Basic ProfiNet and Topology overview

1. Introduction

This document describes the basic ProfiNet features required for the integration and use of ConveyLinx-Ai modules in ProfiNet projects and systems . As such it is not a comprehensive guide , but rather a brief tutorial to explain the features used by ConveyLinx-Ai modules .The specific messages of the protocol will not be detailed , unless specifics are necessary for understanding the main points .

ConveyLinx-Ai has been certified for ProfiNet with FW version 4.1 for Conformance Class-A (CC-A).

2. ProfiNet overview and general idea

The ProfiNet I/O protocol consists of several communication protocols , that must be supported :

- DCP- Basic **D**iscovery and **C**onfiguration **P**rotocol – The PLC uses this protocol to find the I/O devices on the network , to set/reset their IP address , to set/reset their ProfiNet Name**.
- DCERPC - **D**istributed **C**omputing **E**nvironment / **R**emote **P**rocedure **C**alls – This protocol is used by the PLC to establish connection with the I/O devices and to asynchronously read/write data from/to the device . This protocol is transmitted over the UDP/IP protocols .
- ProfiNet RT – This is the realtime cyclic/acyclic data exchange protocol – Once the connection is established the PLC begins to exchange data cyclically with the I/O devices .While the cyclic exchange is running , the PLC can also issue “alarms” , which are non-cyclic .The alarms can be acknowledged or not .
- LLDP – **L**ink **L**ayer **D**iscovery **P**rotocol – This protocol allows the ProfiNet devices to gather data for the physical neighbors on their ports . This data allows the PLC to configure the system topologically .
- SNMP (not required for CC-A)

The general concept is that the PLC uses the DCP to discover all the devices it is looking for and set their Names and IP parameters .The the PLC then uses DCERPC to connect to them and establish the parameters of the connection . After this is completed , the PLC begins to exchange cyclic data with all devices it

has connected to . What data is exchanged is vendor –specific . The length of the data is given in the device description , which is imported from a GSDML*** file into the Engineering environment used to program the PLC .

**The ProfiNet Name will be discussed in the following chapter .
*** The GSDML will be discussed in chapter 6 .

3. ProfiNet device discovery

In a ProfiNet system each ProfiNet device has three important and unique(in the system) parameters :

- MAC address
- IP/Subnet Mask/Gateway
- ProfiNet Name

Upon initial startup it is possible (and permitted) that an I/O device only has a valid MAC address , with IP parameters == 0 and the Name of “”(aka “No name”) . In this case the PLC can discover the device only topologically , if the device supports it . If the I/O device does not support topology , then the PLC cannot find it and cannot configure it properly . In that case the Name of the device needs to be manually set from the engineering tool .

So let’s see how the PLC discovers the modules it wants to connect to .

Let’s assume , that the PLC is looking for three devices named “device1” , “device2” and “device3” . Upon start-up the PLC starts to broadcast on the network the following DCP messages :

- “I am looking for a device with the name of ”device1” “
- “I am looking for a device with the name of ”device2” “
- “I am looking for a device with the name of ”device3” “

All ProfiNet devices on the network will hear all of those messages and compare their own ProfiNet Name with the one the PLC is looking for. If the names match the device will answer, giving back its Name and other data like the type of device, ID, Role and IP. However, this method requires that all devices that the PLC needs already have their names set .If all three devices from above still do not have Names, they will not answer.

The ProfiNet protocol, does provide a way to solve this.

When the engineer is designing the system, he/she can tell the PLC what is the real world topology of the system .This can only be used if the devices, which are added to the project support topological discovery. For example, the engineer tells the PLC, that its Port1 is connected to Port1 of a device, which should have a name of “device1”. Let’s assume the Topology of the system looks like this:

PLC(port 1) ---> (port1) device1 (port2) --->(port1) device2 (port2) ---> (port1) device3

Now the PLC has more discovery options .Please note that the PLC also has a ProfiNet Name .For this example we will assume the Name of the PLC (which is given to the PLC by the engineering tool during programming) is “plc1”.

If the initial DCP broadcast message looking for a specific name has failed , the PLC waits for a little and then issues a slightly different DCP broadcast messages :

- **“I am looking for a device connected to port 1 of “plc1” “**
- “I am looking for a device connected to port 2 of “device1” “
- “I am looking for a device connected to port 1 of “device2” “
- “I am looking for a device connected to port 2 of “device2” “
- “I am looking for a device connected to port 1 of “device3” “

Now the device , which is connected to port 1 of the PLC **will answer** .Although it has no Name , it still knows that it is connected to port 1 of a device with Name “plc1” thanks to the LLDP protocol .The rest of the broadcast DCP messages will not be answered .

Now the PLC knows the MAC address of the device , which is connected to its port 1. This is enough for the PLC and it now sets the name of that device to “device1” .After the name is set , the PLC sets the IP address to the device . All this is done with the DCP protocol . After the Name and IP are set , the PLC will connect to the device and start to exchange realtime data .

So the first device is all good , but what about the other two ?

Well, while the PLC is setting the name , IP and connecting to “device1” , it is still issuing DCP broadcast messages to find the other two devices :

- **“I am looking for a device connected to port 2 of “device1” “**
- “I am looking for a device connected to port 2 of “device2” “
- “I am looking for a device connected to port 1 of “device3” “

Notice that the PLC is now not looking for “device1” anymore. So, what happens now? Now the device, which is connected to port 2 of the already configured “device1”, will answer .In turn the PLC will set its name as “device2”, set its IP and then connect.

The same thing will happen with the third device.

When the PLC has connected to all three device it will transition into “RUN” mode and the system will run as designed.

A very important question here:

“Does this procedure (the long , one-by-one topological discovery) happen every time upon start-up ?”

The answer is “usually no” .The PLC has the option to save the Name and the IP parameters “temporarily” or “permanently”.

- If the PLC tells the device to use the Name “temporarily”, then the device must save this Name in its volatile memory and reset (so – set to “”) its name in its non-volatile memory. So, when the device is reset (for example a power reset) it will start with a Name of “”(“No name”) .The same applies for the IP parameters . If they are to be used “temporarily” , then the device saves and uses them in its volatile memory and the IP parameters in the nonvolatile memory of the device are reset to 0 .
- If the PLC tells the device to use the Name “permanently” , then the device must set this name in both its volatile and non-volatile memory . So upon reset the name will not be changed .The applies for the IP parameters as well.

So , what does the PLC actually do ? It sets the Name of all devices “permanently” and the IP parameters “temporarily” .That way the devices in the system will start upon reset with their Names set (which speeds up the discovery phase considerably) and with their IP addresses set to 0 .

For ConveyLinx-Ai the topological discovery time is around 6s per device .If the Names of the ConveyLinx-Ai devices are set , it takes around 2s per device for the PLC to connect to all .

Bottom line :

- If a device does not support topological discovery , then its name needs to be set to be the same as the one the engineer has put in the project . Otherwise it will not be discovered and connected to .
- If the device supports topological discovery , there is no need for that , as the PLC can find and configure all devices to match the project topology .

4. Establishing the connection

After the PLC has discovered the device, set its Name and IP , then comes the Connect phase. The PLC attempts to connect to the device using the DCERPC protocol .There are three main sub-phases of the Connect phase .

- Connect request/response – With the Connect request begins the establishing of the connection. In this message the PLC defines the connections parameters to the device according to the imported as a GSDML device description . The most important parameters given in this message are :
    - The realtime data exchange interval .
    - The realtime data length
    - The timeout and datahold factors
    - The different Modules to be included in the realtime data and their statuses.
    - The Application relationship ID
    - The offsets of all data and statuses in the RT messages
- Write requests/responses – If the device requires initial configuration (specified in the GSDML file) , then the PLC can write parameters with the Write requests during the Connect phase .

- Control requests/responses – Once the Connect message has been sent and responded to , the Write requests (if any) have also been completed , the Connection is running and needs to be finalized . This means that both devices – the PLC and the I/O slave need to say that their Application (which is providing the RT data) is ready to give consistent RT(realtime) data .Both devices must sent Control requests in the following order :
    - PLC sends Control request
    - The device responds to the PLC's Control request
    - The device sends Control request
    - The PLC responds to the device's Control request

After the Control requests have been sent and responded to , the Connection is considered to be fully established and functioning .Now the PLC and the device exchange their RT data using the time interval specified during the Connect phase. If the PLC or the device does not receive RT message for an interval equal to "Data exchange interval" * "Timeout factor" it will abort the connection .

5. Real-time data exchange and use of the ConveyLinx as a distributed I/O

Once the device has been discovered ,configured and the connection with it has been established , the PLC enters into the real-time data exchange phase . This phase is cyclic. Both the PLC and the ConveyLinx-Ai send cyclic messages using the intervals provided by the PLC in the connect phase . Inside each RT message , there two main statuses – "Data Valid " Status and "I/O" status . If the "Data valid" status is 'bad' the message is ignored – the watchdog counter is incremented . If the "I/O" status is bad the message is accepted , but the data is ignored. The "I/O" status in the message from the PLC is bad, when the PLC is still connecting to a module , or is in Stop mode . It is important to note , that while the PLC is in "Stop" mode , the RT data is ignored by the ConveyLinx-Ai and its outputs are cleared . ConveyLinx-Ai supports RT exchange speeds between 4 and 512 ms .

6. GSDML file

In order for ConveyLinx-Ai to be added into any project in TIA Porta/Step7 engineering environments , the device description of ConveyLinx-Ai must be given (imported into) to the environment .The GSDML file , which is created and distributed by Industrial Software , contains the description .When installed in TIA Portal/Step7 an additional branch is added to the Hardware catalog in :

Other field devices -> PROFINET IO-> I/O

The ProfiNet protocol allows a single physical device to have multiple access ways .This is called a device access point or DAP. 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 six DAPs with different data lengths , configurations and topological discovery :

- With topological discovery and PLC configuration
    - 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 exchanges the 64 bytes long ZPA mode data instances with the PLC.
    - 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 exchanges the 64 bytes long PLC mode data instances with the PLC.
    - ConveyLinx-Ai in 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 exchanges the 16 bytes long PLC mode data instances with the PLC.

- With 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 .

- 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 .

The DAPs are organized in two directories :

- Conveyor Control with standard configuration
- Conveyor Control with topology and full PLC configuration

It is very important to note , that once a ConveyLinx-Ai device has been configured by a PLC , it will erase any previous configuration . Next time this device's power is turned off and on again , it will start in a special "dummy" mode . In this mode the device just awaits a PLC to connect to it and will not do anything .The ProfiNet name would be read from the non-volatile memory upon power-up . In order to be able to use the device from the EasyRoll the Auto-Config procedure must be triggered. If the device has been configured from the EasyRoll , it will not read its name from the non-volatile memory , but will instead form it itself.

Bottom line:

- A device from the directory "Conveyor Control with standard configuration" should not be used for a real-world device , that has not been configured from the EasyRoll.
- A device from the directory "Conveyor Control with topology and full PLC configuration" should not be used for a real-world device , that has been configured from the EasyRoll , as it will erase the previous configuration .

7. Using ConveyLinx-Ai RT data with the PLC and UDT files

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
- By copying the data to/from the I/O memory from/to local variables using the functions SETIO/GETIO .

Lets say , that the engineer has connected 1 ConveyLinx-Ai to the PLC . The ConveyLinx-Ai exchanges 64 bytes of data in ZPA mode . So the module has been allocated 64 bytes in the I/O memory by the PLC . Industrial Software provides documentation , describing what data is carried in every individual byte . However , copying the 64 bytes from the Input memory to a local variable of type “Array of Byte[0-63] “ makes it hard to keep track of what variable is on which offset .This makes it harder to write a PLC program without constantly consulting the documentation and calculating the offsets . Fortunately , the engineering environments allow for UDT(user data type) files to be imported . Those files contain data structures, where every byte (and even bits) are named and have comments attached to them to make it easier to use them when writing the PLC program. In our example from above , the Array of Byte [0-63] would be replaced by the provided in the UDT file structure type ‘CLXZPA_IN’ . Its length is 64 bytes and its members correspond to the logical structure of the data sent by the ConveyLinx-Ai .

The Siemens S-7 1500 (and newer 1200) series allows for a UDT PLC tag to be declared in the device’s I/O memory . S-7 1200(older FW revisions)/300/400 do not allow this .

Industrial Software has compiled two types of UDT files : for TIA Portal and for Step7.