClientOffline", 0)
```

Changes at Run Time

1. Write to the **StartBatchDatabaseUpdate** Tag. In the next example, this Tag was created dynamically before the call.

```
Set StartBatch = AddObject("IOTag", False, "StartBatch")
StartBatch.ParamItem = "StartBatchDatabaseUpdate"
StartBatch.N1 = 1
StartBatch.Activate
StartBatch.WriteEx 1
```

2. Inform all **DataSets**, **DataPoints**, and devices by using the **UpdateDatasetDatabase**, **UpdatePointDatabase**, and **UpdateDeviceDatabase** Tags. A single call must be performed for each component to create, according to the next example.

```
'In this case, there is a folder called SYS
'with a Block Tag with five Elements
'named UpdatePoint.
Set UpdtPoint = Item("SYS").Item("UpdatePoint")
UpdtPoint.WriteEx Array("TAGNAME_AAA",13,0,0,1)
UpdtPoint.WriteEx Array("TAGNAME_BBB",13,0,0,1)
UpdtPoint.WriteEx Array("TAGNAME_CCC",13,0,0,1)
```

3. Execute a call to the **SaveDatabase** Tag.

```
Set SaveDB = Item("SYS").Item("SaveDatabase")
SaveDB.WriteEx "BLT_Name"
```

4. When executing a call to the **SaveDatabase** Tag, the current Bilateral Table is compared to this new Table, with the following rules:
   - If a match is found on both tables, the new table is copied over the existing one
   - If an entry is found on the existing table and not on the new table, the existing entry is deleted
   - If an entry is found on the new table and not on the existing table, a new entry is created

5. If any of these changes implies in changes on **DataSets**, such as adding or changing members, the **DataSets** are deleted and recreated immediately.

IMPORTANT

If a new Tag is created, it must be set to **Advise** mode (in scan) only after saving the database, that is, to add a new Tag to an ICCP Client at run time, first add the point to the Bilateral Table and save it by using the **SaveDatabase** Tag, and then create an I/O Tag in the **E3** or **Elipse Power** application's Driver and set it to **Online** mode by configuring its **AllowRead** property to True.

6. Example of creating a Tag in **E3** or **Elipse Power** at run time.

```
Set NewTag = Item("ICCPDriver").AddObject("IOTag", False, "MyTagName")
NewTag.N1 = 1 'This is a Client Tag
NewTag.AllowRead = True
NewTag.Activate
```

CEPEL SAGE File

Parameters compatible with CEPEL SAGE (Open System for Power Management) configuration files are the following:

- On the **ICCP General** tab:
  - **Local P Selector (CNF - Config, PS)**: 1 (one)
  - **Local S Selector (CNF - Config, SS)**: 1 (one)
  - **Local T Selector (CNF - Config, TS)**: 1 (one)
  - **Local App ID (CNF - Config, Aptitle RRR)**: 1, 1, 3
  - **Local AE Qualifier (CNF - Config, AeQ)**: 1 (one)

- On the **ICCP Client** tab:
  - **Remote Domain (CNF - ID)**: Name of the remote domain
  - **Expected Billateral Table Name (LSC - VerBD)**: 0 (zero)
  - **Remote P Selector (CNF - Config, PS)**: 1 (one)
  - **Remote S Selector (CNF - Config, SS)**: 1 (one)
  - **Remote T Selector (CNF - Config, TS)**: 1 (one)
  - **Remote App ID (CNF - Config, Aptitle LLL)**: 1, 1, 2

- **Remote AE Qualifier (CNF - Config, AeQ)**: 1 (one)

Documentation of I/O Interfaces

This section contains the documentation of I/O Interfaces referring to **ICCP** Driver.

Configuration of a Driver

I/O Interface configuration is performed on a Driver's configuration dialog box. To access the configuration of this dialog box in **Elipse E3** in version 1.0, follow these steps:

1. Right-click a Driver object (IODriver).
2. Select the **Properties** item on the contextual menu.
3. Select the **Driver** tab.
4. Click **Other parameters**.

In **Elipse E3** version 2.0 or later, click **Configure driver** on a Driver's toolbar. In **Elipse SCADA**, follow these steps:

1. Open the Organizer.
2. Select a Driver on Organizer's tree.
3. Click **Extras** on the **Driver** tab.

Currently, an I/O Interface allows opening only one connection for each Driver. This means that, if users want to access two serial ports, they must add two Drivers to an application and then configure each one of these Drivers for each serial port.

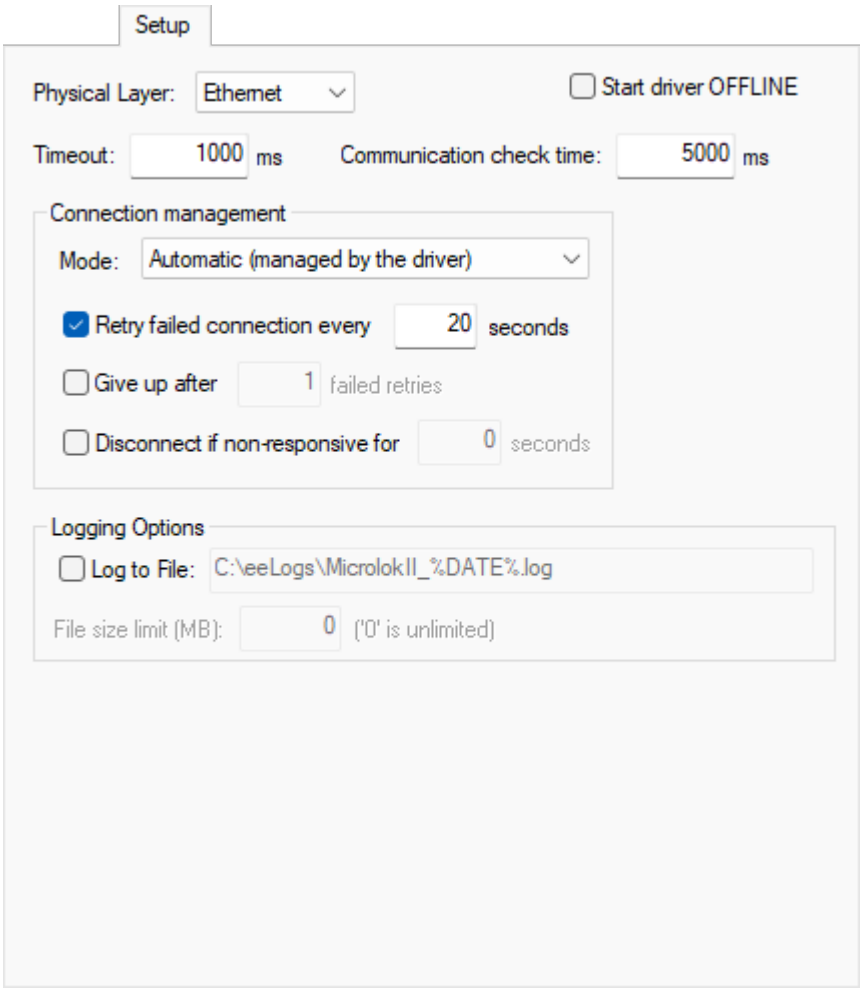
Configuration Dialog Box

The dialog box of I/O Interfaces allows configuring the I/O connection used by a Driver. This dialog box contains the **Setup**, **Serial**, **Ethernet**, **Modem**, and **RAS** tabs, described on the next topics. If a Driver does not implement a specific I/O connection, its corresponding tab is not available for configuration. Some Drivers may contain additional tabs, specific for that Driver, on the configuration dialog box.

Setup Tab

The **Setup** tab contains general configurations of a Driver. This tab is divided into the following groups:

- **General configurations**: Configurations of a Driver's physical layer, time-out, and initialization mode
- **Connection management**: Configurations on how the I/O Interface keeps a connection and which recovery policy is used on failure
- **Logging options**: Controls the generation of log files



Setup tab

General options on the Setup tab

OPTION	DESCRIPTION
Physical Layer	Select the physical layer on a list. Available options are **Serial**, **Ethernet**, **Modem**, and **RAS**. The selected interface must be configured on its specific tab
Timeout	Configure a time-out, in milliseconds, for the physical layer. This is the amount of time an I/O interface waits to receive any byte from the reception's buffer
Communication check time	Set the time, in milliseconds, to define the interval at which communication is considered to be in an inactive state. As long as an I/O Driver receives valid data, its communication state is considered active. However, if during operation an I/O Driver does not receive valid data inside this period of time, the state is considered inactive. The communication state is shown in the **IO.CommunicationStatus** Tag
Start driver OFFLINE	Select this option so that a Driver starts in **Offline** mode or stopped. This means that the I/O interface is not created until this Driver is configured to **Online** mode by using a Tag in an application. This mode enables a dynamic configuration of an I/O interface at run time

Options on the Connection management group

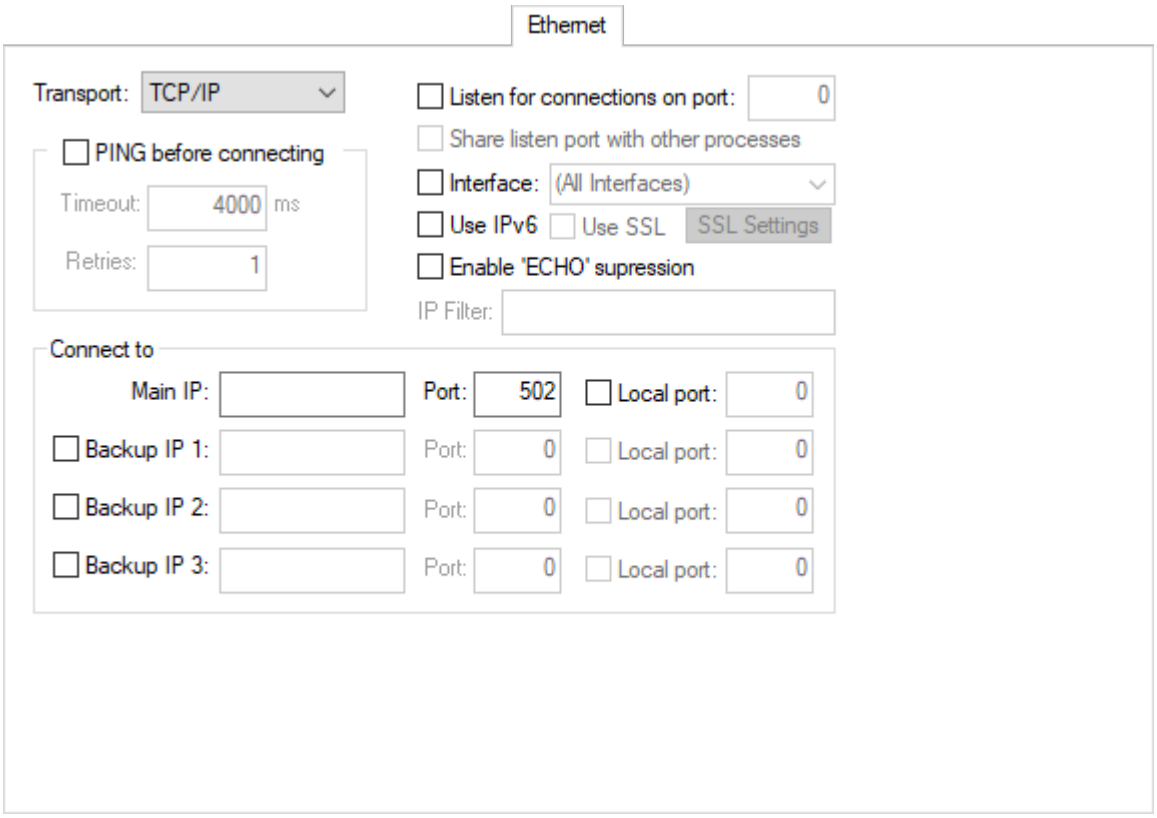
OPTION	DESCRIPTION
Mode	Selects a management mode of a connection. Selecting the **Automatic** option allows a Driver to manage the connection automatically, as specified in the next options. Selecting the **Manual** option allows an application to fully manage a connection. Please check topic **Driver Statuses** for more information
Retry failed connection every ... seconds	Select this option to enable a Driver's connection retry in a certain interval, in seconds. If the **Give up after failed retries** option is not selected, this Driver keeps retrying until a connection is performed, or until the application is stopped
Give up after ... failed retries	Enable this option to define a maximum number of connection retries. When the specified number of consecutive connection retries is reached, a Driver goes to the **Offline** mode, assuming that a hardware problem was detected. If a Driver establishes a successful connection, the number of unsuccessful retries is cleared. If this new connection is lost, then the retry counter starts at zero
Disconnect if non-responsive for ... seconds	Enable this option to force a Driver to disconnect if no byte was received by the I/O interface during the specified time-out, in seconds. This time-out must be greater than the time-out configured in the **Timeout** option

Options on the Logging Options group

OPTION	DESCRIPTION
Log to File	Enable this option and configure the name of a file to write a log. Log files can be large, so use this option for short periods of time, only for testing and debugging purposes. If the **%PROCESS%** macro is used in the log file name, it is replaced by the identifier of the current process. This option is particularly useful when using several instances of the same Driver in **Elipse E3**, thus allowing each instance to generate a separate log file. For example, when configuring this option with value "c:\e3logs\drivers\sim_%PROCESS%.log", it generates a file named c:\e3logs\drivers\sim_00000FDA.log for process **0FDAh**. Users can also use the **%DATE%** macro in the file name. In this case a log file is generated every day, in the format **aaaa_mm_dd**. For example, when configuring this option with value "c:\e3logs\drivers\sim_%DATE%.log", it generates a file named c:\e3logs\drivers\sim_2005_12_31.log in 12/31/2005 and a file named c:\e3logs\drivers\sim_2006_01_01.log in 01/01/2006. Similarly, the **%DATE_HOUR%** macro generates one log file per hour, in the format **aaaa_mm_dd_hh**
File size limit (MB)	Configure the log file size limit, in megabytes. A value equal to 0 (zero) means that there is no size limit for the log file

Ethernet Tab

Use this tab to configure parameters of an **Ethernet** Interface. These parameters, except port configurations, must also be configured for use in the **RAS** Interface.



Ethernet tab

Available options on the Ethernet tab

OPTION	DESCRIPTION
Transport	Select the value **TCP/IP** for a TCP socket (*stream*) or select the value **UDP/IP** to use a UDP socket (*connectionless datagram*)
Listen for connections on port	Use this option to wait for new connections in a specific IP port, common in Slave Drivers. If this option remains unselected, a Driver connects to the address and port specified in the **Connect to** option
Share listen port with other processes	Select this option to share the listening port with other Drivers and processes
Interface	Select the local network interface, identified by its IP address, that a Driver uses to establish and receive connections, or select the value **(All Interfaces)** to allow connection in any network interface
Use IPv6	Select this option to force a Driver to use addresses in **IPv6** format on all Ethernet connections. Leave this option deselected to use the **IPv4** format
Enable 'ECHO' suppression	Enable this option to remove the echo from received data. An echo is a copy of sent data, which can be returned before a reply message
IP Filter	List of restricted or allowed IP addresses from where a Driver accepts connections (*Firewall*). Please check the **IO.Ethernet.IPFilter** property for more information
PING before connecting	Enable this option to execute a **ping** command, that is, to check whether a device can be reached on a network, for a device before trying a socket connection. This is a quick way

OPTION	DESCRIPTION
	of determining a successful connection before trying to open a socket with a device. The time-out of a connection with a socket can be very high. The available options are: • **Timeout**: Specify the number of milliseconds to wait for a reply from a **ping** command. Users must use a **ping** command to check the normal reply time, configuring this option for a value above that average. Usually this value can be configured between 1000 and 4000 milliseconds, that is, between 1 (one) and 4 (four) seconds • **Retries**: Number of retries of a **ping** command, not counting the first attempt. If all attempts fail, then the socket connection is aborted

Available options on the Connect to group

OPTION	DESCRIPTION
Main IP	Type the IP address of a remote device. Users can use an IP address separated by dots, as well as a URL. In case of a URL, a Driver uses the available DNS service to map that URL to an IP address, such as "192.168.0.13" or "Server1"
Port	Type the IP port of a remote device, between 0 (zero) and 65535
Local port	Select this option to use a fixed local IP port when connecting to a remote device
Backup IP 1, 2, and 3	Indicate the IP address, the IP port, and the fixed local IP port of up to 3 (three) backup addresses of a remote device

General Configurations

This section contains information about the configuration of general **I/O Tags** and **Properties** of I/O Interfaces.

I/O Tags

General I/O Interfaces Tags (N2/B2 = 0)

The Tags described next are provided for all supported I/O Interfaces.

IO.CommunicationStatus

Type of Tag	I/O Tag
Type of Access	Reading
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	6 (six)
String Configuration	IO.CommunicationStatus

This Tag informs the communication status of a Driver. It indicates how communication works relative to receiving valid data within a time period arbitrated in the configuration. For more information, please check topic **Setup Tab**. Possible values are **0 - Inactive communication**: The Driver did not receive valid data or stopped receiving data after *n* milliseconds, as configured in the properties window, or **1 - Active communication:** The Driver is receiving valid data.

IO.IOKitEvent

Type of Tag	Block Tag
Type of Access	Read-Only
B1 Parameter	-1 (minus one)
B2 Parameter	0 (zero)
B3 Parameter	0 (zero)
B4 Parameter	1 (one)
Size Property	4 (four)
ParamItem Property	IO.IOKitEvent

This Block returns Driver events generated by several sources in I/O Interfaces. The **TimeStamp** property of this Block represents the moment this event occurred. The Block Elements are the following:

- **Element 0**: Type of event. Possible values are **0**: Information, **1**: Warning, or **2**: Error
- **Element 1**: Source of an event. Possible values are **0**: Driver (specific of a Driver), **-1**: IOKit (generic events of I/O Interfaces), **-2**: **Serial** Interface, **-3**: **Modem** Interface, **-4**: **Ethernet** Interface, or **-5**: **RAS** Interface
- **Element 2**: Error number, specific for each source of event
- **Element 3**: Message of an event, a **String** specific for each event

NOTE

A Driver keeps a maximum number of 100 events internally. If additional events are reported, older events are discarded.

IO.PhysicalLayerStatus

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	2 (two)
String Configuration	IO.PhysicalLayerStatus

This Tag indicates the status of a physical layer. Possible values are the following:

- **0**: Physical layer stopped, that is, a Driver is in **Offline** mode, the physical layer failed when initializing, or exceeded the maximum number of reconnection attempts
- **1**: Physical layer started but not connected, that is, a Driver is in **Online** mode but the physical layer is not connected. If the **Connection management** option is configured with the value **Automatic**, the physical layer can be connecting, disconnecting, or waiting for a reconnection attempt. If the **Connection management** option is configured with the vlaue **Manual**, then the physical layer remains in this status until forced to connect
- **2**: Physical layer connected, that is, the physical layer is ready for use. This **DOES NOT** mean a device is connected, only that the access layer is working

IO.SetConfigurationParameters

Type of Tag	Block Tag
Type of Access	Read-Only
B1 Parameter	-1 (minus one)
B2 Parameter	0 (zero)
B3 Parameter	0 (zero)
B4 Parameter	3 (three)
Size Property	2 (two)
ParamItem Property	IO.SetConfigurationParameters

Use this Tag to change any property of a Driver's configuration dialog box at run time.

This Tag works only while a Driver is in **Offline** mode. To start a Driver in **Offline** mode, select the **Start driver OFFLINE** option on that Driver's configuration dialog box. Users can write to a PLC Tag or to a Block Tag containing the parameters to change. Writing individual Block Elements is not supported, the whole Block must be written at once.

In **Elipse SCADA**, users must use a Block Tag. Every parameter to configure uses two Block Elements. For example, if users want to configure 3 (three) parameters, then the size of the Block must be 6 (six, 3 × 2). The first Element is the property's name, as a **String**, and the second Element is the property's value, according to the next example.

```
// 'Block' must be a Block Tag with automatic reading,
// scan reading, and automatic writings disabled.
// Configure all parameters
Block.element001 = "IO.Type" // Parameter 1
Block.element002 = "Serial"
Block.element003 = "IO.Serial.Port" // Parameter 2
Block.element004 = 1
Block.element005 = "IO.Serial.BaudRate" // Parameter 3
Block.element006 = 19200
// Writes the whole Block
Block.Write()
```

When using **Elipse E3**, the ability to create arrays at run time allows using an I/O Tag as well as a Block Tag. Users can use the **Write** method of a Driver to send the parameters directly to that Driver, without creating a Tag, according to the next example.

```
Dim arr(6)
' Configure all array elements
arr(1) = "IO.Type"
arr(2) = "Serial"
arr(3) = "IO.Serial.Port"
arr(4) = 1
arr(5) = "IO.Serial.BaudRate"
arr(6) = 19200
' There are two methods to send parameters
' Method 1: Using an I/O Tag
tag.WriteEx arr
' Method 2: Without using a Tag
Driver.Write -1, 0, 0, 3, arr
```

A variation of the previous example uses a bidimensional array.

```
Dim arr(10)
' Configure all array elements. Notice the array was resized
' to 10 elements. Empty array elements are ignored by a Driver
arr(1) = Array("IO.Type", "Serial")
arr(2) = Array("IO.Serial.Port", 1)
arr(3) = Array("IO.Serial.BaudRate", 19200)
Driver.Write -1, 0, 0, 3, arr
```

A Driver does not validate parameter names or passed values, therefore be careful when writing parameters and values. The **Write** method fails if the configuration array is incorrectly created. Users can check the log of a Driver or use the *writeStatus* parameter of the **WriteEx** method to find out the exact cause of an error.

```
Dim arr(10), strError
arr(1) = Array("IO.Type", "Serial")
arr(2) = Array("IO.Serial.Port", 1)
arr(3) = Array("IO.Serial.BaudRate", 19200)
If Not Driver.WriteEx -1, 0, 0, 3, arr, , , strError Then
  MsgBox "Failed configuring Driver parameters: " + strError
End If
```

IO.WorkOnline

Type of Tag	I/O Tag
Type of Access	Reading or Writing
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	4 (four)
String Configuration	IO.WorkOnline

This Tag informs the current status of a Driver and allows starting or stopping the physical layer. Possible values are the following:

- **0 - Driver Offline**: Physical layer is closed or stopped. This mode allows a dynamic configuration of a Driver's parameters using the **IO.SetConfigurationParameters** Tag
- **1 - Driver Online**: Physical layer is open or executing. While in **Online** mode, the physical layer can be connected or disconnected and its current status can be checked using the **IO.PhysicalLayerStatus** Tag

In the next example, using **Elipse E3**, a Driver is configured to **Offline** mode, its COM port is changed, and then configured to **Online** mode again.

```
'Configure to Offline mode
Driver.Write -1, 0, 0, 4, 0
'Change port to COM2
Driver.Write -1, 0, 0, 3, Array("IO.Serial.Port", 2)
'Configure to Online mode
Driver.Write -1, 0, 0, 4, 1
```

The **Write** method may fail when configuring a Driver to **Online** mode, that is, writing the value 1 (one). In this case, this Driver remains in **Offline** mode. The cause of failure can be:

- Type of physical layer incorrectly configured, probably an invalid value was configured in the **IO.Type** property
- This Driver may have run out of memory
- Physical layer probably did not create its working thread. Search the log file for a message "Failed to create physical layer thread!"
- Physical layer could not start. The cause of this failure depends on the type of physical layer. It can be an invalid serial port number, a failure when starting Windows Sockets, or a failure when starting TAPI (modem), among others. This cause is recorded on the log file

IMPORTANT

Even if the configuration of a Driver to **Online** mode is successful, this does not necessarily mean the physical layer is ready to use, that is, ready to execute input and output operations with an external device. The **IO.PhysicalLayerStatus** Tag must be checked to ensure the physical layer is connected and ready for communication.

Properties

These are general properties of all supported I/O Interfaces.

IO.ConnectionMode

9 Controls the management mode of a Connection. Possible values are **0**: Automatic mode, in which a Driver manages the connection or **1**: Manual mode, in which an application manages the connection.

IO.GiveUpEnable

◩ When configured to True, defines a maximum number of reconnection attempts. If all reconnection attempts fail, a Driver enters the **Offline** mode. When configured to False, a Driver tries until a reconnection is successful.

IO.GiveUpTries

9 Number of reconnection attempts before this one is aborted. For example, if the value of this property is equal to 1 (one), a Driver tries only one reconnection when the connection is lost. If this one fails, this Driver enters the **Offline** mode.

IO.InactivityEnable

◩ Configure to True to enable and to False to disable inactivity detection. The physical layer is disconnected if inactive for a certain period of time. The physical layer is considered inactive only if it is capable of sending data but not capable of receiving it back.

IO.InactivityPeriodSec

9 Number of seconds to check for inactivity. If the physical layer is inactive for this period of time, it is then disconnected.

IO.RecoverEnable

Configure to True to enable a Driver to recover lost connections and to False to leave a Driver in **Offline** mode when a connection is lost.

IO.RecoverPeriodSec

Delay time between two connection attempts, in seconds.

NOTE

The first reconnection is executed immediately after a connection is lost.

IO.StartOffline

Configure to True to start a Driver in **Offline** mode and to False to start a Driver in **Online** mode.

NOTE

It is pointless to change this property at run time, as it can only be changed when a Driver is already in **Offline** mode. To configure a Driver in **Online** mode at run time, write the value 1 (one) to the **IO.WorkOnline** Tag.

IO.TimeoutMs

Defines a time-out for the physical layer, in milliseconds. One second is equal to 1000 milliseconds.

IO.Type

Defines the type of physical interface used by a Driver. Possible values are the following:

- **N or None**: Does not use a physical interface, that is, a Driver must provide a customized interface
- **S or Serial**: Uses a local serial port (COM*n*)
- **M or Modem**: Uses a local modem, internal or external, accessed via TAPI (*Telephony Application Programming Interface*)
- **E or Ethernet**: Uses a TCP/IP or UDP/IP socket
- **R or RAS**: Uses a **RAS** (*Remote Access Server*) Interface. A Driver connects to a RAS device using the **Ethernet** Interface and then sends an **AT** (*dial*) command

Statistical Configuration

This section contains information about the configuration of **I/O Tags** and **Properties** of I/O Interfaces statistics.

I/O Tags

Tags of I/O Interface Statistics (N2/B2 = 0)

The Tags described next display statistics for all I/O Interfaces.

IO.Stats.Partial.BytesRecv

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1101
Configuration by String	IO.Stats.Partial.BytesRecv

This Tag returns the number of bytes received in the current connection.

IO.Stats.Partial.BytesSent

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1100
Configuration by String	IO.Stats.Partial.BytesSent

This Tag returns the number of bytes sent through the current connection.

IO.Stats.Partial.TimeConnectedSeconds

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1102
Configuration by String	IO.Stats.Partial.TimeConnectedSeconds

This Tag returns the number of seconds a Driver is connected in the current connection or 0 (zero) if a Driver is disconnected.

IO.Stats.Partial.TimeDisconnectedSeconds

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1103
Configuration by String	IO.Stats.Partial.TimeDisconnectedSeconds

This Tag returns the number of seconds a Driver is disconnected since the last connection ended or 0 (zero) if a Driver is connected.

IO.Stats.Total.BytesRecv

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1001
Configuration by String	IO.Stats.Total.BytesRecv

This Tag returns the number of bytes received since a Driver was loaded.

IO.Stats.Total.BytesSent

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1000
Configuration by String	IO.Stats.Total.BytesSent

This Tag returns the number of bytes sent since a Driver was loaded.

IO.Stats.Total.ConnectionCount

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1004
Configuration by String	IO.Stats.Total.ConnectionCount

This Tag returns the number of connections a Driver already established, successfully, since it was loaded.

IO.Stats.Total.TimeConnectedSeconds

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1002
Configuration by String	IO.Stats.Total.TimeConnectedSeconds

This Tag returns the number of seconds a Driver remained connected since it was loaded.

IO.Stats.Total.TimeDisconnectedSeconds

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1003
Configuration by String	IO.Stats.Total.TimeDisconnectedSeconds

This Tag returns the number of seconds a Driver remained disconnected since it was loaded.

Properties

Currently, there are no properties defined specifically to display I/O Interface statistics at run time.

Ethernet Interface Configuration

This section contains information about the configuration of **I/O Tags** and **Properties** of an **Ethernet** Interface.

I/O Tags

Tags of an Ethernet Interface (N2/B2 = 4)

The Tags described next allow controlling and identifying an **Ethernet** Interface at run time and they are also valid when the **RAS** Interface is selected.

IMPORTANT
These Tags are available **ONLY** while a Driver is in **Online** mode.

IO.Ethernet.IPSelect

Type of Tag	I/O Tag
Type of Access	Reading or Writing
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	4 (four)
N4 Parameter	0 (zero)
String Configuration	IO.Ethernet.IPSelect

Indicates the active IP address. Possible values are **0**: The main IP address is selected, **1**: The first alternative or backup IP address is selected, **2**: The second alternative or backup IP address is selected, or **3**: The third alternative or backup IP address is selected.

If the **Ethernet** or **RAS** Interface is connected, this Tag indicates which one of the four configured IP addresses is in use. If the Interface is disconnected, this Tag indicates which IP address is used first on the next attempt to connect.

During the connection process, if the active IP address is not available, the I/O Interface tries to connect using the other IP address. If the connection with the alternative IP address works, it is configured as the active IP address (automatic switchover).

To force a manual switchover, write values from 0 (zero) to 3 (three) to this Tag. This forces a reconnection with the specified IP address (**0**: Main address or **1, 2, 3**: Alternative address) if a Driver is currently connected. If a Driver is disconnected, this Tag configures the active IP address for the next attempt to connect.

IO.Ethernet.IPSwitch

Type of Tag	I/O Tag
Type of Access	Write-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	4 (four)
N4 Parameter	1 (one)
String Configuration	IO.Ethernet.IPSwitch

Any value written to this Tag forces a manual switchover. If the main IP address is active, then the first alternative or backup IP address is activated, and so on for all alternative IP addresses and returning to the main address until a connection is established.

If a Driver is disconnected, this Tag configures the active IP address for the next attempt to connect.

IO.Ethernet.SocketState

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	4 (four)
N4 Parameter	2 (two)
String Configuration	IO.Ethernet.SocketState

The Value property of this Tag corresponds to socket states as a map of bits:

- **Bit 0**: 0 (zero, not listening) or 1 (one, listening)
- **Bit 1**: 0 (zero, disconnected) or 1 (one, connected)

Properties

These properties control the configuration of an **Ethernet** Interface.

NOTE

The **Ethernet** Interface is also used by the **RAS** Interface.

IO.Ethernet.AcceptConnection

Configure to False if a Driver must not accept external connections, that is, if a Driver behaves as a master, or configure to True to enable the reception of connections, that is, if a Driver behaves as a slave.

IO.Ethernet.BackupEnable[2,3]

Configure to True to enable an alternative or backup IP address. If the reconnection attempt with the main IP address fails, a Driver tries to use an alternative IP address. Configure to False to disable its usage.

IO.Ethernet.BackupIP[2,3]

Alternative or backup IP address of a remote device. Users can use a numerical address, as well as a device's host name, such as "192.168.0.7" or "SERVER2".

IO.Ethernet.BackupLocalPort[2,3]

Local port number to be used when connecting to an alternative IP address of a remote device. Used only if **IO.Ethernet.BackupLocalPortEnable** is equal to True.

IO.Ethernet.BackupLocalPortEnable[2,3]

Configure to True to force the use of a specific local port when connecting to an alternative or backup IP address or configure to False to use any available local port.

IO.Ethernet.BackupPort[2,3]

Port number of an alternative or backup IP address of a remote device, used with the **IO.Ethernet.BackupIP** property.

IO.Ethernet.IPFilter

List with a comma-separated IPv4 or IPv6 addresses, which defines from which addresses a Driver accepts or blocks connections. Users can use asterisks, such as "192.168.*.*", or intervals, such as "192.168.0.41-50", in any part of IP addresses. To block an IP address or a range of IP addresses, use the tilde ("~") character at the beginning of the address, according to the next examples:

- **192.168.0.24**: Accepts only connections from IPv4 address 192.168.0.24
- **192.168.0.41-50**: Accepts connections from IPv4 addresses in the interval between 192.168.0.41 and 192.168.0.50
- **192.168.0.***: Accepts connections from IPv4 addresses in the interval between 192.168.0.0 and 192.168.0.255
- **fe80:3bf:877::*:* (expands to fe80:03bf:0877:0000:0000:0000:*:*)**: Accepts connections from IPv6 addresses in the interval between fe80:03bf:0877:0000:0000:0000:0000:0000 and fe80:03bf:0877:0000:0000:0000:ffff:ffff
- **192.168.0.10, 192.168.0.15, 192.168.0.20**: Accepts connections from IPv4 addresses 192.168.0.10, 192.168.0.15, and 192.168.0.20
- **~192.168.0.95, 192.168.0.***: Accepts connections from IPv4 addresses in the interval between 192.168.0.0 and 192.168.0.255, except the IPv4 address 192.168.0.95

When a Driver receives a connection attempt, the list of filters is scanned sequentially from left to right, searching for a specific authorization or block for the IP address where the connection comes from. If no element on the list corresponds to the IP address, the authorization or block are dictated by the last element of that list:

- If the last element on the list is an authorization, such as "192.168.0.24", then all IP addresses not found on the list are blocked
- If the last element on the list is a block, such as "~192.168.0.24", then all IP addresses not found on the list are authorized

If an IP address appears on more than one filter on the list, the leftmost filter has precedence. For example, in case of "~192.168.0.95, 192.168.0.*", the IP address 192.168.0.95 fits both rules, but the rule that wins is the leftmost one, "~192.168.0.95", and therefore this IP address is blocked.

When **IOKit** blocks a connection, it logs a message "Blocked incoming socket connection from {IP}!".

In case of UDP connections in broadcast listening mode, in which a Driver can receive packets from different IP addresses, blocks or permissions are performed at each packet received. If a packet is received from a blocked IP address, it logs a message "Blocked incoming packet from {IP} (discarding {N} bytes)!".

IO.Ethernet.ListenIP

A IP address of the local network interface that a Driver uses to establish and accept connections. Leave this property empty to establish and accepts connections using any local network interface.

IO.Ethernet.ListenPort

9 Number of the IP port used by a Driver to listen to connections.

IO.Ethernet.MainIP

A IP address of a remote device. Users can use a numerical address, as well as a device's host name, such as "192.168.0.7" or "SERVER2".

IO.Ethernet.MainLocalPort

9 Local port number to use when connecting to the main IP address of a remote device. This value is only used if the **IO.Ethernet.MainLocalPortEnable** property is equal to True.

IO.Ethernet.MainLocalPortEnable

◩ Configure to True to force the use of a specific local port when connecting to the main IP address of a remote device or configure to False to use any available local port.

IO.Ethernet.MainPort

9 Number of the IP port of a remote device, used with the **IO.Ethernet.MainIP** property.

IO.Ethernet.PingEnable

◩ Configure to True to enable sending a **ping** command to the IP address of a remote device, before trying to connect to the socket. This socket's connection time-out cannot be controlled, therefore sending a **ping** command before connecting is a fast way to detect if the connection is going to fail. Configure to False to disable a **ping** command.

IO.Ethernet.PingTimeoutMs

9 Delay time to wait for a response from a **ping** command, in milliseconds.

IO.Ethernet.PingTries

9 Maximum number of attempts of a **ping** command. Minimum value is 1 (one), including the first **ping** command.

IO.Ethernet.ShareListenPort

Configure to True to share a listening port with other Drivers and processes or False to open a listening port in exclusive mode. To successfully share a listening port, all Drivers and processes that use that port must open it in shared mode. When a listening port is shared, each incoming connection is distributed to one of the processes listening. This way, if a Slave Driver only supports one connection at a time, users can use several instances of this Driver listening on the same port, therefore simulating a Driver with support for multiple connections.

IO.Ethernet.SupressEcho

Configure to True to eliminate echoes in communication. An echo is the unwanted reception of an exact copy of all data packets a Driver sent to a device.

IO.Ethernet.Transport

Defines a transport protocol. Possible values are **T or TCP**: Uses the TCP/IP protocol or **U or UDP**: Uses the UDP/IP protocol.

IO.Ethernet.UseIPv6

Configure to True to use IPv6 addresses on all Ethernet connections or configure to False to use IPv4 addresses (default).

Driver Revision History

VERSION	DATE	AUTHOR	COMMENTS
2.0.31	12/10/2024	M. Salvador	• Delay option to report bad quality - client side (Case 33894) • Fixed COV Tag behavior (case 34009)
2.0.24	08/02/2022	M. Salvador	• Option to send the smallest possible number of bytes at OSI layers, among with log improvements. [29240] { implemented at ICCP v2.00 Build 19 } • Option to define if server supports Blocked Trransfers, reported at the read of the reserved variable "Supported_Features" [29985] { implemented at ICCP v2.00 Build 21 } • When in client mode a dataset creation fails, now the driver tries to read each tag individually in order to identify which of them is causing the failure. [30099] { implemented at no ICCP v2.00 Build 21 } • Improvements at error interpretation at dataset creation [30647] { implemented at ICCP v2.00 Build 22 } • Fixed ConnectionPresentationPDU

VERSION	DATE	AUTHOR	COMMENTS
			with a member with wrong type definition, working until 4 bytes only. [29239] { appeared at ICCP v1.00 Build 2 , fixed at ICCP v2.00 Build 19 } • Reports sent with field names in a ListOfVariables now allows the scope to be ICC or VCC [29984] { fixed at ICCP v2.00 Build 20 } • Revised disconnection behavior operating as a Server, at the absence of event acknowledge. Fixed also a possible situation where after a reconnection, the driver still demands ack of events that doesn´t exist anymore. [30866] { fixed at ICCP v2.00 Build 22 } • Created option to discard Null Timestamp Responses (NTR).
2.0.18	08/05/2019	M. Ludwig	• Driver ported to Visual Studio 2017 (*Case 27151*). • At ICCP Server mode, after a reconnection, data transfer with client could not work properly. [27037] { corrigido no ICCP v2.00 Build 17 } • Fixed aleatory report blocked processing erros, among log improvements when receiving an EventNotification. [26139] { fixed at ICCP v2.00 Build 16 }
2.0.14	09/24/2018	M. Salvador	• Driver ported to **IOKit** 2.0. • Implemented a function to import files in **XML** format. • Improvements on devices. • Implemented an **Offline** mode.
1.0.7	11/22/2012	M. Salvador	• Status of Server and Client connections. • Batch operations on the Server.
1.0.6	11/03/2011	M. Salvador	• Bug fixes.
1.0.2	08/01/2011	M. Salvador	• First version of this Driver.

Headquarters
Rua Mostardeiro, 322/Cj. 902, 1001 e 1002
90510-002 — Porto Alegre — RS
Phone: (+55 51) 3346-4699
Fax: (+55 51) 3222-6226
E-mail: elipse-rs@elipse.com.br

Branch in Taiwan
9F., No.12, Beiping 2nd St., Sanmin Dist.
807 — Kaohsiung City — Taiwan
Phone: (+886 7) 323-8468
Fax: (+886 7) 323-9656
E-mail: evan@elipse.com.br

Check our website for information about a representative in your country.

www.elipse.com.br

kb.elipse.com.br
forum.elipse.com.br
www.youtube.com/elipsesoftware
elipse@elipse.com.br

Gartner 2014 Cool Vendor

OPC FOUNDATION MEMBER

Gartner, Cool Vendors in Brazil 2014, April 2014.
Gartner does not endorse any vendor, product or service depicted in its research publications, and does not advise technology users to select only those vendors with the highest ratings. Gartner research publications consist of the opinions of Gartner's research organization and should not be construed as statements of fact. Gartner disclaims all warranties, expressed or implied, with respect to this research, including any warranties of merchantability of fitness for a particular purpose.

Microsoft Partner
Gold Independent Software Vendor (ISV)