PLCs, Software,
Conveyor Controls

JK Buckston, BL. 10
1618 Sofia BULGARIA
Phone: (+359 2) 975 11 80
Fax: (+359 2) 975 11 81
E-mail: indsoft@einet.bg
www.indsoft.bg

Connecting ConveyLinx-Ai/Ai2 modules to Siemens S7 PLCs using PROFINET IO

Rev 1.7

December 2019

Glossary of Terms

ConveyLinx-Ai	Two zone conveyor control module with two M8 MDR connectors.
ConveyLinx-Ai2	Two zone conveyor control module with two M8 MDR connectors. Special device chassis design, developed to fit into narrow conveyor frames. From a Profinet point of view ConveyLinx-Ai and ConveyLinx-Ai2 are identical.
M8	This is the type of a particular connector, which has four connector pins and is used on the **ConveyLinx Ai** modules for both sensor connectors and MDR connectors.
DAP	**D**evice **A**ccess **P**oint – The PROFINET term for an interface to a device. Multiple DAPs can be defined for a single physical device.
MDR	**M**otorized **D**rive **R**oller or **M**otor **D**riven **R**oller - Brushless DC motor and gearbox assembly integrated into a single conveyor roller.
PLC	**P**rogrammable **L**ogic **C**ontroller – A wide variety of industrial computing devices that control automatic equipment
GSD	**G**eneral **S**tation **D**escription –the PROFINET device description
GSDML	GSD file written in the XML language
UDT	**U**ser **D**ata **T**ype – this is a vendor pre-prepared data structure, describing the data that is exchanged by the PROFINET device and the PLC. It can be imported into the engineering environments.
TIA Portal	The newer Siemens engineering environment.
STEP 7	The older Siemens engineering environment.
RT data	The cyclic real-time data exchanged between the PLC and the ConveyLinx-Ai
PROFINET	Industrial communication protocol.
PROFINET PLC I/O	A remote device working as an I/O device to a PLC and communicating with the PLC via PROFINET.
PLC I/O mode	A mode of operation of ConveyLinx-Ai, in which the device does not execute any control logic and is simply an I/O for the connected PLC.
ZPA mode	**Z**ero **P**ressure **A**ccumulation. A mode of operation of ConveyLinx-Ai, in which the device does execute its own control logic. The PLC can connect to the ConveyLinx-Ai and influence its operation.
Logix mode	A mode of operation of ConveyLinx-Ai, in which the device does not execute its own control logic, but has a ConveyLogix PLC program running inside.
S-7 PLC	Siemens PLCs
LLDP	**L**ink **L**ayer **D**iscovery **P**rotocol – an auxiliary protocol, used to gather the topological information of the neighbor devices.
DCE/RPC	**D**istributed **C**omputing **E**nvironment / **R**emote **P**rocedure **C**alls – this is the protocol used by the PLC to establish connection with a slave I/O device.
DCP	**D**iscovery and basic **C**onfiguration **P**rotocol – the protocol used for network discovery of modules.

EasyRoll	Engineering tool for configuration of all ConveyLinx-family devices. Created and distributed by Industrial Software
PGD	Pulse Gear Drive. Basically a combination of a Senergy motor and a gearbox, without the roller.

.

Table of Contents

1. Basics

The S7 PLCs from Siemens can use ConveyLinx-Ai modules as remote I/O. ConveyLinx-Ai modules support PROFINET IO communication protocol and can act as PROFINET IO-devices (S7 PLCs act as PROFINET IO controllers).

1.1. There are two distinct ways to use the ConveyLinx-Ai module in a ProfiNet system:

- **With EasyRoll configuration**
  - The ConveyLinx-Ai module should be configured from the EasyRoll. During the configuration the module forms its ProfiNet name based on its mode and IP address. After the module is configured, then the PLC can connect to the module and fully control the module (when in PLC control mode) or influence its operation (when in stand-alone ZPA mode). **It is important to be aware, that the module has formed its name and that exact name must be entered in the project.** The PLC cannot discover the module, unless the names match (the project name and the real ConveyLinx-Ai name). The IP addresses must also match, since the IP address is included in the name of the module. The PLC can change the subnet mask and the gateway (if needed).
- **With topological discovery and full configuration from the PLC**
  - The ConveyLinx-Ai module can also be configured entirely from the PLC. The configuration parameters can be found in the Properties (of the module) -> Module parameters. The ConveyLinx-Ai does not form its own name, but instead it reads it from its non-volatile memory upon start-up. However, this poses a problem with discovery of the module. If the module does not form its own name it has to be set from the engineering environment manually. As this would take a lot of time, there is a way around this – topological discovery. If the engineer has entered the project topology, the PLC can find and configure all modules in the project one by one starting with the one connected to the PLC. For more information on the topological discovery and ProfiNet explanations please refer to Appendix E “**ProfiNet and Topology concepts**”.

1.2. The ConveyLinx device access points (DAPs):

The ProfiNet protocol allows a single physical device to have multiple access ways. This is called a device access point or DAP. In essence every DAP allows the PLC to connect to a physical device in a different way (from the other DAPs). The number of DAPs a device supports is vendor specific – so every manufacturer decides how many access ways to provide for their device. ConveyLinx-Ai supports ten DAPs with different data lengths, configurations and topological discovery:

- **With topological discovery and PLC configuration (Firmware version 4.2 or higher is required)**
  - ConveyLinx-Ai in ZPA mode: 64 bytes data exchange length, 4-512 ms data exchange speeds (between 32 and 512 ms recommended for this mode), full configuration from the PLC .The name and the IP of the module can be freely set by the PLC. Internal logic is executed - the PLC can influence the module. The ConveyLinx-Ai exchanges the 64 bytes long ZPA mode data instances with the PLC. **Revision December 2019: The ZPA update times are now limited between 32 and 512ms. The change affects all GSDML files with Schema version 2.33.**
  - ConveyLinx-Ai in PLC mode: 64 bytes data exchange length, 4-512 ms data exchange speeds, full configuration from the PLC. The name and the IP of the module can be freely set by the PLC.

No internal logic – full control given to the PLC. The ConveyLinx-Ai exchanges the 64 bytes long PLC mode data instances with the PLC.

- ConveyLinx-Ai in reduced PLC mode: 16 bytes data exchange length, 4-512 ms data exchange speeds, full configuration from the PLC. The name and the IP of the module can be freely set by the PLC. No internal logic – full control given to the PLC. The ConveyLinx-Ai exchanges the 16 bytes long PLC mode reduced data instances with the PLC.
- ConveyLinx-Ai in reduced ZPA mode : 30 bytes data exchange length, 4-512 ms data exchange speeds (between 32 and 512 ms recommended for this mode), full configuration from the PLC .The name and the IP of the module can be freely set by the PLC. Internal logic is executed - the PLC can influence the module. The ConveyLinx-Ai exchanges the 30 bytes long ZPA mode reduced data instances with the PLC. **Revision December 2019: The ZPA update times are now limited between 32 and 512ms. The change affects all GSDML files with Schema version 2.33.**
- ConveyLinx-Merger: 64 bytes data exchange length, 4-512 ms data exchange speeds (between 32 and 512 ms recommended for this mode), full configuration from the PLC. The name and the IP of the module can be freely set by the PLC. Internal logic is executed - the PLC can influence the module. This DAP has additional module parameters, which allow it to be configured as a Merger. The ConveyLinx-Ai exchanges the 64 bytes long ZPA mode data instances with the PLC. **Revision December 2019: The ZPA update times are now limited between 32 and 512ms. The change affects all GSDML files with Schema version 2.33.**
- ConveyLinx-Ai in PLC mode with ConveyLogix: 32 bytes data exchange length, 4-512 ms data exchange speeds, full configuration from the PLC. The name and the IP of the module can be freely set by the PLC. No internal logic – full control given to the PLC. The Logix task is automatically started here. When a ConveyLogix program is downloaded, it will be started. The ConveyLinx-Ai exchanges the 32 bytes long Logix mode reduced data instances with the PLC.

- **With EasyRoll (previously called Standard) configuration**
  - ConveyLinx-Ai: 64 bytes data exchange length, 4-512 ms data exchange speeds (between 32 and 512 ms recommended for ZPA mode), configuration must be completed from EasyRoll. The IP put in the project must match the one given to the ConveyLinx-Ai from EasyRoll. The Name put in the project must match the one formed by the ConveyLinx-Ai during the configuration. No topological discovery supported. The data instance transmitted is always 64 bytes long, but the data contents depend on what mode the module is configured in. When the engineer adds this DAP, it has to be given the same name as the real-world ConveyLinx-Ai. This DAP can be used for ConveyLinx-Ai modules running in ZPA, PLC or Logix mode.
  - ConveyLinx-Ai in reduced PLC mode: 16 bytes data exchange length, 4-512 ms data exchange speeds, configuration must be completed from EasyRoll. The IP put in the project must match the one given to the ConveyLinx-Ai from EasyRoll. The Name put in the project must match the one formed by the ConveyLinx-Ai during the configuration. No topological discovery supported.
  - ConveyLinx-Ai in reduced ZPA mode: 30 bytes data exchange length, 4-512 ms data exchange speeds (32-512ms recommended), configuration must be completed from EasyRoll. The IP put in the project must match the one given to the ConveyLinx-Ai from EasyRoll. The Name put in the project must match the one formed by the ConveyLinx-Ai during the configuration. No topological discovery supported.

- ConveyLinx-Ai in PLC mode with ConveyLogix: 32 bytes data exchange length, 4-512 ms data exchange speeds, configuration must be completed from EasyRoll. The IP put in the project must match the one given to the ConveyLinx-Ai from EasyRoll. The Name put in the project must match the one formed by the ConveyLinx-Ai during the configuration. No internal logic – full control given to the PLC. The ConveyLinx-Ai exchanges the 32 bytes long Logix mode reduced data instances with the PLC.

The DAPs are organized in two directories:

- **Conveyor Control with EasyRoll(formerly called "standard") configuration**
- **Conveyor Control with topology and full PLC configuration**

ConveyLinx-Ai modes of operation:

- ZPA mode – This is the stand-alone mode of ConveyLinx-Ai. In this mode ConveyLinx works as one or two zone ZPA controller and executes its own control logic. When communicating with the ConveyLinx-Ai module in this mode, the PLC can control mainly material handling properties – accumulation, discharge, lane full interface, tracking, MDR speed, CONVEYSTOP. PLC can't drive directly MDRs or take control over sensors, as the internal ConveyLinx-Ai ZPA logic controls them. When communicating with ConveyLinx-Ai in ZPA mode, the PLC doesn't need very fast reaction time. 32 ms to 256 ms is more than enough.
- PLC controlled mode - In this mode the ConveyLinx-Ai does not execute any control logic. The PLC can take control over all ConveyLinx-Ai hardware – Read/Configure sensors, Run/Change Direction/Change Speed/Change Acceleration/Deceleration/Change mode. All the logic in this case has to be executed in the PLC. To minimize connections required, in this mode the PLC has access to some of the material handling properties of Upstream/Downstream module – Zone state, tracking…There are also some Servo functions for each MDR to control/measure distance travelled. In this mode of control the PLC has to communicate as fast as possible with the ConveyLinx-Ai modules. This is required for controlling movements and read sensors of diverters, turn tables… ConveyLinx-Ai cards support 4 ms communication time in this mode.
- Logix mode - In this mode, there is a running ConveyLogix program inside the ConveyLinx-Ai module. However, the PLC can still connect and influence the module. Slow communication times (128-512 ms) are advisable for larger programs, since the ConveyLinx-Ai is under a heavy CPU load from the ConveyLogix program.

For more information on the ConveyLinx-Ai modes of operation, configuration from EasyRoll and ProfiNet-unrelated features of ConveyLinx-Ai please refer to the User's Guide.

2. GSDML and UDT files.

a. Industrial Software provides one GSDML file, which includes all DAPs. The GSDML file comes with two small pictures for the devices, which will be used by the Profinet PLC development environment like TIA Portal. The latest revision is:
   - **GSDML -V2.33-IndustrialSoftware-CLAi-20190403.xml –** In the name of this file the schema version (v2.33) and date (03 April 2019) are coded. In the future versions and date may vary with new developments.

- **ConveyLinx Ai2.bmp** and **ConveyLinx Ai2mini.bmp**– those are small images, which represent how ConveyLinx-Ai/Ai2 module will look in your programming tools. The current pictures are for the Ai2 variant.

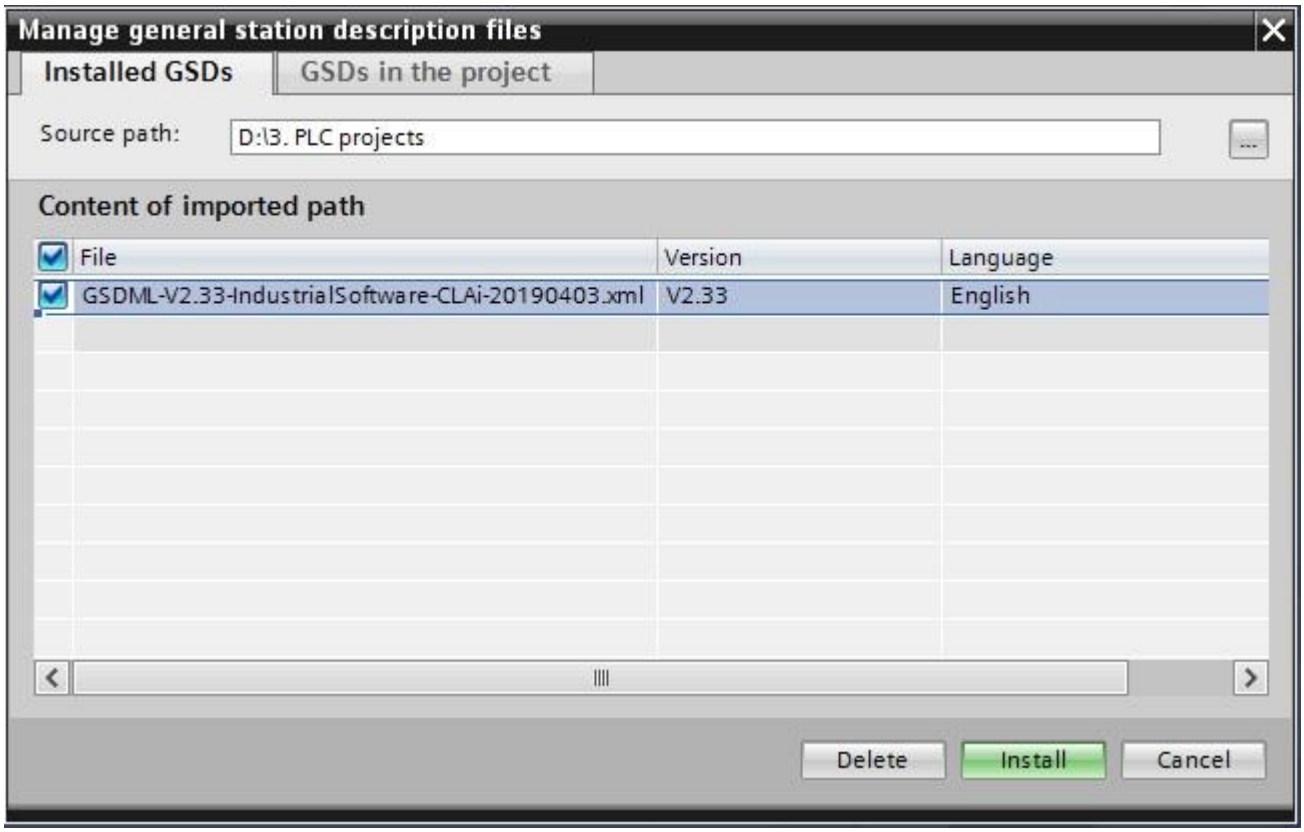
In addition, Industrial Software provides utility UDT (user data type) files, which contain the data structures for the exchanged RT data:

- **CLAi ALL UDTs rev1.7.scl** - This UDT file contains the data structure descriptions of the RT data exchanged between ConveyLinx-Ai and the PLC. This file can be imported in the TIA Portal environment.
- **Step7_UDTs Rev1.2.scl** –This UDT can be imported in Step7. It contains all the data types – full ZPA, full PLC, reduced ZPA and reduced PLC.

3. Installing the GSDML description file in TIA PORTAL V11-V15 and STEP 7 programming tools.

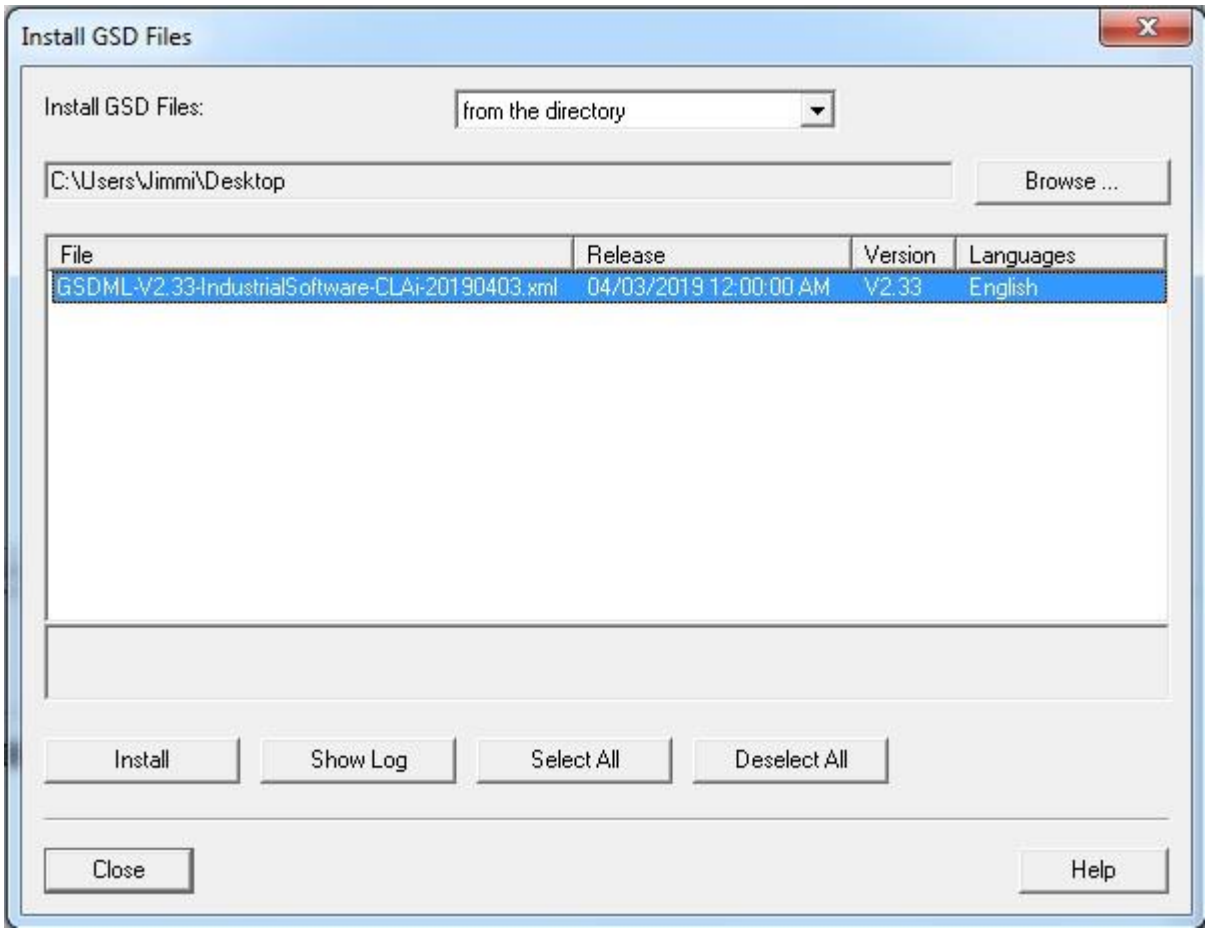
You have to install the description file in your programming tool, before you can use ConveyLinx-Ai modules as ProfiNet field devices.

In TIA Portal V11-V15 select “Options”/” install (or manage) general station description file (GSD)”. Select the xml file, provided by Industrial Software (**GSDML-V2.33-IndustrialSoftware-CLAi-20190403.xml)** and press “Install”.



Note: Please note, that TIA Portal V11 will give an alert, that the GSDML version is higher than it can process. The GSDML is a higher version (v2.33), than the environment can fully process. Please press “Yes” and install the file. All the necessary features will be available.

In Step 7 V5.5 run the HW Config and select “Options”/” Install GSD Files”. Select the xml file, provided by Industrial Software (**GSDML-V2.33-IndustrialSoftware-CLAi-20190403.xml)** and press “Install”.

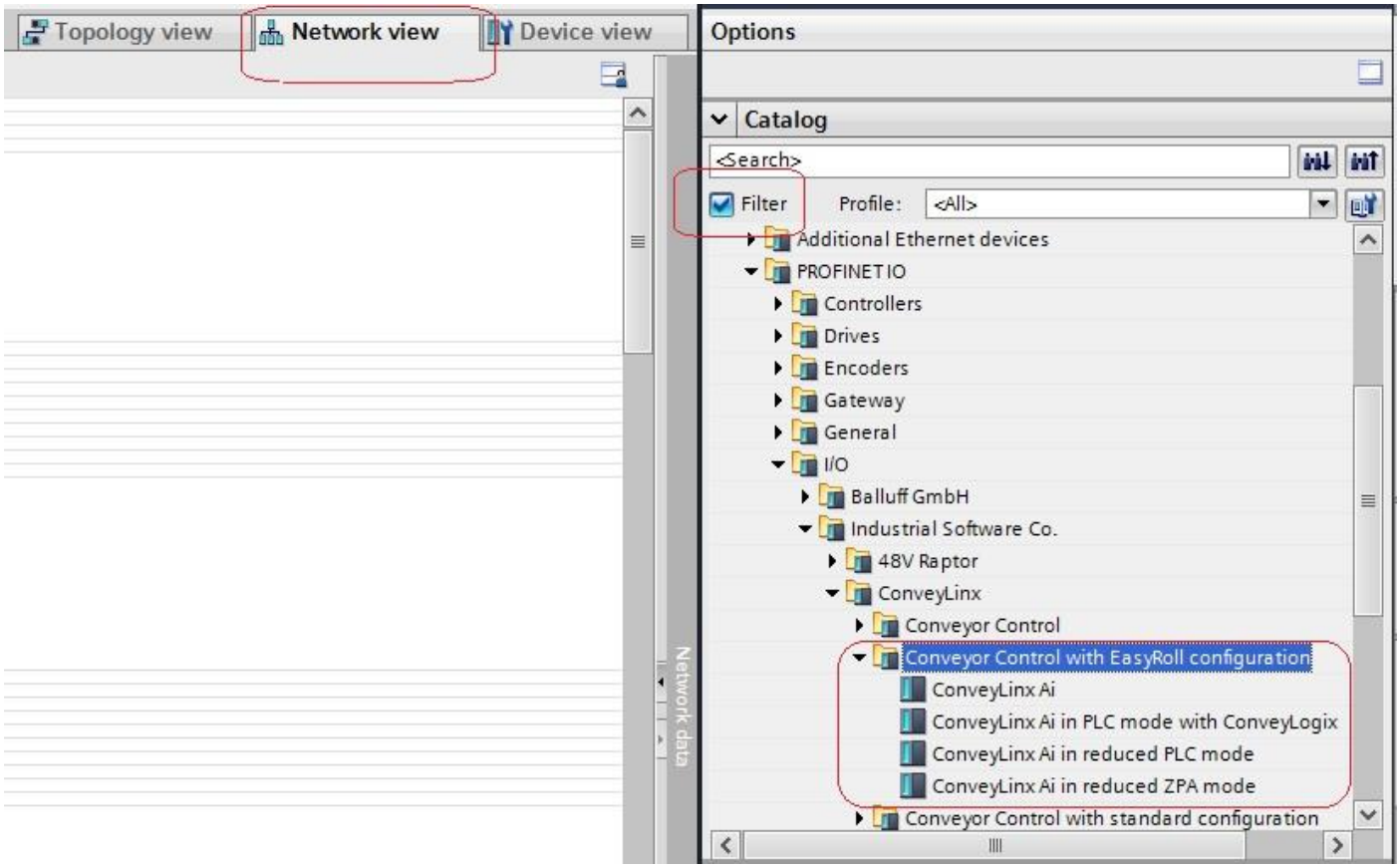


Similar to TIA Portal V11, STEP 7 will also give an alert, when you install the GSDML. Continue with the installation – all necessary features will be available.

4. Adding ConveyLinx-Ai Module as a field Profinet device

4.1. Adding ConveyLinx-Ai devices with EasyRoll configuration.

In TIA Portal V11-V15 select “Device Configuration”, “Network view” and select a device from the Hardware catalog -> “Other Field Devices”/”IO”/” Industrial Software Co.”/” ConveyLinx”/” Conveyor Control with EasyRoll configuration”

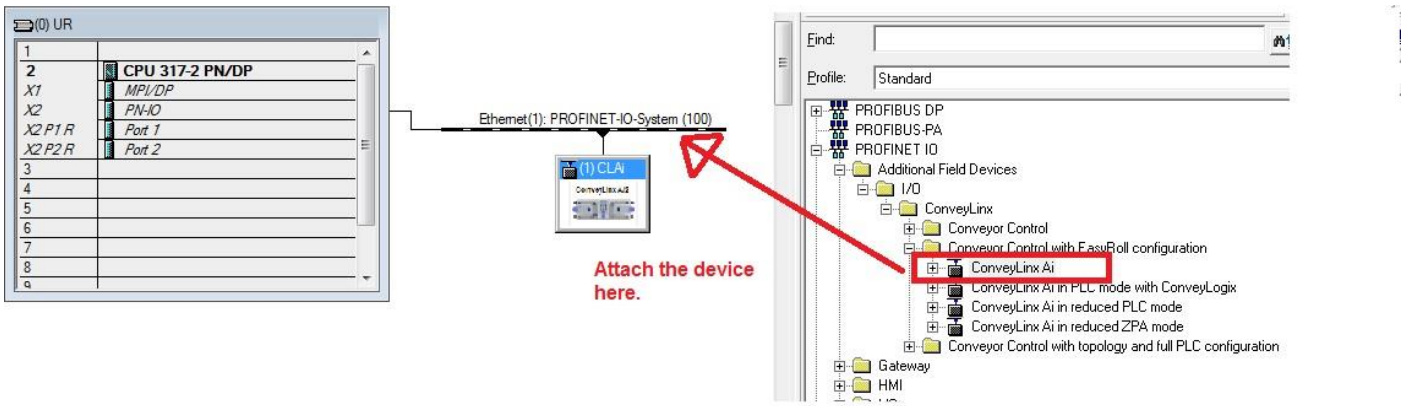


Make sure you are in Network view and the Filter checkbox is checked.

Drag & Drop/ Double click it over Network screen and attach it to the network of your PLC.

Note: In order to use ConveyLinx in full ZPA or PLC mode you need to add the device “ConveyLinx-Ai” and rename it according to convention .If you want to use reduced ZPA mode you need to add “ConveyLinx Ai ZPA control with Reduced I/O” and rename it according to the convention. If you want to use reduced PLC mode you need to add “ConveyLinx Ai PLC control with Reduced I/O” and rename it according to convention. Please refer to **section 5.1** for the naming of the devices.

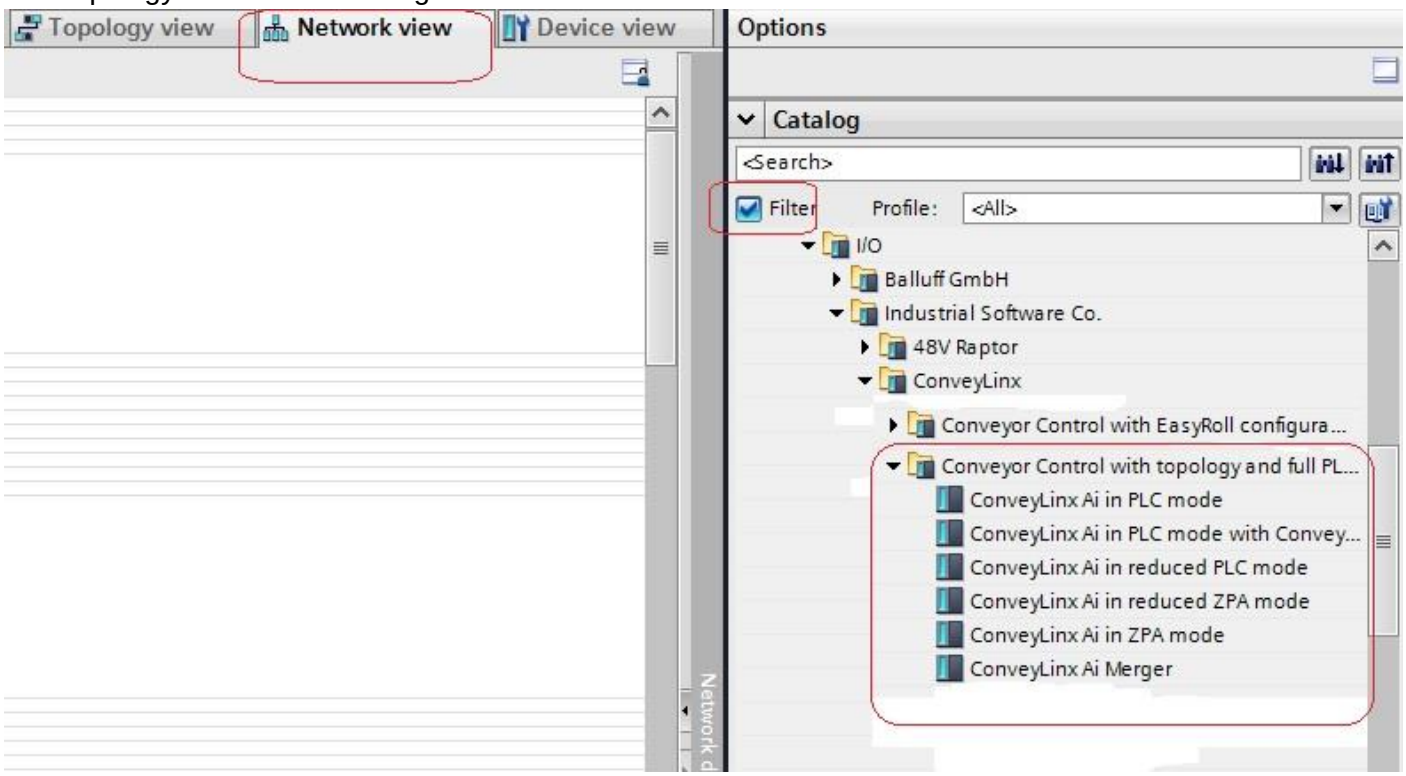
In Step 7 in HW Config Window select a device from the Hardware catalog -> “Profinet IO”/” Additional Field Devices”/”IO”/” ConveyLinx”/” Conveyor Control with EasyRoll configuration”.



Drag&Drop it over Network screen and attach it to the network of your PLC.

4.2. Adding ConveyLinx devices with topology and PLC configuration.

In TIA Portal V11-V15 select “Device Configuration”, “Network view” and select a device from the hardware catalog -> “Other Field Devices”/”IO”/” Industrial Software Co.”/” ConveyLinx”/” Conveyor Control with topology and full PLC configuration”

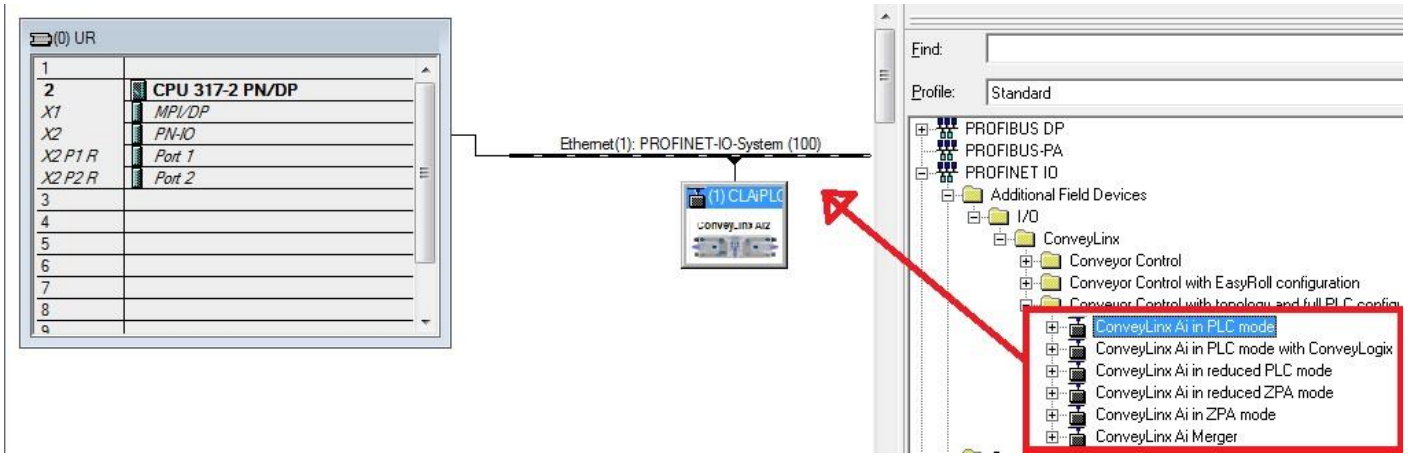


Drag & Drop/ Double click it over Network screen and attach it to the network of your PLC.

Note: In order to use ConveyLinx-Ai in full ZPA mode you need to add the device “ConveyLinx-Ai in ZPA mode”. If you want to use the ConveyLinx-Ai in full PLC mode you would have to add the device

“ConveyLinx-Ai in PLC mode”. If you want to use the ConveyLinx-Ai in reduced PLC mode you would have to add the device “ConveyLinx-Ai in reduced PLC mode”. The names you can give the devices are only limited by the ProfiNet protocol limitations. In order for the PLC to find and connect to the device, the engineer would have to either manually set the name of the device from the engineering tool or to specify the topology of the project. Please refer to Appendix E for more information on ProfiNet names ,discovery and topological discovery.

In Step 7 in HW Config Window select a device from the Hardware catalog -> “Profinet IO”/”Additional Field Devices”/”IO”/”ConveyLinx”/” Conveyor Control with topology and full PLC configuration”.



5. Commissioning of ConveyLinx-Ai module with EasyRoll configuration as PROFINET IO

5.1. Name formation

The main principle when using DAPs with EasyRoll configuration is that the ConveyLinx-Ai modules are already configured from the EasyRoll. Their IP addresses have been assigned, the modules have been put in the desired mode of operation. The modules have formed their names based on their mode and IP addresses. The modules would form their ProfiNet names by concatenating the following strings:

- ‘conveylinx’ or ‘conveylogix’ (if there is a ConveyLogix program running) +
- ‘plc’ (if in PLC I/O mode) or ‘zpa’ (if in ZPA mode) (this step is omitted if there is a ConveyLogix program running) +
- ‘-‘ +
- The value of the third byte of the IP address (for example if IP == 192.168.100.20 , then here would be ‘100’) +
- ‘-‘ +
- The value of the fourth byte of the IP address (for example if IP == 192.168.100.20 , then here would be ‘20’)

Example 1: A ConveyLinx-Ai has been configured in ZPA mode with an IP == 192.168.133.47. It would form its name as: ‘conveylinxzpa-133-47 ‘

Example 2: A ConveyLinx-Ai has been configured in PLC I/O mode with an IP == 192.168.211.107. It would form its name as: ‘conveylinxplc-211-107 ‘

Example 3: A ConveyLinx-Ai has been configured in PLC I/O mode with an IP == 192.168.0.67. A ConveyLogix program has been downloaded and is running inside the module. It would form its name as: ‘conveylogix-0-67 ‘

5.2. Additional considerations

Each ConveyLinx-Ai module is identified (discovered) on the ProfiNet Network by its name. If the engineer wants to connect to the ConveyLinx-Ai device from example 1, then he/she needs to add the “ConveyLinx-Ai” DAP from this directory, change the name of the added device to ‘conveylinxzpa-133-47’, and its IP address to 192.168.133.47. The subnet mask can be adjusted automatically by the PLC, but the Name and the IP must match.

What happens if the Name does not match?

- The PLC will not be able to discover the module, since the DAPs from this directory do not support topological discovery.

What happens if the IP address does not match?

- After **FW revision 4.11** the PLC cannot change the IP address of the module, if the module still assumes it will be used with EasyRoll configuration.

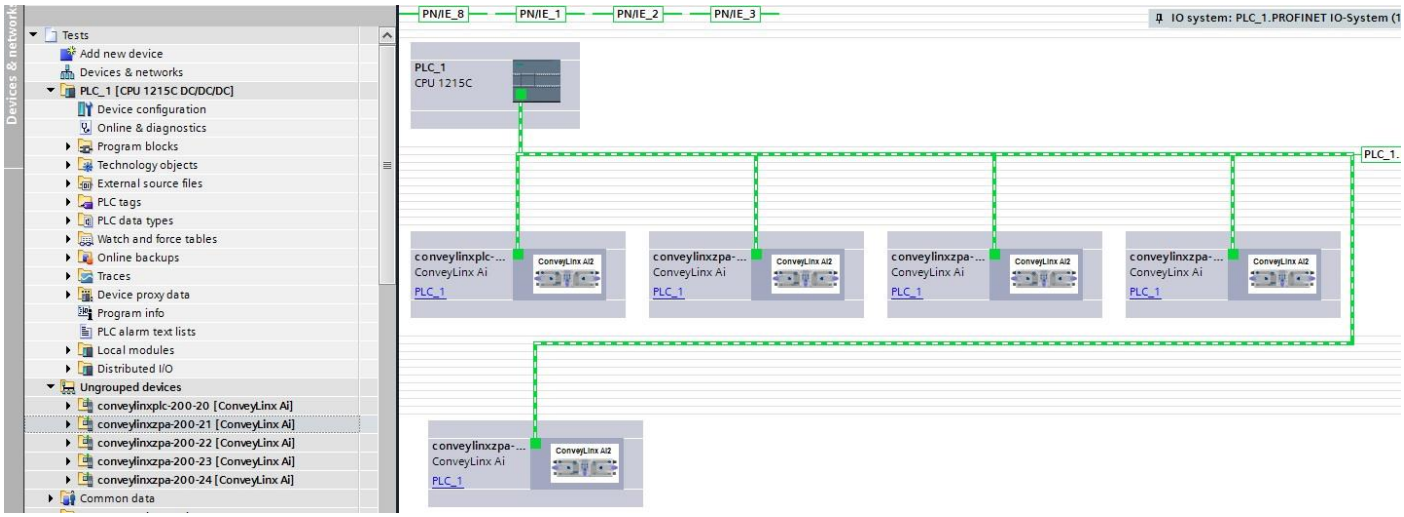
<Before revision 4.11 -**NOT APPLICABLE ANYMORE**> : If the engineer has entered the correct Name, but not the correct IP address, the PLC will change the IP of the ConveyLinx-Ai to make it match the one set in the project. Since this action will cause the ConveyLinx-Ai to form a different name (using the different IP) and become un-discoverable by the PLC upon the next power-up, the ConveyLinx-Ai will assume, that the PLC is trying to connect to it using a DAP from the “Conveyor Control with topology and full PLC configuration” directory (as it is normal for the PLC to change the IP when using those DAPs). As a consequence, the ConveyLinx-Ai will not form its name next time upon start-up .Instead it will read it from its non-volatile memory. If the module has not been configured from the PLC yet, an empty string will be read.

5.3. Additional parameters

In addition to the ProfiNet name and the IP address, there are two more parameters, that need to be adjusted - Module’s Update Time and Watchdog Timeout. The Module’s update time specifies the time interval for the Real-time communication between the PLC and the ConveyLinx-Ai. The ConveyLinx-Ai supports update times from 4ms to 512ms. When the ConveyLinx-Ai module is working in ZPA mode (internal logic is executed) , it is recommended to use 32ms or higher update time to avoid communication drops . When the module is working in PLC I/O mode 4ms is perfectly fine. When the module is working in Logix mode, the recommended update time starts from 64ms and should be increased as the ConveyLogix program becomes more complex. Update time of 4ms can be used only for simple Logix programs, otherwise disconnects can happen.

The Watchdog timeout is given as a multiple of the Update time – so number of times a message is missed .The default value (3) is fine for all modes.

Below is an example project with 5 modules. One of them is in PLC mode (192.168.200.20) and the rest are in ZPA mode, with IP addresses 192.168.200.21 to 192.168.200.24:



5.4. Assigning module's name.

When a DAP is added to a project it has a default name. This default Name needs to be changed.

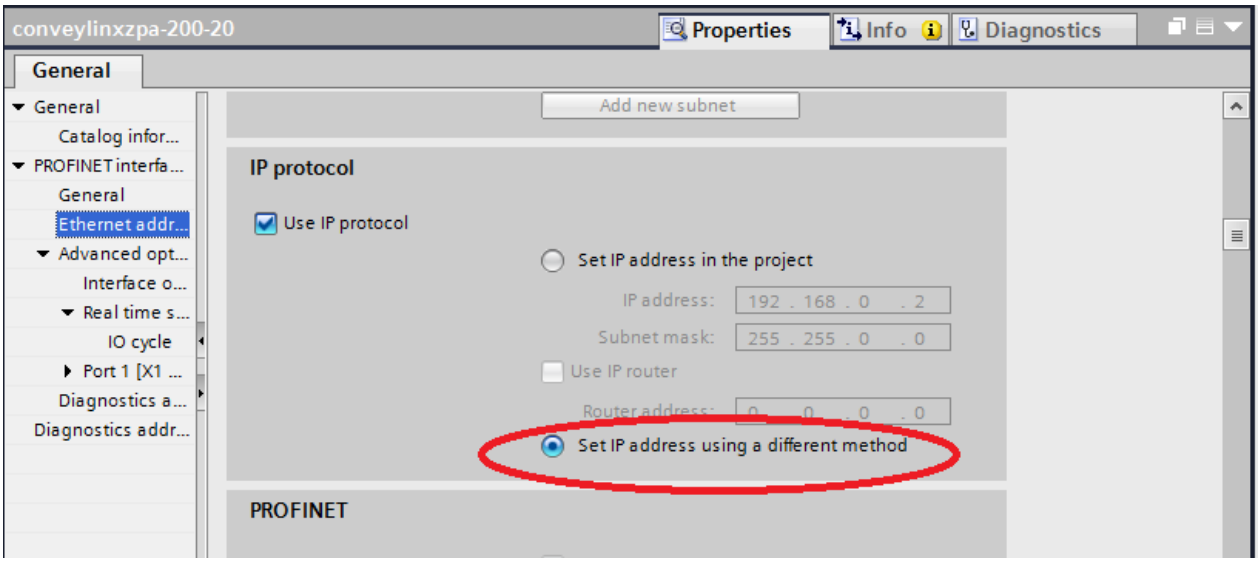
- If the ConveyLinx-Ai module , that the engineer wants to connect to, is in ZPA mode , then the device in the project needs to be named :
- "conveylinxzpa-" + third byte of the IP +"-" + fourth byte of the IP .
- For example: the real module, that the engineer wants to connect to, is in ZPA mode and with an IP of 192.168.110.20. It has formed its name as "conveylinxzpa-110-20". That's how the device in the project must be named, in order for the PLC to connect to the real-world ConveyLinx-Ai device.
- If the ConveyLinx-Ai module is in PLC I/O mode, then the device in the project needs to be named:
- "conveylinxplc-" + third byte of the IP + "-" + fourth byte of the IP
- For example: the real module, which the engineer wants to connect to is in PLC I/O mode and with an IP of 10.10.150.20 .It has formed its name as "conveylinxzpa-150-20". That's how the device in the project must be named, in order for the PLC to connect to the real-world ConveyLinx-Ai device.
- If the ConveyLinx-Ai module is in PLC I/O mode WITH a ConveyLogix program running inside , then the device in the project needs to be named:
- "conveylogix-" + third byte of the IP +"-" + fourth byte of the IP.
- For example : the real module, that the engineer wants to connect to is in PLC I/O mode WITH a ConveyLogix program running inside and with an IP of 10.10.0.20 .It has formed its name as "conveylogix-0-20" . That's how the device in the project must be named, in order for the PLC to connect to the real-world ConveyLinx-Ai device.

The example given in the last picture shows 4 modules working in ZPA mode and named accordingly and another one working in PLC mode.

5.5. Assigning Module's IP address.

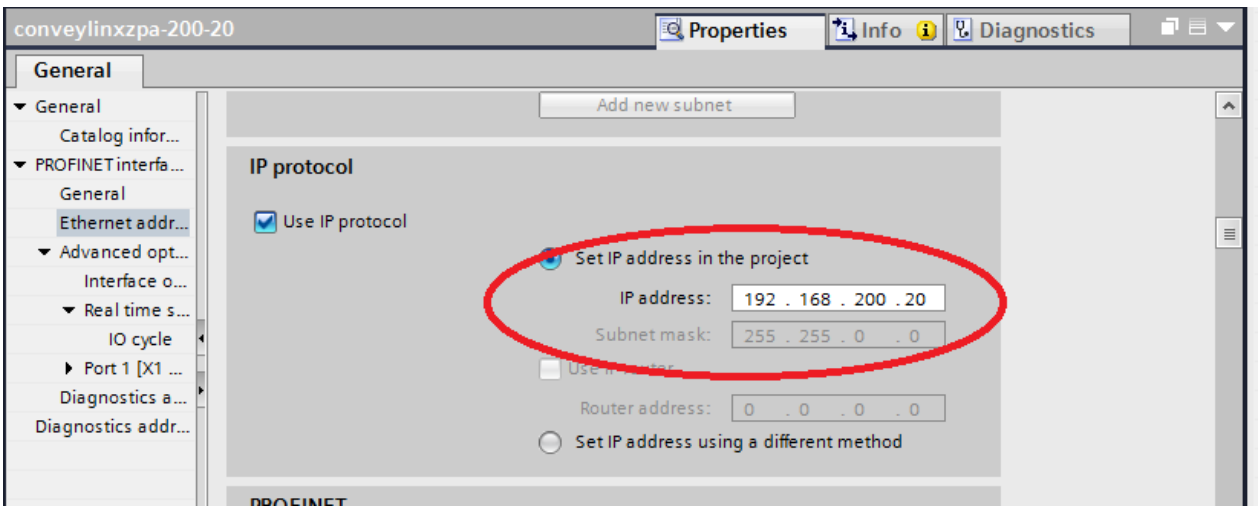
The ConveyLinx-Ai system is configured separately using EasyRoll (see **ConveyLinx-Ai User's Guide**). So their IP addresses are preconfigured and must not (the PLC can change them, but since that would lead to problems, the IP addresses must match) be changed by the PLC. So you have 2 ways to configure IP addresses of ConveyLinx modules:

5.5.1. Just select "Set IP address using different method"



When "Set IP address using a different method" is selected, the PLC will NOT attempt to change the ConveyLinx module's IP address or Subnet Mask. It will report a problem with this connection if PLC's and module's subnet mask do not match and will NOT even attempt to establish it. In this scenario, it is the responsibility of the engineer to ensure that when using the "Set IP address using a different method", the Subnet Mask of the PLC and the Subnet Masks of ALL the ConveyLinx modules are IDENTICAL. However, this method will avoid the PLC setting the wrong IP address, and the module changing its name as a result.

5.5.2. Select the IP address of the module.



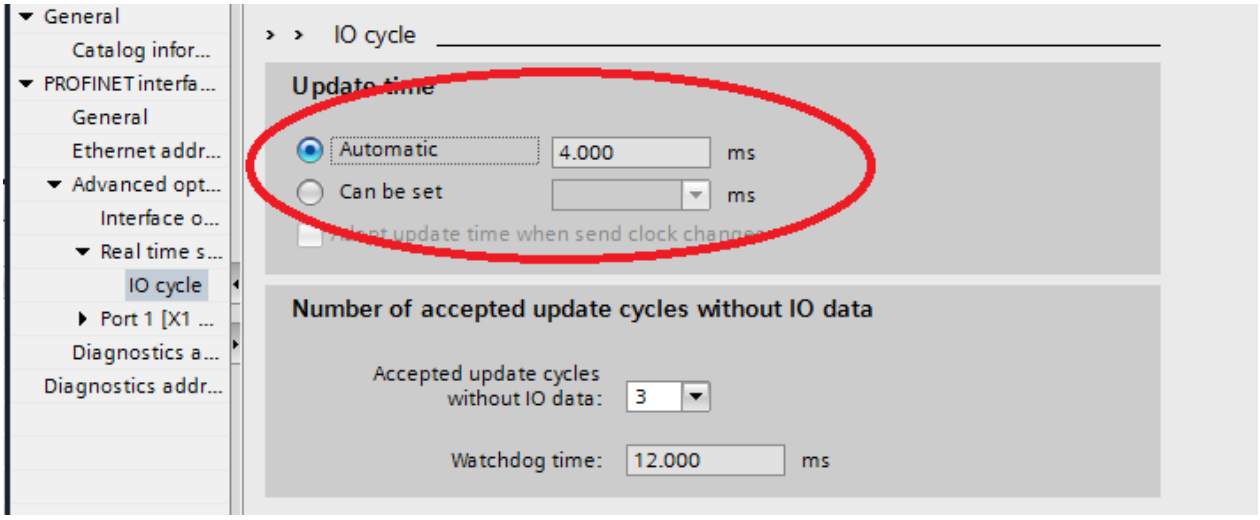
When setting the IP address through the PLC programming tool, the IP address must be Identical to the current IP address of the ConveyLinx-Ai module. Prior to FW 4.11, the ConveyLinx-Ai module will accept an IP address different than the one already set with EasyRoll, but this will cause problems with the connection upon the next power reset. After FW revision 4.11 the modules will not accept a different IP address if they are used with EasyRoll configuration.

In this case you don't have to care about the IP subnet mask of ConveyLinx-Ai modules, as the PLC will adjust their IP masks according to its IP mask.

5.6. Assigning Module's Update Time and Watchdog Timeout.

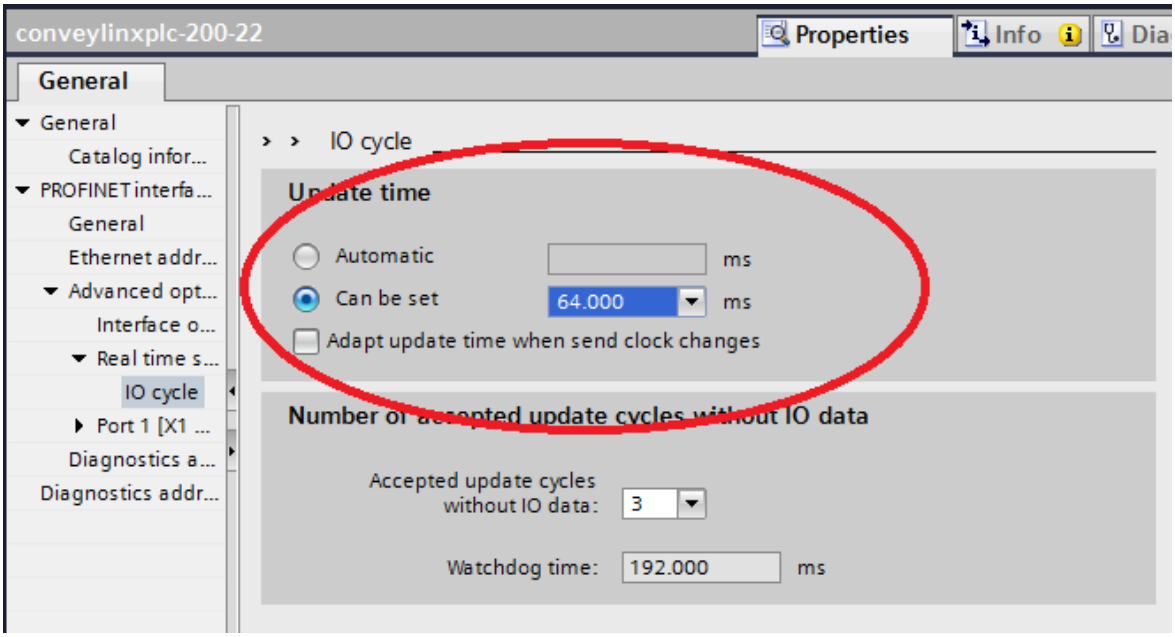
5.6.1. Assigning Modules Update Time for ConveyLinx-Ai module in PLC mode.

- If you use the ConveyLinx-Ai module as a distributed device, which is configured to operate in PLC IO mode, you may use default 4ms Update Time Interval.



5.6.2. Assigning Modules Update Time for ConveyLinx-Ai module in ZPA mode.

When in ZPA mode, ConveyLinx-Ai modules run more than 30 different tasks to accomplish ZPA conveyor zone-to-zone operation. Overloading it with unnecessary frequently communication with the PLC may cause performance and networking problems with the ConveyLinx-Ai module. For modules in ZPA mode always select Update Time of 32ms or higher.

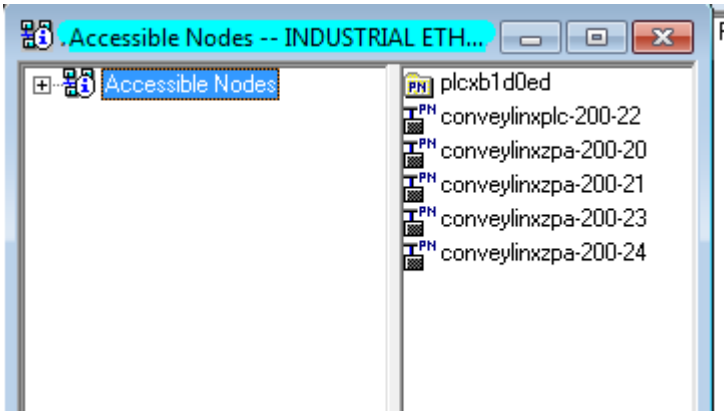


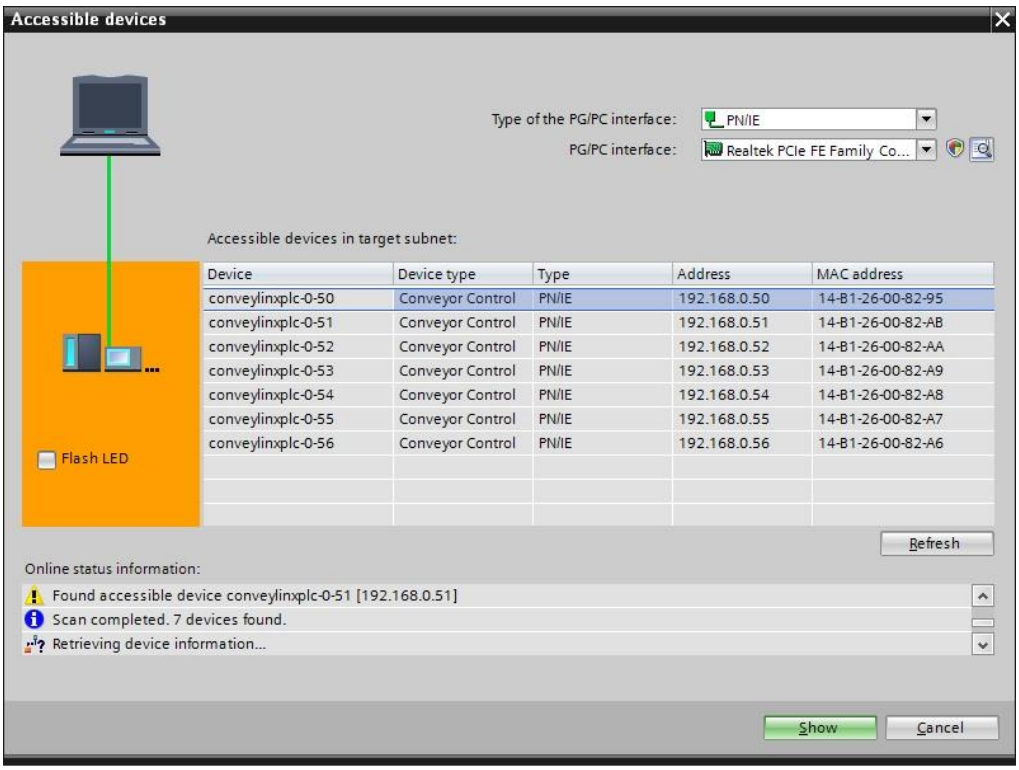
USING UNNECESSARY LOW UPDATE TIME FOR CONVEYLINX-Ai MODULES IN ZPA MODE MAY CAUSE COMMUNICATION AND PERFORMANCE PROBLEMS!

5.7. Troubleshooting

5.7.1. PLC can't communicate with ConveyLinx module. Possible reasons:

5.7.1.1. Wrong name. Check that the name you entered follows the Naming rules, described in 5.1. If Online, you may check names of all ConveyLinx-Ai modules, connected to the PLC network, using "Accessible Nodes" function of your PLC programming tool.





5.7.1.2. Wrong IP address entered. Check the IP address you selected for the module.

5.7.1.3. Wrong PLC IP masks. PLC IP mask has to allow communication with all ConveyLinx-Ai modules in the system. Good start for PLC IP mask is 255.255.0.0. This will help you to start your project with less effort.

5.7.1.4. PLC IP mask and ConveyLinx IP mask didn't match. Please refer to 5.2 of this documentation.

5.7.1.5. Incompatible addresses of PLC and ConveyLinx modules. If the PLC has an IP address of 10.10.10.1 and the ConveyLinx-Ai system has IP addresses like 192.168.1.20, 21, 22…., the PLC will simply not be able to connect to those devices.

5.7.2. Occasional communication drops – Check "Update Interval" of all ConveyLinx-Ai modules, operating in ZPA mode. It should be 32ms or higher.

6. Commissioning of ConveyLinx-Ai modules with PLC based configuration as PROFINET IO

Using ConveyLinx-Ai with the EasyRoll configuration meant dividing the engineering processing two steps – Pre-engineering (PLC project) and on-site commissioning (Auto-Configuration from EasyRoll) .If the engineer chooses to use DAPs from the directory "Conveyor Control with topology and full PLC configuration ", those two steps are merged into one single engineering phase – designing the PLC project .By using DAPs from this directory you can configure all the network topology , ConveyLinx-Ai properties , ConveyLinx-Ai configuration parameters , IP addresses , PROFINET names and modes of

operation in the PLC project . When you connect your PLC to the ConveyLinx -Ai modules, your design is automatically downloaded to the ConveyLinx-Ai modules, thus minimizing the commissioning effort. The main prerequisite for this scenario to work properly is that network topology in your PLC project is identical to how the devices are wired in the field.

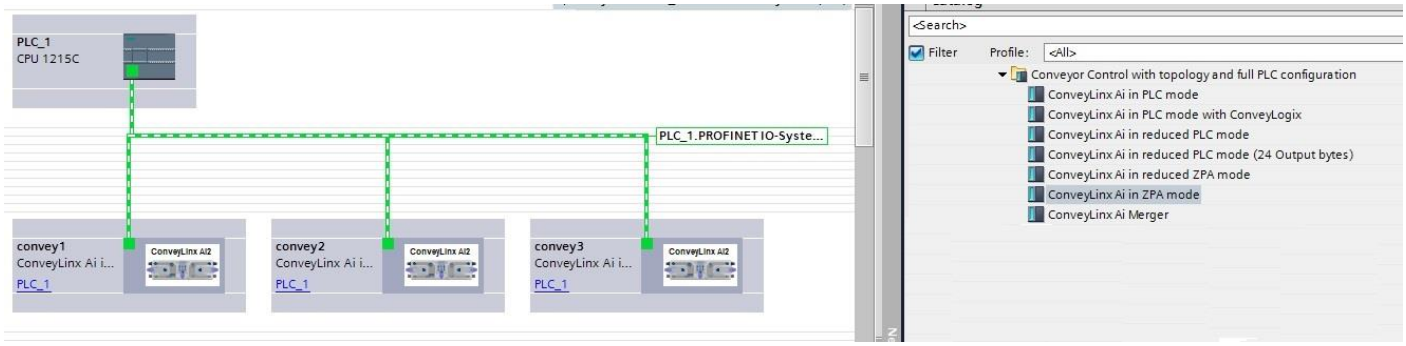
ConveyLinx-Ai modules are fully configured by the PLC when it establishes connection with the modules. This means, that the ConveyLinx-Ai will dynamically switch its mode of operation, depending on the commands coming from the PLC. When configuring the ConveyLinx-Ai module from the PLC, no external software is required. This means, that there is no need to pre-configure the ConveyLinx-Ai with EasyRoll. If there is a pre-existing configuration, it will be overwritten .The only thing that the PLC cannot overwrite, when it discovers a module topologically is the Profinet name. If a ConveyLinx-Ai module already has a set name (for example the name was set previously in a different project), and the PLC discovers it topologically, it will not attempt to set the correct name for this module.

6.1. Preparing the PLC project

6.1.1. Adding ConveyLinx-Ai modules as remote I/O to the PLC

In TIA Portal, add your PLC to the project, then go to the “Network view”.

Browse in the Hardware catalog : “Other field devices” /”PROFINET I/O” / ”I/O” / “Industrial Software Co.” / “ConveyLinx” / “Conveyor Control with topology and full PLC configuration ”. If you want to use a ConveyLinx-Ai module in ZPA mode, drag and drop/double click the “ConveyLinx-Ai in ZPA mode” DAP as shown below.



If you want to use a ConveyLinx-Ai module in PLC mode, drag and drop/double click the “ConveyLinx-Ai in PLC mode” DAP.

The same way you can add the DAP “ConveyLinx-Ai in reduced PLC mode” .This DAP has the same configuration parameters as the “ConveyLinx-Ai in PLC mode” , but exchanges 16 bytes of real-time(RT) data with the PLC .This is useful , when I/O memory is limited and not all RT data is needed for a specific device .

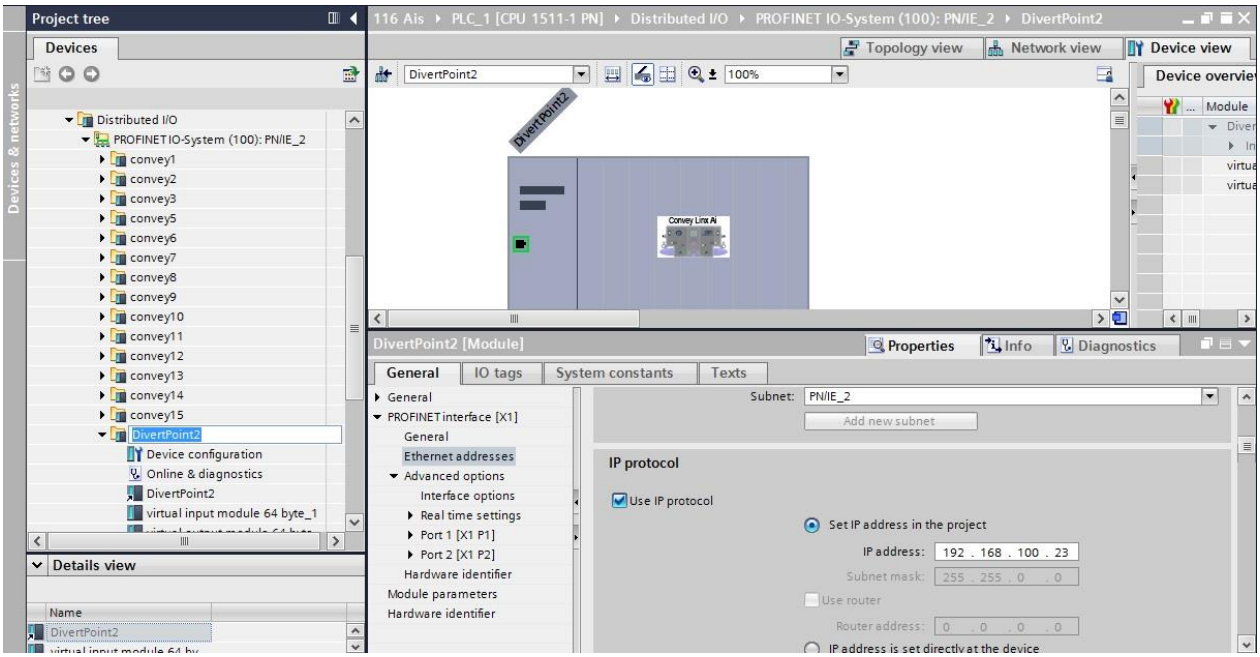
6.1.2. Configuration of ConveyLinx-Ai PROFINET properties

For each ConveyLinx-Ai module, open device configuration properties screen and choose the IP address. You can select ANY IP address, which is inside your project boundaries. Please, find how to select your system IP addresses in Siemens guides. There are no ConveyLinx-Ai specific limitations for the IP address, you select. Only general PROFINET compatibility rules are applied, which are:

- IP address – the first byte must NOT be 0, 127 or higher than 223. The last byte must NOT be 0 or 255.

Change the name of the module, so it reflects its role in the system, for example:

“Recirculation1” or “DivertPoint2”. There are no ConveyLinx-Ai specific limitations for the names you select:



<! Important note!> **Capital letters are not allowed in a Profinet name, along with special symbols like “_” (underscore). TIA Portal will automatically convert the name you set to a Profinet-compatible name. However, Step7 Manager will not do that for you. Extra care must be taken to avoid symbols and capital letters, when using Step7 Manager.**

Lastly set the Update time and the Watchdog time as you would on a DAP with EasyRoll configuration.

6.1.3. Create your system topology

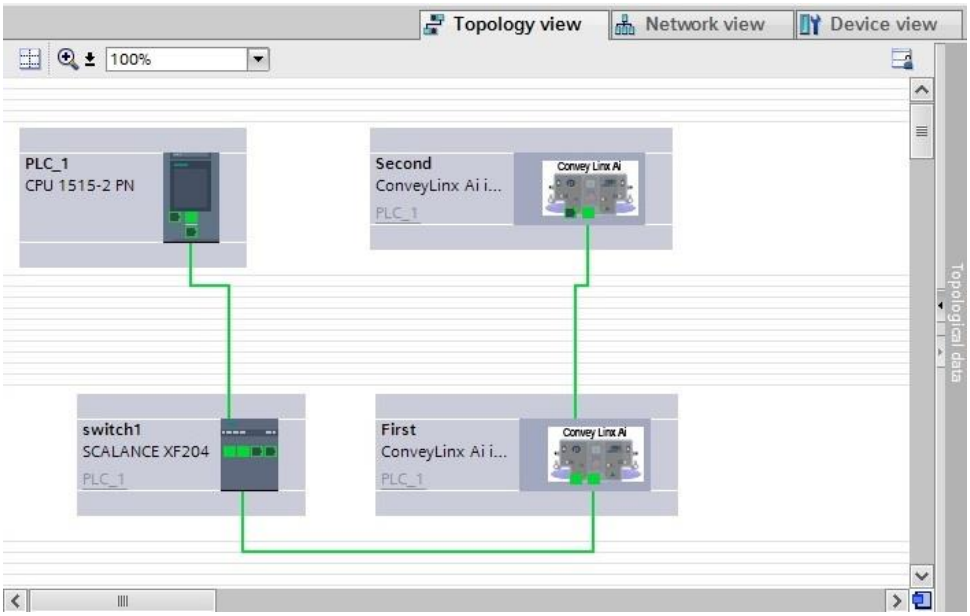
This is one of the main differences between the DAPs with “EasyRoll” and DAPs with PLC-based configuration. The PLC-based configuration DAPs support topological discovery. This means that the PLC, if given the system topology, will be able to discover the modules it needs based on their position in the topological chain. If the name of a device for a particular place does not match, it will only be changed to the one the engineer has specified in the PLC project, **IF** the module has an empty name or is coming from PulseRoller stock or has been through Auto-configuration. Once the PLC sets a name, it will not attempt to

change it (for example a module from one system is moved to another) .Please refer to Appendix E for more information on topological discovery. With new PLCs, there is a special option that would allow the PLC to change the name of a module, which already has one set. The option is a checkbox called "**Permit overwriting of device names of all assigned IO devices**". Great care must be taken, when using this option. A wrong name/port connection somewhere in the chain can result in the wrong configuration to spread like cancer in the system. We recommend for this option to not be used in a ready working system.

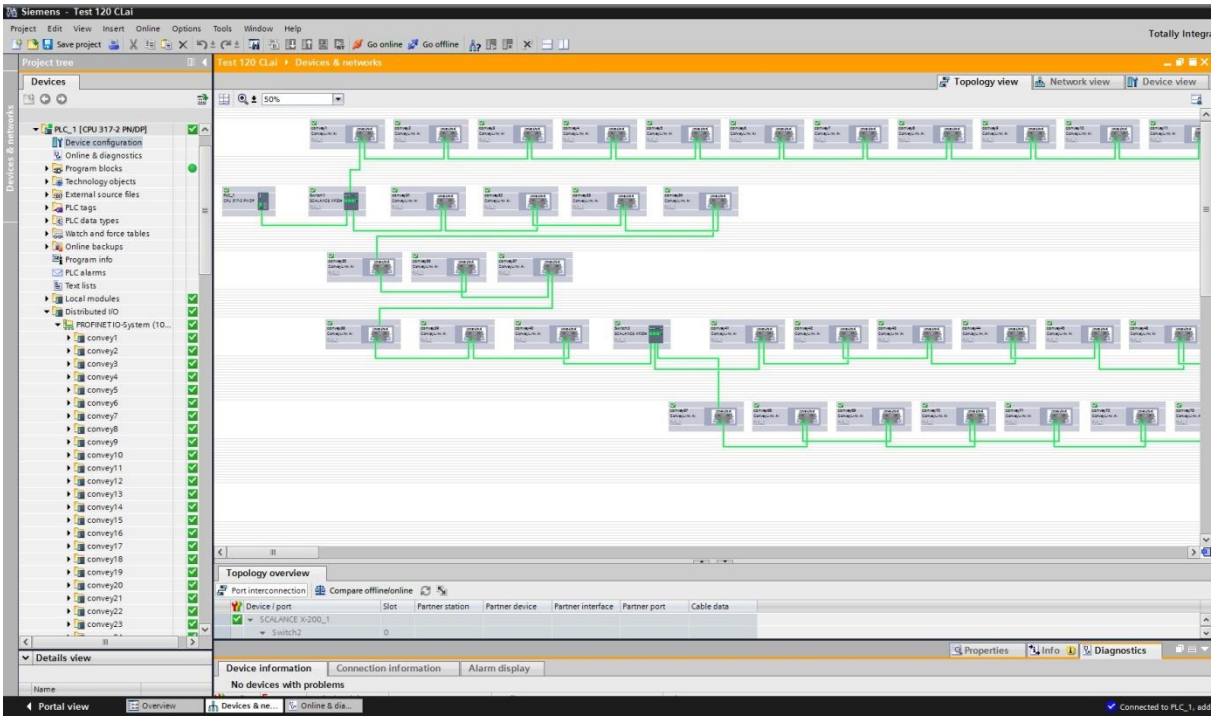
In TIA Portal, open "Devices & networks"/"Topology view". Connect the PLC to the ConveyLinx-Ai modules and switches exactly how you expect them to be wired in the field. Example:

Connect PLC's P1 port to Port 1 of the switch. Connect Right port of first ConveyLinx-Ai card to Port 2 of the switch. Connect Left port of first

ConveyLinx-Ai card to Right port of second ConveyLinx-Ai card…



Topology example 1 (close-up)



Topology example 2 (larger system)

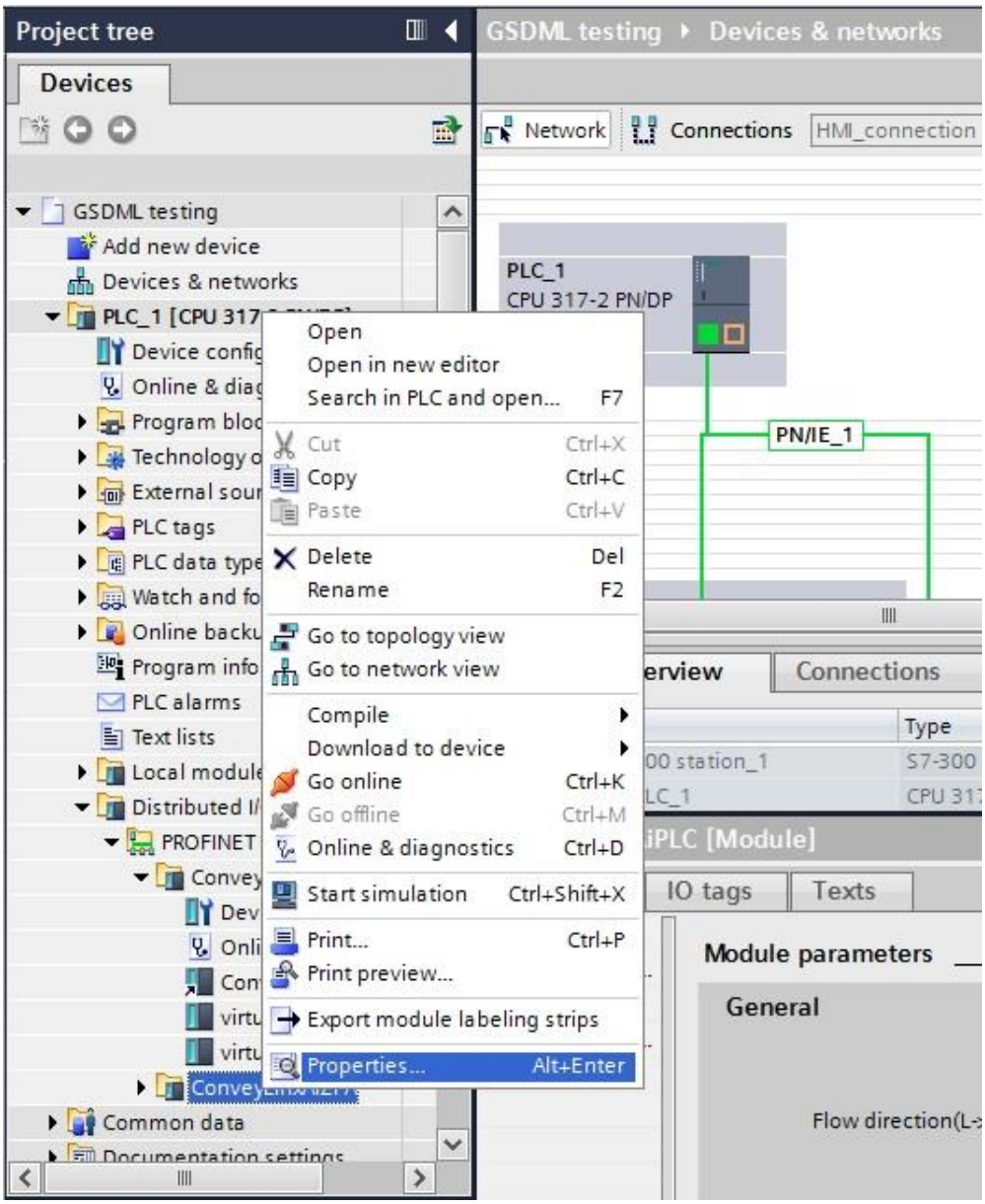
6.2. ConveyLinx-Ai configuration parameters

The dynamic configuration of the ConveyLinx-Ai with parameters given by the PLC is another major difference between the DAPs with EasyRoll configuration and those with PLC-based configuration. No configuration parameters are transmitted by the PLC to a device with EasyRoll configuration.

For each ConveyLinx-Ai module, you may configure almost all parameters that are available from EasyRoll with several notable differences:

- Extension module in ZPA mode. EasyRoll allows a module to be made to be an extension of a particular zone of another module. This forces the extension module to mirror the actions of its master zone. This is unavailable for PLC-based configuration
- The Lane full interfaces for Merger zones. It is possible to configure from the EasyRoll up to two Lane full interfaces to a merger zone. This function specifies specific zone of a module, which will be monitored. If this zone remains blocked for a certain amount of time, the Merger priority will change in order to prioritize this line.
- Separate configuration of the JAM timer and the Auto-clear timer.
- When using configuration from a Profinet PLC, the induct can only be in millimeters (distance) and not in time.
- The modules cannot be upgraded.
- The Outputs of a module in PLCIO mode will always go to OFF state upon PLC disconnect.
- Hardware controlled mode is not available via PLC-based configuration

In “Distributed I/O”, in your PROFINET I/O system, right-click on the module you want to work on and select “Properties”. In the “Properties” screen, select “Module Parameters”:



6.2.1. ConveyLinx-Ai in ZPA mode

For ZPA modules, you have the following parameter tabs and can configure the following parameters:

6.2.1.1. General tab

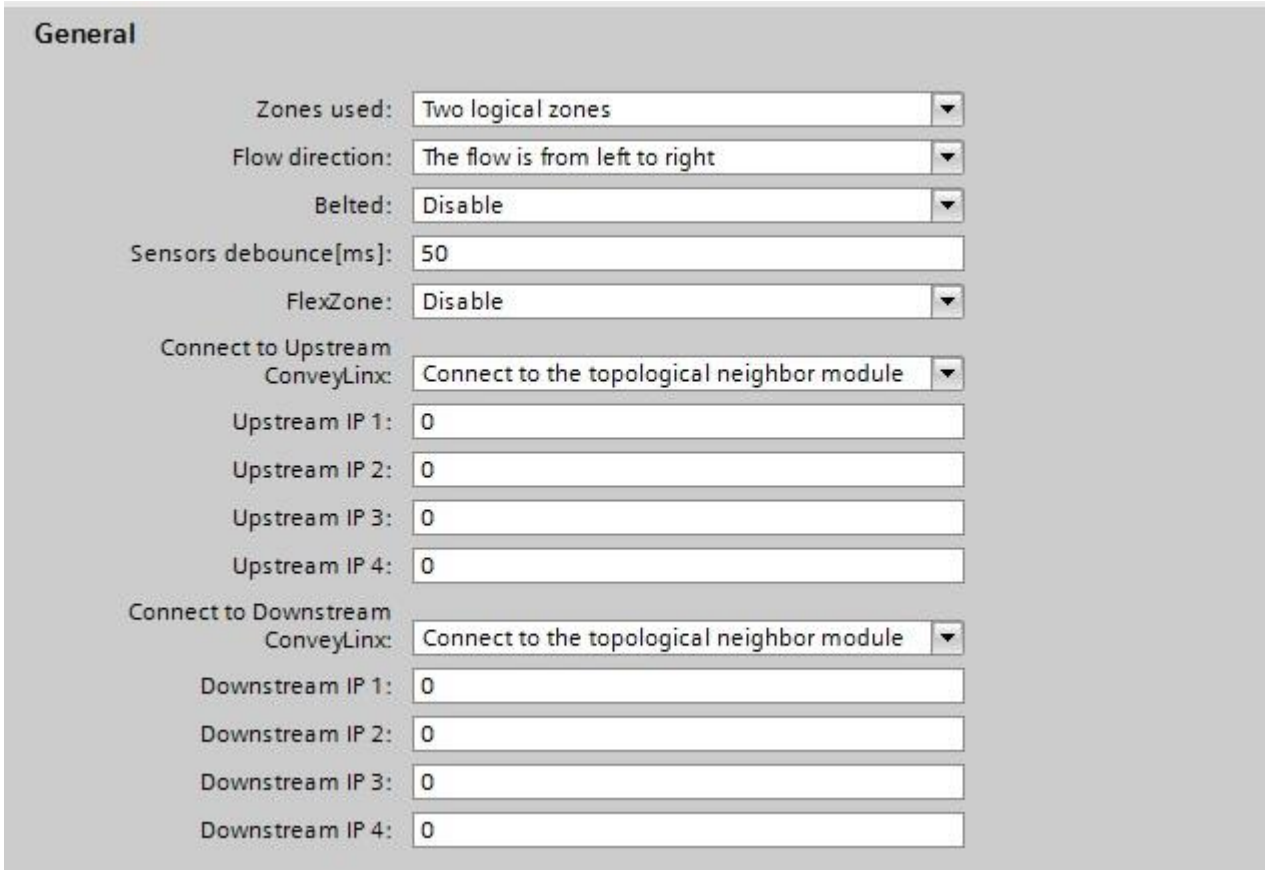
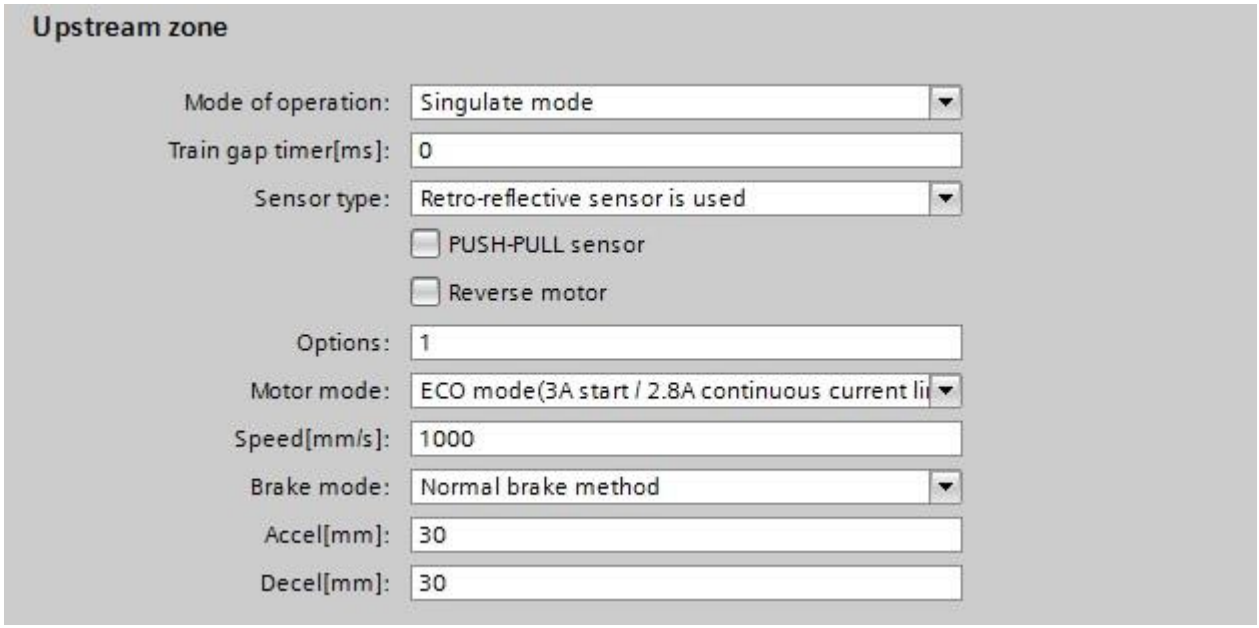


Fig. The General parameters tab

- How many zones you want to use per module – 1 or 2
- Flow direction – “Left to Right” or “Right to Left”
- If you want to use 2 MDRs on a single logical zone, you should activate the Belted function. Keep in mind that this function will only be activated if a single logical zone is selected.
- FlexZone activation (refer to ConveyLinx-Ai Users Guide”)
- Select the logical connections of the module. You may select default connection to topologically upstream/downstream module, disable the connection or specify an IP address for this module to connect to.
    - If “**Connect to topological neighbor module**” is chosen, the module will connect to the topologically Upstream/Downstream module. The network topology must be designed in the PLC project for this to happen. Even if an IP is specified in the Upstream IP 1-4 fields, it will be ignored.
    - If “**Connect to the module with the below IP address**” is chosen and all “Upstream/Downstream IP 1-4” fields are 0, then no connection will be established.
    - If “**Connect to the module with the below IP address**” is chosen and the fields “Upstream/Downstream IP 1-4” specify an IP address, then the module will make a connection to this IP address.

6.2.1.2. Upstream/Downstream zones configuration tab



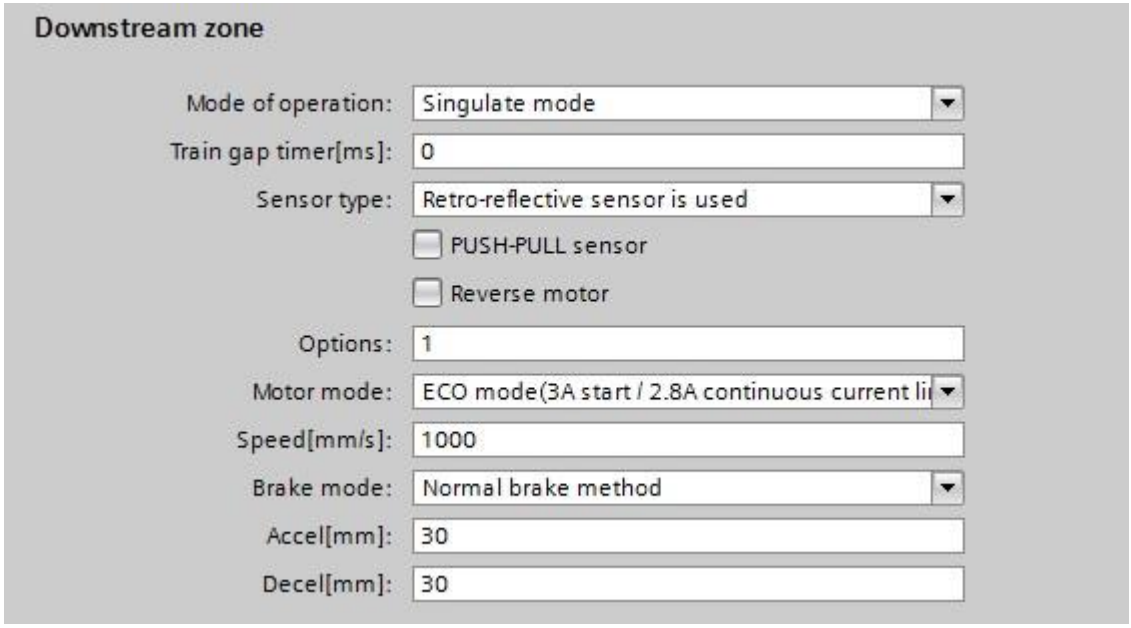
For the Upstream zone, you can configure:

- Mode of operation : Singulation or Train
- If Train mode is selected, additionally you can configure the Train Gap Timer. The GAP timer will produce a gap between a line of products. Instead of starting at the same time, each product will wait for the gap time, before being released.
- You may select Type of Sensor used
- You may reverse the motor direction. This should be used for example, when for some reason the motor is mounted on the opposite side (its tail is not on the conveyor side of the module).
- Zone's MDR operation properties- MDR mode, MDR speed (the minimum and maximum speed depends on the speed code of the roller), Acceleration and Deceleration.
- Options for ZPA operation. This field allows the engineer to enter specific codes that unlock special functionalities:
    - 1 – Enable the JAM auto clear timer (default). When an arrival JAM happens, it will be auto-cleared after the Auto clear timeout expires.
    - 2 – Reserved
    - 4 – Reserved
    - 8 – Reserved
    - 16 – Disable the arrival JAM reset delay – This option will reset an arrival JAM immediately upon JAM registration. The JAM error counter will still be incremented.
    - 32 – Disable the Sensor JAM – The sensor JAM is disabled.
    - 64 – Reserved
    - 128 – Reserved
    - 256 – Reserved

- 512 – Disable arrival Timeout. This option will cause the module not to wait for any arrival confirmation. No arrival JAM will be registered if a product does not arrive to the downstream zone.
- 1024 – Reserved
- 2048 – Reserved
- 4096 – Reserved
- 8192 – Touch-and-Go in forward direction is activated. This function will wake up a zone, when its roller is rotated for some impulses by an outside force
- 16384 – Touch-and-Go in backward direction is activated. Same as the above, but a forceful rotation in the opposite to default roller direction is needed to wake up.
- 32768 – Reserved

In order to select multiple functions, the values need to be added into one number and written to the Options menu. Unfortunately, no better visualization method is available for TIA Portal as of this writing.

The same parameters are available for the downstream zone:



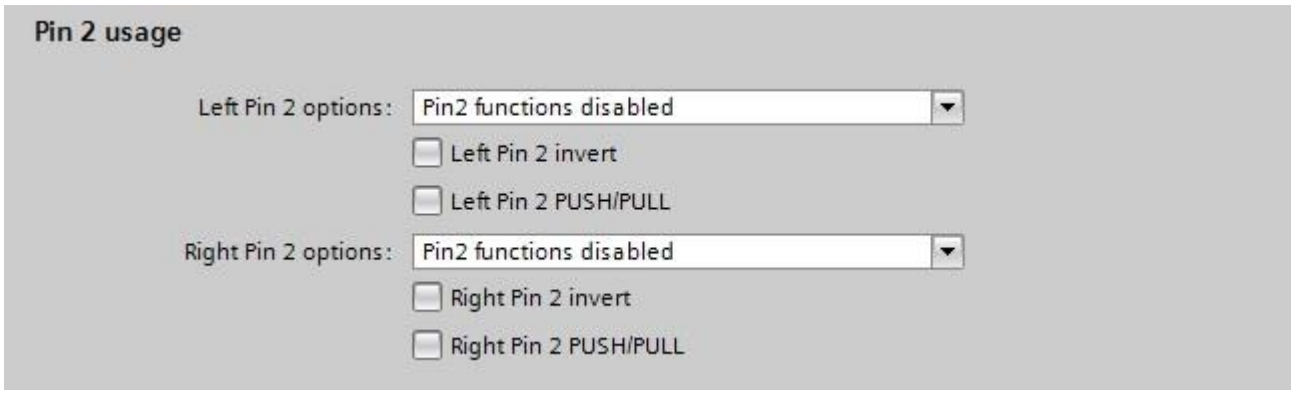
6.2.1.3. Upstream/Downstream Timing tab

In these tabs you can configure the timing properties for the Upstream/Downstream zones:

- Time , that the roller needs to run after the product has left the sensor
- Induct distance – the distance, the product should move after it has appeared on the sensor if it is to be accumulated.
- Jam timer – how long before a Jam condition is activated, when the product either jams the Sensor or does not arrive on the downstream zone.
- Slow down speed and fast release – These settings are related to “Look ahead” functions. Please refer to ConveyLinx-Ai User’s guide for more information.

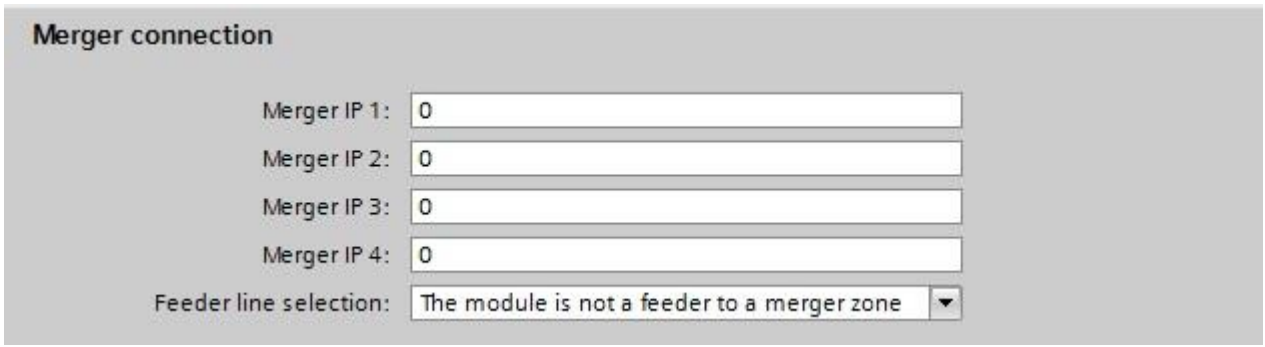
6.2.1.4. Pin 2 usage tab

In this tab you can select the Pin 2 functions, as well as the Pin 2 polarity and signal type.



The Pin2 functions that are supported in the latest FW (5.4.0) are described in appendix F:

6.2.1.5. Merger Connection



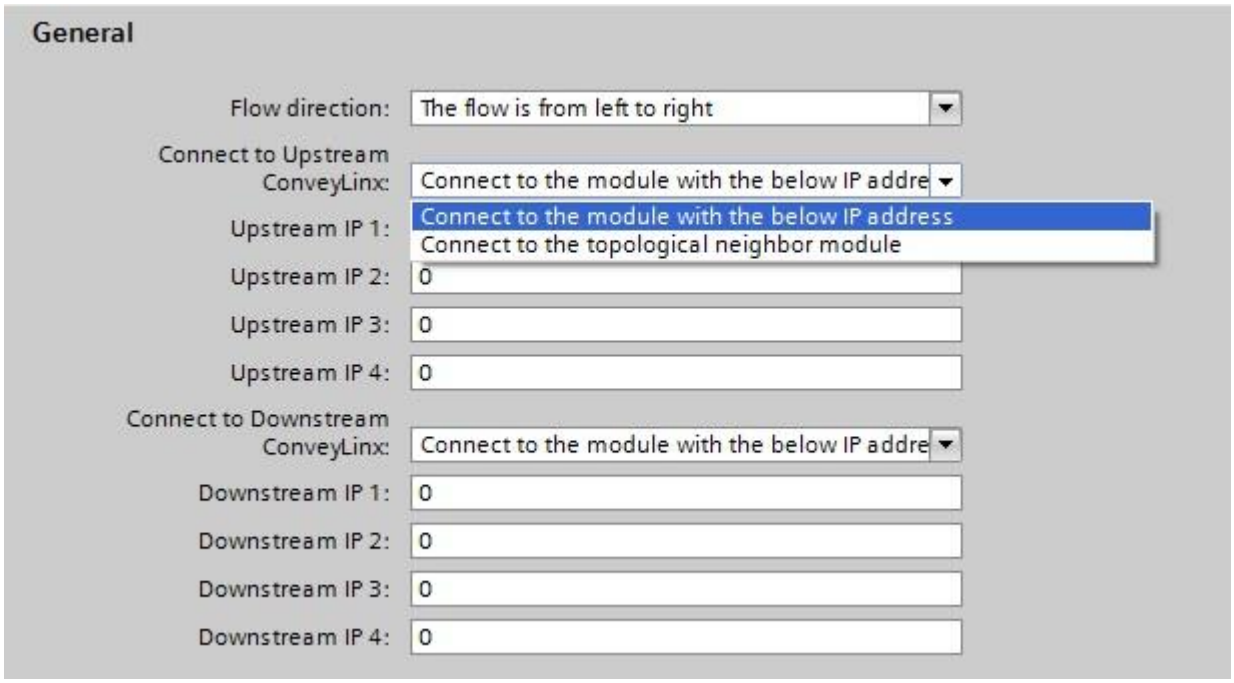
In this tab, you can specify if this zone would be a feeder zone to a merger zone. These settings are always applied to the downstream zone of the module. The IP address should be of the module that holds the merger zone.

6.2.2. ConveyLinx-Ai in PLC mode

For the ConveyLinx-Ai working in PLC I/O mode you may configure the following:

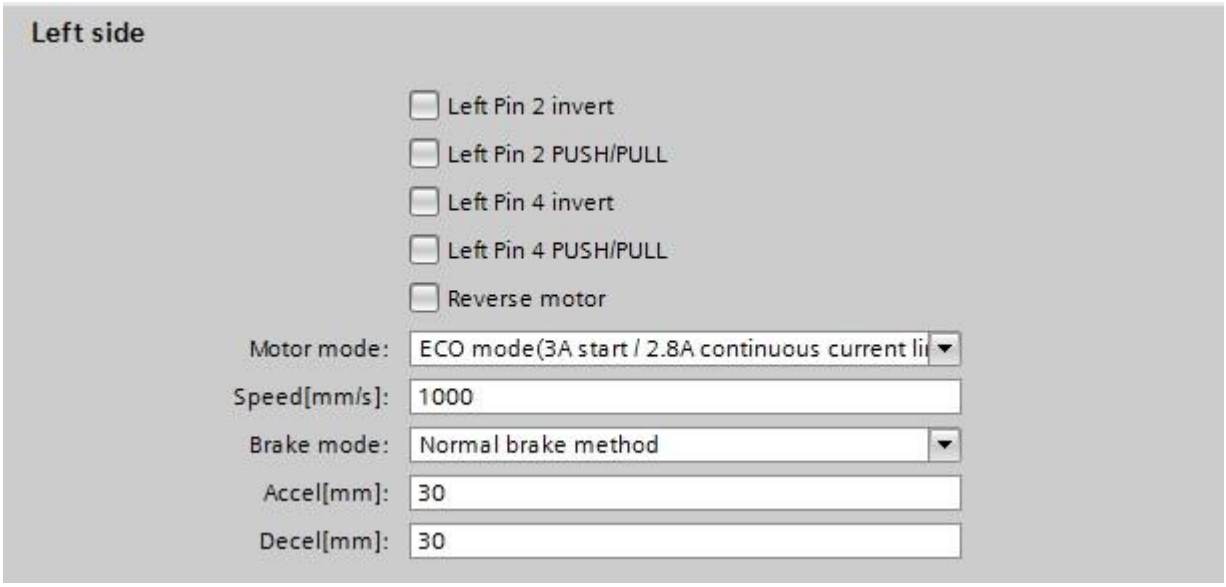
6.2.2.1. General tab

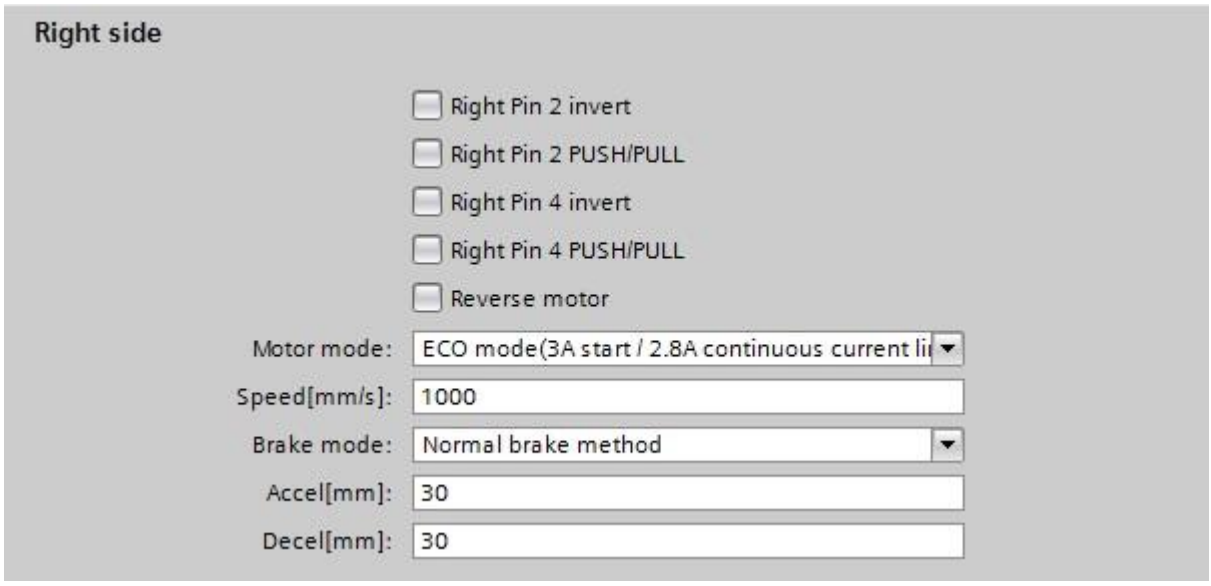
- Flow direction for this module. This information is used to build proper logical connections to the Upstream and/or Downstream modules (if needed).



6.2.2.2. Left/Right side configuration tab

You may configure the Sensor type, the signal polarity and motor properties.




Right side
Right Pin 2 invert
Right Pin 2 PUSH/PULL
Right Pin 4 invert
Right Pin 4 PUSH/PULL
Reverse motor
Motor mode:
ECO mode(3A start / 2.8A continuous current li
Speed[mm/s]:
1000
Brake mode:
Normal brake method
Accel[mm]:
30
Decel[mm]:
30

6.3. Frequently asked questions

This section applies for the DAPs with PLC based configuration.

6.3.1. After I made my PLC project, do I need to use other tools to run my system?

- No. Just connect your PLC's Ethernet port to the switch and/or ConveyLinx-Ai modules. If the topology in your project corresponds to the wired one, your system will "magically" run the way you engineered it in your PLC project. All modules, you want to be in ZPA mode will be in ZPA mode. All modules, you want to be in PLC mode, will be in PLC mode. All IP addresses, PROFINET names and module's properties will follow your project.

6.3.2. All ConveyLinx-Ai cards in my stock are with same IP address. That's how they come from manufacturer. What Should I do?

- You may use ConveyLinx -Ai cards with same IP address or with crazy IP addresses. PLC will take care to change them.

6.3.3. How does it work?

- When the project is downloaded to the PLC, the PLC starts to inspect the network topology and to search for the modules it needs. If the physical topology corresponds to the project, then the PLC will find and change the ProfiNet properties of each ConveyLinx-Ai (ProfiNet name, IP address) based on its position in the network topology. Once this is done, then the PLC will download the specific configuration to all modules (PLC/ZPA mode, parameters, etc.). Please refer to Appendix E for more information on the topological discovery and ProfiNet concepts.

6.3.4. How long does it take for the PLC to download the configuration to the ConveyLinx-ai modules?

- If this is the first time, that the PLC is configuring the system, then the PLC has to discover and change the names of all ConveyLinx-Ai modules. The PLC orders the modules to save their names in the non-volatile memory. This way, upon next power-up, all modules can be discovered simultaneously, and not one-by-one (the topological discovery is one-by-one). The initial topological discovery and name setting is a slow process, taking between 4 and 8 seconds (depends on the PLC type) per ConveyLinx-Ai module. From then on, any system startup (power-up, project change) takes much less time - less than 1 s per ConveyLinx-Ai module.

6.3.5. How does the system behave after power-up or PLC mode change from RUN to STOP and vice versa?

- Whenever the PLC is in STOP mode, ALL ZPA modules will accumulate product and all PLC controlled modules will switch their motors/digital outputs OFF. Therefore, your conveyor will be pretty quiet while the PLC is in STOP mode. After power up, when the PLC changes to RUN mode, ALL ZPA modules will execute "searching" function for products stuck in-between zone's sensors and then proceed with normal ZPA operation.

6.3.6. What happens if I need to replace a ConveyLinx-Ai module in a system?

- Just grab one module from your stock and replace it. Done. You do not need to take any additional steps. The PLC will take care of all the rest.

6.3.7. May I use general-purpose switches/hubs in my system?

- Sorry, no. The LLDP protocol is not supported by general purpose switches /hubs and it is essential for the topological discovery. So your switches need to be managed and they have to support ProfiNet IO. Good example for switches, which work for you are SCALANCE 200/300/400/500 managed switches.

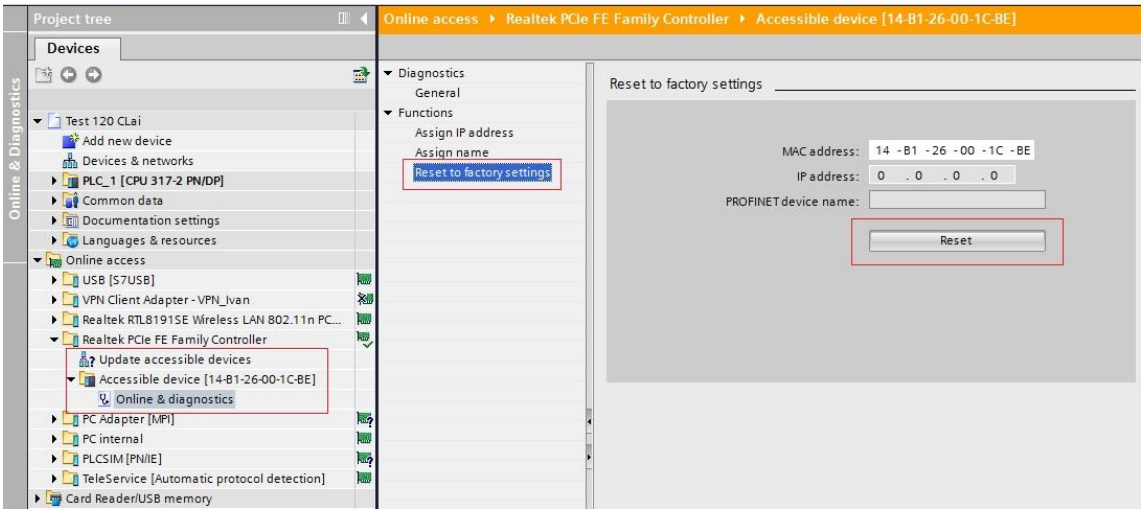
6.3.8. Interesting. May I try this with the ConveyLinx-Ai samples you sent me?

- You would need ConveyLinx-Ai firmware revision 4.1 or later and GSD file, dated after May 1, 2015.

6.3.9. My PLC refuses to configure my ConveyLinx-Ai system the “magical way” way you described. Why?

There are three main reasons that may be the cause:

- The topology in your project does not correspond to what your electricians wired. Maybe they connected your PLC not to port1 of the switch but to port 4. Or they switched the cables for left and right port of some ConveyLinx-Ai module. Please take a look in PLC diagnostics. You can also check which module has not been topologically discovered.
- Some of ConveyLinx-Ai cards you installed in the field, have already been used in the same way, you try to use it (PLC based installation). They may have some other name, not corresponding to your project set to the modules and the PLC will refuse to change it, assuming you messed-up. Only new modules, or modules installed with “Auto configuration” before, can be used. To fix this, use “Accessible devices” in your TIA Portal or Step7 tool. If you see any device, which name has not beent cleared or default or not corresponding to your project, clear it using “Reset to factory” in your TIA Portal or Step7.



!!! Important !!!

 With Firmware version 4.4 or later this requirement is no longer valid. After this FW version the PLC will change the module name to correspond with the project, even if this module has had a name set before.

 In Firmware version 4.7 and later this requirement was again introduced. Allowing the PLC to change the name of a module, which already has one set may lead to problems in rare situations.

- If in your Ethernet network, a device exists with duplicated IP address with some device in your project. Example: You have in your project a ConveyLinx-Ai module with IP address 10.10.10.33 and the lady in purchasing department uses PC with the same IP address. Please, contact your System Administrator. Moreover, disconnect from purchasing department network, please.

6.3.10. May I use this method to expand existing systems?

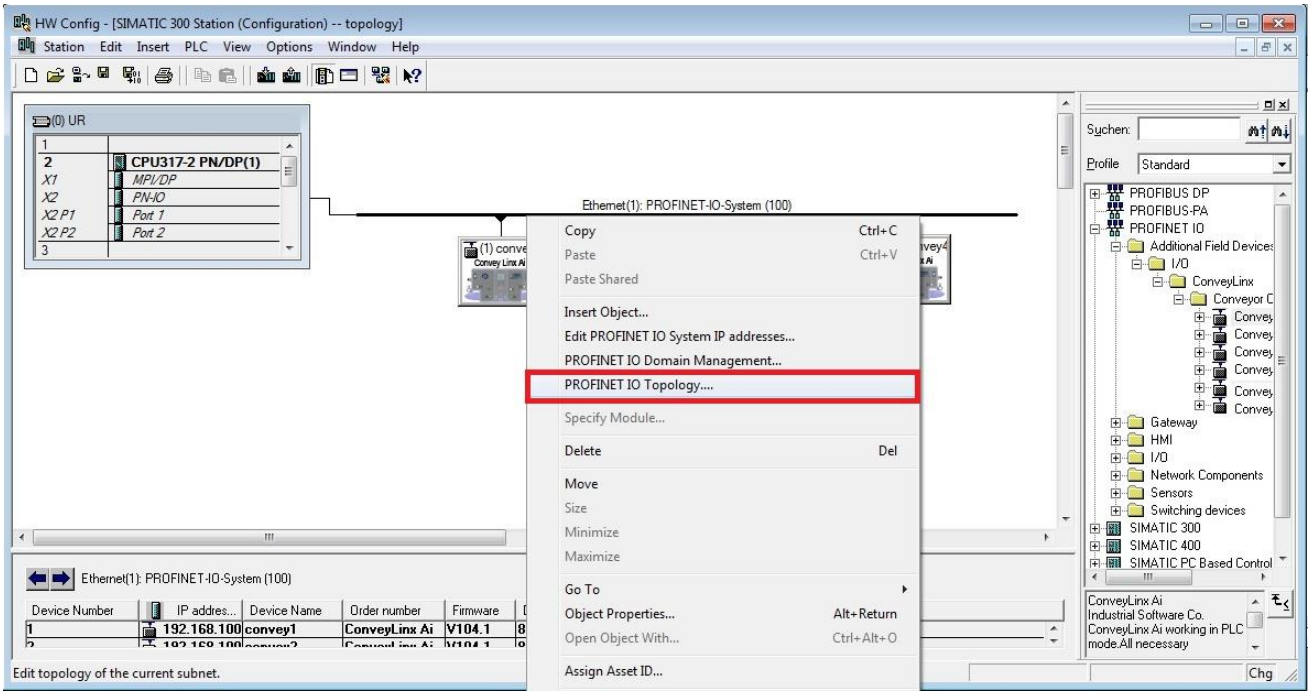
- Sure. Define the topology only for the newly added devices and leave the older part of the system with the default names, created during “Autoconfiguration” installation. Just take care to connect the new modules to different port of the switch.

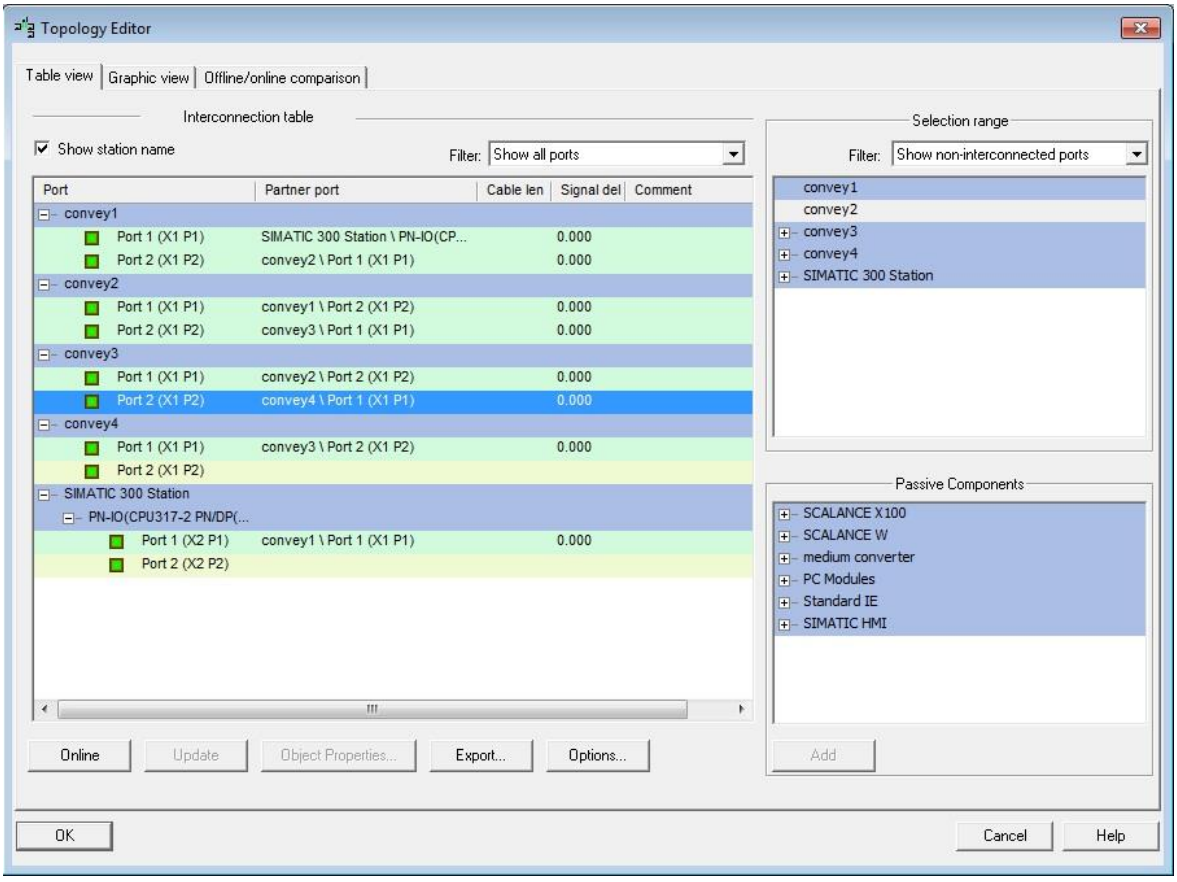
6.3.11. I tried the method you described in this chapter, but I do not like it. May I revert back?

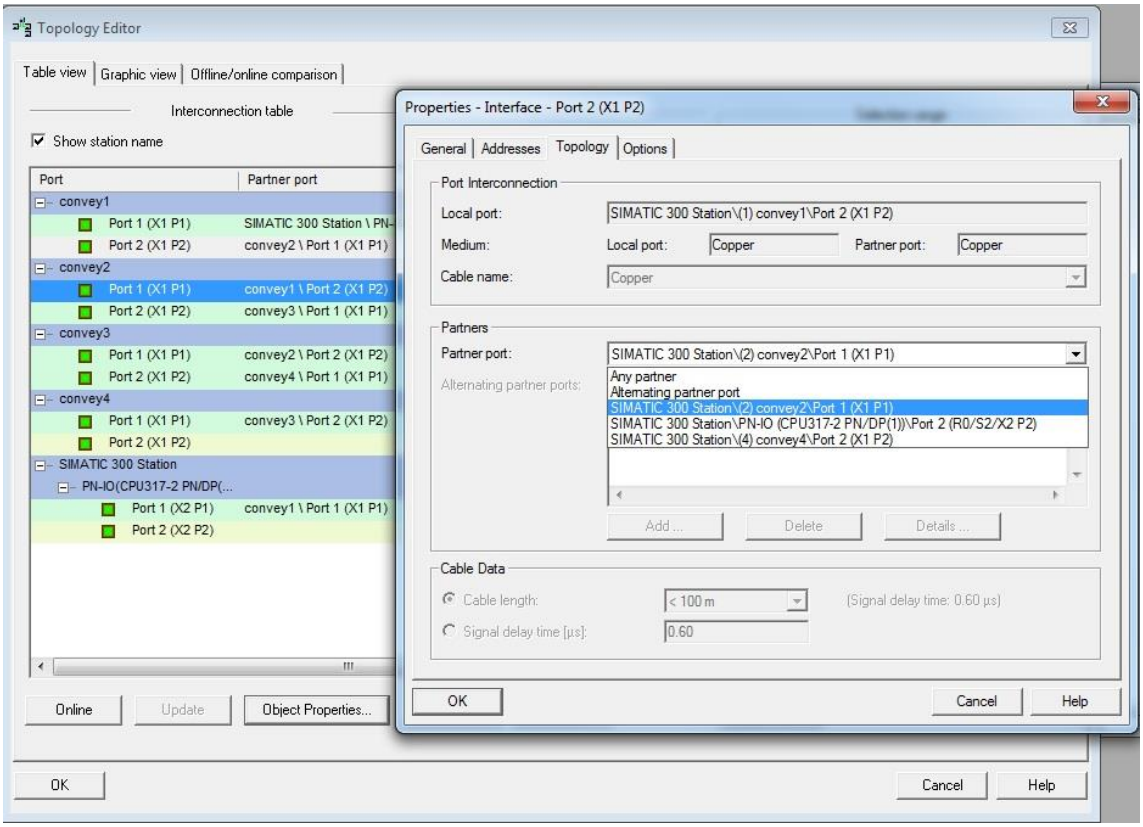
- Yes. Just run “Autoconfiguration” installation procedure for the ConveyLinx-Ai modules again. The ProfiNet names given to the modules by the PLC will be reset to the EasyRoll self-formed ones and all PLC configuration will be erased and replaced. Just do not forget to remove the DAPs from the directory “Conveyor control with topology and full PLC configuration” from your PLC project and replace them with DAPs from “Conveyor control with EasyRoll configuration”.

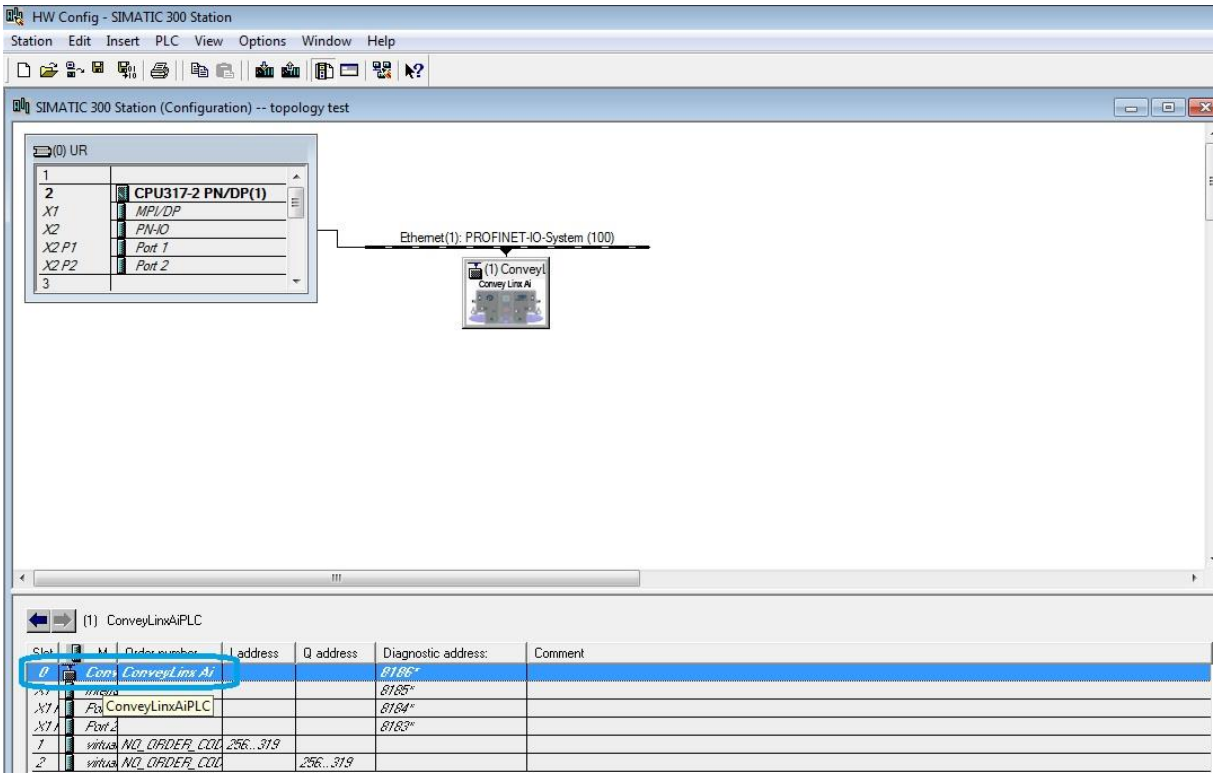
6.3.12. All your examples are for TIA Portal. However, I am a STEP7 geek. Should I change my habits?

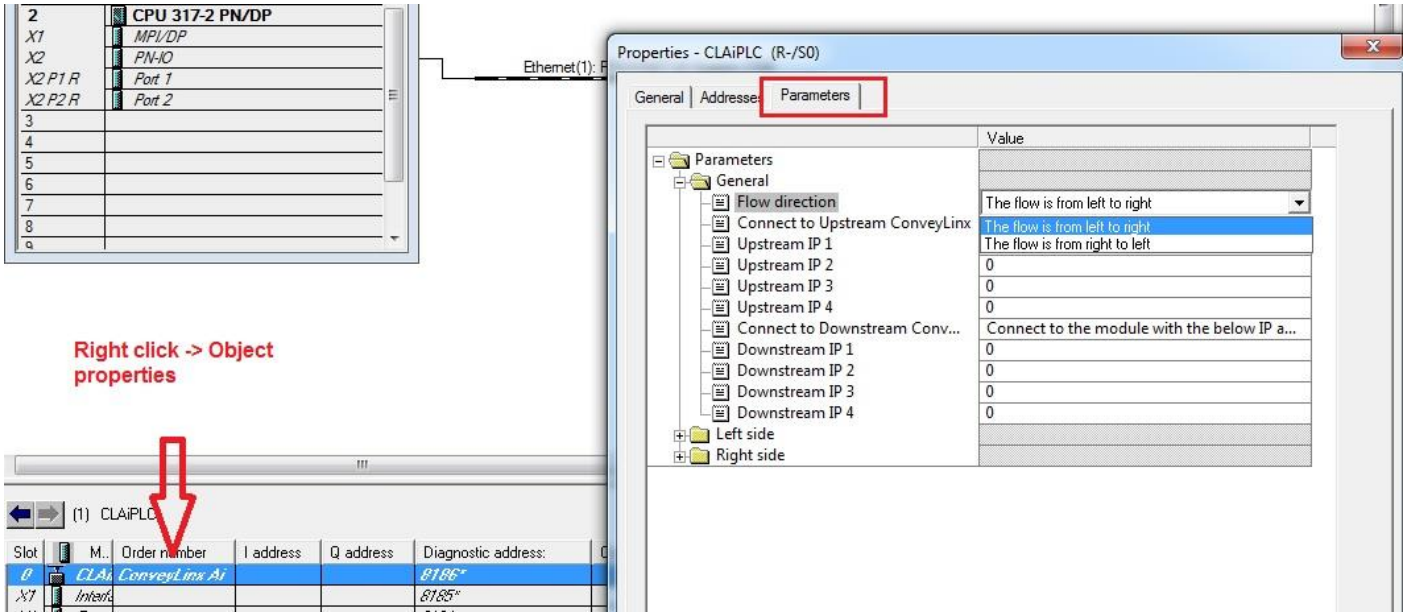
- You may continue using STEP 7. Making topology project is a little bit trickier, but all the rest is the same. You may find bellow some example of topology and module configuration screens in Step 7.


HW Config - [SIMATIC 300 Station (Configuration) -- topology]
Station Edit Insert PLC View Options Window Help
(0) UR
CPU317-2 PN/DP(1)
MPI/DP
PN-IO
Port 1
Port 2
Ethernet(1): PROFINET-IO-System (100)
Copy Ctrl+C
Paste Ctrl+V
Paste Shared
Insert Object...
Edit PROFINET IO System IP addresses...
PROFINET IO Domain Management...
PROFINET IO Topology....
Specify Module...
Delete Del
Move
Size
Minimize
Maximize
Go To
Object Properties... Alt+Return
Open Object With... Ctrl+Alt+O
Assign Asset ID...
Suchen:
Profile Standard
PROFIBUS DP
PROFIBUS-PA
PROFINET IO
Additional Field Devices
I/O
ConveyLinx
Gateway
HMI
I/O
Network Components
Sensors
Switching devices
SIMATIC 300
SIMATIC 400
SIMATIC PC Based Control
ConveyLinx Ai
Industrial Software Co.
ConveyLinx Ai working in PLC mode.All necessary
Ethernet(1): PROFINET-IO-System (100)
Device Number
IP addres...
Device Name
Order number
Firmware
1
192.168.100
convey1
ConveyLinx Ai
V104.1
Edit topology of the current subnet.
Chg


Topology Editor
Table view
Graphic view
Offline/online comparison
Interconnection table
Show station name
Filter: Show all ports
Port
Partner port
Cable len
Signal del
Comment
convey1
Port 1 (X1 P1) SIMATIC 300 Station \ PN-IO(CP... 0.000
Port 2 (X1 P2) convey2 \ Port 1 (X1 P1) 0.000
convey2
Port 1 (X1 P1) convey1 \ Port 2 (X1 P2) 0.000
Port 2 (X1 P2) convey3 \ Port 1 (X1 P1) 0.000
convey3
Port 1 (X1 P1) convey2 \ Port 2 (X1 P2) 0.000
Port 2 (X1 P2) convey4 \ Port 1 (X1 P1) 0.000
convey4
Port 1 (X1 P1) convey3 \ Port 2 (X1 P2) 0.000
Port 2 (X1 P2)
SIMATIC 300 Station
PN-IO(CPU317-2 PN/DP(...
Port 1 (X2 P1) convey1 \ Port 1 (X1 P1) 0.000
Port 2 (X2 P2)
Online
Update
Object Properties...
Export...
Options...
Selection range
Filter: Show non-interconnected ports
convey1
convey2
convey3
convey4
SIMATIC 300 Station
Passive Components
SCALANCE X100
SCALANCE W
medium converter
PC Modules
Standard IE
SIMATIC HMI
Add
OK
Cancel
Help


Topology Editor
Table view
Graphic view
Offline/online comparison
Interconnection table
Show station name
Port
Partner port
convey1
Port 1 (X1 P1)
SIMATIC 300 Station \ PN-
Port 2 (X1 P2)
convey2 \ Port 1 (X1 P1)
convey2
Port 1 (X1 P1)
convey1 \ Port 2 (X1 P2)
Port 2 (X1 P2)
convey3 \ Port 1 (X1 P1)
convey3
Port 1 (X1 P1)
convey2 \ Port 2 (X1 P2)
Port 2 (X1 P2)
convey4 \ Port 1 (X1 P1)
convey4
Port 1 (X1 P1)
convey3 \ Port 2 (X1 P2)
Port 2 (X1 P2)
SIMATIC 300 Station
PN-IO(CPU317-2 PN/DP(...
Port 1 (X2 P1)
convey1 \ Port 1 (X1 P1)
Port 2 (X2 P2)
Online
Update
Object Properties...
OK
Cancel
Help
Properties - Interface - Port 2 (X1 P2)
General
Addresses
Topology
Options
Port Interconnection
Local port:
SIMATIC 300 Station\(1) convey1\Port 2 (X1 P2)
Medium:
Local port:
Copper
Partner port:
Copper
Cable name:
Copper
Partners
Partner port:
SIMATIC 300 Station\(2) convey2\Port 1 (X1 P1)
Alternating partner ports:
Any partner
Alternating partner port
SIMATIC 300 Station\(2) convey2\Port 1 (X1 P1)
SIMATIC 300 Station\PN-IO (CPU317-2 PN/DP(1))\Port 2 (R0/S2/X2 P2)
SIMATIC 300 Station\(4) convey4\Port 2 (X1 P2)
Add ...
Delete
Details ...
Cable Data
Cable length:
< 100 m
(Signal delay time: 0.60 µs)
Signal delay time [µs]:
0.60
OK
Cancel
Help


HW Config - SIMATIC 300 Station
Station
Edit
Insert
PLC
View
Options
Window
Help
SIMATIC 300 Station (Configuration) -- topology test
(0) UR
1
2
CPU317-2 PN/DP(1)
X1
MPI/DP
X2
PN-IO
X2 P1
Port 1
X2 P2
Port 2
3
Ethernet(1): PROFINET-IO-System (100)
(1) ConveyLinxAiPLC
Diagnostic address:
Comment
ConveyLinx Ai
8186*
8185*
8184*
8183*
NO_ORDER_COD
256...319



7. Using the IO data from ConveyLinx-Ai modules.

Each ConveyLinx module occupies:
- 64 Input bytes and 64 Output bytes for full ZPA and PLC mode from PLC IO memory;
- 30 Input bytes and 30 Output bytes for reduced ZPA mode from PLC IO memory;
- 16 Input bytes and 16 Output bytes for reduced PLC mode from PLC IO memory.
- 32 Input bytes and 32 Output bytes for Logix mode from the PLC IO memory

The exchanged RT data is sent and received by the PLC in its I/O memory. The engineer has two ways to access the received data and to modify the data-to-be-sent:

- By reading/writing directly in the I/O memory using PLC tags. For example to read a “Word” from input address %IW256. For PLC series 1500 and 1200, it is possible to declare UDT tags directly into the IO memory of the PLC. **Please refer to section 8.3.**
- By copying the data to/from the I/O memory from/to local variables by using the functions SETIO/GETIO. **Please refer to section 8.2.**

Please see Appendix E, Chapter 7 for more information on UDT usage.

UDTs for PLC mode and ZPA mode ConveyLinx-Ai modules are distributed in UDTs.scl files. All fields are named accordingly and commentaries are added in order to speed up the usage of the ConveyLinx Ai data in the PLC program. Basic description of the fields is given in **section 9** below.

8. Using User Defined Types (UDT) for ConveyLinx-Ai in S7 PLC programs

8.1. UDTs basics

When a ConveyLinx-Ai module is configured as PROFINET I/O, data from/to the module is placed in the PLC's I/O memory. You may access this memory as BYTE or WORD, but that's not very user friendly (and it gets impractical as the data you use increases). Instead, you may copy this data to/from UDTs and then use their fields in your program.

There are 8 UDTs defined:

CLXZPA_IN – in a tag of this type you may get data from ConveyLinx-Ai in ZPA mode – 64 bytes.

CLXZPA_OUT – in a tag of this type you may send data to ConveyLinx-Ai in ZPA mode – 64 bytes.

CLXZPAMINI_IN – in a tag of this type you may get data from ConveyLinx-Ai in ZPA mode – 30 bytes.

CLXZPAMINI_OUT – in a tag of this type you may send data to ConveyLinx-Ai in ZPA mode – 30 bytes.

CLXPLC_IN – in a tag of this type you may get data from ConveyLinx-Ai in PLC IO mode – 64 bytes.

CLXPLC_OUT – in a tag of this type you may send data to ConveyLinx-Ai in PLC IO mode – 64 bytes.

CLXPLCMINI_IN – in a tag of this type you may get data from ConveyLinx-Ai in PLC IO mode – 16 bytes.

CLXPLCMINI_OUT – in a tag of this type you may send data to ConveyLinx-Ai in PLC IO mode – 16 bytes.

8.2. Defining tags with UDTs in your Function block and using them

To start using data to/from ConveyLinx with UDTs, first you should define tags with appropriate type. To communicate with 1 ConveyLinx-Ai module, you should define a pair of tags to read from and write to the module. Example of this definition for a module in full PLC IO mode is:

7		Static			
8		in	"CLXPLC_IN"	0.0	
9		out	"CLXPLC_OUT"	64.0	

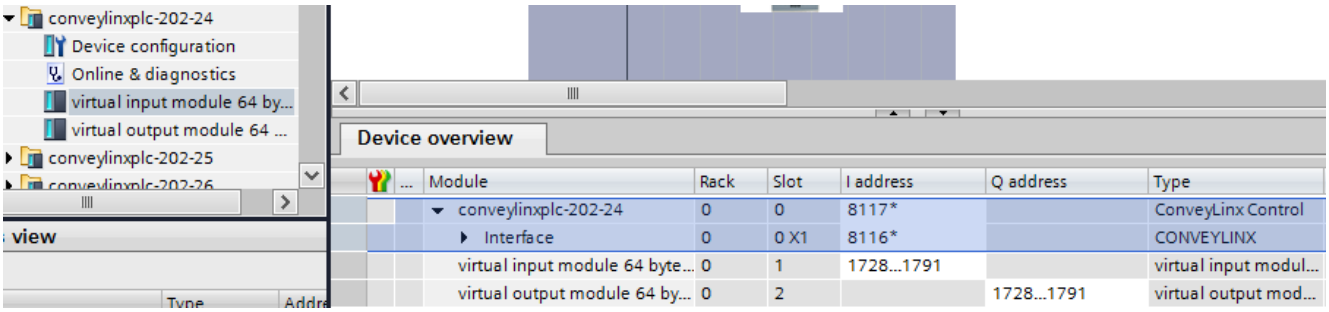
Example of definition for a module in full ZPA mode is:

12		inz	"CLXZPA_IN"	168.0	
13		outz	"CLXZPA_OUT"	232.0	

You may define arrays of UDTs to work with more than one module. In order to fill the Input tag you would need to use the function GETIO. In order to transfer the data from your Output tag to the Output memory of the controller you would need to use the SETIO Function block.

To start using GETIO and SETIO Function blocks, first you should define instances of this function blocks in your program:

In your program, first use GETIO block to get data into the input UDT tag, process the data, make decisions and finally use SETIO to set the data from output UDT tag to the Output memory area of the PLC. Then the PLC will take care to send it to the module. Example for a program, communicating with ConveyLinx card in PLC mode, which is mapped to PLC IO memory at address 1728 to 1791 (both inputs and outputs) is below:




```
#GETIO1(ID:=1728,
        INPUTS:=#in);  // Now data from ConveyLinx-Ai module is in tag #in
// Here Process it, and write control for ConveyLinx-Ai module in tag #out
// ……
//…….
#SETIO1(ID:=1728,
        STATUS=>#Status,
        OUTPUTS:=#out); //  Now write data from #out to ConveyLinx-Ai module
```

Note: For S7-1500 PLC Series the functions GETIO and SETIO use the hardware ID of the virtual input/output module of the ConveyLinx-Ai !

Using this method of data transfer ensures that all fields of the Input data are updated at the same time. And all Output data is written at the same time. This approach is more deterministic.

8.3. Defining UDT tags directly into the PLC IO memory

For PLC series S7-1200/1500, it is possible to declare complex tags directly into the IO memory. This allows the programmer to declare the input/output UDT tags directly into the IO memory and to not have to use the GETIO/SETIO functions. This is extremely useful in case, where the system consists of many modules. The PLC will place them on consecutive IO addresses and it becomes extremely easy to declare an Array of (for example) CLXPLC_IN at the starting address. With this one action, the entire Input data from all modules is available to be worked with in the programming.

One aspect should be kept under consideration. The data may be updated one field at a time. And the data may be updated while the PLC is mid-cycle.

To declare your UDT tags directly into the IO memory of the PLC, go to the PLC tags.

Here either use the default tag table or create your own. Declare the input tag at the input address of the module data and the output tag at the output address.

9. Description of UDT fields

Some fields are called 0-protected. It means that when 0 is written to this field, no action is taken – the last command is kept. These fields are marked with **"Zero Protected**" next to their data type

9.1. CLXPLC_IN

9.1.1. Convey_stop_status *Struct*

Reserved0: Bool;

Reserved1: Bool;

StopActiveCommandPLC (Offset 2): Bool; // Stop active due to stop command from the PLC

Reserved 3 to 12: Bool;

StopActiveOtherModule (Offset 13): Bool - Stop condition is active on a module in the ConveyStop group

StopActiveLostConn (Offset 14): Bool - Stop is active due to loss of a communication connection

StopActiveLostPLC (Offset 15): Bool - Stop active due to a loss of connection with the PLC

9.1.2. AllSensors *Struct*

Bit Offsets 0 to 6 are reserved

Heartbeat (Offset 7): Bool - This bit toggles every 2 seconds.

LeftPin2 (Offset 8): Bool - Left sensor port state (Pin2)

"Reserved [9]" (Offset 9): Bool;

RightPin2 (Offset 10): Bool - Right sensor port state (Pin2)

"Reserved [11]" (Offset 11): Bool;

LeftPin4 (Offset 12): Bool - Left sensor port state (Pin4)

"Reserved [13]" (Offset 13): Bool;

RightPin4 (Offset 14): Bool - Right sensor port state (Pin4)

"Reserved [15]" (Offset 15): Bool;

All these values are filtered with the "Debounce timer. Once a change of state is registered, the Debounce timer is started. For the duration of the timer, all state changes are ignored.

9.1.3. SensorDetect *Struct*

Reserved (Offset 0-7): Byte

RightSensorDetect (Offset 8): Bool - If this bit is TRUE a sensor has been detected on the Right sensor port

LeftSensorDetect (Offset 9): Bool- If this bit is TRUE a sensor has been detected on the Left sensor port

Offsets 10 to 15 are reserved and not used.

9.1.4. Voltage ***Int*** - Power supply voltage of the module in mV – 24000 for 24V

This value is averaged for a 500ms interval.

9.1.5. LeftCurrent ***Int*** - Left MDR current in mA – 5000 for 5A

This value is averaged for a 500ms interval.

9.1.6. LeftFreq ***Int*** Frequency on Hall Effect sensors on Left MDR in Hz – 500 for 500Hz.

9.1.7. LeftCalcTemp ***Byte*** Calculated temperature of Left MDR in deg. Celsius

9.1.8. ModuleTemp ***Byte*** Temperature measured by CPU's thermo sensor in deg. Celsius

9.1.9. LeftMDRDiagnostic ***Struct*** Diagnostic information for the Left MDR. See **Appendix A** for more information. Please note that the places of individual bytes are swapped, due to the internal processing of the I/O data in the PLC.

9.1.10. RightCurrent ***Int*** Same as 9.1.5, but for the Right MDR.

9.1.11. RightFreq ***Int*** Same as 9.1.6, but for the Right MDR.

9.1.12. RightCalcTemp ***Byte*** Same as 9.1.7, but for the Right MDR.

9.1.13. ModuleTemp ***Byte*** Same as 9.1.8, but for the Right MDR.

9.1.14. RightMDRDiagnostic ***Struct*** Diagnostic information for the Right MDR. The fields are the same as for the **LeftMDRDiagnostic struct .**

9.1.15. LeftMDR_DIOstatus : Struct This is the status when the MDR is used as a digital IO. Please refer to **Appendix B.**

9.1.16. RightMDR_DIOstatus : Struct

Same as 9.1.15, but for the Right MDR.

9.1.17. UpstreamModuleStatus ***Word*** ZPA status of the Upstream ConveyLinx module. See **Appendix C** for more information.

9.1.18. DownstreamModuleStatus ***Word*** ZPA status of the Downstream ConveyLinx module. See **Appendix C** for more information.

9.1.19. TrackingFromUpstream ***DWord*** ZPA Tracking Data of Upstream ConveyLinx module. See **Appendix C** for more information.

9.1.20. DistanceLeft ***Word*** Distance traveled by Left MDR, in 'mm'. See **Appendix D** how to use Servo (position) Control with ConveyLinx module. The value is in pulses, when a PGD is used.

9.1.21. DistanceRight ***Word*** Distance traveled by Right MDR, in 'mm'. See **Appendix D** how to use Servo (position) Control with ConveyLinx module. The value is in pulses, when a PGD is used.

9.1.22. ServoStatusLeft ***Word*** Status Word for Servo commands for Left MDR. See **Appendix D** how to use Servo (position) Control with ConveyLinx module

9.1.23. ServoStatusRight ***Word*** Status Word for Servo commands for Right MDR. See **Appendix D** how to use Servo (position) Control with ConveyLinx module

9.1.24. LeftMotRealSpeed ***Word*** Real MDR speed in (mm/sec) – field value is 500 for 0.5 m/sec, bit 14 is 1 when Set Speed is higher than the maximum possible speed, bit 15 is 1 when Set Speed is lower than minimum possible speed. Bits are counted from 0. The base unit is one tenth of an RPM, when a PGD is used. The value of 1000 would equate to 100 RPM.

For example for MDR with speed code 45 and tube diameter 50 mm maximum possible speed is 1.1 m/s and minimum possible speed is 0.1 m/s. If you set speed 1.2 m/s bit 14 will be 1 and if you set speed 0.05 m/s bit 15 will be 1.

9.1.25. RightMotRealSpeed ***Word*** Same as 9.1.24, but for Right MDR.

9.2. CLXPLC_OUT

9.2.1. Convey_stop_control *Word*

If ConveyLinx module belongs to ConveyStop group, writing 1 to this word will force Stop condition to all modules in the group. Writing 2(**there needs to be a transition from 0 to 2**) will clear Stop on the entire group, if all ConveyStop forcing conditions are cleared (All Stop buttons are in default (ON) state, module to module communication and PLC connections are OK).

9.2.2. LeftMDRasDIO : Struct // Use the MDR as a digital IO

Please refer to Appendix B

9.2.3. RightMDRasDIO : Struct // Use the MDR as a digital IO

Please refer to Appendix B

9.2.4. SensorsDigitalOutputs ***Struct*** This data field is used to set Pin2 on each port as outputs and to drive them:

Offsets 0 to 7 are reserved and not used .

DriveLeftPIN2 (Offset 8): Bool - Setting this bit drives the Left PIN2 to logical 1

DriveRightPIN2 (Offset 9): Bool - Setting this bit drives the Right PIN2 to logical 1

"Reserved [10]" (Offset 10): Bool

"Reserved [11]" (Offset 11): Bool

"Reserved [12]" (Offset 12): Bool

SetLeftPIN2asDO (Offset 13): Bool - Setting this bit sets the Left PIN2 to be used as Digital Output / 0 = Digital input

SetRightPIN2asDO (Offset 14): Bool - Setting this bit sets the Right PIN2 to be used as Digital Output/ 0 = Digital input

"Reserved [15]" (Offset 15): Bool

9.2.5. LeftMDRControl ***Struct*** This field allows the PLC to control the direction and On/Off state of the Left MDR.

MDR_Direction (Offset 0): Bool - 0 = Configured direction /1 = opposite to configured direction

Offsets 1 to 7 are reserved and not used.

Run_MDR (Offset 8): Bool - MDR Run = 1 / MDR Stop = 0

Offsets 9 to 15 are reserved and not used.

9.2.6. LeftMDRBrakeMode *Word* **Zero Protected**

0 – Left MDR uses the last set Brake method.

1 – Normal Brake (3 wires in short)

2 – Free

3 – Servo Brake

9.2.7. Reserved *Word*

9.2.8. RightMDRControl ***Struct*** This field allows the PLC to control the direction and On/Off state of the Right MDR.

MDR_Direction (Offset 0): Bool - 0 = Configured direction /1 = opposite to configured direction

Offsets 1 to 7 are reserved and not used.

Run_MDR (Offset 8): Bool - MDR Run = 1 / MDR Stop = 0

Offsets 9 to 15 are reserved and not used.

9.2.9. RightMDRBrakeMode *Word* **Zero Protected**

0 – Right MDR uses the last set Brake method.

1 – Normal Break (3 wires in short)

2 – Free

3 – Servo Break 1

4 – Servo Break 2

9.2.10. Reserved *Word*

9.2.11. LeftMDRSpeed *Word* **Zero Protected**

0 – Use last set Speed

40 to 6400 – set point for the Speed (0.04m/sec to 6.4m/sec). For MDR the unit base is mm. For PGD the unit base is in RPM/10(value of 1000 would be equal to 100 RPM).

9.2.12. RightMDRSpeed *Word* **Zero Protected**

0 – Use last set Speed

40 to 6400 – set Real Speed (0.04m/sec to 6.4m/sec). For MDR the unit base is mm. For PGD the unit base is in RPM/10(value of 1000 would be equal to 100 RPM).

9.2.13. LeftMDRAccel *Word* **Zero Protected**

0 – Use last set Acceleration. For MDR the unit base is mm. For PGD the unit base is pulses.

1-10000 (1 to 10000mm for acceleration).

 The minimum acceleration the MDR will accept is 30mm. Even if lower values are set, the MDR will use 30mm or 30 pulses for PGD.

9.2.14. LeftMDRDDeccel *Word* **Zero Protected**

0 – Use last set Deceleration. For MDR the unit base is mm. For PGD the unit base is pulses.

1-10000 (1 to 10000mm for deceleration)

9.2.15. RightMDRAccel *Word* **Zero Protected**

0 – Use last set Acceleration. For MDR the unit base is mm. For PGD the unit base is pulses.

1-10000 (1 to 10000mm for acceleration).

 The minimum acceleration the MDR will accept is 30mm. Even if lower values are set, the MDR will use 30mm or 30 pulses for PGD.

9.2.16. RightMDRDeccel *Word* **Zero Protected**

0 – Use last set Deceleration. For MDR the unit base is mm. For PGD the unit base is pulses.

1-10000 (1 to 10000mm for deceleration)

9.2.17. ClearMDRError *Word*

Setting 1 to this register clears the unrecoverable Short Circuit MDR error. This action takes effect on both Left and Right MDRs. Please note that a transition from 0 to 1 is needed to clear the error.

9.2.18. StatusToDownstream *Word* **Zero Protected**

From This word you may inform downstream ZPA module about product presence, state… See **Appendix C**

9.2.19. StatusToUpstream *Word* **Zero Protected**

From this word you may inform upstream ZPA module about product arrival, discharge… See **Appendix C**

9.2.20. TrackingToDownstream ***DWord***

together with StatusToDownstream you may use this ***DWord*** to send product tracking information to downstream ZPA module. See **Appendix C**

9.2.21. SensorPolarity *Struct*

You may use bits of Sensor polarity to switch the polarity of the signals on the sensor and control pins. Example is the use of retro-reflective photo-eyes. When there is NO product on the zone, PE will send ON signal to ConveyLinx and appropriate LED will be also ON. When there IS a product on the zone, PE will send OFF signal to ConveyLinx and appropriate LED will be OFF. To change this, you may switch the polarity of the PE signal (ON when there is product and OFF when there is no product). The switch will also affect the appropriate LED behavior.

Offsets 0 to 7 are reserved and not used.

Left_Pin2 (Offset 8): Bool - Normal polarity = 0 / Inverted polarity = 1

"Reserved [9]" (Offset 9): Bool

Right_Pin2 (Offset 10): Bool - Normal polarity = 0 / Inverted polarity = 1

"Reserved [11]" (Offset 11): Bool

Left_Pin4 (Offset 12): Bool - Normal polarity = 0 / Inverted polarity = 1

"Reserved [13]" (Offset 13): Bool

Right_Pin4 (Offset 14): Bool - Normal polarity = 0 / Inverted polarity = 1

"Reserved [15]" (Offset 15): Bool

9.2.22. ServoControlDistanceLeft *Word*

ServoControlCommandLeft ***Word***

ServoControlDistanceRight ***Word***

ServoControlCommandRight ***Word***

These words are used for servo (distance based) control or Left and Right MDRs. See **Appendix D**.

9.3. CLXPLCMINI_IN

9.3.1. AllSensors Struct

State of all sensors of the module. Same as **9.1.2**

9.3.2. SensorDetect Struct

Same as **9.1.3**

9.3.4. LeftCalcTemp *Byte*

Calculated temperature of Left MDR in deg. Celsius

9.3.5. ModuleTemp *Byte*

Temperature measured by CPU's thermo sensor in deg. Celsius

9.3.6. LeftMDRDiagnostic *Struct*

Same as **10.1.9**

9.3.7. RightCalcTemp *Byte* Same as 9.3.4, but for Right MDR.

9.3.8. ModuleTemp *Byte* Same as 9.3.5, but for Right MDR.

9.3.9. RightMDRDiagnostic *Struct* Same as 9.3.6, but for Right MDR.

9.3.10. LeftMDR_DIOstatus : Struct // This is the status when the MDR is used as a digital IO

Same as 10.1.15

9.3.11. RightMDR_DIOstatus: Struct //This is the status when the MDR is used as a digital IO

Same as 9.1.16

9.4. CLXPLCMINI_OUT

9.4.1. LeftMDRasDIO : Struct // Use the MDR as a digital IO

Same as 9.2.2.

9.4.2. RightMDRasDIO : Struct // Use the MDR as a digital

Same as 9.2.3.

9.4.3. SensorsDigitalOutputs Struct

Same as 9.2.4 .

9.4.4. LeftMDRControl Struct

Same as 9.2.5

9.4.5. RightMDRControl Struct

Same as 9.2.8 .

9.4.6. LeftMDRSpeed *Word* **Zero Protected**

Same as 9.2.11 .

9.4.7. RightMDRSpeed *Word* **Zero Protected**

Same as 9.2.12 .

9.4.8. ClearMDRError *Word*

Same as 9.2.17 .

9.5. CLXZPA_IN

9.5.1. StateUpstreamZoneInverce Byte

This Byte is used only in bi-directional conveyor operation.

9.5.2. StateUpstreamZone Byte

State of upstream zone of ConveyLinx in ZPA mode:

1 – **Empty** – there is no product on the zone and no product is arriving from upstream.

2- **Accepting** – Zone is empty, but product is arriving from upstream zone

4-**Full and Running** – There is a product on the zone and product is traveling to downstream

5-**Full and Stopped** – There is a product on the zone and it's accumulated there

6-**Zone busy** – this state is used for special purposes (example – during power up)

In states 1 and 2 sensor of the zone is unblocked.

In states 4 and 5 sensor of the zone is blocked.

In states 1 and 5 MDR of the zone is not running.

In states 2 and 4 MDR of the zone is running.

For more detail description of ZPA zone states see **Appendix C**.

9.5.3. StateDownstreamZoneInverce *Byte*

This Byte is used only in bi-directional conveyor operation.

9.5.4. StateDownstreamZone *Byte*

State of downstream zone of ConveyLinx in ZPA mode. Same as 9.3.2.

9.5.5. ArrivalCounterUpstreamZone *Word*

This word is incremented each time new product arrives on Upstream zone. When it reaches 65535, it overflows back to 0.

9.5.6. DisarrivalCounterUpstreamZone *Word*

This word is incremented each time a product disappears from Upstream Zone. When it reaches 65535, it overflows back to 0.

9.5.7. ArrivalCounterDownstreamZone *Word*

This word is incremented each time new product arrives on Downstream zone. When it reaches 65535, it overflows back to 0.

9.5.8. DisarrivalCounterDownstreamZone *Word*

This word is incremented each time a product disappears from Downstream Zone. When it reaches 65535, it overflows back to 0.

9.5.9. Diagnostic *Struct*

Diagnostic of module in ZPA module – Jams, MDR, Sensors…

See Appendix A.

9.5.10. TrackingUpstreamZone *DWord*

If there is a product on Upstream Zone, this DWORD contains product tracking data for it. For tracking manipulation, please see **Appendix C**.

9.5.11. TrackingDownstreamZone DWord

If there is a product on Downstream Zone, this DWORD contains product tracking data for it. For tracking manipulation, please see **Appendix C**.

9.5.12. ReleaseCounterUpstreamZone *Word*

This word is used to release product on upstream zone. See **CLXZPA_OUT.ReleaseControlUpstream (9.6.7)**

9.5.13. ReleaseCounterDownstreamZone *Word*

This word is used to release product on downstream zone. See **CLXZPA_OUT.ReleaseControlDownstream (9.6.8)**

9.5.14. ModuleDischargeTracking *DWord*

If this is the last module of MDR conveyor line, you can get from their product tracking information of last discharged product. See Example 4 in **Appendix C**.

9.5.15. AllSensors *Struct*

State of all sensors of the module. Same as **9.1.2**

9.5.16. Convey_stop_status *Struct*

ConveyStop status of the module. Same as **9.1.1**

9.6. CLXZPA_OUT

9.6.1. InductTrackingOnUpstreamZone *DWord*

You may use this DWord to induct product tracking for product on Upstream Zone. See **Appendix C** for examples.

9.6.2. InductTrackingOnDownstreamZone *DWord*

You may use this DWord to induct product tracking for product on Downstream zone. See **Appendix C** for examples.

9.6.3. AccumulateControlUpstream *Struct*

Accumulate (Offset 8) – Force Accumulation

On Upstream Zone of this ConveyLinx Module.

AccumUpstreamToThisZone (Offset 0) – Force Accumulation

On Upstream to this module. This bit is useful in merge/divert operations. When a product arrives on the Upstream zone and you start a divert operation, you may prevent the upstream module from sending you new product, until you finish the divert operation.

***FakeConfirm (Offset 1) – "Fake" confirmation. This bit is useful in divert operations when a product doesn't reach the Upstream zone and it is diverted to the spur zone. The PLC may use the spur zone sensor to copy the signal to the Upstream zone.

All other Offsets are reserved and not used.

9.6.4. AccumulateControlDownstream Struct

Same as 9.6.3. , but for the Downstream zone .

9.6.5. SpeedLeftMDR *Word* **Zero Protected**

Same as 9.2.11.

9.6.6 SpeedRightMDR *Word* **Zero Protected**

Same as 9.2.12.

9.6.7. ReleaseControlUpstream *Word*

This word is used to release product, accumulated on the Upstream zone.

To release a product and accumulate the next arrived, do the following:

- Force accumulation on the zone: AccumulateControlUpstream.Accumulate:=1

- When product arrives, do your job (for example induct tracking InductTrackingONUpstream := NUMBER

- Get current release counter from CLXZPA_IN, increment it and write it to ReleaseControlUpstream –

CLXZPA_OUT.ReleaseControlUpstream := CLXZPA_IN.ReleaseCounterUpstreamZone+1;

For more information about arrival/ release operations see **Appendix C**.

9.6.8. ReleaseControlDownstream *Word*

This word is used to release product, accumulated on Downstream zone. See **9.4.7**

9.6.9. InductControlState *Word* Zero Protected

This word can be used to force first zone of conveyor line to accept product (simulate wake-up PE). See **Appendix C**.

9.6.10. DishargeControlState *Word* Zero Protected

This word can be used to control last zone of conveyor line (simulate lane full signal). See **Appendix C**.

9.6.11. ModuleInductTrackingOnInductSide *DWord*

This Dword can be used together with InductControlState to wake up first zone of conveyor line and at the same time induct tracking for the product. See **Appendix C**.

9.6.12. ModuleInductTrackingOnDishargeSide *Dword*

This Dword can be used together with DischargeControlState to stop last zone of conveyor line and at the same time read tracking of the product. See **Appendix C**.

9.6.13. ClearMotorError *Word*

If there is a clearable motor error, a transition in this field of 0 to 1 would clear it.

9.6.14. Reserved *Word*

9.6.15. Reserved *Word*

9.6.16. Convey_stop_control *Word*

If ConveyLinx module belongs to ConveyStop group, writing 1 to this word will force Stop condition to all group. Writing 2 will clear Stop on all group, if all ConveyStop forcing conditions are cleared (All Stop buttons are in default (ON) state, module to module communication and PLC connections are OK).

9.6.17. JamClearUpstream Word

Clear Jam error on Upstream zone

9.6.18. JamClearDownstream Word

Clear Jam error on Downstream zone

9.6.19. GlobalDirectionControlUpstream" Word

Allows changing the direction, accumulation and mode of parts of the conveyor line.
The high byte specifies the number of **zones** that would be affected downstream. The low byte contains the command.
The values for the low byte are:

0 = Do nothing
1 = Accumulate
2 = Accumulate
3 = Change Accumulation mode of the conveyor line (Singulate or Train)
4 = Change Accumulation mode of the conveyor line back to configured one
10 = Set conveyor flow direction to configured direction (See limitations)
11 = Set conveyor flow direction to opposite to configured direction

Note: The two values for accumulation can be used to accomplish the accumulate-on-next functionality by changing the value of the field from 1 to 2 and from 2 to 1.

Limitations: The direction of the conveyor line can only be changed from the most upstream zone of the conveyor line. Attempting to reverse the flow from a zone that is not the most upstream will result in strange and unpredictable behavior.

9.6.20. **GlobalDirectionControlDownstream** Word

Same as 9.6.19. The direction of the conveyor should not be changed from here.

9.7. CLXZPAMINI_IN

9.7.1. StateUpstreamZoneInverce *Byte*

This Byte is used only in bi-directional conveyor operation.

9.7.2. StateUpstreamZone *Byte*

State of upstream zone of ConveyLinx in ZPA mode:

1 – **Empty** – there is no product on the zone and no product is arriving from upstream.

2 - **Accepting** – Zone is empty, but product is arriving from upstream zone

4 - **Full and Running** – There is a product on the zone and product is traveling to downstream

5 - **Full and Stopped** – There is a product on the zone and it's accumulated there

6 - **Zone busy** – this state is used for special purposes (example – during power up)

In states 1 and 2 sensor of the zone is unblocked.

In states 4 and 5 sensor of the zone is blocked.

In states 1 and 5 MDR of the zone is not running.

In states 2 and 4 MDR of the zone is running.

For more detail description of ZPA zone states see **Appendix C**.

9.7.3. StateDownstreamZoneInverce *Byte*

This Byte is used only in bi-directional conveyor operation.

9.7.4. StateDownstreamZone *Byte*

State of downstream zone of ConveyLinx in ZPA mode. Same as 9.7.2.

9.7.5. ArrivalCounterUpstreamZone *Word*

This word is incremented each time new product arrives on Upstream zone. When it reaches 65535, it overflows back to 0.

9.7.6. DisarrivalCounterUpstreamZone *Word*

This word is incremented each time a product disappears from Upstream Zone. When it reaches 65535, it overflows back to 0.

9.7.7. ArrivalCounterDownstreamZone *Word*

This word is incremented each time new product arrives on Downstream zone. When it reaches 65535, it overflows back to 0.

9.7.8. DisarrivalCounterDownstreamZone *Word*

This word is incremented each time a product disappears from Downstream Zone. When it reaches 65535, it overflows back to 0.

9.7.9. Diagnostic **Struct**

Diagnostic of module in ZPA module – Jams, MDR, Sensors…

See Appendix A.

Same as 9.5.9 .

9.7.10. ReleaseCounterUpstreamZone *Word*

This word is used to release product on upstream zone. See **CLXZPAMINI_OUT.ReleaseControlUpstream (9.8.6)**

9.7.11. ReleaseCounterDownstreamZone *Word*

This word is used to release product on downstream zone. See **CLXZPAMINI_OUT.ReleaseControlDownstream (9.8.7)**

9.7.12. AllSensors **Struct**

State of all sensors of the module. Same as **9.1.2**

9.7.13. Future **Array [11 .. 14] of Word**

9.8. CLXZPAMINI_OUT

9.8.1. AccumulateControlUpstream Struct

Same as 9.6.3 .

9.8.2. AccumulateControlDownstream Struct

Same as 9.6.3

9.8.3. SpeedLeftMDR *Word* **Zero Protected**

Same as 9.2.11 .

9.8.4. SpeedRightMDR *Word* **Zero Protected**

Same as 9.2.12 .

9.8.5. ReleaseControlUpstream *Word*

This word is used to release product, accumulated on Upstream zone.

To release a product and accumulate the next arrived, do the following:

- Force accumulation on the zone: AccumulateControlUpstream.Accumulate :=1

- When product arrives, do your job (for example induct tracking InductTrackingONUpstream := NUMBER

- Get current release counter from CLXZPA_IN, increment it and write it to ReleaseControlUpstream –

CLXZPA_OUT.ReleaseControlUpstream := CLXZPA_IN.ReleaseCounterUpstreamZone+1;

For more information about arrival/release operations see **Appendix C**.

9.8.6. ReleaseControlDownstream *Word*

This word is used to release product, accumulated on Downstream zone. See **9.8.6**

9.8.7. InductControlState *Word* **Zero Protected**

This word can be used to force first zone of conveyor line to accept product (simulate wake-up PE). See **Appendix C**.

9.8.8. DishargeControlState *Word* **Zero Protected**

This word can be used to control last zone of conveyor line (simulate lane full signal). See **Appendix C**.

9.8.9. ClearMotorError *Word*

9.8.10. Reserved *Word*

9.8.11. Reserved *Word*

9.8.12. JamClearUpstream *Word*

Clear Jam error on Upstream zone

9.8.13. JamClearDownstream

Clear Jam error on Downstream zone

9.8.14. GlobalDirectionControlUpstream *Word*

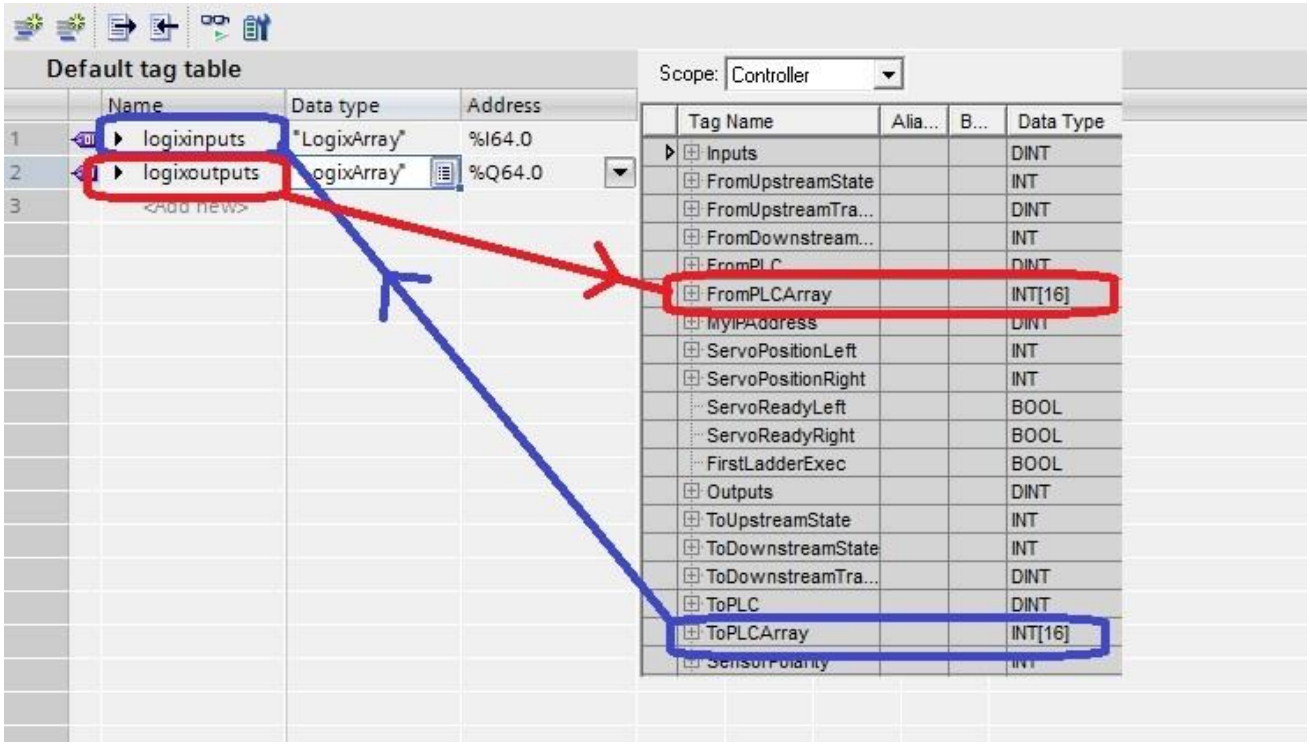
Same as 9.6.19 .

9.8.15. GlobalDirectionControlDownstream *Word*

Same as 9.6.20

9.9. Logix DAPs

For the ConveyLogix data, there are no UDTs. The reason is that when the module is in Logix mode, it exchanges two fixed data arrays with the PLC. The programmer will decide what kind of data is put into those data arrays. It cannot be pre-defined by us. When the PLC connects, it will send the Output data to the ConveyLogix tag named “FromPLCArray[16] of Int”. The PLC will receive the data from the ConveyLogix tag named “FromPLCArray[16] of Int”. The Int data type in ConveyLogix is 16-bit long signed tag. In the below picture, the LogixArray data type is Array[0-15] of Int. To the Left is the TIA Portal tag table, while to the right is the ConveyLogix Controller tags panel.



10. Reading from/Writing to ConveyLinx-Ai internal registers.

All data in ConveyLinx-Ai (both configuration and run-time) is organized as an array of 16-bit registers. You may, for example find power supply voltage in register No. 24. You may read the Current of Left MDR in register No. 55. For Right motor current you can read register No.79.

ConveyLinx registers are 512 (0<No<513).
You may read up to 50 registers at once.
To read any internal register you may use RDREC (read record) function block:

```
#RDREC_Instance(REQ:=NOT #TimerOn.Q,
                ID:=336,
                INDEX:=503,
                VALID=>#v,
                BUSY=>#b,
```

```
        ERROR=>#e,
        MLEN:=20,
        STATUS=>#RecordStatus,
        RECORD:=#RecordData);
```

In the function, INDEX is ConveyLinx-Ai register No.
MLEN is in bytes.

In the example above, RDREC reads 10 registers (20 bytes), starting from register 503 and puts the data in RecordData. RecordData is of type **Array [0..9] of Word**.

Writing to registers is a dangerous operation without deep understanding of ConveyLinx-Ai internal operation.
Because of this, writing to ConveyLinx is limited to 1 register (2 bytes) at a time only.

```
#WRREC_Instance(REQ:=NOT #TimerOn.Q,
        ID:=336,
        INDEX:=512,
        LEN:=2,
        STATUS=>#wrrecordstatus,
        RECORD:=#wrdata);
```

In the example above, WRREC writes data from Word tag wrdata to register 512.

11. Our support experience

In our support experience we have encountered several issues with our clients using ConveyLinx-Ai modules in PROFINET systems. Here we have excluded the most commonly occurring issue – an incorrect name set for the modules in the project. The incorrect name is usually only an issue, if DAPs with EasyRoll configuration are used. Below, we have listed the main problems and the solutions we offered for them:

Problem 1: In a system, where the S-7 PLC was connected to a SCALANCE managed switch and through it to the ConveyLinx-Ai system with topological configuration, the system did not configure.

Cause: When a Siemens PROFINET device (PLC, Switch, etc.) is added to a system, the engineering environment adds it with the default (for the environment) firmware version. However, this might not correspond to the actual firmware version of the device. For example TIA Portal V12 has a different default FW version for SCALANCE XF204 switch, than TIA Portal V13. In this case the switch was added to the project with FW version 5.0(TIA Portal V13 was used), while the switch had a lower FW version. The PLC saw, that the versions do not match and stopped the configuration process at the switch.

Solution: The firmware version of the real-world devices must match the one selected in the engineering environment. When the versions were adjusted the system ran as expected.

Problem 2: A client using S-7 PLC with two interfaces was unable to connect to the ConveyLinx-Ai devices.

Cause: The second interface of the PLC was not configured in the engineering environment (TIA Portal). The second interface had an IP address, which resided in the same subnet as the first interface. This is not allowed. The PLC was refusing to initiate any communication on both interfaces.

Solution: When the second interface was configured to have an IP address in a different subnet, everything worked fine.

Problem 3: A problem was reported, where after replacing the DAPs with EasyRoll configuration with the DAPs with PLC configuration in the PLC project, the system did not work.

Cause: The modules were used with EasyRoll configuration. This means that they did not have any names set in their non-volatile memory. Instead, they formed their own upon startup. When they were configured from the PLC, they were instructed by it from now on not to form their own name, but to read it upon startup from their non-volatile memory. Since no Name was previously set to the modules, upon startup the modules had empty names. Without a specified topology in the project, there was no way for the PLC to discover them.

Solution: Either the system needs to be reverted back to DAPs with EasyRoll configuration, or the topology has to be configured.

Problem 4: A client having previously used S-7 300 PLC series, switched to S-7 1500 series PLC. In project with the new PLC, no data was received with the GETIO/SETIO functions.

Cause: The GETIO/SETIO (and RDREC/WRREC) functions use different parameters for the input ID parameter, when used for different PLCs. For the 1500 series, the functions need to be passed the “Hardware Identifier” of the Input/Output module of ConveyLinx-Ai. For all other PLC series, the logical I/O address needs to be passed to those functions.

Solution: When the customer passed the HW identifier to the functions, instead of the logical address, the data was successfully read/written.

Industrial Software has also created several sample projects, which show how to use the functionalities of the ConveyLinx-Ai:

- A sample project for TIA Portal to operate a straight conveyor line, ending with a RAT.
- A sample project for STEP7 to operate a straight conveyor line, ending with a RAT.
- A sample project for STEP7 that gives an overview of the Servo functions of ConveyLinx-Ai.

All of the sample projects are accompanied by a brief description, fully commented and of course freely distributed.