[1. The eWON IO Servers 5](#_Toc503192854)

[1.1. Introduction 5](#_Toc503192855)

[1.2. IO Servers Setup 5](#_Toc503192856)

[1.2.1. IO Server Clear 6](#_Toc503192860)

[1.2.2. IO Server Init 6](#_Toc503192861)

[1.2.3. Assisted Edition 7](#_Toc503192862)

[1.2.4. Global Config 8](#_Toc503192863)

[1.2.5. Standard IO server configuration page 8](#_Toc503192864)

[2. MODBUS IO Server 10](#_Toc503192865)

[2.1. Introduction 10](#_Toc503192866)

[2.2. Setup 10](#_Toc503192867)

[2.2.1. Setup for eWON Server 10](#_Toc503192868)

[2.2.2. Setup for eWON IO server and Gateway - COM Setup 11](#_Toc503192869)

[2.2.3. Topic configuration 12](#_Toc503192870)

[2.2.4. Advanced parameters 12](#_Toc503192878)

[2.2.5. Additional advanced parameters 13](#_Toc503192879)

[2.3. Tag name convention 14](#_Toc503192880)

[2.3.1. ValueName 14](#_Toc503192882)

[2.3.2. Slave Address 16](#_Toc503192889)

[2.3.3. IP Address 16](#_Toc503192890)

[2.3.4. Device specific information 16](#_Toc503192891)

[3. NETMPI IO Server 17](#_Toc503192892)

[3.1. Introduction 17](#_Toc503192893)

[3.2. Setup 17](#_Toc503192894)

[3.3. Tag name convention 17](#_Toc503192895)

[3.3.1. valueName 18](#_Toc503192902)

[3.3.2. Device Address 18](#_Toc503192903)

[4. UNITE IO Server 19](#_Toc503192904)

[4.1. Introduction 19](#_Toc503192905)

[4.2. Setup 19](#_Toc503192906)

[4.2.1. Communication Setup 19](#_Toc503192907)

[4.2.2. Topic configuration 20](#_Toc503192908)

[4.3. Tag name convention 21](#_Toc503192914)

[4.3.1. Value Name 21](#_Toc503192915)

[4.3.2. The device address syntax 22](#_Toc503192916)

[5. DF1 IO Server 23](#_Toc503192917)

[5.1. Introduction 23](#_Toc503192918)

[5.2. Setup 23](#_Toc503192919)

[5.2.1. Communication Setup 23](#_Toc503192920)

[5.2.2. Topic configuration 24](#_Toc503192929)

[5.3. Tag name convention 26](#_Toc503192930)

[5.3.1. Value Name 26](#_Toc503192931)

[5.3.2. Destination Device Type and Address 29](#_Toc503192932)

[6. FINS IO Server 30](#_Toc503192933)

[6.1. Introduction 30](#_Toc503192934)

[6.2. Setup 30](#_Toc503192935)

[6.2.1. Communication Setup 30](#_Toc503192936)

[6.2.2. Topic Configuration 31](#_Toc503192937)

[6.2.3. Gateway Configuration 32](#_Toc503192938)

[6.3. Tag Name Convention 33](#_Toc503192939)

[6.3.1. Value Name 33](#_Toc503192940)

[6.3.2. Global Device Address 35](#_Toc503192941)

[7. S5-AS511 IO Server 36](#_Toc503192942)

[7.1. Introduction 36](#_Toc503192943)

[7.2. Setup 36](#_Toc503192944)

[7.2.1. Communication setup 36](#_Toc503192945)

[7.3. Tag name convention 37](#_Toc503192946)

[7.3.1. ValueName 37](#_Toc503192947)

[8. ABLOGIX IO Server 39](#_Toc503192948)

[8.1. Introduction 39](#_Toc503192949)

[8.2. Setup 39](#_Toc503192950)

[8.2.1. Communication Setup 40](#_Toc503192951)

[8.2.2. Topic configuration 41](#_Toc503192952)

[8.3. Tag name convention 42](#_Toc503192953)

[8.3.1. ValueName 42](#_Toc503192954)

[9. eWON IO Server 44](#_Toc503192955)

[9.1. Introduction 44](#_Toc503192956)

[9.2. Standard eWON I/O Item Names 44](#_Toc503192957)

[9.2.1. Tag address convention 44](#_Toc503192958)

[9.2.2. Basic Principle of the eWON Flexy I/O Tag Addresses 47](#_Toc503192959)

[9.2.3. Energy configuration with the LIx Tags 48](#_Toc503192960)

[9.2.4. Analog Input Value Ranges (eWON4002 & Flexy) 49](#_Toc503192961)

[9.3. Setup 52](#_Toc503192962)

[9.3.1. Configuration of the counter pulse length 53](#_Toc503192963)

[10. MEM IO Server 54](#_Toc503192964)

[10.1. Introduction 54](#_Toc503192965)

[10.2. Setup 54](#_Toc503192966)

[10.3. Tag name convention 54](#_Toc503192967)

[11. S7-300&400 (MPI/Profibus - ISOTCP) 55](#_Toc503192968)

[11.1. Introduction 55](#_Toc503192969)

[11.2. Setup 55](#_Toc503192970)

[11.2.1. COM configuration 55](#_Toc503192971)

[11.2.2. Advanced Routing Setup 56](#_Toc503192972)

[11.2.3. Topic configuration 57](#_Toc503192973)

[11.3. Tag name convention 58](#_Toc503192974)

[11.3.1. Value Name 58](#_Toc503192975)

[11.3.2. Global Device Address 59](#_Toc503192976)

[12. S7-200 (PPI - ISOTCP) 61](#_Toc503192977)

[12.1. Introduction 61](#_Toc503192978)

[12.2. Setup 61](#_Toc503192979)

[12.2.1. Gateway configuration 61](#_Toc503192980)

[12.2.2. COM configuration (serial port) 62](#_Toc503192981)

[12.2.3. COM configuration (MPI port) 63](#_Toc503192982)

[12.2.4. Topic configuration 64](#_Toc503192983)

[12.3. Tag name convention 64](#_Toc503192984)

[12.3.1. Value Name 65](#_Toc503192985)

[12.3.2. Global Device Address 67](#_Toc503192986)

[13. HITACHI IO Server 68](#_Toc503192987)

[13.1. Introduction 68](#_Toc503192988)

[13.2. Setup 68](#_Toc503192989)

[13.2.1. COM configuration 68](#_Toc503192990)

[13.2.2. Topic configuration 69](#_Toc503192991)

[13.3. Tag name convention 69](#_Toc503192992)

[13.3.1. Value Name 70](#_Toc503192993)

[13.3.2. Global Device Address 72](#_Toc503192994)

[14. MITSUBISHI FX IO Server 73](#_Toc503192995)

[14.1. Introduction 73](#_Toc503192996)

[14.2. Setup 73](#_Toc503192997)

[14.2.1. COM configuration 73](#_Toc503192998)

[14.2.2. Topic configuration 74](#_Toc503192999)

[14.3. Tag name convention 74](#_Toc503193000)

[14.3.1. Value Name 75](#_Toc503193001)

[15. SNMP IO Server 77](#_Toc503193002)

[15.1. Introduction 77](#_Toc503193003)

[15.2. Setup 77](#_Toc503193004)

[15.3. Tag name convention 78](#_Toc503193005)

[15.3.1. Value Name 78](#_Toc503193006)

[16. MITSUBISHI MELSEC IO Server 80](#_Toc503193007)

[16.1. Introduction 80](#_Toc503193008)

[16.2. Setup 80](#_Toc503193009)

[16.2.1. Topic configuration 80](#_Toc503193010)

[16.3. Tag name convention 81](#_Toc503193011)

[16.3.1. Value Name 82](#_Toc503193012)

# The eWON IO Servers

## Introduction

An IO Server is the interface between a changing value and the eWON monitoring engine. It is a kind of driver. Any variable from any *data source*

must have a common representation for all IO Servers in order to define common interface in the eWON.

The *data-source* representation in the eWON uses 3 fields for the definition of a Tag:

* **The IO Server Name**
* **The Topic name**
* **The Item Name**

A Tag’s data-source will be uniquely identified with these 3 parameters:

|  |  |
| --- | --- |
| **IO Server name:** | Is a kind of driver name. For each IO Server there is a specific Topic Name and Item Name syntax. Example: MODBUS, EWON, MEM |
| **Topic Name:** | Is used to group items inside an IO Server, for example the memory IO Server uses the blank topic ("") and the retentive topic ("ret"). All Tags of the MEM IO Server defined in the "ret" topic will have their value saved and restored when the eWON boots.  All IO servers do not use a Topic Name. In that case the Topic Name field must be left empty. |
| **Item Name:** | The item name is a string of characters; its syntax is specific to each IO Server. The Item Name describes the physical variable to monitor that uses the IO Server. |
| For example, the MODBUS IO Server needs to poll registers or coils from a slave, so it uses an item name representation to define the *register type, register address and slave address.* (Example "40001,5" => Where 4 means "read write register", 0001 is the register number and 5 is the slave Modbus address). | |

**Table 1: Tags data-source parameters**

**Important note: For optimization purpose, the eWON may disable the polling of "invalid tags" .**

## IO Servers Setup

Some of the IO servers are configurable.

The IO setup window proposes a list of IO servers:

**Figure 1: eWON IO servers scrolling list**

Click on the ***Edit*** hyper link or select another IO server to display its edition window. There are 3 possible cases regarding the IO server configuration:

* **The IO server is not configurable**
* **The IO server has a dedicated configuration page (ex: MODBUS, UNITE, NETMPI, DF1,...)**
* **The IO server uses the standard IO server configuration page.**



### IO Server Clear

To avoid the useless consumption of CPU and memory, the unused IO Servers need to be “unloaded” from CPU tasks and the used memory need to be cleared.

For that purpose, the Clear function must be used.

Only set to *disable* the Baud Rate of an IO Server make it idle, but this IO Server is still in memory and use some CPU time (for nothing).

Click on the ***Clear*** link to reset the config of the displayed IO Server and unload it from memory.

Some IO Servers are not dynamically stoppable and need an eWON reboot to ensure the unload from memory. You will be warned with the following message.



### IO Server Init

Click on the ***Init*** link to initialize an IOServer.

This initialization will reset all IOServer counters and restart the tags validation process.

## Assisted Edition

eWON features an Assisted Edition for the destination device address (the PLC to reach). This allows to encode the correct destination device address for each IOserver.

According to the IOserver you selected, eWON will display the different possibilities for the PLC device address. For example, for some IO servers you first need to specify the protocol that will be used for the polling.

You can with a mouse click select one of the proposed items, or simply enter the text directly inside the address field.

As long as the device address path is not complete, eWON will display the address field in black color. This means that some items are still missing inside the address path.

To continue the address path, select or enter one of the new proposed items.

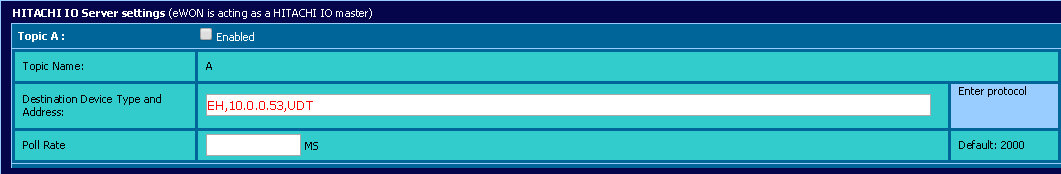
Once the encoded address is a valid address path, eWON will display the address in green.

You can now leave the address as it and save the configuration.

Or, if required, you can define supplementary optional address path options.

For example, in the picture here above clicking (or typing) the [,] option allows to encode a PLC address that can be reached on a second network behind the first PLC.

An incorrect address syntax will be displayed in red and you'll not be allowed to save the configuration.



**Note:** Assisted Edition is not applicable for MEM, EWON or MODBUS servers

## Global Config

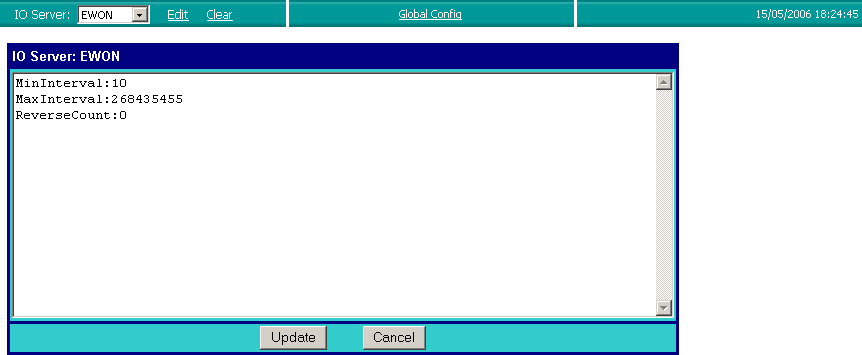
**Figure 2: IOServer Global Config page**

|  |  |
| --- | --- |
| **Item** | **Description** |
|  | Encode here the TCP Port number used for the Modbus TCP server protocol. *Default is 502* |
| **Ethernet/IP Port** | Encode here the TCP Port used for the Ethernet/IP protocol (Allen Bradley). *Default is 44818* |
| **ISOTCP Port** | Encode here the TCP Port number used for the ISOTCP protocol (Siemens). *Default is 102* |
| **FINS Port** | Encode here the TCP Port number used for the FINS protocol (Omron). *Default is 9600* |
| **Disable Tags in Error** | Activate the Tags Validation |
| **Default TCP RX/TX Timeout** | Value in mSec for IOServer TCP timeout (need reboot to apply) |

**Note**: To disable ports used for transparent gateway, set the corresponding port to 0 and reboot the eWON.

## Standard IO server configuration page

When no dedicated configuration page is defined for configuring an IO server, the standard configuration page is used.



**Figure 3: Standard IO server configuration page**

As you can see in the above example, the standard configuration screen is a simple text edition area. Each parameter is entered on a separate line, the parameter value is separated from the parameter name by a colon '**:**'

The generic format of a line is: **PARAM\_NAME:PARAM\_VALUE**

**Warning: Extra space must be removed.**

When using this configuration, you must respect the correct syntax of each parameter and the values for each parameter.

The list of valid parameters and their corresponding valid values are listed in the corresponding IO server documentation (see following chapters).

Any error that would occur when the eWON applies the configuration that you have entered would be written to the event file.

# MODBUS IO Server

## Introduction

The MODBUS IO Server setup is the standard remote IO communication setup of the eWON. It is used to configure:

* **The eWON as a Modbus RTU master.**
* **The eWON as a Modbus TCP slave and master.**

The first feature (Modbus TCP slave) is specific to the MODBUS IO server; it is actually designed to provide access to eWON Tag values and, unlike all the other IO servers, for interfacing field values with the eWON.

The second feature (MODBUS Master) is the actual IO server feature that provides an interface to the field values as a common IO server. The eWON MODBUS IO server will give access to values in equipments having a MODBUS interface.

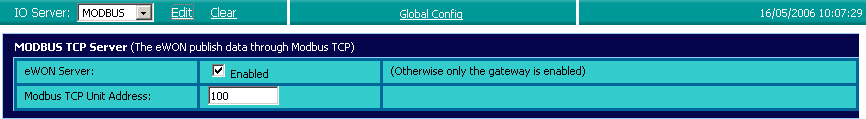
The interface can be:

* **RS485 / RS422 / RS232 – MODBUS RTU protocol will be used**
* **ETHERNET/PPP – MODBUS TCP protocol will be used.**

The eWON can mix access to MODBUS RTU and MODBUS TCP, depending on the way the Tag address is defined.

## Setup

### Setup for eWON Server



**Figure 4: Modbus TCP Server configuration**

This page defines the eWON configuration when used as a Modbus TCP slave.

As described in the Tag configuration paragraph, each Tag can be published to Modbus TCP so that a Modbus TCP can read their values. This setup screen defines the eWON address, and globally enables or disables the Modbus TCP slave feature.

|  |  |
| --- | --- |
| **eWON Server Properties** | **Description** |
| **eWON server enable** | Globally enables or disables the Modbus TCP Server feature. If disabled, then any request from a Modbus TCP master will be rejected, even if Tags are published. |
| **Modbus TCP Unit address** | This feature is used by some gateway but can usually be left to 1 because Modbus TCP appears as a point to point connection. |

**Table 2: eWON server configuration - eWON as Modbus TCP slave**

### Setup for eWON IO server and Gateway - COM Setup

**Figure 5: Modbus communications configuration**

If more than one Serial port are available, you must choose on which COM the MODBUS request will be sent.

This example of configuration shows an RS485 setup.

|  |  |
| --- | --- |
| **Field name** | **Description** |
| **COM Port** | Physical COM Port allocated to the Modbus IO Server. Drop down shows alternatives only if the relevant eWON has multiple serial ports (1). |
| **Baud Rate** | Baud rate of the serial link (has to be the same than the target device). Default is Disabled. |
| **Parity** | Parity |
| **Stop Bit(s)** | Stop bit(s) |
| **HW Mode** | HW mode (half Duplex, Full Duplex HW Handshaking, Full Duplex NO Handshaking |
| **Reply Timeout** | Time (in msec) the eWON waits for an answer from the connected Modbus device. The default value is 1000 msec |

**For the eWON Flexy:** The numbering of the serial COM ports starts with the serial port of the Base Unit (if available) and then continues with the upper port of the first Extension Module starting from the left. A maximum of 5 serial ports are supported.

**Warning:**

If multiple IO servers are potentially using the same serial port, the unused IO server must be cleared or the baud rate must be set to Disabled.

**Example:**

If Modbus and UniTE IO servers are configured to the same port, at least one of them must have its baud rate set to Disabled. If not the case, one of the IO servers will not be able to use the serial line and it will be disabled, with an error written in the event log.

### Topic configuration

**Figure 6: Modbus topics configuration**

Three topics can be used for the IO Server.

These topics are used to give a common property to a group of MODBUS Tag like:

* **Enable/Disable**
* **Poll rate**
* **Slave address (Modbus RTU)**
* **Unit address and TCP/IP address (Modbus TCP)**

|  |  |
| --- | --- |
| **Modbus Server Properties** | **Description** |
|  | Enables or disables polling of all the Tags in the topic. |
| **Slave address** | The slave address is a global parameter for all the Tags of the topic. If the slave is connected with Modbus RTU, Slave address must be entered and the IP address must be blank. If the slave is Modbus TCP, its unit address and its IP address must be entered. |
| **Poll rate** | This defines the rate to which the Tag will be refreshed. In a complex application, we can imagine that some Tags must be refreshed every second – typically for digital input - and other every minute – typically: temperature-. |

**Table 3: Modbus topic configuration**

**Warning:**

**Any slave address that is defined in the Topic configuration overwrites the slave address configured per Tag.**

**If a Tag is defined with Tag Address: 40010,5 and the global address of the topic is 5 and 10.0.0.81, the Tag is entered as Modbus RTU but it is polled as Modbus TCP. So, if you need to address slaves Tag by Tag, let the topic address configuration empty.**



### Advanced parameters

**Figure 7: Modbus advanced parameters configuration**

Extended parameters have been added to accommodate various special operating conditions.

They are entered in the edit box at the bottom of the configuration page, conforming to the syntax below.

Each parameter has a default value,

so the advanced parameter edit box must only be filled with the parameters for which default values must be changed.

|  |  |
| --- | --- |
| **Parameter name** | **Description** |
| **PreDelayFrom** | Used in conjunction with the next parameter (PreDelayVal), starting at that slave address, the eWON will insert a delay when switching from one slave address to another. If the PreDelayVal feature is not used, then the value for PreDelayFrom must be set to 256 (default value). |
| **PreDelayVal** | Used to define the delay (in Msec) to be inserted before a request to a slave if a request was previously sent to another slave address (not between 2 requests to the same slave).  The PreDelayVal is placed only for slaves with an address higher than PreDelayFrom. |
| **ErrorRetry** | Defines the number of errors before the device enters in slow poll mode. (Minimum 1) |
| **SlowPollSkip** | Defines the number of times the slave is skipped when in slow poll mode. This delay depends on the poll rate. |
| **GatewayIPCTimeout** | Maximum event lock while waiting for a response to a modbus gateway request from the modbus IO server (router) (in Msec – minimum 1000). |
| **PostErrorDelay** | This delay (in Msec) is added each time a device responds with a framing error or a CRC error, in order to help the bus to perform its re synchronization (default value = 50). |
| **MaxDeltaCoil*X* (X= A,B,C)** | Maximum number of Coils that can be grouped in a request (per topic) max=256 |
| **MaxDeltaReg*X* (X= A,B,C)** | Maximum number of registers that can be grouped in a request (per topic) max=124 |

**Table 4: extended IO server configuration controls**

### Additional advanced parameters

* **Gwdestaddr**

When the eWON is used as a Modbus gateway, it uses the UnitID from the ModbusTCP request to determine the Modbus RTU destination address.

It is possible to bypass this mechanism and force all the requests received by the eWON in ModbusTCP to be forwarded to a single ModbusRTU device (EXCEPT for requests with UnitID corresponding to the eWON's Modbus TCP Unit address (usually 100) when the eWON Server is enabled - see explanations about this precise point in the above configuration fields descriptions).

Every request is forwarded to the device with Slave address defined by the Modbus IO server advanced parameter called **gwdestaddr**. If the advanced parameters textfield contains the following entry:

#### gwdestaddr:10

Then all the requests will be forwarded to the slave with address 10.

REMARK: the eWON will change the address before sending the request to the slave, then it will change it back when sending the response to the master (Modbus TCP master). So the **gwdestaddr** will never appear in any communication between the Master and the eWON.

* **FastSrv**

FastSrv is a mode which allows a supervisor to read in Modbus TCP more easily the Modbus tags published by the eWON. This mode completely changes the tag’s addressing, and when activated, the Modbus addresses are no more compatible.

You have just to enter "FastSrv:1" in the Advanced Parameters text area to activate it. When done, the Modbus tags can be read as follows:

|  |  |
| --- | --- |
| **X** | Integer (with scale factors and offset defined) |
| **X+2048** | Float (X+2048: 1ST float, X+2048+2: 2nd float, etc.) |
| **X+4096** | Alarm status |
| **X+6144** | Alarm type |

**Notes:**

* If the tag is binary read, its value is worth 0 if tag==0, and 1 if tag <>0
* Writing 0 in AlarmStatus acknowledges the alarm (will be logged by eWON as acknowledged by administrator)
* It is not possible to write a coil in the float area (coil address range: X+2048 to X+4094)
* It is not possible to address more than 1024 registers in float.

Click on the ***Update Config*** button to validate your inputs or use the ***Cancel*** button to discard changes

* **TCP port**

Use the 'TcpPort' parameter to change the default 502 port used when the eWON is MODBUS TCP CLIENT.

If not specified, the port 502 is the default value used.

This Port value is used for all the MODBUS TCP client connections.

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server configuration** | | |
|  | MODBUS | |
| **Topic Name** | A or B or C | |
| **Item Name** | ValueName,SlaveAddress | The PLC Address is defined Tag by Tag on serial link (RTU Master) |
|  | ValueName,SlaveAddress,IPAddress | The PLC Address is defined Tag by Tag on TCP link |
|  | ValueName | The Topic PLC address is used |



### ValueName

The Modbus IO Server Tags can be classified following ranges of values. Two types of ranges are used. The two following tables describe the different ranges of value, for each of the two standards.

#### First standard:

|  |  |  |  |
| --- | --- | --- | --- |
| **Modbus Type** | **IO Type** | **Access** | **Register address** |
|  | Digital Output | R/W | 1  9999 |
|  | Digital Input | R | 10001  19999 |
|  | Analog Input | R | 30001  39999 |
|  | Analog Output | R/W | 40001  49999 |
|  | Digital Output | W | 50001  59999 |
|  | Analog Output | W | 60001  69999 |

**Table 5: Modbus IO server Tag name convention: first standard**

* **Second standard:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Modbus Type** | **IO Type** | **Access** | **Register address** |
|  | Digital Output | R/W | +1  +65535 |
|  | Digital Input | R | +100001  +165535 |
|  | Analog Input | R | +300001  +365535 |
|  | Analog Output | R/W | +400001  +465535 |
|  | Digital Output | W | +500001  +565535 |
|  | Analog Output | W | +600001  +665535 |

**Table 6: Modbus IO server Tag name convention: second standard**

The second standard allows more than 9999 values in each range. Notice the "+" sign before the register value.

\* The two last ranges are used with non-standard equipments that do not allow the reading of some of their values. In this case, specifying the address in the "write only" ranges informs the eWON not to read the values after setting them, which is normally done in the other cases. If those registers are read, the returned value will always be 0.

After the numerical value, the characters F, L, I, D or W can be used to specify how to read the value.

The following table describes the different character meaning.

|  |  |  |
| --- | --- | --- |
| **Character** | **Description** | **Automatic Tag type** |
| **W** | Reads 1 register considered as 16 bits unsigned integer (DEFAULT IF NOT SPECIFIED) | DWord |
| **I** | Reads 1 register considered as 16 bits signed integer | Integer |
| **D** | Reads 2 regs R1, R2 as a DWORD R1 is Less significant, R2 is most significant (32 bits, unsigned) (\*) | DWord |
| **E** | Reads 2 regs R1, R2 as a DWORD R2 is Less significant, R1 is most significant (32 bits, unsigned) (\*) | DWord |
| **L** | Reads 2 regs R1, R2 as a LONG R1 is Less significant, R2 is most significant (32 bits, signed) (\*) | Integer |
| **M** | Reads 2 regs R1, R2 as a LONG R2 is Less significant, R1 is most significant (32 bits, signed) (\*) | Integer |
| **F** | Reads 2 regs R1, R2 as a FLOAT R1 is Less significant, R2 is most significant (32 bits, signed) | Float |
| **H** | Reads 2 regs R1, R2 as a FLOAT R2 is Less significant, R1 is most significant (32 bits, signed) | Float |

**Table 7: the characters used to specify how to read the value**

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

When reading a 32 bits value, two consecutive registers or coils are read and combined

e.g. 40001L,11 to access in Long representation the reg 1 on the slave 11.

**Examples:**

|  |  |
| --- | --- |
| **Address** | **meaning** |
| **40001,10** | access the *Holding Register* at address 1 on the UnitID 10 |
| **1,11** | access the *Coil* at address 1 on the UnitID 11 |
| **+320234,12** | access the *Input Register* at address 20234 on the UnitID 12 |
| **40001,100,10.0.0.53** | access the *Hoding Register* at address 1 on the UnitID 100 at IP address 10.0.0.53 |
| **40010L,12** | access the LONG *Holding Register* at address 10 (and 11) on the UnitID 12 |
| **40008F,15** | access the FLOAT *Holding Register* at address 8 (and 9) on the UnitID 15 |

**Table 8: Modbus address examples**

* **STATUS TAG:**

The STATUS Tag is a special Tag that returns information about the current state of the communication for a given device.

As for other Tags, the status Tag ItemName is composed of:

* **STATUS,Address:**

If the address is omitted, the Topic address will be used (e.g.: status,11) points to the status of the slave 11 You can define a status Tag for each address used.

If you use the status MODBUS address, the Tag must be configured as analog:

|  |  |
| --- | --- |
| **Tag value** | **Meaning** |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 9: MODBUS status values**



### Slave Address

This is the address of the slave device that you want to access. It is a number from 0 to 255.

**Example:**

|  |  |
| --- | --- |
| **30001,11** | Polls a RTU device at address 11. |

### IP Address

This is the IP address of the device on an Ethernet network.

It is composed of 4 numbers separated by a dot.

**Example:**

|  |  |
| --- | --- |
| **30001,11,10.0.0.50** | Polls a device configured with IP address 10.0.0.50 and with Modbus slave address 11. |

### Device specific information

**Warning for new users of WAGO modules:**

Keep in mind that coil read and write don’t use the same address (offset of 0x200); please consult the Wago™ documentation.

**Example:**

If you use Wago*™* systems with two digital inputs and two digital outputs, inputs have addresses 1 and 2, and outputs have the same.

The only way to distinguish them it’s the read-only access or R/W access.

Tags: station 11

|  |  |  |
| --- | --- | --- |
| **Tag** | **Modbus address** | **Comment** |
| **MB\_DigIn1** | 10001,11 | Digital input module 1 -- read-only |
| **MB\_DigIn2** | 10002,11 | Digital input module 2 -- read-only |
| **MB\_DigOut1** | 00001,11 | Digital output module 1 for writing - Encode all leading zeroes! |
| **MB\_DigOut1Read** | 10513,11 | Digital output module 1 for reading only |
| **MB\_DigOut2** | 00002,11 | Digital output module 2 for writing Encode all leading zeroes! |
| **MB\_DigOut2Read** | 10514,11 | Digital output module 2 for reading only |

**Table 10: Wago*™* modules - addresses examples**

In *View I/O page*, you can change the value of MB\_DigOut1 with the update link (set to 1), and if you do that, you view that the value read is always 0.

**Why?**

Because the eWON reads the value at the WAGO address 1 (thus, DigIn1)!

If you want to read the state of the DigOut1, you must read it at WAGO address 513!

The same remark is applied for analog Modbus registers.

It’s the documented behavior of Wago*™*-Modbus modules; keep it in mind.

# NETMPI IO Server

## Introduction

NETMPI IO server is used to communicate with SIEMENS PLC.

The eWON will connect to the PLC’s MPI interface by means of the eLINK adapter.

The eLINK adapter is an interface with an MPI interface on one side and an Ethernet TCP/IP interface on the other side.

The eLINK interface will be configured with an IP that the eWON will use to poll data.

In addition to the IP address, the eWON will need the MPI address of the PLC on the MPI bus.

Using that eLINK interface, it is possible to poll different types of item in the PLC. These items types are described below.

## Setup

**Figure 8: NETMPI configuration window**

As for the Modbus and Unite protocols, it is possible to define topics, so that a single configuration can be applied to several Tags.

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server configuration** | | |
|  | NETMPI | |
| **Topic Name** | A | |
| B | |
| C | |
| **Item Name** | ValueName,RemoteStationAddress,eLinkIpAddress | PLC Address is defined Tag by Tag |
| ValueName | Topic PLC Address is used |

**Table 11: NETMPI - IO server configuration**

The Item Name can contain the PLC address where the value is polled, or not.

If address is also specified at topic level, the address specified at Tag level will be ignored.



### **v**alueName

|  |  |  |
| --- | --- | --- |
| **Syntax** | **Description** | **Automatic Tag type** |
| **DBxBy** | Data block x Byte offset y (0-255) | DWord |
| **DBxWy** | Data block x Word offset y (0 - 65535) | DWord |
| **DBxFy** | Data block x Float offset y (+-3.4e38) | Float |
| **DBxDy** | Data block x DWord offset y (-2147483648->2147483647) (\*) | Integer |
| **MxB** | Memento x as byte | DWord |
| **MxW** | Memento x as word | DWord |
| **MxF** | Memento x as real | Float |
| **MxD** | Memento x as double (\*) | Integer |
| **Tx** | Timer x (0-65535) | DWord |
| **Cx** | Counter x (0-65535) | DWord |

**Table 12: value names for NETMPI addresses**

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

* **Bit access modifier:**

In any of the above items, it is possible to access a single bit (except for float items).

#x must be appended to the Value Name. (Bit index goes from 0..31) The syntax can be used for reading bits and for writing them as well.

**Example:**

**DB1W13#3 represents bit 3 of W13 in DB 1**

**REM:**

**Bit index range depends on a referenced item (for W, bit index maximum value is 15)**

* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of the communication for a given device.

As for other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* + You can define a status Tag for each PLC used.
  + If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 13: Tag Status meaning**

### Device Address

The device address is either appended to the ValueName in the Item Name definition, or entered in the Topic global address fields. The device address is composed of: RemoteStationAddress,eLinkIpAddress:

* **RemoteStationAddress: is the PLC’s MPI address**
* **eLinkIpAddress: is the eLINK IP address.**

**Important:**

**If the PLC address is defined at the Topic level, it can be omitted in the Tag definition.**

**In that case the Tag name will only contain the "ValueName".**

**If the PLC address is specified at the Topic level, it will replace any address defined Tag by Tag.**

**Note:** Assisted Edition is enable for this IO server

# UNITE IO Server

## Introduction

The eWON RS485 link can be configured as an UNITELWAY SLAVE INTERFACE.

When the Baud Rate in the UNITE IO Server is set to a value different than "Disabled", the Unitelway slave module in the eWON will be enabled.

This Unitelway slave provides 2 features:

* **Poll items in a Unitelway capable device.**

The device can be the Unitelway master itself or a device addressable through the Unitelway master on the XWay network.

* Forward XIP requests from TCP/IP XIP to Unitelway bus and thus, act as a gateway between XIP and Unitelway.

Using that feature, it is possible to access a PLC connected to the eWON’s Unitelway link by connecting PL7PRO using the XIP driver started with the eWON IP address as destination.

## Setup

### Communication Setup

**Figure 9: XWAY-UNITELWAY configuration**

If more than one Serial port are available, you must choose on which COM the Unitelway request will be sent.

|  |  |
| --- | --- |
| **Field name** | **Description** |
| **COM Port** | Physical COM Port allocated to the Unite IO Server. Drop down shows alternatives only if the relevant eWON has multiple serial ports (1). |
| **Baud Rate** | Baud rate of the serial link (has to be the same than the target device). Default is 9600.. |
| **Parity** | The parity to apply: none / odd / even. This field is set by default to Odd, as in the main cases in a typical UniTE topology. However, the eWON allows you to define a different parity type (Even or None), in case this is needful to comply with your industrial network installations. |
| **Stop Bit(s)** | Number of stop bits |
| **Master Response Timeout** | Maximum time the eWON will wait for a valid message from the Unitelway master. This value can be critical for a correct operation, depending on the responsiveness of the master. A value of 1000 should be selected to guarantee a correct operation. |
| **Rx message timeout (MSEC)** | Maximum time between a request is posted and the response is received |
| **Tx message timeout (MSEC)** | Maximum time for a request to be sent |

**Table 14: XWAY communication setup controls (Part 1)**

|  |  |
| --- | --- |
| **Field name** | **Description** |
| **Force UnitelWay V2** | If checked, the eWON will initiate a communication in V2 with the devices. When used with a TSX PLC, this check box can be left unchecked. |
| **ADO** | Link address base. The eWON will respond to AD0 and AD0+1 on the Unitelway link.  The eWON will act as an Unitelway slave, it will respond to 2 consecutive link addresses AD0 and AD0+1, doing this improves the throughput of data across the eWON when acting as a gateway. |
| **Xway Network Station** | Address of the eWON on the XWAY network.  When acting as an XIP to Unitelway gateway, the eWON will only respond the XWay network station defined here. Any XIP frame addressed to another network station will be ignored. |

**Table 14: XWAY communication setup controls (Part 2)**

* + - * 1. **For the eWON Flexy:** The numbering of the serial COM ports starts with the serial port of the Base Unit (if available) and then continues with the upper port of the first Extension Module starting from the left. A maximum of 5 serial ports are supported.

**Warning**:

When there are multiple IO servers potentially using the serial line, the unused IO servers must be cleared or the baud rate must be set to ***Disabled***.

**Example**:

If Modbus **and** UniTE IO servers are available, at least one of them must have its baud rate configured to Disabled.

If not, one of the IO servers will not be able to use the serial line and it will be disabled, with an error written in the event log.

### Topic configuration

**Figure 10: XWAY-UNITELWAY topics configuration**

Three topics can be used for the IO Server.

These topics are used to give a common property to a group of UNITE Tags such as:

* **Enable/Disable**
* **Poll rate**
* **Global Slave address**

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
|  | Enables or disables polling of all the Tags in the topic. |
| **Global Slave Address** | This Slave address is a global parameter for all the Tags of the topic. See below for the Device Address syntax. If an address is specified here, it will replace (overload) the address defined Tag by Tag. |
| **Poll rate** | Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag name must be refreshed every second – typically for digital input - and other every minute – typically: temperature-. |

**Table 15: UNITE - topics configuration**



## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server configuration** | | |
|  | UNITE | |
| **Topic Name** | A | |
| B | |
| C | |
| **Item Name** | ValueName,Network,Station,Gate,Module,Channel | PLC Address is defined Tag by Tag (and the gate requires a 5 level addressing) |
| ValueName,Network,Station,Gate | PLC Address is defined Tag by Tag |
| ValueName | Topic PLC Address is used |

**Table 16: UNITE - IO server configuration**

The Item Name can contain the PLC address where the value is polled, or not.

If the address is also specified at topic level, the address specified at Tag level will be ignored.

If the Topic address is not defined either, then address 0,254,0 will be used.

### Value Name

Value name follows the syntax below:

|  |  |  |
| --- | --- | --- |
| **Syntax** | **Description** | **Automatic Tag type** |
| **MWxW** | Internal data word 16 bits (unsigned) | DWord |
| **MWxI** | Internal data word 16 bits (signed) | Integer |
| **MWxD** | Internal data word 32 bits as DWORD (unsigned) (\*) | DWord |
| **MWxF** | Internal data word 32 bits as IEEE float | Float |
| **MWxL** | Internal data word 32 bits as LONG (signed) (\*) | Integer |
| **SWxW** | System data word 16 bits (unsigned) | DWord |
| **SWxI** | System data word 16 bits (signed) | Integer |
| **SWxD** | System data word 32 bits as DWORD (unsigned) (\*) | DWord |
| **SWxL** | System data word 32 bits as LONG (signed) (\*) | Integer |
| **Mx** | Internal data bit | DWord |
| **Sx** | System data bit | DWord |

**Table 17: value names for UNITE addresses**

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

**Notes:**

SW type cannot be formatted as float.

The eWON allows you to optimize the requests in case you need to read a lot of Tags that have been created on the UniTelWay device.

Imagine you have 100 Tags to read, the eWON will group the Tags within a predefined limit in order to make the less as reading operations as possible.

The number of Tags that can be read depends of the types of words or bits that have to be read:

* SW and MW types: by groups of 50
* S and M types: by groups of 200

It is possible to read one bit from a word, the syntax to add is as follows: #0 to #31

That means that if you want to read the fifth bit from internal data word 16 bits unsigned that you address MW0, you have to add "#4" at the end of the address: MW0#4.

The type of words for which this syntax can be applied are: MWxW, MWxI, MWxD, MWxL, SWxW, SWxI, SWxD and SWxL (please report to the table above).

**Examples:**

|  |  |
| --- | --- |
| **Address** | **point to** |
| **MW10 or MW10W** | the Internal data Word (16bits) at address 10 |
| **MW10D** | the Internal data Word (as DWord 32bits) at address 10 |
| **MW10W#5** | the bit 5 (the sixth) from the Word at address 10 |
| **SW2**  **or SW2W** | the System data Word (16bits) at address 2 |

**Table 18: UNITE address examples**

* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of the communication for a given device. As for other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* You can define a status Tag for each PLC used
  + - * If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 19: Tag Status meaning**

### The device address syntax

The Device Address is used in the topic definition or in the Tag definition.

If used in the Tag definition, it will be separated from the value name by a coma (’,’)

* **Network,Station,Gate**

**Example:** 0,254,0 to access the UnitelWay device directly connected on the eWON

or

* **Network,Station,Gate,Module,Channel**

The second case applies to addresses with 5 levels:

* ***Network*: 0..127**
* ***Station*: 0..63**
* **Gate:**
* **Module:**
* **Channel:**

Module and channel can be omitted if not required.

**Note:** If the address is not specified, 0,254,0 will be used.

**Important: If an address is specified in a Topic definition it will replace any address defined Tag by Tag. Note:** Assisted Edition is enable for this IO server

# DF1 IO Server

## Introduction

The eWON serial link can be configured as a DF1 INTERFACE. When the Baud Rate in the DF1 IO Server is set to a value different than "Disabled", the DF1 module in the eWON will be enabled.

This DF1 module provides 3 features:

* **Poll items in SLC50x and PLC5 controllers using PCCC requests.**
* **Forward PCCC requests from EIP (TCP/IP) to DF1 bus and thus, acts as an adapter between EIP and DF1.**

Thanks to that feature, it is possible to access a PLC connected to the eWON’s DF1 link by connecting RSLogix 500 using RSLinx TCP driver started with the eWON IP address as destination.

* **Poll DH+ device using ControlLogix gateway**

The DF1 IO Server must be configured in **Full Duplex mode** (eWON serial link must be configured in RS232 mode, with dipswitch). **Note**: Please check in the “Rockwell compatibility matrix” if your device is supported.

## Setup

### Communication Setup

#### Figure 11: DF1 communications setup

If more than one Serial port are available, you must choose which COM the DF1 request will be sent to. The following parameters can be modified:

|  |  |
| --- | --- |
| **Field name** | **Description** |
| **COM Port** | Physical COM Port allocated to the DF1 IO Server.  **Note**: In the example above, the port selection drop down is not displayed because the eWON has only one single serial port. For multiple serial ports see remark (1). |
| **Baud Rate** | Select the Baud rate from 1200 to 38400 Bauds |
| **Parity** | The parity to apply: none / odd / even |
| **Stop Bit(s)** | Number of stop bits |
| **Frame Error Detection** | Cyclic Redundancy Check (CRC) or Block Check Character (BCC) |
| **HW mode** | Full Duplex no handshaking or Half duplex |
| **Master Response Timeout** | Maximum time the eWON will wait for a valid message from the DF1 master. This value can be critical for a correct operation, depending on the responsiveness of the master. A value of 1000 should be selected to guarantee correct operations |

**Table 20: DF1 communication setup controls (Part 1)**

|  |  |
| --- | --- |
| **Field name** | **Description** |
| **Rx message timeout (MSEC)** | Maximum time between a request is posted and the response is received |
| **Tx message timeout (MSEC)** | Maximum time for a request to be sent |
| **eWON DF1 address** | The eWON’s device address on the DF1 link  When the eWON acts as a DF1 slave, it will respond to 2 consecutive link addresses; doing this improves the throughput of data across the eWON when acting as a gateway. (eWON DF1 address and eWON DF1 address +1) |
| **Destination DF1 address** | Device Address of Destination on DF1 link when EIP is used |
| **Bridge EIP connection** | When bridging is configured, EIP connections requests are forwarded to the PLC on the DF1 interface (mandatory for Logix family). This is the default configuration.  When not configured, connection requests are canceled by the eWON (must be disabled for SLC family). |

**Table 20: DF1 communication setup controls (Part 2)**

* + - * 1. **For the eWON Flexy:** The numbering of the serial COM ports starts with the serial port of the Base Unit (if available) and then continues with the upper port of the first Extension Module starting from the left. A maximum of 5 serial ports are supported.

**Warning**: When there are multiple IO servers potentially using the serial line, the unused IO servers must be cleared or the baud rate must be set to ***Disabled***.

**Example:**

If Modbus **and** DF1 servers are available, at least one of them must have its baud rate configured to Disabled.

If not, one of the IO servers will not be able to use the serial line and it will be disabled, with an error written in the event log.



### Topic configuration

**Figure 12: DF1 topics configuration**

Three topics can be used for the IO Server.

These topics are used to give a common property to a group of DF1 Tags like:

* **Enable/Disable**
* **Poll rate**
* **Destination Device Type and Address**

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
|  | Enables or disables the polling of all the Tags in the topic |
| **Destination Device Type and Address** | The Destination Device Type and Address is a global parameter for all the Tags of the topic. See below for the Device Address syntax. If an address is specified here, it will replace (overload) the address defined Tag by Tag |
| **Poll rate** | Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag name must be refreshed every second – typically for digital input - and other every minute  – typically: temperature- |

**Table 21: DF1 topics configuration setup**

#### DF1: serial link

The Destination Device Type and address are:

*SLC500-x* (where *x* is the address of your SLC500 or Micrologix Device, range 0..254)

*PLC5-x* (where *x* is the address of your PLC5 Device, range 0..254)

**Example :**

**Topic A Destination = SLC500-1**

**Topic B Destination = PLC5-6**

**For accessing a Micrologix, use the SLC500 syntax**

**Topic C Destination = SLC500-3 (accessing the Micrologix at address 3)**

#### DF1: Ethernet routing

Thanks to the ABLogix IOServer, the eWON is now able to poll data on SLC500/PLC5 devices by its Ethernet link.

Even SLC500 connected behind your ControlLogix Network becomes available for polling.

With this use of DF1 IOServer, the Serial config can be left without configuration (baud rate=disabled), only one Topic enabled is required. You need to use a similar syntax than in ABLogix IOServer.

***[DeviceType-] IP Address [, Port, Link]***

* ***DeviceType*** = type of PLC ( SLC500 or PLC5 )
* ***IP Address*** = address on your Ethernet network (i.e.: 10.0.0.50)
* ***Port*** = value from 1 to 3 representing: 1 = Backplane, 2 = Channel A, 3 = Channel B
* ***Link*** could be:

¤ *Slot*: representing the Slot on the Backplane (0=CPU)

¤ *Node ID*: value from 0 to 99 (for ControlNet)

¤ *IP address*

* To reach a SLC500 with Ethernet address 10.0.0.60:

Topic A Destination = ***10.0.0.60 or SLC500-10.0.0.60***

* To reach a PLC5 with Ethernet address 10.0.0.65: Topic A Destination = ***PLC5-10.0.0.65***
* To reach a SLC500 connected behind a ControlLogix:

Topic A Destination = ***10.0.0.80,1,3,2,45 or SLC500-10.0.0.80,1,3,2,45***

* ***10.0.0.80*** = IP address of the ControlLogix
* ***1*** = BackPlane
* ***3*** = Slot 3 (COM card in ControlNet)
* ***2*** = Channel A (of the Card present in Slot 3)
* ***45*** = NodeID of the SLC500 (in the ControlNet network)
* To reach a PLC5 connected behind a ControlLogix:

Topic A Destination = ***PLC5-10.0.0.90,1,3,2,44***

* ***10.0.0.90*** = IP address of the ControlLogix
* ***1*** = BackPlane
* ***3*** = Slot 3 (COM card)
* ***2*** = Channel A (of the Card present in Slot 3)
* ***44*** = NodeID of the PLC5 (in the COM network)

#### DH+ polling using ControlLogix gateway

This feature allows the eWON to poll PLC Tags located on a DH+ network, behind a ControlLogix equipped with one or more DHRIO cards. Polling is done using eWON EIP protocol.

The DF1 IO server is used to access the "SLC500" PLC family. The ABLOGIX IO server is used to access the "LOGIX" PLC family.

The syntax *[Device Type-] IP address [,Port,Link]* is still applicable but in this particular case:the [,Port,Link] part becomes mandatory

The DH+ channel used to reach the "SLC500" PLC must be specified using the "A" or "B" character.

**Example:**

To reach a SLC500 with DH+ node Id 5, using channel A of a DHRIO card at slot 9 of a ControlLogix at IP address 10.0.30.21.

Topic A Destination = ***SLC500-10.0.30.21,1,9,A,5***

* ***10.0.30.21* =** IP address of the ControlLogix
* ***1* =** BackPlane
* ***9* =** Slot 9 (COM card)
* ***A* =** Channel A (of the Card present in Slot 9)
* **5 =** NodeID of the SLC500 (in the COM network)

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server configuration** | | |
|  | DF1 | |
| **Topic Name** | A | |
| B | |
| C | |
| **Item Name** | ValueName, DeviceType-DeviceAddress | Device Type and Address are defined Tag by Tag Only device Type SLC500 is supported  Device Address is a number between 0..254 |
| ValueName | Topic PLC Address is used |

**Table 22: DF1 IO server configuration**

The Item Name can contain the PLC address where the value is polled, or not.

If the address is also specified at the topic level, the address specified at Tag level will be ignored.

### Value Name

#### General Description

The general format of value names for data from SLC-500 controllers matches the naming convention used by the programming software. The format is shown below (The parts of the name shown between square brackets are optional).

**General Value Name Format: X[file]:element [.field][/bit]**

X: Identifies the file type. The below table summarizes the valid file types, default file number for each type and the fields allowed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **X** | **File Type** | **Default file Number** | **Fields** | **Automatic Tag type** |
| **O** | Output | 0 |  | DWord |
| **I** | Input | 1 |  | DWord |
| **S** | Status | 2 |  | DWord |
| **B** | Binary | 3 |  | DWord |
| **T** | Timer | 4 | .PRE, .ACC, .EN, .TT, .DN | DWord |
| **C** | Counter | 5 | .PRE, .ACC, .CU, .CD, .DN, .OV, .UN, .UA | DWord |
| **R** | Control | 6 | .LEN, .POS, .EN, .DN, .ER, .UL, .IN, .FD | DWord |
| **N** | Integer | 7 |  | DWord |
| **F** | Floating | 8 |  | Float |
| **A** | ASCII | none |  | DWord |

**Table 23: value names for DF1 addresses**

* **File: File number must be 0-255 decimal**
* **Element: Element number within the file**
* **Field: Valid only for Counter, Timer and Control files**
* **/bit: Valid for all types except Floating**

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

#### Output File Items

**Output File Item Format: O[n]:e.s[/b]**

* **"n" represents the file number and is optional. Value is always zero**
* **"e" indicates the element number in the file (0..30)**
* **"s" indicates the sub-element number (0..255)**
* **"b" specifies the bit (0..15) decimal**

**Note:**

For the Micrologix Series, the syntax for Output File Items is slightly different:

Basically, it drops 1 digit to show O:1 instead of O:0.1

Actually, the eWON reads consecutive words.

Example for the Micrologix 1400

|  |  |
| --- | --- |
| **PLC Addressing** | **eWON Addressing** |
| O:0.2 | O:2 |
| O:1.0 | O.6 |

This example shows that when stepping to another output card,

the incrementation continues in the eWON (from O:5 to O:6) while in the PLC itself it increments the first digit (from O:0.5 to O:1.0).

#### Input File Items

**Input File Item Format: I[n]:e.s[/b]**

* **"n" represents the file number and is optional. Value is always one**
* **"e: indicates the element number in the file (0..30)**
* **"s" indicates the sub-element number (0..255)**
* **"b" specifies the bit (0..15) decimal**

**Note:**

For the Micrologix Series, the syntax for Input File Items is slightly different:

Basically, it drops 1 digit to show O:1 instead of O:0.1

Actually, the eWON reads consecutive words.

Example for the Micrologix 1400

|  |  |
| --- | --- |
| **PLC Addressing** | **eWON Addressing** |
| I:0.7 | I:7 |
| I:1.0 | I.8 |

This example shows that when stepping to another input card,

the incrementation continues in the eWON (from I:7 to I:8) while in the PLC itself it increments the first digit (from I:0.7 to I:1.0).

#### Status File Items

**Status File Item Format: S[n]:e[/b]**

* **"n" represents the file number and is optional. If not specified, it is assumed to be 2**
* **"e" indicates the element number in the file (0..255)**
* **"b" specifies the bit (0..15) decimal**

#### Binary File Items

**Binary File Item Format: B[n]:e/b**

* **"n" represents the file number & is optional.**

**If not specified, it is assumed to be 3, otherwise must be between 3 and 255 decimal**

* **"e" indicates the element number in the file (0..255)**
* **"b" specifies the bit (0..15) decimal**

**Note:**

The format B[n]/b is not supported.

#### Timer File Items

**Timer File Item Format: T[n]: e [.f] [/b]**

* **"n" represents the file number and is optional.**

**If not specified, it is assumed to be 4, otherwise must be between 4 and 255 decimal**

* **"e" indicates the element number (3 words per element) in the file (0..255)**
* **"f" identifies one of the valid values for timer fields specified in the table above.**

**If omitted it is assumed to be the word containing the status bits**

* **"b" specifies the bit (0..15) decimal**

#### Counter File Items

**Counter File Item Format: C[n]: e [.f] [/b]**

* **"n" represents the file number and is optional.**

**If not specified, it is assumed to be 5, otherwise must be between 5 and 255 decimal**

* **"e" indicates the element number (3 words per element) in the file (0..255)**
* **"f" identifies one of the valid values for counter fields specified in the table above.**

**If omitted it is assumed to be the word containing the status bits**

* **"b" specifies the bit (0..15) decimal**

#### Control File Items

**Counter File Item Format: C[n]: e [.f] [/b]**

* **"n" represents the file number and is optional.**

**If not specified, it is assumed to be 6, otherwise must be between 6 and 255 decimal**

* **"e" indicates the element number (3 words per element) in the file (0..255)**
* **"f" identifies one of the valid values for counter fields specified in the table above.**

**If omitted it is assumed to be the word containing the status bits**

* **"b" specifies the bit (0..15) decimal**

#### Integer File Items

**Integer File Item Format: N[n]:e[/b]**

* **"n" represents the file number and is optional.**

**If not specified, it is assumed to be 7, otherwise must be between 7 and 255 decimal**

* **"e" indicates the element number in the file (0..255)**
* **"b" specifies the bit (0..15) decimal**

#### Floating File Items

**Floating File Item Format: F[n]:e**

* **"n" represents the file number and is optional.**

**If not specified, it is assumed to be 8, otherwise must be between 8 and 255 decimal**

* **"e" indicates the element number in the file (0..255)**

#### ASCII File Items

**ASCII File Item Format: An:e[/b]**

* **"n" represents the file number and is not optional**
* **"e" indicates the element number in the file (0...255)**
* **"b" specifies the bit (0..15) decimal**

#### Status register

The STATUS Tag is a special Tag that returns information about the current state of the communication for a given device.

As for other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* You can define a status Tag for each PLC used.
* If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 24: Tag Status meaning**

### Destination Device Type and Address

The Device Address is used in the topic definition or in the Tag definition.

If used in the Tag definition, it will be separated from the value name by a coma (’,’).

**Format: DeviceType-DeviceAddress**

The Destination Device Type and address are:

* *SLC500-x* (where *x* is the address of your SLC500 or Micrologix Device, range 0..254)
* *PLC5-x* (where *x* is the address of your PLC5 Device, range 0..254)

**Example:**

**Topic A Destination = SLC500-1**

**Topic B Destination = PLC5-6**

**For accessing a Micrologix, use the SLC500 syntax**

**Topic C Destination = SLC500-3 (accessing the Micrologix at address 3)**

**Format: [DeviceType-] IP address [,Port,Link]**

**Example:**

10.0.0.80 to access a SLC500 (or a Micrologix)

**Note:**

Assisted Edition is enable for this IO server

# FINS IO Server

## Introduction

The FINS IO Server includes the configuration of:

**The eWON as a FINS Hostlink client (master)**

**to give access on values in CSCJ series OMRON equipments reachable using eWON serial port(s).**

**The eWON as a FINS TCP/UDP client (master)**

**to give access on values in CSCJ OMRON equipments reachable using eWON Ethernet interface.**

**The eWON as a FINS TCP/UDP server acting as a gateway between the Ethernet/PPP interface and the serial interface**

**(used to connect remotely programming/monitoring software to OMRON FINS supporting equipment reachable using eWON serial ports).**

The FINS IO Server is designed to provide simultaneous access to OMRON equipments on its serial interface and Ethernet interface. The correct protocol will depend on the topic that the Tag belongs to. UDP and TCP protocols can be mixed as well on the Ethernet interface.

When the Baud Rate in the FINS IO Server is set to a value different than "Disabled", the serial Hostlink Client will be enabled. The FINS IO Server can be configured in 3 modes:

* **Full Duplex mode (eWON serial link must be configured in RS232 mode) without HW handshaking**
* **Full Duplex mode with HW handshaking**
* **Half Duplex slave mode (RS485 mode)**

## Setup

### Communication Setup

**Figure 13: FINS IO server COM setup**

The following parameters can be modified:

|  |  |
| --- | --- |
| **Field name** | **Description** |
| **COM Port** | Physical COM Port allocated to the FINS IO Server.  **Note**: The port selection drop down is not displayed if the eWON has only one single serial port. For multiple serial ports see remark (1). |
| **Baud Rate** | Disabled, 1200, 2400, 4800, 9600, 19200, 38400, 57600 |
| **Parity** | None, Odd, Even |
| **Data Bits** | 7, 8 |
| **Stop Bit(s)** | 1,2 |
| **HW mode** | Full Duplex no HW handshaking, Full Duplex HW handshaking, Half Duplex |
| **ReplyTimeout** | The maximum time the eWON will wait for a valid FINS message response (applicable for Ethernet and serial interface). |
| **Ethernet FINS network** | Source Network Address (SNA) filled in a FINS request message originating from the eWON and sent out on the Ethernet interface. |
| **Ethernet FINS node** | Source Node Address (SA1) filled in a FINS request message originating from the eWON and sent out on the Ethernet interface. It uniquely identifies the eWON on the Ethernet network. |
| **Serial FINS network** | Source Network Address (SNA) filled in a FINS request message originating from the eWON and sent out on the serial interface. |
| **Serial FINS node** | Source Node Address (SA1) filled in a FINS request message originating from the eWON and sent out on the serial interface. It uniquely identifies the eWON on the serial network. |

**Table 25: FINS IO server COM setup configuration fields**

* + - * 1. **For the eWON Flexy:** The numbering of the serial COM ports starts with the serial port of the Base Unit (if available) and then continues with the upper port of the first Extension Module starting from the left. A maximum of 5 serial ports are supported.

**Warning**: When there are multiple IO servers potentially using the serial line, the unused IO servers must be cleared or the baud rate must be set to ***Disabled***.

**Example:** if Modbus **and** FINS servers are available, at least one of them must have its baud rate configured to Disabled. If not, one of the IO servers will not be able to use the serial line and it will be disabled, with an error written in the event log.

### Topic Configuration

**Figure 14: FINS IO server topic configuration**

Three (3) topics can be used for the IO Server.

These topics are used to give a common property to a group of FINS Tags like:

* **Enable/Disable**
* **Protocol**
* **Global Device Address**
* **Polling Rate**

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
|  | Enables or disables the polling of all the Tags in the topic. |
| **Protocol** | Protocol used for the tags belonging to this topic: FINS Serial, FINS UDP, FINS TCP. |
| **Global Device Address** | See below for the Device Address Syntax. If an address is specified here, it will replace (overload) the address-defined Tag by Tag. |
| **Poll rate** | Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag names must be refreshed every second - typically for digital input - and other every minute - typically: temperature-. |

**Table 26: FINS IO server topic configuration fields**

### Gateway Configuration

The following parameters can be modified:

**Figure 15: FINS IO server gateway configuration**

|  |  |
| --- | --- |
| **FINS TCP Server Node** | The eWON FINS server node address used during FINS TCP session establishment (exchange of the FINS node address messages, and allocation of a FINS TCP Client node if necessary). |
| **Routing Entry 1..4** | For each defined destination network, gives the matching relay destination node. This is used to fill in the host link unit ID in the hostlink frame which encapsulates the FINS message sent out on the serial interface. |

**Table 27: FINS IO server gateway configuration fields**

## Tag Name Convention

|  |  |  |
| --- | --- | --- |
| **IO Server configuration** | | |
| **IO Server Name** | FINS | |
| **Topic Name** | A | |
| B | |
| C | |
| **Item Name** | ValueName, FINS Network, FINS Node, Hostlink or Ip Address | If FINS Serial has been chosen at topic level, Hostlink value has to be defined.  If FINS UDP or FINS TCP has been chosen at topic level, IP address has to be defined. |
| ValueName | Topic PLC Address is used. |

The Item Name can contain the PLC address where the value is polled, or not.

If address is also specified at the topic level, the address specified at the Tag level will be ignored.

### Value Name

#### General Description

The format of value names for data from CS/CJ OMRON controllers is shown below. Its is based on the naming convention used by the CX Programmer programming software. The format is shown below (The parts of the name shown in square brackets are optional).

**General Value Name Format: X[bank number:][modifier]word address[#bit address]**\*

* **X identifies the Memory area acronym.**
* **[bank number:] is only supported by the E memory area. Values 0 to max values for memory area.**
* **[modifier] allow you to interpret the data in a specific type (Word if omitted).**
* **[#bit address] is only supported by A, D, CIO, H and W memory areas. Values 0 to 15.**

*\*items between brackets "[]" are optional (the brackets should not be used!)*

**Supported memory areas:**

|  |  |
| --- | --- |
| **X** | **Memory area** |
| **A** | Auxiliary area |
| **C** | Counter area |
| **CIO** | Core I/O area |
| **D** | Data Memory area |
| **E** | Extended Data Memory area |
| **H** | Holding area |
| **T** | Timer Area |
| **W** | Work Area |

**Table 29: FINS IO server supported memory areas table**

**Modifier:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Modifier** | **value range** | **Automatic Tag type** |
| **W** | Word (default) | 0 .. 65535 | DWord |
| **S** | signed Word | -32768 .. 32767 | Integer |
| **D** | DWord | 0 .. 4294967296 (\*) | DWord |
| **L** | signed DWord | -2147483648 .. 2147483647 (\*) | Integer |
| **F** | Float | +/- 3.4e38 | Float |

**Table 30: FINS Modifiers**

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

**Examples:**

|  |  |
| --- | --- |
| **Address** | **point to** |
| **A5** | the Word at address 5 in the Auxiliary Area (equivalent to AW5) |
| **AS5** | the Signed Word at address 5 in the Auxiliary Area |
| **A2#5** | the bit 5 from the Word at address 2 in the Auxiliary Area |
| **E0:S3** | the signed Word at address 3 from the Extended Data Memory 0 |
| **C5** | the Counter at address 5 |
| **D4** | the Data Memory at address 4 |

**Table 31: FINS address examples**

* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of communication for a given device. As for the other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* You can define a status Tag for each PLC used.
* If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 32: Tag Status meaning**

### Global Device Address

The global device address is used in the topic definition or in the Tag definition.

If it is used in the Tag definition, it will be separated from the value name by a coma.

**Format: Network,Node,HostLin or Network,Node,IP**

**Figure 16: FINS IOServer topic setup**

**Note:** Assisted Edition is enable for this IO server

# S5-AS511 IO Server

## Introduction

This IO server is intended for use with Siemens S5 PLCs communicating via the front programming port using AS511 protocol.

The AS511 protocol is specific for each Siemens device. This IO Server has been designed to operate with a set range for Siemens equipment. Use of the IO Server on devices other than those listed is not recommended and not supported.

The Siemens S5 PLC family has a unique memory structure. Data within the PLC are not located at fixed locations within the PLC's memory space. This memory space is continuously updated and revised as you create and modify your PLC logic. When these revisions occur, the location of key data elements such as flags, timers, counters, I/O and data blocks can move around in the PLC's memory. The Siemens S5 IO Server has been designed to read the location of these memory elements when the driver first begins operation upon detecting a communication error or after a request (which is not a read or a write request) has been transmitted to the PLC. If you change your PLC configuration you must restart the Siemens S5 IO Server or simply unplug/re-plug the connection cable. Both actions will cause the Siemens S5 IO Server to reacquire the location of all PLC memory elements.

## Setup

**Figure 17: S5-AS511 IO server setup window**

**For the eWON Flexy:** The numbering of the serial COM ports starts with the serial port of the Base Unit (if available) and then continues with the upper port of the first Extension Module starting from the left. A maximum of 5 serial ports are supported.

**Warning**: When there are multiple IO servers potentially using the serial line, the unused IO servers must be cleared or the baud rate must be set to ***Disabled***.

**Example:** if Modbus **and** S5-AS511 servers are available, at least one of them must have its baud rate configured to Disabled. If not, one of the IO servers will not be able to use the serial line and it will be disabled, with an error written in the event log.

### Communication setup

The AS511 link uses a RS-232 Current Loop functioning mode.

A specific cable (with electronic) is therefore required (Siemens S5 Cable: ACT’L ref EW40910)

The communication parameters are fixed to the following values:

* 9600 Baud (Fixed)
* Even Parity (Fixed)
* 8 Data Bits (Fixed)
* 1 Stop Bit (Fixed)
* Full duplex no handshaking (Fixed)

#### Supported Devices

|  |  |  |
| --- | --- | --- |
| * Siemens S5 - 90U * Siemens S5 - 95U | * Siemens S5 - 100U - 100 * Siemens S5 - 100U - 101 * Siemens S5 - 100U - 103 * Siemens S5 - 101U | * Siemens S5 - 115U - 941 * Siemens S5 - 115U - 942 * Siemens S5 - 115U - 943 * Siemens S5 - 115U - 944 * Siemens S5 - 115U - 945 |

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO server Name** | S5-AS511 |  |
| **Topic name** | A or B or C |  |
| **Item Name** | ValueName, ComPortNum | COM port is defined Tag by Tag |
|  | ValueName | Topic COM port is used (or default) |

**Table 33: S5-AS511 IO server Tag name convention table**

**ComPortNum**: The port number used to access the PLC; if not specified, the default COM port is used.

* + - * If a port address is specified in the topic, it is used and overloads per Tag address.
      * If no address is specified, neither in the topic, nor at the Tag level, then the default port is used (Default port is the ECIA Port - COM:1).

### ValueName

|  |  |  |
| --- | --- | --- |
| **Syntax** | **Description** | **Automatic Tag type** |
| **DBxLy** | Data block x Word offset y, left byte of word (0 - 255) | DWord |
| **DBxYy** | Data block x Word offset y, left char of word (-128 - 127) | Integer |
| **DBxRy** | Data block x Word offset y, right byte of word (0 - 255) | DWord |
| **DBxZy** | Data block x Word offset y, right char of word (-128 - 127) | Integer |
| **DBxWy** | Data block x Word offset y, full word (0 - 65535) | DWord |
| **DBxSy** | Data block x Word offset y, integer (-32768 - 32767) | Integer |
| **DBxDy** | Data block x DWord offset y (0 - 4294967296 ) (\*) | DWord |
| **Tx or Kx** | Timer x (0-65535) | DWord |
| **Cx or Zx** | Counter x (0-65535) | DWord |
| **MxB or FxB** | Memento as Byte | DWord |
| **MxW or FxW** | Memento as Word | DWord |
| **MxD or FxD** | Memento as DWord (\*) | DWord |
| **Ix or Ex** | Input | DWord |
| **Qx or Ax** | Output | DWord |

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

* **Bit access modifier:**

In any of the above items, it is possible to access a single bit.

#x must be appended to the Value Name (Bit index goes from 0..31).

The syntax can be used for reading bits and for writing them as well.

**Example:**

**DB1W13#3 represents bit 3 of W13 in DB 1 Q32B#0 represents bit 0 of Q32.**

**Note:**

That you must define the type byte (B) to successfully extracts a bit (Q32#0 will not work).

**REM:** **Bit index range depends on item referenced (for W, bit index maximum value is 15)**

**Notes:**

* For DB the smallest element is a WORD and we count addresses in Word.

This means that DB1W0 and DB1W1 WILL NOT overlap.

* Timers and Counters are always Words and addresses are counted in WORD as well.

This means that T0 and T1 WILL NOT overlap.

* For M, I, Q addresses are counted in BYTES.

This means that M0W and M1W WILL overlap.

* Both syntax (German and English) can be used for I, Q, C which in German gives: E, A, Z.
* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of communication for a given device. As for other Tags, the status Tag ValueName is composed of:

**Status,ComPortNum**

* You can define a status Tag for each COM port used.
* If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 34: Tag status meaning**

**Note:** Assisted Edition is enable for this IO server

# ABLOGIX IO Server

## Introduction

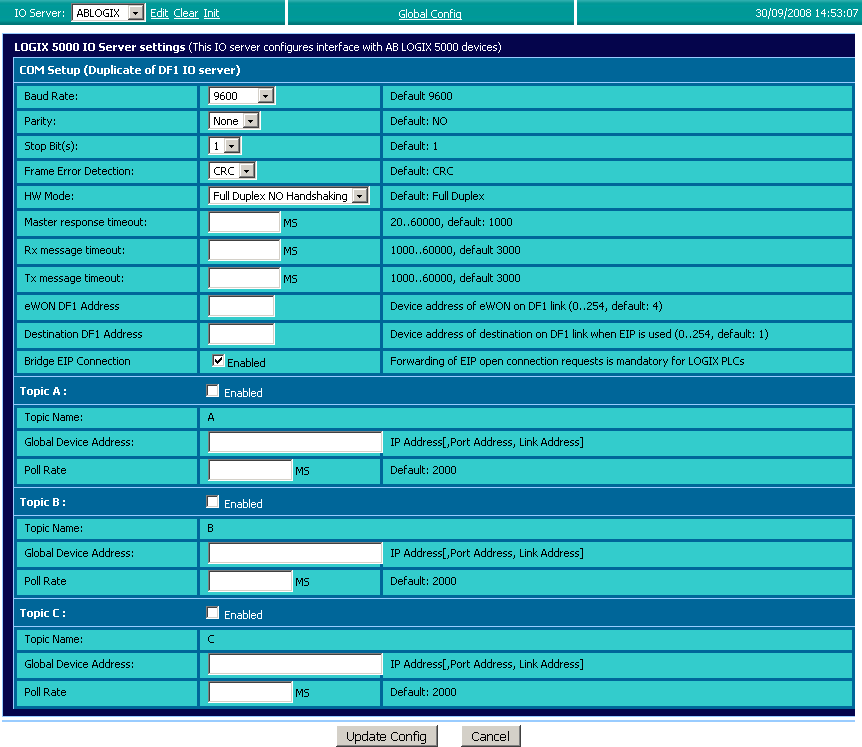
ABLogix IO server is used to communicate with Allen Bradley Logix Series PLCs on the Ethernet link or on the Serial link (in DF1 protocol).

* **It allows the eWON to poll ABLOGIX PLCs in order to read data.**
* **It allows the eWON to act as EIP-DF1 gateway for remote maintenance of the ABLOGIX PLCs.**
* **It allows the eWON to poll DH+ device using ControlLogix gateway.**

The eWON only allows DF1 communication with devices directly connected to the eWON Serial link (the ABLOGIX PLC may not be located behind another PLC connected on the DF1 serial link acting itself as gateway).

**Note:** Please check in the “Rockwell compatibility matrix” if your device is supported.

## Setup

******

**Figure 18: ABLOGIX IO server setup window**

### Communication Setup

In the current implementation,

the serial link configuration may be done using the DF1 IO Server configuration page or the AB LOGIX IO server page.

The same settings apply for both IO servers (DF1 and ABLogix).

**Note**: In the config.txt file the serial link parameters are defined in the DF1 IO server entry.

|  |  |
| --- | --- |
| **Field name** | **Description** |
| **COM Port** | Physical COM Port allocated to the ABLOGIX IO Server.  **Note**: The port selection drop down is not displayed if the eWON has only one single serial port. For multiple serial ports see remark (1). |
| **Baud Rate** | Select the Baud rate from 1200 to 38400 Bauds |
| **Parity** | The parity to apply: none / odd / even |
| **Stop Bit(s)** | Number of stop bits |
| **Frame Error Detection** | Cyclic Redundancy Check (CRC) or Block Check Character (BCC) |
| **HW mode** | Full Duplex no handshaking or Half duplex |
| **Master Response Timeout** | Maximum time the eWON will wait for a valid message from the DF1 master. This value can be critical for a correct operation, depending on the responsiveness of the master. A value of 1000 should be selected to guarantee correct operation |
| **Rx message timeout (MSEC)** | Maximum time between a request is posted and the response is received |
| **Tx message timeout (MSEC)** | Maximum time for a request to be sent |
| **eWON DF1 address** | Device Address of eWON on DF1 link  When the eWON acts as a DF1 slave, it will respond to 2 consecutive link addresses; doing this improves the throughput of data across the eWON when acting as a gateway. (eWON DF1 address and eWON DF1 address +1) |
| **Destination DF1 address** | Device Address of Destination on DF1 link when EIP is used |
| **Bridge EIP connection** | When bridging is configured, EIP connections requests are forwarded to the PLC on the DF1 interface (mandatory for Logix family). This is the default configuration.  When not configured, connection requests are canceled by the eWON (must be disabled for SLC family). |

**Table 35: ABLOGIX-DF1 communication setup controls**

* + - * 1. **For the eWON Flexy:** The numbering of the serial COM ports starts with the serial port of the Base Unit (if available) and then continues with the upper port of the first Extension Module starting from the left. A maximum of 5 serial ports are supported.

**Warning**: When there are multiple IO servers potentially using the serial line, the unused IO servers must be cleared or the baud rate must be set to ***Disabled***.

**Example:** if Modbus **and** ABLOGIX servers are available, at least one of them must have its baud rate configured to Disabled. If not, one of the IO servers will not be able to use the serial line and it will be disabled, with an error written in the event log.

### Topic configuration

Three topics can be used for the IO Server. These topics are used to give a common property, to a group of Tags, like:

* **Enable/Disable**
* **Global Device Address**
* **Poll rate**

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
|  | Enables or disables the polling of all the Tags in the topic |
| **Global Device Address** | The Global Device Address is a global parameter for all the Tags of the topic. See below for the Device Address syntax. If an address is specified here, it will replace (overload) the address defined Tag by Tag |
| **Poll rate** | Defines the refresh rate (in mSec) of the Tag name. In a complex application, we can imagine that some Tag name must be refreshed every second – typically for digital input - and other every minute – typically: temperature- |

**Table 36: ABLOGIX topics configuration setup**

#### ABLogix : DF1 serial link

The Global Device Address has the following syntax: ***Address***

* ***Address*** = DF1 Node ID of the device (i.e.: 5)

#### ABLogix : EIP Ethernet link

The Global Device Address has the following syntax : ***IP Address,Port,Link***

* ***IP Address*** = address on your Ethernet network (i.e.: 10.0.30.7)
* ***Port*** = value from 1 to 3 representing: 1 = Backplane, 2 = Channel A, 3 = Channel B
* ***Link*** could be:

¤ *Slot*: representing the Slot on the Backplane (for example: 0=CPU)

¤ *Node ID*: value from 0 to 99 (for ControlNet)

¤ *IP address*

**REM: In the case of using a ControlLogix as Gateway, the Device Address could be like following:**

**IP Address, Port, Link[, Port,Link][, Port,Link]…**

#### DH+ polling using ControlLogix gateway

This feature allows the eWON to poll PLC tags located on a DH+ network, behind a ControlLogix equipped with one or more DHRIO cards. Polling is done using the eWON EIP protocol.

The ABLOGIX IO server is used to access the "LOGIX" PLC family.

The DF1 IO server is used to access the "SLC500" PLC family

The general syntax *IP address,Port,Link* is still applicable. There is no specific character to indicate the DH+ channel ("2" or "3" should be used).

**Example:**

To reach a SLC500 (with CPU in slot 0)

with DH+ node Id 3 using channel A of a DHRIO card at slot 9 of a ControlLogix at IP address 10.0.30.21.

Topic A Destination = ***SLC500-10.0.30.21,1,9,2,3,1,0***

* ***10.0.30.21*** = IP address of the ControlLogix
* ***1*** = BackPlane
* ***9*** = Slot 9 (COM card)
* = Channel A (of the Card present in Slot 9)
* = NodeID of the PLC5 (in the COM network)
* **1** = BackPlane
* **0** = slot (of the CPU in our example

## Tag name convention

The Item Name can contain the PLC address where the value is polled, or not.

If the address is also specified at topic level, the address specified at Tag level will be ignored.

|  |  |  |
| --- | --- | --- |
| **IO Server Configuration** | | |
| **IO Server Name** | ABLOGIX | |
| **TOPIC Name** | A or B or C | |
| **Item Name** | ValueName | Topic PLC address is used |
| ValueName, IP address[, Port, Link] | PLC address is defined Tag by Tag |
| ValueName, NodeID | PLC address is defined Tag by Tag |

**Table 37: ABLOGIX IO server Tag name convention table**

### ValueName

ValueName follows the below syntax:

**[PROGRAM:ProgName.]SymbolicTagName**

* **ProgName is the name of the program where the Tag is.**

If no ProgName, the Tag is in the global scope (tag is controller type)

* **SymbolicTagName:**

Symbol of the tag.

Only following atomic type are supported: BOOL, SINT, INT, FLOAT, DINT (\*), BIT ARRAY

Bit selection with <SymbolicTagName>/bit where bit is the bit number (from 0 to 31)

ex: controlbit/4 read the bit 4 of the controlbit register

TIMER, CONTROL, COUNTER predefined types with <SymbolicTagName>.acc (or ctl or pre)

ex: MyVar.acc read the counter of MyVar

Element of a table <Symbol\_1>[idx\_1].<Symbol\_2>[idx\_2].<Symbol\_3>[idx\_3].<Symbol> With maximum 3 index.

ex: table1[2].subtable[6].element read data named 'element' on the index 6 of the 'sub table' from the index 2 of the 'table1'

Element of a structure <Symbol\_1>.<Symbol\_2>

ex: CounterObj.init read the 'init' part of the structure 'CounterObj'

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

**If you choose “Automatic” Tag type, eWON will use Floating Point as storage type.**

**Examples:**

|  |  |
| --- | --- |
| **Address** | **point to** |
| **myvar** | the data named “myvar” in the scope *global* (controller) of the PLC |
| **PROGRAM:myprog.myvar** | the data named “myvar” in the scope of the program named “myprog” |
| **myvar/4** | the bit 4 (the fifth) from the data “myvar” |
| **mystruct.mydata** | the data named “mydata” from the structure named “mystruct” |
| **mytable[6]** | the element 6 (the seventh) of the table “mytable” |
| **mytablestruct[4].mydata[12]** | the element 12 of the table “mydata” from the structure at index 4 of the table “mytablestruct” |

**Table 38: ABLogix address examples**

* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of communication for a given device. As for other Tags, the status Tag ValueName is composed of:

**Status,GlobalDeviceAddress**

* You can define a status Tag for each COM port used.
* If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 39: Tag status meaning**

**Note:** Assisted Edition is enable for this IO server

# eWON IO Server

## Introduction

The eWON IO server is used to interface the eWON INPUTS and OUTPUTS. Depending on your eWON model, you have:

|  |  |  |
| --- | --- | --- |
| **eWON types** | **Flexy 10x - 20x**  **(Base Units)** | **Flexy FLX 3401**  **(Extension Card)** |
| **Digital Inputs** | 2 | 8 |
| **Digital Outputs** | 1 | 2 |
| **Analog Inputs** | 0 | 4 |

**Table 40: eWON Inputs/Outputs**

Additionally, there are a number of Tags that can be addressed with this IO server and which are computed by the eWON IO Server. These additional Tags are used for energy management. In Energy management, the following requirements are taken into accounts:

* **Using digital inputs as counter inputs**
* **Count for a given interval and latch computed result (also save it in historical)**
* **Reject the measurement interval if too long or too short**
* **Adjust the eWON’s Real Time Clock based on a digital input**

These Tags will be computed if the energy module is enabled. There is no topic name to define for the eWON IO server.

## Standard eWON I/O Item Names

### Tag address convention

The following Tags addresses are available for standard eWON Inputs & Outputs access:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **IO Server configuration** | | **Comment** | | |
| **IO Server Name** | **EWON** |  | | |
| **Topic Name** | **Empty** |
| **Item Address** | **DI#** | Digital Input (1) |  | Boolean |
| **CI#** | Counter Input (1) | 0 to 2.147.400.000(2) | Analog |
| **FI#** | Counter Input | 0 to 255 | Analog |
| **LI#** | Latched Counter Input (1) |  | Analog |
| **DO#** | Digital Output (1) |  | Boolean |
| **AI#** | Analog Input (1) |  | Analog |
| **BI#** | Button Input (1) |  | Boolean |

**Table 41: eWON IO server configuration parameters - no topic defined**

**Note:**

The button input (BI1) can be used during normal eWON operations, if it is pressed for more than 4 seconds while the eWON is booting, the flash file system will be erased.

* + - * 1. The number of items depend of the eWON type.For the eWON Flexy range, the numbering of the IOs starts with the embedded IOs of the Base Unit and continues with the first IO Extension Card starting from the left (if available).
        2. From firmware 4.1S3 to 4.3, the CIx value wrapped at 1.000.000. Before 4.1S3, the CIx was misinterpreted as negative when value passes over 231.

|  |  |  |  |
| --- | --- | --- | --- |
| **IO Server configuration** |  | **Comment** | |
|  | **EWON** |  | |
| **Topic Name** | **SYS** |
|  | **SN\_LO** | Serial number, low part (see example of use below) | DWord |
|  | **SN\_HI** | Serial number, high part (see example of use below) | DWord |
|  | **SYS\_UP** | Number of seconds since Power Up | DWord |
|  | **GSM\_REG** | GSM Status: 1: Home network  2: Searching registration  3: Registration denied  4: unknown registration  5: Roaming  100: Not applicable 101: Registration in progress  usually:  1 or 5: means registered Other: not registered | DWord |
|  | **GSM\_LEV** | GSM level (antenna reception) | DWord |

**Table 42: eWON IO server configuration parameters - SYS topic**

Example of use of SN\_LO and SN\_HI items in a Basic program:

|  |
| --- |
| **a% = Int(SN\_HI@ \* 65536) + Int(SN\_LO@)** |
| **Rem Product code** |
| **b% = a% Mod 256** |
| **Rem Sequential number** |
| **c% = Int(a% / 256) Mod 1024** |
| **Rem Week number** |
| **d% = Int(a% / 262144) Mod 64** |
| **Rem Year number** |
| **e% = Int(a% / 16777216) Mod 128** |
| **Print "SN: ";e%;" ";d%;"-";c%;"-"b%** |

For instance, you can define a Tag for the digital input 1 as follows:

***Server name***: *EWON* - ***Topic***: empty - ***Address***: *DI1*

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Syntax** | **Description** | **Automatic Tag type** |
| **eWON500 eWON2001(CD), eWON4001(CD) eWON2101(CD), eWON4101(CD)**  **eWON2005(CD) eWON4005(CD) eWON2104, eWON4104** | **DI1** | Status of input 1 on connector Input/output (bottom side) | Boolean |
| **CI1** | Counter on input 1 on connector Input/output (bottom side) | DWord |
| **LI1** | Latched counter on input 1 on connector Input/output | DWord |
| **DO1** | Digital command of output 1 on connector Input/output | Boolean |
| **BI1** | Status of button (reset) on the front face | Boolean |
| **eWON4002 eWON4102** | **DI1... DI8** | Status of input 1 to 8 on connector DI1-DI8 (top side) | Boolean |
| **CI1... CI8** | Counter on input 1 to 8 on connector DI1-DI8 (top side) | DWord |
| **LI1... LI8** | Latched counter on input 1 to 8 on connector DI1-DI8 (top side) | DWord |
| **DI9** | Status of input 9 on connector Input/output (bottom side) | Boolean |
| **CI9** | Counter on input 9 on connector Input/output (bottom side) | DWord |
| **LI9** | Latched counter on input 9 on connector Input/output (bottom) | DWord |
| **DO1... DO2** | Digital command of relay 1 and 2 on relay connector (bottom side) | Boolean |
| **DO3** | Digital command of output 1 on connector Input/output (bottom) | Boolean |
| **BI1** | Status of button (reset) on the front face | Boolean |
| **AI1... AI4** | Value of analog input 1 to 4 | DWord |
| **AI5... AI6** | Value of analog input 5 to 6 (PT100 probes) | DWord |
| **eWON Flexy 1xx eWON Flexy 2xx** Base Units only. See note below. | **DI1... DI2** | Status of input 1 and 2 on the Base Unit main connector | Boolean |
| **CI1** | Counter on input 1 and 2 on the Base Unit main connector | DWord |
| **LI1** | Latched counter on input 1 and 2 on the Base Unit main connector | DWord |
| **DO1** | Digital command of relay 1 on the Base Unit main connector |  |
| **BI1** | Status of button (reset) on the front face |  |

**Table 43: Available Items depending on eWON product type**

### Basic Principle of the eWON Flexy I/O Tag Addresses

The internal I/O tag addresses of the eWON Flexy always start with the Inputs/Outputs of the Base Unit. Remember that all Base Units feature 1 Digital Output and 2 Digital Inputs, those are the first ones that have to be considered when creating tags in the eWON.

The example on the right helps you to understand the syntax of the I/O tag addresses in the case of 2 I/O Extension Cards.

In the case of multiple I/O cards, the tag numbering sequence is following the left-to-right order of slots. First the I/O Extension Card plugged in the most left slot, then the next on its right and so on. Removing an I/O card other than the utmost right one will result in an internal reallocation of tag addresses that may result in a mismatch between physical I/Os and their software configuration. The software tag addresses can be “frozen” by adding the slot number in the tag definition. This prevents unwanted I/O mismatch (see below).

Slot number append to prevent tag address mismatch:

To prevent any risk of mismatching tag addresses you can append the slot number after the I/O address. The syntax to use is as follows:

**ABx,Ey**

Where AB is the type of I/O (DI, AI, DO) x is the order number

E is a constant prefix to the slot number

y is the slot number of the card (0 = main board, 1 = slot #1, 2 = Slot #2, etc..)

Let’s imagine an eWON Flexy featuring 2 I/O cards FLX 3401 in slots #2 and #3.

The main I/O syntax and order numbering remains unchanged.

In the example above, the I/O server tag addresses are as follows:

Tag Syntax Explanation

DO1,EO Digital Output 1, main board (though E0 is not necessary in this case)

DO2 Digital Output 2, no position specified = second detected DO = first DO on first extension card detected (in slot #2 most left). DO3,E2 Digital Output 3, board in slot #2 = second DO on extension card in slot #2

DO4,E3 Digital Output 4, board in slot #3 = first DO on extension card in slot #3 DO5,E3 Digital Output 5, board in slot #3 = second DO on extension card in slot #3

Behavior if the card in slot #2 is ***removed***: In the ***Tag View*** page,

* DO2 will appear normally as it was not frozen by a slot number append. But it can no longer reflect the status of the first DO of the board in slot #2 that was removed. DO2 now reflects the status of the first DO of the board in slot #3.
* The 3 other DOs with specified slot number E2 and E3 will all appear in error (red cross, value 0), because:

1. The address of DO3,E2 of the card in slot #2 could no longer be found, and
2. The software addresses of DO4,E3 and DO5,E3 of the card in slot #3 do no longer match with the detected order of physical addresses**.** The error messages in the ***Trace*** log file are “Invalid IO Tag name (DOx, addr. DOx,Ex)”.

To have the DOs of the card in slot #3 responding, edit the tags and change their software addresses as follows:

DO4,E3 to DO2,E3 and DO5,E3 to DO3,E3

### Energy configuration with the LIx Tags

The following parameters can be added to the eWON IO server in order to activate the energy support (enter *Energy:1*, validate and the other parameters will be automatically added):

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Default Value** | **Description** | |
| ***Energy*** | **0 or 1** | Enabled or not | Once Energy is defined, the eWON will automatically add the other parameters with default value.  When 0, no energy computation at all are performed, saving eWON CPU resources. |
| ***Debug*** | **0 or 1** | Not automatically added | When :1 all synchronizations are logged in the Real time event log. |
| ***RefTime*** |  | This is the reference time to compute interval. Letting this empty will start at 1/1/1970 00:00. You may enter a date with hour in the form 08/07/2002 11:15:00 to set a new reference (usually not required). | |
| ***IntTime*** | **15** | This is the integration time in MINUTES. | |
| ***IntTolS*** | **5** | This is a tolerance on the integration time in SECONDS. If the measurement interval is shorter or longer than this number of seconds, the period is considered as valid. For example IntTime=15 IntTolS =5 means "interval is valid if between 11.14:55 and 11.15:05". | |
| **SyncIO** | **5** | This is the IO number for clock synchronization. *0* means no synchro available, *1*..*8* is the IO number. Synchronization is based on the counter input associated with the digital input, each time the IO changes the eWON will try to perform a synchronization (see also ***SyncTolS***) | |
| ***SyncTolS*** | **5** | This is the tolerance for accepting the synchronization pulse in SECONDS.  If the synchronization pulse arrives outside the interval, the pulse is rejected. *ERROR REPORT*: the first time the pulse is rejected, an error is logged in the event log. Next errors are not logged (except in Real time log if Debug is enabled); When sync are accepted again, a trace is logged in the event file for the first accepted synchronization.  REM: Because the internal clock has a precision of 1 second, the time is only updated if the absolute value of the offset is greater than 1 (2 or more). Otherwise the offset jitters between -1 and +1 all the time. | |

**Example:**

**Figure 19: eWON IO server default parameters**

#### ENERGY TAGS

The following tags are available for energy management:

|  |  |  |
| --- | --- | --- |
| **LI1..LI8** | Latched input | When the Integration Period expires, the number of pulses counted during the integration period are logged in the corresponding LIx. |
| **ST1..ST4** | **Status variables:** | |
| **ST1** | | This is the absolute time when counter were latched.  Expressed in seconds since 1/1/1970 minus 0x30000000 HEX (or 805306368 decimal). This huge value is subtracted to maintain precision in the Float storage. (\*) |
| **Latch time - 0x30000000** | |
| **ST2** | | 0 means period is valid (within tolerance), 1 means period is invalid. |
| **Period status** | |
| **ST3** | | Length of the period in second. |
| **Period length sec** | |
| **ST4** | | This 32 bit counter counts is increased by 1 after each integration period. It can be used with ONCHANGE to perform operations when period expires. |
| **Period Num** | |

**REM1:** Energy tags are logged after each new period IF Logging is enabled, even if the value has not changed since previous period. **REM2:** Even if Deadband is -1 and Interval is 0. If not, you will have additional points and even maybe duplicate points.

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

#### 32 BITS COUNTER TAGS:

The eWON IO Server provides 8 32 bits (\*) counter tags named: CI1..CI8. These counters are writable but writing in these registers affects the LIx inputs of the Energy IO server module.

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

### Analog Input Value Ranges (eWON4002 & Flexy)

On the eWON4002, the Analog Inputs (AI1 to AI6) are processed by a 12 bit ADC.

The converted values range from 0 to 4095.

On the eWON Flexy Extension Card FLX 3401, the Analog Inputs (AI1 to AI4) are processed by a 16 bit ADC.

The converted values range from 0 to 65535.

#### eWON4002 Configurable analog input AI1 to AI4

* **In 0-10V mode**

|  |  |
| --- | --- |
| **AI#** | **Volt** |
| 0 | 0 |
| 4095 | 10 |

If you need to read the Tag value converted in Volt, you can set a factor of 0.0024420 = 10/4095 with an offset of 0.

**Figure 20: Tag’s configuration: reading the tag value in Volt (eWON4002)**

**In 0-20 mA mode**

The shunt resistor for measuring current is a 220 Ohm.

|  |  |
| --- | --- |
| **AI#** | **Volt** |
| 0 | 0 |
| 1802 | 20 |

If you need to read the Tag value converted in mA, you can set a factor of 0.011099 = 20/1802 with an offset of 0.

**Figure 21: Tag’s configuration: reading the tag value in mA (eWON4002)**

#### eWON4002 PT100 input AI5 and AI6

These two inputs measure a Resistance value in Ohm.

|  |  |
| --- | --- |
| **AI#** | **Ohm** |
| 0 | 162.13 |
| 4095 | 79.625 |

As you can see, the slope is negative, and if you need to read the Tag value converted in Ohm, you can set a factor of -0.02014774 = -82.505/ 4095 with an offset of 162.13.

**Figure 22: Tag’s configuration: reading the tag value in Ohms (eWON4002)**

If you need to convert into °C: temperature = AI \* (-0.0523311) + 161.376. The default range of PT100 in eWON is -50°C to 150°C.

Full range of ADC is: -52.92°C to 161.376°C (79.625 Ohm to 162.13 Ohm).

**Note:**

* eWON 4002 models prior to S/N 0517-xxxx-89 have a PT100 range from 0°C to 131.9°C and must apply the following formulas:

Ohm = IOValue \* (-0.012402) + 150.8

°C = IOValue \* (-0.0322) + 131.9

* eWON 4002 models from S/N 0517-xxxxx-89 to S/N 0521-xxxx-89 have a PT100 range from -56°C to 151°C and must apply following formulas:

Ohm = IOValue \* (-0.0195068) + 158.2

°C = IOValue \* (-0.050677) + 151.2

#### eWON Flexy Extensions Card FLX 3401 analog inputs AI1 to AI4

|  |  |
| --- | --- |
| **AI#** | **Volt** |
| 0 | 0 |
| 65535 | 10 |

If you need to read the Tag value converted in Volt, you can set a factor of 0.0001526 = 10/65535 with an offset of 0.

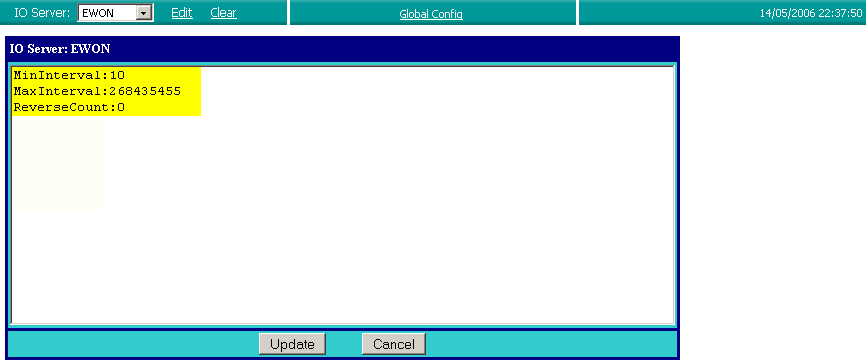
**Figure 23: Tag’s configuration: reading the tag value in Volt (Flexy FLX 3401)**

## Setup

The Energy IO Server is based on the eWON IO Server.

The configuration consists in advanced parameters from the latter IO Server.

The following screen shot shows the standard empty eWON IO Server configuration page.



**Figure 24: eWON IO server configuration page**

### Configuration of the counter pulse length

The pulse length for counter can be configured on all the eWON types. The configuration must be entered in the eWON IO server edit area.

The following parameters are used:

|  |  |
| --- | --- |
| **MinInterval:** | default=10, min=10 |
| **MaxInterval:** | default=268435455, max=268435455 |
| **MinInterval and MaxInterval are entered in milliseconds** | |
| **ReverseCount:** | default=0 |
| The pulse length must be between MinInterval and MaxInterval to be accepted, the measurement resolution is 5 msec (the precision of the pulse length measured is 5 msec) | |
| ReverseCount can be used if pulses entered in the eWON are reversed, reversed means that signal is normally high on the eWON input and it goes low when the pulse occurs. | |

**Table 44: counter pulse length configuration**

**Example of configuration:**

MinInterval:40 MaxInterval:1000 ReverseCount:0

Counts pulse with length longer than 40 msec and shorter than 1000 msec, other pulses are ignored.

# MEM IO Server

## Introduction

The MEMORY IO Server is not a real IO server because values do not come from a peripheral.

Memory Tags (Tags defined with the MEM IO server) are rather sorts of variables that can be modified by a user input or by a BASIC application.

These Tags are very useful for combining different Tags and consider the result as an actual Tag

i.e. having data logging capabilities and alarm management capabilities like for all other Tags.

## Setup

There is no setup for the MEM eWON IO server.

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server configuration** | | |
|  | MEM |  |
|  |  | If no topic is specified, the Tag is a standard memory Tag. Its value is set to 0 when the eWON boots and the Tag is read/write, it can be updated through user actions with script or web pages |
| RET | If topic is set to RET, the Tag is retentive, each change is saved to flash and when the eWON boots, the last value of the Tag is restored. This feature can be convenient to use Tags as parameters. |
| **Item Name** | DON’T CARE | The MEM "Item Name" is insignificant, the TagName is automatically copied in the MEM address by the eWON, the user can change it, but the value will immediately be overwritten again by the TagName.  Data type “Automatic” is equivalent to Floating Point. |

**Table 45: MEM IO server Tag name convention**

**Important:**

**RET values are saved in the FLASH memory, this is not an immediate operation and the FLASH memory can only be written and erased a limited (even if huge 100.000) number of times.**

**This means that you should not use a BASIC script for changing retentive values at very high rate. 1 Tag/Sec. is a maximum AVERAGE rate (occasional higher rates are not prohibited at all).**

# S7-300&400 (MPI/Profibus - ISOTCP)

## Introduction

S73&400 IO Server is dedicated to direct communication with SIEMENS PLC (S7-300 & S7-400). The eWON will connect directly to the PLC’s MPI interface or Profibus DP interface.

**WARNING:**

**The MPI/Profibus interface requires a special serial port hardware!**

**ONLY the eWONs MPI/Profibus are able to communicate in MPI/Profibus**

**eWON product codes like EWxxx6x (i.e.: EW41264 eWON4001 MPI with PSTN modem)**

**eWONs MPI/Profibus have a colored (in light blue) DB9 connector labeled MPI**

**eWONs MPI/Profibus (no CD series) have a Serial Gender-Changer**

**fixed on DB9 connector to provide female DB9 connector.**

**Others Serial IO Server (i.e.: modbus RTU, DF1, UnitelWay)**

**are not available on eWON MPI/Profibus.**

**The ISOTCP (Ethernet) is available on all eWON types.**

With an eWON500 (with a standard Serial Port),

you can have access to a S7-300&400 Siemens PLC by Ethernet thanks to the ISOTCP protocol.

With an eWON500-MPI, you can have access by the MPI port **and** by ISOTCP to your S7-300&400, if it has both interfaces.

**Multi Session:** Since firmware 5.4, the ISOTCP-MPI gateway allows multiple clients to access the same PLC at the same time.

**PPI MultiMaster (for S7-200):** Since firmware 5.5, it is supported by the MPI port of the eWON

## Setup

### COM configuration

**Figure 25: MPI link configuration**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| **Gateway MPI/Profibus Destination Node** | Identifies the PLC MPI/Profibus Destination address when ISOTCP is used |
| **Protocol Type** | Select protocol **MPI or PROFIBUS**.  As the MPI port configuration is shared by the S73&400 and the S7-200 IOServers, this combo box shows also the PPI MULTIMASTER protocol  *PPI MultiMaster and MPI/Profibus are mutually exclusive. It is not allowed to use the S7-200 and the S73&400 IOServers on the MPI port at the same time*  *.* |

**Table 46: MPI/Profibus settings (Part 1)**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| **Baud Rate** | Speed of the MPI port. Available speeds are:   * Disable * 9.6 k Bauds * 19.2 k Bauds * 45.45 k Bauds * 93.75 k Bauds * 187.5 k Bauds * 1.5 M Bauds * 3 M Bauds *(only for eWON CD-MPI)* * 6 M Bauds *(only for eWON CD-MPI)* * 12M Bauds *(only for eWON CD-MPI)* |
| **Reply Timeout** | The maximum time the eWON will wait for a valid MPI message response |
| **MPI Address** | The device address of the eWON on the MPI link (0..126, default is 0) |
| **MPI Highest Station Address** | The highest station address polled by eWON.  Select between 15, 31, 63 or 127. |

**Table 47: MPI/Profibus settings (Part 2)**

### Advanced Routing Setup

Advanced Routing Setup is a feature that has been introduced from firmware version 6.3 onwards.

It allows to access devices that make part of another MPI/Profibus network connected indirectly to the eWON through another (or various other) PLC(s).

Advanced Routing Setup items are visible when the ***Enable*** checkbox is ticked.

**Note**:

Parameters that would have been set in this section remain in memory but are visually hidden AND not taken into account by the firmware when the ***Enable*** checkbox is unticked.

**Figure 26: Advanced MPI configuration screenshot**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Description** | **Default value** |
| **Advanced Routing Setup** | Enable/disable Advanced Routing Setup feature | 0 (disabled) |
| **Gateway/Destination S7-Subnet ID:** | Used when no subnet ID is specified in the ISOTCP message to relay on the MPI bus | 0000-0000 |
| **ISOTCP S7-Subnet ID:** | eWON subnet ID at the ISOTCP interface side | 0000-0000  (invalid subnet ID) |
| **MPI/PROFIBUS S7-Subnet ID:** | eWON subnet ID at the MPI interface side | 0000-0000 |
| **Default Next MPI/PROFIBUS Node:** | MPI node of the PLC towards which the MPI messages will be sent out if the destination subnet ID specified is not the one of the eWON (see previous field), nor one of the ones specified in the gateway entries (see next fields) | 2 |
| **Gateway Entry 1..3 Destination S7-Subnet ID** | MPI messages with destination subnet ID equal to this destination ID will be forwarded to the next MPI/Profibus Node (see next field) | 0000-0000 |
| **Gateway Entry 1..3 Next MPI/Profibus Node ID** | See previous row | 2 |

**Table 48: Advanced MPI/Profibus routing configuration parameter description**

### Topic configuration

**Figure 27: Topic configuration**

Three (3) topics can be used for the IO Server.

These topics are used to give a common property to a group of MPI/ISOTCP Tags like:

* Enable/Disable
* Global Device Address
* Polling Rate

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
|  | Enables or disables polling of all the Tags in the topic. |
| **Global Device Address** | See below for the Device Address Syntax. If an address is specified here, it will replace (overload) the address-defined Tag by Tag. |
| **Poll rate** | Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag names must be refreshed every second - typically for digital input - and other every minute - typically: temperature-. |

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server Name** | S73&400 |  |
| **Topic Name** | A |
| B |
| C |
| **Item Name** | ValueName, Global Device Address | PLC address is defined Tag by Tag |
| ValueName | Topic PLC Address is used. |

**Table 49: S73&400 IO server - Tag name convention table**

The Item Name can contain the PLC address where the value is polled, or not. If the address is also specified at the topic level, the address specified at Tag level will be ignored.

### Value Name

The syntax is the following:

**<Memory Type><Modifier><address>**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Memory Type** | **Modifier** | **Address** |
|  | Data Block number x | B, C, W, S, D, L, F | Byte offset |
| **M** | Internal Memory | B, C, **W**, S, D, L, F | Byte offset |
| **C** | Counter | **W**, S | Object number |
| **T** | Timer | **W**, S | Object number |
| **I** | Discrete Inputs | B, C, **W**, S, D, L, F | Byte offset |
| **Q** | Discrete Outputs | B, C, **W**, S, D, L, F | Byte offset |
| **PI** | Peripheral Inputs | B, C, **W**, S, D, L, F | Byte offset |
| **PQ** | Peripheral Outputs | B, C, **W**, S, D, L, F | Byte offset |

**Table 50: S7 300-400 Memory types**

**Important: All the addresses are always in BYTES (except for Counter and Timer that are objects)**

**Note:** The Modifier can by omitted, the modifier in bold will be used (DB type must have a modifier)**.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Modifier** | **value range** | **Automatic Tag type** |
| **B** | Byte | 0 .. 255 | DWord |
| **C** | signed Byte | -128 .. 127 | Integer |
| **W** | Word | 0 .. 65535 | DWord |
| **S** | signed Word | -32768 .. 32767 | Integer |
| **D** | DWord | 0 .. 4294967296 **(\*)** | DWord |
| **L** | signed DWord | -2147483648 .. 2147483647 **(\*)** | Integer |
| **F** | Float | +/- 3.4e38 | Float |

**Table 51: S7 300-400 Modifiers**

**(\*) Important:** To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.

**Examples**

|  |  |
| --- | --- |
| **Address** | **point to** |
| **MW4** | the Word at address 4 (in bytes) in the Internal Memory |
| **CS1** | the Counter number 1, read it as Signed Word |
| **DB2L5** | the DWord at address 5 (in bytes) in the Data Block 2 |
| **IB3** | the Byte at address 3 (in bytes) in the Discrete Inputs zone |
| **I5#2** | the bit 2 from the Byte (read ’Bit access modifier’ note below) at address 5 in the Discrete Inputs zone |

**Table 52: S7 300-400 register address examples**

* **Bit access modifier:**

In any Memory Type (excluding Counter and Timer), it is possible to access a single bit. A **#x** must be appended to the Value name.

**As the address is always in byte, the Bit index goes only from 0 to 7, no modifier are allowed**

(except B to separate DB block from the address).

The syntax can be used for reading bits and for writing them as well.

If Tag type “Automatic” is used, the type Boolean will be chosen.

**Example:**

**DB1B13#3 represents bit 3 of Byte 13 in DB 1**

**invalid: IW5#2 wrong because there is a Modifier**

**I5#10 wrong because bit number greater than 7**

* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of the communication for a given device.

As for other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* + You can define a status Tag for each PLC used.
  + If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 53: Tag Status meaning**

### Global Device Address

The device address is either appended to the ValueName in the Item Name definition or entered in the Topic global address fields. The device address is composed like following:

|  |  |
| --- | --- |
| **MPI** | |
| **MPI, *MPI node address*** | ***node address***: MPI node address of the PLC    Reach by MPI the PLC with the node address 2 |
| **PROFIBUS, *Profibus node addr*** | ***node address***: PROFIBUS node address of the PLC    Reach by Profibus the PLC with the node address 9 |

**Table 54: MPI/ISOTCP device address syntax (Part 1)**

|  |  |
| --- | --- |
| **MPI** | |
| **MPI, Subnet ID, *node address*** | When the Advanced Routing Setup is ***Enabled***, the following syntax can be applied to reach a device on the MPI network indirectly connected to the eWON (behind one or several other PLCs)    ***Subnet ID:*** ID of the destination S7-Subnet ID  ***node address:*** MPI/Profibus node address of the PLC to reach  Reach by MPI/Profibus the PLC with the node address 4 on the subnet with subnet ID 4859-4565 |
| **ISOTCP** | |
| **ISOTCP, *IP address*, *CalledTSAP*** | ***IP address*** : IP address of the PLC (ex: 10.0.120.204)  ***Called TSAP*** : Transport Service Access Point (ex: 03.02)    Reach by ISOTCP the PLC at IP 10.0.120.204 with the TSAP 03.02 |
| **ISOTCP, *IP address*, *CalledTSAP*, *Subnet ID*, *MPI node address*** | ***IP address*** : IP address of the PLC (ex: 10.0.120.204) ***Called TSAP*** : Transport Service Access Point (ex: 03.02) ***Subnet ID*** : ID of the MPI subnet  ***MPI node address*** : address of the PLC |

**Table 55: MPI/ISOTCP device address syntax (Part 2)**

**Important:**

**If the PLC address is defined at the Topic level, it can be omitted in the Tag definition.**

**In that case the Tag name will only contain the "ValueName".**

**If the PLC address is specified at the Topic level, it will replace any address defined Tag by Tag.**

**Note:**

For an S7-300, it’s generally always *03.02* .

For an S7-400, you have to take into account the slot number of the CPU.

For example if the CPU is on slot 4 rack 0 use TSAP 03.04, if the CPU is on slot 5 rack 1 use TSAP 03,25.

**Note:** Assisted Edition is enable for this IO server

# S7-200 (PPI - ISOTCP)

## Introduction

The S7200 IO Server is dedicated to direct communication with SIEMENS PLC of family S7-200.

**Since firmware 5.7,**

**all eWON types, with serial port or with MPI port, are able to communicate in both PPI modes (MonoMaster and MultiMaster).**

**Then, to communicate with a S7-200 device you could use either an eWON with a serial port or an eWON with an MPI port.**

|  |  |  |
| --- | --- | --- |
| **Usage** | **eWON with SERIAL port (EWxxx0x)** | **eWON with MPI port (EWxxx6x)** |
| **MonoMaster**  eWON is the ONLY Master on the PPI network. | **Yes** | **Yes**  (since firmware 5.5) |
| **MultiMaster**  eWON is not the only Master on the PPI network.  Other masters (HMI or PPI multimaster programming cables) may be present at the same time on the PPI bus. | **Yes**  (since firmware 5.7) | **Yes**  (since firmware 5.5) |

**Table 56: PPI MonoMaster/MultiMaster usage**

Usually, the eWON will be connected directly to the PLC’s PPI interface.

**Since firmware 5.5, eWONs with an MPI port are able to communicate on a PPI MultiMaster bus.**

**Since firmware 5.7, eWONs with a serial port are able to communicate on a PPI MultiMaster bus.**

**The eWON acts always as MASTER.**

**The standard serial port of eWON must be configure in RS-485 (with dipswitch) to communicate in PPI.**

**The ISOTCP (Ethernet) is available on all the eWON types.**

## Setup

### Gateway configuration



**Figure 28: Gateway parameters**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| **Destination PPI Address** | Identifies the PLC PPI Destination address when ISOTCP is used |

### COM configuration (serial port)

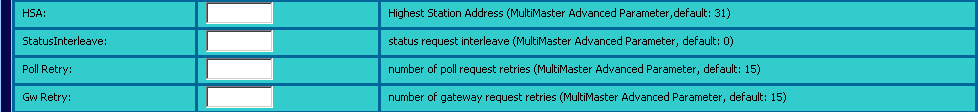
**Figure 29: PPI link configuration**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| ***COM Port*** | Choose your COM port (only on eWON4002) |
| **Baud Rate** | Speed of the PPI port. Available speeds are 9600, 19200 Bauds or disable |
| **Parity** | The parity to apply (None / Even /Odd) |
| **Databits** | Number of bits in data (7 or 8) |
| **Stop bit(s)** | Number of Stop bits (1 or 2) |
| **HW mode** | Choose the Hardware mode of communication: **Half-Duplex** by default or Full Duplex with/without hardware handshaking. |
| **Reply Timeout** | Maximum time the eWON will wait for a valid PPI message response |
| **PPI Address** | Device address of eWON on PPI link (0..126, default is 0) |

**Note:** By default, eWON with Serial port will act as MonoMaster on the PPI link.

#### Serial PPI MultiMaster configuration

To configure the serial port as PPI MultiMaster you need to configure the following parameters.

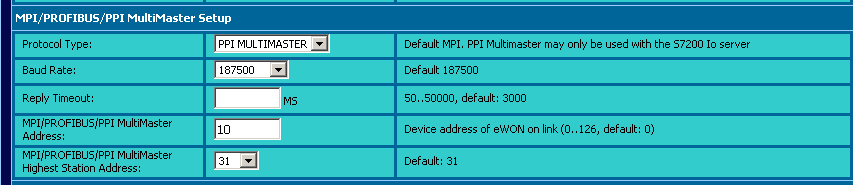


**Figure 30: PPI Multimaster configuration parameters**

|  |  |
| --- | --- |
| **Serial Multimaster Parameter** | **Description** |
| **StatusInterleave** | Number of token messages sent by the eWON before one status request message is sent out. Status messages are used to get the configuration of the network. In the case interleave number is high bandwith is used more efficiently for data exchange but network discovery and recovery is slow.  Value range: 0..20 - default value: 0 (eWON is mono master no exchange of status or token pass messages) |
| **NbrPollRetry** | Number of retrials for one user request (poll = ask for the response after reception of a positive ACK from the PLC) before aborting the request. Useful in case of high PLC load.  Value range: 0..100 (0 means only one poll for one request) - default value: 15 |
| **NbrGwRetry** | Number of retrials for one gateway request (poll = ask for the response after reception of a positive ACK from the PLC) before aborting the request. Useful in case of bus conflict due to high PLC load (reception of lot of NACK).  Value range: 0..100 - default value: 15 |
| **HSA** | Highest station active  value range: 15..127 - default value: 31 |

**Table 57: Serial PPI MultiMaster**

### COM configuration (MPI port)



**Figure 31: MPI link configuration**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| **Protocol Type** | Select protocol **PPI MULTIMASTER**.  As the MPI port configuration is shared by the S7-200 and the S73&400 IOServers, this combo box shows also the MPI & Profibus protocols.  *PPI MultiMaster and MPI/Profibus are mutually exclusive. It is not allowed to use at the same time the S7-200 and the S73&400 IOServers on the MPI port.* |
| **Baud Rate** | Speed of the MPI port. Available speeds are:   * Disable * 9.6 k Bauds * 19.2 k Bauds * 45.45 k Bauds * 93.75 k Bauds * 187.5 k Bauds * 1.5 M Bauds * 3 M Bauds *(only for 2005CD-MPI or 4005CD-MPI)* * 6 M Bauds *(only for 2005CD-MPI or 4005CD-MPI)* * 12 M Bauds *(only for 2005CD-MPI or 4005CD-MPI)* |
| **Reply Timeout** | The maximum time the eWON will wait for a valid MPI message response |
| **PPI MultiMaster Address** | The device address of eWON on PPI link (0..126, default is 0) |
| **PPI Highest Station Address** | The highest station address polled by eWON. Select between 15, 31, 63 or 127. |

**Note:** When using the PPI MultiMaster, the status counters to be checked are the MPI counters and not the PPI counters.

### Topic configuration

**Figure 37: Topic configuration**

Three (3) topics can be used for the IO Server.

These topics are used to give a common property to a group of PPI/ISOTCP Tags like:

* Enable/Disable
* Global Device Address
* Polling Rate

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
|  | Enables or disables polling of all the Tags in the topic. |
| **Global Device Address** | See below for the Device Address Syntax. If an address is specified here, it will replace (overload) the address-defined Tag by Tag. |
| **Poll rate** | Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag names must be refreshed every second - typically for digital input - and other every minute - typically: temperature-. |

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server Name** | S7200 |  |
| **Topic Name** | A |
| B |
| C |
| **Item Name** | ValueName, Global Device Address | PLC address is defined Tag by Tag |
| ValueName | Topic PLC Address is used. |

**Table 58: S7200 IO server - Tag name convention table**

The Item Name can contain the PLC address where the value is polled, or not.

If the address is also specified at topic level, the address specified at Tag level will be ignored.

### Value Name

The syntax is the following:

**<Memory Type><Modifier><address>**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Memory Type** | **Modifier allowed** | **Address** |
|  | Internal Memory | B, C, **W**, S, D, L, F | Byte offset |
| **SM** | Special Memory | B, C, **W**, S, D, L, F | Byte offset |
| **V** | Variable Memory | B, C, **W**, S, D, L, F | Byte offset |
| **C** | Counter | **W**, S | Object number |
| **HC** | High-Speed Counter | D, **L** | Object number |
| **T** | Timer | D, **L** | Object number |
| **AI** | Analog Inputs | **W**, S | Byte offset |
| **AQ** | Analog Outputs | **W**, S | Byte offset |
| **I** | Discrete Inputs | B, C, **W**, S, D, L, F | Byte offset |
| **Q** | Discrete Outputs | B, C, **W**, S, D, L, F | Byte offset |

**Table 59: S7200 Memory types and address scheme**

**Important: All addresses are always in BYTES (except for Counters and Timer that are objects). Note: The Modifier can be omitted, the modifier in bold will be used.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Modifier** | **value range** | **Automatic Tag type** |
| **B** | Byte | 0 .. 255 | DWord |
| **C** | signed Byte | -128 .. 127 | Integer |
| **W** | Word | 0 .. 65535 | DWord |
| **S** | signed Word | -32768 .. 32767 | Integer |
| **D** | DWord | 0 .. 4294967296 (\*) | DWord |
| **L** | signed DWord | -2147483648 .. 2147483647 (\*) | Integer |
| **F** | Float | +/- 3.4e38 | Float |

**Table 60: S7200 Modifiers**

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

**Examples:**

|  |  |
| --- | --- |
| **Address** | **point to** |
| **MW4** | the Word at address 4 (in bytes) in the Internal Memory |
| **HCL1** | the HighSpeed Counter number 1 and read it in Signed DWord |
| **AQW5** | the Word at address 5 (in bytes) in the Analog Output zone |
| **IB3** | the Byte at address 3 (in bytes) in the Discrete Inputs zone |
| **I5#2** | the bit 2 from the Byte (read the ’Bit access modifier’ note below) at address 5 (in bytes) in the Discrete Inputs zone |

**Table 61: S7 200 register address examples**

* **Bit access modifier:**

For Memory Type **M, SM, V, I and Q**, it is possible to access a single bit.

A **#x** must be appended to the Value name.

#### As the address in always in byte, the Bit index goes only from 0 to 7, and no modifiers are allowed.

The syntax can be used for reading bits and for writing them as well.

If Tag type “Automatic” is used, the type Boolean will be chosen.

**Example:**

**AQ10#5 represents bit 5 of Byte 10 in Analog Output zone.**

**Invalid: IW5#2 wrong because there is a Modifier**

**I5#10 wrong because bit number greater than 7**

* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of communication for a given device. As for the other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* + You can define a status Tag for each PLC used.
  + If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 62: Tag Status meaning**

### Global Device Address

The device address is either appended to the ValueName in the Item Name definition or entered in the Topic global address fields. The device address is composed like the following:

|  |  |
| --- | --- |
| **PPI** | |
| **PPI, *PPI node address*** | ***node address***: PPI node address of the PLC    Reach by PPI the PLC with the node address 4 |
| **ISOTCP** | |
| **ISOTCP, *IP address*, *CalledTSAP*** | ***IP address*** : IP address of the PLC (ex: 10.0.120.204)  ***Called TSAP*** : Transport Service Access Point (ex: 4D.57)    Reach by ISOTCP the PLC at IP 10.0.120.204 with the TSAP 4D.57 |

**Table 63: PPI/ISOTCP device address syntax**

#### ISOTCP polling:

You can always use the TSAP 4D.57. This is the default TSAP for a Step 7-Micro/win connection. If you specify your own connection in the S7-200, be sure to:

* Use the same TSAP for the Server and the Client (ex: Local TSAP:12.00, Remote TSAP:12.00)
* Check the "Enable the Keep Alive function for this connection" feature for this connection otherwise the eWON will not be able to poll the device after a connection lost due to the eWON reboot.

**Important:**

**If the PLC address is defined at the Topic level, it can be omitted in the Tag definition.**

**In that case the Tag name will only contain the "ValueName".**

**If the PLC address is specified at the Topic level, it will replace any address defined Tag by Tag.**

**Note:** Assisted Edition is enable for this IO server

# HITACHI IO Server

## Introduction

The Hitachi IOServer allows the eWON to poll tags belonging to a HITACHI EH series PLC using its serial link or Ethernet interface. eWON uses standard H protocol (for serial RS232), station number H protocol (for serial RS485), and Ethernet H-protocol (for Ethernet).

No remote maintenance functionality is provided by this IO server.

## Setup

### COM configuration

**Figure 32: HITACHI IOServer: COM setup**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| **Baud Rate** | Speed of the Serial port.  Available speeds are 1200, 2400, 4800, 9600, 19200 Bauds or disabled |
| **Parity** | The parity to apply (None / Even / Odd) |
| **Databits** | The number of bits in data (7 or 8) |
| **Stop bit(s)** | The number of Stop bits (1 or 2) |
| **HW mode** | Choose the Hardware mode of communication:   * **Full Duplex HW handshaking (by default)** * **Full Duplex NO handshaking** * **Half duplex** |
| **Reply Timeout** | Maximum time the eWON will wait for a valid message response |

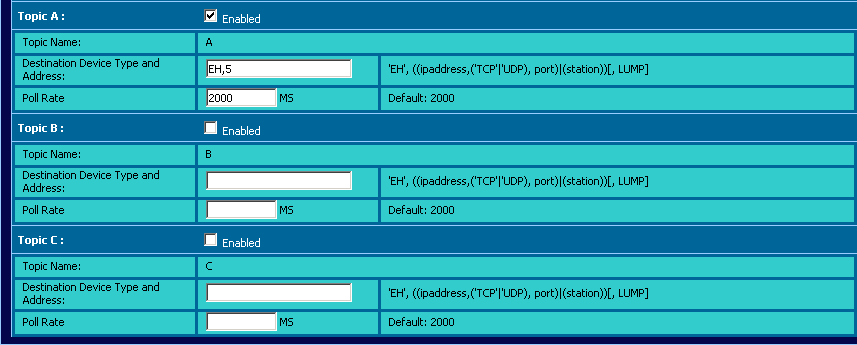
**Table 64: HITACHI IOServer: COM parameters**

The eWON implements Transmission Control Procedure 1 (One-way activation Control Procedure).

When half-duplex mode is used (RS485), the eWON acts as master.

So, no other master may be on the bus (eg: control panel or host).

### Topic configuration



**Figure 33: HITACHI IOServer: Topic configuration**

Three (3) topics can be used for the IO Server.

These topics are used to give a common property to a group of Hitachi Tags like:

* Enable/Disable
* Global Device Address
* Polling Rate

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
|  | Enables or disables polling of all the Tags in the topic. |
| **Global Device Address** | See below for the Device Address Syntax. If an address is specified here, it will replace (overload) the address-defined Tag by Tag. |
| **Poll rate** | Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag names must be refreshed every second - typically for digital input - and other every minute - typically: temperature-. |

**Table 65: HITACHI IOServer: Topic configuration item definition**

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server Name** | HITACHI | |
| **Topic Name** | A | |
| B | |
| C | |
| **Item Name** | ValueName, Global Device Address | PLC address is defined Tag by Tag |
| ValueName | Topic PLC Address is used. |

**Table 66: HITACHI IOserver - Tag name convention table**

The Item Name can contain the PLC address where the value is polled or not.

If an address is also specified at topic level, the address specified at Tag level will be ignored.

### Value Name

The syntax is the following:

**<Memory Type>[<Modifier>]<address>**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Memory type** | **Modifier allowed (optional)** | **Address** |
| **X** | External Input Bit |  | **Hrusbb**  H stands for hexadecimal  r: remote number (1 digit hexadecimal) u: unit number (1digit hexadecimal)  s: slot number (1 digit hexadecimal)  bb: bit offset (00 ..FF 2 digits hexadecimal) |
| **WX** | External Input Word | **W**, S | **Hrusw**  H stands for hexadecimal  r: remote number (1 digit hexadecimal) u: unit number (1digit hexadecimal)  s: slot number (1 digit hexadecimal) w: word offset (1digit hexadecimal) |
| **DX** | External Input Double | **D**, L, F | **Hrusw**  H stands for hexadecimal  r: remote number (1 digit hexadecimal) u: unit number (1digit hexadecimal)  s: slot number (1 digit hexadecimal) w: word offset (1digit hexadecimal) |
| **Y** | External Output Bit |  | **Hrusbb**  H stands for hexadecimal  r: remote number (1 digit hexadecimal) u: unit number (1 digit hexadecimal)  s: slot number (1 digit hexadecimal)  bb: bit offset (00..FF: 2 digits hexadecimal) |
| **WY** | External Output Word | **W**, S | **Hrusw**  H stands for hexadecimal  r: remote number (1 digit hexadecimal) u: unit number (1digit hexadecimal)  s: slot number (1 digit hexadecimal) w: word offset (1digit hexadecimal) |
| **DY** | External Output Double | **D**, L, F | **Hrusw**  H stands for hexadecimal  r: remote number (1 digit hexadecimal) u: unit number (1digit hexadecimal)  s: slot number (1 digit hexadecimal) w: word offset (1digit hexadecimal) |
| **R** | Internal Output Bit dedicated |  | **Hxxxxxx**  H stands for hexadecimal  x: 1 to 6 digits hexadecimal indicating bit offset |
| **WR** | Internal Output Word dedicated | **W**, S | **Hxxxxxx**  H stands for hexadecimal  x: 1 to 6 digits hexadecimal indicating word offset |
| **DR** | Internal Output Word dedicated Double | **D**, L, F | **Hxxxxxx**  H stands for hexadecimal  x: 1 to 6 digits hexadecimal indicating word offset |
| **M** | Internal Bit Output Common |  | **Hxxxxxx**  H stands for hexadecimal  x: 1 to 6 digits hexadecimal indicating bit offset |
| **WM** | Internal Word Output Common | **W**, S | **Hxxxxxx**  H stands for hexadecimal  x: 1 to 6 digits hexadecimal indicating word offset |
| **DM** | Internal Word Output Common Double | **D**, L, F | **Hxxxxxx**  H stands for hexadecimal  x: 1 to 6 digits hexadecimal indicating word offset |

**Table 67: HITACHI Memory types and address scheme (Part 1)**

|  |  |  |  |
| --- | --- | --- | --- |
| **L** | CPU Link Bit |  | **Hxxxxxx**  H stands for hexadecimal  x: 1 to 6 digits hexadecimal indicating bit offset |
| **WL** | CPU Link Word | **W**, S | **Hxxxxxx**  H stands for hexadecimal  x: 1 to 6 digits hexadecimal indicating word offset |
| **DL** | CPU Link Double | **D**, L, F | **Hxxxxxx**  H stands for hexadecimal  x: 1 to 6 digits hexadecimal indicating word offset |

**Table 67: HITACHI Memory types and address scheme (Part 2)**

**Note:** The Modifier can by omitted, the modifier in bold will be used.

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Modifier** | **value range** | **Automatic Tag type** |
| **W** | Word | 0 .. 65535 | DWord |
| **S** | signed Word | -32768 .. 32767 | Integer |
| **D** | DWord | 0 .. 4294967296 (\*) | DWord |
| **L** | signed DWord | -2147483648 .. 2147483647 (\*) | Integer |
| **F** | Float | +/- 3.4e38 | Float |

**Table 68: HITACHI Modifiers**

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

**Examples**

|  |  |
| --- | --- |
| **Address** | **point to** |
| **XH00103** | selection of bit3 of word 0 of the input device located at remote:0 unit:0 slot:1 |
| **WYH0011** | selection of word 0 of the output device located at at remote:0 unit:0 slot 1 |
| **RH105** | bit 105 hex (261 decimal) of the Internal Output Bit Dedicated area |
| **WRSH10** | short located at word 10 hex of the WR area |
| **WMH10** | word 10 hex of the Internal Word Output Common area |
| **DMH10** | double word 10 + 11 hex of the Internal Word Output Common area |
| **WMH000010** | word 10 hex of the Internal Word Output Common area (WMH10 is more optimal) |

**Table 69: HITACHI register address examples**

* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of communication for a given device. As for the other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* You can define a status Tag for each PLC used.
* If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 70: Tag Status meaning**

### Global Device Address

The device address is either appended to the ValueName in the Item Name definition, or entered in the Topic global address fields.

The device address syntax is the following:

**EH[,[[ipaddress,TCP|UDP,port]|[station]][,LUMP]]**

|  |  |
| --- | --- |
| **EH Serial** | |
| **EH,*station*** | **station** : address of the PLC on the network  *EH,4 the eWON will access PLC number 4.* |
| **EH** | By configuration, an HITACHI PLC could have an undefined station number.  If this PLC is alone on a serial link, you don’t need to specify a station number. |
| **EH,*station,LUMP*** | **station** : address of the PLC on the network  **LUMP** : Link Unit Port Module  *EH,4,FFFF0000 the eWON will access PLC number 4 with the LUMP FFFF0000.* |
| **EH,*LUMP*** | By configuration, an HITACHI PLC could have an undefined station number.  If this PLC is alone on a serial link, you don’t need to specify a station number.  **LUMP** : Link Unit Port Module  *EH,FFFF0000 the eWON will access PLC with the LUMP FFFF0000.* |
| **EH ethernet** | |
| **EH,ipaddress,TCP|UDP,port** | **ipaddress** : IP address of the PLC **TCP|UDP** : select the protocol used **port** : IP port used  *EH,10.0.120.203,TCP,3004 the eWON will access the PLC at 10.0.120.203 with protocol TCP and port 3004* |
| **EH,ipaddress,TCP|UDP,port,LUMP** | **ipaddress** : IP address of the PLC **TCP|UDP** : select the protocol used **port** : IP port used  **LUMP** : Link Unit Port Module  *EH,10.0.120.203,TCP,3004,FFFF0000*  *the eWON will access the PLC at 10.0.120.203, protocol TCP, port 3004 and LUMP FFFF0000* |

**Table 71: HITACHI device address syntax**

**Important:**

**If the PLC address is defined at the Topic level, it can be omitted in the Tag definition.**

**In that case the Tag name will only contain the "ValueName".**

**If the PLC address is specified at the Topic level, it will replace any address defined Tag by Tag.**

**Note:** Assisted Edition is enable for this IO server

# MITSUBISHI FX IO Server

### Introduction

This MITSUFX IOServer allows the eWON to poll tags belonging to a MITSUBISHI FX series PLC using its RS-422 base unit programming connector. No remote maintenance functionality is provided by this IO server.

The VCOM feature of the eWON has to be used to cover a remote maintenance.

MITSUFX IOServer is available in eWON with Firmware 5.5 and above.

## Setup

### COM configuration

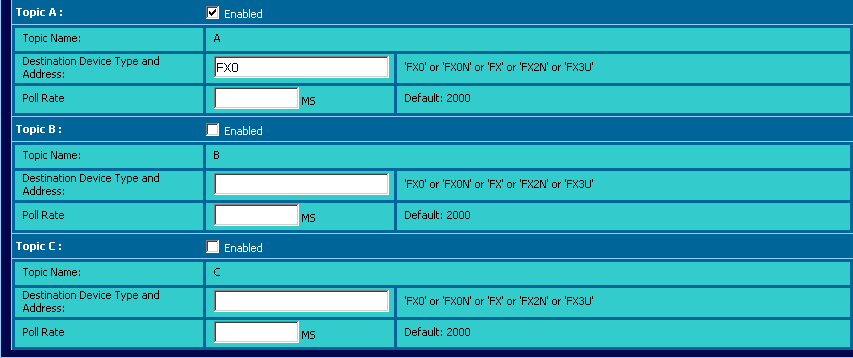
**Figure 34: MITSUFX IOServer: COM setup**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| **Baud Rate** | Speed of the Serial port.  Available speeds are from 9600 until 115200 Bps or disabled |
| **Parity** | The parity to apply (None / Even / Odd) |
| **Databits** | The number of bits in data (7 or 8) |
| **Stop bit(s)** | The number of Stop bits (1 or 2) |
| **HW mode** | Choose the Hardware mode of communication:   * **Full Duplex HW handshaking** * **Full Duplex NO handshaking (by default)** * **Half duplex** |
| **Reply Timeout** | The maximum time the eWON will wait for a valid message response |

**Table 72: MITSUFX IOServer: COM parameters**

**Note:** The switches of the eWON must be configured in RS485/RS422 mode.

### Topic configuration



**Figure 35: MITSUFX IOServer: Topic configuration**

Three (3) topics can be used for the IO Server.

These topics are used to give a common property to a group of Mitsubishi Tags like:

* Enable/Disable
* Global Device Address
* Polling Rate

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
|  | Enables or disables polling of all the Tags in the topic. |
| **Global Device Address** | Select the type of Mitsubishi PLC to link. Available FX type are:   * **FX** * **FX0** * **FX0N** * **FX2N** * **FX3U**   There is no address (number) to add because the eWON is connected directly to one PLC by its programming port.  If a device is specified here, it will replace (overload) the device-defined Tag by Tag. |
| **Poll rate** | Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag names must be refreshed every second - typically for digital input - and other every minute - typically: temperature-. |

**Table 73: MITSUFX IOServer: Topic configuration item definition**

## Tag name convention

|  |  |  |  |
| --- | --- | --- | --- |
| **IO Server Name** | MITSUFX | | |
| **Topic Name** | | A | |
| B | |
| C | |
| **Item Name** | ValueName, Global Device Address | PLC address is defined Tag by Tag | |
| ValueName | Topic PLC Address is used. | |

**Table 74: MITSUFX IOserver - Tag name convention table**

The Item Name can contain the PLC address where the value is polled, or not.

If the address is also specified at topic level, the address specified at Tag level will be ignored.

### Value Name

The syntax is the following:

**<Memory Type Symbol>[<Modifier>]<address>**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Memory type** | **Modifier allowed (optional)** | **Address** |
|  | Input Bit (Boolean) |  | 1 to 3 **OCTAL** digits |
| **Y** | Output Bits (Boolean) |  | 1 to 3 **OCTAL** digits |
| **M** | Auxiliary relays (Boolean) |  | 1 to 4 decimal digits (max 7999) |
| **M** | Special Auxiliary relays (Boolean) |  | 1 to 4 decimal digits (min 8000) |
| **S** | States (Boolean) |  | 1 to 4 decimal digits |
| **TC** | Timer Contacts (Boolean) |  | 1 to 3 decimal digits |
| **CC** | Counter Contacts (Boolean) |  | 1 to 3 decimal digits |
| **TR** | Timer Reset (Boolean) |  | 1 to 3 decimal digits |
| **CR** | Counter Reset (Boolean) |  | 1 to 3 decimal digits |
| **T** | Timer Value | **W**, S | 1 to 3 decimal digits |
| **C** | Counter Value | **W**, S | 1 to 3 decimal digits (max 199) |
| **C** | High Speed Counter Value | L, **D** | 1 to 3 decimal digits (min 200) |
| **D** | Data Registers | **S**, W, L, D, F | 1 to 4 decimal digits (max 7999) |
| **D** | Special Data Registers | **S**, W, L, D, F | 1 to 4 decimal digits (min 8000) |

**Table 75: MITSUFX Memory types and address scheme**

**Note:** The Modifier can be omitted, the modifier in bold will be used.

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Modifier** | **value range** | **Automatic Tag type** |
| **W** | Word | 0 .. 65535 | DWord |
| **S** | signed Word | -32768 .. 32767 | Integer |
| **D** | DWord | 0 .. 4294967296 (\*) | DWord |
| **L** | signed DWord | -2147483648 .. 2147483647 (\*) | Integer |
| **F** | Float | +/- 3.4e38 | Float |

**Table 76: MITSUFX Modifiers**

**(\*) Important: To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.**

**Examples**

|  |  |
| --- | --- |
| **Address** | **point to** |
| **X14** | input bit at octal address 14 |
| **D3** | data register at address 3 (read as Signed Word) |
| **DD3** | data register at address 3 (read as DWord) |
| **DF3** | data register at address 3 (read as Float) |
| **D8010** | special data register at address 8010 (read as Signed Word) |
| **C199** | counter value at address 199 (read as Word) |
| **C200** | high speed counter at address 200 (read as DWord) |

**Table 77: MITSUFX register address examples**

* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of communication for a given device.

As for the other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* + You can define a status Tag for each PLC used.
  + If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 78: Tag Status meaning**

**Note:** Assisted Edition is enable for this IO server

# SNMP IO Server

## Introduction

The SNMP IO server (Simple Network Management Protocol) is used to communicate with any SNMPv1 device on the network.

It allows both read and write operations on 32-bit objects. It supports access to several different devices at the same time.

The following object types are supported: INTEGER, 32bits COUNTER, GAUGE, UNSIGNED, TIMETICK and extracted FLOAT.

There is no handling of MIB files.

SNMP IOServer is available in eWON with Firmware 5.6s2 and above.

## Setup

**Figure 36: SNMP IOServer: setup**

Three (3) topics can be used for the IO Server.

These topics are used to give a common property to a group of SNMP Tags like:

* Enable/Disable
* Global Device Address
* Polling Rate

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
| **Topic enabled** | Enables or disables polling of all the Tags in the topic. |
| **Global Device Address** | Select the target SNMP Server device. Syntax:   * **IP\_Addr,Read\_Community,Write\_community IP\_Addr:** IP address of the SNMP Server **Read\_Community:** Community used for Reading **Write\_Community:** Community used for Writing * **IP\_Addr,Community**   **IP\_Addr:** IP address of the SNMP Server  **Community:** Community used for Reading and Writing  If an address is specified here, it will replace (overload) the address-defined Tag by Tag. |
| **Poll rate** | Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag names must be refreshed every second - typically for digital input - and other every minute - typically: temperature-. |

**Table 79: SNMP IOServer: Topic configuration item definition**

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server Name** | SNMP | |
| **Topic Name** | A | |
| B | |
| C | |
| **Item Name** | ValueName, Global Device Address | PLC address is defined Tag by Tag |
| ValueName | Topic PLC Address is used. |

**Table 80: SNMP IOserver - Tag name convention table**

The Item Name can contain the Device address where the value is polled, or not.

If the address is also specified at topic level, the address specified at Tag level will be ignored.

### Value Name

As the SNMP use an Abstract Syntax Notation (ASN), encode the full OID (Object ID) of the data you want to Read/Write.

You cannot import MIB files into the eWON.

INTEGER, 32bits COUNTER, GAUGE, UNSIGNED, TIMETICK and extracted FLOAT (see Note below) are supported

**For all integer objects (not for the extracted FLOAT), use the Integer type to save them.**

**Choosing the Automatic type will use the Integer type. Otherwise, they could be truncated when eWON will store them as Float).**

**(\*) Note**: *Since firmware 6.4s4 the SNMP IOServer can read STRING-type OID and converts them in FLOAT. If, for example, a tag contains the string [AnyText 6.78], the extracted result will be [6.78] while eWON converts it by keeping only the number part of the string. As FLOAT does not exist in SNMP, this is a work around to this limitation. This works in read only, not in write.*

**Examples**

|  |  |
| --- | --- |
| **Address** | **point to** |
| **.1.3.6.1.2.1.4.3.0** | read the .iso.org.dod.internet.mgmt.mib-2.ip.ipInReceives object |
| **.1.3.6.1.2.1.4.3.0,10.0.0.55,public,private** | read the .iso.org.dod.internet.mgmt.mib-2.ip.ipInReceives object at the IP  10.0.0.55 using community "public" for reading and the community "private" for writing. |

**Table 81: SNMP address examples**

**IMPORTANT:**

**The address of SCALAR objects ends with a 0 while the address of array items ends with their index in the array (starting from 1).**

For example, to read *.iso.org.dod.internet.mgmt.mib-2.ip.ipInReceives* that have the address ".1.3.6.1.2.1.4.3",

you must add a ".0" at the end to obtain ".1.3.6.1.2.1.4.3.0".

And to read *.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifMtu* (".1.3.6.1.2.1.2.2.1.4") of the second interface,

because this object is a table, you must add ".2" at the end to obtain ".1.3.6.1.2.1.2.2.1.4.2".

**NOTE:**

We recommend you to check the SNMP address with an SNMP Client (like Getif <http://www.wtcs.org/snmp4tpc/getif.htm)>to find the right address.

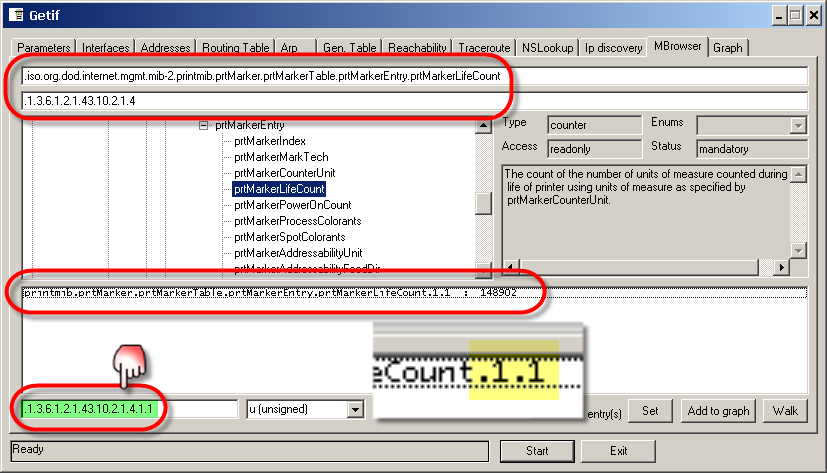
**Example:**

Searching in the MIB file of a network printer to find the address of the "TotalNumberOfPagePrinted",

we found :*.iso.org.dod.internet.mgmt.mib-2.printmib.prtMarker.prtMarkerTable.prtMarkerEntry.prtMarkerLifeCount* at address ".1.3.6.1.2.1.43.10.2.1.4"

Using the MIB file with Getif to retrieve the whole address of this Counter,

we see that the Counter is in a 2 dimensions table and we need to add ".1.1" to get its value.



* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of communication for a given device.

As for the other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* You can define a status Tag for each Device used.
* If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 82: Tag Status meaning**

**Note:** Assisted Edition is enable for this IO server

# MITSUBISHI MELSEC IO Server

## Introduction

This MELSEC IOServer allows the eWON to poll tags belonging to the MITSUBISHI Q-Series PLC using the protocol on the Ethernet interface. To cover remote maintenance for the Mitsubishi PLCs, the Ethernet routing feature or VCOM feature (serial) of the eWON have to be used.

The MELSEC IOServer is available from firmware version 6.3 onwards.

Since firmware 7.0 eWON also integrates the E1 frames of the MC protocol to allow polling the FX3U and A-Series PLC’s through their Ethernet interface

## Setup

### Topic configuration

**Figure 37: MELSEC IOServer: Topic configuration**

Three (3) topics can be used for the IO Server.

These topics are used to give a common property to a group of Mitsubishi Tags like:

* Enable/Disable
* Global Device Address
* Polling Rate

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
| **Topic enabled** | Enables or disables polling of all the Tags in the topic. |
| **Global Device Address Q-Series**  QCPU,QnACPU & QOCPU | **Syntax**:  Model,Protocol,IP Address,Port Number,Network Number,PC Number [,ReqDestModuleIONo,ReqDestModuleStationNo], where:   * **Model**: QCPU,QnACPU or QOCPU * **Protocol**: TCP, UDP * **IP Address**: is the IP address of the PLC * **Port Number**: is the Port of the PLC on which it is listening for MC protocol * **Network Number**: network number of the target (1..239) (0) * **PC Number**: Target Station Number (1..64) (FF) * Optional **ReqDestModulelONo** default FF 03 * Optional **ReqDestModuleStationNo** default 0   **Example**: QCPU,UDP,198.162.120.155,5000,1,1 |

|  |  |
| --- | --- |
| **Topic configuration item** | **Description** |
| **Global Device Address A-Series**  ACPU | **Syntax**:  Model,Protocol,IP Address,Port Number,PC Number where:   * **Model**: ACPU * **Protocol**: TCP, UDP * **IP Address**: is the IP address of the PLC * **Port Number**: is the Port of the PLC on which it is listening for MC protocol * **Network Number**: network number of the target (1..239) (0)   **Example**: ACPU,UDP,198.162.120.155,5000,1 |
| **Global Device Address FX-Series**  FX3UCPU | **Syntax**:  Model,Protocol,IP Address,Port Number where:   * **Model**: FX3UCPU * **Protocol**: TCP, UDP * **IP Address**: is the IP address of the PLC * **Port Number**: is the Port of the PLC on which it is listening for MC protocol   **Example**: FX3UCPU,UDP,198.162.120.155,5000 |
| **Poll rate** | Defines the refresh rate of the Tag name. In a complex application, we can imagine that some Tag names must be refreshed every second - typically for digital input - and other every minute - typically: temperature-. |

**Table 83: MELSEC IOServer: Topic configuration item definition**

Temporary hidden table

**Examples of Global Device Address syntax**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CPU** | **Protocol** | **IP Address** | **Port #** | **Network ID**  **of PLC** | **Network ID of PC** | **IOServer settings** |
| Q | TCP | 192.168.140.10 | 5000 | 1 | 5 | QCPU, TCP,192.168.140.10, 5000,1,5 |
| Q | UDP | 192.168.140.10 | 5000 | 1 | 5 | QCPU, UDP,192.168.140.10, 5000,1,5 |
|  |  |  |  |  |  |  |
| QNA | TCP | 192.168.140.10 | 5000 | 1 | 5 | QNACPU, TCP,192.168.140.10, 5000,1,5 |
| QNA | UDP | 192.168.140.10 | 5000 | 1 | 5 | QNACPU, UDP,192.168.140.10, 5000,1,5 |
|  |  |  |  |  |  |  |
| QO | TCP | 192.168.140.10 | 5000 | 1 | 5 | QOCPU, TCP,192.168.140.10, 5000,1,5 |
| QO | UDP | 192.168.140.10 | 5000 | 1 | 5 | QOCPU, UDP,192.168.140.10, 5000,1,5 |
|  |  |  |  |  |  |  |
| A | TCP | 192.168.1.70 | 1284 | None | 255 | ACPU, TCP,192.168.1.70, 1284, 255 |
| A | UDP | 192.168.1.70 | 1284 | None | 255 | ACPU, UDP,192.168.1.70, 1284, 255 |
|  |  |  |  |  |  |  |
| FX3U | TCP | 192.168.140.10 | 5001 | None | None | FX3UCPU, TCP,192.168.140.10, 5001,1,5 |
| FX3U | UDP | 192.168.140.10 | 5002 | None | None | FX3UCPU, UDP,192.168.140.10, 5002,1,5 |

## Tag name convention

|  |  |  |
| --- | --- | --- |
| **IO Server Name** | MELSEC | |
| **Topic Name** | A | |
| B | |
| C | |
| **Item Name** | ValueName, Global Device Address | PLC address is defined Tag by Tag |
| ValueName | Topic PLC Address is used. |

**Table 84: MITSUFX IOserver - Tag name convention table**

The Item Name can contain the PLC address where the value is polled, or not.

If the address is also specified at topic level, the address specified at Tag level will be ignored.

### Value Name

The syntax is the following:

**<Memory Type Symbol>[<Modifier>]<address>**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Memory type** | **Modifier allowed (optional)** | **Address** |
|  | Input |  | 1 to 6 hexadecimal bits |
| **DX** | Direct Input |  | 1 to 6 hexadecimal bits |
| **Y** | Output |  | 1 to 6 hexadecimal bits |
| **DY** | Direct Output |  | 1 to 6 hexadecimal bits |
| **B** | Link Relay |  | 1 to 6 hexadecimal bits |
| **SB** | Special Link Relay |  | 1 to 6 hexadecimal bits |
| **M** | Internal Relay |  | 1 to 6 decimal digits |
| **SM** | Special Internal Relay |  | 1 to 6 decimal digits |
| **L** | Latch Relay |  | 1 to 6 decimal digits |
| **F** | Annunciator Relay |  | 1 to 6 decimal digits |
| **V** | Edge Relay |  | 1 to 6 decimal digits |
| **S** | Step relay |  | 1 to 6 decimal digits |
| **TS** | Timer Contact |  | 1 to 6 decimal digits |
| **TC** | Timer Coil |  | 1 to 6 decimal digits |
| **TN** | Timer Value | **W**, S | 1 to 6 decimal digits |
| **SS** | Integrating Timer Contact |  | 1 to 6 decimal digits |
| **SC** | Integrating Timer Coil |  | 1 to 6 decimal digits |
| **SN** | IntegratingTimer Value | **W**, S | 1 to 6 decimal digits |
| **CS** | Counter Contact |  | 1 to 6 decimal digits |
| **CC** | Counter Coil |  | 1 to 6 decimal digits |
| **CN** | Counter Value | **W**, S | 1 to 6 decimal digits |
| **D** | Data Register | **S**, W, L, D, F | 1 to 6 decimal digits |
| **SD** | Special Data Register | **S**, W, L, D, F | 1 to 6 decimal digits |
| **W** | Link Register | **S**, W, L, D, F | 1 to 6 decimal digits |
| **SW** | Special Link Register | **S**, W, L, D, F | 1 to 6 decimal digits |
| **R** | File Register | **S**, W, L, D, F | 1 to 6 decimal digits |
| **Z** | Index Register | **S**, W, L, D, F | 1 to 6 decimal digits |

**Table 85: MELSEC Memory types and address scheme**

**Note:** The Modifier can be omitted, the modifier in bold will be used**.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Modifier** | **value range** | **Automatic Tag type** |
| **W** | Word | 0 .. 65535 | DWord |
| **S** | signed Word | -32768 .. 32767 | Integer |
| **D** | DWord | 0 .. 4294967296 (\*) | DWord |
| **L** | signed DWord | -2147483648 .. 2147483647 (\*) | Integer |
| **F** | Float | +/- 3.4e38 | Float |

**Table 86: MELSEC Modifiers**

**(\*) Important:** To avoid loss of precision due to Integer to float conversion, choose the right storage DataType for your Tag.

**Examples**

|  |  |
| --- | --- |
| **Address** | **point to** |
| **X14** | input bit at octal address 14 |
| **D3** | data register at address 3 (read as Signed Word) |
| **DD3** | data register at address 3 (read as DWord) |
| **DF3** | data register at address 3 (read as Float) |
| **D8010** | special data register at address 8010 (read as Signed Word) |

**Table 87: MELSEC register address examples**

* **Status register:**

The STATUS Tag is a special Tag that returns information about the current state of communication for a given device.

As for the other Tags, the status Tag ValueName is composed of:

**Status, Global Device Address**

* You can define a status Tag for each PLC used.
* If you use the status address, the Tag must be configured as analog.

|  |  |
| --- | --- |
| **0** | Communication not initialized. Status UNKNOWN.  If no Tag is polled on that device address, the communication status is unknown. |
| **1** | Communication OK. |
| **2** | Communication NOT OK. |

**Table 88: Tag Status meaning**

**Note:** Assisted Edition is enable for this IO server