CAN-PN

PROFINET IO / CAN Gateway

Software Manual

to Product C.2920.02

esd electronic system design gmbh
Vahrenwalder Str. 207 • 30165 Hannover • Germany
http://www.esd.eu
Phone: +49 (0) 511 3 72 98-0 • Fax: +49 (0) 511 3 72 98-68

N O T E

The information in this document has been carefully checked and is believed to be entirely reliable. **esd** makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors that may appear in this document. In particular descriptions and technical data specified in this document may not be constituted to be guaranteed product features in any legal sense.

esd reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance or design.

esd electronic system design gmbh
Vahrenwalder Str. 207
30165 Hannover
Germany

Phone:	+49-511-372 98-0
Fax:	+49-511-372 98-68
E-Mail:	info@esd.eu
Internet:	www.esd.eu

⚠	This manual contains important information and instructions on safe and efficient handling of the CAN-PN. Carefully read this manual before commencing any work and follow the instructions. The manual is a product component, please retain it for future use.

Document file:	I:\Texte\Doku\MANUALS\CAN\CAN-PN\Englisch\CAN-PN_Software_en_16.odt
Date of print:	2017-03-20

Firmware version:	from Rev. 2.01

Document History

The changes in the document listed below affect changes in the hardware as well as changes in the description of the facts, only.

Revision	Chapter	Changes versus previous version	Date
1.6	-	Classification of Notes inserted	2017-03-20
	3.	Notes on the configuration via TIA Portal and about the GSDML file inserted	
	4.	Notes on unpacking of the GSDML file inserted.	
	6.	New Chapter: “Using GSDML File and CAN-PN Gateway with the TIA Portal”	
1.5	1.2, 2.2, 5.3.2.2	Notes on automatic data transmission if Tx data changes, even if cyclic transmission is activated.	2015-02-03
	5.6.4.1, 5.1.6.2	Notes on 29 bit identifiers inserted	
1.4	4.	New GSDML-file	2014-01-02
1.3	2.2, 5.3.1	Information about in- and output data inserted	2011-09-01
	4.	New GSDML-file	
	5.3.2.2	Description of the value range added	
1.2	4.	New GSDML-file	2009-12-21
	5.6.2	New chapter inserted: "Communication Window Output / Input"	
1.1	2.2	Number of input and output bytes corrected.	2009-09-25
	4.	New GSDML-file	
1.0	all	First version	2009-09-01

Technical details are subject to change without further notice.

Classification of Notes

This manual contains noticeable descriptions for a safe use of the CAN-PN and important or useful information.

NOTICE

Notice statements are used to notify people on hazards that could result in things other than personal injury, like property damage.

NOTICE
This NOTICE statement contains the general mandatory sign and gives information that must be heeded and complied with for a safe use.

INFORMATION

INFORMATION
Notes to point out something important or useful.

Typographical Conventions

Throughout this manual the following typographical conventions are used to distinguish technical terms.

Convention	Example
File and path names	**`/dev/null`** or **`<stdio.h>`**
Function names	***open()***
Programming constants	`NULL`
Programming data types	`uint32_t`
Variable names	*Count*

Number Representation

All numbers in this document are base 10 unless designated otherwise. Hexadecimal numbers have a prefix of 0x. For example, 42 is represented as 0x2A in hexadecimal format.

Table of Contents

1. Overview

1.1 About this Manual

This manual describes the local firmware of the CAN-PN module. The local firmware controls the data exchange between PROFINET IO and CAN.

CAN Layer 2 Implementation
The manual describes the CAN Layer 2 implementation.

11-Bit- and 29-Bit CAN Identifier
The module CAN-PN supports 11-bit and 29-bit CAN identifier according to ISO 11898-1 (CAN2.0A/B).

1.2 Introduction into Functionality of the Firmware

The gateway simulates in the PROFINET network an IO device with a defined number of input and output bytes. After the gateway has been configured, CAN devices can be operated as PROFINET IO slaves.

The PROFINET IO output bytes are transmitted to the CAN bus.
1 to 8 output bytes are assigned to a CAN transmit identifier (Tx-identifier). The transmission is triggered if output data is changing. Additionally cyclic transmission of output data is selectable. In this case output data is send cyclically AND at each data value change.

CAN receive identifiers (Rx-identifiers) are assigned to the input bytes on CAN side. Received CAN data is processed as input data by the PROFINET IO.

1.3 Configuration via PROFINET IO

The CAN-PN module is configured via the PROFINET IO. The Siemens SIMATIC Manager for S7 or the TIA Portal by Siemens , for example, can be used as a configuration tool. Here, the gateway is assigned with logical slots (modules). During configuration further parameters such as the PLC address, data direction, data length and CAN identifier are assigned to these modules.

2. Functionality of the Local Firmware

The following figure represents the functionality of the firmware.

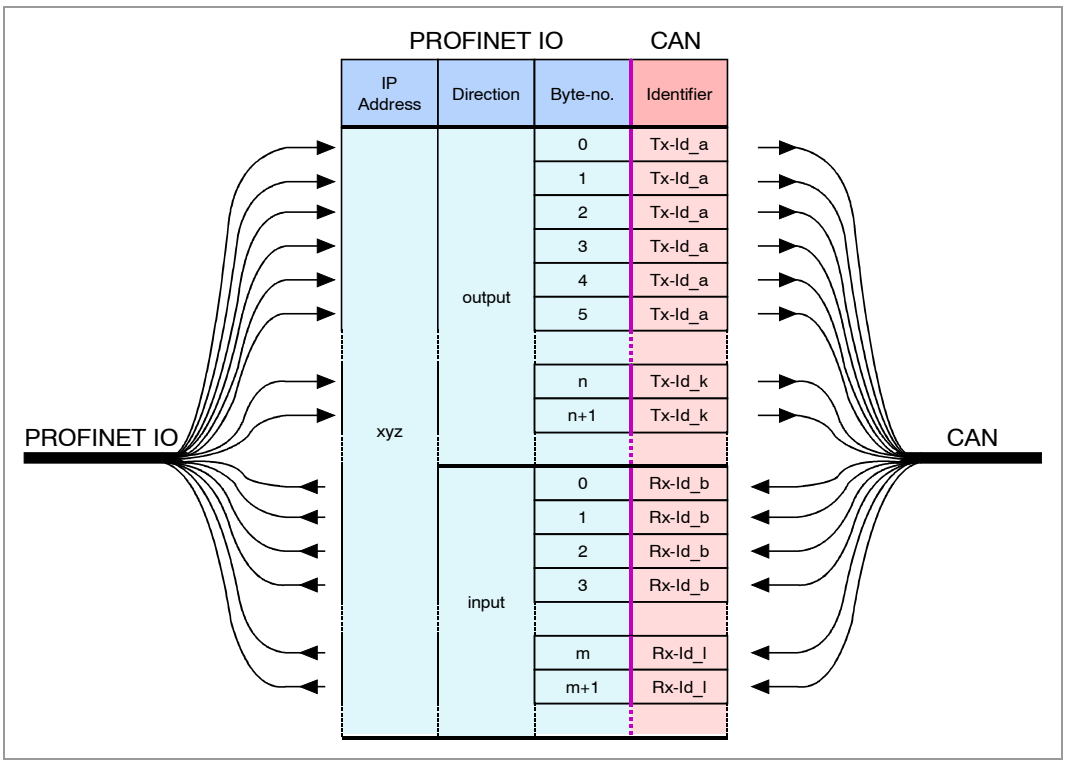


Fig. 1: Functionality of the local firmware

2.1 PROFINET IO IP-Address and Device Name

The CAN-PN module simulates an IO device on PROFINET IO side. The IP address and the device name are assigned by the IO supervisor.

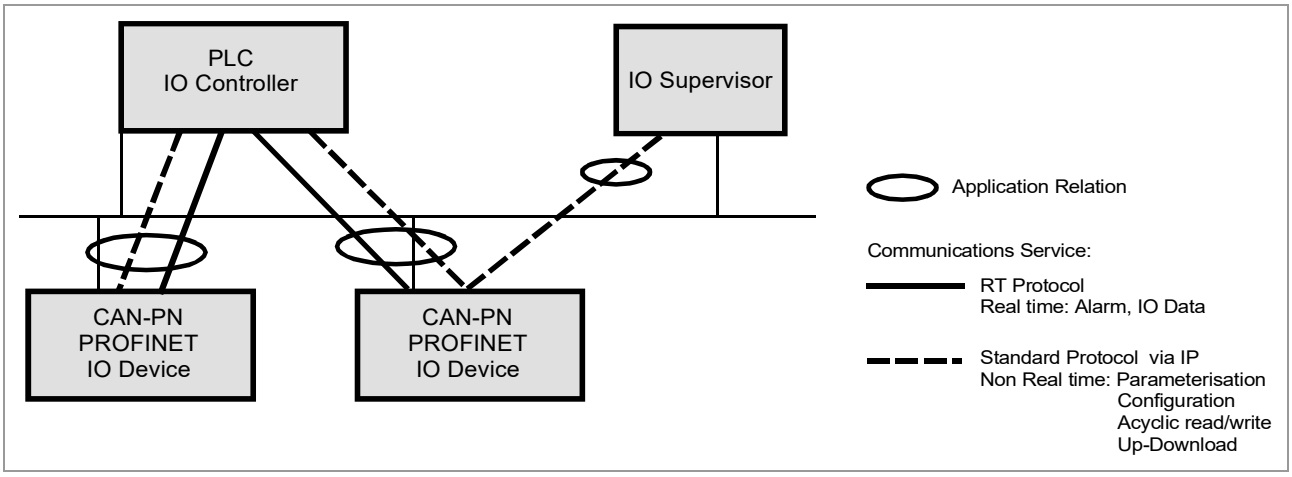


Fig. 2: PROFINET IO Communication

2.2 User Data

The module CAN-PN can link any PROFINET IO controller to a CAN network. The CAN-PN gateway itself operates as a PROFINET IO device with a maximum data transmission of 256 bytes input data and 256 bytes output data on the PROFINET IO.

NOTICE
The CAN-PN allows 256 bytes input data and 256 bytes output data.
For each input module 1 byte IOPS (IO-producer status) has to be added to the input data.
Each output module has 1 byte IOCS (IO-consumer status) in the output data and 1 byte IOPS in the input data.

For this reason the sum of input bytes + number of input modules + number of output modules must not exceed 256.
The sum of output bytes + number of output modules must not exceed 256 either.

One to eight bytes (16 bytes when using the communication window, see page 32) each are assigned to a Tx- or Rx-identifier. The same identifier cannot be used as Tx-and Rx-identifier at the same time.
The transmission of the Tx data is triggered if output data is changing. As an option additionally to this event triggered transmission cyclic transmission of output data is possible.

2.3 Diagnosis

The CAN-PN module is equipped with 4 LEDs in the front panel. The status of the LEDs can be analysed for the diagnosis (see page 11).
Furthermore an error on the CAN bus is indicated by sending an alarm telegram on the PROFINET IO. This alarm can be evaluated for diagnosis. For a detailed description of the alarm see page 13 et seq..

2.4 Parametrization of the CAN Bit Rate

The CAN bit rate is set during the parametrization of the PROFINET IO device (CAN-PN) by the PROFINET IO controller. Setting the bit rate is described on page 21.

2.5 PROFINET IO Profile

The PROFINET IO profiles are not being supported yet.

3. Implementing and Diagnosis

3.1 Prerequisites for Implementation

This chapter describes the implementation of the CAN-PN in PROFINET IO which is controlled by a Siemens SIMATIC-S7-300, S7-400, S7-1200 or S71500.
In order to be able to implement the module as described here, you need the configuration program 'SIMATIC-Manager' with the tool 'HW-configurator' or the TIA Portal.

NOTICE for CAN
First configure the CAN-PN with the PLC via the SIMATIC-Manager or the TIA Portal.
See page 18 for a description of the configuration with the SIMATIC-Manager.
The configuration with the TIA Portal is described from page 40.
Only after having carried out configuration, the CAN-PN can be identified as CAN device on the CAN bus!

3.2 Implementation

3.2.1 Procedure

Please make the following steps for implementation:

1.	Install and wire the CAN-PN (power supply, CAN bus) as described in the hardware manual.
2.	Connect the PROFINET IO connector to the PROFINET IO interface of the CAN-PN.
3.	Unzip the delivered GSDML file (`.ZIP` format) and save it in an appropriate directory.
4.	Configure the settings of the CAN-PN in the PLC via the SIMATIC manager or with the TIA Portal.
5.	Switch on the power supply for the CAN-PN. Now the module has to run. Set the IP address and the name of the CAN-PN. The module can be identified via the LEDs. The CAN-PN is now automatically parametrized via the PLC.

INFORMATION
Please note that in particular the CAN bit rate has to be set via the PROFINET IO.

3.2.2 Start-Up

After switching on the supply voltage, the CAN-PN starts automatically.

During start-up the LEDs “E” (PROFINET IO Link-Status), LED “P” (Firmware Bootup) and the LED “C" (CAN-Bus Status) turn on.
The LED "E" (PROFINET IO Link-Status) turns off when a valid PROFINET IO link has been established. The LEDs "P" and "C" remain on.

The CAN-PN now receives parametrization from the PN master and evaluates the specifications in it. If the projection complies with the structure, the CAN-PN starts the data transfer.

3.2.3 Data Transfer

If the device is configured, the data transfer starts automatically after start-up: If the PLC master changes transmission data of an identifier, the data is transmitted from the CAN-PN to the CAN bus. When the CAN-PN receives data, it provides these to the PROFINET IO controller.

The configuration with the SIMATIC manager is described from page 18.
The configuration with the TIA Portal is described from page 40.

3.3 Diagnosis via LED Indication

The indication of the LEDs is defined by the firmware.
The sequence of flashing, shown in the table below, is repeated approximately every 6 seconds.

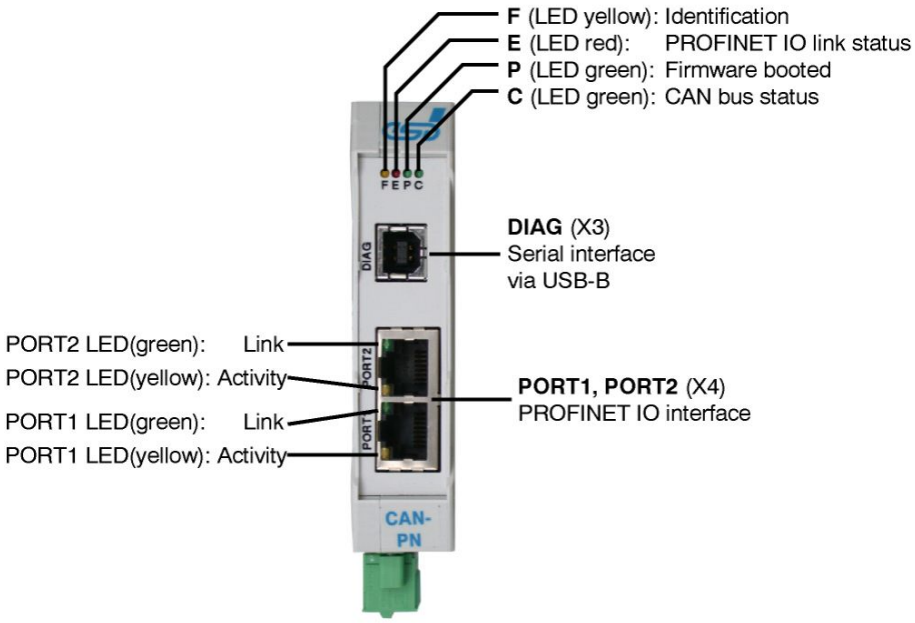


Fig. 3: Position of the LEDs in the front panel

LED	Colour	Function	Indication	Description	LED name in schematic diagram
F	yellow	Identification	off	no request of the Simatic-Manager for identification of the unit	LED1A
			blinking	request of the Simatic-Manager for identification of the unit	
E	red	PROFINET IO link status	off	valid PROFINET IO connection established	LED1B
			on	no valid PROFINET IO connection	
P	green	Firmware booted	off	firmware not already booted.	LED1C
			on	firmware booted.	
C	green	CAN bus status	off	no power supply	LED1D
			1x short flash	CAN Error (morse code "E")	
			3x short flash	CAN Off (morse code "O")	
			short-long-long	CAN Warning (morse code "W")	
			on	CAN bus OK	

Table 1: Indication of the LEDs

PROFINET-IO RJ45-Socket-LEDs to PORT1 and PORT2

LED	Colour	Indication	Description
Link	green	off	no PROFINET IO or Ethernet connection
		on	PROFINET IO or Ethernet connection established
Activity	yellow	off	no PROFINET IO connection
		blinking	phase of configuration
		on	PROFINET IO connection active

Table 2: Indication of the RJ45-LEDs

3.4 Alarm

An error of the CAN bus is indicated by a PN-alarm on the PROFINET IO in sending an alarm telegram. This alarm is indicated in the PROFINET IO controller on slot 0, subslot 1 - the module, to which the CAN bus is assigned to - and can be evaluated for the diagnosis.

The content of the alarm telegram is:

Octet number	Name	Content	
0	reserved	0x00	
1	btype	0x01 (Blocktype:Alarm Notification High)	
2	blen0	0x00	(Blocklength: 30 Byte)
3	blen1	0x1e	
4	verh(1)	0x01 (VersionHigh: 1)	
5	verl(0)	0x00 (VersionLow: 0)	
6	atyp0	0x00	(AlarmType: Diagnosis)
7	atyp1	0x01	
8	ap0	0x00	(API:0)
9	ap1	0x00	
10	ap2	0x00	
11	ap3	0x00	
12	sl0	0x00	(SlotNumber: 0)
13	sl1	0x00	
14	ssl0	0x00	(SubslotNumber: 1)
15	ssl1	0x01	
16	mid0	0x10	(ModuleID: 0x10000000)
17	mid1	0x00	
18	mid2	0x00	
19	mid3	0x00	
20	smid0	0x10	(SubModuleID: 0x10000000)
21	smid1	0x00	
22	smid2	0x00	
23	smid3	0x00	
24	a0dmcsss	0xA8	(AR Problem Indicator:1 Diagnosis Exist:1 Manufacturer Specific:0 Channel specific: 1 Sequence number: 0..2047 (hier 0)
25	ssssssss	0x00	
26	usi0	0x80	(Use Structure ID: 0x8000 = Channel related diagnosis)
27	usi1	0x00	
28	chn0	0x80	(Channel number: 0x8000)
29	chn1	0x00	
30	dddssmra	0x68 (Direction: 3 = I/O Specifier: 1 = Appears MaintenanceDemanded:0 MaintenanceRequired:0 Accumulative:0	
31	type	0x00 Type: 0=unspecified	
32	cet0	0x00	(ChannelErrorType = 0x06 = line break)
33	cet1	0x06	

The SIMATIC manager or the TIA Portal detects a line break in slot 0, subslot 1.

The CAN-PN transmits the complete alarm telegram. In the PLC only the content of octet-number 32 and 33, here 'line break', is displayed.

After ending the error the alarm telegram looks similar. Only the following entries are different:

Octet number	Name	Content	
:	:	:	
6	atyp0	0x00	(AlarmType: Diagnosis disappears)
7	atyp1	0x0C	
:	:	:	
24	a0dmcsss	0x00	(AR Problem Indicator:0 Diagnosis Exist:0 Manufacturer Specific:0 Channel specific: 0 Sequence number: 0..2047 (here 1, incremented by one)
25	ssssssss	0x01	
:	:	:	
30	dddssmra	0x10 (Direction: 0 = unspecified Specifier: 2 = Disappears MaintenanceDemanded:0 MaintenanceRequired:0 Accumulative:0	
:	:	:	
32	cet0	0x00	(ChannelErrorType = 0x00 = no error)
33	cet1	0x00	

The SIMATIC manager or the TIA Portal detects error-free operation (no error).

The CAN-PN transmits the complete alarm telegram. In the PLC only the content of octet-number 32 and 33, here 'no error', is displayed.

4. GSDML-File

The GSDML-file (Device master data-file) of the CAN-PN is described below. These data are solely described exemplary. The data in the electronic GSDML-file delivered are decisive. The GSDML file is delivered in .ZIP format. Unzip the file and save it in an appropriate directory.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<ISO15745Profile xmlns="http://www.profibus.com/GSDML/2003/11/DeviceProfile"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.profibus.com/GSDML/2003/11/DeviceProfile ..\xsd\GSDML-DeviceProfile-v2.3.xsd">
  <ProfileHeader>
    <ProfileIdentification>PROFINET Device Profile</ProfileIdentification>
    <ProfileRevision>1.00</ProfileRevision>
    <ProfileName>Device Profile for PROFINET Devices</ProfileName>
    <ProfileSource>PROFIBUS Nutzerorganisation e. V. (PNO)</ProfileSource>
    <ProfileClassID>Device</ProfileClassID>
    <ISO15745Reference>
      <ISO15745Part>4</ISO15745Part>
      <ISO15745Edition>1</ISO15745Edition>
      <ProfileTechnology>GSDML</ProfileTechnology>
    </ISO15745Reference>
  </ProfileHeader>
  <ProfileBody>
    <DeviceIdentity VendorID="0x015D" DeviceID="0x0001">
      <InfoText TextId="Profinet / CAN Gateway" />
      <VendorName Value="esd electronic system design gmbh" />
    </DeviceIdentity>
    <DeviceFunction>
      <Family MainFamily="Gateway" ProductFamily="CAN Profinet-IO" />
    </DeviceFunction>
    <ApplicationProcess>
      <DeviceAccessPointList>
        <DeviceAccessPointItem ID="CAN Bus" PNIO_Version="V2.2" PhysicalSlots="0..127"
ModuleIdentNumber="0x10000000" MinDeviceInterval="32" ImplementationType="ERTEC200" DNS_CompatibleName="CAN-PN"
ExtendedAddressAssignmentSupported="true" FixedInSlots="0" ObjectUUID_LocalIndex="1" RequiredSchemaVersion="V2.2"
MaxSupportedRecordSize="4068" ParameterizationSpeedupSupported="false" NameOfStationNotTransferable="true"
DeviceAccessSupported="true" >
          <ModuleInfo>
            <Name TextId="CAN-PN" />
            <InfoText TextId="Profinet / CAN Gateway" />
            <VendorName Value="esd electronic system design gmbh"/>
            <OrderNumber Value="C.2920.02"/>
            <HardwareRelease Value="130"/>
            <SoftwareRelease Value="2.1.1"/>
          </ModuleInfo>
          <SubslotList>
            <SubslotItem SubslotNumber="32768" TextId="TOK_Subslot_8000" />
            <SubslotItem SubslotNumber="32769" TextId="TOK_Subslot_8001" />
            <SubslotItem SubslotNumber="32770" TextId="TOK_Subslot_8002" />
          </SubslotList>
          <IOConfigData MaxInputLength="256" MaxOutputLength="256" />
            <UseableModules>
              <ModuleItemRef ModuleItemTarget="11_I1" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_I2" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_I3" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_I4" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_I5" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_I6" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_I7" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_I8" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_O1" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_O2" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_O3" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_O4" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_O5" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_O6" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_O7" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="11_O8" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_I1" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_I2" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_I3" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_I4" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_I5" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_I6" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_I7" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_I8" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_O1" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_O2" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_O3" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_O4" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_O5" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_O6" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="29_O7" AllowedInSlots="1..127" />
```

```
              <ModuleItemRef ModuleItemTarget="29_O8" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="CommWinIn" AllowedInSlots="1..127" />
              <ModuleItemRef ModuleItemTarget="CommWinOut" AllowedInSlots="1..127" />
            </UseableModules>
          <VirtualSubmoduleList>
            <VirtualSubmoduleItem ID="CAN Bus" SubmoduleIdentNumber="0x10000000">
              <IOData />
              <RecordDataList>
                <ParameterRecordDataItem Index="1" Length="1" TransferSequence="0">
                  <Name TextId="CAN bus parameters" />
                  <Const Data="0x00" ByteOffset="0" />
                  <Ref ValueItemTarget="CAN bitrate" DataType="Unsigned8" ByteOffset="0" DefaultValue="0"
AllowedValues="0..13" TextId="CAN bitrate" />
                </ParameterRecordDataItem>
              </RecordDataList>
              <ModuleInfo><Name TextId="CAN-PN" /><InfoText TextId="Profinet / CAN Gateway" /></ModuleInfo>
            </VirtualSubmoduleItem>
          </VirtualSubmoduleList>
          <SystemDefinedSubmoduleList>
            <InterfaceSubmoduleItem ID="IDS_CAN-PNI" SubslotNumber="32768" SubmoduleIdentNumber="0x0002"
SupportedRT_Classes="RT_CLASS_1" TextId="TOK_DAP_InterfaceModule" SupportedProtocols="SNMP;LLDP"
SupportedMibs="MIB2" NetworkComponentDiagnosisSupported="true" >
              <ApplicationRelations     NumberOfAdditionalInputCR="0" NumberOfAdditionalMulticastProviderCR="0"
NumberOfAdditionalOutputCR="0" NumberOfMulticastConsumerCR="0" PullModuleAlarmSupported="true">
                <TimingProperties SendClock="32" ReductionRatio="1 2 4 8 16 32 64 128 256 512" />
              </ApplicationRelations>
            </InterfaceSubmoduleItem>
            <PortSubmoduleItem ID="IDS_CAN-PNP1" SubslotNumber="32769" SubmoduleIdentNumber="0x0003"
MAUType="100BASETXFD" TextId="TOK_Port1" PortDeactivationSupported="true" LinkStateDiagnosisCapability="Up+Down"/>
            <PortSubmoduleItem ID="IDS_CAN-PNP2" SubslotNumber="32770" SubmoduleIdentNumber="0x0003"
MAUType="100BASETXFD" TextId="TOK_Port2" PortDeactivationSupported="true" LinkStateDiagnosisCapability="Up+Down"/>
          </SystemDefinedSubmoduleList>
          <Graphics>
            <GraphicItemRef Type="DeviceSymbol" GraphicItemTarget="1" />
          </Graphics>
        </DeviceAccessPointItem>
      </DeviceAccessPointList>
      <ModuleList>
        <ModuleItem ID="11_I1" ModuleIdentNumber="0x00000001">
          <ModuleInfo>
            <Name TextId="1 Byte Input 11bit" />
            <InfoText TextId="1 Byte Input 11bit" />
          </ModuleInfo>
          <VirtualSubmoduleList>
            <VirtualSubmoduleItem ID="11_I1" SubmoduleIdentNumber="0x00000001">
              <IOData IOPS_Length="1">
                <Input Consistency="All items consistency">
                  <DataItem DataType="OctetString" Length="1" TextId="CAN data" />
                </Input>
              </IOData>
              <RecordDataList>
                <ParameterRecordDataItem Index="1" Length="4" TransferSequence="0">
                  <Name TextId="CAN-Frame parameters" />
                  <Const Data="0x00,0x00,0x00,0x00" ByteOffset="0" />
                  <Ref DataType="Unsigned16" ByteOffset="0" DefaultValue="0" AllowedValues="0..2047" TextId="CAN-
ID 11bit (dec)" />
                  <Ref DataType="Unsigned8" ByteOffset="2" DefaultValue="0" TextId="Format byte (dec)" />
                  <Ref DataType="Bit" ByteOffset="3" BitOffset="0" DefaultValue="0" Changeable="true"
Visible="true" TextId="RTR Frames" />
                </ParameterRecordDataItem>
              </RecordDataList>
              <ModuleInfo>
                <Name TextId="1 Byte Input 11bit" />
                <InfoText TextId="1 Byte Input 11bit" />
              </ModuleInfo>
            </VirtualSubmoduleItem>
          </VirtualSubmoduleList>
        </ModuleItem>
        •
        •
        •

        •
        •
        •
      </ModuleList>
      <ValueList>
        <ValueItem ID="CAN bitrate">
          <Assignments>
            <Assign Content="0" TextId="1000 kbit/s" />
            <Assign Content="1" TextId="666.6 kbit/s" />
            <Assign Content="2" TextId="500 kbit/s" />
            <Assign Content="3" TextId="333.3 kbit/s" />
            <Assign Content="4" TextId="250 kbit/s" />
            <Assign Content="5" TextId="166 kbit/s" />
            <Assign Content="6" TextId="125 kbit/s" />
            <Assign Content="7" TextId="100 kbit/s" />
            <Assign Content="8" TextId="66.6 kbit/s" />
            <Assign Content="9" TextId="50 kbit/s" />
            <Assign Content="10" TextId="33.3 kbit/s" />
            <Assign Content="11" TextId="20 kbit/s" />
```

```
            <Assign Content="12" TextId="12.5 kbit/s" />
            <Assign Content="13" TextId="10 kbit/s" />
          </Assignments>
        </ValueItem>
      </ValueList>
      <GraphicsList>
        <GraphicItem ID="1" GraphicFile="GSDML-015D-0001-N" />
      </GraphicsList>
      <ExternalTextList>
        <PrimaryLanguage>
          <Text TextId="Profinet / CAN Gateway" Value="Profinet / CAN Gateway" />
          <Text TextId="CAN-PN" Value="CAN-PN" />
          <Text TextId="CAN bus" Value="CAN bus" />
          <Text TextId="CAN bus data" Value="CAN bus data" />
          <Text TextId="CAN bus parameters" Value="CAN bus parameters" />
          <Text TextId="CAN bitrate" Value="CAN bitrate" />
          <Text TextId="1 Byte Input 11bit" Value="1 Byte Input 11bit" />
          <Text TextId="2 Byte Input 11bit" Value="2 Byte Input 11bit" />
          <Text TextId="3 Byte Input 11bit" Value="3 Byte Input 11bit" />
          <Text TextId="4 Byte Input 11bit" Value="4 Byte Input 11bit" />
          <Text TextId="5 Byte Input 11bit" Value="5 Byte Input 11bit" />
          <Text TextId="6 Byte Input 11bit" Value="6 Byte Input 11bit" />
          <Text TextId="7 Byte Input 11bit" Value="7 Byte Input 11bit" />
          <Text TextId="8 Byte Input 11bit" Value="8 Byte Input 11bit" />
          <Text TextId="1 Byte Output 11bit" Value="1 Byte Output 11bit" />
          <Text TextId="2 Byte Output 11bit" Value="2 Byte Output 11bit" />
          <Text TextId="3 Byte Output 11bit" Value="3 Byte Output 11bit" />
          <Text TextId="4 Byte Output 11bit" Value="4 Byte Output 11bit" />
          <Text TextId="5 Byte Output 11bit" Value="5 Byte Output 11bit" />
          <Text TextId="6 Byte Output 11bit" Value="6 Byte Output 11bit" />
          <Text TextId="7 Byte Output 11bit" Value="7 Byte Output 11bit" />
          <Text TextId="8 Byte Output 11bit" Value="8 Byte Output 11bit" />
          <Text TextId="1 Byte Input 29bit" Value="1 Byte Input 29bit" />
          <Text TextId="2 Byte Input 29bit" Value="2 Byte Input 29bit" />
          <Text TextId="3 Byte Input 29bit" Value="3 Byte Input 29bit" />
          <Text TextId="4 Byte Input 29bit" Value="4 Byte Input 29bit" />
          <Text TextId="5 Byte Input 29bit" Value="5 Byte Input 29bit" />
          <Text TextId="6 Byte Input 29bit" Value="6 Byte Input 29bit" />
          <Text TextId="7 Byte Input 29bit" Value="7 Byte Input 29bit" />
          <Text TextId="8 Byte Input 29bit" Value="8 Byte Input 29bit" />
          <Text TextId="1 Byte Output 29bit" Value="1 Byte Output 29bit" />
          <Text TextId="2 Byte Output 29bit" Value="2 Byte Output 29bit" />
          <Text TextId="3 Byte Output 29bit" Value="3 Byte Output 29bit" />
          <Text TextId="4 Byte Output 29bit" Value="4 Byte Output 29bit" />
          <Text TextId="5 Byte Output 29bit" Value="5 Byte Output 29bit" />
          <Text TextId="6 Byte Output 29bit" Value="6 Byte Output 29bit" />
          <Text TextId="7 Byte Output 29bit" Value="7 Byte Output 29bit" />
          <Text TextId="8 Byte Output 29bit" Value="8 Byte Output 29bit" />
          <Text TextId="Communication Window" Value="Communication Window" />
          <Text TextId="Communication Window Input" Value="Communication Window Input" />
          <Text TextId="Communication Window Output" Value="Communication Window Output" />
          <Text TextId="CAN-ID 11bit (dec)" Value="CAN-ID 11bit (dec)" />
          <Text TextId="CAN-ID 29bit (dec)" Value="CAN-ID 29bit (dec)" />
          <Text TextId="CAN-Frame parameters" Value="CAN-Frame parameters" />
          <Text TextId="Format byte (dec)" Value="Format byte (dec)" />
          <Text TextId="RTR Frames" Value="RTR Frames" />
          <Text TextId="CAN data" Value="CAN data" />
          <Text TextId="Cycle time" Value="Cycle time" />
          <Text TextId="1000 kbit/s" Value="1000 kbit/s" />
          <Text TextId="666.6 kbit/s" Value="666.6 kbit/s" />
          <Text TextId="500 kbit/s" Value="500 kbit/s" />
          <Text TextId="333.3 kbit/s" Value="333.3 kbit/s" />
          <Text TextId="250 kbit/s" Value="250 kbit/s" />
          <Text TextId="166 kbit/s" Value="166 kbit/s" />
          <Text TextId="125 kbit/s" Value="125 kbit/s" />
          <Text TextId="100 kbit/s" Value="100 kbit/s" />
          <Text TextId="66.6 kbit/s" Value="66.6 kbit/s" />
          <Text TextId="50 kbit/s" Value="50 kbit/s" />
          <Text TextId="33.3 kbit/s" Value="33.3 kbit/s" />
          <Text TextId="20 kbit/s" Value="20 kbit/s" />
          <Text TextId="12.5 kbit/s" Value="12.5 kbit/s" />
          <Text TextId="10 kbit/s" Value="10 kbit/s" />
           <Text TextId="TOK_Subslot_8000" Value="X1" />
           <Text TextId="TOK_Subslot_8001" Value="P1" />
           <Text TextId="TOK_Subslot_8002" Value="P2" />
           <Text TextId="TOK_DAP_InterfaceModule" Value="Interface" />
           <Text TextId="TOK_Port1" Value="Port 1"/>
           <Text TextId="TOK_Port2" Value="Port 2"/>
        </PrimaryLanguage>
      </ExternalTextList>
    </ApplicationProcess>
  </ProfileBody>
</ISO15745Profile>
```

5. Configuration with the SIMATIC Manager

5.1 Course of Configuration

The CAN-PN is configured via PROFINET IO.

Please follow the 7 steps below to configure the CAN-PN:

INFORMATION
Without correct configuration via the SIMATIC manager the CAN-PN and the CAN participants connected do not operate together. Thus, the operation of the CAN participants connected can be disturbed.
In particular the CAN bit rate configured in the CAN-PN must match the settings of the CAN participants connected!
If problems should occur, further information can be obtained by diagnosis as described in the chapters “3.3 Diagnosis via LED Display” and “3.4 Alarm”.

Step		see chapter	see page
1.	**Select CAN-PN** Select the esd CAN-PN in the menu tree of the Hardware Catalogue.	5.1.1	19
2.	**Set PROFINET IO device name in the project**	5.1.2	20
3.	**Parametrization of the CAN bit rate**	5.2	21
4.	**Assignment of the in-/output modules of the IO devices**	5.3.1	23
5.	**Parametrization of the in-/output modules**	5.3.2	25
6.	**Save project settings on hard disk**	5.4	30
7.	**Assign device name**	5.5	30

5.1.1 Insert CAN-PN to PROFINET IO

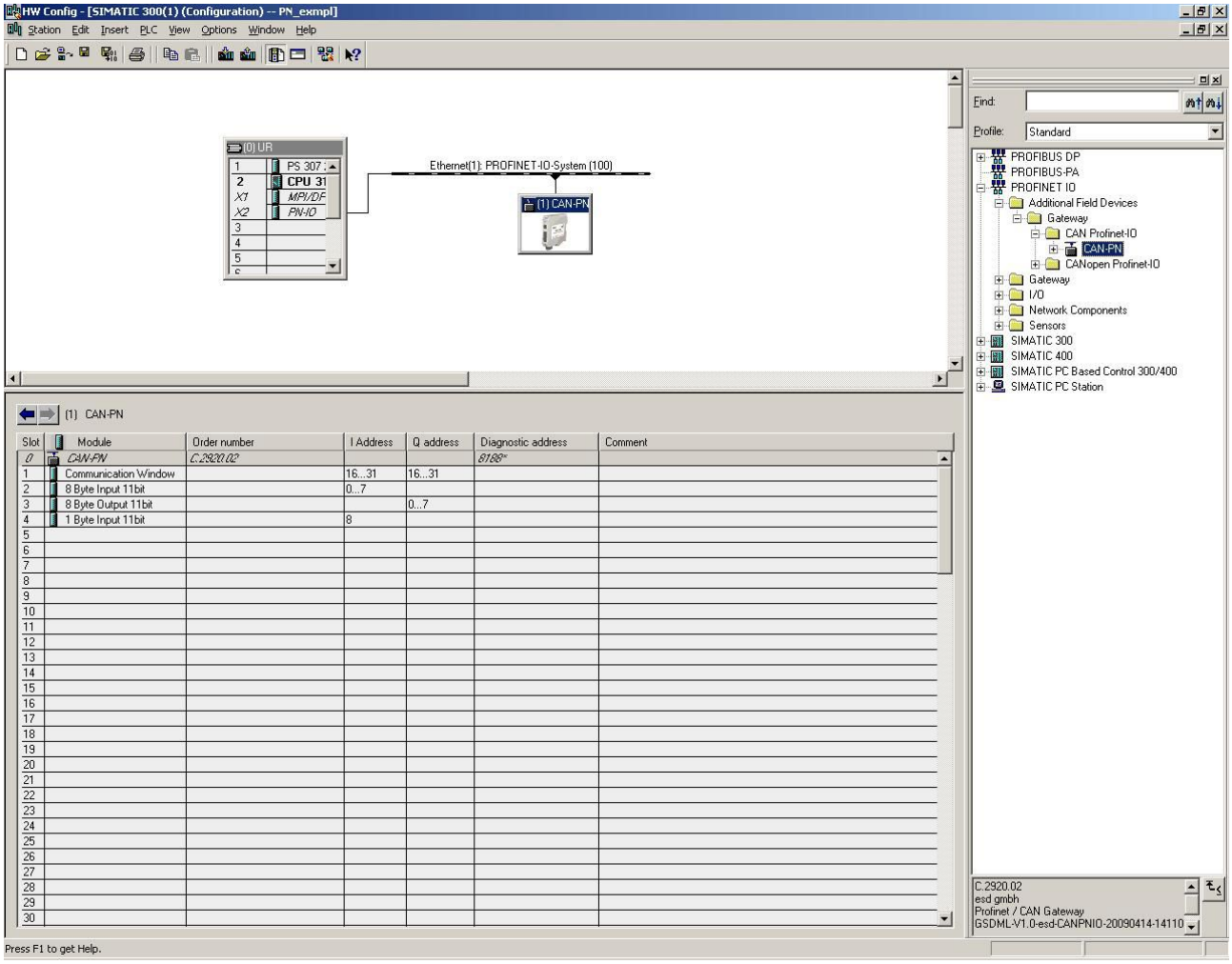


Fig. 4: Selection of CAN-PN

Click on the directory *PROFINET IO* in the menu tree of the Hardware Catalogue, then the subdirectories *Additional Field Devices* and *Gateway.* Select the esd CAN-PN under *CAN-PROFINET-IO.*

Click the CAN-PN with the left mouse button. Hold the mouse button and drag the CAN-PN to the left in the upper window area to the PROFINET IO system.

The symbol of the CAN-PN is now shown in the upper window area. The CAN-PN is entered in the list of the selected modules.

5.1.2 Specify a Device Name in the Project

Double click the symbol of the CAN-PN to open the *Properties* window.

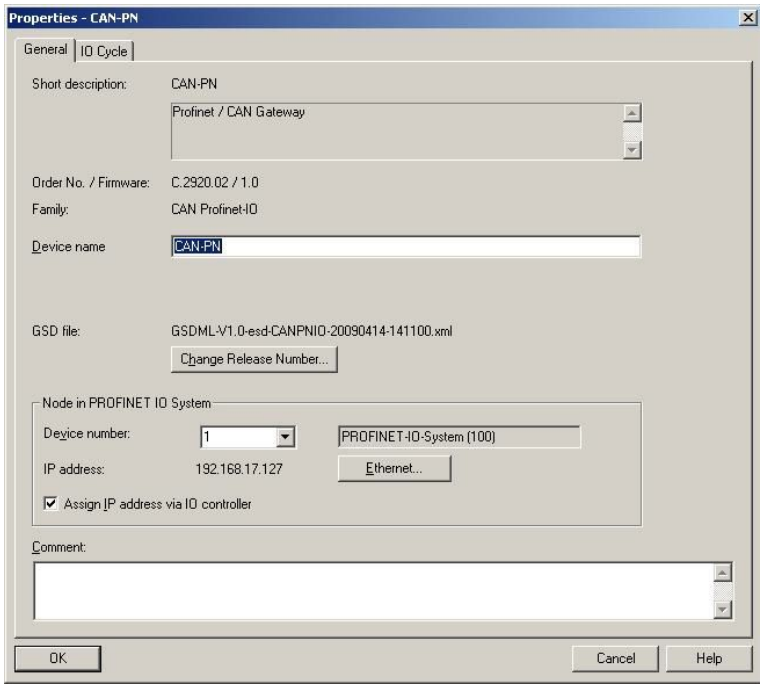


Fig. 5: Properties CAN-PN

In this window the following entries can be made:

Device name: In this input field you can specify a name for the PROFINET IO device. For the CAN-PN the device name entered per default is CAN-PN. The device name is assigned to the CAN-PN gateway for this project.

Device number: Select a free device number (1-128)

Assign IP-address via IO controller By selecting this check box the IP-address is assigned automatically.

Ethernet... To specify an IP-address manually, click the *Ethernet* button. The window *Properties Ethernet interface* opens. Select the register *Parameter* and enter the IP-address.

Comment: A comment can be entered here.

5.2 Parametrizing the Bit Rate of the CAN Bus

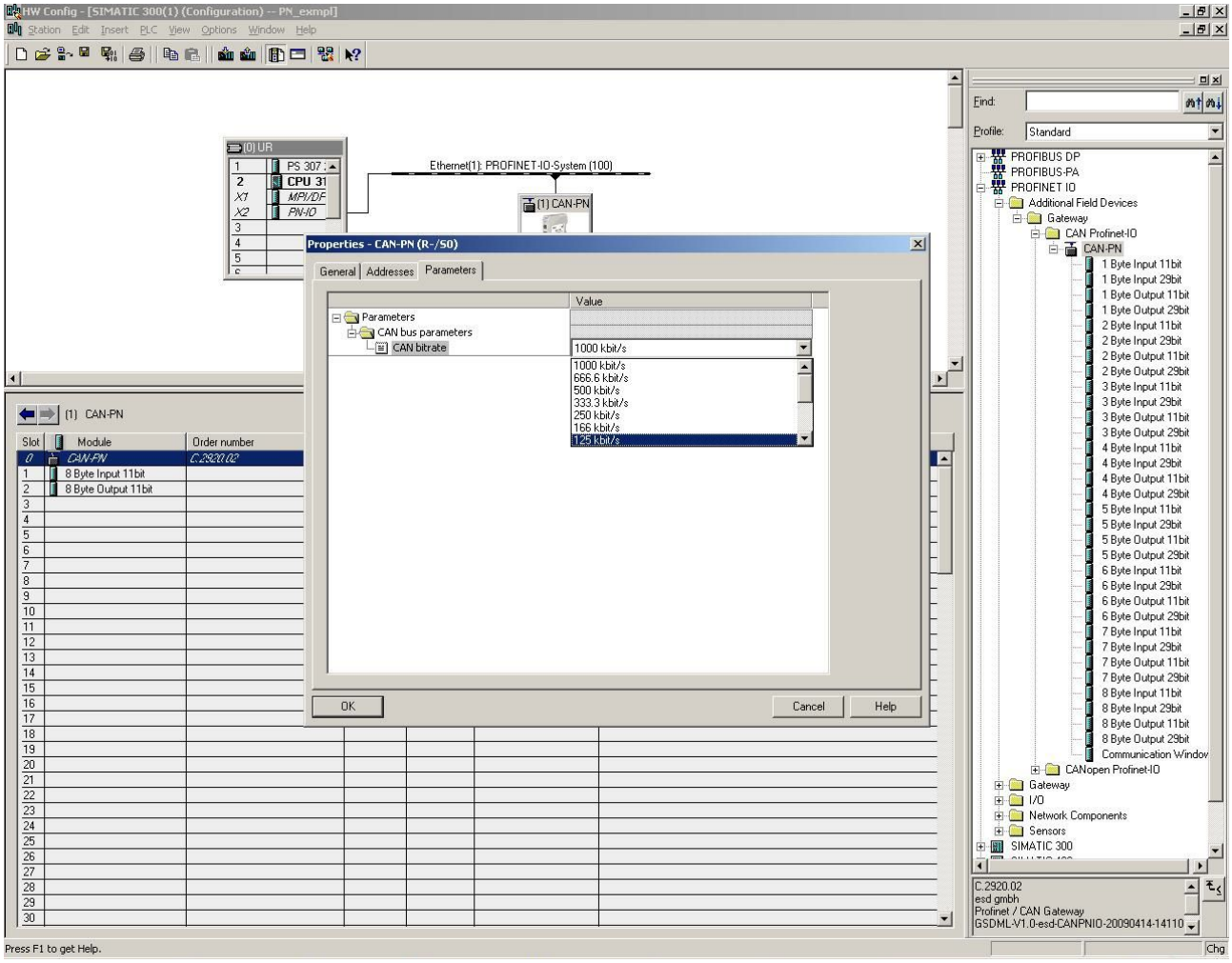


Fig. 6: Parametrizing the bit rate of the CAN bus

To configure the CAN bit rate of the CAN interface of the CAN-PN double click to CAN-PN (the first row in the example above) in the list of the selected modules.

The window *Properties* of the selected IO device CAN-PN opens.

The window contains the following register :

- *General* short information, containing name, comment, order number, hardware- and software- version

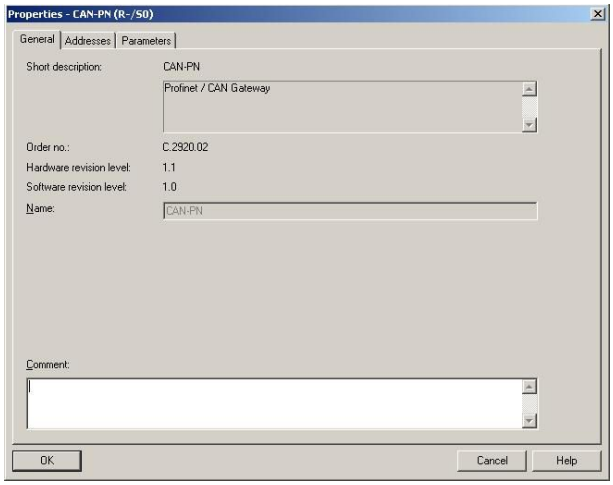


Fig. 7: Properties / General

- *Address* Diagnosis address

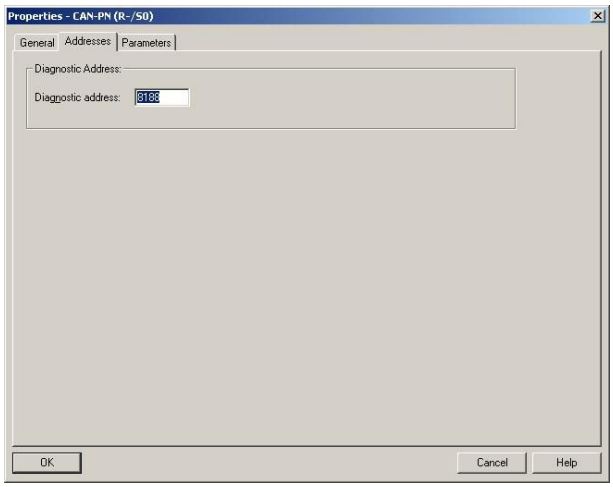


Fig. 8: Properties / Address

- *Parameter* CAN bus parameter, set the CAN bit rate

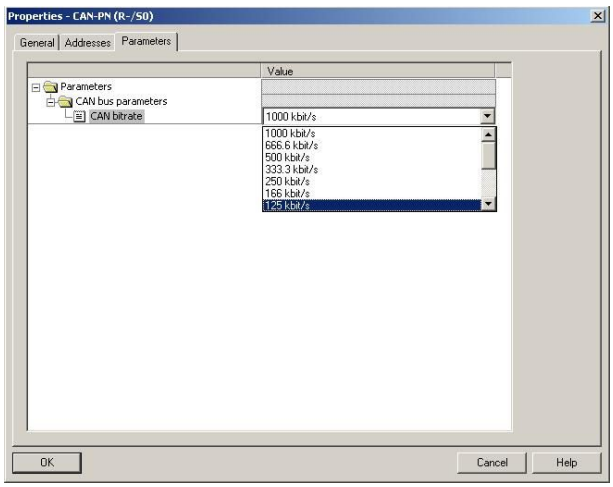


Abb. 9: Properties / Parameter

Click the register *Parameter* and select the chosen CAN bit rate from the list.

5.3 Input and Output Modules

5.3.1 Select Input and Output Modules

If you have given a name to the IO device identified, the modules can be assigned to the slots of the IO device.
A module corresponds to a CAN telegram that can be transmitted (output) or received (input).
Input or output modules with 11-bit or 29-bit-identifier can be chosen.

To get a list of the modules available, open the subdirectory of the IO device in the menu tree of the profile.

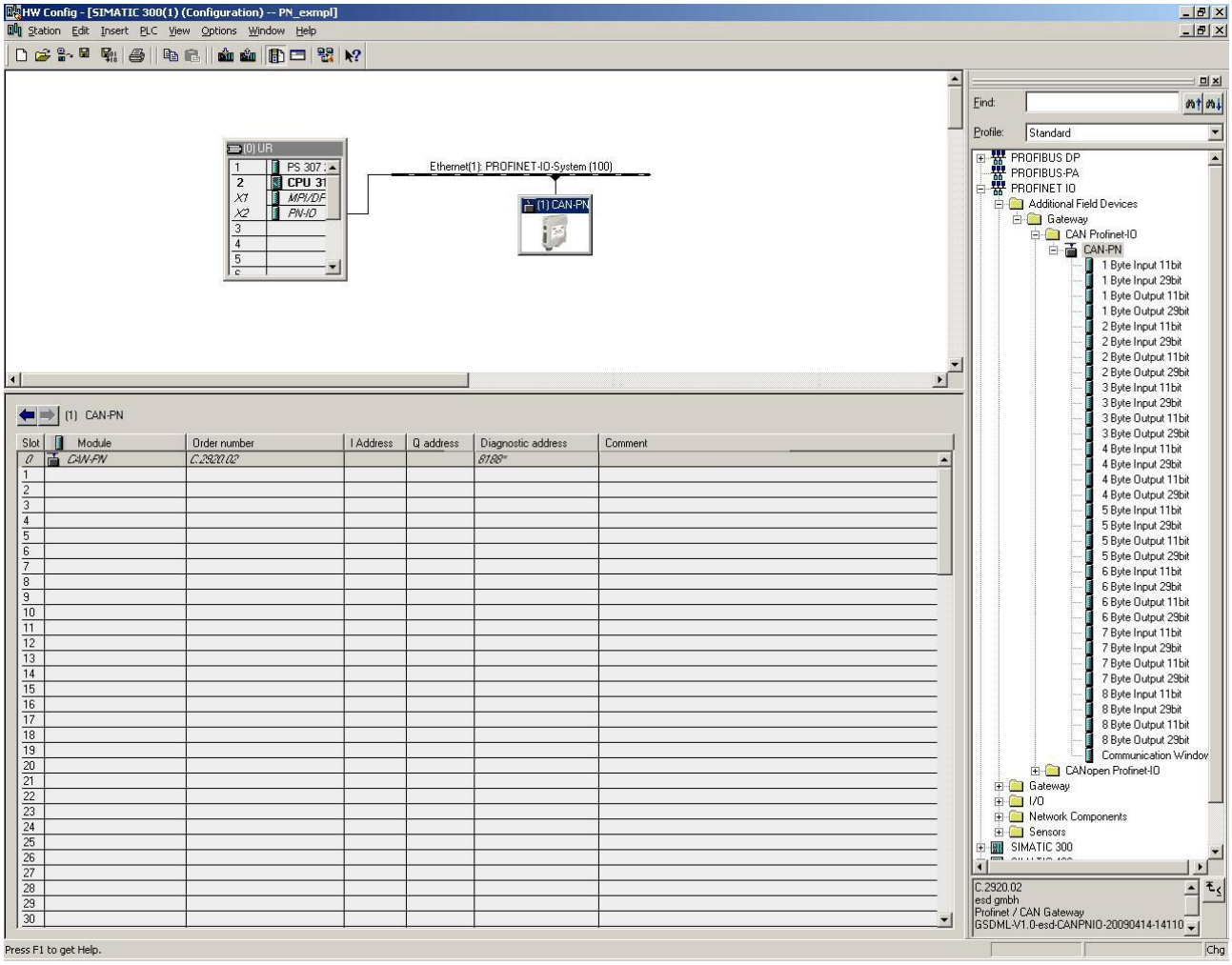


Fig. 10: List of the modules of an IO device

Click the directory *PROFINET IO* in the menu tree of the *Hardware Catalogue*, then the subdirectories *Additional Field Devices* and *Gateway.* Select the esd CAN-PN under *CAN-PROFINET-IO*. In the subdirectory of the CAN-PN all available modules are listed.

Click a module with the left mouse button. Hold the mouse button and drag the module to the left on the chosen slot entry in the list of the selected modules.

Configuration with the SIMATIC Manager

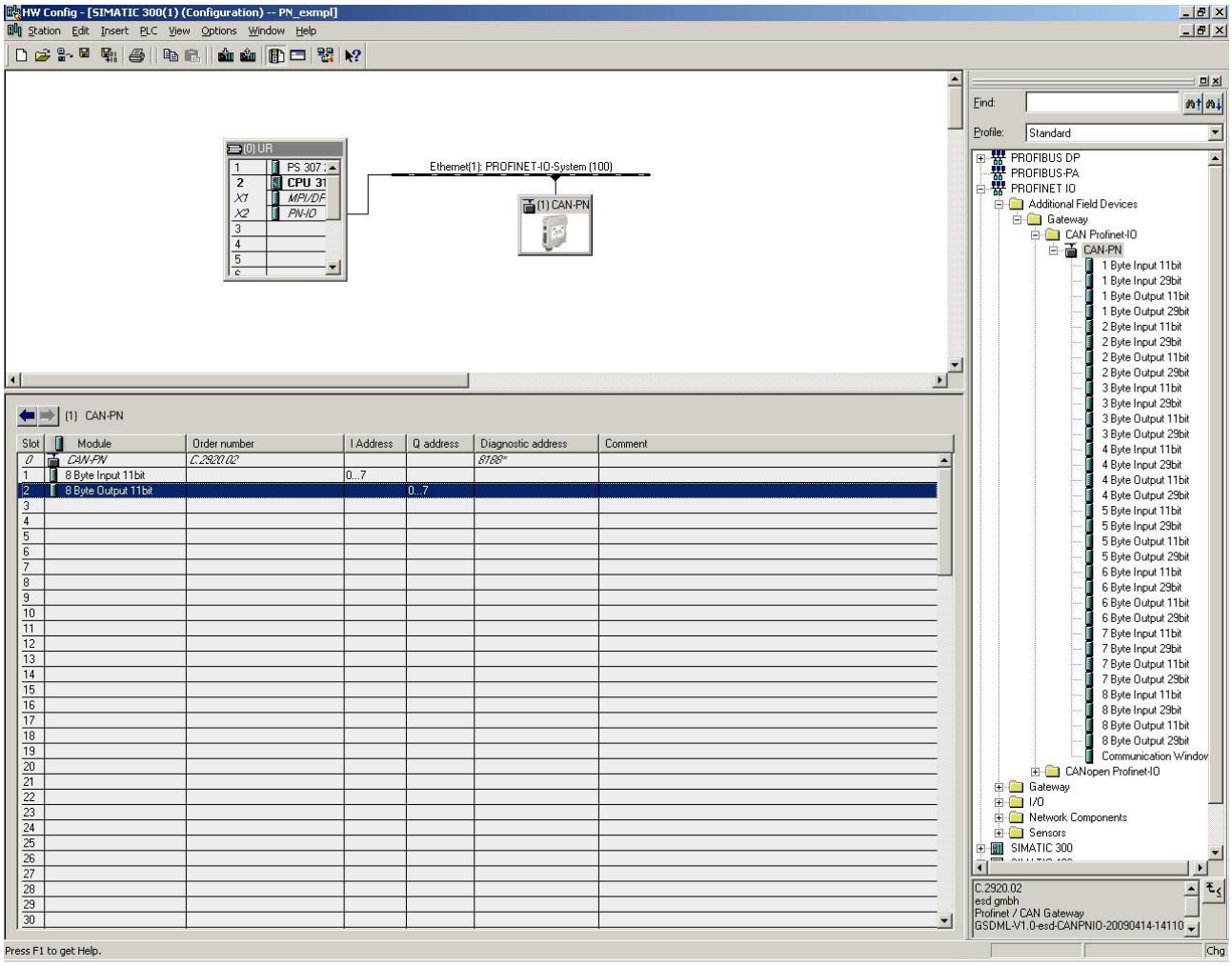


Fig. 11: Example of an IO device with selected modules

In this example an 8-byte input module and an 8-byte output module each with 11-bit identifier are assigned to the IO device.

NOTICE
The CAN-PN allows 256 bytes input data and 256 bytes output data.
For each input module 1 byte IOPS (IO-producer status) has to be added to the input data.
Each output module has 1 byte IOCS (IO-consumer status) in the output data and 1 byte IOPS in the input data.

For this reason the sum of input bytes + number of input modules + number of output modules must not exceed 256.
The sum of output bytes + number of output modules must not exceed 256 either.

5.3.2 Parametrization of the Input/Output Modules

To parametrize a selected I/O-module double click the slot entry of the module in the list.
The *Properties* window of the selected module opens, in which the simulated PLC slots can be parametrized.

The *Properties* window contains the following register :

- *General* short information, the name of the I/O module can be specified here.

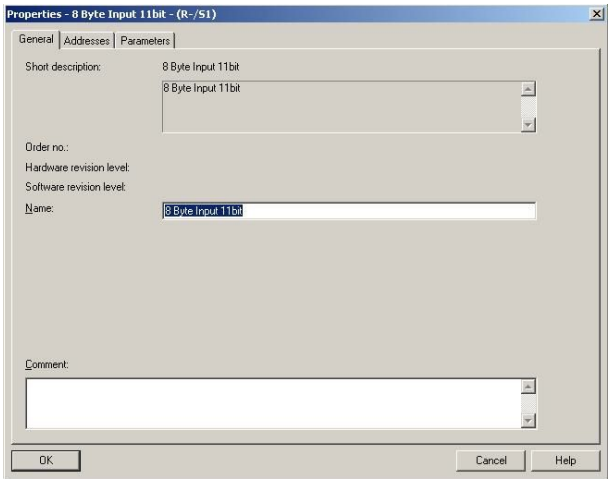


Fig. 12: General/8 byte input 11 bit (example)

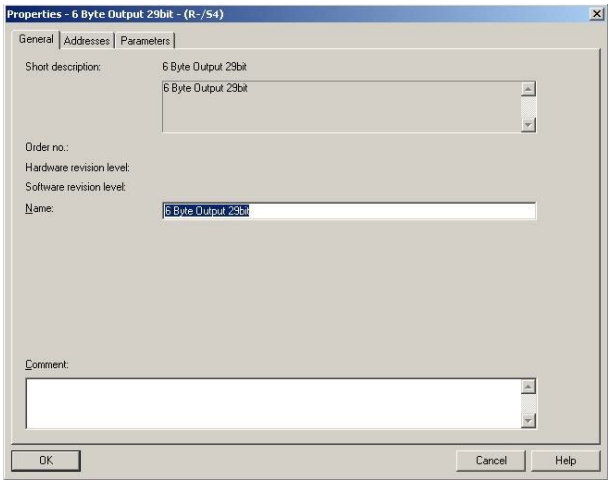


Fig. 13: General/6 byte output 29 bit (example)

- *Addresses* I/O-address, can be modified

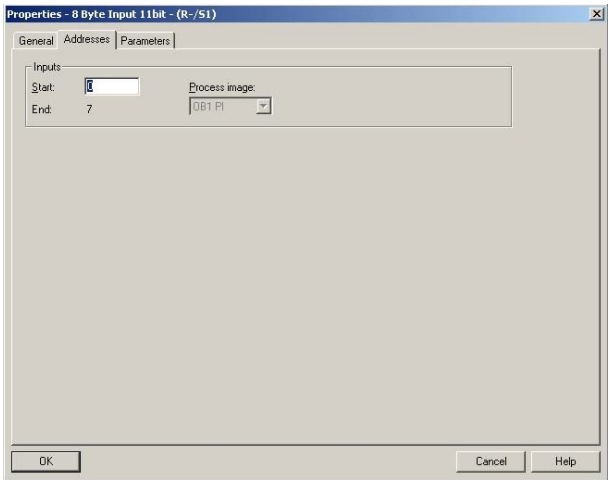


Fig. 14: Addresses/8 byte input 11 bit (example)

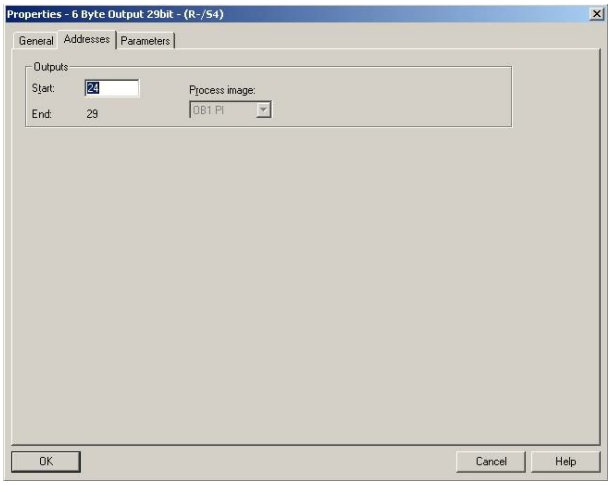


Fig. 15: Addresses/6 byte output 29 bit (example)

- ***Parameters*** CAN-Frame parameters: Set CAN identifier (CAN-ID), Format byte, RTR Frames (input module) or Cycle time (output module)

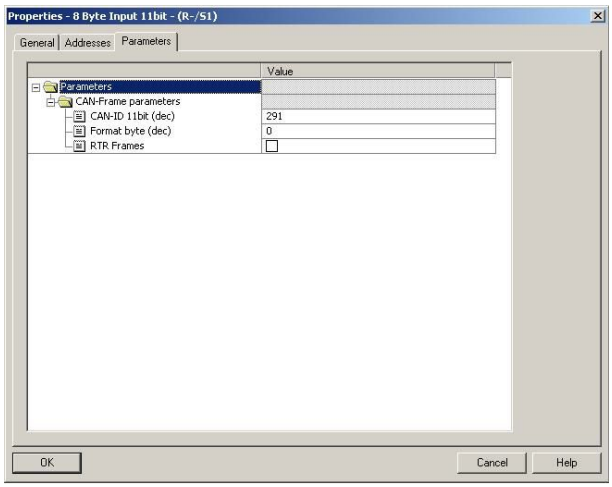


Fig. 16: Parameters/ 8 byte input 11 bit (example)

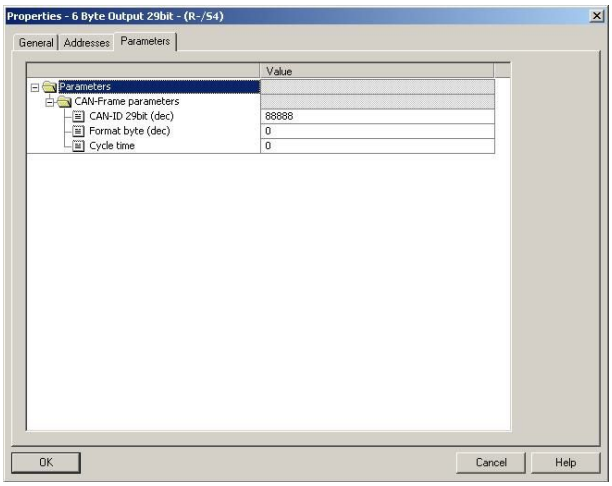


Fig. 17: Parameters/ 6 byte output 29 bit (examples)

5.3.2.1 Parameter of Input Modules

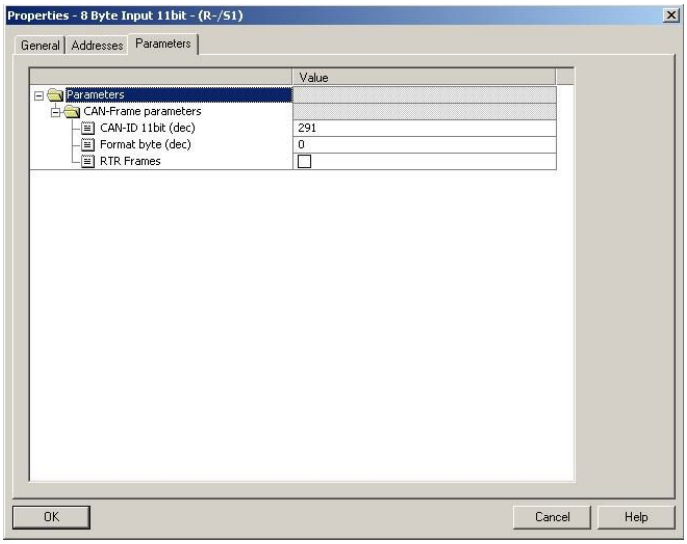


Fig. 18: Example: 8 byte input module with 11-bit identifier

CAN-Frame parameters:

CAN-ID (11bit) CAN identifier to be received,
value range of the CAN-ID 11-bit (decimal).: 0...2047

Format byte describes an optionally required byte-swapping (see page 29)
for word and long word data

RTR Frames When this check box is activated, the CAN-PN will request the current data of the associated CAN participant via an remote-frame when switched on

The description of the *CAN-Frame parameters* applies accordingly for input modules with 29-bit identifier.
The bit 29, usually set to distinguish between 11-bit and 29-bit identifiers (as defined by CANopen), is not specified here but is automatically added by the CAN-PN:
Value range of the CAN-ID 29-bit (decimal): 0...536870911

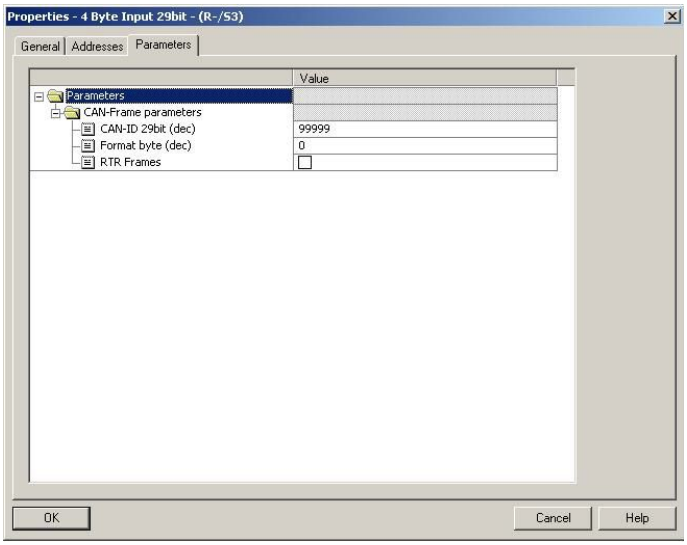


Fig. 19: Example: 4 byte input module with 29-bit identifier

5.3.2.2 Parameter of Output Modules

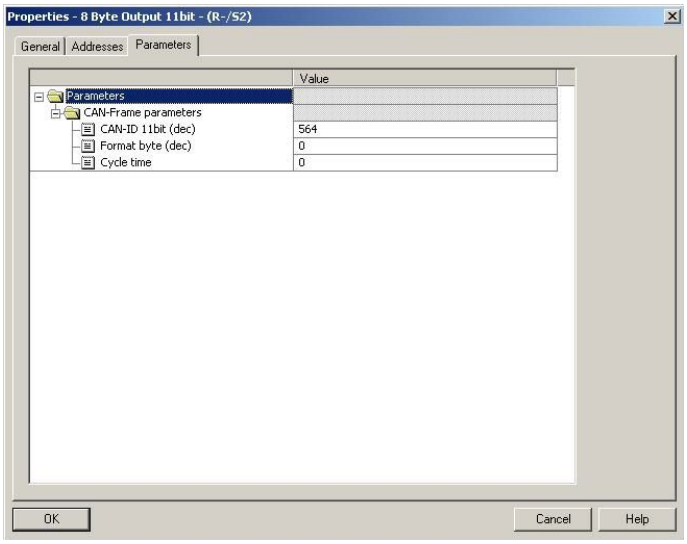


Fig. 20: Example: 8 byte output module with 11-bit identifier

CAN Frame Parameters:

CAN-ID (11bit) CAN identifier to be transmitted.
Value range of the CAN-ID 11-bit (decimal).: 0...2047

Format byte describes an optionally required byte-swapping (see page 29) for word and long word data

Cycle time A cycle time can be specified in milliseconds (0 = no cyclic transmission).
When the PLC is operating, the parametrized CAN telegram will be transmitted cyclically after every time interval specified in *Cycle time*.
Value range C*ycle time [ms]*: 0 (no cyclic transmission) or 10...65535
Values between 1 and 9 are interpreted as 0.

INFORMATION
Even if cyclic data transmission is selected the data is additionally transmitted with every change of Tx data.

The description of the CAN-Frame parameters of output modules with 11-bit identifier applies accordingly for output modules with 29-bit identifier. Only the value range of the 29-bit CAN identifier is higher (0..536870911).

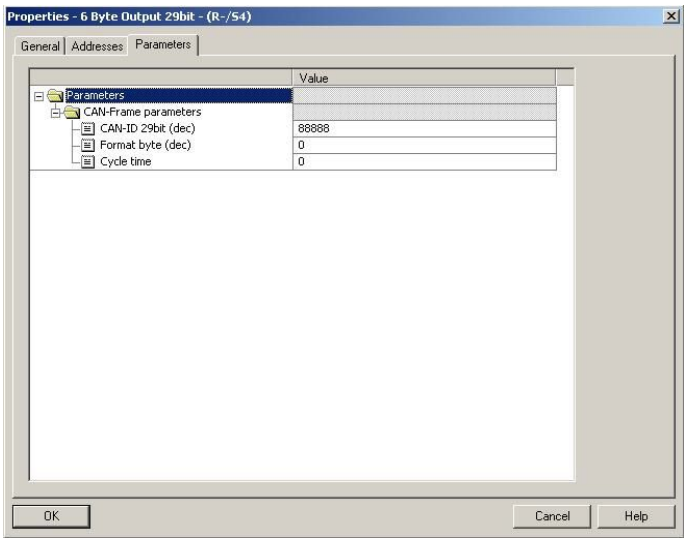


Fig. 21: Example: 6 byte output module with 29-bit identifier

5.3.2.3 Setting Data Format with the Format Byte

The *Format Byte* can be set in the *Properties* window of the module.
The control byte *Format Byte* is used to convert the user data from Motorola format (Big Endian, high byte first) into Intel format (Little Endian, low byte first).

Background: Messages which are longer than one byte are normally transmitted on a CAN network in Intel notation, while the Siemens PLC operates in Motorola format.

Starting with bit 7 of the format byte you can decide whether the following byte is to be converted as well, i.e. swapped, or not. If a ‘1’ is specified for a byte, the following bytes are converted until the next ‘0’ transmitted. The functionality can be explained best by means of an example.

Example: A CAN telegram has got a date in Intel format in the first two byte, followed by two bytes which are not to be swapped and a long word in the last four bytes which is in Intel format again.
Binary the following description for the format byte:

Bit No.	7	6	5	4	3	2	1	0
Bit of *Format Byte*	**1**	**0**	**0**	**0**	**1**	**1**	**1**	**0**
hexadecimal	8				E			
decimal	142							
action	begin swap	end swap	un-changed	un-changed	begin swap	swap	swap	end swap

Data bytes	1	2	3	4	5	6	7	8
CAN Frame	2 byte Intel format		byte 3	byte 4	4 byte Intel format			
PLC data	2 byte Motorola format		byte 3	byte 4	4 byte Motorola format			

From this the format byte results in 8Eh= 142d.
If all eight bytes are to be swapped, for instance, value FEh= 254d is specified for the format byte.
The lowest bit is generally without significance, because the telegram and therefore the formatting have been completed. The bit should always be set to '0'.

INFORMATION
The parameter *format byte* must be set to ‘00’, if byte swapping is not wanted.

5.4 Save Settings on Hard Disk

To save the settings to hard disk click menu item *Station/Save*.
Afterwards the settings are transferred to the PLC by clicking the menu items *PLC/Download*.

5.5 Assign Device Name to the IO Device

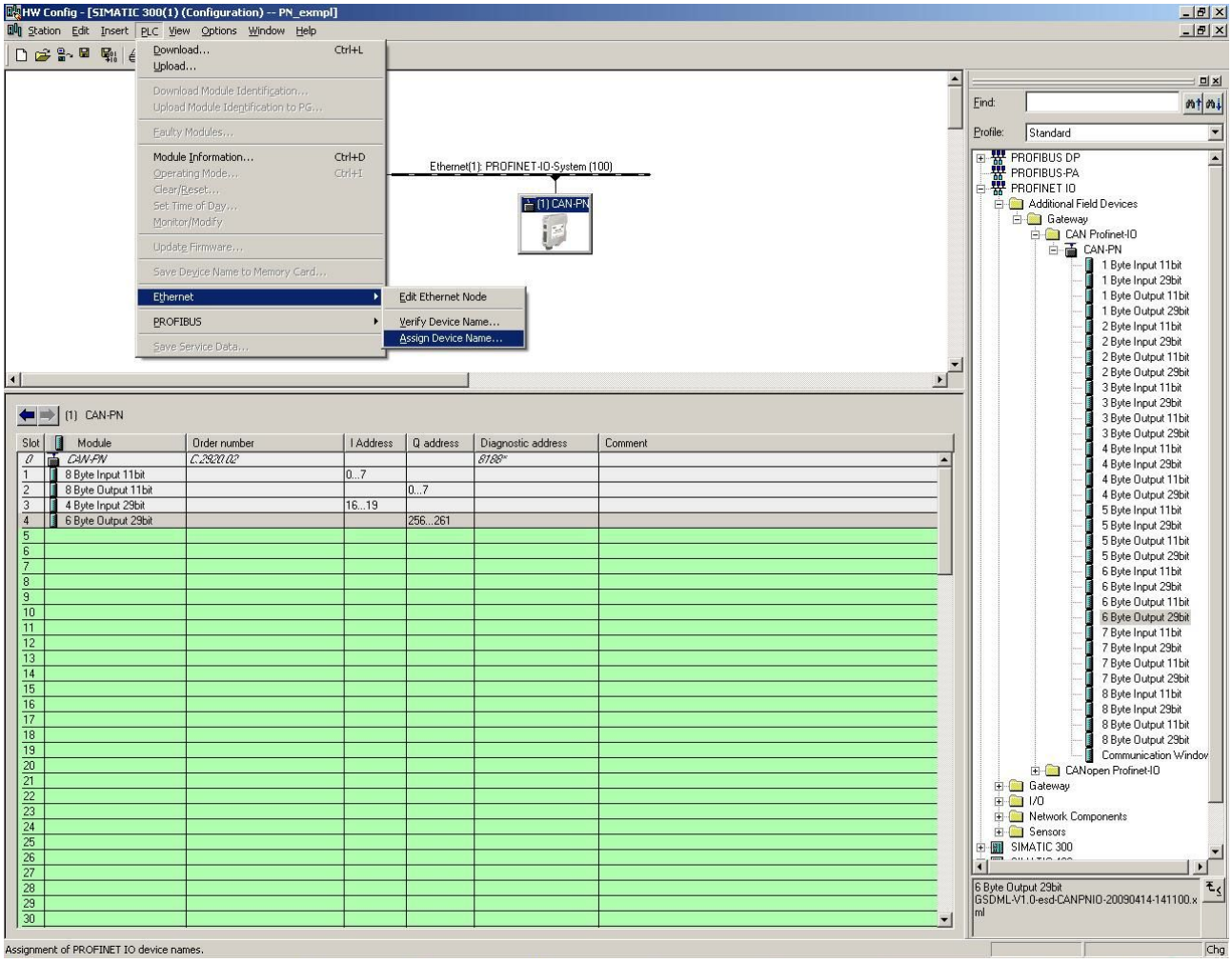


Fig. 22: Open window *Assign device name*

To assign a device name, select the menu item *PLC* in the main menu of the window. Click *Ethernet* and *Assign Device Name...* in the drop-down menu.

The following window opens:

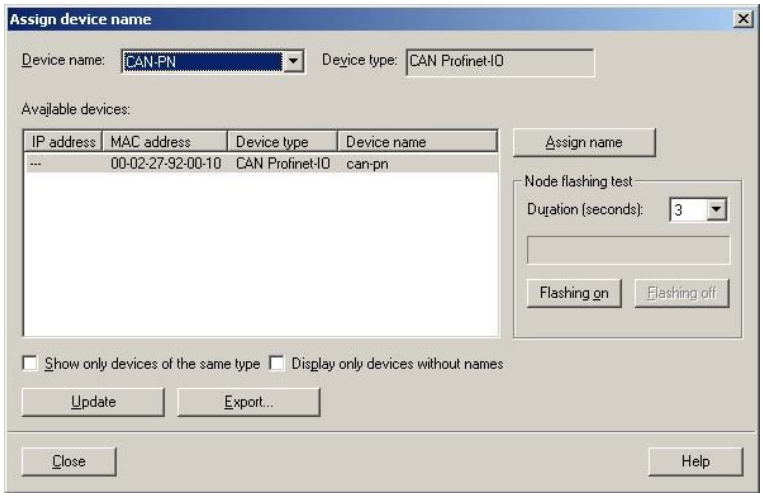


Fig. 23: Assign device name

Device name: A name for the PROFINET IO device can be selected (CAN-PN is entered here for the CAN-PN per default).

Available devices: The available IO devices are listed here.
To select an IO device click the line in the list of the available devices.

Assign name To assign a name to an IO device selected in the list, click the button *Assign name* . The name specified in the input field *Device name* is assigned to the selected IO device then.
Before you assign a name to an IO device ensure that the correct IO device is selected. Click the button *Flashing on* and control if the yellow LED of the selected IO device is flashing.

Flashing on Select an IO device in the list of available devices. Clicking the button *Flashing on* , makes the yellow LED of the selected IO device flash for the period specified in *Duration (seconds)*.
Thus the selected IO device can easily be located.

Check box: By activating the check boxes:
❑ *Show only devices of the same type* and
❑ *Display only devices without names*
the number of devices shown in the list can be limited. Only devices of the same type or devices without device name can be displayed.

Update Clicking the button *Update* updates the list of the available devices.

Export.. To save the list of available devices click the button *Export..* .

5.6 The Communication Window

5.6.1 Introduction

If the connected CAN modules are addressed as described in chapter '5.1 Course of Configuration', each CAN identifier needs its own PLC address. The Communication Window offers the advantage that individual PLC addresses for different Tx-identifiers and different Rx-identifiers can be used. This is possible, because the identifiers of the CAN modules are transmitted as parameters together with the data at each access.
The disadvantage of the Communication Window is the lower data flow, though. Therefore it is recommendable to use the Communication Window for non-time-critical accesses such as writing CANopen SDOs after starting up the device.
The Communication Window will be described in detail on the following pages.

5.6.2 Communication Window Output / Input

The Communication Window consists of the modules Communication Window Output and Communication Window Input.

NOTICE
First the Communication Window **Output** has to be entered, **then** the Communication Window **Input**. If one of the modules is missing or the order is not kept, the Communication Window will not work.

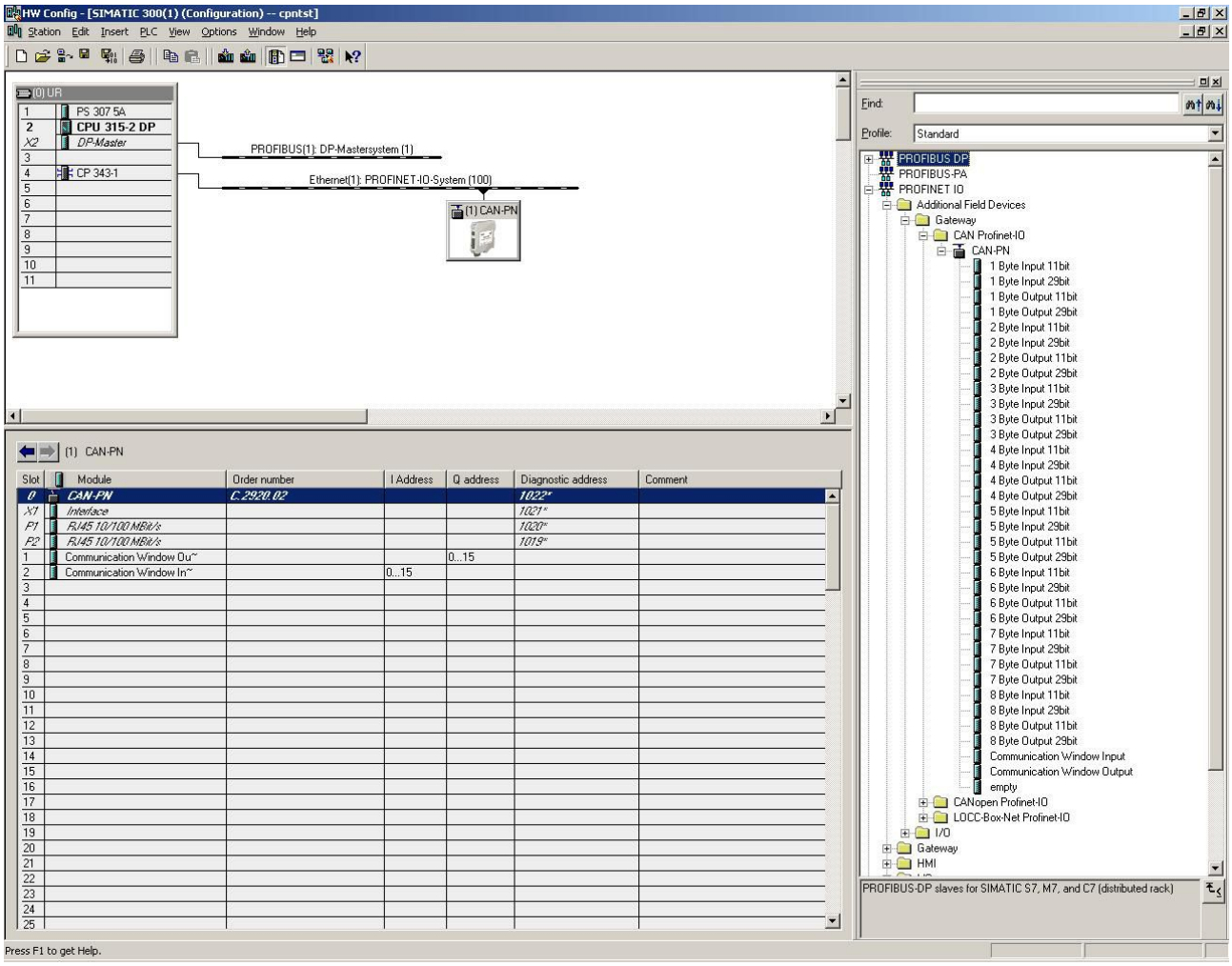


Fig. 24: Communication Window Output / Input

5.6.3 Parametrizing the Communication Window

A parametrization of the Communication Window is not necessary, because the Communication Window does not have parameters.

5.6.4 Format of the Communication Window

The 16 bytes of the Communication Window are assigned differently, according to data direction.

5.6.4.1 Write Bytes of the Communication Window

(Command call and transmitting data PLC → Gateway → CAN)

Byte of the Communication Window	Contents
0 1	High_Byte of the CAN identifier (identifier bit [15] 10...8) Low_Byte of the CAN identifier (identifier bit 7...0)
2 3	with 11-bit CAN identifier byte 2 and 3 are always ‘0’ with 29-bit CAN identifier byte 2: identifier bits 28...24 byte 3: identifier bits 23...16 **NOTICE** If 29 bit identifiers are used, always set bit 29 to ‘1‘!
4 5 6 7 8 9 10 11	data byte 0 data byte 1 data byte 2 data byte 3 data byte 4 data byte 5 data byte 6 data byte 7
12	data length for transmission jobs (Tx)
13	PLC loop counter value (has to be incremented in pulse with OB1 in order to synchronize the gateway with the OB1 cycle)
14	sub command (always set to ‘0’)
15	command (description refer to page 35)

Table 3: Write bytes of the Communication Window

5.6.4.2 Read Bytes of the Communication Window

(command acknowledge and reception of data CAN → Gateway → PLC)

Byte of the Communication Windows	Contents
0 1	as long as no receive data are available ‘EEEE’$_h$, otherwise High_Byte of CAN identifier (identifier bits [15] 10...8) Low_Byte of CAN identifier (identifier bits 7...0)
2 3	with 11-bit CAN identifier byte 2 and 3 are always ‘0’ with 29-bit CAN identifier byte 2: identifier bits 28...24 byte 3: identifier bits 23...16 **NOTICE** If 29 bit identifiers are used, always set bit 29 to ‘1‘!
4 5 6 7 8 9 10 11	data byte 0 data byte 1 data byte 2 data byte 3 data byte 4 data byte 5 data byte 6 data byte 7
12	number of received data bytes
13	return value of the PLC loop counter which has been transmitted to the gateway via the last PROFINET telegram
14	return of the sub command
15	error code of the read function (not supported at the moment)

Table 4: Read bytes of the Communication Window reading CAN data (Rx)

The following table shows commands which are currently supported. The sub command is not yet evaluated and should always be set to ‘0’, therefore.

Command	Function
1	transmit data
3	receive data at enabled Rx-identifiers
4	enable Rx-identifier for data reception
5	deactivate reception (command 4)
6	transmits an RTR frame
7	executes command 4 and command 6
(11)	(reserved)

Table 5: Commands of the Communication Window

NOTICE
A command is only completely processed, if, when reading the Communication Window, byte 13 of the CAN-PN module provides the value of the PLC-loop counter which was transferred during the command call.
Before the following command is called, it is therefore advisable to check byte 13 first!

Explanations of the commands:

Command 1: Send data

In order to send data via the Communication Window the CAN identifier has to be specified in bytes 0 and 1 (or 0...3 for 29-bit identifiers). In addition to the number of bytes to be transmitted, a PLC-loop counter has to be specified. The loop counter has to be realised by the user. It is required to provide the CAN-PN-gateway with the OB1-cycle of the PLC.

Command 3: Reception on enabled Rx-identifiers

The reception of data requires the CAN-Rx-identifiers, which are to receive data, to be enabled (see command 4).
After reception command 3 has been written, read accesses to the Communication Window will give you the data structure shown on page 34. The Rx-data is received asynchronously to the PLC-cycle. Until valid data has been received you will be returned the value ‘EEEE’$_h$ in the first bytes in read accesses. Only after valid data has been received the Rx-identifier of the read frame in the first bytes becomes readable. In addition, the read command which requested the reception of data is assigned by means of the returned PLC-loop counter in byte 13.

The module has got a FIFO-memory for 255 CAN-frames to buffer the received Rx-data. If several Rx-frames are to be received on one Rx-identifier, or if frames of various Rx-identifiers enabled for reception are received, the data is not lost, as long as the PLC reads out the FIFO memory quicker than it is being filled.

Command 4: Enabling Rx-identifiers for reception

By means of this command the Rx-identifier whose data is to be received has to be enabled. More than one Rx-identifier can be enabled at the same time. For this, the command has to be called an according number of times.

Command 5: Deactivate reception (command 4)

After this command has been called no data is received any longer on the specified Rx-identifiers.

Command 6: Sending an RTR-frame

By means of this command a remote-request frame is transmitted. Prior to the transmission the reception on the Rx-identifier has to be enabled by command 4.

Command 7: Executes command 4 and command 6

See there.

5.6.5 Examples of the Communication-Window

5.6.5.1 Transmitting Data

1. Basic Setting of the Communication Window

The basic setting has to be made only once when setting up the Communication Window.

1.1 Program PLC-loop counter

8-bit loop counter for handshake function between PLC and gateway

	PLC-Cycle (Pseudo Code):
...	...
1.	**Read** Byte 13 (returned loop counter) of 'Read Bytes of Communication-Windows' (refer to page 34)
2.	Compare Byte 13 of the '**Read** Bytes of Communication-Windows' with PLC-loop counter byte 13 of the '**Write** Bytes of Communication-Windows'(refer to page 33), if unequal go to 6., if equal go to 3.
3.	Increase PLC-loop counter (Byte 13) of '**Write** Bytes of Communication-Window' (refer to page 33)
4.	Evaluation of '**Read** Bytes of Communication-Windows' (refer to page 34), i.e. the evaluation of the answer to the last command or received CAN frame (depending on the application).
5.	Send new '**Write** Bytes of Communication-Window' (refer to page 33) with increased loop-counter value of 3. and if necessary new application data.
6.	Continue PLC program (new request at the next program cycle)

2. Start Transmission Command by Writing the 16 Bytes of the Communication Window

Byte of Communication Window	Contents	Example here
0 1	High_Byte of CAN identifier (identifier bit [15] 10...8) Low_Byte of CAN identifier (identifier bit 7...0)	0x00 0x12
2 3	bytes 2 and 3 always '0' for 11-bit identifier	0x00 0x00
4 5 6 7 8 9 10 11	data byte 0 data byte 1 data byte 2 data byte 3 data byte 4 data byte 5 data byte 6 data byte 7	0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07
12	data length for transmission commands (Tx)	0x08
13	PLC-loop counter 8-bit counter	8-Bit Counter
14	sub-command (always set to '0')	0x00
15	command 'transmit data''	0x01

The data bytes 00, 01, 02, 03, 04, 05, 07 are transmitted on Tx -identifier 0x0012.
In order to acknowledge the execution of the command a read access to byte 13 of the Communication Window should follow. It has to have the same value of the PLC-loop counter as when the command was called.

5.6.5.2 Receiving Data

1. Basic Setting of the Communication Window

The basic setting of the Communication Window has already been described in the example 'Transmitting Data' above.

2. Receiving Data

2.1 Enabling the Rx-Identifier for Reception

In this example the data of the Rx-identifier 0x0123 are to be received.

Byte of Communication Window	Contents	Example here
0 1	High_Byte of CAN identifier (identifier bit [15] 10...8) Low_Byte of CAN identifier (identifier bit 7...0)	0x01 0x23
2 3	bytes 2 and 3 always '0' for 11-bit identifier	0x00 0x00
4 5 6 7 8 9 10 11	data byte 0 data byte 1 data byte 2 data byte 3 data byte 4 data byte 5 data byte 6 data byte 7	0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
12	data length for transmission command (Tx)	0x00
13	PLC-loop counter 8-bit counter	8-Bit Counter
14	sub-command (always set to '0')	0x00
15	command 'Enable Rx-Identifier'	0x04

In order to acknowledge the execution of the command a read access of byte 13 of the Communication Window should be made with every command call. It has to have the same value of the PLC-loop counter as it had when the command was called.

2.2 Initiate Reception of Data of the Enabled Rx-Identifier

Byte of the Communication Window	Contents	Example here
0 1	High_Byte of CAN identifier (identifier bit [15] 10...8) Low_Byte of CAN identifier (identifier bit 7...0)	0x01 0x23
2 3	bytes 2 and 3 always '0' for 11-bit identifier	0x00 0x00
4 5 6 7 8 9 10 11	data byte 0 data byte 1 data byte 2 data byte 3 data byte 4 data byte 5 data byte 6 data byte 7	0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
12	data length for transmission command (Tx)	0x00
13	PLC-loop counter 8-bit counter	8-Bit Counter + n
14	sub-command (always set to '0')	0x00
15	command 'Read Rx-Identifier'	0x03

2.3 Reading the Data

After an undetermined time the Rx-data is received and can be accessed by reading the Communication Window. Since the data is received asynchronously to the PLC-cycles the Communication Window has to be read again and again until the data was received (polling). By comparing the values of the PLC-loop counter you can determine, whether the data received is the correct data from the read command.
A read access returns the following bytes:

Byte of the Communication Window	Contents	Example here
0 1	High_Byte of CAN identifier (identifier bit [15] 10...8) Low_Byte of CAN identifier (identifier bit 7...0)	0x01 0x23
2 3	bytes 2 and 3 always ‘0’ for 11-bit identifier	0x00 0x00
4 5 6 7 8 9 10 11	received data byte 0 received data byte 1 received data byte 2 received data byte 3 received data byte 4 received data byte 5 received data byte 6 received data byte 7	0xAA 0xBB 0xCC 0xDD 0xEE 0xFF 0x00 0x11
12	Data length	0x08
13	PLC-loop counter	8-Bit Counter + n
14	returned sub-command (without significance)	0x00
15	error code of the read function (without significance)	0x00

2.4 Deactivate Reception of Data on this Rx-Identifier

If no further data is to be received on this identifier, the reception is to be disabled again.

Byte of the Communication Windows	Contents	Example here
0 1	High_Byte of CAN identifier (identifier bit [15] 10...8) Low_Byte of CAN identifier (identifier bit 7...0)	0x01 0x23
2 3	bytes 2 and 3 always ‘0’ for 11-bit identifier	0x00 0x00
4 5 6 7 8 9 10 11	data byte 0 data byte 1 data byte 2 data byte 3 data byte 4 data byte 5 data byte 6 data byte 7	0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
12	data length for transmission command (Tx)	0x00
13	PLC-loop counter 8-bit counter	8-Bit Counter + m
14	sub-command (always set to ‘0’)	0x00
15	command ‘Disable Rx-Identifier’	0x05

6. Configuration with the TIA Portal

6.1 Using GSDML File and CAN-PN Gateway with the TIA Portal

INFORMATION
This chapter does only describe the steps which are relevant for the usage of the GSDML file and the CAN-PN gateway with the TIA Portal.
Please read the documentation of your TIA Portal for a detailed description.

6.1.1 Quick Start

Step	Action	see page
1.	Disconnect the online connection in your TIA-Portal, because the hardware and software have to be compiled in offline mode!	-
2.	Change into the project view of your TIA-Portal.	-
3.	Install the GSDML file as described in chapter „Installation of the GSDML File“.	41
4.	Insert the CAN-PN in your project, see chapter „Insert CAN-PN Hardware and Network Configuration“.	43
5.	Compile and load the hardware and software as described in chapter „Compile and Download Hardware and Software“.	50
6.	Go online again. See Fig. 44.	51

6.1.2 Installation of the GSDML File

Change in the program window of your TIA Portal into project view to import the device description file. Click onto menu item *Options* in the main menu.

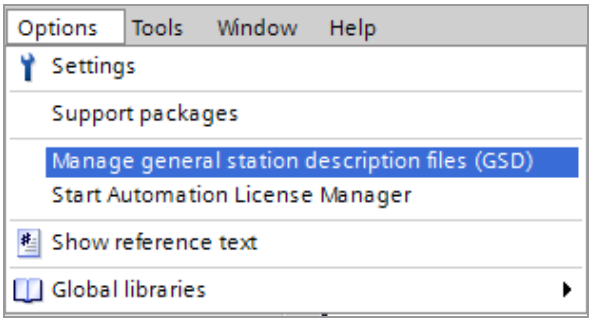
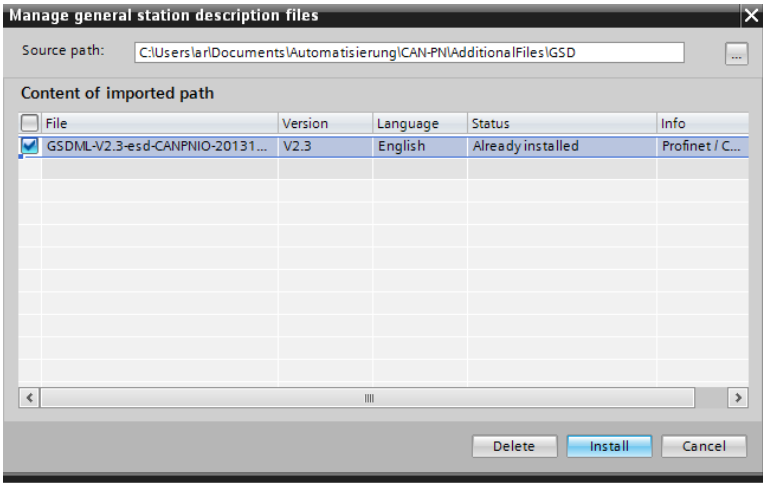


Figure 25: *Extras/Manage GSD*

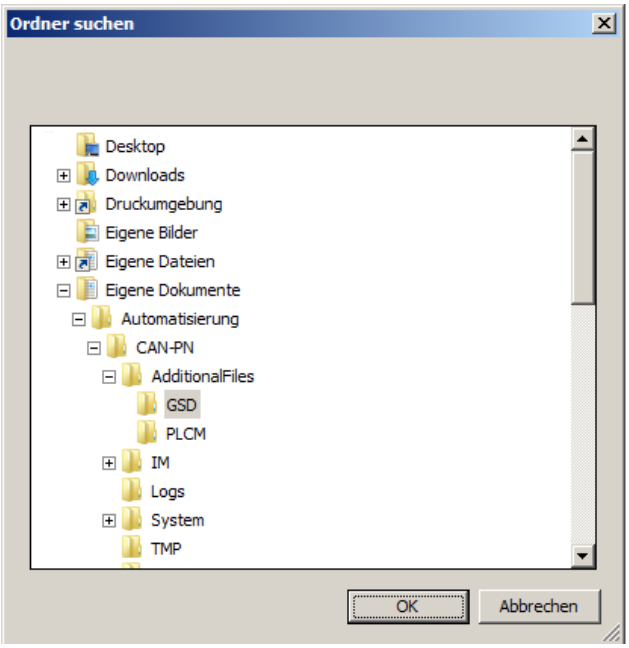
Choose *Manage general station description files (GSD)* in the pull-down menu, see Figure 25.

Now the window *Manage general station description files* opens.



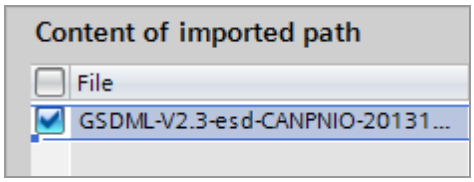
If the path to the folder is not already entered in the input field *Source path* correctly, click onto the button [...] in the upper right part of the window *Manage general station description files* to choose the path.

Figure 26: Window *Manage general station description files*



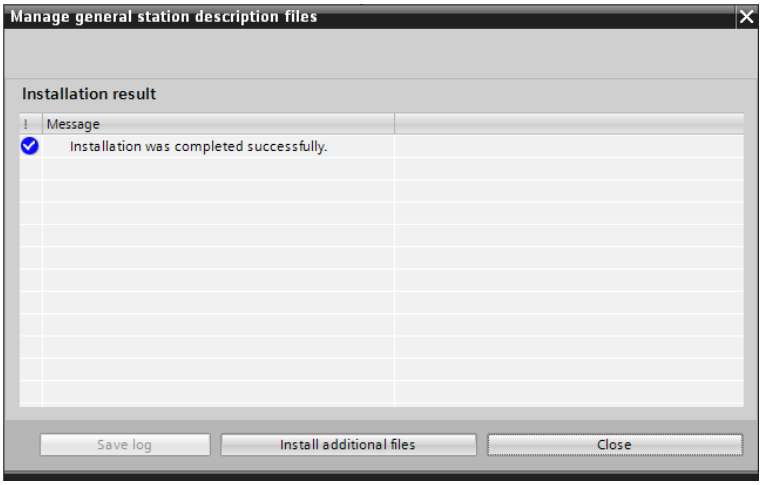
In this window you can now select the path to your GSDML folder.
Click on the button *OK* to confirm the path.

Figure 27: Select the path to the folder



All available GSDML files are listed in the *Manage general station description files window* now.
Activate the corresponding check-box on the left, to select a file, see Figure 26.
Click on the button *Install* to confirm the selection.

Figure 28: Activate check-box



If the installation of the selected file has been successful, you see the message as shown in the window on the left.

Figure 29: Successful installation

6.1.3 Insert CAN-PN Hardware and Network Configuration

Now you are able to assemble the devices and nets for your project in the TIA Portal.
Therefore click under *Project tree / Devices* in the upper left part of the window onto *Devices & networks* as shown in the following figure.

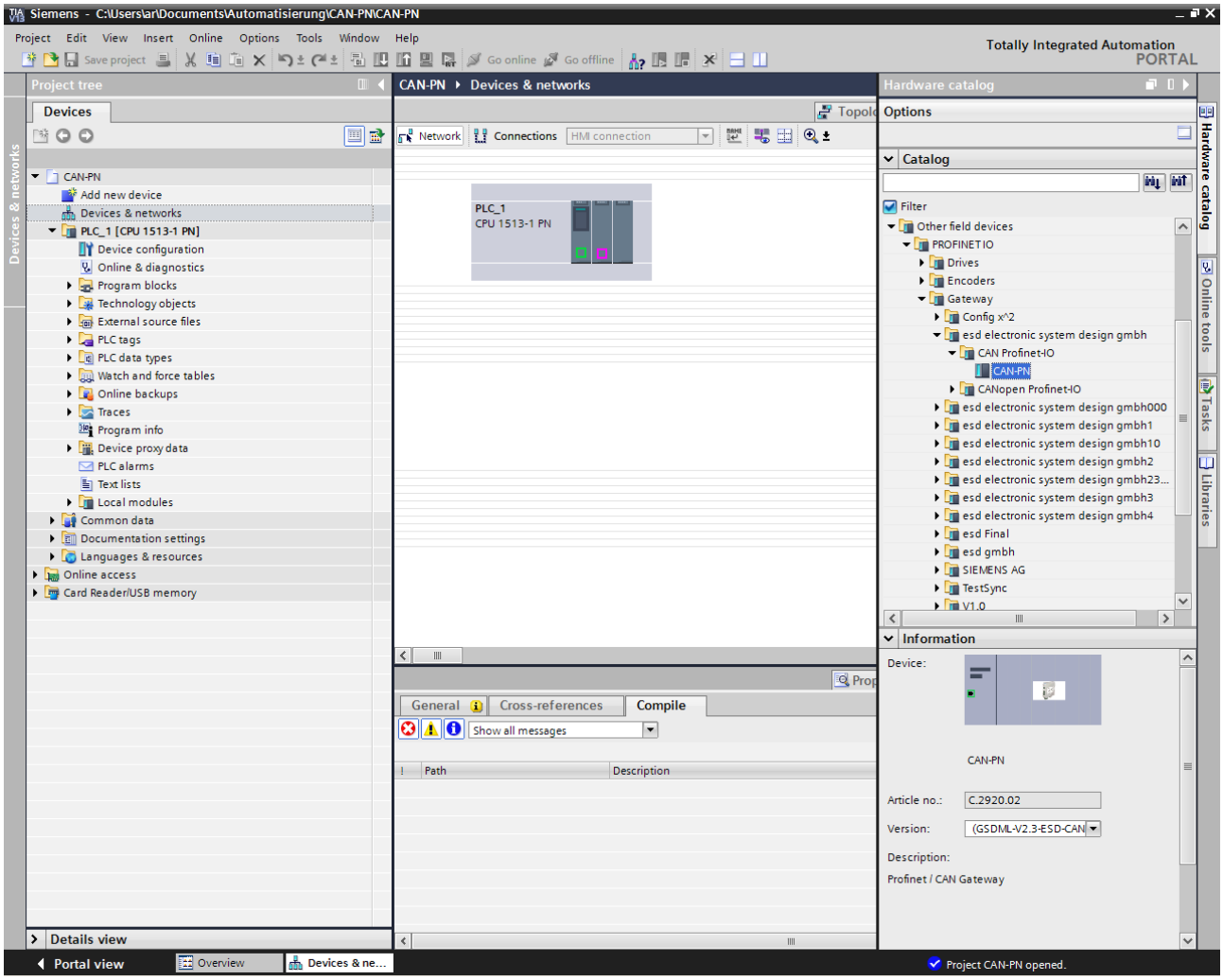


Figure 30: Devices and networks

Now choose your CAN-PN device in the tree structure of the *Hardware Catalogue,* which is in the right part of the window.

> **IMPORTANT**
> Please note, that it is absolutely necessary that you select the appropriate CAN network configuration for your CAN-PN device!
> Therefore please proceed as described in the following.

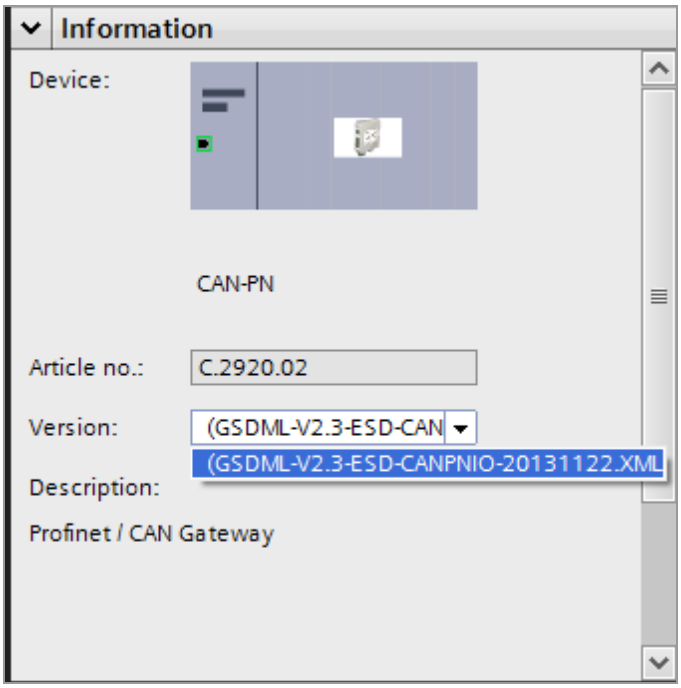


Figure 31: Choose the version (detail)

In the field *Information,* which is in the lower right part of the TIA Portal program window, the CAN-PN is listed with *Article no.* and *Version*.

Select the new CAN network configuration in the pull-down menu of *Version*.

In the example (see Figure 31) the following file is displayed:
`GSDML_V2.3_CANPNIO-2013122.XML`

To import the device in your project, select the CAN-PN with the left mouse button in the tree-structure in the field *Catalog,* drag it into the field *Network view* and drop it.

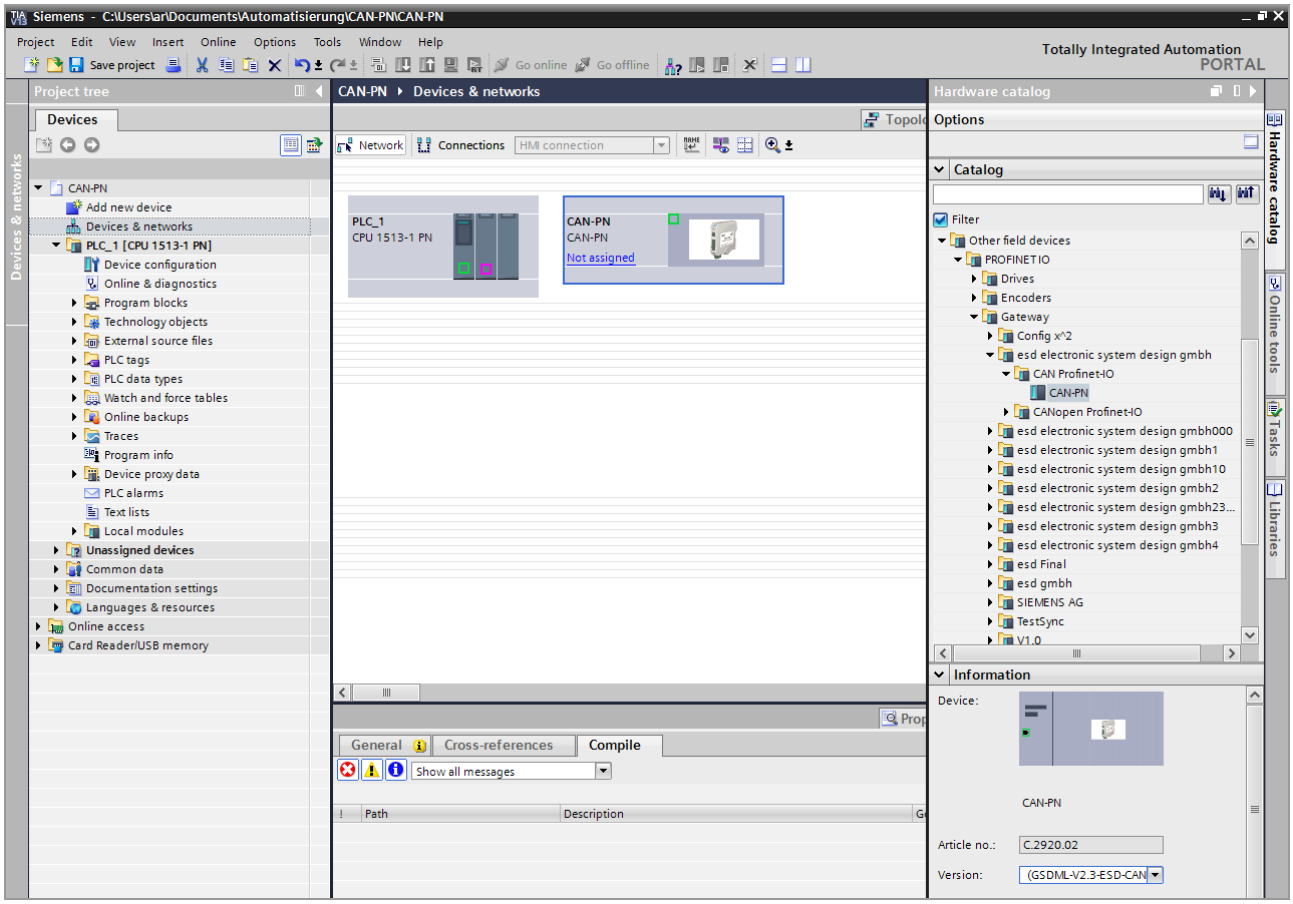


Figure 32: Program window with new CAN-PN in the window *Network view*

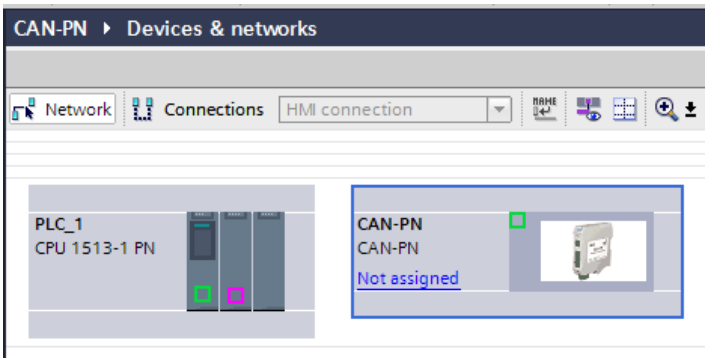


Figure 33: Net view (detail)

The CAN-PN is now displayed in the field *Network view* as CAN-PN, but it is still not connected to PROFINET (see Figure 33).

To assign the device, click onto the button *Not assigned.*

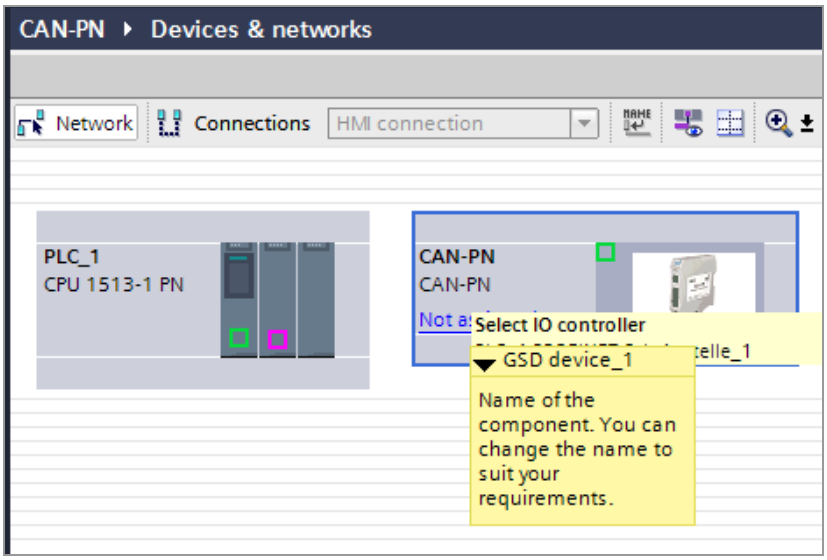


Figure 34: Select IO controller

The drop-down menu *Select IO controller* is opened. In this example the *GSD device_1* can be chosen.

Now choose the network, to which the CAN-PN shall be connected, by a mouse-click.

The CAN-PN is connected to the network now.

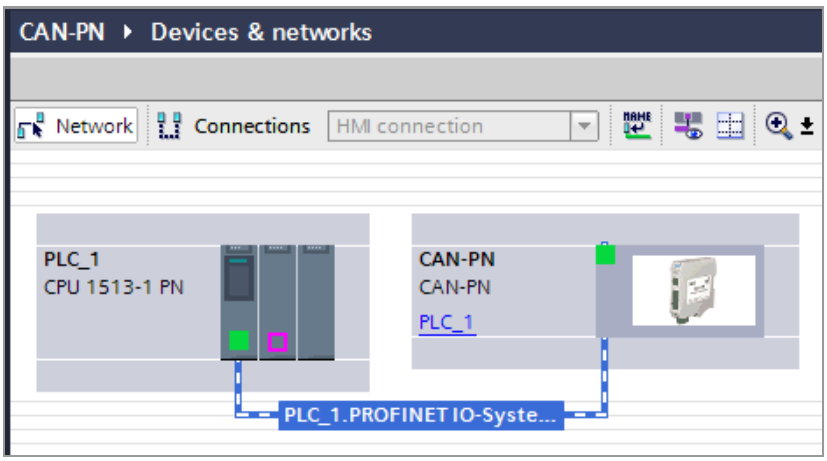


Figure 35: Connection CAN-PN and network

Double click on the CAN-PN in the *network view* to transfer the CAN-PN in the *Device overview* (see Figure 36)

Now configure the baud rate of the CAN-PN. Therefore click on the CAN-PN in the field *Device overview*.

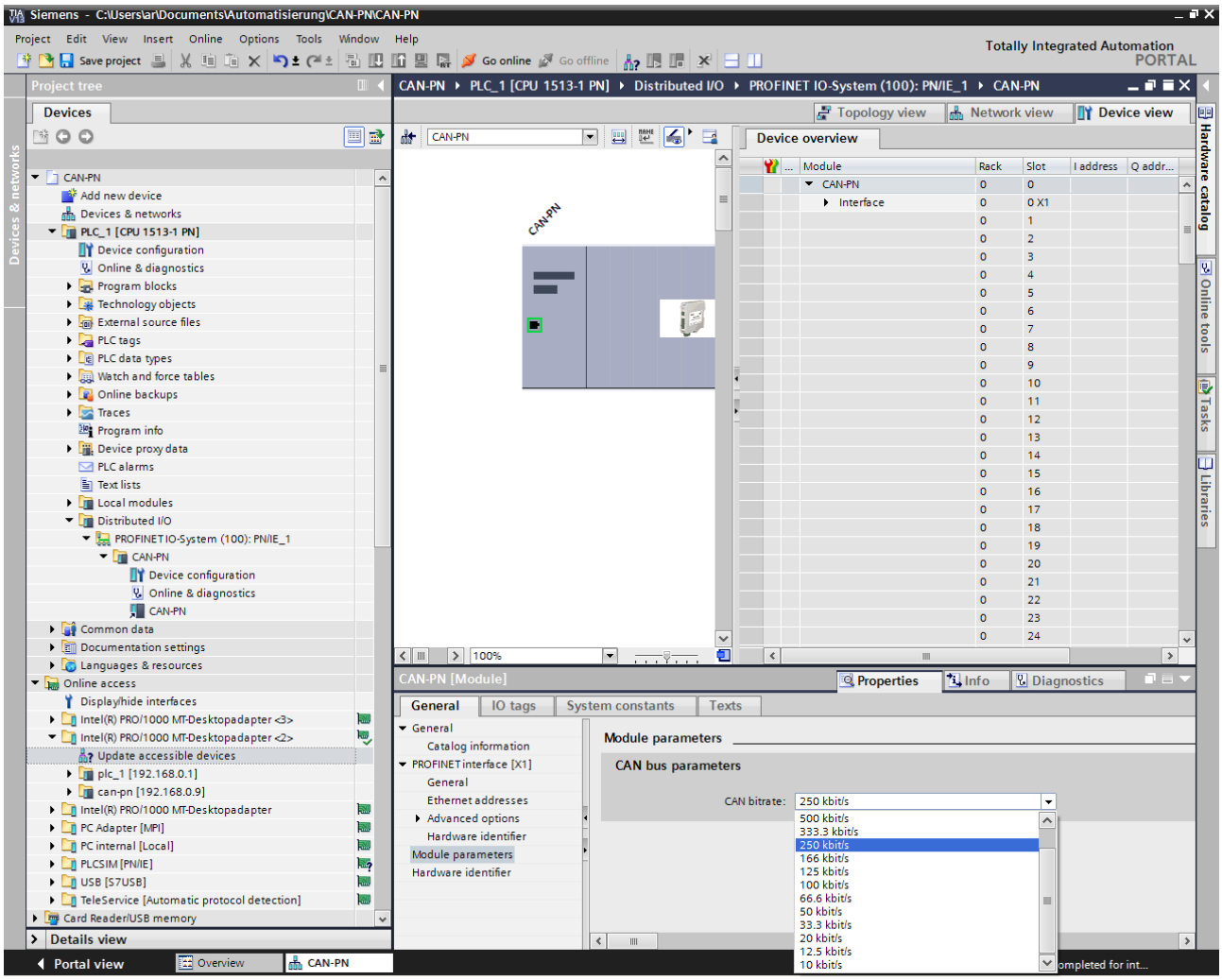


Figure 36: Configure the CAN-PN baud rate

The baud rate of the CAN-PN can be selected in the lower part of the program window in the *Properties* field. Select the menu item *Module parameter* in the tab *General* and select the CAN bit rate.

To insert a module, open the view *Hardware catalog* on the right side of the program window of the TIA Portal.

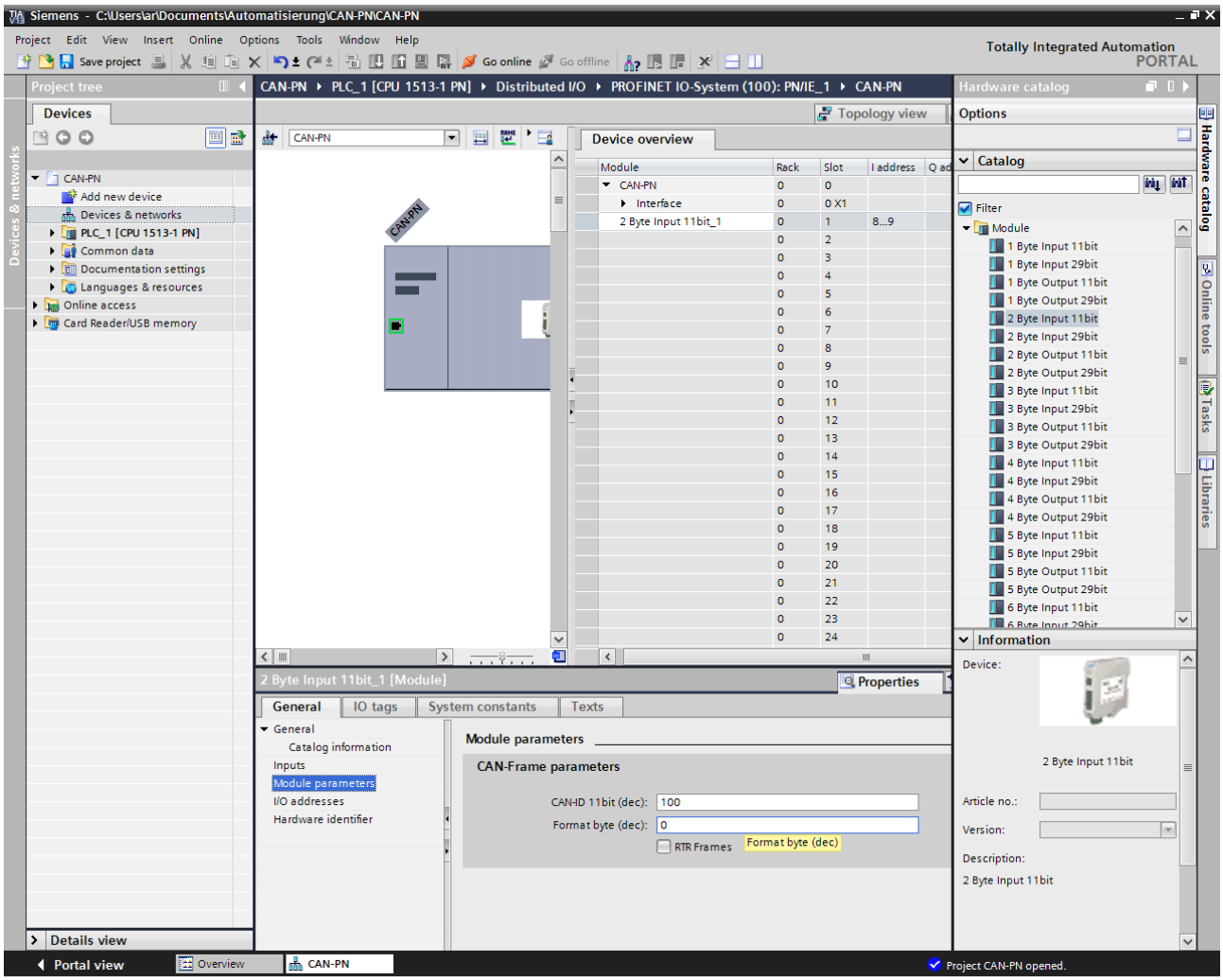


Figure 37: Select module

Choose an object (in this example *2 Byte Input 11bit*) in the *Hardware catalog* under *Module* and transfer it per drag'n'drop into the *Device overview*.
You can now select other input or output modules and transfer them into the *Device overview*.

To parametrize the input module or an output module choose the tab *Properties* in the lower part of the program window. Then select menu item *Module parameters* in the tab *General* and enter your values.

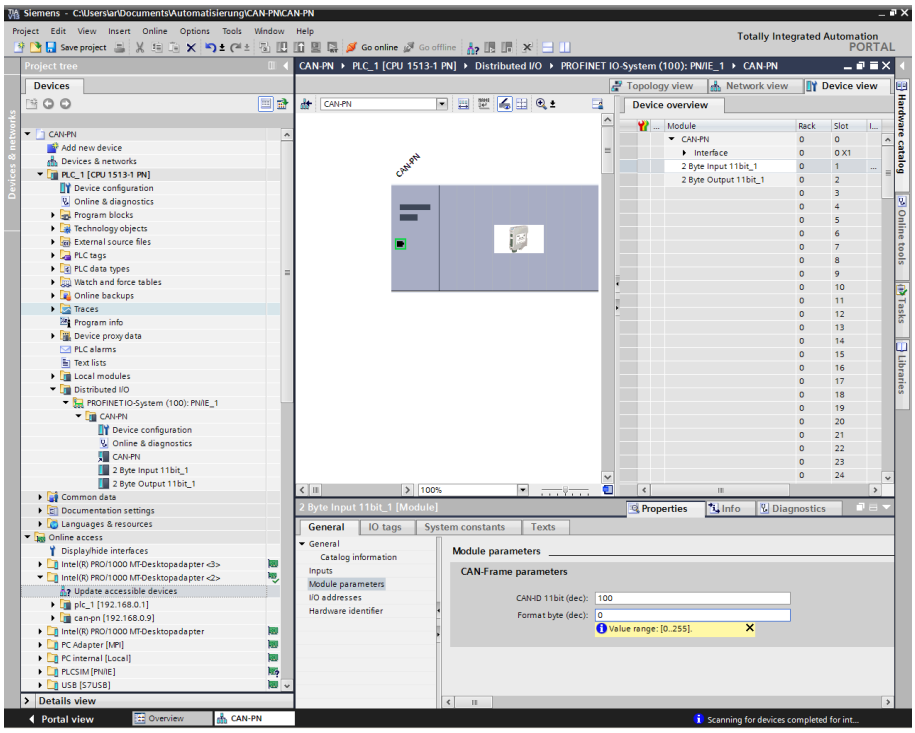


Figure 38: Parametrize input module

For the input module the *CAN-Frame parameters*, as *CAN-ID* and *Format byte* (see description on page 29) can be specified under *Module parameters* (Figure 38).

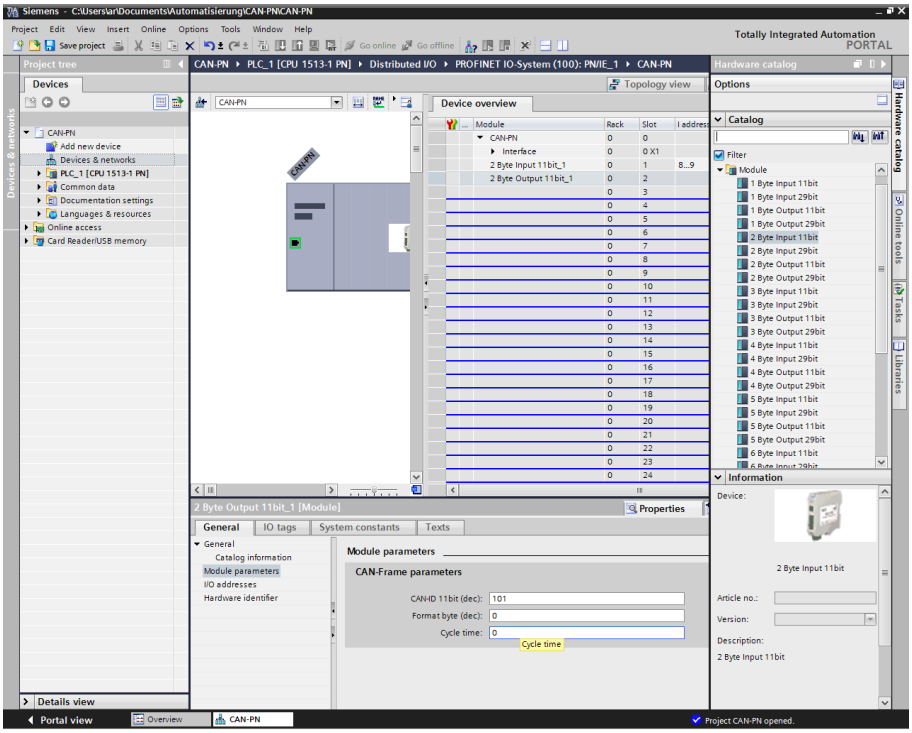


Figure 39: Parametrize output module

For the output module the *CAN-Frame parameters*, as *CAN-ID, Format byte* and *Cycle Time* can be specified under *Module parameters* (Figure 39).

To insert the Communication Window select in the *Hardware catalog* under *Catalog* a Communication Window and transfer it to the *Device overview* (see Figure 40).
The usage of the Communication Windows is described from page 32 on.

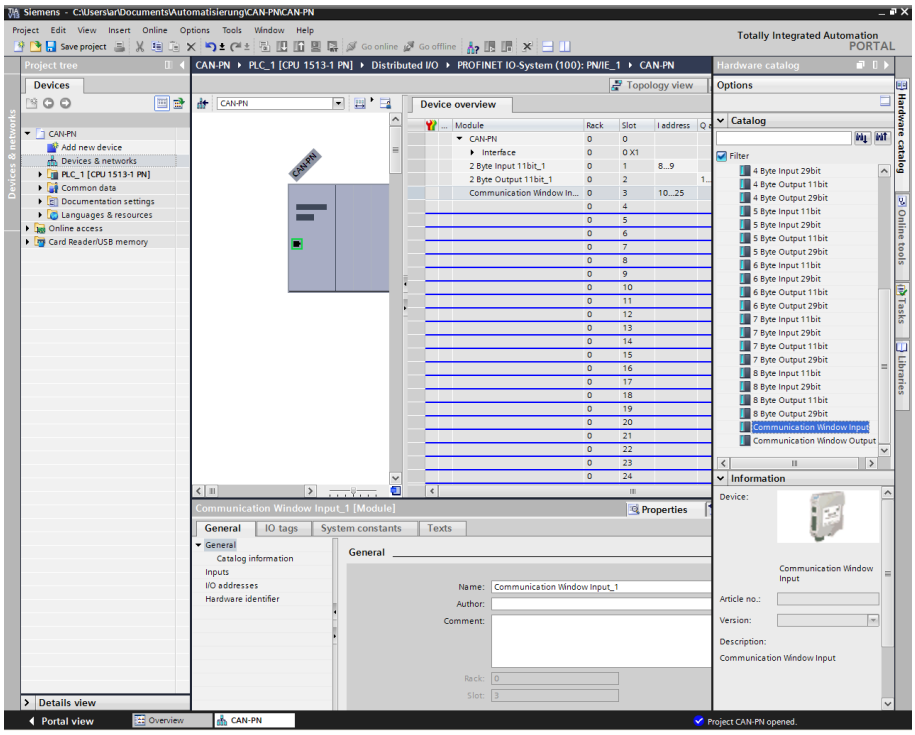


Figure 40: Insert Communication Window

Assign the start addresses to the chosen modules. Therefore click on a module in the *Device overview.* In the *Properties* tab under *General* the menu item *IO addresses* can be selected and the *Start address* can be entered.

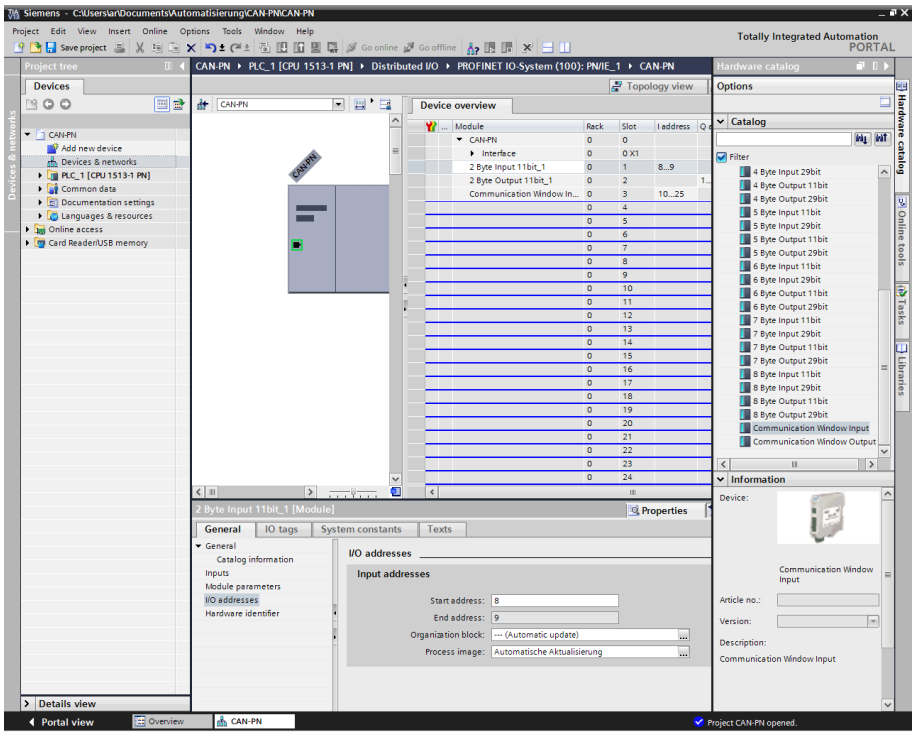


Figure 41: Assign IO addresses

6.1.4 Compile and Download Hardware and Software

Before you can download the software, it must be compiled.
To compile the software select your device (*PLC_1 here)* in the field *Project tree / Devices* and further in the pull-down menu *Compile* and then *Hardware and Software (only Changes)*.
You must be still in offline mode for this.

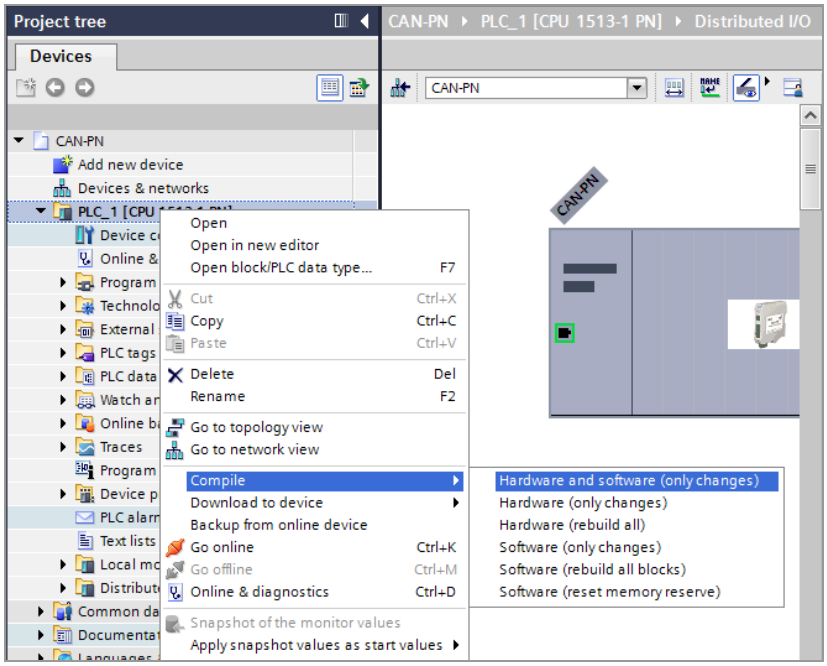


Figure 42: Compile Hardware and Software (detail)

The configuration is compiled. Continue to work offline!
Now the hardware and software can be downloaded in the device as described in the following:
Select your device (*PLC_1 here)* again and further in the pull-down menus the menu items *Download to device* and then *Hardware and software*.

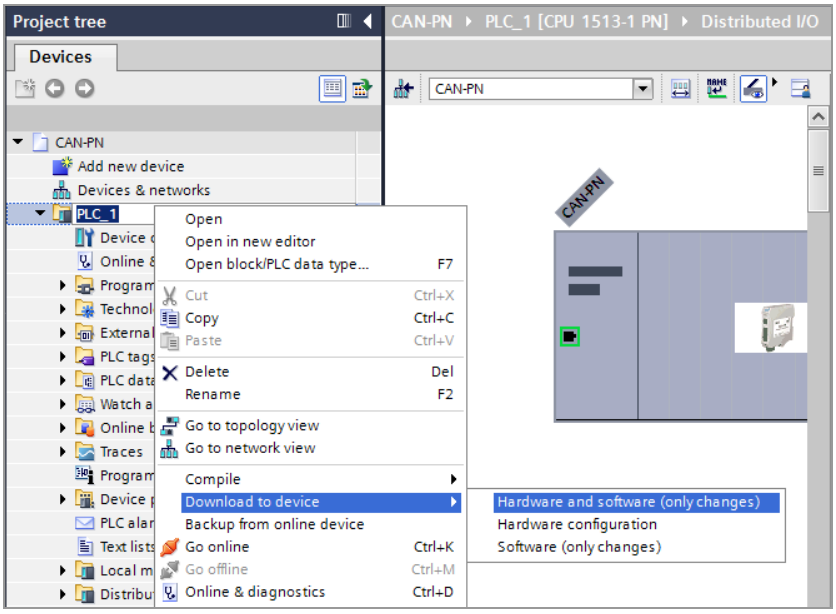


Figure 43: Download hardware and software to device (detail)

Hardware and software are downloaded now.

Click on the button *Go online* in the tool bar under the main menu to go online.

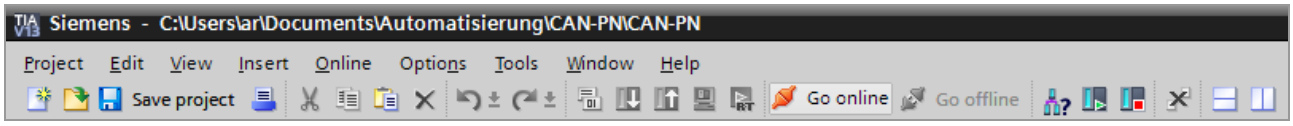


Figure 44: Toolbar with button *Go online*

The online connection is now established.
In the program window the components of the decentralised periphery are marked with check marks.
The I/O addresses of the single units are shown in detail in the *Device overview*.

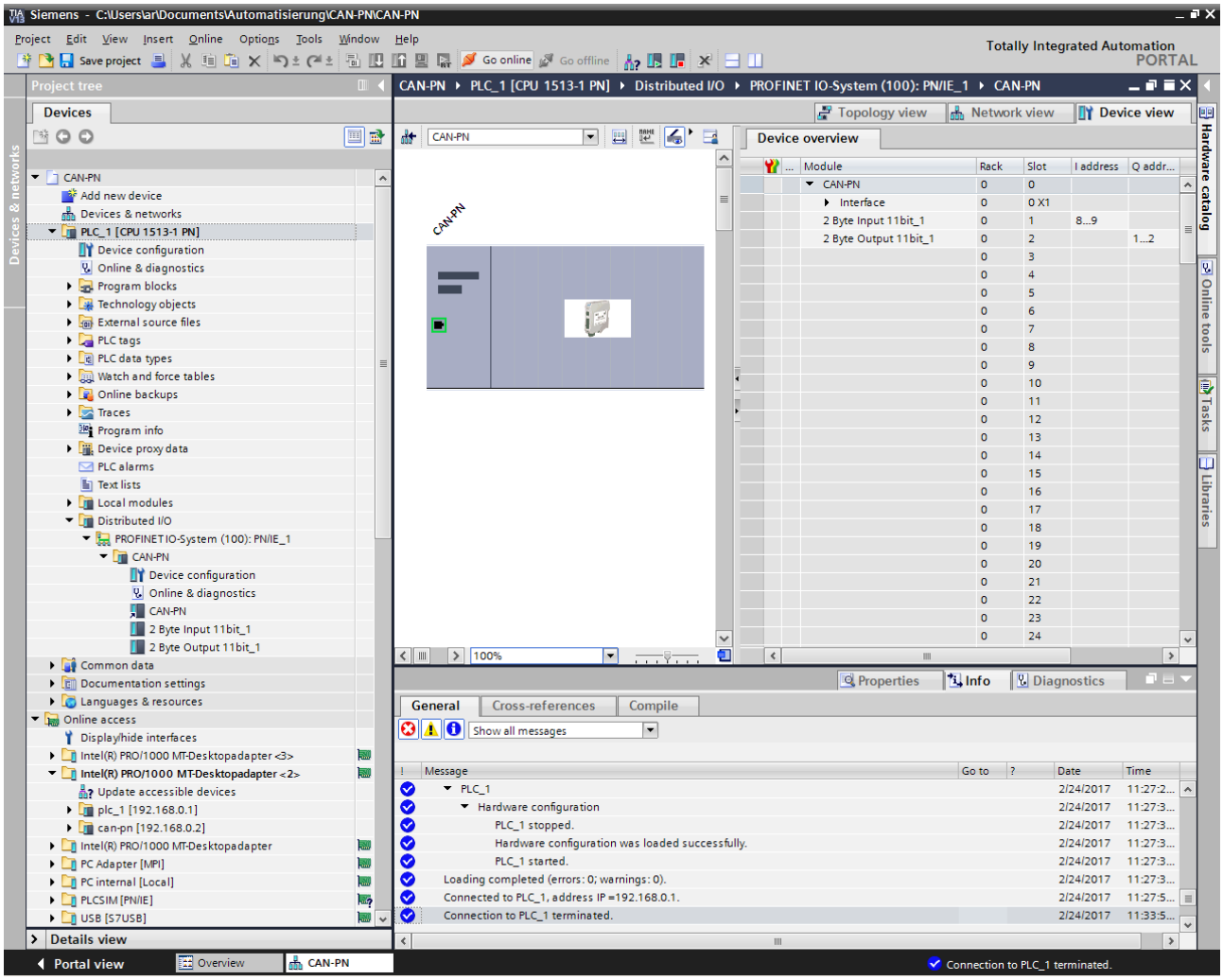


Figure 45: Main window with device overview