OPTE9

Dual port ethernet option board installation manual

EATON
Powering Business Worldwide

Table of contents

Table of contents, continued

1. Safety

This manual contains clearly marked cautions and warnings that are intended for your personal safety and to avoid any unintentional damage to the product or connected appliances.

Read the information included in cautions and warnings carefully.

The cautions and warnings are marked as follows:

Table 1. Warning signs

⚠	**= DANGER! Dangerous voltage**
⚠	**= WARNING or CAUTION**
⚠	**= Caution! Hot surface**

1.1 Danger

The components of the power unit are live when the drive is connected to mains potential. Coming into contact with this voltage is extremely dangerous and may cause death or severe injury.

The motor terminals U, V, W and the brake resistor terminals are live when the AC drive is connected to mains, even if the motor is not running.

After disconnecting the AC drive from the mains, wait until the indicators on the keypad go out (if no keypad is attached, see the indicators on the cover). Wait 5 more minutes before doing any work on the connections of the drive. Do not open the cover before this time has expired. After expiration of this time, use a measuring equipment to absolutely ensure that no voltage is present. Always ensure absence of voltage before starting any electrical work!

The control I/O-terminals are isolated from the mains potential. However, the relay outputs and other I/O-terminals may have a dangerous control voltage present even when the AC drive is disconnected from mains.

Before connecting the AC drive to mains make sure that the front and cable covers of the drive are closed.

During a ramp stop (see the Application Manual), the motor is still generating voltage to the drive. Therefore, do not touch the components of the AC drive before the motor has completely stopped. Wait until the indicators on the keypad go out (if no keypad is attached, see the indicators on the cover). Wait additional 5 minutes before starting any work on the drive.

1.2 Warnings

The AC drive is meant for fixed installations only.

Do not perform any measurements when the AC drive is connected to the mains.

The earth leakage current of the AC drives exceeds 3.5mA AC. According to standard EN61800-5-1, a reinforced protective ground connection must be ensured. See Chapter 1.3.

If the AC drive is used as a part of a machine, the machine manufacturer is responsible for providing the machine with a supply disconnecting device (EN 60204-1).

Only spare parts delivered by Eaton can be used.

At power-up, power brake or fault reset the motor will start immediately if the start signal is active, unless the pulse control for Start/Stop logic has been selected. Furthermore, the I/O functionalities (including start inputs) may change if parameters, applications or software are changed. Disconnect, therefore, the motor if an unexpected start can cause danger.

The motor starts automatically after automatic fault reset if the auto restart function is activated. See the Application Manual for more detailed information.

Prior to measurements on the motor or the motor cable, disconnect the motor cable from the AC drive.

Do not touch the components on the circuit boards. Static voltage discharge may damage the components.

Check that the EMC level of the AC drive corresponds to the requirements of your supply network.

1.3 Earthing and earth fault protection

⚠ CAUTION!

The AC drive must always be earthed with an earthing conductor connected to the earthing terminal marked with ⏚.

The earth leakage current of the drive exceeds 3.5mA AC. According to EN61800-5-1, one or more of the following conditions for the associated protective circuit must be satisfied:

a. The protective conductor must have a cross-sectional area of at least 10 mm2 Cu or 16 mm2 Al, through its total run.

b. Where the protective conductor has a cross-sectional area of less than 10 mm2 Cu or 16 mm2 Al, a second protective conductor of at least the same cross-sectional area must be provided up to a point where the protective conductor has a cross-sectional area not less than 10 mm2 Cu or 16 mm2 Al.

c. Automatic disconnection of the supply in case of loss of continuity of the protective conductor.

The cross-sectional area of every protective earthing conductor which does not form part of the supply cable or cable enclosure must, in any case, be not less than:

– 2.5mm^2 if mechanical protection is provided or

– 4mm^2 if mechanical protection is not provided

The earth fault protection inside the AC drive protects only the drive itself against earth faults in the motor or the motor cable. It is not intended for personal safety.

Due to the high capacitive currents present in the AC drive, fault current protective switches may not function properly.

WARNING

Do not perform any voltage withstand tests on any part of the AC drive. There is a certain procedure according to which the tests must be performed. Ignoring this procedure can cause damage to the product.

Note: You can download the English and French product manuals with applicable safety, warning and caution information from www.eaton.com/drives.

REMARQUE Vous pouvez télécharger les versions anglaise et française des manuels produit contenant l'ensemble des informations de sécurité, avertissements et mises en garde applicables sur le site www.eaton.com/drives.

2. OPTE9 dual port ethernet – general

The Eaton AC drives can be connected to the Ethernet networks using the Eaton OPTE9 Dual Port Ethernet fieldbus option board (OPTE9). The drives can be daisy chained by utilizing two Ethernet ports of OPTE9. The option board supports PROFINET IO, Ethernet/IP and Modbus TCP field bus protocols. The following network topologies are supported. See details in Chapter 4.4 "Connections and wiring".

- Star
- Daisy chain
- Ring

Every appliance connected to an Ethernet network has two identifiers: a MAC address and an IP address. The MAC address (Address format: xx:xx:xx:xx:xx:xx) is unique for each appliance and cannot be changed.The Ethernet board's MAC address can be found on the sticker attached to the board.

In a local network, IP addresses can be defined by the user as long as all the units connected to the network are given the same network portion of the address. Overlapping IP addresses cause conflicts between appliances. For more information about setting IP addresses, see Chapter 6.

Table 2. List of abbreviations used in this document

Abbreviation	Explanation
CRC	Cyclic redundancy check is an error-detecting code commonly used in field busses to detect accidental changes to raw data.
HI	Upper 8/16 bits in a 16/32 bit value.
LO	Lower 8/16 bits in a 16/32 bit value.
DHCP	Dynamic host configuration protocol is used for dynamical resolving of network configuration parameters like an IP address.
FB	Fieldbus
GW	Gateway
LWIP	Light weight TCP/IP protocol stack for embedded systems.
Modbus TCP	Simple and vendor-neutral communication protocol intended for monitoring and controlling of field devices.
PLC	Programmable logic controller
PDI	Process data in (Profinet IO)
PDO	Process data out (Profinet IO)
PHY(X)	Ethernet physical interface X, where X represents the number of interface
PNU	Parameter number (Profinet IO)
Profinet IO	Profinet is a standard for industrial automation in ethernet network. Profinet IO describes the exchange of data between controllers and field devices.
RPM	Revolutions per minute
TCP	Transmission control layer provides reliable, ordered and error-checked delivery of data streams between computers that are connected to a local area network.
RSTP	Rapid spanning tree protocol
ACD	Address conflict detection

Table 3. List of data types used in this document

Type name	Bit size	Explanation
INT8	8	Signed short integer
UINT8	8	Unsigned short integer
INT16	16	Signed integer
UINT16	16	Unsigned integer
INT32	32	Signed long integer
UINT32	32	Unsigned long integer
FLOAT32	32	32-bit floating point
STRING3	24	Three byte string
STRING5	40	Five byte string

2.1 New features

The following table shows the new features that are added in the OPTE9 Dual Port Ethernet's firmware versions.

Table 4. New features

New feature	Firmware version
EtherNet/IP protocol	V004
Ethernet ring support [RSTP]	V004

3. OPTE9 ethernet board technical data

3.1 General

Table 5. Technical data

General	Board name	OPTE9
Ethernet connections	Interface	Two RJ-45 connectors
	Transfer cable	Shielded twisted pair (STP) CAT5e
Communications	Speed	10/100 Mb
	Duplex	half/full
	Default IP-address	By default the board is in DHCP mode
Protocol	Modbus TCP, Profinet I/O,	EtherNet/IP
Environment	Ambient operating temperature	-10°C…50°C
	Storing temperature	-40°C…70°C
	Humidity	<95%, no condensation allowed
	Altitude	Max. 1000 m
	Vibration	0.5 G at 9...200 Hz
Safety		Fulfills EN50178 standard

3.2 Cables

For connecting the OPTE9 devices, use only Ethernet cables that meet at least the requirements of category 5 (CAT5) according to EN 50173 or ISO/IEC 11801.

4. Layout and connections

The Eaton OPTE9 Dual Port Ethernet option board is connected to the Ethernet bus using the standard RJ45 connectors (1 and 2). The communication between the control board and the AC drive takes place through a standard Eaton Interface Board Connector.

4.1 Layout and connections

Figure 1. The OPTE9 option board

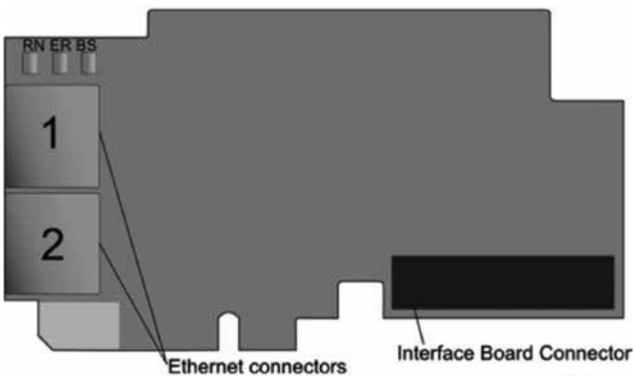
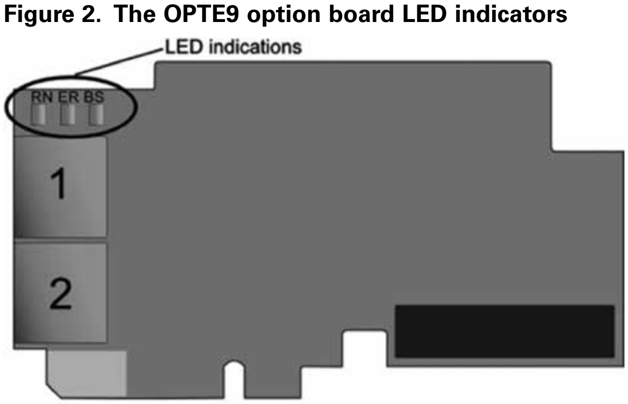


Table 6. OPTE9 Ethernet ports

Ethernet port	Description
1	Ethernet port 1 (PHY1)
2	Ethernet port 2 (PHY2)

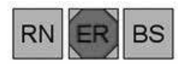
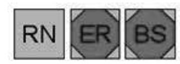
4.2 LED Indications

Figure 2. The OPTE9 option board LED indicators



The table below lists possible LED combinations and their meanings. When the EtherNet/IP is active, the option board follows CIP standard for LED indications. Therefore, the indications described in Table 7 do not apply. See Chapter 9.1.4 "LED functionality".

Table 7. List of possible LED combinations

LED combinations	Description
	No power. All LEDs are OFF.
	Option board firmware is corrupted or its software is missing. ER is blinking (0.25s ON/0.25s OFF)
	Option board failure. Option board is not operational. BS and possibly ER are blinking (2.5s ON/2.5s OFF)
	Option board is operational.
	Protocol is ready for communications. RN is blinking (2.5s ON/2.5s OFF).
	Protocol is communicating.
	Protocol communication fault. ER is blinking to indicate a fault. RN is blinking to indicate that protocol is again ready for communications.
	Protocol is communicating with an active fault. ER is blinking.
	Duplicate IP address detected. RN is blinking.
	**Profinet IO only!** In node flashing test all three LEDs are blinking.

4.2.1 Profinet IO

When using the "Node Flashing Test" function, you can determine to which device you are directly connected. For example, in Siemens S7, by using the menu command "PLC > Diagnostics/Setting > Node Flashing Test..." you can identify the station directly connected to the PG/PC if all three LEDs are flashing green.

4.3 Ethernet devices

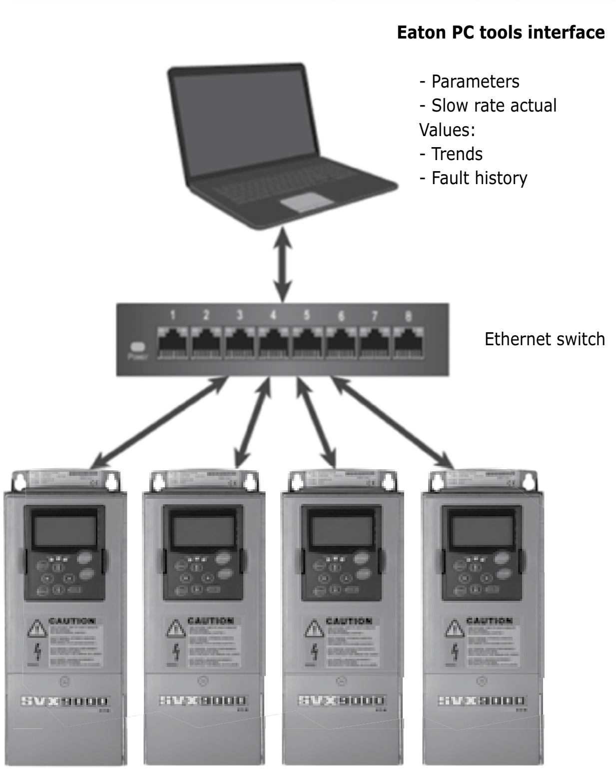
The common-use cases of Ethernet devices are 'human to machine' and 'machine to machine'. The basic features of these two cases are presented in the pictures below.

4.3.1 Human to machine

Requirements:

- Graphical User Interface
- Relatively slow communication in use

Figure 3. Ethernet, human to machine



Note: 9000xdrive can be used in SVX and SPX drives via Ethernet.

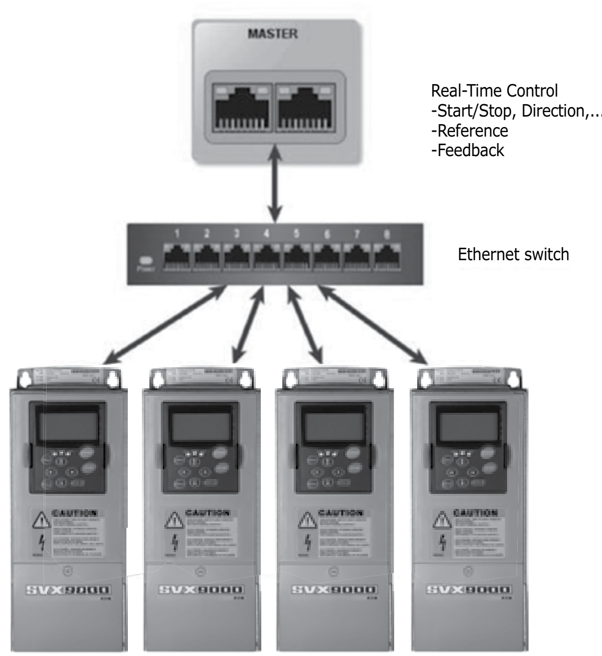
4.3.2 Machine to machine

Requirements:

– Industrial environment

– Fast communication in use

Figure 4. Ethernet, machine to machine



4.4 Connections and wiring

The OPTE9 has two Ethernet ports and an embedded switch. The option board is seen in network as a single device as it has only one MAC and IP address. This configuration enables three different topologies:

- Star (see Chapter 4.4.1)
- Daisy chain (see Chapter 4.4.2)
- Ring (see Chapter 4.4.3)

Each of these topologies has their own advantages and disadvantages. When designing the network you must carefully consider the risks and benefits against the cost of the selected topology.

The OPTE9 supports 10/100Mb speeds in both Full- and Half-duplex modes. However, real-time process control requires the Full-duplex mode and the 100-megabit speed. The boards must be connected to the Ethernet network with a Shielded Twisted Pair (STP) CAT-5e cable (or better).

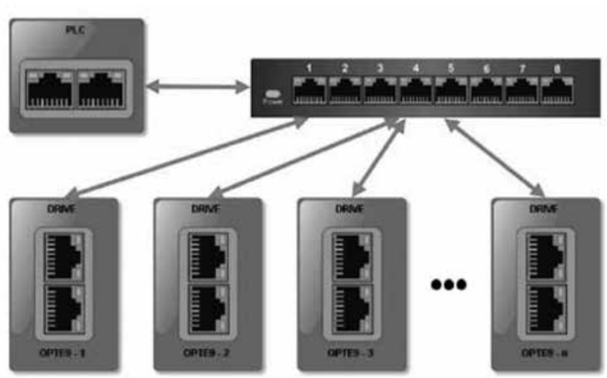
Use only industrial standard components in the network and avoid complex structures to minimize the length of response time and the amount of incorrect dispatches. Because of the internal switch in OPTE9, it does not matter in what port of the option board the Ethernet cables are connected to.

4.4.1 Topology: Star

In star network, all the devices are connected to the same switch(es). This topology reduces the damage caused by single cable failure. It would affect only to a single drive instead of them all. In this setup, a drive will receive only broadcast/multicast messages and messages directed to this drive.

Only one port from the OPTE9 can be connected to a switch in the star topology. Connecting both ports to switch(es) will cause an involuntary Ethernet ring which, in this setup, will break the network.

Figure 5. Star topology

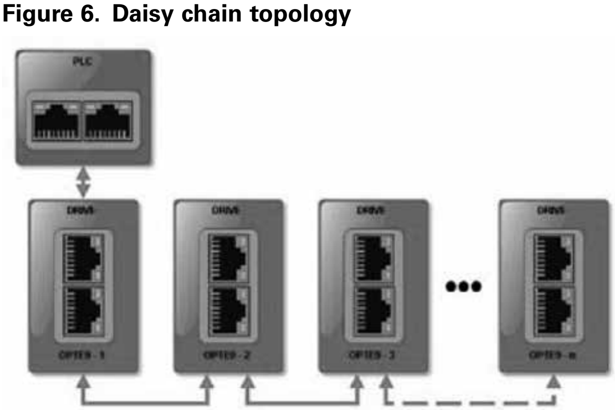


4.4.2 Topology: Daisy chain

The daisy-chaining allows you to reduce the costs for cabling and networking equipment such as switches. The maximum number of daisy-chained boards is 32. This restriction comes from the average latency (20 to 40 microseconds) per Ethernet switch. The drawback in the daisy chain topology is that it increases traffic in all except the last drive. The first drive in the daisy chain sees all the traffic in the chain. Also damage to a single cable will drop all drives behind it from the network.

Both in daisy chain topology and in star topology, the last drive's port must not be connected back to the same line. This would cause an involuntary Ethernet ring which will break the network.

Figure 6. Daisy chain topology



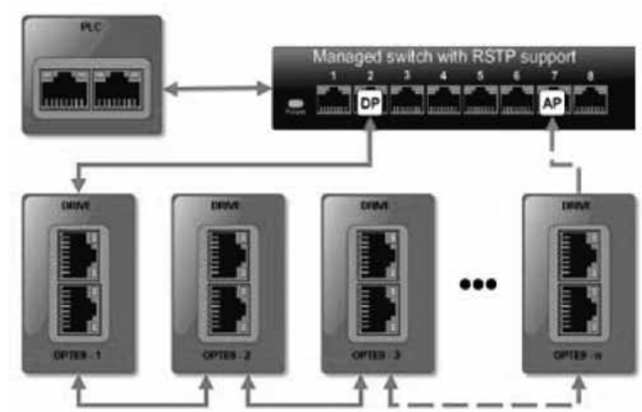
4.4.3 Topology: Ring

It is possible to use the OPTE9 option board in a ring topology by adding a managed Ethernet switch that supports the RSTP protocol. This topology gains the same reduced cabling cost as the daisy chain topology, but decreases the damage caused by a single cable failure. If a single link is broken, the RSTP switch will notice this and start sending data from the PLC to both directions effectively creating two daisy chains. When the link has been repaired, the switch will notice this too and reverts back to normal operating mode. Compared to the star topology, the ring topology adds more network traffic to almost all drives. Damage to two cables will always create an isolated subnetwork.

In the RSTP configuration, one of the ports in the switch is "Designated Port" (DP) and the other "Alternative Port" (AP). When the network is functioning properly, the traffic flows through the designated port. Only the BPDU (Bridge Protocol Data Unit) packets are transferred through the AP port. The BPDU packets are used by the switch to determine if the network is working properly. If it detects that the BPDU packets do not go through the ring, it will change the alternative port to a second designated port. Now the switch will send packets to both directions in the broken ring (see Figure 8).

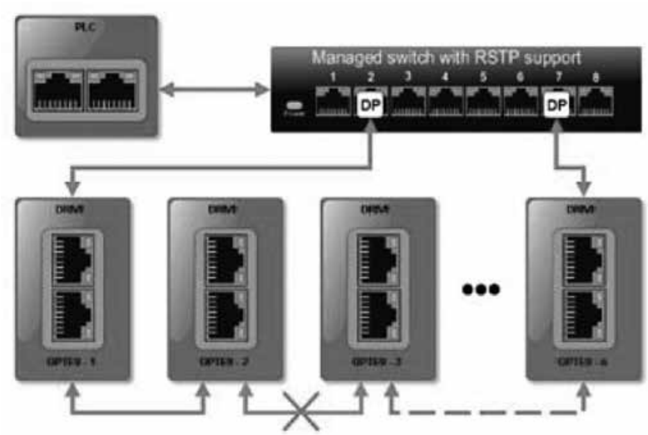
Each designated port has a list of MAC addresses which are behind that port. Only frames directed to the device in the MAC list are forwarded into that designated port. The broadcast and multicast frames are sent to all designated ports.

Figure 7. Ring topology



In the example below, the Ethernet communication will be interrupted to device number three and other devices after that when the link is broken. The Fieldbus communication maybe faulted when the link is broken, but when the switch enables the second designated port, the connections can be reopened. In the RSTP protocol, it generally takes few seconds before the second designated port will be activated.

Figure 8. Ring °topology: Error in network

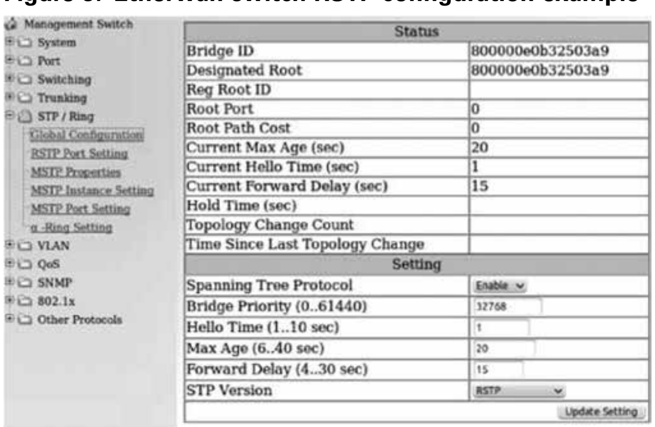


Note: The OPTE9 switch itself does not implement the RSTP protocol, so the network will always need a third party switch to support it.

Configuration example

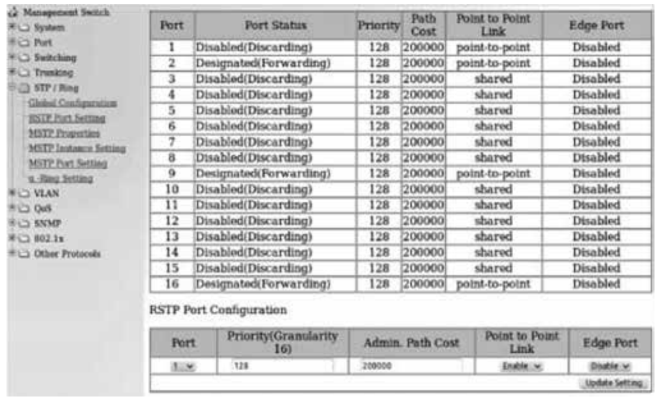
The screenshots below (Figure 9, Figure 10) show one example of configuring the RSTP in the switch (in this case an EtherWAN switch). Port two is the designated port and port one is the alternative port. The PLC was connected to port nine (the laptop taking the screenshots was in port 16). When configuring your switch, refer to the switch manufacturer's manual.

Figure 9. Etherwan switch RSTP configuration example



Status	
Bridge ID	800000e0b32503a9
Designated Root	800000e0b32503a9
Reg Root ID	
Root Port	0
Root Path Cost	0
Current Max Age (sec)	20
Current Hello Time (sec)	1
Current Forward Delay (sec)	15
Hold Time (sec)	
Topology Change Count	
Time Since Last Topology Change	
Setting	
Spanning Tree Protocol	Enable
Bridge Priority (0..61440)	32768
Hello Time (1..10 sec)	1
Max Age (6..40 sec)	20
Forward Delay (4..30 sec)	15
STP Version	RSTP

Figure 10. Etherwan switch RSTP configuration example – port settings



Port	Port Status	Priority	Path Cost	Point to Point Link	Edge Port
1	Disabled(Discarding)	128	200000	point-to-point	Disabled
2	Designated(Forwarding)	128	200000	point-to-point	Disabled
3	Disabled(Discarding)	128	200000	shared	Disabled
4	Disabled(Discarding)	128	200000	shared	Disabled
5	Disabled(Discarding)	128	200000	shared	Disabled
6	Disabled(Discarding)	128	200000	shared	Disabled
7	Disabled(Discarding)	128	200000	shared	Disabled
8	Disabled(Discarding)	128	200000	shared	Disabled
9	Designated(Forwarding)	128	200000	point-to-point	Disabled
10	Disabled(Discarding)	128	200000	shared	Disabled
11	Disabled(Discarding)	128	200000	shared	Disabled
12	Disabled(Discarding)	128	200000	shared	Disabled
13	Disabled(Discarding)	128	200000	shared	Disabled
14	Disabled(Discarding)	128	200000	shared	Disabled
15	Disabled(Discarding)	128	200000	shared	Disabled
16	Designated(Forwarding)	128	200000	point-to-point	Disabled

RSTP Port Configuration

Port	Priority(Granularity 16)	Admin. Path Cost	Point to Point Link	Edge Port
1	128	200000	Enable	Disable

4.5 ACD (address conflict detection)

The OPTE9 option board implements ACD algorithm (IETF RFC 5227). The implementation includes requirements from the EtherNet/IP protocol. The ACD algorithm tries to actively detect if the IP address configured to this device is been used by another device in the same network. To accomplish this, ACD sends four ARP request packets when the device's Ethernet interface goes up or when its IP address changes. ACD prevents the use of the Ethernet interface until the ARP probing finishes. This delays the startup of fieldbus protocols about one second. During the delay or after it, the ACD passively checks incoming ARP messages for use of the device's IP address. If another device with the same IP address is detected, the ACD will try to defend its IP address with a single ARP message. If the other device with the same IP address also supports ACD, it should stop using the address. If not, the ACD will close the Ethernet connection and indicate the situation with LEDs.
This is done according the "DefendWithPolicyB".

Other policies are not supported. If the fieldbus protocol has been active, a fieldbus fault may be activated (depends on the fieldbus and drive application configuration).

5. Installation

The Eaton OPTE9 Dual Port Ethernet option board can be used with the following Eaton AC drives.

Table 8. Supported drives and slots

Drive	Slots	From drive SW version on	From OPTE9 SW version on
SPX	D, E	SPX00032V025	V001
SVX	D, E	SVX00031V030	V001

EtherNet/IP support

EtherNet/IP protocol was added to OPTE9 firmware version V004. The table below shows required minimum drive firmware version.

Table 9. Required minimum drive firmware versions

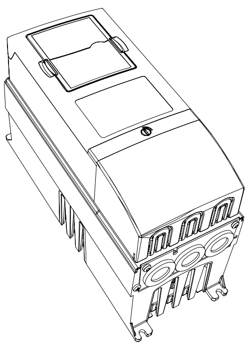
Drive	From drive SW version on
SPX	SPX00002V191
SVX	SVX00002V181

5.1 Installation in 9000x drives

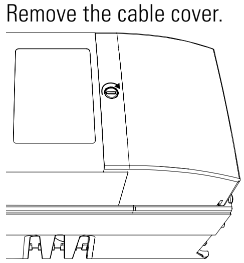
WARNING

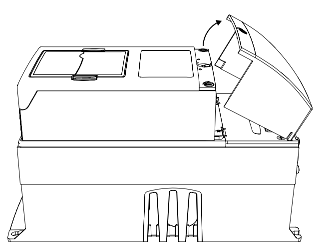
Make sure that the AC drive is switched off before an option or fieldbus board is changed or added!

1. Eaton 9000x AC drive.

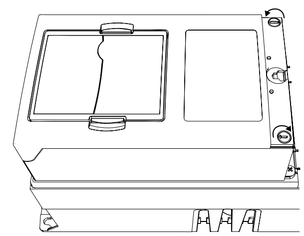


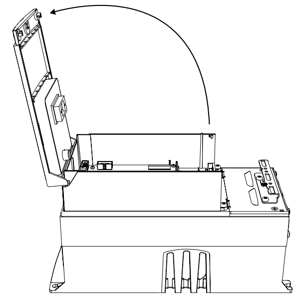
2. Remove the cable cover.



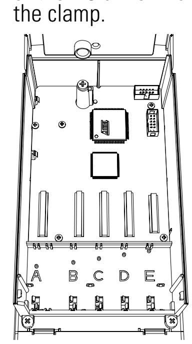


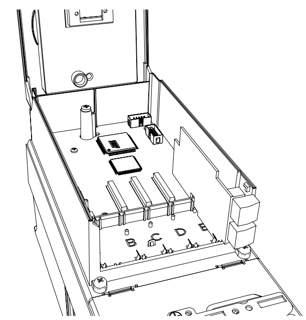
3. Open the cover of the control unit.



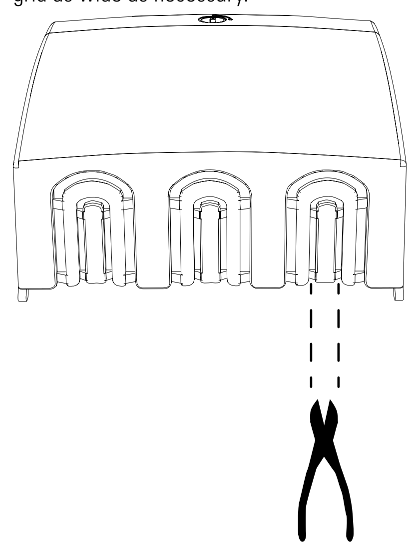


4. Install the OPTE9 Option Board in slot D or E on the control board of the AC drive. Make sure that the grounding plate fits tightly in the clamp.





5. Make a sufficiently wide opening for your cable by cutting the grid as wide as necessary.



6. Close the cover of the control unit and the cable cover.

5.2 PC Tools

Before connecting the OPTE9 option board to the network, its IP addresses must be set according to the network. By default, the option board uses a DHCP server to get an IP address. If your network does not have a DHCP server, you need to set an IP address manually. This can be accomplished with the PC tools described in this chapter or with the drive's keypad (see Chapter 6).

For more information about IP addresses or a DHCP server, contact your network administrator.

5.2.1 PC Tool support

This table describes what PC tools are supported in each drive type. The connection type "serial" means a direct connection to the drive. The connection type "Ethernet" means a connection via the OPTE9 Ethernet port.

Table 10. The supported PC tools with different drives

	9000x	
Tool	**Serial**	**Ethernet**
9000xLoad	x	
NCIPConfig		x
9000xdrive		x
9000xLoad	Not supported with OPTE9 Dual Port Ethernet	

5.2.2 Updating the OPTE9 option board firmware with MaxLoader

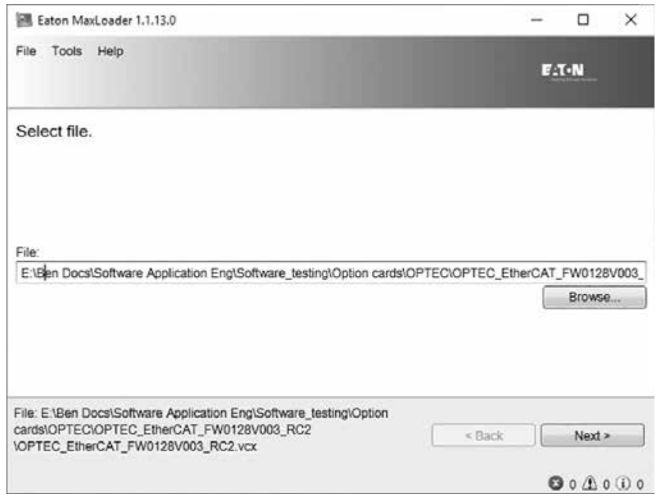
The MaxLoader can be downloaded from www.eaton.com/drives. It has been bundled with the Eaton Live software package.

To update the option board firmware, follow the steps below.

Step 1. Connect your PC to the controller by using the USB/RS485 cable.

Then select the firmware file which you want to load to the option board and double click it. This will start the MaxLoader software. You can also start the program from the Windows Start menu. In this case, select the firmware file using the "Browse"-button (see Figure 11).

Figure 11. EatonLoader: File selection



Step 2. Press 'next' and wait for the loader to find the network drives.

Then select a drive from the list and press 'Connect to Selected'. See Figure 12.

Figure 12. EatonLoader: Connecting to drive

Step 3. Select the modules to be updated, press 'next' and wait until the operation is finished. See Figure 13 and Figure 14.

Figure 13. Option board slot selection

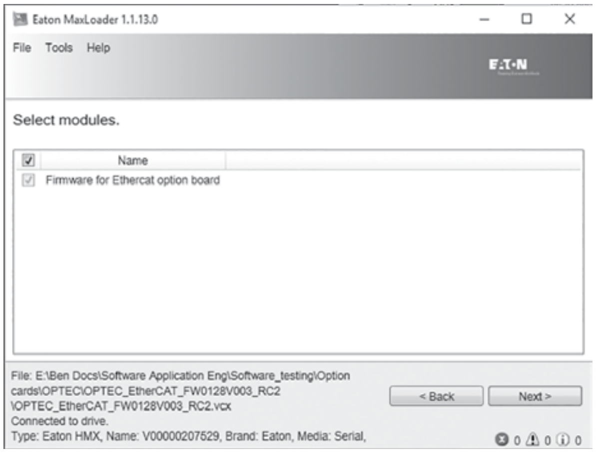
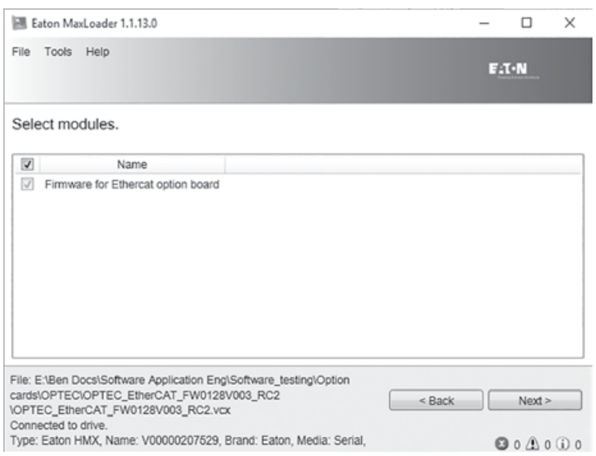


Figure 14. Eatonloader: Loading is finished



5.2.3 PC Tools for 9000x/NCIPConfig

The Eaton OPTE9 Dual Port Ethernet option board can be configured with the NCIPConfig tool.

Before the option board can be used, a valid IP address must be set. By default, the OPTE9 uses a DHCP server. If your network does not have a DHCP server, you will need to set an IP address manually and change the "IP Mode" to "static".

For more information about IP addresses or a DHCP server, contact your network administrator.

To install the NCIPConfig tool, start the installation program from the CD or download it from www.Eaton.com website. After starting the installation program, follow the on-screen instructions.

Once the program is installed successfully, you can launch it by selecting it in the Windows Start menu. Follow these instructions to set the IP addresses. Select Help --> Manual if you want more information about the software features.

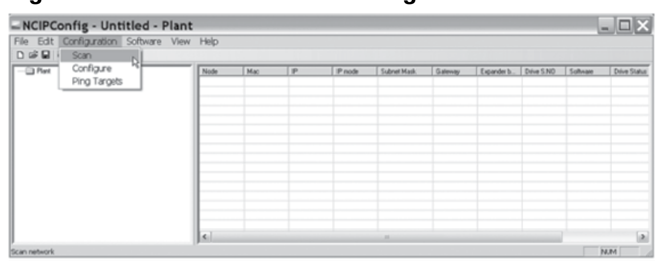
Step 1. Connect your PC to the ethernet network with an ethernet cable.

You can also connect the PC directly to the device using a crossover cable. This option may be needed if your PC does not support the Automatic crossover function.

Step 2. Perform network nodes scanning.

Select **Configuration --> Scan** (Figure 16) and wait until the devices connected to the bus in the tree structure are displayed on the left side of the screen.

Figure 15. Network nodes scanning



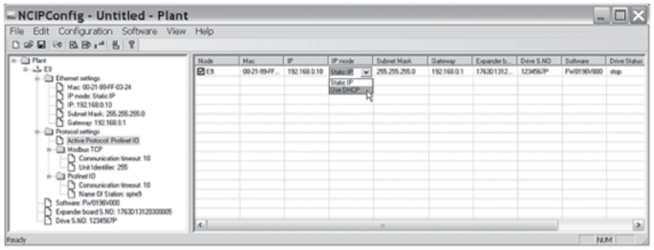
Note: The NCIPConfig uses broadcast messages for scanning devices. Some network switches might block the broadcast messages. In this case, each network node must be scanned separately.

Step 3. Set the option board settings.

To change the board name, select the cell in the column 'Node' and enter the name of the node. To change the node IP settings, select the cell in the right column and enter the value according to the network IP settings. The program will report conflicts with a red color in table cells. To change the IP Mode, click the cell and select the desired mode from the dropdown list (Figure 17).

To commit the changes, mark the checkbox and select Configuration->Configure- from the menu.

Figure 16. Change the option board settings



Step 4. Change the protocol settings.

To change the currently active protocol, select the setting from the tree structure. A dialog box opens. Select the desired protocol from the dropdown list (Figure 18). After clicking "ok" the setting will be activated.

The rest of the settings can be changed similarly, but values are edited in the tree (Figure 19). See Chapter 6 for more information about the settings.

Figure 17. Change the currently active protocol value

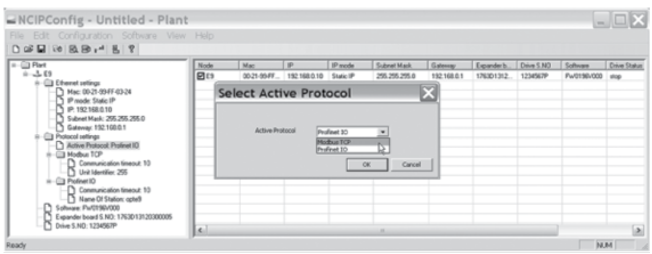
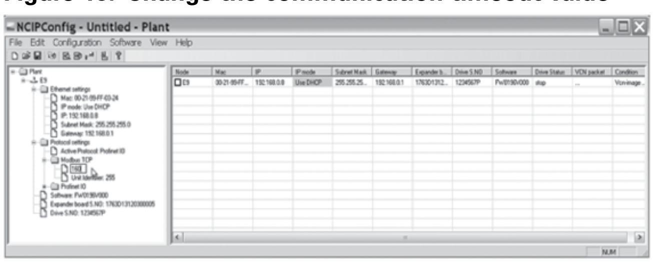


Figure 18. Change the communication timeout value



5.2.4 PC Tools for 9000xDrive

You can configure the drive parameters with the 9000xdrive. Some of the OPTE9 parameters can be configured with the 9000xdrive. However, it is recommended to use the NCIPConfig tool for the OPTE9 Dual Port Ethernet configuration in the 9000x drives.

You need to have a PC with an Ethernet connection and the 9000xdrive tool installed. To install the 9000xdrive, start the installation program from the CD or download it from www.Eaton.com website. After starting the installation program, follow the on-screen instructions.

Once the program is installed successfully, you can launch it by selecting it in the Windows Start menu. Select Help –> Contents if you want more information about the software features.

Before using the 9000xdrive, you need to configure the option board IP settings with NCIPConfig. If the option board does not have valid IP settings you will not be able to connect with the 9000xdrive.

Step 1. Connect your PC to the ethernet network with an ethernet cable.

You can also connect the PC directly to the device using a crossover cable. This option may be needed if your PC does not support Automatic crossover function.

Step 2. In order to connect to the drive, you need to select the active drive first. Press the "Drive Select" button (see Figure 20) to scan the network drives.

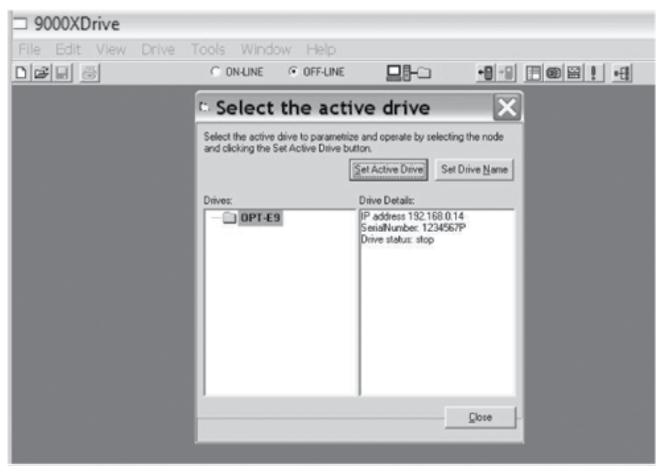
Figure 19. 9000X Drive: "Drive select"

Step 3. In the "Select the active drive" dialog (see Figure 21), select the drive you want to connect to.

Then press the "Set Active Drive" button. Now you can close the dialog.

The IP information presented in the dialog comes from the option board, other information comes from the drive.

Figure 20. 9000X Drive: Active drive selection



to change "IP Mode" to "Fixed IP" in order to activate the settings.

For more information about these settings, see Chapter 6.1.

Figure 21. 9000X Drive: OPTE9 parameters

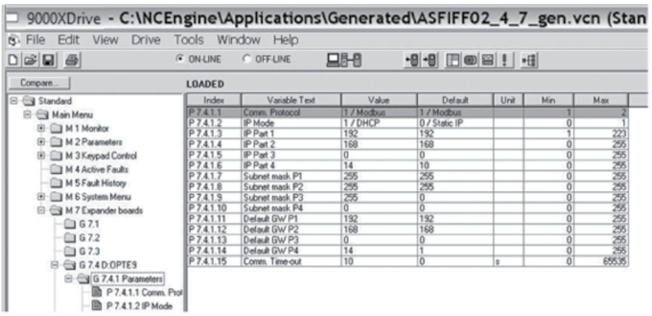
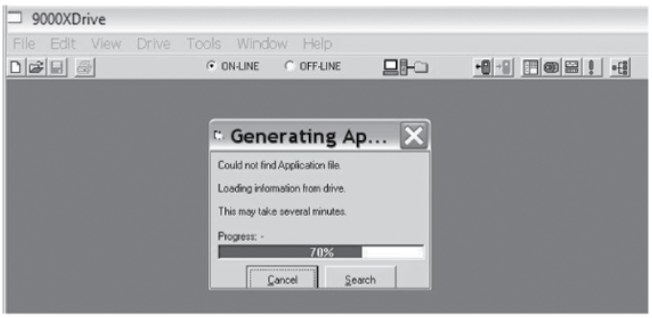


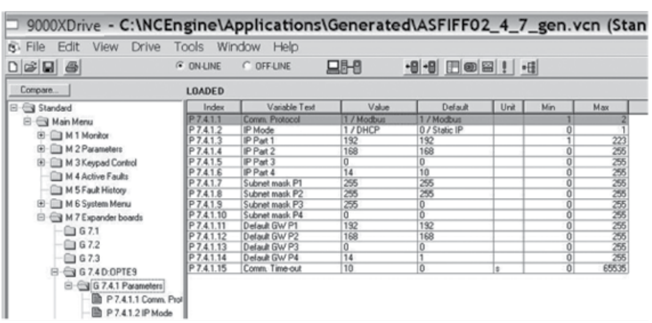
Figure 22. Eaton live: The "startup mode" dialogue box



Step 5. To change the option board settings, navigate to the "M7Expander boards" menu and select the slot that the OPTE9 is connected to. You can change the IP address, network mask and default gate address in the menu item "G 7.x". After you have changed the IP settings, you need to change "IP Mode" to "Fixed IP" in order to activate the settings.

For more information about these settings, see Chapter 6.1.

Figure 23. 9000X Drive: OPTE9 parameters



Note: The 9000xdrive software can be used with the Ethernet board in SVX, SPX and 9000xL drives.

Note: The 9000xdrive software is recommended to be used in LAN (Local Area Network) only.

6. Commissioning

The Eaton OPTE9 Dual Port Ethernet option board is commissioned with the control keypad by giving values to appropriate parameters in the option board menu (or via PC tools, see Chapter 5.6 "PC Tools").

Keypad commissioning procedures and location of parameters differ a little with different drive types:

- In the SPX/SVX option board, parameters are located under the menu M5 (Expander board menu)

6.1 Option board menu

The control keypad makes it possible for the user to see which expander boards are connected to the control board and to reach and edit the parameters associated with the expander board.

6.1.1 Option board parameters-menu

The OPTE9 board parameters are listed in the table below. All the option board parameters are saved to the option board (not to the control board). If the Ethernet board is replaced by a new one, you must re-configure the new board.

Table 11. Parameters menu structure

#	Name	Default	Range	Description
1	Comm. Protocol	Modbus TCP	Modbus TCP (1), Profinet IO (2), EtherNet/IP (3)	Active protocol
2	IP Mode	DHCP	Fixed IP (1), DHCP (2)	IP mode. When in DHCP mode, the IP address cannot be changed manually.
3	IP Part 1	192	1…223	IP address part 1
4	IP Part 2	168	0…255	IP address part 2
5	IP Part 3	0	0…255	IP address part 3
6	IP Part 4	10	0…255	IP address part 4
7	Subnet mask P1	255	0…255	Subnet mask part 1
8	Subnet mask P2	255	0…255	Subnet mask part 2
9	Subnet mask P3	0	0…255	Subnet mask part 3
10	Subnet mask P4	0	0…255	Subnet mask part 4
11	Default GW P1	192	0…255	Default gateway part 1
12	Default GW P2	168	0…255	Default gateway part 2
13	Default GW P3	0	0…255	Default gateway part 3
14	Default GW P4	1	0…255	Default gateway part 4
15	Comm. Timeout	10	0…65535 s	Comm. Timeout
16	PNIO Name of station	See Chapter 6.1.7	1...240 char	For profinet io only. Text is not visible in the panel.
17	EIP Output instance	21	"20" (1), "21" (2), "23" (3), "25" (4), "101" (5), "111" (6), "128" (7), "131" (8)	Ethernet/IP output assembly instance.
18	EIP Input instance	71	"70" (1), "71" (2), "73" (3), "75" (4), "107" (5), "117" (6), "127" (7), "137" (8)	Ethernet/IP input assembly instance.
19	EIP Product code offset	0	0…99	Ethernet/IP product code offset. User can add value between 0 and 99 to product code base value. Final product code can be viewed from monitoring-menu

6.1.2 Option board monitor menu

The monitor menu shows the currently active IP settings. For example, these values will show ‘0’ when a DHCP server is trying to get an IP address. After the address is received, these values are updated.

Table 12. Monitor menu structure

#	Name	Range	Description
1	IP Part 1	1…223	Current IP address part 1
2	IP Part 2	0…255	Current IP address part 2
3	IP Part 3	0…255	Current IP address part 3
4	IP Part 2	0…255	Current IP address part 4
5	Subnet mask P1	0…255	Current subnet mask part 1
6	Subnet mask P2	0…255	Current subnet mask part 2
7	Subnet mask P3	0…255	Current subnet mask part 3
8	Subnet mask P4	0…255	Current subnet mask part 4
9	Default GW P1	0…223	Current default gateway part 1
10	Default GW P2	0…255	Current default gateway part 2
11	Default GW P1	0…255	Current default gateway part 3
12	Default GW P4	0…255	Current default gateway part 4
13	Fieldbus protocol status	Initializing (1), Stopped (2), Operational (3), Faulted (4)	
14	Communication status	0.0…64.999	0-64 Number of messages with errors 0-999 Number of messages without communication errors
15	Drive control word	–	Control word in drive format (hex)
16	Drive status word	–	Status word in drive format (hex)
17	Protocol control word	–	Control word in protocol format (hex)
18	Protocol status word	–	Status word in protocol format (hex)
19	EIP Product Code	–	Currently used EtherNet/IP Product Code

6.1.3 Communication protocol

The OPTE9 option board comes with several fieldbus protocols. The user can select the one used in their network from the list. Only one protocol can be active at a time.

6.1.4 IP Mode

The IP mode determines how the option board IP settings are set. If a DHCP server is selected, then the option board will try to retrieve its IP settings from the DHCP server connected to the local network. If the option board is unable to retrieve its IP settings, it will set a link-local address as the current IP address after about one minute (for example 169.x.x.x).

If “Fixed IP” is set as IP mode, the settings IP Part 1-4, Subnet Part 1-4 and Default gateway 1-4 are used.

6.1.5 IP Address

IP is divided into 4 parts. (Part = Octet). Changing these values does not have any effect if the current IP mode is “DHCP”. The value will become active when the mode is changed to “fixed IP”. When these values are changed and the mode is “fixed IP”, the changes are taken into use immediately.

6.1.6 Communication timeout

It defines how much time can pass from the last received message from the Master Device before a fieldbus fault is generated. The functionality of this value is protocol-specific.

A fieldbus fault is also generated if the Ethernet link is down for over 60 seconds after the device startup. The Ethernet link status is being checked until the fieldbus communication is activated. After that the active fieldbus protocol controls the activation of the fieldbus fault.

6.1.6.1 Modbus TCP

See Chapter "7.1.3.5.11. Modbus Communication and connection timeout".

6.1.6.2 Profinet IO and EtherNet/IP

For these protocols, this value is considered as an additional timeout. The protocol itself has timeout mechanism. When it notices that the connection has been lost, a fault activation is started. If communication timeout value is zero, the fault is activated immediately, otherwise the fault activates after a specified time. If the connection is reopened before the specified time has elapsed, no fault is activated.

6.1.7 Profinet IO – Name of station

The Profinet IO "Name of Station" parameter can be set via 9000XDrive or NCIPConfig. Other possibility is to set this name by writing it from the PLC. The parameter can be found from the same list as protocol selection and IP settings. The parameter is not visible in the keypad, only in the PC tools.

If no name is set, the option board will generate a temporary name. The name is formed from the drive power unit serial number or, if that value is not available, from the option board MAC address and from slot ID. The format is:

opt-<slot>-<unique identifier>.

Example: opt-e-v00000030473

Example: opt-e-mac-002199ff0329

Table 13. Mode values

Mode value	Description
Normal	Option board will identify itself as OPTE9 (depends on fieldbus protocol)
9000x Mode	Option board will identify itself as old C-series counterpart and will emulate selected features. Currently only when using EtherNet/IP protocol.

6.1.8 EIP Input and output instance

These parameters will show what instances are being used now. The instances actually used are taken from the IO connection open request. So, although these values are parameters they act more like monitoring values.

6.1.9 EIP product code offset

This value can be used to differentiate drives for the PLC program. For example, if one drive is running a different application (with different parameters) than other drives, this offset in the product code will enable the PLC to use a different EDS file to read those parameters from this drive.

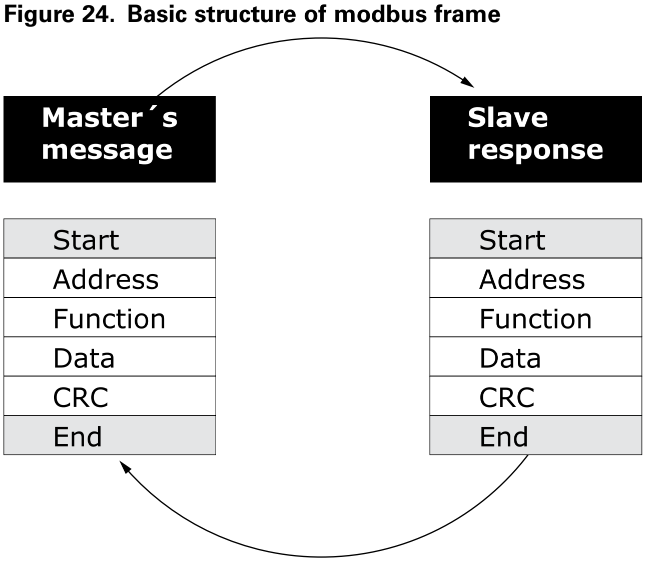
Remember that if you change this value, you need also to change the EDS file used or change the product code value in your EDS file.

7. Modbus TCP

Modbus is a communication protocol developed by Modicon systems. In simple terms, it is a way of sending information between electronic devices. The device requesting the information is called the Modbus Master (or the Client in Modbus TCP) and the devices supplying information are Modbus Slaves (in Modbus TCP servers). In a standard Modbus network, there is one Master and up to 247 Slaves, each with a unique Slave Address from 1 to 247. The Master can also write information to the Slaves. Modbus is typically used to transmit signals from instrumentation and control devices back to the main controller or data gathering system.

The Modbus communication interface is built around messages. The format of these Modbus messages is independent of the type of physical interface used. The same protocol can be used regardless of the connection type. Because of this, Modbus gives the possibility to easily upgrade the hardware structure of an industrial network, without the need for large changes in the software. A device can also communicate with several Modbus nodes at once, even if they are connected with different interface types, without the need to use a different protocol for every connection.

Figure 24. Basic structure of modbus frame



On simple interfaces like RS485, the Modbus messages are sent in plain form over the network. In this case, the network is dedicated to Modbus. When using more versatile network systems like TCP/IP over Ethernet, the Modbus messages are embedded in packets with the format necessary for the physical interface. In that case Modbus and other types of connections can co-exist at the same physical interface at the same time. Although the main Modbus message structure is peer-to-peer, Modbus is able to function on both point-to-point and multidrop networks.

Each Modbus message has the same structure. Four basic elements are present in each message. The sequence of these elements is the same for all messages, to make it easy to parse the content of the Modbus message. A conversation is always started by a master in the Modbus network. A Modbus master sends a message and depending of the contents of the message a slave takes action and responds to it. There can be more than one master in a Modbus network. Addressing in the message header is used to define which device should respond to a message. All other nodes on the Modbus network ignore the message if the address field does not match their own address.

If you need to contact Eaton service in problems related to Modbus TCP, send a description of the problem together with the Drive Info File to tech.supportVDF@Eaton.com. If possible, also send a "Wireshark" log from the situation if applicable.

7.1 Modbus TCP – Communications

The Modbus-Eaton interface features are presented below:

- Direct control of Eaton drive (e.g. Run, Stop, Direction, Speed reference, Fault reset)
- Access to Eaton parameters
- Eaton status monitoring (e.g. Output frequency, Output current, Fault code)

7.1.1 Data addresses in modbus messages

All data addresses in Modbus messages are referenced to zero. The first occurrence of a data item is addressed as item number zero. For example:

- The coil known as 'Coil 1' in a programmable controller is addressed as 'Coil 0000' in the data address field of a Modbus message
- Coil 127 decimal is addressed as 'Coil 007E hex' (126 decimal)
- Holding register 40001 is addressed as register 0000 in the data address field of the message. The function code field already specifies a 'holding register' operation. Therefore the '4XXXX' reference is implicit
- Holding register 40108 is addressed as register 006B hex (107 decimal)

7.1.2 Modbus memory map

The Eaton variables and fault codes as well as the parameters can be read and written from Modbus. The parameter addresses are determined in the application. Every parameter and actual value has been given an ID number in the application. The ID numbering of the parameters as well as the parameter ranges and steps can be found in the application manual in question. The parameter value are given without decimals. If several parameters/actual values are read with one message, the addresses of the parameters/actual values must be consecutive.

Table 14. Supported functions

Function code	Current terminology	Access type	Address range (hex)
1 (0x01)	Read coils	Discrete	00000-0FFFF
2 (0x02)	Read Input Discrete	Discrete	10000-1FFFF
3 (0x03)	Read holding registers	16bit	40000-4FFFF
4 (0x04)	Read input registers	16bit	30000-3FFFF
5 (0x05)	Force single coils	Discrete	00000-0FFFF
6 (0x06)	Write single register	16bit	40000-4FFFF
15 (0x0F)	Force multiple coils	Discrete	00001-0FFFF
16 (0x10)	Write multiple registers	16bit	40000-4FFFF
23 (0x17)	Read/Write multiple registers	16bit	40000-4FFFF

Note: Broadcasting is not supported in TCP.

7.1.3 Modbus data mapping

7.1.3.1 Coil registers

Coil registers contain binary data (Read/Write). See Table 15.

Table 15. Defined coil registers

Address	Function	Purpose
0001	RUN/STOP	Control Word, bit 0
0002	Direction	Control Word, bit 1
0003	Fault reset	Control Word, bit 2
0017	Reset	Clears operation days trip counter
0018	Reset	Clears energy trip counter

7.1.3.2 Clearing resettable counters

The Eaton drives have trip counters for operation days and energy. These counters can be reset to zero by writing value '1' to addresses defined in Table 16. Resetting the counters is not supported in Eaton 20, Eaton 20 X or Eaton 20 CP.

Table 16. Clearing trip counters

Address	Function	Purpose
40101	Reset	Clears operation days trip counter
40301	Reset	Clears energy trip counter

For compatibility with OPT-CI, these registers can be cleared also by writing '1' to these coils.

Address	Function	Purpose
0017	Reset	Clears operation days trip counter
0018	Reset	Clears energy trip counter

7.1.3.3 Input discrete registers

Input discrete registers contain binary data (Read). See Table 17.

Table 17. Defined input descrete registers

Address	Function	Purpose
10001	Ready	Status Word, bit 0
10002	Run	Status Word, bit 1
10003	Direction	Status Word, bit 2
10004	Fault	Status Word, bit 3
10005	Alarm	Status Word, bit 4
10006	At reference	Status Word, bit 5
10007	Zero speed	Status Word, bit 6
10008	Flux ready	Status Word, bit 7

7.1.3.4 Input registers

The values can be read with function code 4. These are for compatibility with the OPT-CI option board. They return the same values as holding register counterparts.

Address range	Purpose	Access type	See	R/W	Max R/W size
1 – 5	Operation day counter	16bit	Table 25	RO	5/0
101 – 105	Resettable operation day counter	16bit	Table 27	R, Write 1 to first index to reset	5/0
201 – 203	Energy counter	16bit	Table 29	RO	5/0
301 – 303	Resettable energy counter	16bit	Table 31	R, Write 1 to first index to reset	5/0
401 – 430	Fault history	16bit	Table 32	RO	30/0

7.1.3.5 Holding registers

The values can be read with function code 3. Modbus registers are mapped to drive IDs as follows:

Table 18. Defined holding registers

Address range	Purpose	Access type	See	R/W	Max R/W size
0001 – 2000	Eaton Application ID´s	16bit	Table 19	RW	30/30
2001 – 2011	FBProcessDataIN	16bit	Table 20	RW	11/11
2051 – 2070	FBProcessDataIN	32bit[1)]	Table 20	RW	20/20
2101 – 2111	FBProcessDataOUT	16bit	Table 21	RO	11/0
2151 – 2170	FBProcessDataOUT	32bit[1)]	Table 21	RO	20/0
2200 – 10000	Eaton Application ID´s	16bit	Table 19	RW	30/30
10501 – 10530	IDMap	16bit	Figure 28	RW	30/30
10601 – 10630	IDMap Read/Write	16bit	Table 22	RW	30/30[2)]
10701 – 10760	IDMap Read/Write	32bit[1)]	Table 22	RW	30/30
20001 – 40000	Eaton Application ID's	32bit[1)]	Table 19	RW	30/30
40001 – 40005	Operation day counter	16bit	Table 25	RO	5/0
40011 – 40012	Operation day counter	32bit[1)]	Table 24	RO	2/0
40101 – 40105	Resettable operation day counter	16bit	Table 27	R, Write 1 to first index to reset	5/0
40111 – 40112	Resettable operation day counter	32bit	Table 26	RO	2/0
40201 – 40203	Energy counter	16bit	Table 29	RO	3/0
40211 – 40212	Energy counter	32bit	Table 28	RO	2/0
40301 – 40303	Resettable energy counter	16bit	Table 31	R, Write 1 to first index to reset	3/0
40311 – 40312	Resettable energy counter	32bit	Table 30	RO	2/0
40401 – 40430	Fault history	16bit	Table 32	RO	30/0
40501	Communication timeout	16bit	Table 34	RW	1/1
40511-40568	Fault history with 16 bit fault codes	16bit	Table 33	RO	30/0

1) Not supported in current version. See chapter 5.

7.1.3.5.1. Eaton application IDs

Application IDs are parameters that depend on the drive’s application. These parameters can be read and written by pointing the corresponding memory range directly or by using the so-called ID map (more information below). The easiest way to read a single parameter value or parameters with consecutive ID numbers is to use a straight address. It is possible to read 30 consecutive ID addresses. Notice that the operation will fail if even one of the consecutive IDs do not exist.

Parameters which have 32 bit value can be read from their own range. For example, if you want to read the value for ID 864 (FB Status Word), the address must be set to 21726. This address value comes from values: 20000 + ((ID -1) * 2). The ID value is reduced with one because of zero-based addressing and the result is multiplied with 2 because one 32 bit value will take two (16 bit) addresses.

Table 19. Parameter IDs

Address range	Purpose	ID range
0001-2000	16 bit application parameters	1-2000
2200-10000	16 bit application parameters	2200-10000
20001-40000	32 bit application parameters	1-20000

7.1.3.5.2. FB process data in

The process data fields are used to control the drive (e.g. Run, Stop, Reference, Fault Reset) and to quickly read actual values (e.g. Output frequency, Output current, Fault code). The values in these indexes can be read and written. The fields are structured as follows.

Process data master -> Slave (max 22 bytes)

Table 20. Fieldbus process data IN

Address 16-bit*	32-bit	Name	Range/Type
2001	2051 = High data 2052 = Low data	FB Control Word	Binary coded
2002	–	In case of 16-bit, FB General Control Word (High data)	Binary coded
2003	2053 = High data 2054 = Low data	FB Speed Reference	0…10000 %
2004	2055 = High data 2056 = Low data	FB Process Data In 1	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2005	2057 = High data 2058 = Low data	FB Process Data In 2	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2006	2059 = High data 2060 = Low data	FB Process Data In 3	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2007	2061 = High data 2062 = Low data	FB Process Data In 4	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2008	2063 = High data 2064 = Low data	FB Process Data In 5	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2009	2065 = High data 2066 = Low data	FB Process Data In 6	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2010	2067 = High data 2068 = Low data	FB Process Data In 7	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2011	2069 = High data 2070 = Low data	FB Process Data In 8	See Chapter 11 "APPENDIX 1 – PROCESS DATA"

Control word bits

See Chapter 12 "APPENDIX 2 – CONTROL AND STATUS WORD" for control word bit descriptions.

7.1.3.5.3. FB Process data out

Values in these indexes can be only read, not written.

Table 21. Fieldbus process data OUT

Address 16-bit*	32-bit	Name	Range/Type
2101	2151 = High data 2152 = Low data	FB Status Word	Binary coded
2102	–	In case of 16-bit, FB General Status Word (High data)	Binary coded
2103	2153 = High data 2154 = Low data	FB Actual Speed	0…10000 %
2104	2155 = High data 2156 = Low data	FB Process Data Out 1	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2105	2157 = High data 2158 = Low data	FB Process Data Out 2	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2106	159 = High data 2160 = Low data	FB Process Data Out 3	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2107	2161 = High data 2162 = Low data	FB Process Data Out 4	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2108	2163 = High data 2164 = Low data	FB Process Data Out 5	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2109	2165 = High data 2166 = Low data	FB Process Data Out 6	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2110	2167 = High data 2168 = Low data	FB Process Data Out 7	See Chapter 11 "APPENDIX 1 – PROCESS DATA"
2111	2169 = High data 2170 = Low data	FB Process Data Out 8	See Chapter 11 "APPENDIX 1 – PROCESS DATA"

Status word bits

See Chapter 12 "APPENDIX 2 – CONTROL AND STATUS WORD" for status word bit descriptions.

The use of process data depends on the application. In a typical situation, the device is started and stopped with the Control Word (CW) written by the Master and the Rotating speed is set with Reference (REF). With PD1… PD8 the device can be given other reference values (e.g. Torque reference).

With the Status Word (SW) read by the Master, the status of the device can be seen. Actual Value (ACT) and PD1… PD8 show the other actual values.

7.1.3.5.4. ID map

Using the ID map, you can read consecutive memory blocks that contain parameters whose IDs are not in a consecutive order. The address range 10501 – 10530 is called 'IDMap', and it includes an address map in which you can write your parameter IDs in any order. The address range 10601 -10630 is called 'IDMap Read/Write,' and it includes values for parameters written in the IDMap. As soon as one ID number has been written in the map cell 10501, the corresponding parameter value can be read and written in the address 10601, and so on. The address range 10701 – 10730 contains the ID Map for 32bit values. Maximum of 30 IDs and ID values can be written and read with single request except in Eaton MMX it is possible to access only 12 ID value items at a time.

Note: 32 bit data not supported in the current version. See chapter 5.

Figure 25. ID Map initialization example

Parameters	
ID	Value
699	123
700	321
701	456
702	654
703	1789
704	987
705	2741
706	1147
707	258
708	3852

ID Map			
Address	Data: ID	Address	Data: ID
10501	700	10601	321
10502	702	10602	654
10503	707	10603	258
10504	704	10604	987

Once the ID Map address range has been initialized with the parameter IDs, the parameter values can be read and written in the IDMap Read/Write address range address (IDMap address + 100).

Table 22. Parameter values in 16-bit IDMap read/write registers

Address	Data
10601	Data included in parameter ID700
10602	Data included in parameter ID702
10603	Data included in parameter ID707
10604	Data included in parameter ID704

If the ID Map table has not been initialized, all the fields show index as '0'. If it has been initialized, the parameter IDs included in it are stored in the flash memory of the OPTE9 option board.

Table 23. Example of parameter values in 32-bit IDMap Read/Write registers

Address	Data
10701	Data High, parameter ID700
10702	Data Low, parameter ID700
10703	Data High, parameter ID702
10704	Data Low, parameter ID702

7.1.3.5.5. Operation day counter

Control unit operating time counter (total value). This counter cannot be reset. The values are read only.

Operation day counter as seconds

This counter in registers 40011d to 40012d holds the value of operation days as seconds in a 32-bit unsigned integer.

Table 24. Operation days counter as seconds

Address	Description
40011 High data 40012 Low data	Holds the counter value as seconds.

Operation day counter

This counter in registers 40001d to 40005d holds the value of operation days counter. The values are read only.

Table 25. Operation day counter

Holding register addres	Input register address	Purpose
40001	1	Years
40002	2	Days
40003	3	Hours
40004	4	Minutes
40005	5	Seconds

7.1.3.5.6. Resettable operation day counter

This register holds the value for resettable control unit operating time counter (trip value). The values are read only.

For resetting this counter see Chapter 7.1.3.2 "Clearing resettable counters".

Resettable operation day counter as seconds

This counter in registers 40111d to 40112d holds the value of resettable operation days as seconds in a 32-bit unsigned integer.

Table 26. Resettable operation days counter as seconds

Address	Description
40111 High data 40112 Low data	Holds the counter value as seconds.

Resettable operation day counter

This counter in registers 40101d to 40105d holds the value of operation days counter.

Table 27. Resettable operation day counter

Holding register addres	Input register address	Purpose
40101	101	Years
40102	102	Days
40103	103	Hours
40104	104	Minutes
40105	105	Seconds

7.1.3.5.7 Energy counter

This counter holds the value of total amount of energy taken from a supply network. This counter cannot be reset. The values are read only.

Energy counter as kWh

This counter is in registers 40211d to 40212d and is a 32-bit floating point (IEEE 754) value containing the number of kilowatt-hours (kWh) that is in the drive's energy counter. This value is read-only.

Table 28. Energy counter as kWh

Address	Description
40211 High data 40212 Low data	Holds the value of energy counter in kWh. Datatype is 32 bit float IEEE 754

Energy counter

These registers hold three values for the energy counter, amount of energy used, format of the energy value and unit of the energy value.

Example: If energy = 1200, format = 52, unit = 1, then actual energy is 12.00 kWh.

Table 29. Energy counter

Holding register address	Input register address	Purpose	Description
40201	201	Energy	Amount of energy taken from a supply network.
40202	202	Format	The last number of the Format field indicates the decimal point place in the Energy field. **Example:** 40 = 4 number of digits, 0 fractional digits 41 = 4 number of digits, 1 fractional digit 42 = 4 number of digits, 2 fractional digits
40203	203	Unit 1 = kWh 2 = MWh 3 = GWh 4 = TWh	Unit of the value.

7.1.3.5.8. Resettable energy counter

This counter holds the value of total amount of energy taken from a supply network since the counter was last reset. For resetting this counter see Chapter 7.1.3.2 "Clearing resettable counters". The values are read only.

Resettable energy counter as kWh

This counter is in registers 40311d to 40312d and is a 32-bit floating point (IEEE 754) value containing the number of kilowatt-hours (kWh) that is in the drive's resettable energy counter.

Table 30. Resettable energy counter as kWh

Address	Description
40311 High data 40312 Low data	Holds the value of energy counter in kWh since last counter reset. Datatype is 32 bit float IEEE 754

Resettable energy counter

These registers hold three values for the energy counter, amount of energy used, format of the energy value and unit of the energy value.

Example: If energy = 1200, format = 52, unit = 1, then actual energy is 12.00 kWh.

Table 31. Resettable energy counter

Holding register address	Input register address	Purpose	Description
40301	301	Energy	Amount of energy taken from a supply network.
40302	302	Format	The last number of the Format field indicates the decimal point place in the Energy field. **Example:** 40 = 4 number of digits, 0 fractional digits 41 = 4 number of digits, 1 fractional digit 42 = 4 number of digits, 2 fractional digits
40303	303	Unit 1 = kWh 2 = MWh 3 = GWh 4 = TWh	Unit of the value.

7.1.3.5.9. Fault history

The fault history can be viewed by reading from address 40401 onward. The faults are listed in chronological order so that the latest fault is mentioned first and the oldest last. The fault history can contain 29 faults at the same time.

Note: Reading the fault history items is slow. Reading all 30 items at once might take up to three seconds.

The fault history contents are represented as follows:

Table 32. Fault history

Holding register address	Input register address	Purpose
40401	401	Upper byte is a fault code, lower byte is a sub code
40402	402	
40403	403	
...	...	
40429	429	

7.1.3.5.10. Fault history with 16-bit error codes

The fault history can be viewed by reading from address 40511 onward. The faults are listed in a chronological order so that the latest fault is mentioned first and the oldest last. These addresses contain the fault code and the subcode for the fault. Reading can be started from any address.

Note: Reading the fault history items is slow. Reading all 30 items at once might take up to three seconds.

Table 33. Fault history with 16-bit error codes

Holding register address	Purpose	Description
40511	Fault code 1	16-bit fault code in index 1.
40512	Sub code 1	16-bit sub code for the fault in index 1.
40513	Fault code 2	16-bit fault code in index 2.
40514	Sub code 2	16-bit sub code for the fault in index 2.
...	...	
40567	Fault code 29	
40568	Sub code 29	

7.1.3.5.11. Modbus communication and connection timeout

It is possible to open up to three connections to the OPTE9 option board. One of the connections could be used for process data and other just for reading monitoring data. In most cases it is desirable that if "monitor" connection gets disconnected, no fault is generated but when the connection is handling the process data, a fault should be generated in the time specified.

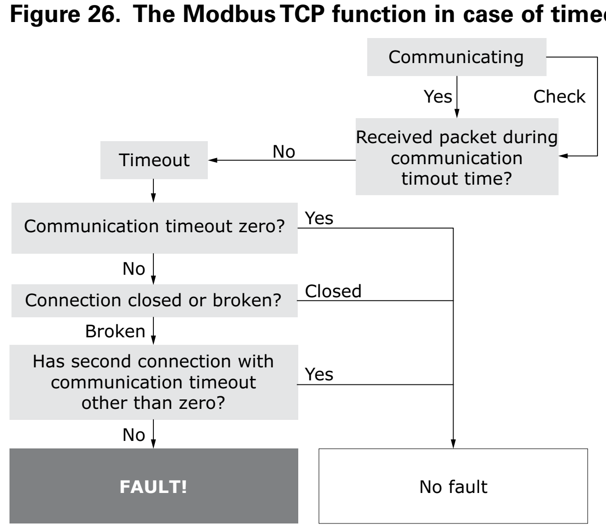
This register address enables the user to give custom communication timeout for each connection. If a custom timeout value is used, it must be given every time a connection is opened. Timeout can be set only to the connection which is been used to access this register. By default the connection uses the communication timeout value given via panel parameters.

If the cable is disconnected, a fieldbus fault is activated after the timeout period. When communication timeout is zero, no fault is activated.

Table 34. Communication timeout register

Holding register address	Purpose	Description
40501	Communication timeout	Connection timeout value for this connection in seconds.

Figure 26. The Modbus TCP function in case of timeout



7.1.4 Quick setup

Following these instructions, you can easily and fast set up your Modbus for use:

In the AC drive application: Choose Fieldbus as the active control place (see drives User's Manual).

In the master software:

1. Set the settings in the master software
2. Set the Control Word to '0' (2001)
3. Set the Control Word to '1' (2001)
4. Drive's status is RUN
5. Set the Reference value to '5000' (50.00%) (2003)
6. Actual speed is 5000 (25.00 Hz if MinFreq is 0.00 Hz and MaxFreq is 50.00 Hz)
7. Set the Control Word to '0' (2001)
8. Drive's status is STOP

7.2 Modbus TCP – Example messages

7.2.1 Example 1 – Write process data

Write the process data 42001…42003 with command 16 (Preset Multiple Registers).

Command master – Slave:

Address		01 hex	Slave address 1 hex (= 1)
Function		10 hex	Function 10 hex (= 16)
Data	Starting address HI	07 hex	Starting address 07D0 hex (= 2000)
	Starting address LO	D0 hex	
	No. of registers HI	00 hex	Number of registers 0003 hex (= 3)
	No. of registers LO	03 hex	
	Byte count	06 hex	Byte count 06 hex (= 6)
	Data HI	00 hex	Data 1 = 0001 hex (= 1). Setting control word run bit to 1.
	Data LO	01 hex	
	Data HI	00 hex	Data 2 = 0000 hex (= 0).
	Data LO	00 hex	
	Data HI	13 hex	Data 3 = 1388 hex (= 5000), Speed reference to 50.00%
	Data LO	88 hex	
Error check	CRC HI	C8 hex	CRC field C8CB hex (= 51403)
	CRC LO	CB hex	

Message frame:

01 10 07 D0 00 03 06 00 01 00 00 13 88 C8 CB

The reply to Preset Multiple Registers message is the echo of 6 first bytes.

Answer slave – Master:

Address		01 hex	Slave address 1 hex (= 1)
Function		10 hex	Function 10 hex (= 16)
Data	Starting address HI	07 hex	Starting address 07D0 hex (= 2000)
	Starting address LO	D0 hex	
	No. of registers HI	00 hex	Number of registers 0003 hex (= 3)
	No. of registers LO	03 hex	
Error check	CRC HI	80 hex	CRC 8085 hex (= 32901)
	CRC LO	85 hex	

Reply frame:

01 10 07 D0 00 03 80 85

7.2.2 Example 2 – Read process data

Read the Process Data 42103…42104 with command 4 (Read Input Registers).

Command master – Slave:

Address		01 hex	Slave address 1 hex (= 1)
Function		04 hex	Function 4 hex (= 4)
Data	Starting address HI	08 hex	Starting address 0836 hex (= 2102)
	Starting address LO	36 hex	
	No. of registers HI	00 hex	Number of registers 0002 hex (= 2)
	No. of registers LO	02 hex	
Error check	CRC HI	93 hex	
	CRC LO	A5 hex	CRC 93A5 hex (= 37797)

Message frame:

01 04 08 36 00 02 93 A5

The reply to the Read Input Registers message contains the values of the read registers.

Answer slave – Master:

Address		01 hex	Slave address 1 hex (= 1)
Function		04 hex	Function 4 hex (= 4)
Data	Byte count	04 hex	Byte count 4 hex (= 4)
	Data HI	13 hex	Speed reference = 1388 hex (=5000 => 50.00%)
	Data LO	88 hex	
	Data HI	09 hex	Output Frequency = 09C4 hex (=2500 =>25.00Hz)
	Data LO	C4 hex	
Error check	CRC HI	78 hex	
	CRC LO	E9 hex	CRC 78E9 hex (= 30953)

Reply frame:

01 04 04 13 88 09 C4 78 E9

7.2.3 Example 3 – Exception response

In an exception response, the Slave sets the most-significant bit (MSB) of the function code to 1. The Slave returns an exception code in the data field.

Command master – Slave:

Address		01 hex	Slave address 1 hex (= 1)
Function		04 hex	Function 4 hex (= 4)
Data	Starting address HI	17 hex	Starting address 1770 hex (= 6000)
	Starting address LO	70 hex	
	No. of registers HI	00 hex	Invalid number of registers 0005 hex (=5)
	No. of registers LO	05 hex	
Error check	CRC HI	34 hex	
	CRC LO	66 hex	CRC 3466 hex (= 13414)

Message frame:

01 04 17 70 00 05 34 55

Exception response

Answer slave – Master:

Address		01 hex	Slave address 1 hex (= 1)
Function		84 hex	Most significant bit set to 1
Data	Error code	04 hex	Error code 04 => Slave device failure
Error check	CRC HI	42 hex	
	CRC LO	C3 hex	CRC 42C3 hex (= 17091)

Reply frame:

01 84 04 42 C3

8. Profinet IO

PROFINET is the Ethernet-based automation standard of PROFIBUS International for the implementation of an integrated and consistent automation solution based on Industrial Ethernet. PROFINET supports the integration of simple distributed field devices and time-critical applications in (switched) Ethernet communication, as well as the integration of component-based distributed automation systems for vertical and horizontal integration of networks.

8.1 PROFIDrive 4.1 profile

To provide interoperability between devices from different manufacturers, a "standard" must be defined so that:

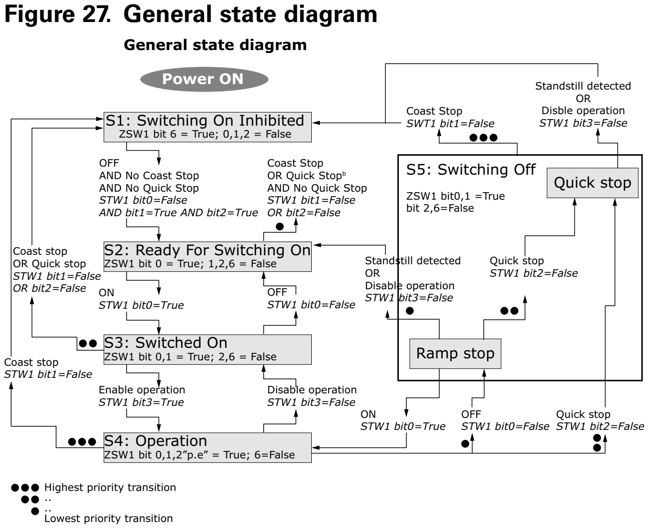
- The devices behave in the same way
- They produce and/or consume the same basic set of I/O data
- They contain the same basic set of configurable attributes

The formal definition of this information is known as a device profile.

8.2 PROFIDrive 4.1 State machine

STW1 (Control Word) and ZSW1 (Status Word) follow the state machine presented below:

Figure 27. General state diagram

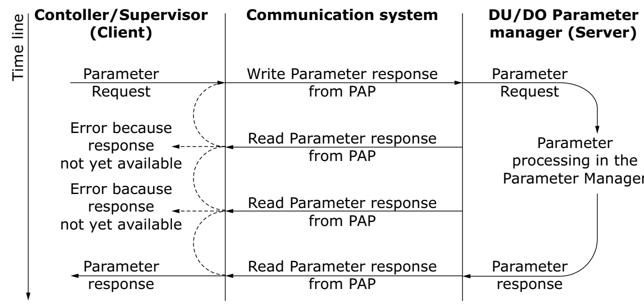


8.3 PROFIDrive parameters implemented by OPTE9

8.3.1 Base mode parameter access model

The PROFIDrive parameters are accessed according to the model presented below:

Figure 28. PROFIDrive parameter access model



Both indexes can be used to access PROFIDrive parameters. There is no difference in operation between them with current implementation.

Table 35. Parameter access services

Parameter access service	Index	
Base Mode Parameter – Local	0x	B02E
Base Mode Parameter – Global	0x	B02F

The structure of parameter requests is described in the table below:

Table 36. Parameter request

Block definition	Byte n+1	Byte n	n
Request header	Request reference	Request ID	0
	Axis-No./DO-ID	No. of parameters = n	2
1st Parameter address	Attribute	No. of elements	4
	Parameter number (PNU)		
	Subindex		
nth Parameter address	...		4 + 6 x (n – 1)
1st Parameter value(s) (only for request “change parameter”)	Format	No. of values	4 + 6 x n
	Values		
	...		
nth Parameter values	...		
			4 + 6 x n + ... + (For-mat_n x Qty_n)

The structure of parameter responses is described in the table below:

Table 37. Parameter response

Block definition	Byte n+1	Byte n	n
Response header	Request ref. Mirrored	Response ID	0
	Axis-No./DO-ID mirrored	No. of parameters = n	2
1st Parameter value(s) (Only for request “Request”)	Format	No. of values	4
	Values or error values		
	...		
nth Parameter values	...		
			4 + ... + (Format_n x Qty_n)

The table below contains descriptions of parameters.

Table 38. Parameter description

Sub-index	Field name	Data type	Description
1	Identifier (ID)	Unsigned16	A bitmask with information about the parameter characteristics.
2	Number of array elements	Unsigned16	For array parameters, the number of elements in the array.
3	Standardisation factor	FloatingPoint (IEEE 754)	If the information represented by the parameter can be converted into a standardised form, this field contains a factor for this conversion.
4	Variable attribute	Array of two Unsigned8	Contains two index numbers for describing the parameter information.
5	Reserved	Array of four Unsigned8	Reserved, always 0.
6	Name	ASCII string, 16 characters	Symbolic name of the parameter.
7	Low limit	Array of four Unsigned8	Limit for valid values of the parameter.
8	High limit	Array of four Unsigned8	Limit for valid values of the parameter.
9	Reserved	Array of two Unsigned8	Reserved, always 0.
10	ID extension	Unsigned16	Not used, always 0.
11	Normalisation reference parameter	Unsigned16	Parameter number, the value of which is used as normalisation reference for the parameter whose description this is.
12	Normalisation field	Unsigned16	Contains information about normalisation of this parameter.

8.3.1.1 Parameter requests

There are two types of parameter requests in PROFIDrive:

- Read requests for reading parameters from the device
- Change requests for writing parameters to the device

Each parameter request consists of three elements:

- Request header
- Parameter address
- Parameter value (only in Change requests)

Request header	Parameter address(es)	Parameter value(s)

8.3.1.2 Request header

The request header consists of 4 fields, each one octet in size.

Table 39. Request header

Octet number	Field name	Description	Allowed values
1	Request Reference	Unique number for each request/response pair. This value is changed by the master for each new request. It is mirrored by the slave in the response.	A bitmask with information about the parameter characteristics.
2	Request ID	Defines the type of request.	Use 0x01 for Read requests. Use 0x02 for Change requests. Other values are not allowed.
3	Axis Number	Not used, should be set to 1.	Use 1 for OPTE9 PROFINET IO. Other values should not be used.
4	Requested number of parameters	The number of parameters affected by the request.	Values 1 to 39 are allowed. The value 0 is not allowed. Values 40 to 255 are not allowed.

8.3.1.3 Parameter address

The parameter address consists of 4 fields, totaling six octets in size.

Table 40. Parameter address

Octet number	Field name	Description	Allowed values
1	Attribute	Describes which part of a parameter should be accessed.	Use 0x10 for reading/writing the value of a parameter. Use 0x20 for reading the description of a parameter. Use 0x30 for reading the text of a parameter (not supported). Other values should not be used.
2	Number of elements	Specifies the number of elements that are addressed in an array.	Values 0 and 1 are allowed for non-array parameters. Values 1 to 234 are allowed for array parameters. Other values should not be used.
3...4	Parameter number	The number of the parameter to be addressed.	Allowed values are those of supported parameters, see chapter 8.3.2.2.
5...6	Subindex	Defines the first array element of the parameter to be accessed.	Values 0 to 65535 are allowed. Other values are not allowed.

The “Parameter number” and “Subindex” fields are two-octet fields, while the “Attribute” and “No. of elements” fields are one-octet fields.

8.3.1.4 Parameter value

The parameter value field is included only in Change requests (not in Read requests). The parameter value field consists of a two-octet parameter value header followed by a list of values. Depending on the format of the parameter, the octet size of a single value is one, two or four octets. The total size of the parameter value field thus depends on the format and number of values in the message.

Table 41. Parameter value

Octet number	Field name	Description	Allowed values
1	Format	Describes the data type of the parameter.	Use 0x41 for Byte. Use 0x42 for Word. Use 0x43 for Double Word. Value 0x44 is used for Error. Other values should not be used.
2	Number of values	Defines the number of values in the parameter value field.	Values 0 to 234 are possible. Subject to limitations as described below.
3	Value	The value of the parameter.	Values 0 to 65535 are allowed. Other values are not allowed.

The number of values which can be transmitted in one frame has been limited to 39.

8.3.2 Parameter responses

There are two types of parameter responses in PROFIDrive:

- Write response (response to a Write request)
- Read response (response to a Read request)

A read response consists of two elements:

- Response header
- Parameter value(s) (depending on the request type)

Response header	Parameter value(s)

8.3.2.1 Error response

If an error occurred in the Parameter Access, the response provided by the slave is an error response. An error response contains 4 octets as described below.

Table 42. Error response

Octet number	Field name	Description	Allowed values
1	Function Number	Operation number.	The slave returns 0xDE to indicate an error read response. The slave returns 0xDF to indicate an error write response. Other values are not used in the Parameter Access.
2	Error Decode	Defines how the error information in the following two fields should be decoded.	Always 128 in PROFIDrive. Other values are not used in the Parameter Access.
3	Error Code 1	High 4 bits indicate error class, 4 lower bits indicate error code.	See Chapter 8.3.2.3 “PROFIDrive Parameter Access errors”
4	Error Code 2	Application-specific.	Always 0 in PROFIDrive.

8.3.2.2 PROFIDrive 4.1 error classes and codes

The table below lists PROFIDrive profile 4.1 error classes and codes.

Table 43. PROFIDrive 4.1 error classes and codes

Error class	Error codes	PROFIDrive meaning
0x0 0x9 = reserved (not used)		
0xA = application	0x0 = read error 0x1 = write error 0x2 = module failure 0x3…0x7 = reserved (not used) 0x8 = version conflict 0x9 = feature not supported 0xA…0xF = user-specific (not used)	
0xB = access	0x0 = invalid index	0xB0 = parameter requests are not supported.
	0x1 = write length error 0x2 = invalid slot 0x3 = type conflict 0x4 = invalid area	
	0x5 = state conflict	0xB5 = parameter access is temporarily not possible due to internal processing status.
	0x6 = access denied	
	0x7 = invalid range	0xB7 = Write request with error in the parameter request header.
	0x8 = invalid parameter 0x9 = invalid type	
	0xA…0xF = user-specific (not used)	
0xC = resource	0x0 = read constraint conflict 0x1 = write constraint conflict 0x2 = resource busy 0x3 = resource unavailable 0x4…0x7 = reserved (not used) 0x8…0xF = user-specific (not used)	
0xD…0xF = user-specific (not used)		

8.3.2.3 PROFIDrive parameter access errors

In addition to the error indications in the error response field, details about the error are provided in the parameter value field. The third octet in the parameter value is set to 0x00 and the fourth octet is assigned the error number, as described in Table 44.

Table 44. PROFIDrive parameter access errors

Error number	Meaning	When used
0x00	Impermissible parameter number	Access to unavailable parameter.
0x01	Parameter value cannot be changed	Change request to a read-only parameter.
0x02	0x0 = invalid index	0xB0 = parameter requests are not supported.
0x03	Invalid subindex	Access to an unavailable subindex of an array parameter.
0x04	Non-array parameter	Attempt to access subindex of a non-array parameter.
0x05	Incorrect data type	Change request containing invalid data type for the accessed parameter.
0x06	Setting not permitted (may only be reset)	Change request to a non-zero value, where this is not allowed.
0x07	Description element cannot be changed	Change request to a read-only parameter description element.
0x08	Reserved (not used)	
0x09	No description data available	Access to unavailable parameter description.
0x0A	Reserved (not used)	
0x0B	No operation priority	Change request without access rights to perform the change.
0x0C...0x0E	Reserved (not used)	
0x0F	No text array available	Access to unavailable parameter text array.
0x10	Reserved (not used)	
0x11	Request cannot be executed	Access is temporarily not possible due to unspecified reasons.
0x12...0x13	Reserved (not used)	Change request with a value within the allowed range, but is otherwise not permissible.
0x14	Value impermissible	The length of the respose exceeds the maximum transmittable length.
0x15	Response too long	
0x16	Impermissible parameter address	Error in the parameter address field.
0x17	Illegal format	Illegal format was provided in write request.
0x18	Number of values are not consistent	The number of values in the write request does not match the number of values in the parameter.
0x19	Axis non-existent	Access to non-existent axis number
0x20	Parameter text cannot be changed	Change request to unavailable parameter text.
0x21	Invalid request ID	If a parameter request does not have the request ID 01h or 02h, this error code is returned.
0x22...0x64	Reserved (not used)	
0x65	Invalid request reference	Unallowed value for request reference.
0x66	Invalid request ID	Unallowed value in request ID (neither Request Parameter nor Change Parameter).
0x67	Reserved (not used)	
0x68	Invalid number of parameters	Invalid number of parameters in request (0 or greater than 39).
0x69	Invalid attribute	Invalid attribute specified in request.
0x6A	Reserved (not used)	Not enough parameter value data was transmitted in a Change request. Alternatively, the request did not contain a complete parameter address.
0x6B	Request is too short	
0x6C	Parameter not found.	Parameter was not found or error occurred while accessing it.
0x6D	Invalid change request.	Issue with request was found and request was not handled.
0x6E	General error	General error occurred.
0x6F 0xFF	Reserved (not used)	

8.3.2.4 Response header

The response header consists of 4 fields, each one is octet in size.

Table 45. Response header

Octet number	Field name	Description	Allowed values
1	Request reference	Unique number for each request/response pair.	Mirrored by the slave.
2	Response ID	Defines the type of response. An error in the execution of a request is indicated by setting bit 7 in this field.	Uses 0x01 for successful request parameter operation. Uses 0x02 for successful change parameter operation. Uses 0x80 to indicate that an invalid request ID was received. Uses 0x81 for unsuccessful request parameter operation. Uses 0x82 for unsuccessful change parameter operation. Other values are not used.
3	Axis number	Not used, should be set to 1. Mirrored by the slave.	Mirrored by the slave.
4	Requested number of parameters	The number of parameters affected by the request.	Number of parameters in the response. Mirrored from the request.

8.3.2.5 Parameter values

Parameter values are included in the response only if the request was of "Request parameter" type. For details on the contents of this field, see Parameter value on Chapter 8.3.1.4 "Parameter value"

8.3.3 Drive parameter access using application ID

It is possible to read and write drive parameters using the application ID number using the PNU 10001. The targeted application ID is put into the subindex field.

8.3.4 Parameter channel examples

8.3.4.1 Request first element of PNU964 value

The following information is used for this request:

Table 46. Request first element of PNU964

Field	Contents
Request reference	0x01
Request ID	0x01 = Request parameter
Axis Number	0x01
No. of Parameters	0x01
Attribute	0x10 = Value
No. of elements	0x01
Parameter Number	0x03C4 (964d)
Subindex	0x0000 (0d)

The request from the master is:

Table 47. Request from master

PROFINET Write request header	Request header	Parameter address
...	0x01 0x01 0x01 0x01	0x10 0x01 0x03 0xC4 0x00 0x00

The slave responds to the write request with a PROFINET write response header.

Table 48. Response from slave

PROFINET Write response header
...

The master reads the result of the operation from the drive using a PROFINET read request header.

Table 49. Read request header

PROFINET Read request header
...

The slave responds to the read request:

Table 50. Read request response

PROFINET Read response header	Request header	Parameter value
...	0x01 0x01 0x01 0x01	0x42 0x01 0x01 0xBA

The parameter value reveals that the format of the value is "Word" (0x42), there is one value in the response (0x01) and the actual value is 0x01BA.

8.3.4.2 Request all elements of PNU964 value

The following information is used for this request.

Table 51. Request all elements of PNU 964

Field	Contents
Request reference	0x02
Request ID	0x01 = Request parameter
Axis Number	0x01
No. of Parameters	0x01
Attribute	0x10 = Value
No. of elements	0x06
Parameter Number	0x03C4 (964d)
Subindex	0x0000 (0d)

The request from the master is:

Table 52. Request from master

PROFINET Write request header	Request header	Parameter address
...	0x02 0x01 0x01 0x01	0x10 0x06 0x03 0xC4 0x00 0x00

The slave responds to the write request with a PROFINET write response header.

Table 53. Response from slave

PROFINET Write response header
...

The master reads the result of the operation from the drive using a PROFINET read request header.

Table 54. Read request from master

PROFINET Read request header
...

The slave responds to the read request:

Table 55. Slave response to read request

PROFINET Read response header	Request header	Parameter value
...	0x02 0x01 0x01 0x01	0x42 0x06 0x01 0xBA 0x00 0x02 0x00 0x64 0x07 0xDE 0x00 0x65 0x00 0x01

The returned value consists of six words (0x42 means Word, 0x06 is the number of values returned), and the values are 0x01BA, 0x0002, 0x0064, 0x07DE, 0x0065, and 0x0001. Thus the following information can be determined about the device:

- Manufacturer code is 0x01BA
- Drive Unit type is 0x0002
- Software version is 1.0 (0x0064 = 0100d)
- Firmware date (year) is 2014 (0x07DE)
- Firmware date (day/month) is 1/1 (0x0065 = 101d)
- The device contains one axis

8.3.4.3 Requesting the value of drive parameter ID 103

The following information is used for this request:

Table 56. Request value of parameter ID 103

Field	Contents
Request reference	0x03
Request ID	0x01 = Request parameter
Axis Number	0x01
No. of Parameters	0x01
Attribute	0x10 = Value
No. of elements	0x01
Parameter Number	0x2711 (10001d)
Subindex	0x0067 (103d)

The request from the master is:

Table 57. Request from master

PROFINET Write request header	Request header	Parameter address
...	0x03 0x01 0x01 0x01	0x10 0x01 0x27 0x11 0x00 0x67

The slave acknowledges the request with a PROFINET write response header.

Table 58. Response from slave

PROFINET Write response header
...

The master reads the result of the operation from the drive using a PROFINET read request header.

Table 59. Read request from master

PROFINET Read request header
...

The slave responds to the read request:

Table 60. Slave response to read request

PROFINET Read response header	Request header	Parameter address
...	0x03 0x01 0x01 0x01	0x42 0x01 0x00 0x0A

The parameter value reveals that the format of the value is "Word" (0x42), there is one value in the response (0x01) and the actual value is 0x000A. Because this value was read from the drive application, the Drive Application Manual contains details on how to interpret the value. In this example, the Acceleration time would be one second.

8.4 Profinet IO – Communications

8.4.1 Parameters of the PROFIDrive

The table below lists the basic PROFIDrive parameters.

Table 61. PROFIDrive basic parameters

PNU	Significance	Data type	Explanation
915	Selection switch for DO IO Data in the setpoint telegram	Array[n] Unsigned16	Describes the data in the setpoint telegram. The parameter is an array of signals' numbers that creates the setpoint telegram.
916	Selection switch for DO IO Data in the actual value telegram	Array[n] Unsigned16	Describes the data in the actual value telegram. The parameter is an array of signals' numbers that creates the actual value telegram.
922	Telegram selection	Unsigned16	Currently selected standard telegram is read. It returns for example 1 for ST1. See chapter 8.4.3.2., chapter 8.4.3.3, chapter 8.4.3.4, chapter 8.4.3.5 for possible values.
923	List of all parameters for signals	Array[n] Unsigned16	The parameter is an array. The index of the array indicates for a signal number and its value for corresponding parameter number. Not supported standard signals, those in range 1-99, have values set to 0. Gaps between the device-specific signals are also filled with 0. Refer to Table 63.
930	Operating mode	Unsigned16	1 = Speed control mode
944	Fault message counter	Unsigned16	The fault message counter is incremented each time that the fault buffer changes. This means that it is guaran-teed that the fault buffer is consistently read-out. Without this parameter, it is not guaranteed that the fault buffer does not change while reading-out.
947	Fault number	Array[n] Unsigned16	The parameter is an array of 8 elements. The first element indicates an active unacknowledged fault. The following elements contain acknowledged ones. The latest acknowledged fault number is at index 1 and oldest one at index 7.
964	Drive Unit Identification	Array[n] Unsigned16	An array is structured in the following way: Index meaning: 0 = Manufacturer code (0x01BA) 1 = Drive Unit Type (0x0002) 2 = Software version – XXYYd (XX – major revision, YY – minor revision) 3 = Firmware date (year) – YYYYd 4 = Firmware date (day/month) – DDMMd 5 = Number of Drive Objects (0x0001)
965	Profile identification number	OctetString2	Two bytes to identify the profile that is used. 1st – profile number; PROFIDrive (3d) 2nd – profile version number; 4.1 (41d)
975	DO identification	Array[n] Unsigned16	An array is structured in the following way: Index meaning: 0 = Manufacturer code (0x01BA) 1 = Drive Unit Type (0x0003) 2 = Software version – XXYYd (XX – major revision, YY – minor revision) 3 = Firmware date (year) – YYYYd 4 = Firmware date (day/month) – DDMMd 5 = Drive Object Type Class – Axis (0x0001) 6 = Drive Object Sub-class 1 – Only Application Class 1 (0x0001) 7 = Drive Object ID (value 1)
980 – 989	Number list of defined parameter	Array[n] Unsigned16	980: This is a list of the parameter numbers of all the implemented parameters. The list does not contain the number 980- 989. Parameters are listed in the ascending (growing) order. The end-of-list is indicated by the value 0. 981-989: Not used. Length of each is 1 and value is 0, indicating an empty list.

8.4.1.1 PROFIDrive parameters for PROFINET IO communication interface

The table below lists the PROFINET IO communication interface parameters.

Table 62. PROFIDrive parameters

PNU	Significance	Data type	Explanation
61000	Name of station	Octect-String[240] with-out null termination	Name of station for the PROFINET IO network interface, which is related to this drive unit.
61001	IP of station	Unsigned32	IP Address of the station for the PROFI-NET IO network interface.
61002	Mac of station	OctetString [6]	MAC Address of the station for the PRO-FINET IO network interface
61003	Default gateway of station	Unsigned32	Default gateway for the station for the PROFINET IO network interface.
61004	SubnetMaskOfStation	Unsigned32	Subnet mask of the station for the PRO-FINET IO network interface.

8.4.1.2 PROFIDrive signal numbers in OPTE9

The table below lists the PROFIDrive signal numbers.

Table 63. PROFIDrive signal numbers

Signal no.	Signal name	PNU	PNU name
1	Control word 1	10100	PROFIDrive control word (STW1)
2	Status word 1	10102	PROFIDrive status word (ZSW1)
5	Speed setpoint A	10101	PROFIDrive speed setpoint value (NSOLL_A)
6	Speed actual value A	10103	PROFIDrive speed actual value (NIST_A)
51	Output current	10104	Always returns zero.
52	Active current (torque proportional)	10105	Always returns zero.
54	Active power	10106	Always returns zero.
57	Speed actual value A	10107	Always returns zero.
58	Drive status/fault word	10108	Always returns zero.

* 32 bit data not supported in current version. See chapter 5.

8.4.2 Vendor-specific PROFIDrive parameters

The table below lists vendor-specific PROFIDrive parameters.

Table 64. PROFIDrive drive-specific parameters

PNU	Significance	Data type	Explanation
9900	Test parameter (non-array)	Unsigned16	For testing purposes. Does not affect the operation of the drive.
9901	Test parameter (array)	Array[n] Unsigned16	An array of 16 elements. Used only for testing purposes. Does not affect the operation of the drive.
10001	Drive parameter access	Array[n] Special case, data type depends from the sub index	A parameter used to access parameters from the drive application. You can do this by putting the desired drive parameter ID into the sub index field of the parameter request. See chapter 8.3.4
10100	Profile control word (STW1)	Unsigned16	PROFIDrive 4.1 control word (STW1).
10101	Profile speed setpoint value (NSOLL_A)	Integer16	PROFIDrive 4.1 speed setpoint value (NSOLL_A).
10102	Profile status word (ZSW1)	Unsigned16	PROFIDrive 4.1 status word (ZSW1).
10103	Profile speed actual value (NIST_A)	Integer16	PROFIDrive 4.1 speed actual value (NIST_A).
10109	16-bit Process Data In	Array[n] Unsigned16	An array of 8 elements. From PDI1 (index 0) to PDI8 (index 7).
10110	16-bit Process Data Out	Array[n] Unsigned16	An array of 8 elements. From PDO1 (index 0) to PDO8 (index 7).
10111	Speed physical reference parameter	Unsigned16	The parameter describes how many RPM is meant by 100% in the PROFIDrive 4.1 speed setpoint and actual value fields.
10112	Fixed Control Word	Unsigned16	Fixed control word.
10113	Fixed Status Word	Unsigned16	Fixed status word.
10114	Speed reference	Unsigned16	Speed reference.
10115	Speed Actual value	Unsigned16	Actual speed value.
10118	Clear fault history	Unsigned16	To clear the fault history, add a value to the parameter.
10119	Read fault history	Array[n] Unsigned16	An array of 5 elements.
10120	General Control word	Unsigned16	General control word.
10121	General Status word	Unsigned16	General status word.
10122	32-bit Process Data In	Array[n] Unsigned32	An array of 8 elements. From PDI1 (index 0) to PDI8 (index 7).
10123	32-bit Process Data Out	Array[n] Unsigned32	An array of 8 elements. From PDO1 (index 0) to PDO8 (index 7).
10124	Drive operation time counter	Unsigned32	Drive operation time in seconds as 32 bit unsigned integer.
10125	Drive operation time trip counter	Unsigned32	Drive operation time trip counter in seconds as 32 bit unsigned integer. Writing zero will reset trip counter.
10126	Drive energy counter	Float32	Drive energy counter in KWh as 32 bit float (IEEE 754).
10127	Drive energy trip counter	Float32	Drive energy trip counter in KWh as 32 bit float (IEEE 754). Writing zero will reset trip counter.

8.4.3 Telegrams implemented by OPTE9

8.4.3.1 Telegram building blocks

8.4.3.1.1. STW1 – Control word 1 31

The table below lists the assignments of bits in the control word 1.

Table 65. Overview of the assignments of bits of the control word 1

Bit	Significance speed control mode bit value is 1	Bit value is 0
0	ON	OFF
1	No Coast Stop (no OFF2)	Coast Stop (OFF2)
2	No Quick Stop (no OFF3)	Quick Stop (OFF3)
3	Enable Operation	Disable Operation
4	Enable Ramp Generator	Reset Ramp Generator
5	Unfreeze Ramp Generator	Freeze Ramp Generator
6	Enable Setpoint	Disable Set Point
7	Fault Acknowledge (0 -> 1)	
8	Not used	
9	Not used	
10*	Control by PLC	No control by PLC
11	Device-specific	Device-specific
12-15	Device-specific	Device-specific

* Bits in a control word do not have any effect unless bit 10 is enabled.

Bit 0: Switching ON/OFF

This bit is used in combination with other bits to enable operation of the drive. When this bit is set to 0 during operation, the drive performs a ramp stop.

Bit 1: Coast stop command

This bit is used to request a coast stop to be executed. When it is set to 0 during operation, the drive performs a coast stop.

Bit 2: Quick stop command

This bit is used to request a quick stop to be executed. When it is set to 0 during operation, the drive quickly ramps down to zero speed and stops.

Bit 3: Enabling of operation

This bit is used in combination with other bits to enable operation of the drive. When it is set to 0 during operation, the drive performs a coast stop.

Bit 4: Enabling of ramp generator

This bit is used in combination with other bits to enable operation of the drive. When it is set to 0 during operation, the drive quickly decelerates to zero speed.

Bit 5: Freezing of ramp generator

This bit can be used to freeze the setpoint value used by the drive. The value is frozen if this bit is set to 0.

If the bit is 1, the setpoint value provided by the master is continuously updated.

Bit 6: Enabling of setpoint value

This bit can be used to disable the fieldbus setpoint value. If this bit is set to 0, the option board ignores the setpoint value by the master and instead uses a setpoint value of 0. During operation, if this bit is set to 0, the drive decelerates to a standstill.

Bit 7: Fault acknowledge

This bit is used to acknowledge faults in the drive. When a rising edge (0 -> 1) is seen in this bit by the option board, it requests the drive to acknowledge present faults. The functionality of this bit is rising-edge sensitive only.

Bit 10: Control by PLC

This bit is used by the master to indicate that it is in control of the slave and that the commands sent via fieldbus are valid.

During operation, this bit must be 1. If the drive is not operating and this bit is 0, the drive cannot be started.

If the drive is operating, and this bit becomes 0, the option board freezes the process data provided to the drive, and sets its state to FAULT. The drive reaction to this fieldbus fault depends on the drive parameterization.

8.4.3.1.2. ZSW1 – Status word 1 32

The table below lists the assignments of the status word 1.

Table 66. Overview of the assignments of bits of the status word 1

Bit	Significance speed control mode bit value is 1	Bit value is 0
0	Ready to switch on	Not ready to switch on
1	Ready to operate	Not ready to operate
2	Operation enabled (drive follows setpoint)	Operation disabled
3	Fault present	No fault
4	Coast stop not activated (No OFF2)	Coast stop activated (OFF2)
5	Quick stop not activated (No OFF3)	Quick stop activated (OFF3)
6	Switching on inhibited	Switching on not inhibited
7	Warning present	No warning
8	Speed error within tolerance Range	Speed error out of tolerance range
9	Control by plc requested	No control by PLC requested
10	F or n reached or exceeded	F or n not reached
11	Device-specific	Device-specific
12	Drive running	Drive stopped
13	Drive is ready	Drive is not ready
14-15	Device-specific	Device-specific

Bit 0: Readiness to switch on

This bit indicates whether the drive is ready to switch on the power electronics. When the bit has the value 0, the drive is not ready to switch on the power electronics. When the bit has the value 1, the drive is ready to switch on the power electronics.

Bit 1: Readiness to operate

This bit indicates whether the drive is ready to begin operation. When the bit has the value 0, the power electronics is switched off and the drive is unable to begin operation. When the bit has the value 1, the power electronics is switched on and the drive can begin operation if requested by the master.

Bit 2: State of operation

This bit indicates whether the drive is operating or not. When the bit has the value 0, the drive is not operating. When the bit has the value 1, the drive is operating.

Bit 3: Presence of fault

This bit indicates the presence of unacknowledged faults in the drive. When the bit has the value 0, no unacknowledged faults are present in the drive. When the bit has the value 1, at least one unacknowledged fault is present in the drive.

Bit 4: Coast stop activated

This bit indicates whether a coast stop command is active or not. When the bit has the value 0, a coast stop command is active. When the bit has the value 1, no coast stop command is active.

Bit 5: Quick stop activated

This bit indicates whether a quick stop command is active or not. When the bit has the value 0, a quick stop command is active. When the bit has the value 1, no quick stop command is active.

Bit 6: Switching on inhibition

This bit indicates whether the power electronics may be switched on or not. When the bit has the value 0, the power electronics may be switched on. When the bit has the value 1, the power electronics are prevented from switching on.

Bit 7: Presence of warning

This bit indicates the presence of warning/alarm information in the drive. When the bit has the value 0, no warning is present. When the bit has the value 1, a warning is present.

Bit 8: Running at setpoint

This bit indicates whether the drive is operating and the actual speed value matches the setpoint value. When the bit has the value 0, the actual speed value does not match the setpoint value. When the bit has the value 1, the actual speed value matches the setpoint value.

Bit 9: Request control by master

This bit indicates whether the field bus master should take control of the drive. When this bit has the value 0, the master does not take control of the drive. When this bit has the value 1, the master is requested to take control of the drive.

In OPTE9, this bit depends on the configuration for the drive control place. If the control place is assigned to fieldbus, the bit has the value 1. If the control place is elsewhere, the bit has the value 0.

Bit 10: Setpoint reached or exceeded

This bit indicates whether the setpoint value has been reached or exceeded. When this bit has the value 0, the setpoint value has not been reached or exceeded. When this bit has the value 1, the setpoint value has been reached or exceeded.

Bit 12: Drive running

This bit indicates drive state. If bit is 1, the motor is running. If bit is zero, the motor has been stopped.

Bit 13: Drive ready

This bit indicates drive state. If bit is 1, the drive is ready for transition to running state.

8.4.3.1.3. PROFIDrive speed setpoint value NSOLL_A

Normalised 16-bit speed setpoint (containing a sign bit and a 15-bit integer).

- NSOLL_A = 0x4000 corresponds to 100% of the parameterized maximum motor speed
- NSOLL_A = 0xC000 corresponds to -100% of the parameterized maximum motor speed

8.4.3.1.4. PROFIDrive speed actual value NIST_A

Normalised 16-bit actual speed.

- NIST_A = 0x4000 corresponds to 100% of the parameterized maximum motor speed
- NIST_A = 0xC000 corresponds to -100% of the parameterized maximum motor speed

8.4.3.1.5. Vendor-specific fbfixedcontrolword

For details about vendor control word, see Chapter 12 "APPENDIX 2 – CONTROL AND STATUS WORD"

8.4.3.1.6. Vendor-specific fbspeedreference

The FBSpeedReference value is unsigned in the range 0...10000d (0...2710h). The value 0 corresponds to MinimumFrequency and the value 10000d corresponds to MaximumFrequency. Requested direction is indicated using bit 1 in the FBFixedControlWord.

8.4.3.1.7. Vendor-specific fbfixedstatusword

For details about vendor status word, see Chapter 12 "APPENDIX 2 – CONTROL AND STATUS WORD".

8.4.3.1.8. Vendor-specific fbspeedactual

The FBActualSpeed value is unsigned in the range 0...10000d (0...2710h). The value 0 corresponds to MinimumFrequency and the value 10000d corresponds to MaximumFrequency. The direction is indicated using bit 2 in the FBFixedStatusWord.

8.4.3.1.9. Vendor-specific fbgeneralcontrolword

FB General Control Word is 16-bit in length and it is completely application-dependent.

8.4.3.1.10. Vendor-specific fbgeneralstatusword

FB General Status Word is 16-bit in length and it is completely application-dependent.

8.4.3.1.11. Vendor-specific process data

The Process Data variables are vendor-specific variables that can be communicated to and from the drive. There can be up to eight Process Data variables communicated in a single telegram. Values sent from the option board to the master are called ProcessDataOut variables, while the values sent from the master to the option board are called ProcessDataIn variables. The contents of the ProcessDataOut variables can be parameterised in the drive using a feature known as Fieldbus Process Data mapping. See the drive's Application Manual for further details.

8.4.3.2 PROFIDrive standard telegram 1 and vendor-specific variants

These telegrams use the PROFIDrive-defined control word, status word, speed setpoint value and speed actual value. It is also possible to use up to eight Process Data fields. Using these telegrams, the process data fields are communicated as 16-bit values.

Table 67. PROFIDrive standard telegram

Telegram no.	Telegram
1	Standard telegram 1
102	Standard telegram 1 + PD1
103	Standard telegram 1 + PD[1..2]
104	Standard telegram 1 + PD[1..3]
100	Standard telegram 1 + PD[1..4]
105	Standard telegram 1 + PD[1..5]
106	Standard telegram 1 + PD[1..6]
107	Standard telegram 1 + PD[1..7]
101	Standard telegram 1 + PD[1..8]

8.4.3.2.1. Setpoint data

Standard telegram 1 – setpoint data

Table 68. PROFIDrive standard telegram 1 – setpoint data

Standard telegram 1		Additional process Data (16-bit each)							
STW1 (16-bit	NSOLL_A (16-bit)	PDI1	PDI2	PDI3	PDI4	PDI5	PDI6	PDI7	PDI8

STW1 (16-bit) (See 8.4.3.1.1. STW1 – Control Word 1 31)

NSOLL_A (16-bit) (See 8.4.3.1.3. PROFIDrive speed setpoint value NSOLL_A)

PDI1-PDI8 (See 8.4.3.1.11. Vendor-specific Process Data)

8.4.3.2.2. Actual data

Standard telegram 1 – actual data

Table 69. PROFIDrive standard telegram 1 – actual data

Standard telegram 1		Additional process Data (16-bit each)							
ZSW1 (16-bit)	NIST_A (16-bit)	PDO1	PDO2	PDO3	PDO4	PDO5	PDO6	PDO7	PDO8

ZSW1 (16-bit) (See8.4.3.1.2. ZSW1 – Status Word 1 32)

NIST_A (16-bit) (See 8.4.3.1.4. PROFIDrive speed actual value NIST_A)

PDO1-PDO8 (See 8.4.3.1.11. Vendor-specific Process Data)

8.4.3.3 Vendor-specific telegram 1 and its variant

These telegrams use the vendor-specific control word, status word, speed setpoint value and speed actual value. It is also possible to use up to eight vendor-specific Process Data fields. Using these telegrams, the process data fields are communicated as 16-bit values.

Table 70. Vendor-specific telegram 1

Telegram no.	Telegram
108	Vendor-specific Telegram 1
109	Vendor-specific Telegram 1 + PD1
110	Vendor-specific Telegram 1 + PD[1..2]
111	Vendor-specific Telegram 1 + PD[1..3]
112	Vendor-specific Telegram 1 + PD[1..4]
113	Vendor-specific Telegram 1 + PD[1..5]
114	Vendor-specific Telegram 1 + PD[1..6]
115	Vendor-specific Telegram 1 + PD[1..7]
116	Vendor-specific Telegram 1 + PD[1..8]

8.4.3.3.1. Setpoint data

Vendor specific telegram 1 – setpoint data

Table 71. Vendor-specific telegram 1 – setpoint data

Vendor-specific telegram 1		Additional process data (16-bit each)							
FB Fixed CW (16-bit)	FB Speed Reference 16-bit)	PDI1	PDI2	PDI3	PDI4	PDI5	PDI6	PDI7	PDI8

FB Fixed CW (16-bit) (See 8.4.3.1.5. Vendor-specific FBFixedControlWord)

FB Speed Reference (16-bit) (See 8.4.3.1.6. Vendor-specific FBSpeedReference) PDI1-PDI8 (See 8.4.3.1.11. Vendor-specific Process Data)

8.4.3.3.2. Actual data

Vendor specific telegram 1 – actual data

Table 72. Vendor-specific telegram 1 – actual data

Vendor-specific telegram 1		Additional process data (16-bit each)							
FB Fixed SW (16-bit)	FB Speed Actual (16-bit)	PDO1	PDO2	PDO3	PDO4	PDO5	PDO6	PDO7	PDO8

FB Fixed SW (16-bit) (See 8.4.3.1.7. Vendor-specific FBFixedStatusWord) FB Speed Actual (16-bit) (See 8.4.3.1.8. Vendor-specific FBSpeedActual) PDO1-PDO8 (See 8.4.3.1.11. Vendor-specific Process Data)

8.4.3.4 Vendor-specific telegram 2 and its variants

These telegrams use the vendor-specific fixed and general control word, status word, speed setpoint value and speed actual value. It is also possible to use up to eight vendor-specific Process Data fields. Using these telegrams, the process data fields are communicated as 32-bit values.

Table 73. Vendor-specific telegram 2

Telegram no.	Telegram
117	Vendor-specific Telegram 2
118	Vendor-specific Telegram 2 + PD1
119	Vendor-specific Telegram 2 + PD[1..2]
120	Vendor-specific Telegram 2 + PD[1..3]
121	Vendor-specific Telegram 2 + PD[1..4]
122	Vendor-specific Telegram 2 + PD[1..5]
123	Vendor-specific Telegram 2 + PD[1..6]
124	Vendor-specific Telegram 2 + PD[1..7]
125	Vendor-specific Telegram 2 + PD[1..8]

8.4.3.4.1. Setpoint data

Vendor-specific telegram 2 – setpoint data

Table 74. Vendor-specific telegram 2 – setpoint data

Vendor-specific telegram 2			Additional process data (32-bit each)							
FB Fixed CW (16-bit)	FB Gen CW (16-bit)	FB Speed Reference (16-bit)	PDI1	PDI2	PDI3	PDI4	PDI5	PDI6	PDI7	PDI8

Note: 32 bit data not supported in current version. See chapter 5.

FB Fixed CW (16-bit) (See 8.4.3.1.5. Vendor-specific FBFixedControlWord)

FB Gen CW (16-bit) (See 8.4.3.1.9. Vendor-specific FBGeneralControlWord)

FB Speed Reference (16-bit) (See 8.4.3.1.6. Vendor-specific FBSpeedReference)

PDI1-PDI8 (See 8.4.3.1.11. Vendor-specific Process Data)

8.4.3.4.2. Actual data

Vendor-specific telegram 2 – actual data

Table 75. Vendor-specific telegram 2 – actual data

Vendor-specific telegram 2			Additional process data (32-bit each)							
FB Fixed SW (16-bit)	FB Gen SW (16-bit)	FB Speed Actual (16-bit)	PDO1	PDO2	PDO3	PDO4	PDO5	PDO6	PDO7	PDO8

Note: 32 bit data not supported in current version. See chapter 5.

FB Fixed SW (16-bit) (See 8.4.3.1.7. Vendor-specific FBFixedStatusWord)

FB Gen SW (16-bit) (See 8.4.3.1.10. Vendor-specific FBGeneralStatusWord)

FB Speed Actual (16-bit) (See 8.4.3.1.8. Vendor-specific FBSpeedActual)

PDO1-PDO8 (See 8.4.3.1.11. Vendor-specific Process Data)

8.4.3.5 Vendor-specific telegram 3 and its variants

These telegrams use a mix of PROFIDrive and vendor-specific data. It is possible to use also up to eight vendor-specific Process Data fields. Using these telegrams, the process data fields are communicated as 32-bit values.

Table 76. Vendor-specific telegram 3

Telegram no.	Telegram
126	Vendor-specific telegram 3
127	Vendor-specific telegram 3 + PD1
128	Vendor-specific telegram 3 + PD[1..2]
129	Vendor-specific telegram 3 + PD[1..3]
130	Vendor-specific telegram 3 + PD[1..4]
131	Vendor-specific telegram 3 + PD[1..5]
132	Vendor-specific telegram 3 + PD[1..6]
133	Vendor-specific telegram 3 + PD[1..7]
134	Vendor-specific telegram 3 + PD[1..8]

8.4.3.5.1. Setpoint data

Vendor-specific telegram 3 – setpoint data

Table 77. Vendor-specific telegram 3 – setpoint data

Vendor-specific telegram 3			Additional process data (32-bit each)							
STW1 (16-bit)	FB Gen CW (16-bit)	NSOLL_A (16-bit)	PDI1	PDI2	PDI3	PDI4	PDI5	PDI6	PDI7	PDI8

Note: 32 bit data not supported in current version. See chapter 5.

STW1 (16-bit) (See 8.4.3.1.1. STW1 – Control Word 1 31)

FB Gen CW (16-bit) (See 8.4.3.1.9. Vendor-specific FBGeneralControlWord)

NSOLL_A (16-bit) (See 8.4.3.1.3. PROFIDrive speed setpoint value NSOLL_A)

PDI1-PDI8 (See 8.4.3.1.11. Vendor-specific Process Data)

8.4.3.5.2. Actual data

Vendor-specific telegram 3 – actual data

Table 78. Vendor-specific telegram 3 – actual data

Vendor-specific telegram 3			Additional process data (32-bit each)							
ZSW1 (16-bit)	FB Gen SW (16-bit)	NIST_A (16-bit)	PDO1	PDO2	PDO3	PDO4	PDO5	PDO6	PDO7	PDO8

Note: 32 bit data not supported in current version. See chapter 5.

ZSW1 (16-bit) (See 8.4.3.1.2. ZSW1 – Status Word 1 32)

FB Gen SW (16-bit) (See 8.4.3.1.10. Vendor-specific FBGeneralStatusWord)

NIST_A (16-bit) (See 8.4.3.1.11. Vendor-specific Process Data

PDO1-PDO8 (See 8.4.3.1.11. Vendor-specific Process Data)

8.4.3.6 Vendor-specific telegram 4 and its variants

These telegrams use the vendor-specific fixed control word, status word, speed setpoint value and speed actual value. It is also possible to use up to eight vendor-specific Process Data fields. Using these telegrams, the process data fields are communicated as 16-bit values.

Table 79. Vendor-specific telegram 4

Telegram no.	Telegram
135	Vendor-specific telegram 4
136	Vendor-specific telegram 4 + PD[1..4]
137	Vendor-specific telegram 4 + PD[1..8]

8.4.3.6.1. Setpoint data

Vendor-specific telegram 4 – setpoint data

Table 80. Vendor-specific telegram 4 – setpoint data

Vendor-specific telegram 4		Additional process data (16-bit each)							
FB Fixed CW (16-bit)	Speed Reference (16-bit)	PDO1	PDO2	PDO3	PDO4	PDO5	PDO6	PDO7	PDO8

FB Fixed CW (16-bit) (See Chapter "8.4.3.1.5. Vendor-specific FBFixedControlWord")

FB Speed Reference (16-bit) (See Chapter "8.4.3.1.6. Vendor-specific FBSpeedReference")

PDI1-PDI8 (See Chapter "8.4.3.1.11. Vendor-specific Process Data")

8.4.3.6.2. Actual data

Vendor-specific telegram 4 – actual data

Table 81. Vendor-specific telegram 4 – actual data

Vendor-specific telegram 4		Additional process data (16-bit each)							
FB Gen SW (16-bit)	FB Speed Actual (16-bit)	PDO1	PDO2	PDO3	PDO4	PDO5	PDO6	PDO7	PDO8

FB Gen SW (16-bit) (See Chapter "8.4.3.1.10. Vendor-specific FBGeneralStatusWord")

FB Speed Actual (16-bit) (See Chapter "8.4.3.1.8. Vendor-specific FBSpeedActual")

PDO1-PDO8 (See Chapter "8.4.3.1.11. Vendor-specific Process Data")

8.4.4 Quick setup

By following these instructions, you can easily and fast set up your Profinet IO for use:

In the AC drive application: Choose Fieldbus as the active control place (see the drive's User's Manual).

In the master software:

1. Set the Control Word value to 0hex
2. Set the Control Word value to 47Ehex
3. Set the Control Word value to 47Fhex
4. AC drive status is RUN
5. Set the Reference value to '2000Hex' (=50.00%)
6. Actual speed is 2000Hex (25.00 Hz if MinFreq is 0.00 Hz and MaxFreq is 50.00 Hz)
7. Set the Control Word value to 47Ehex
8. AC drive status is STOP

9. Ethernet/IP

9.1 General information

The EtherNet/IP™ is an industrial Ethernet network solution available for manufacturing automation. The CIP™ (Common Industrial Protocol) encompasses a comprehensive suite of messages and services for a variety of manufacturing automation applications, including control, safety, synchronization, motion, configuration and information. The CIP provides users with a unified communication architecture throughout the manufacturing enterprise.

More information on the EtherNet/IP can be found at www.odva.org.

9.1.1 Overview

The EtherNet/IP (Ethernet/Industrial Protocol) is a communication system suitable for use in industrial environments. The EtherNet/IP allows industrial devices to exchange time-critical application information. These devices include simple I/O devices such as sensors/actuators, as well as complex control devices such as robots, programmable logic c ontrollers, welders, and process controllers.

The EtherNet/IP uses CIP (Control and Information Protocol), the common network, transport and application layers also shared by ControlNet and EtherNet/IP. The EtherNet/IP then makes use of standard Ethernet and TCP/IP technology to transport CIP communications packets. The result is a common, open application layer on top of open and highly popular Ethernet and TCP/IP protocols.

The EtherNet/IP Messaging Forms:

- Unconnected Messaging is used for connection establishment and for infrequent, low priority messages
- Connected Messaging utilizes resources which are dedicated in advance to a particular purpose such as real-time I/O data transfer. EtherNet/IP Messaging Connections
- Explicit Messaging Connections are general purpose point-to-point connections. Messages are sent through the TCP protocol
- Implicit (I/O Data) Connections are established to move application specific I/O Data at regular intervals. They are often set up as one-to-many relationships in order to take full advantage of the producer-consumer multicast model. Implicit messages are sent through the UDP protocol

9.1.2 AC/DC drive profile

OPTE9 implements the CIP AC/DC drive profile.

In order to provide interoperability between devices from different manufacturers, there must be a defined "standard" in which those devices:

- exhibit the same behaviour
- produce and/or consume the same basic set of I/O data
- contain the same basic set of configurable attributes. The formal definition of this information is known as a device profile

9.1.3 EDS file

You can provide configuration support for your device by using a specially formatted ASCII file, referred to as the EDS (Electronic Data Sheet). An EDS provides information about the device configuration.

The information in an EDS allows configuration tools to provide informative screens that guide a user through the steps necessary to configure a device. An EDS provides all of the information necessary to access and alter the configurable parameters of a device. This information matches the information provided by instances of the Parameter Object Class. The CIP Object Library describes the Parameter Object Class in detail.

OPTE9 has multiple EDS files, at least one per drive type. You can download the drive specific EDS from Eaton web site (www.eaton.com/drives). See also Chapter 6.1.9 "EIP Product code offset".

9.1.4 LED functionality

The OPTE9 LEDs follow the CIP standard when the EtherNet/IP is set as the active protocol. The LEDs on the option board are not labeled as the CIP defines. The LED labels are described in table below.

Table 82. OPTE9 etherNet/IP LED definitions

LED label	Description	CIP definition
RN	Network Status Indicator	NS
ER	Not used	-
BS	Module Status indicator	MS

9.1.4.1 Module status LED

The Module status LED is labeled on the board as "BS". It shows the status of the module, that is, if a fault has occurred or if the module has been configured. The MS LED functionality is described in the table below.

Table 83. EtherNet/IP module status LED functionality

LEDs	State	Summary	Requirement
MS	Steady off	No power	If no power is supplied to the device, the module status indicator is steady off.
MS	Steady green	Device operational	If the device is operating correctly, the module status indicator is steady green.
MS	Flashing green	Standby	If the device has not been configured, the module status indicator is flashing green.
MS	Flashing red	Minor fault	If the device has detected a recoverable minor fault, the module status indicator is flashing red.
MS	Steady red	Major fault	If the device has detected a non-recoverable major fault, the module status indicator is steady red.
MS	Flashing green /red	Self-test	While the device is performing its power up testing.

9.1.4.2 Network status LED

The Network status LED is labeled on the board as "RN". It shows the connectivity status of the device, that is, if there is a connection to the device, or the IP settings status. The NS LED functionality is described in the table below.

Table 84. EtherNet/IP network status LED functionality

LEDs	State	Summary	Requirement
	Steady off	Not powered, no IP address	The device is powered off, or is powered on but with no IP address configured (Interface Configuration attribute of the TCP/IP Inter-face Object).
	Flashing green	No connections	An IP address is configured, but no CIP con-nections are established, and an Exclusive Owner connection has not timed out.
	Steady green	Connected	At least one CIP connection (any transport class) is established, and an Exclusive Owner connection) has not timed out.
	Flashing red	Connection timeout	An Exclusive Owner connection for which this device is the target has timed out. The LED returns to steady green only when all timed out Exclusive Owner connections are re-established. Timeout of connections other than Exclusive Owner connections will not cause the indicator to flash red.
	Steady red	Duplicate IP	The device has detected that its IP address is already in use by another device in the network.
	Flashing green / Red	Self-test	While the device is performing its power up testing

9.1.5 Explicit messaging

Explicit Messaging is used in commissioning and parameterizing of the EtherNet/IP board.

The explicit messages provide multipurpose, point-to-point communication paths between two devices. They provide the typical request/response-oriented network communication used to perform node configuration and problem diagnosis. The explicit messages typically use low priority identifiers and contain the specific meaning of the message right in the data field. This includes the service to be performed and the specific object attribute address.

Note: If Class 1 connection (cyclic data) has been established, the Explicit Messages should not be used to control the Output Data. However this restriction does not apply for the IO Data reading.

9.1.5.1 List of data types

The elementary data types in CIP are (among others):

Table 85. EtherNet/IP datatypes

Name	Description	Bit size	Range	
			Minimum	Maximum
BOOL	Boolean	8	0 = FALSE	1 = TRUE
SINT	Short integer	8	-128	127
INT	Integer	16	-32768	32767
DINT	Double integer	32	-2^{31}	$2^{31} - 1$
LINT	Long integer	64	-2^{63}	$2^{63} - 1$
USINT	Unsigned short integer	8	0	255
UINT	Unsigned integer	16	0	65535
UDINT	Unsigned double integer	32	0	$2^{32} - 1$
ULINT	Unsigned long integer	64	0	$2^{64} - 1$
REAL	Floating point	32	See IEEE 754	
LREAL	Long floating point	64	See IEEE 754	
STRING*	Character string (1 octet per char.)	N		
SHORT_ STRING*	Character string (1 octet per char., 1 Octet length indicator)	N+ 1		
BYTE	Bit string (8 bits)	8		
WORD	Bit string (16 bits)	16		
DWORD	Bit string (32 bits)	32		
LWORD	Bit string (64 bits)	64		

* ISO/IEC-8859-1 encoding.

9.1.5.2 General CIP error codes

The table below contains the error codes used by the OPTE9 EtherNet/IP implementation.

Table 86. General CIP error codes

Code	Status name	Description
0	Success	Service was successfully performed by the object specified.
1	Connection failure	A connection related service failed along the connection path.
2	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
3	Invalid parameter value	See Status Code 0x20, which is the preferred value to use for this condition.
4	Path segment error	The path segment identifier or the segment syntax was not understood by the processing node.
5	Path destination unknown	The path is referencing an object class, instance or struc-ture element that is not known or is not contained in the processing node.

Table 86. General CIP error codes (continued)

Code	Status name	Description
6	Partial transfer	Only part of the expected data was transferred.
8	Service not supported	The requested service was not implemented or was not defined for this Object Class/Instance.
9	Invalid attribute value	Invalid attribute data detected.
$12_d/0C_h$	Object state conflict	The object cannot perform the requested service in its current mode/state.
$14_d/0E_h$	Attribute not settable	A request to modify a non-modifiable attribute was received.
$15_d/0F_h$	Privilege violation	A permission/privilege check failed.
$16_d/10_h$	Device state conflict	The device's current mode/state prohibits the execution of the requested service.
$17_d/11_h$	Reply data too large	The data to be transmitted in the response buffer is larger than the allocated response buffer.
$19_d/13_h$	Not enough data	The service did not supply enough data to perform the specified operation.
$20_d/14_h$	Attribute not supported	The attribute specified in the request is not supported.
$21_d/15_h$	Too much data	The service supplied more data than was expected.
$30_d/1E_h$	Embedded service error	An embedded service resulted in an error.
$31_d/1F_h$	Vendor specific error	A vendor specific error has been encountered. The Additional Code Field of the Error Response defines the particular error encountered.
$32_d/20_h$	Invalid parameter	A parameter associated with the request was invalid.
$38_d/26_h$	Path Size Invalid	The size of the path which was sent with the Service Request is either not large enough to allow the Request to be routed to an object or too much routing data was included.
29_h	Member not settable	A request to modify a non-modifiable member was received.
$46_d/2E_h$	Service Not Supported for Specified Path	The object supports the service, but not for the designated application path (e.g. attribute).

9.1.5.3 Connection manager object error codes

These are the extended status codes used when the general status code is 1.

Table 87. Connection manager object error codes

Extended Status	Descriptions
$256_d/100_h$	Connection in use or duplicate forward open
$259_d/103_h$	Transport class and trigger combination not supported
$262_d/106_h$	Ownership conflict
$263_d/107_h$	Target connection not found
$264_d/108_h$	Invalid network connection parameter
$265_d/109_h$	Invalid connection size
$272_d/110_h$	Target for connection not configured
$273_d/111_h$	RPI not supported
$274_d/112_h$	RPI value(s) not acceptable
$275_d/113_h$	Out of connections

Table 87. Connection manager object error codes (continued)

Extended Status	Descriptions
$276_d/114_h$	Vendor id or product code mismatch
$277_d/115_h$	Product type mismatch
$278_d/116_h$	Revision mismatch
$279_d/117_h$	Invalid produced or consumed application path
$280_d/118_h$	Invalid or inconsistent configuration application path
$281_d/119_h$	Non-listen only connection not opened
$283_d/11B_h$	RPI is smaller than the production inhibit time
$294_d/126_h$	Invalid configuration size
$295_d/127_h$	Invalid originator to target size
$296_d/128_h$	Invalid target to originator size
$297_d/129_h$	Invalid configuration application path
$298_d/12A_h$	Invalid consuming application path
$299_d/12B_h$	Invalid producing application path
$306_d/132_h$	Null forward open function not supported
$517_d/205_h$	Parameter error in unconnected request service
$789_d/315_h$	Invalid segment in connection path
Range 320h – 7FFh are vendor specific	
$800_d/320_h$	Internal: Connection disabled
$64258_d/FB02_h$	Internal: Bad socket
$64259_d/FB03_h$	Internal: Bad originator to target net parameter
$64260_d/FB04_h$	Internal: Bad target to originator net parameter
$64261_d/FB05_h$	Internal: Bad UDP port
$64262_d/FB06_h$	Internal: Join multicast
$64263_d/FB07_h$	Internal: Prepare IO packet
$64267_d/FB0B_h$	Internal: Consumption
$64268_d/FB0C_h$	Internal: FW close
$64270_d/FB0E_h$	Internal: Adapter stopped

9.1.5.4 Supported CIP and vendor objects

The Communication Interface supports the following object classes.

Table 88. CIP objects

Type	Class	Object	Details
Required by ether-net/ip	1	Identity object	See chapter 9.2.1.1
	2	Message router object	See chapter 9.2.1.2
	4	Assembly object	See chapter 9.2.2.1
	6	Connection manager object	See chapter 9.2.1.3
	245d/F5h	Tcp/ip interface object	See chapter 9.2.1.4
	246d/F6h	Ethernet link object	See chapter 9.2.1.5
Required by drive profile	40d/28h	Motor data object	See chapter 9.2.2.2
	41d/29h	Control supervisor object	See chapter 9.2.2.3
	42d/2Ah	Ac/dc drive object	See chapter 9.2.2.4
Vendor-specific	160d/A0h	Vendor parameters object	See chapter 9.2.3.1
	161d/A1h	Motor control mode object	See chapter 9.2.3.3
	162d/A2h	Fault history object	See chapter 9.3.3.4
	190d/BEh	Assembly instance selector object	See chapter 9.2.3.2

9.2 Common industrial objects implemented by the OPTE9

9.2.1 Cip common required objects

9.2.1.1 Identity object, class 0x01

The Identity Object provides identification of and general information about the device.

Table 89. Identity object

Class name		Identity Object			
Class identifier		**1**			
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	1	Get	Revision	UINT	Class revision (1)
	2	Get	Max Instance	UINT	Maximum instance number (1)
	3	Get	Number of Instances	UINT	Number of object instances (1)
Class services	**Id**	**Name**		**Description**	
	1	Get_Attributes_All		Get all attributes	
	14_d $0E_h$	Get_Attribute_Single		Get single attribute	
Instance attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	1	Get	Vendor ID	UINT	Vendor identification
	2	Get	Device Type	UINT	General type of product
	3	Get	Product Code	UINT	Product identification
	4	Get	Revision	STRUCT of	Revision of the item the Identity Object represents
			Major Revision	USINT	
			Minor Revision	USINT	
	5		Status	WORD	Summary status of device
	6		Serial number	UDINT	Serial number of the device
	7		Product Name	SHORT STRING	Human readable identification
Instance services	**Id**	**Name**		**Description**	
	1	Get_Attributes_All		Get all attributes	
	5	Reset		Only reset type 0	
	14_d $0E_h$	Get_Attribute_Single		Get single attribute	

9.2.1.1.1. Instance attributes

Instance attribute "Vendor ID"

This number is assigned to vendors of CIP devices by the ODVA user organization. The vendor ID for Eaton Plc is $01BB_h$(443_d).

Instance attribute "Device Type"

This attribute indicates which device profile is implemented by the device. For Eaton drives this device number is 02_h("AC Drive" profile).

Instance attribute "Product Code"

This attribute reveals the vendor-assigned product code for a particular product within a device type.

Separate products must have different product codes if their configuration and/or runtime options are different.

OPTE9 will return the product code value based on the drive where the option board has been installed. This value can also be modified with the "Product Code Offset" parameter (see Chapter 6 "Commissioning").

When the "Mode" setting is set to "SVX/SPX Mode", setting the "Product Code Offset" has no affect and OPTE9 will emulate the OPTCQ option board and return value to 2.

Table 90. Drive type specific product code values

Drive type	Product code base value
SVX/SPX drives	1200
Emulating OPTCQ	2

Instance attribute "Revision"

This attribute, which consists of the Major and Minor Revision fields, identifies the revision of the item/device that the Identity Object is representing. The Major Revision is limited to values between 1 and 127, as the eighth bit is reserved by CIP and is zero.

Instance attribute "Status"

The value of the attribute presents the current status of the entire device. The coding of the field is defined in the table below.

Table 91. Status bit descriptions

Bit(s)	Called	Definition	
0	Owned	TRUE, if device has owner	
1		Reserved, is zero	
2	Configured	TRUE, if device has been configured (always true in OPTE9)	
3		Reserved, is zero	
		Value	**Description**
4-7	Extended device status	0	Self-testing or unknown
		1	Firmware upgrade in progress
		2	At least one faulted I/O connection
		3	No I/O connections established
		4	Non-Volatile configuration bad
		5	Major fault – either bit 10 or bit 11 is true
		6	At least one I/O connection in run mode
		7	At least one I/O connection established, all in idle mode
		8	The Status attribute is not applicable to this instance. Valid only for instances greater than one (1).
		9	Reserved
		10 thru 15	Vendor specific, not used by Eaton
8	Minor recoverable fault	TRUE, if recoverable problem detected.	
9	Minor unrecoverable fault	TRUE, if unrecoverable problem detected.	
10	Major recoverable fault	TRUE, if recoverable problem detected.	
11	Major unrecov erable fault	TRUE, if unrecoverable problem detected.	
12-15	Extended device status 2	Reserved, is zero	

The OPTE9 drive implements bits 0, 2, and 4-11 according to the specification (Extended Device Status values 1, 4 and 8 to 15 are not used by Eaton). The bits 8-11 must be set according to the faults occurring in the drive.

Instance attribute "Serial Number"

This attribute can be used in conjunction with the Vendor ID to form a unique identifier for each device on any CIP network.

The serial number is formed so that the first octet is 00 and the last 3 octets are taken from the end of the MAC address of the drive. For example, when the MAC address is 00:21:99:AA:BB:CC, then the serial number would be 00AABBCCh.

Instance attribute "Product Name"

This attribute contains human readable name identification for this instance. The OPTE9 will return the value which is combined from the drive type and application name.

When emulating the OPTCQ option board, the OPTE9 will always return text "OPTCQ"

9.2.1.1.2. Services

Instance service "Reset"

The OPTE9 supports only reset type 0.

The reset type 0 means that the device represented by the Identity Object will as closely as possible emulate the cycling of power.

If an error is detected, an error response is returned. Otherwise a successful Reset response is returned.

9.2.1.2 Message router object, class 0x02

The Message Router Object is mandatory in all CIP devices. It provides a messaging connection point through which a Client may address a service to any object class or instance in a target device. Although the object is mandatory, there are no mandatory attributes or services.

The OPTE9 drive does not currently implement any of the object's services or attributes

Table 92. Message router object

Class name	Message router object				
Class identifier	2				
Class attributes	Id	Access rule	Name	Datatype	Description
	–	–	–	–	–
Class services	Id	Name		Description	
	–	–		–	
Instance attributes	Id	Access rule	Name	Datatype	Description
	–	–	–	–	–
Instance services	Id	Name		Description	
	–	–		–	

9.2.1.3 Connection manager object, class 0x06

The communication characteristics between the applications in different devices are modelled using Connection Objects. The entities (devices) involved in a connection are referred to as endpoints. A Connection Manager is required in some CIP networks to control the aspects of Connection object instances.

The Connection Manager class allocates and manages the internal resources associated with both I/O and Explicit Messaging connections.

Table 93. Connection manager object

Class name		Connection manager object			
Class identifier		6			
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	1	Get	Revision	UINT	Class revision (1)
	2	Get	Max Instance	UINT	Maximum instance number (1)
	3	Get	Number of Instances	UINT	Number of object instances(1)
Class services	**Id**	**Name**		**Description**	
	1	Get_Attributes_All		Returns content of all (implemented) attributes in the class.	
	14d 0Eh	Get_Attribute_Single		Used to read single attribute value.	
Instance attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	1	Get	Open Requests	UINT	Number of Forward Open service requests received.
	2	Get	Open Format Rejects	UINT	Number of Forward Open service requests which were rejected due to bad format.
	3	Get	Open Resource Rejects	UINT	Number of Forward Open service requests which were rejected due to lack of resources.
	4	Get	Open Other Rejects	UINT	Number of Forward Open service requests which were rejected for other reasons.
	5	Get	Close Requests	UINT	Number of Forward Close service requests received.
	6	Get	Close Format Rejects	UINT	Number of Forward Close service requests which were rejected due to bad format.
	7	Get	Close Other Rejects	UINT	Number of Forward Close service requests which were rejected for other reasons
	8	Get	Connection Timeouts	UINT	Total number of connection timeouts that have occurred in connections controlled by this Connection Manager.
Instance services	**Id**	**Name**		**Description**	
	1	Get_Attributes_All		Returns content of all (implemented) attributes in the instance	
	14d 0Eh	Get_Attribute_Single		Used to read the single attribute value	
	78d 4Eh	Forward_Open		Opens a connection (maximum data size is 511 bytes)	
	84d 54h	Forward_Close		Closes a connection	

9.2.1.3.2. Services

Instance service "Forward Open"

The Forward Open service is used to open a connection to a target device. If the path between devices consists of multiple links, then local connections between these are also established.

The minimum time for the RPI (Request Packet Interval) is 4 ms. The connection object instance number is 103_d (67_h) except when emulating the OPTCQ option board. Then connection object instance number is 1.

Instance service "Forward Close"

The Forward Close service is used to close a connection between two devices (and all nodes in the connection path).

9.2.1.4 TCP/IP interface object, class 0xF5

The TCP/IP Interface Object provides an interface to configure the device's TCP/IP settings. With this object, you can configure, for example, the device's IP address, network mask and so on.

Table 94. TCP/IP Interface object

Class name		**TCP/IP object**			
Class identifier		245_d/$F5_h$			
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	1	Get	Revision	UINT	Class revision (4)
	2	Get	Max Instance	UINT	Maximum instance number (1)
	3	Get	Number of Instances	UINT	Number of object instances(1)
Class services	**Id**	**Name**		**Description**	
	1	Get_Attributes_All		Returns content of all (implemented) attributes in the class.	
	14_d $0E_h$	Get_Attribute_Single		Used to read single attribute value.	
Instance attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	1	Get	Status	DWORD	Interface status
	2	Get	Configuration Capability	DWORD	Interface capability flags
	3	Get/set	Configuration Control	DWORD	Interface control flags
	4	Get	Physical Link Object	STRUCT of	Path to physical link object
			Path size	UINT	Size of the path
			Path	Padded EPATH	Logical segments identifying the physical link object
	5	Get/set	Instance Configuration	STRUCT of	TCP/IP network interface configuration
			IP Address	UDINT	The device's IP address
			Network Mask	UDINT	The device's network mask
			Gateway Address	UDINT	Default gateway address
			Name Server	UDINT	Primary name server
			Name Server 2	UDINT	Secondary name server
			Domain Name	STRING	Default domain name
	6	Get/set	Host Name	STRING	Host name
	10_d / $0A_h$	Get/set	Select ACD	BOOL	Activates the use of ACD (enabled by default)
	11_d/ $0B_h$	Get	Last Conflict Detected	STRUCT of:	Structure containing information related to the last conflict detected.
			ACD activity	USINT	State of ACD activity
			Remote MAC	Array of 6 USINT	MAC address of last conflict source.
			ARP PDU	Array of 28 USINT	Copy of the last ARP PDU in which a conflict was detected.
	13_d / $0D_h$	Set	Encapsulation Inactivity Time-out	UINT	Number of seconds of inactivity before TCP connection is closed
Instance services	**Id**	**Name**		**Description**	
	1	Get_Attributes_All		Returns content of all (implemented) attributes in the instance	
	14_d $0E_h$	Get_Attribute_Single		Used to read single attribute value.	
	16_d 10_h	Set_Attribute_Single		Used to write a single attribute value.	

9.2.1.4.1. Instance attributes

Instance attribute "Status"

This attribute presents the status of the TCP/IP network interface.

Table 95. Status bit descriptions

Bit(s)	Called	Definition	Value	Definition
0-3	Interface configuration status	Indicates the status of the interface configuration attribute	0	The Interface Configuration Attribute has not been configured
			1	The Interface Configuration Attribute contains valid configuration obtained from BOOTP, DHCP or non-volatile storage.
			2	The IP address member of the Interface Configuration Attribute contains valid configuration, obtained from hardware settings
			3-15	Reserved for future use
4	Mcast pending	Indicates a pending configuration change in the TTL Value and/ or Mcast Config attributes.		
5	Interface configuration pending	Indicates a pending configuration change in the Interface Configuration attribute.		
6	Acdstatus	Set(1) Address Conflict Detected, Clear(0) No Address Conflict Detected		
7-31	Reserved	Always zero		

Instance attribute "Configuration Capability"

This attribute presents the capability flags (that is, the support for the optional network configuration capability) of the TCP/IP network interface.

Table 96. Configuration capability bit descriptions

Bit(s)	Called	Definition
0	BOOTP Client	Supports BOOTP (FALSE)
1	DNS Client	Supports capable of resolving DNS names (FALSE)
2	DHCP Client	Supports DHCP (TRUE)
3	DHCP-DNS Update	Always zero
4	Configuration Settable	TRUE, if configuration settable
5	Hardware Configurable	Configuration can be obtained from hardware settings (FALSE)
6	Interface Configuration Change Requires Reset	Configuration change results in reset (FALSE)
7	AcdCapable	Supports ACD (TRUE)
8-31	Reserved	Reserved, always zero

Instance attribute "Configuration Control"

This attribute allows control of the TCP/IP network interface configuration.

When using the Configuration Control attribute, the device can be configured to use statically assigned IP values or DHCP. If the value is changed from DHCP to statically assigned, the device will continue using the current IP address. When changing from statically assigned to DHCP, the drive will try to get an IP address from the DHCP server. If this fails, the communication with the drive cannot be re-opened and you must set the IP address manually from the panel or enable the DHCP server in the network.

Changing the Configuration Control is not allowed if the I/O connection is open.

Table 97. Configuration control bit descriptions

Bit(s)	Called	Definition	Value	Definition
0-3	Configuration Method	Determines how the device obtains its IP related configuration	0	The device uses statically-assigned IP configuration values.
			1	The device obtains its interface configuration values via BOOTP
			2	The device obtains its interface configuration values via DHCP
			3-15	Reserved for future use.
4	DNS Enable	If TRUE, the device resolves host names by querying a DNS server		
5-31	Reserved	Reserved, always zero		

In the OPTE9 product, if the "IP Address Mode" panel parameter is "Fixed IP", the value of the Configuration Control is 0. If the "IP Address Mode" is "DHCP", the value of the Configuration Control is 2.

The OPTE9 does not support BOOTP or DNS.

Instance attribute "Physical Link Object"

This attribute identifies the object which is associated with the underlying physical communications interface (in the case of Ethernet, for example, the IEEE 802.3 interface). The attribute consists of two components; a Path Size, which reveals the number of UINT values in the path, and the Path itself.

In the OPTE9, the path points to an instance of the EtherNet Link Object. The value of the Path Size is 2 (total of four octets) and the value of the Path is 20_h $F6_h$ 24_h XX_h, where XX is the instance number of the EtherNet Link object.

Instance attribute "Instance Configuration"

This attribute contains the configuration parameters required for a device to operate as a TCP/IP node. The contents of the attribute depend on how the device has been configured to obtain its IP parameters (the "Configuration Method" field in the Configuration Control attribute). If the device uses a static IP address (Configuration Method value is 0), the values in the Interface Configuration are those statically assigned and stored in the non-volatile memory. If the device uses DHCP (or BOOTP) (Configuration Method value is 1 or 2), the Interface Configuration values will contain the configuration obtained through this channel. Until the BOOTP/DHCP reply is received, the values are 0.

Changing the Instance Configuration is not allowed when the I/O connection is open or Configuration Control-attribute is not set to "statically-assigned".

The IP address, Network Mask and Gateway address consists of four bytes. For example, the IP address 192.168.0.10 would be in format: $C0_h$, $A8_h$, 00_h, $0A_h$.

Table 98. Instance configuration

Interface configuration	STRUCT of:	Description	Semantics of the value
IP address	UDINT	The device's IP address	Value of 0 indicates no IP address has been configured. Otherwise, the IP address must be set to a valid Class A, B, or C address and must not be set to the loopback address (127.0.0.1).
Network mask	UDINT	The Device's network mask	Value of 0 indicates no network mask address has been configured.
Gateway address	UDINT	Default gateway address	Value of 0 indicates no IP address has been configured. Otherwise, the IP address must be set to a valid Class A, B, or C address and must not be set to the loopback address (127.0.0.1).
Name server	UDINT	Primary name server	Value of 0 indicates no name server address has been configured. Otherwise, the name server address must be set to a valid Class A, B, or C address.
Name server 2	UDINT	Secondary name server	Value of 0 indicates no secondary name server address has been configured. Otherwise, the name server address must be set to a valid Class A, B, or C address.
Domain name	STRING	Default domain name	ASCII characters. Maximum length is 48 characters. Must be padded to an even number of characters (pad not included in length). A length of 0 indicates that no Domain Name is configured.

Instance attribute "Host Name"

This attribute contains the device's host name. The maximum length is 64 ASCII characters. The name is padded to an even number of characters. The Attribute Host Name is used only for information purpose.

Instance attribute "Encapsulation Inactivity Timeout"

The Encapsulation Inactivity Timeout attribute is used to enable the TCP socket cleanup (closing) when the defined number of seconds have elapsed with no Encapsulation activity. The default value is 120 seconds. The TCP keep-alive traffic does not count as Encapsulation activity.

Table 99.

Value	Description
0	Disable
1-3600	Timeout in seconds

Instance attribute "Select ACD"

This attribute is used to enable or disable ACD (Address Conflict Detection) functionality.

For more information see Chapter 4.5 "ACD (Address Conflict Detection)".

Instance attribute "Last Conflict Detected"

This attribute contains information of the last IP address conflict. The content of this attribute can be resetted by writing zero to this attribute.

The struct member "ACD Activity" tells the state of ACD algorithm when the last conflict was detected. Possible values are defined in the table below.

Table 100. ACD activity values

Value	ACD mode	Description
0	No conflict detected (default)	No conflict has been detected since this attribute was last cleared.
1	Probe IPV4 address	Last conflict detected during IPV4 address probe state
2	Ongoing detection	Last conflict detected during OngoingDetection-state or subsequent DefendWithPolicyB state
3	Semi active probe	Last conflict detected furint SemiActiveProbe-state or subsequent DefendWithPolicyB-state

The struct member "Remote MAC" tells the MAC address the source of the last IP address conflict.

The struct member "ARP PDU" contains the ARP message (raw copy) received from the source of the IP address conflict. Content of the ARP message is described in the table below.

Table 101. The ARP PDU in binary format

Field size (bytes)	Field description
2	Hardware type (1 for Ethernet HW)
2	Protocol type (0x800 for IP)
1	Hardware size (6 for Ethernet HW)
1	Protocol size (4 for IP)
2	Operation code (1 for request or 2 for response)
6	Sender MAC address
4	Sender IP address
6	Target MAC address
4	Target IP address

9.2.1.5 Ethernet link object, class 0xf6

Ethernet Link Object provides interface to Ethernet link counters and attributes. With this object, user can retrieve for example link speed.

Table 102. Ethernet link object

Class name		Ethernet link object			
Class identifier		246_d/$F6_h$			
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	1	Get	Revision	UINT	Class revision (3)
	2	Get	Max Instance	UINT	Maximum instance number (1)
	3	Get	Number of Instances	UINT	Number of object instances(1)
Class services	**Id**		**Name**		**Description**
	1		Get_Attributes_All		Returns content of all (implemented) attributes in the class.
	14_d $0E_h$		Get_Attribute_Single		Used to read single attribute value.
Instance attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	1	Get	Interface Speed	UDINT	Interface speed currently in use
	2	Get	Interface Flags	DWORD	Interface status flags
	3	Get	Physical Address	ARRAY of 6 USINTs	MAC layer address
	4	Get	Interface Counters	STRUCT of 11 UDINTs	Interface counters. See Table 104
	5	Get	Media Counters	STRUCT of 12 UDINTs	Media specific counters. See Table 105
	7	Get	Interface Type	USINT	Type of interface: twisted pair, fiber, internal, etc
	8	Get	Interface State	USINT	Current state of the interface: operational, disabled, etc
	9	Get/Set	Admin State	USINT	Administrative state: enable, disable
	10_d $0A_h$	Get	Interface Label	SHORT STRING	Human readable identification
Instance services	**Id**	**Name**		**Description**	
	1	Get_Attributes_All		Returns content of all (implemented) attributes in the instance	
	14_d $0E_h$	Get_Attribute_Single		Used to read single attribute value	
	16_d 10_h	Set_Attribute_Single		Used to write a single attribute value.	
	76_d $4C_h$	Get_and_Clear		Gets then clears the specified attribute (Interface Counters, Media Counters). Not instance/class dependent service.	

9.2.1.5.1. Instance attributes

Instance attribute "Interface Speed"

The attribute reveals the currently used speed in the interface. The speed is announced as an integer number, with the unit Mbps, e.g. 0, 10, 100 etc. The value 0 indicates that the interface speed is indeterminate.

Instance attribute "Interface Flags"

The attribute contains status and configuration information about the physical interface.

Table 103. Interface flag bit descriptions

<table>
<tr><th>Bit(s)</th><th>Called</th><th colspan="2">Definition</th></tr>
<tr><td>0</td><td>Link Status</td><td colspan="2">One, if link is active</td></tr>
<tr><td>1</td><td>Half/Full Duplex</td><td colspan="2">One, if full duplex</td></tr>
<tr><td rowspan="6">2-4</td><td rowspan="6">Negotiation Status</td><td>Value</td><td>Definition</td></tr>
<tr><td>0</td><td>Auto-negotiation in progress</td></tr>
<tr><td>1</td><td>Auto-negotiation and speed detection failed. Using default values for speed and duplex.</td></tr>
<tr><td>2</td><td>Auto-negotiation failed but detected speed. Duplex was defaulted.</td></tr>
<tr><td>3</td><td>Successfully negotiated speed and duplex.</td></tr>
<tr><td>4</td><td>Auto-negotiation not attempted. Forced speed and duplex.</td></tr>
<tr><td>5</td><td>Manual Setting Requires Reset</td><td colspan="2">0 indicates the interface can automatically activate changes to link parameters (auto-negotiate, duplex mode, interface speed). 1 indicates the device requires a Reset service be issued to its Identity Object in order for the changes to take effect.</td></tr>
<tr><td>6</td><td>Local Hardware Fault</td><td colspan="2">0 indicates the interface detects no local hardware fault; 1 indicates a local hardware fault is detected.</td></tr>
<tr><td>7-31</td><td>Reserved</td><td colspan="2">Always zero</td></tr>
</table>

Instance attribute "Physical Address"

The attribute reveals the MAC layer address of the physical interface.

Instance attribute "Interface Counters"

The attribute is a collection of counters related to the Ethernet physical interface. The OPTE9 option board has only single MAC address and therefore implements only single set of counter values. Only packets sent or received by the device itself are counted.

Table 104. Interface counters

Field name	Data type	Description
In octets	UDINT	The number of octets received on the interface (including framing characters).
In unicast packets	UDINT	The number of unicast packets received on the interface.
In nonunicast packets	UDINT	The number of non-unicast packets received on the interface.
In discards	UDINT	Inbound packets received on the interface but which were discarded.
In errors	UDINT	Inbound packets received on the interface but which contained errors (excluding Discards).
In unknown protocols	UDINT	Inbound packets received on the interface which belonged to unknown protocols.
Out octets	UDINT	The number of octets sent on the interface (including framing characters).
Out unicast packets	UDINT	The number of unicast packets requested to be transmitted on the interface, including those that were discarded or not sent.
Out nonunicast packets	UDINT	The number of non-unicast packets requested to be transmitted on the interface, including those that were discarded or not sent.
Out discards	UDINT	Outbound packets which were discarded.
Out errors	UDINT	Outbound packets which contained errors (excluding Discards).

Instance attribute “Media Counters”

The attribute is a collection of counters related to the Ethernet physical interface.

Table 105. Media counters

Field name	Data type	Description
Alignment errors	UDINT	Frames received that are not an integral number of octets in length.
Fcs errors	UDINT	Frames received that do not pass the FCS check.
Single collisions	UDINT	Successfully transmitted frames which experienced exactly one collision.
Multiple collisions	UDINT	Successfully transmitted frames which experienced more than one collision.
Sqe test errors	UDINT	The number of times SQE test error message is generated.
Deferred transmissions	UDINT	Frames for which the first transmission attempt is delayed because the medium is busy.
Late collisions	UDINT	Number of times a collision is detected later than 512 bit-times into the transmission of a packet.
Excessive collisions	UDINT	Frames for which transmission fails due to excessive collisions.
Mac transmit errors	UDINT	Frames for which transmission fails due to an internal MAC sub layer transmit error.
Carrier sense errors	UDINT	Times that the carrier sense condition was lost or never asserted when attempting to transmit a frame.
Frame too long	UDINT	Frames received that exceed the maximum permitted frame size.
Mac receive errors	UDINT	Frames for which reception on an interface fails due to an internal MAC sub layer receive error.

Instance attribute “Interface Type”

The attribute indicates the type of the Ethernet interface, i.e. twisted-pair cable, optical fiber, device-internal etc. The OPTE9 will always return the value 2, twisted-pair cable.

Instance attribute “Interface State”

The attribute indicates the current state of the Ethernet interface, i.e. operational, disabled etc.

Table 106. Interface state

Value	Interface state
0	Unknown interface state
1	The interface is enabled and is ready to send and receive data
2	The interface is disabled
3	The interface is testing
4-255	Reserved

Instance attribute “Admin State”

The attribute indicates the ability to use the Ethernet interface for administration, for example, for changing the settings.

The OPTE9 supports this attribute with the value 01_h (administration enabled). An attempt to disable the administration (by writing value 02_h) will result in an error.

9.2.2 Objects present in an AC/DC drive

9.2.2.1 Assembly object, class 0x04

The assembly object groups (or assembles) the attribute values into a single block of data.

Table 107. Assembly object

Class name		Assembly object				
Class identifier		4				
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**	
	1	Get	Revision	UINT	Class revision (2)	
	2	Get	Max Instance	UINT	Maximum instance number (137_h)	
	3	Get	Number of Instances	UINT	Number of object instances(17)	
Class services	**Id**		**Name**	**Description**		
	1		Get_Attributes_All	Returns content of all (implemented) attributes in the class.		
	14_d $0E_h$		Get_Attribute_Single	Used to read single attribute value.		
Instance attributes	**Id**		**Access rule**	**Name**	**Datatype**	**Description**
	3		Set	Data	ARRAY of BYTE	Assembly data
Instance services	**Id**		**Name**		**Description**	
	14_d $0E_h$		Get_Attribute_Single		Used to read single attribute value	
	16_d 10_h		Set_Attribute_Single		Used to write a single attribute value	

9.2.2.1.1. Instance attributes

Instance attribute "Data"

This attribute can be used to get assembly data. The content and length of the data depends on the configuration of the assembly instance.

9.2.2.2 Motor data object, class 0x28

Motor Data Object provides interface to the motor data attributes, for example "motor type".

Table 108. Motor data object

Class name		Motor data object			
Class identifier		$40_d/28_h$			
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	–	–	–	–	–
Class services	**Id**	**Name**		**Description**	
	–	–		–	
Instance attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	3	Get/Set	MotorType	USINT	Motor type
	6	Get/Set	RatedCurrent	UINT	Rated Stator Current Units: [100mA]
	7	Get/Set	RatedVoltage	UINT	Rated Base Voltage Units: [V]
	9	Get/Set	RatedFreq	UINT	Rated Electrical Frequency Units: [Hz]
	12d 0Ch	Get	PoleCount	UINT	Number of poles in the motor.
	15d 0Fh	Get/Set	Base Speed	UINT	Nominal speed at rated frequency from nameplate Units: [RPM]
Instance services	**Id**	**Name**		**Description**	
	14d 0Eh	Get_Attribute_Single		Used to read single attribute value.	
	16d 10h	Set_Attribute_Single		Used to write a single attribute value.	

9.2.2.2.1. Instance attributes

Instance attribute "Motortype"

The OPTE9 supports values 3 (Permanent Magnet Synchronous Motor) and 7 (Squirrel Cage Induction Motor).

Instance attribute "Ratedcurrent"

This attribute allows reading and writing of the motor rated current. The unit of the attribute is 100 milliamperes.

Instance attribute "Ratedvoltage"

This attribute allows reading and writing of the motor rated voltage. The unit of the attribute is 1 volt.

Instance attribute "Ratedfreq"

This attribute allows reading and writing of the motor rated electrical frequency. The unit of the attribute is 1 hertz.

Instance attribute "Polecount"

This attribute allows reading and writing of the number of poles in the motor. The unit of the attribute is 1.

Instance attribute "Base Speed"

This attribute allows reading and writing of the nominal speed at rated frequency. The unit of the attribute is 1 RPM

9.2.2.3 Control supervisor object, class 0x29

Control Supervisor Object provides an interface for drive management. You can, for example, start and stop the motor with this object.

Table 109. Control supervisor object

Class name		Control supervisor object			
Class identifier		$41_d/29_h$			
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	–	–	–	–	–
Class services	**Id**	**Name**		**Description**	
	1	Get_Attributes_All		Returns content of all (implemented) attributes in the class.	
	14d 0Eh	Get_Attribute_Single		Used to read single attribute value.	
Instance attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	3	Get/Set	Run1	BOOL	Run forward
	4	Get/Set	Run2	BOOL	Run reverse
	5	Get/Set	NetCtrl	BOOL	Request Run/Stop control to be local or from network
	6	Get	State	USINT	State. See Table 111.
	7	Get	Running1	BOOL	True, when running forward
	8	Get	Running2	BOOL	True, when running in reverse
	9	Get	Ready	BOOL	True, when Ready or Enabled or Stopping
	10d 0Ah	Get	Faulted	BOOL	True, when fault is active
	11d 0Bh	Get	Warning	BOOL	True, when warning/alarm is active
	12d 0Ch	Get/Set	FaultRst	BOOL	Resets fault when transits from zero to one
	13d 0Dh	Get	FaultCode	UINT	If in Faulted-state, Fault- Code indicates the active fault. Otherwise last error or zero after startup.
	15d 0Fh	Get	CtrlFromNet	BOOL	True, control is from network False, control is local.
	21d 15h	Get/Set	NetIdleMode	USINT	Mode on reception of CIP communication IDLE event.
Instance services	**Id**	**Name**		**Description**	
	5	Reset		Resets drive to startup state.	
	14d 0Eh	Get_Attribute_Single		Used to read single attribute value.	
	16d 10h	Set_Attribute_Single		Used to write a single attribute value.	

9.2.2.3.1. Instance attributes

Instance attribute "Run1"

This attribute affects the run/stop behavior of the drive. See Table 110.

Instance attribute "Run2"

This attribute affects the run/stop behavior of the drive. See Table 110.

Table 110. Run/stop event matrix

Run1	Run2	Trigger event	Run type
0	0	Stop	N/A
0 -> 1	0	Run	Run1
0	0 -> 1	Run	Run2
0 -> 1	0 -> 1	No Action	N/A
1	1	No Action	N/A
1 -> 0	1	Run	Run2
1	1 -> 0	Run	Run1

Instance attribute "Netctrl"

This attribute allows the network to request the run/stop control to be assigned to the network. If the bit is 0, given control word is not updated to the drive. If the bit is 1, this means that the run/stop control is requested to this network interface.

Note: The actual assignment of the run/stop control to this network interface is reflected in attribute 15_d.

The OPTE9 will not force control to the network. You must change the control location (fieldbus/IO/keypad) from the drive parameters.

Instance attribute "State"

This attribute reveals the state of the device according to the table below. See also Chapter "9.2.2.3.3. Control Supervisor State Machine".

Table 111. State value descriptions

Value	Definition
0	Vendor Specific
1	Startup
2	Not_Ready
3	Ready
4	Enabled
5	Stopping
6	Fault_Stop
7	Faulted

Instance attribute "Running1"

This attribute is used to describe the run state of the drive. The value of the attribute is 1, if one of the below conditions are fulfilled:

- The "State" attribute has the value 4 ("Enabled") and the bit "Run1" has the value 1, or
- The "State" attribute has the value 5 ("Stopping") and the bit "Running1" has the value 1, or
- The "State" attribute has the value 6 ("Fault_Stop") and the bit "Running1" has the value 1

Otherwise, the value of this attribute is 0.

Instance attribute "Running2"

This attribute is used to describe the run state of the drive. The value of the attribute is 1, if one of the below conditions are fulfilled:

- The "State" attribute has the value 4 ("Enabled") and the bit "Run2" has the value 1, or
- The "State" attribute has the value 5 ("Stopping") and the bit "Running2" has the value 1, or
- The "State" attribute has the value 6 ("Fault_Stop") and the bit "Running2" has the value 1

Otherwise, the value of this attribute is 0.

Instance attribute "Ready"

This attribute is used to signal the state of the drive that it is ready for operation. The value of the attribute is 1 if the value of the "State" attribute is either 3 ("Ready"), 4 ("Enabled") or 5 ("Stopping"). Otherwise the value of this attribute is 0.

Instance attribute "Faulted"

This attribute is used to signal that one or several faults have occurred in the drive. The value of the attribute is 1 if a fault has occurred and has not been acknowledged. Otherwise, the attribute has the value 0 indicating that no faults are present.

Instance attribute "Warning"

This attribute is used to signal that one or several warnings have appeared in the drive. The value of the attribute is 1 if a warning has appeared and has not been acknowledged. Otherwise, the attribute has the value 0 indicating that no warnings are present.

Instance attribute "Faultrst"

This attribute is used to reset faults and warnings in the drive. The attribute is write-only. Changing the value of the attribute from 0 to 1 (rising-edge) resets the faults in the drive. If the value is static 0, no reset action is started.

Instance attribute "Faultcode"

This attribute is used to read the kind of fault which has caused the device to transition into the "Faulted" state. In the case of multiple faults occurring simultaneously, only one code is reported. If the device is not in the Faulted state, the FaultCode attribute indicates the fault which caused the last transition to the Faulted state.

Instance attribute "Ctrlfromnet"

It indicates whether the run/stop control is assigned to the local interface or to this network interface. When the value of the attribute is 0, the control is local. When the value of the attribute is 1, the run/stop control is assigned to the network interface.

Instance attribute "Netidlemode"

This attribute establishes the mode of operation on reception of network idle communication. Default value for this attribute is zero. Possible values are listed in the table below.

Table 112. NetIdleMode values

Mode	Action	Error/ Warning
0	Stops motor	Fault
1	Ignored. IO data is not used, drive stays in state which was active before reception of IDLE	–
2	Vendor specific, fieldbus fault is activated (actual behavior depends on drive application and parametrization).	Fault
Other values are not supported		

9.2.2.3.2. Services

Instance service "Reset"

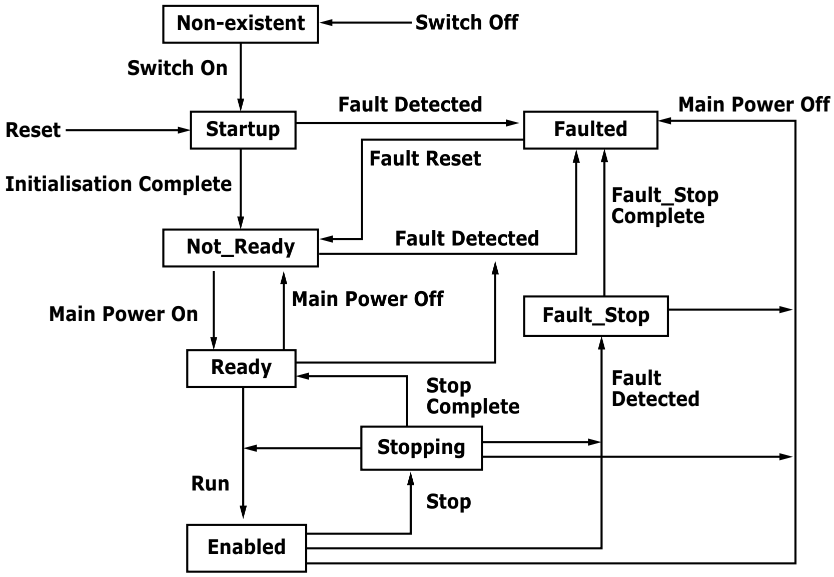
The Control Supervisor Object has a instance service named "Reset" which has the Service Code 05_h. The service resets the drive to the start-up state.

If an error is detected, an error response is returned. Otherwise a successful Reset response is returned.

9.2.2.3.3. Control supervisor state machine

The Control Supervisor Object defines a state machine for governing the behaviour of devices. The figure below describes the states and transitions of the state machine.

Figure 32. Control supervisor state machine



9.2.2.4 AC/DC Drive object, class 0x2A

The AC/DC Drive Object models the functions specific to an AC or DC drive.

Table 113. AC/DC drive object

Class name	**AC/DC drive object**				
Class identifier	42_d/$2A_h$				
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	–	–	–	–	–
Class services	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	–	–	–	–	–
Instance attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	3	Get/Set	AtReference	BOOL	True, when drive actual at reference (speed or torque reference) based on mode
	4	Get/Set	NetRef	BOOL	Requests torque or speed reference to be from the network. False, when Set Reference not DN Control True, when Set Reference at DN Control
	5	Get/Set	NetProc	BOOL	Requests process control reference to be from the network. False, when Set Process not DN Control True, when Set Process at DN Control
	6	Get	DriveMode	USINT	Drive mode. See Table 114.
	7	Get	SpeedActual	INT	Actual drive speed Units: RPM/$2^{SpeedScale}$ where SpeedScale is attribute 22_d
	8	Get	SpeedRef	INT	Speed reference Units: RPM/$2^{SpeedScal}$ where SpeedScale is attribute 22_d
	11_d $0B_h$	Get	TorqueActual	INT	Actual torque Units: Nm/$2^{TorqueScale}$ where TorqueScale is attribute 24_d
	12_d $0C_h$	Get/Set	TorqueRef	INT	Torque reference Units: Nm/$2^{TorqueScale}$ where TorqueScale is attribute 24_d
	13_d $0D_h$	Get	ProcessActual	INT	Actual process control value Units: % ProcessScale is not supported.
	14_d $0E_h$	Get/Set	ProcessRef	INT	Process control reference set point. Units: % ProcessScale is not supported
	22_d 16_h	Get/Set	SpeedScale	SINT	Speed scaling factor. Scaling is accomplished as follows: ScaledSpeed = RPM/$2_{SpeedScale}$. Range: -4 .. 7
	24_d 18_h	Get/Set	TorqueScale		Torque scaling factor. Scaling is accomplished as follows: ScaledTorque = Nm/$2^{TorqueScale}$ Range: -8 .. 7
	29_d $1D_h$	Get	RefFromNet	BOOL	Status of torque/speed reference False, when local torque/speed reference. True, when network torque/ speed reference
Instance services	**Id**		**Name**		**Description**
	14_d $0E_h$		Get_Attribute_Single		Used to read single attribute value.
	16_d 10_h		Set_Attribute_Single		Used to write a single attribute value.

9.2.2.4.1. Instance attributes

Instance attribute "AtReference"

This attribute indicates whether the actual value is at the reference value (e.g. the drive actual speed is the same as what is requested in the speed reference). If the bit is 1, the drive actual value is at the reference value.

Instance attribute "NetRef"

When the bit is 1, the torque or speed reference is requested to be allocated to this network interface. If the bit is 0, then no such request is made.

NOTE! The actual assignment of the reference to this network interface is reflected in the attribute 29_d.

Instance attribute "NetProc"

This attribute is used to request torque or speed reference to be local or from network. Values are:

- 0 = Set reference not DN control
- 1 = Set reference at DN control

Instance attribute "DriveMode"

Allowed values for Drive Mode defined in the table below.

Note: The actual drive mode support depends on the used drive and application. Check the actual drive mode support from the drive application manual.

Table 114. Supported drive modes

Value	Name	Supported
0	Vendor-specific mode	Yes (Open loop frequency)
1	Open loop speed	Yes (Open loop speed)
2	Closed loop speed control	Yes (Closed loop speed)
3	Torque control	Yes (Open loop torque)
4	Process control (e.g. PI control)	No
5	Position control	No

Changes to DriveMode attribute must be reflected in the "Motor Control Mode Object". The drive will respond with "Invalid attribute value" status code 0x09, if other values are written.

Instance attribute "SpeedActual"

This attribute allows reading of the speed actual value. The unit of the attribute must be ($RPM/2^{SpeedScale}$), where the SpeedScale is attribute 22_d.

If the SpeedScale attribute is not used by the master, the default unit [1 RPM] is assumed. This is equivalent to the value 0 being used for the SpeedScale.

Instance attribute "SpeedRef"

This attribute allows reading and writing of the speed reference set point. The unit of the attribute must be ($RPM/2^{SpeedScale}$), where SpeedScale is attribute 22_d.

If the SpeedScale attribute is not used by the master, the default unit [1 RPM] is assumed. This is equivalent to the value 0 being used for SpeedScale.

Instance attribute "TorqueActual"

This attribute allows reading of the torque actual value. The unit of the attribute must be ($Nm/2^{TorqueScale}$), where the TorqueScale is attribute 24_d.

If the TorqueScale attribute is not used by the master, the default unit [1 Nm] is assumed. This is equivalent to the value 0 being used for the TorqueScale.

Instance attribute "TorqueRef"

This attribute allows reading and writing of the torque reference set point. The unit of the attribute must be ($Nm/2^{TorqueScale}$), where the TorqueScale is attribute 24_d.

If the TorqueScale attribute is not used by the master, the default unit [1 Nm] must be assumed. This is equivalent to the value 0 being used for TorqueScale. To set the drive to the torque control, see Chapter 15 "APPENDIX 5 – FIELDBUS PARAMETRISATION".

Instance attribute "ProcessActual"

This attribute allows reading of the process actual value. The unit of the attribute must be ($\%/2^{ProcessScale}$), where the ProcessScale is attribute 25_d. The OPTE9 EtherNet/IP does not support ProcessScale. See Chapter "9.3.1.1.4. Assembly Instance 25 – Extended Process Control Output" for details.

Instance attribute "ProcessRef"

This attribute allows reading and writing of the process reference set point. The unit of the attribute must be ($\%/2^{ProcessScale}$), where ProcessScale is attribute 25_d. The OPTE9 EtherNet/IP does not support ProcessScale. See Chapter "9.3.1.1.4. Assembly Instance 25 – Extended Process Control Output" for details.

Instance attribute "SpeedScale"

The SpeedActual and SpeedRef values are scaled according to the value of this attribute. The default value is 0_d.

The largest allowed value for this attribute in the OPTE9 drive is 7 (allowing resolution of 0.0078 RPM) and the minimum allowed value is -4 (allowing resolution of 16 RPM). The maximum speed value for input/output is thus ca 524000 RPM.

Instance attribute "TorqueScale"

The TorqueActual and TorqueRef values are scaled according to the value of this attribute. The default value is 0_d.

The largest allowed value for this attribute in the OPTE9 is 7 (allowing resolution of 0.0078 Nm) and the minimum allowed value is -8 (allowing resolution of 256 Nm). The maximum torque value for input/output is thus ca 8.4 MNm.

Instance attribute "RefFromNet"

This attribute reveals whether the torque or speed reference is local or from the network. If the reference is local, the value of the attribute is 0. If the reference is from the network, then the value of the attribute is 1.

9.2.3 Vendor specific objects

9.2.3.1 Vendor parameters object, class 0xA0

The Vendor Parameters Object is a vendor-specific object which allows the user to access any application parameter from the drive.

Table 115. Vendor parameter object

Class name		Vendor parameter object			
Class identifier		160_d / $A0_h$			
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	–	–	–	–	–
Class services	**Id**	**Name**			**Description**
	–	–			–
Instance attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	XX	Get/Set	Parameter value	UINT	Parameter Value
Instance services	**Id**	**Name**			**Description**
	14_d $0E_h$	Get_Attribute_Single			Used to read single attribute value.
	16_d 10_h	Set_Attribute_Single			Used to write a single attribute value.

9.2.3.1.1. Instance attributes

Instance attribute "Parameter Value"

If you want to read the value of a drive parameter, for example ,"Motor control mode" $ID600_d$, set the instance attribute to value 600_d and the instance number to 1 to the request. The data type of the parameter value can be 8, 16 or 32 bits.

The OPTE9 also supports an old method from the OPTCQ option board of reading/writing ID values when the PLC supports only 8 bit instance attributes. This method is bit more complex.

In this mode, set the instance number to high octet of the ID, and the instance attribute as low octet of the ID. For example, if you want to read the value of ID 2291_d ($08F3_h$), the Get_Attribute_Single service request is targeted at the Vendor Parameters class, instance 08_h and attribute $F3_h$.

9.2.3.1.2. Services

Instance service "Get_Attribute_Single"

When invoked in an instance, the parameter ID to be fetched from the drive is calculated, then the read operation is started and once available, a response is provided to the master.

The format of the message is as follows:

Table 116.

Field	Data
Service Code	$0E_h$
Class Code	$A0_h$
Instance Number	01_h
Attribute ID	$XXXX_h$

The old format of the message is as follows (OPTCQ option board):

Table 117.

Field	Data
Service Code	$0E_h$
Class Code	$A0_h$
Instance Number	YY_h
Attribute ID	XX_h

Instance service "Set_Attribute_Single"

When invoked in an instance, the parameter ID to be modified in the drive is calculated. The data type, write permission etc. are verified before the write operation is started. When the operation finishes, or if an error occurs, an appropriate response is provided to the master.

The format of the message is as follows:

Table 118.

Field	Data
Service Code	10_h
Class Code	$A0_h$
Instance Number	01_h
Attribute ID	$XXXX_h$
Attribute Data	Parameter-specific

The old format of the message is as follows (OPTCQ option board):

Table 119.

Field	Data
Service Code	10_h
Class Code	$A0_h$
Instance Number	YY_h
Attribute ID	XX_h
Attribute Data	Parameter-specific

9.2.3.2 Assembly instance selector object, class 0xBE

The Assembly Instance Selector Object is a vendor-specific object available in the OPTCQ option board and the OPTE9. It allows the user to get and set the input and output instances used.

The OPTE9 option board will automatically change the current assembly instances by what is requested in the connection opening. This means that it is not mandatory to pre-set assembly instances.

Table 120. Assembly instance selector object

Class name		Assembly instance selector object			
Class identifier		190_d / BE_h			
Class attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	–	–	–	–	–
Class services	**Id**	**Name**			**Description**
	–	–			–
Instance attributes	**Id**	**Access rule**	**Name**	**Datatype**	**Description**
	3	Get/Set	InputInstance	USINT	Input assembly instance
	4	Get/Set	OutputInstance	USINT	Output assembly instance
	Id	**Name**			**Description**
Instance services	14_d $0E_h$	Get_Attribute_Single			Used to read single attribute value.
	16_d 10_h	Set_Attribute_Single			Used to write a single attribute value.

9.2.3.2.1. Instance attributes

Instance attribute "Inputinstance"

This attribute shows the currently active (or what was last used) input assembly instance id.

Instance attribute "Outputinstance"

This attribute shows the currently active (or what was last used) output assembly instance id.

9.2.3.2.2. Services

Instance service "Get_Attribute_Single"

The format of the message is as follows.

Table 121.

Field	Data
Service Code	$0E_h$
Class Code	BE_h
Instance Number	01_h
Attribute ID	03_h or 04_h

Instance service "Set_Attribute_Single"

The format of the message is as follows:

Table 122.

Field	Data
Service Code	10_h
Class Code	BE_h
Instance Number	01_h
Attribute ID	03_h or 04_h
Attribute Data	(Assembly number)

9.2.3.2.3. Rejection of set_attribute_single request

If an I/O connection has been established with a master through the Forward_Open request and a successful response, any request to set the selected assembly through the Assembly Instance Selector object is rejected. These attributes may only be changed when no I/O connection is established, i.e. before the Forward_Open request.

9.2.3.3 Motor control mode object, class 0xA1

The Motor Control Mode Object is a vendor-specific object available in the OPTE9 which more clearly indicates to the user which motor control mode is used, and allows the user to configure this mode.

Table 123. Motor control mode object

Class name		Motor control mode object			
Class identifier		161_d / $A1_h$			
Class attributes	Id	Access rule	Name	Datatype	Description
	–	–	–	–	–
Class services	Id	Name			Description
	–	–			–
Instance attributes	Id	Access rule	Name	Datatype	Description
	1	Get/Set	ControlMode	USINT	Motor control mode
	2	Get/Set	FeedbackMode	USINT	Motor feedback mode
Instance services	Id	Name			Description
	14_d $0E_h$	Get_Attribute_Single			Used to read single attribute value.
	16_d 10_h	Set_Attribute_Single			Used to write a single attribute value.

9.2.3.3.1. Instance attributes

Instance attribute “ControlMode”

This attribute is used to detect or change the used motor control mode. The values allowed for this attribute are listed in the table below.

Table 124. Motor control mode values

Value	Description
0_d	Frequency control
1_d	Speed control
2_d	Torque control

Instance attribute “FeedbackMode”

This attribute is used to detect or change the used feedback mode. The values allowed for this attribute are listed in the table below.

After you have changed the feedback mode, you need to set the control mode, too. Only after that, the new feedback mode value is activated.

Table 125. Motor feedback mode values

Value	Description
0_d	Open loop
1_d	Closed loop

ControlMode and FeedbackMode combinations

ControlMode and FeedbackMode combinations depends on used drive and application. Check the actual mode support from the AC drive’s application manual.

9.2.3.3.2. Services

Instance service “Get_Attribute_Single”

The service is used to get the value of an instance attribute.

The format of the message is as follows.

Table 126.

Field	Data
Service code	$0E_h$
Class code	BE_h
Instance number	01_h
Attribute ID	01_h or 02_h

Instance service “Set_Attribute_Single”

The service is used to set the value of an instance attribute.

The format of the message is as follows.

Table 127.

Field	Data
Service code	10_h
Class code	BE_h
Instance number	01_h
Attribute ID	01_h or 02_h
Attribute data	(Mode number)

9.2.3.3.3. Link to AC/DC drive object “drivemode” attribute

The Motor Control Mode Object is linked to the “DriveMode” attribute of the AC/DC Drive Object so, that changes in one affects the values in the other.

In the OPTE9, when the following values are set to the AC/DC Drive Object “DriveMode” attribute, the Motor Control Mode Object attributes are set to the following values:

Table 128.

Set DriveMode value	ControlMode value	FeedbackMode value
0_d (Vendor specific)	0_d (Frequency control)	0_d (Open Loop)
1_d (Open loop speed)	1_d (Speed control)	0_d (Open Loop)
2_d (Closed loop speed)	1_d (Speed control)	1_d (Closed Loop)
3_d (Torque control)	2_d (Torque control)	*

* Feedback mode does not change. CIP does not define is torque control open or closed loop.

In the OPTE9, when the following values are set to the Motor Control Mode Object "ControlMode" and "FeedbackMode" attributes, the AC/DC Drive Object "DriveMode" attribute is set as follows:

Table 129.

Set Motor Control Mode Object values

ControlMode	FeedbackMode	DriveMode value
0_d (Frequency)	0_d (Open loop)	1_d (Vendor/frequency)
1_d (Speed)	0_d (Open loop)	1_d (Open loop speed)
1_d (Speed)	1_d (Closed loop)	2_d (Closed loop speed)
2_d (Torque)	0_d (Open loop) or 1_d (Closed loop)	3_d (Torque control)

9.2.3.4 Fault history object, class 0xA2

The Fault History Object is a vendor-specific object available in the OPTE9 and in Eaton HMX which allows access to the contents of the fault history over the EtherNet/IP network.

Each entry in a fault history is represented by an instance of the Fault History Object. Low instance numbers correspond to the most recent fault entries.

For the OPTE9, the revision of the object is 2.

Table 130. Fault history object

Class name		Fault history object			
Class identifier		162d / A2h			
	Id	**Access rule**	**Name**	**Datatype**	**Description**
Class attributes	1	Get	Revision	UINT	Class revision (2)
	2	Get	Max Instance	UINT	Maximum instance number (40)
	3	Get	Number of Instances	UINT	Number of object instances. Depends on drive type.
	Id	**Name**			**Description**
Class services	1	Get_Attributes_All			Returns content of all (implemented) attributes in the class.
	5	Reset			Reset fault history
	14d 0Eh	Get_Attribute_Single			Used to read single attribute value.
	Id	**Access rule**	**Name**	**Datatype**	**Description**
Instance attributes	1	Get	FaultCode	UINT	Fault code
	2	Get	Fault ID	UINT	Fault ID
	3	Get	Fault year	UINT	Fault date: Year
	4	Get	Fault month	USINT	Fault date: Month
	5	Get	Fault day	UINT	Fault date: Day
	6	Get	Fault millis after midnight	UDINT	Fault date: Time, milliseconds after midnight
	Id	**Name**			**Description**
Instance services	1	Get_Attributes_All			Returns content of all (implemented) attributes in the instance
	14_d $0E_h$	Get_Attribute_Single			Used to read single attribute value

9.2.3.4.1. Changes from revision 1 to revision 2

The attributes FaultCode and FaultDay were changed from 8 bit to 16 bit.

9.2.3.4.2. Class attributes

Class attribute "Revision"

This attribute presents the revision of the Fault History Object which is implemented by the device. The current revision is 2.

Class attribute "Max Instance"

This attribute presents the maximum number of instances that can exist of the Fault History Object in the device. The maximum value is 40.

Class attribute "Number of Instances"

This attribute presents the number of instances that currently exist of the Fault History Object in the device.

Table 131. Fault history object – number of instances

Drive	Maximum number of faults
Eaton MMX	10
Eaton SPX	10
Eaton SPX	10

9.2.3.4.3. Instance attributes

Instance attribute “FaultCode”

The value returns the fault code of a fault entry which is represented by the Fault History Object instance.

Instance attribute “FaultID”

The value returns the fault ID of a fault entry which is represented by the Fault History Object instance. This value more exactly specifies which kind of fault is in question.

In case of SVX/SPX, the FaultID is the subcode of the fault.

Instance attribute “FaultMillisAfterMidnight”

The attribute contains the number of milliseconds after midnight when the fault is time-stamped. The purpose of this format is to provide high accuracy and give you the possibility of representation according to 24-hour or 12-hour clock.

9.2.3.4.4. Class services

Class service “reset”

This service is used to reset the fault history of the drive.

9.3 Assembly instances implemented by OPTE9

9.3.1 ODVA I/O Assembly instances for AC/DC Drive

The OPTE9 supports the Output Assembly Instances 20, 21, 23 and 25, and the Input Assembly Instances 70, 71, 73 and 75.

The control word bits are mapped to the object data according to the table below.

Table 132. CIP Control word mapping to object data

Bit	Bit name	Object	Attribute name	Attribute ID
0	RunFwd	Control supervisor object	Run1	3_d
1	RunRev	Control supervisor object	Run2	4_d
2	FaultRst	Control supervisor object	FaultRst	12_d
3	–	–	–	–
4	–	–	–	–
5	NetCtrl	Control supervisor object	NetCtrl	5_d
6	NetRef	AC/DC drive object	NetRef	4_d
7	NetProc	AC/DC drive object	NetProc	5_d

The status word bits are mapped to the object data according the table below.

Table 133. CIP Status word mapping to object data

Bit	Bit name	Object	Attribute name	Attribute ID
0	Faulted	Control supervisor object	Faulted	10_d
1	Warning	Control supervisor object	Warning	11_d
2	Running1	Control supervisor object	Running1	7_d
3	Running2	Control supervisor object	Running2	8_d
4	Ready	Control supervisor object	Ready	9_d
5	CtrlFromNet	Control supervisor object	CtrlFromNet	15_d
6	RefFromNet	AC/DC drive object	RefFromNet	29_d
7	AtReference	AC/DC drive object	AtReference	3_d

The process data IN is mapped according to the table below.

Table 134. Process data in mapping to object data

Parameter name	Object	Attribute name	Attribute ID
Speed Reference	AC/DC drive object	SpeedRef	8_d
Torque Reference	AC/DC drive object	TorqueRef (ProcessDataIn1)	12_d
Process Reference	AC/DC drive object	ProcessRef If Drive Mode is: 0 = ProcessDataIn1 4 = ProcessDataIn2	14_d
Drive Mode	AC/DC drive object	DriveMode	6_d

The process data OUT is mapped according to the table below.

Table 135. Process data out mapping to object data

Parameter name	Object	Attribute name	Attribute ID
Speed Actual	AC/DC drive object	SpeedActual	7_d
Torque Actual	AC/DC drive object	TorqueActual	11_d
Process Actual	AC/DC drive object	ProcessActual (ProcessDataOut1)	13_d
Drive State	Control supervisor object	State	6_d

The Speed Reference is updated to the drive only when the NetRef bit is set to 1. The torque reference is updated to the drive only when the motor control mode is “Torque Control”.

9.3.1.1 ODVA Output instances

9.3.1.1.1. Assembly instance 20 – basic speed control output

Table 136.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
20 (length 4)	0						Fault Reset		Run Fwd
	1								
	2	Speed Reference (Low Octet)							
	3	Speed Reference (High Octet)							

9.3.1.1.2. Assembly instance 21 – extended speed control output

Table 137.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
21 (length 4)	0		NetRef	NetCtrl			Fault Reset	Run Rev	Run Fwd
	1								
	2	Speed Reference (Low Octet)							
	3	Speed Reference (High Octet)							

9.3.1.1.3. Assembly instance 23 – extended speed and torque control output

Table 138.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
23 (length 6)	0		NetRef	NetCtrl			Fault Reset	Run Rev	Run Fwd
	1								
	2	Speed Reference (Low Octet)							
	3	Speed Reference (High Octet)							
	4	Torque Reference (Low Octet)							
	5	Torque Reference (High Octet)							

9.3.1.1.4. Assembly instance 25 – extended process control output

The extended process control assembly can be used to send process reference value directly to the application. The Process Reference value destination can be selected with the Drive Mode byte according to the table below. This should be configured in the application as the receiving input. To the other direction (input assembly instance 75), the ProcessDataOut1 is always mapped to the Process Actual. Note that the process reference value is sent to the drive only when NetProc bit is set.

The Eaton drives do not support the actual "Process Control"-mode. Therefore the Drive Mode cannot be set to the Process Control by using the AC/DC Drive Object attribute, and the Drive Mode parameter in this assembly is only used to select the process reference. With these mappings, the Process Reference and Process Actual can be used with an application that supports the PID control.

Table 139. Drive mode selection in process control

Drive Mode	Process reference mapping
0	ProcessDataIn1
4	ProcessDataIn2
Other	Not valid

Table 140.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
23 (length 6)	0	Netproc	NetRef	NetCtrl			Fault Reset	Run Rev	Run Fwd
	1	Drive Mode							
	2	Speed Reference (Low Octet)							
	3	Speed Reference (High Octet)							
	4	Process Reference (Low Octet)							
	5	Process Reference (High Octet)							

9.3.1.2 ODVA Input instances

9.3.1.2.1. Assembly instance 70 – basic speed control input

Table 141.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
70 (length 4)	0						Runnin g1		Faulted
	1								
	2	Speed Actual (Low Octet)							
	3	Speed Actual (High Octet)							

9.3.1.2.2. Assembly instance 71 – extended speed control input

Table 142.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
71 (length 4)	0	AtReference	Ref-From-Net	Ctrl-From-Net	Ready	Runnin g2 (Rev)	Runnin g1 (Fwd)	Warning	Faulted
	1	Drive State							
	2	Speed Actual (Low Octet)							
	3	Speed Actual (High Octet)							

9.3.1.2.3. Assembly instance 73 – extended speed and torque control input

Table 143.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
73 (length 6)	0	AtReference	Ref-From-Net	Ctrl-From-Net	Ready	Runnin g2 (Rev)	Runnin g1 (Fwd)	Warning	Faulted
	1	Drive State							
	2	Speed Actual (Low Octet)							
	3	Speed Actual (High Octet)							
	4	Torque Actual (Low Octet)							
	5	Torque Actual (High Octet)							

9.3.1.2.4. Assembly instance 75 – extended process control input

See Chapter "9.3.1.1.4. Assembly Instance 25 – Extended Process Control Output" for details.

Table 144.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
75 (length 6)	0	AtReference	Ref-From-Net	Ctrl-From-Net	Ready	Running2 (Rev)	Running1 (Fwd)	Warning	Faulted
	1	Drive state							
	2	Speed actual (low octet)							
	3	Speed actual (high octet)							
	4	Process actual (low octet)							
	5	Process actual (high octet)							

9.3.2 Vendor-specific I/O assembly instances

The OPTE9 drive supports the assemblies 101, 111, 128, 131, 141, 107, 117, 127, 137 and 147.

The instances 101 and 107 use the CIP control and status words. Others use the vendor specific control and status words. All items started with “FB” are sent directly to the drive without any modifications.

Some of the vendor assemblies use the 32 bit process data values. These assemblies can be used in other drives too, but the data will always be only16 bit.

9.3.2.1 Vendor output instances

9.3.2.1.1. Assembly instance 101

Table 145.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
101 (length 8)	0		NetRef	NetCtrl			Fault Reset	Run Rev	Run Fwd
	1								
	2	FBSpeedReference (Low Octet) in %							
	3	FBSpeedReference (High Octet) in %							
	4	FBProcessDataIn1 (Low Octet)							
	5	FBProcessDataIn1 (High Octet)							
	6	FBProcessDataIn2 (Low Octet)							
	7	FBProcessDataIn2 (High Octet)							

9.3.2.1.2. Assembly instance 111

Table 146.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
111 (length 20)	0	Fbfixedcontrolword (low octet)							
	1	Fbfixedcontrolword (high octet)							
	2	Fbspeedreference (low octet) in %							
	3	Fbspeedreference (high octet) in %							
	4	Fbprocessdatain1 (low octet)							
	5	Fbprocessdatain1 (high octet)							
	6	Fbprocessdatain2 (low octet)							
	7	Fbprocessdatain2 (high octet)							
	8	Fbprocessdatain3 (low octet)							
	9	Fbprocessdatain3 (high octet)							
	10	Fbprocessdatain4 (low octet)							
	11	Fbprocessdatain4 (high octet)							
	12	Fbprocessdatain5 (low octet)							
	13	Fbprocessdatain5 (high octet)							
	14	Fbprocessdatain6 (low octet)							
	15	Fbprocessdatain6 (high octet)							
	16	Fbprocessdatain7 (low octet)							
	17	Fbprocessdatain7 (high octet)							
	18	Fbprocessdatain8 (low octet)							
	19	Fbprocessdatain8 (high octet)							

9.3.2.1.3. Assembly instance 128

Table 147.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
128 (length 20)	0	Fbfixedcontrolword (low octet)							
	1	Fbgeneralcontrolword (high octet)							
	2	Fbspeedreference (low octet) in %							
	3	Fbspeedreference (high octet) in %							
	4	Fbprocessdatain1 (low octet)							
	5	Fbprocessdatain1 (high octet)							
	6	Fbprocessdatain2 (low octet)							
	7	Fbprocessdatain2 (high octet)							
	8	Fbprocessdatain3 (low octet)							
	9	Fbprocessdatain3 (high octet)							
	10	Fbprocessdatain4 (low octet)							
	11	Fbprocessdatain4 (high octet)							
	12	Fbprocessdatain5 (low octet)							
	13	Fbprocessdatain5 (high octet)							
	14	Fbprocessdatain6 (low octet)							
	15	Fbprocessdatain6 (high octet)							
	16	Fbprocessdatain7 (low octet)							
	17	Fbprocessdatain7 (high octet)							
	18	Fbprocessdatain8 (low octet)							
	19	Fbprocessdatain8 (high octet)							

The contents of the assembly are otherwise identical to the output assembly 111, except that the second octet of the assembly is the high octet of FBGeneralControlWord instead of FBFixedControlWord.

9.3.2.1.4. Assembly instance 131

Table 148.

Instance	Offset	Octet 0	Octet +1	Octet +2	Octet +3
131 (length 40)	0	FBFixedControlWord (Low Octet)	FBGeneralConWord (High Octet)	FBFixedControltrolWord (Low Octet)	FBFixedControltrolWord (High Octet)
	4	Reserved	Reserved	FBSpeedRef (Low Octet)	FBSpeedRef (High Octet)
	8	FBProcessDataIn 1 Bits 7:0	FBProcessDataIn 1 Bits 15:8	FBProcessDataIn 1 Bits 23:16	FBProcessDataIn 1 Bits 31:24
	12	FBProcessDataIn 2 Bits 7:0	FBProcessDataIn 2 Bits 15:8	FBProcessDataIn 2 Bits 23:16	FBProcessDataIn 2 Bits 31:24
	16	FBProcessDataIn 3 Bits 7:0	FBProcessDataIn 3 Bits 15:8	FBProcessDataIn 3 Bits 23:16	FBProcessDataIn 3 Bits 31:24
	20	FBProcessDataIn 4 Bits 7:0	FBProcessDataIn 4 Bits 15:8	FBProcessDataIn 4 Bits 23:16	FBProcessDataIn 4 Bits 31:24
	24	FBProcessDataIn 5 Bits 7:0	FBProcessDataIn 5 Bits 15:8	FBProcessDataIn 5 Bits 23:16	FBProcessDataIn 5 Bits 31:24
	28	FBProcessDataIn 6 Bits 7:0	FBProcessDataIn 6 Bits 15:8	FBProcessDataIn 6 Bits 23:16	FBProcessDataIn 6 Bits 31:24
	32	FBProcessDataIn 7 Bits 7:0	FBProcessDataIn 7 Bits 15:8	FBProcessDataIn 7 Bits 23:16	FBProcessDataIn 7 Bits 31:24
	36	FBProcessDataIn 8 Bits 7:0	FBProcessDataIn 8 Bits 15:8	FBProcessDataIn 8 Bits 23:16	FBProcessDataIn 8 Bits 31:24

9.3.2.2 Vendor Input Instances

9.3.2.2.1. Assembly instance 107

Table 149.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
107 (length 8)	0	AtReference	RefFromNet	CtrlFromNet	Ready	Running2 (Rev)	Running1 (Fwd)	Warning	Faulted
	1	Drive State							
	2	FBSpeedActual (Low Octet) in %							
	3	FBSpeedActual (High Octet) in %							
	4	FBProcessDataOut1 (Low Octet)							
	5	FBProcessDataOut1 (High Octet)							
	6	FBProcessDataOut2 (Low Octet)							
	7	FBProcessDataOut2 (High Octet)							

9.3.2.2.2. Assembly instance 117

Table 150.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
117 (length 34)	0	FBFixedStatusWord (Low Octet)							
	1	FBFixedStatusWord (High Octet)							
	2	FBSpeedActual (Low Octet) in %							
	3	FBSpeedActual (High Octet) in %							
	4	RPMSpeedActual (Low Octet) in rpm							
	5	RPMSpeedActual (High Octet) in rpm							
	6	RPM With Slip SpeedActual (Low Octet) in rpm							
	7	RPM With Slip SpeedActual (High Octet) in rpm							
	8	Reserved (=0)							
	...	Reserved (=0)							
	17	Reserved (=0)							
	18	FBProcessDataOut1 (Low Octet)							
	19	FBProcessDataOut1 (High Octet)							
	20	FBProcessDataOut2 (Low Octet)							
	21	FBProcessDataOut2 (High Octet)							
	22	FBProcessDataOut3 (Low Octet)							
	23	FBProcessDataOut3 (High Octet)							
	24	FBProcessDataOut4 (Low Octet)							
	25	FBProcessDataOut4 (High Octet)							
	26	FBProcessDataOut5 (Low Octet)							
	27	FBProcessDataOut5 (High Octet)							
	28	FBProcessDataOut6 (Low Octet)							
	29	FBProcessDataOut6 (High Octet)							
	30	FBProcessDataOut7 (Low Octet)							
	31	FBProcessDataOut7 (High Octet)							
	32	FBProcessDataOut8 (Low Octet)							
	33	FBProcessDataOut8 (High Octet)							

9.3.2.2.3. Assembly instance 127

Table 151.

Instance	Octet	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
127 (length 20)	0	FBFixedStatusWord (Low Octet)							
	1	FBGeneralStatusWord (High Octet)							
	2	FBSpeedActual (Low Octet) in %							
	3	FBSpeedActual (High Octet) in %							
	4	FBProcessDataOut1 (Low Octet)							
	5	FBProcessDataOut1 (High Octet)							
	6	FBProcessDataOut2 (Low Octet)							
	7	FBProcessDataOut2 (High Octet)							
	8	FBProcessDataOut3 (Low Octet)							
	9	FBProcessDataOut3 (High Octet)							
	10	FBProcessDataOut4 (Low Octet)							
	11	FBProcessDataOut4 (High Octet)							
	12	FBProcessDataOut5 (Low Octet)							
	13	FBProcessDataOut5 (High Octet)							
	14	FBProcessDataOut6 (Low Octet)							
	15	FBProcessDataOut6 (High Octet)							
	16	FBProcessDataOut7 (Low Octet)							
	17	FBProcessDataOut7 (High Octet)							
	18	FBProcessDataOut8 (Low Octet)							
	19	FBProcessDataOut8 (High Octet)							

The contents of the assembly are otherwise identical to the input assembly 117, except that the second octet of the assembly is the high octet of FBGeneralStatusWord instead of FBFixedStatusWord.

9.3.2.2.4. Input assembly instance 137

Table 152.

Instance	Offset	Octet 0	Octet +1	Octet +2	Octet +3
137 (length 40)	0	FBFixedStatusWord (Low Octet)	FBFixedStatusWord (High Octet)	FBGeneralStatusWord (Low Octet)	FBGeneralStatus-Word (High Octet)
	4	Reserved	Reserved	FBSpeedActual (Low Octet)	FBSpeedActual (High Octet)
	8	FBProcessDataOut1 Bits 7:0	FBProcessDataOut1 Bits 15:8	FBProcessDataOut1 Bits 23:16	FBProcessDataOut1 Bits 31:24
	12	FBProcessDataOut2 Bits 7:0	FBProcessDataOut2 Bits 15:8	FBProcessDataOut2 Bits 23:16	FBProcessDataOut2 Bits 31:24
	16	FBProcessDataOut3 Bits 7:0	FBProcessDataOut3 Bits 15:8	FBProcessDataOut3 Bits 23:16	FBProcessDataOut3 Bits 31:24
	20	FBProcessDataOut4 Bits 7:0	FBProcessDataOut4 Bits 15:8	FBProcessDataOut4 Bits 23:16	FBProcessDataOut4 Bits 31:24
	24	FBProcessDataOut5 Bits 7:0	FBProcessDataOut5 Bits 15:8	FBProcessDataOut5 Bits 23:16	FBProcessDataOut5 Bits 31:24
	28	FBProcessDataOut6 Bits 7:0	FBProcessDataOut6 Bits 15:8	FBProcessDataOut6 Bits 23:16	FBProcessDataOut6 Bits 31:24
	32	FBProcessDataOut7 Bits 7:0	FBProcessDataOut7 Bits 15:8	FBProcessDataOut7 Bits 23:16	FBProcessDataOut7 Bits 31:24
	36	FBProcessDataOut8 Bits 7:0	FBProcessDataOut8 Bits 15:8	FBProcessDataOut8 Bits 23:16	FBProcessDataOut8 Bits 31:24

9.3.3 Mapping of standard output assemblies onto Eaton data

This section specifies how the data in the Standard Output Assemblies are mapped into Eaton data.

9.3.3.1 FBGeneralControlWord and FBGeneralStatusWord

FBGeneralControlWord and FBGeneralStatusWord are purely application specific. Check the used application's manual for description of their content.

9.3.3.2 FBFixedControlWord

See Chapter 12 "appendix 2 – control and status word".

9.3.3.3 Start/Stop bit in FBFixedControlWord

If one of the "RunFwd" or "RunRev" bits in an Output Assembly has the value 1, the "Start/Stop" bit 0 in the Eaton FBFixedControlWord is set to 1. Otherwise the bit is set to 0.

If both the "RunFwd" and "RunRev" bits have the value 1, no changes are done to FBFixedControlWord.

9.3.3.4 Direction bit in FBFixedControlWord

If the "RunRev" bit in an Output Assembly has the value 1, and the "RunFwd" bit in the assembly has the value 0, then the "Direction" bit 1 in the Eaton FBFixedControlWord is set to 1. Otherwise the bit is set to 0.

9.3.3.5 Fault reset bit in FBFixedControlWord

The "Fault Reset" bit in an Output Assembly is mapped to the "Fault Reset" bit 2 in the Eaton FBFixedControlWord. Both bits are rising-edge sensitive.

9.3.3.6 Request fieldbus control bit in FBFixedControlWord

The "NetCtrl" bit in an Output Assembly is not mapped to the "Request Fieldbus Control" bit 8 in the Eaton FBFixedControlWord because this way you can place the drive to the IO or keypad control while fieldbus is active.

9.3.3.7 Request fieldbus reference bit in FBFixedControlWord

The "NetRef" bit in an Output Assembly is not mapped to the "Request Fieldbus Reference" bit 9 in the Eaton FBFixedControlWord.

9.3.3.8 Master connection state bit in FBFixedControlWord

If the EtherNet/IP communication with the master device is functional, the bit 15 in the Eaton FBFixedControlWord is set to 1. If the communication with the master device is not working, the bit 15 in the FBFixedControlWord is set to 0.

9.3.4 Mapping of data onto standard input assemblies

This section specifies how the Eaton data is mapped onto the data in the Standard Input Assemblies.

9.3.4.1 FBFixedStatusWord

See Chapter 12 "appendix 2 – control and status word".

9.3.4.2 Ready indication bit in FBFixedStatusWord

The "Ready Indication" bit 0 in the Eaton FBFixedStatusWord is mapped to the "Ready" bit in an Input Assembly which supports this bit.

9.3.4.3 Run/Stop indication bit in FBFixedStatusWord

The "Run/Stop indication" bit 1 in the Eaton FBFixedStatusWord is mapped to the "Running1" and "Running2" bits in an Input Assembly which supports these bits. The state of the Running1 and

Running2 bits depends further on the "Direction Indication" bit 2 of the Eaton FBFixedStatusWord as follows:

Table 153. Run/Stop bit indication map

	Run/Stop = 0	Run/Stop = 1	
	Direction = X	**Direction = 0**	**Direction = 1**
Running1	0	1	0
Running2	0	0	1

9.3.4.4 Direction indication bit in FBFixedStatusWord

See chapter 9.3.4.3.

9.3.4.5 Fault indication bit in FBFixedStatusWord

The "Fault Indication" bit 3 in the Eaton FBFixedStatusWord is mapped to the "Faulted" bit in an Input Assembly which supports this bit.

9.3.4.6 Alarm indication bit in FBFixedStatusWord

The "Alarm Indication" bit 4 in the Eaton FBFixedStatusWord is mapped to the "Warning" bit in an input Assembly which supports this bit.

9.3.4.7 Setpoint reached indication bit in FBFixedStatusWord

The "Setpoint Reached Indication" bit 5 in the Eaton FBFixedStatusWord is mapped to the "AtReference" bit in an Input Assembly which supports this bit.

9.3.4.8 Fieldbus control indication in input assemblies

The selected control place is indicated in the Input Assemblies which contain the "CtrlFromNet" bit. If the control place is assigned to fieldbus then this bit is set to 1, else it is 0.

9.3.4.9 Fieldbus reference indication in input assemblies

The selected reference is indicated in Input Assemblies which contain the "RefFromNet" bit. If the reference is assigned to fieldbus then this bit is set to 1, otherwise it is 0.

9.3.4.10 FBSpeedReference in percentage

This is the reference 1 to the AC drive. The allowed scaling is from 0 to 10000. In the application, the value is scaled in percentage of the frequency area between set minimum and maximum frequency.

9.4 EtherNet/IP connection example

Preparing the connection

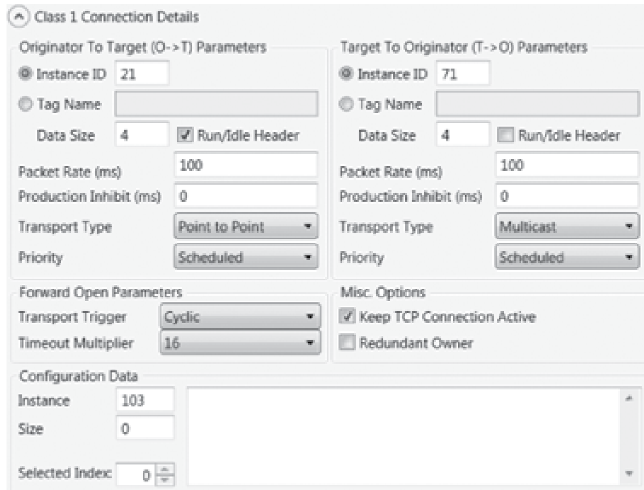
1. Set the EtherNet/IP as the active protocol from the OPTE9 panel parameters
2. Set proper IP addresses
3. Open a connection with the settings described in the table below
4. Before trying to run motor, see Chapter 15 “APPENDIX 5 – FIELDBUS PARAMETRISATION” for information how to configure the drive

Table 154.

Description	Instance	Size
Configuration instance	103 d/67h	0
Output instance	21 d/15h	4
Input instance	71 d/47h	4

1. Set control word to 0x0 (00000000)
2. Set control word to 0x61 (01100001) (NetRef, NetCtrl and Run Fwd enabled)
3. Drive status is: RUN
4. Set speed reference to 0x05EE (=25%)
5. Actual speed is 0x05EE (= 25% if MinFreq is 0Hz and MaxFreq is 50Hz)
6. Set control word 0x61 (01100000)
7. Drive status is: STOP

Figure 33. Configuration example from EIPScan Tool



10. Fault tracing

When the option board or the AC drive control diagnostics detect an unusual operating condition, the drive opens a notification, for example, on the keypad. The keypad shows the ordinal number of the fault, the fault code and a short fault description.

You can reset the fault with the Reset button on the control keypad, via the I/O terminal or via the used fieldbus protocol. The faults are stored in the Fault history menu, which can be browsed. The fault table presents only the fault conditions related to the fieldbus in use.

Note: When you contact a distributor or a factory because of a fault condition, always write down all the texts and codes on the keypad display. Then send the problem description together with the Drive Info File to the following address: TRCDrivestechsupport@eaton.com. If possible, also send a "Wireshark" log from the situation if applicable.

10.1 Typical fault conditions

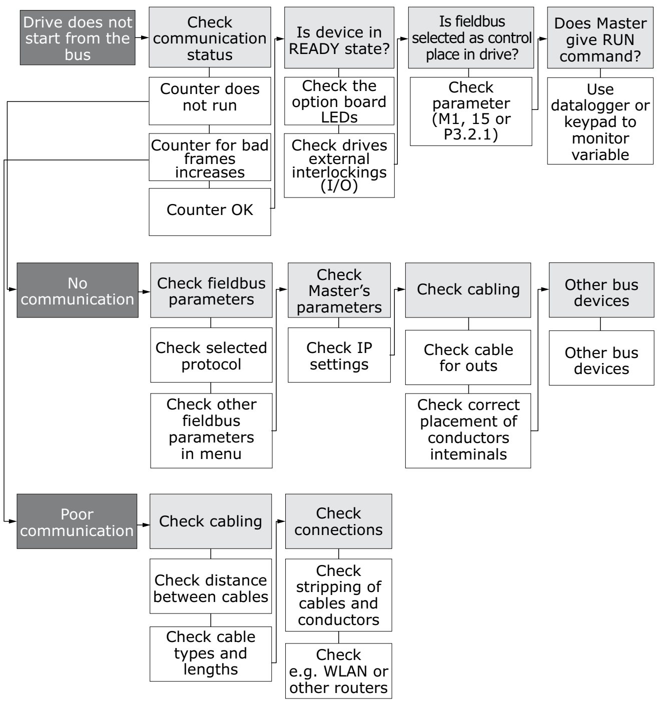
Table 155. Typical fault conditions

Fault condition	Possible cause	Remedy
Cabling	Supply or motor cables are located too close to the fieldbus cable	
	Wrong type of fieldbus cable	
	Too long cabling	
Grounding	Inadequate grounding.	Ensure grounding in all the points on the net.
Connections	Faulty connections: Excessive stripping of cables	
	Faulty connections: Conductors in wrong terminals	
	Faulty connections: Too loose connections of conductors	
Parameter	Faulty address	
	Overlapping slave addresses	
	Wrong control place selected	

10.2 Other fault conditions

The following fault tracing diagram will help you to locate and fix some of the most usual problems. If the problem persists, contact your local distributor.

Figure 34. Fault tracing diagram for OPTE9



11. Appendix 1 – Process data

Process data IN (master to slave)

The use of Process Data In variables depends on the used application. The configuration of the data is free.

Process data OUT (slave to master)

The use of Process Data Out variables depends on the used application. The Fieldbus Master can read the AC drive's actual values using process data variables. The control applications use the process data as follows:

Table 156. Process data OUT variables

ID	Data	Value	Unit	Scale
2104	Process data OUT 1	Output Frequency	Hz	0,01 Hz
2105	Process data OUT 2	Motor Speed	rpm	1 rpm
2106	Process data OUT 3	Motor Current	A	0,1 A
2107	Process data OUT 4	Motor Torque	%	0,10 %
2108	Process data OUT 5	Motor Power	%	0,10 %
2109	Process data OUT 6	Motor Voltage	V	0,1 V
2110	Process data OUT 7	DC link voltage	V	1 V
2111	Process data OUT 8	Active Fault Code	–	–

12. Appendix 2 – Control and status word

12.1 Control word bit description

The Control word is composed of 32 bits. FBFixedControlWord consist of the first 16 bits. FBGeneralControlWord consist of the remaining 16 bits. While the functionality of FBFixedControlWord is fixed in the Eaton standard applications, the functionality of FBGeneralControlWord is totally application specific and can vary even in the standard applications.

The meanings of FBFixedControlWord bits are described below. Unused bits have to be set to zero.

Note: This table is valid for standard applications.

Note: There are some control word bit modifications in SPX drive. These modifications are described in Table 158. Unused bits have to be set to zero.

Table 157. FBFixedControlWord bits

Bit	Function		Description
B0	Start/Stop	0	Stop request from fieldbus.
		1	Run request from fieldbus.
B1	Direction	0	Requested direction is “FORWARD”.
		1	Requested direction is “REVERSE”.
B2	Fault reset	0	No action.
		1	No action. Rising edge (0->1) = Active faults, alarms and infos are reset.
B3	Stop mode 1	0	Stop mode is unmodified.
		1	Stop mode is overridden to “Coasting”.
B4	Stop mode 2	0	Stop mode is unmodified.
		1	Stop mode is overridden to “Ramping”.
B5	Quick ramp time	0	Normal deceleration ramp time.
		1	Deceleration ramp time is switched to shorter than normal.
B6	Freeze Setpoint	0	Changes in the setpoint value from fieldbus (FB Speed Reference) are taken into use by the application.
		1	Changes in the setpoint value from fieldbus (FB Speed Reference) are not taken into use by the application.
B7	Setpoint to Zero	0	The setpoint value from fieldbus is taken from FB Speed Reference.
		1	The setpoint value from fieldbus is changed to 0.
B8	Request Fieldbus Control	0	Control Place is as parameterized in the drive (unchanged).
		1	Control Place is overridden to Fieldbus Control.
B9	Request Fieldbus Reference	0	Source of the setpoint value is as parameterized in the drive (unchanged).
		1	Source of the setpoint value is overridden to Fieldbus.
B10	Jogging 1	0	No action.
		1	Jogging request with ref1.
B11	Jogging 2	0	No action.
		1	Jogging request with ref2.
B12	Quick stop	0	Drive operates as normal.
		1	Drive executes quick stop/emergency stop.
B13	Reserved	0	–
		1	–
B14	Reserved	0	–
		1	–
B15	Reserved	0	–
		1	–

Table 158. FBFixedControWord modifications in Eaton SPX

Bit	Function	Value	Description
3	Fieldbus DIN 1	0	Fieldbus DIN 1 off
		1	Fieldbus DIN 1 on
4	Fieldbus DIN 2	0	Fieldbus DIN 2 off
		1	Fieldbus DIN 2 on
5	Fieldbus DIN 3	0	Fieldbus DIN 3 off
		1	Fieldbus DIN 3 on
6	Fieldbus DIN 4	0	Fieldbus DIN 4 off
		1	Fieldbus DIN 4 on
7	Fieldbus DIN 5	0	Fieldbus DIN 5 off
		1	Fieldbus DIN 5 on

12.2 Status word descriptions

The Status word is composed of 32 bits. FBFixedStatusWord consist of the first 16 bits. FBGeneralStatusWord consist of the remaining 16 bits. While the functionality of FBFixedStatusWord is fixed in the standard applications, the functionality of FBGeneralStatusWord is totally application specific and can vary even in the Eaton standard applications.

The meanings of FBFixedStatusWord bits are described below. Unused bits have to be set to zero.

In SVX/SPX series drives the FBFixedStatusWord comes from firmware variable "MCStatus".

Table 159. FBFixedStatusWord bits

Bit	Function		Description
B0	Ready	0	Drive is not ready.
		1	Drive is ready to run.
B1	Run	0	Motor is not running.
		1	Motor is running.
B2	Direction	0	Motor is running clockwise.
		1	Motor is running counterclockwise.
B3	Fault	0	No fault active.
		1	Drive has an active fault.
B4	Alarm	0	No alarm active.
		1	Drive has active alarm.
B5	At reference	0	Motor is not running at reference speed.
		1	Motor is running at reference speed.
B6	Zero speed	0	Motor is not at zero speed.
		1	Motor is running at zero speed.
B7	Flux ready	0	Motor is not magnetized.
		1	Motor is magnetized.
B8-B12	Reserved		

12.3 Control word bit support in drives

This table describes the control word bit support in different drives. Notice that this table is valid only for the standard applications. Always check the application-specific status from the application manual.

Table 160. Control word

Bit	Function	SVX/SPX
B0	Start/Stop	x
B1	Direction	x
B2	Fault reset	x
B3	Stop mode 1	
B4	Stop mode 2	
B5	Quick ramp time	
B6	Freeze setpoint	
B7	Setpoint to zero	
B8	Request fieldbus control	x
B9	Request fieldbus reference	x
B10	Jogging 1	
B11	Jogging 2	
B12	Quick stop	
B13	Reserved	
B14	Reserved	
B15	Reserved	

12.4 Status word bit support in drives

This table describes the status word bit support in different drives. Notice that this table is valid only for the standard applications. Always check the application-specific status from the application manual.

Table 161. Status word

Bit	Function	SVX/SPX
B0	Ready	x
B1	Run	x
B2	Direction	x
B3	Fault	x
B4	Alarm	x
B5	At reference	x
B6	Zero speed	x
B7	Flux ready	x
B8	Reserved	
B9	Reserved	
B10	Reserved	
B11	Reserved	
B12	Reserved	
B13	Reserved	
B14	Reserved	
B15	Reserved	

13. Appendix 3 – Operating an SVX Drive with a Siemens PLC via PROFINET

Introduction

The purpose of this application note is to demonstrate how to operate and monitor an SVX drive via a PROFINET network with a Siemens Simatic PLC. The OPTE9 dual port, Ethernet option card is used to interface the SVX drive to the PROFINET network.

A GSDML file is required for the OPTE9 option card. This file may be downloaded from: www.eaton.com/drives

The IP Address, subnet mask and gateway address for the SVX drive are configured via the keypad on the drive.

While this application example uses a Siemens Simatic S7-1200 PLC with an integrated PROFINET Ethernet port to control and monitor the SVX drive, any PROFINET master may be used for this purpose. Siemens Simatic Step 7 Basic, V13 programming software was used for this application example. The Simatic S7 PLC will be configured to poll the SVX drive to operate and monitor it.

This document will demonstrate how to configure the PROFINET master to monitor status parameters from the SVX drive as well as to control the drive.

System overview

The devices used for this application example are as follows:

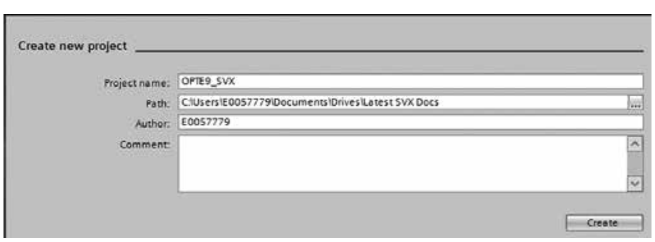
1. OPTE9 dual port Ethernet option card
1. SVX drive
1. Siemens S7 PLC with an integrated PROFINET master port
1. Siemens Totally Integrated Automation Portal software

Creating a project in Siemens Simatic Software

Create a project in Simatic software by starting the software and selecting Create New Project.

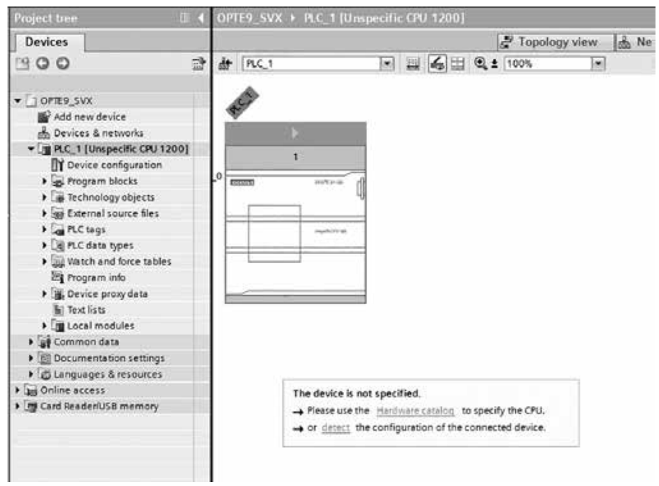
Enter a Project name and Path where the project will be stored, then select the Create button per the fol-lowing:

Figure 35.



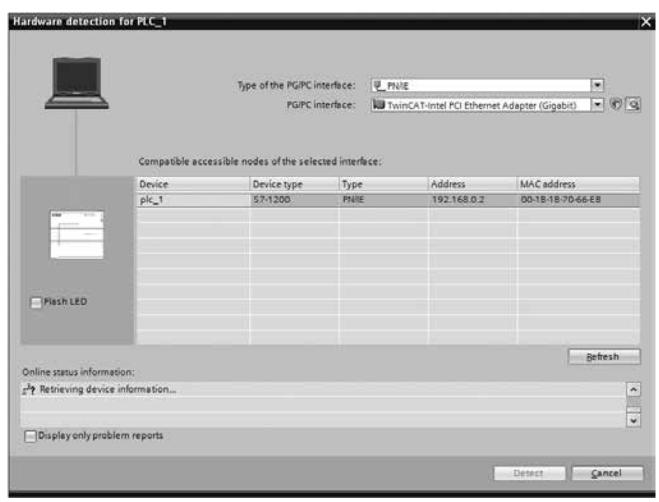
From the next screen, select Configure a device, then select Add new device. An S7-1200 PLC is being used for this application. Select the CPU under Unspecified CPU 1200. Choose the correct version (V3.0 for this example) and select the ADD button. The following Project View will be displayed, showing a generic CPU.

Figure 36.



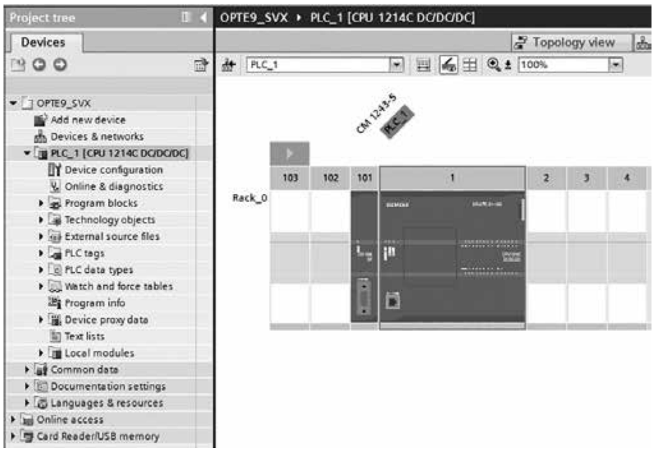
With the computer connected to the PLC via Ethernet, click the CPU box to select it, then select “detect” in the yellow area below it. The Hardware Detection screen will be displayed as follows:

Figure 37.



For this example, the Ethernet port on the computer is being used to communicate with the PLC. It will also be used to upload/download the project later and to poll the SVX drive as well. Set up your computer and software to communicate with your PLC. Once communications is properly set up, select the Detect button and the software will detect the actual controller type as follows:

Figure 38.



Connect the SVX drive (OPTE9 card), the PLC Ethernet port and the computer to the same Ethernet switch. For this application, each device will use the following IP Addresses:

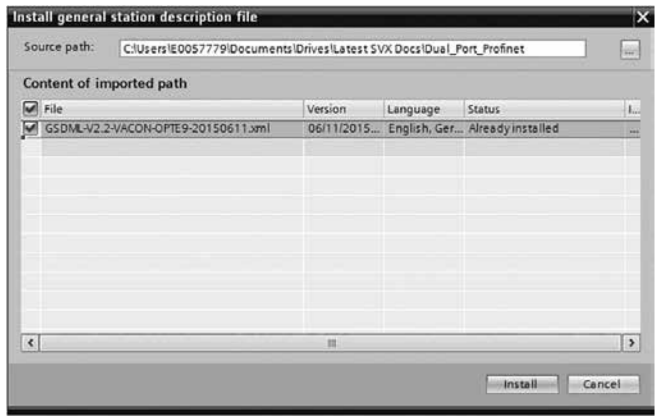
1. Computer 192.168.0.50
2. OPTE9 Option Card 192.168.0.4
3. Siemens PLC PROFINET port 192.168.0.2

The subnet mask is 255.255.255.0 for this example.

Importing the SVX drive GSDML file into Siemens Software

In the Simatic software, select the Options drop down menu and choose: "Install general station descrip-tion file (gsd)". Install the GSDML file downloaded from the Eaton website. To accomplish this, search for it on your hard drive by selecting the ellipses in the upper right hand corner of the following screen:

Figure 39.

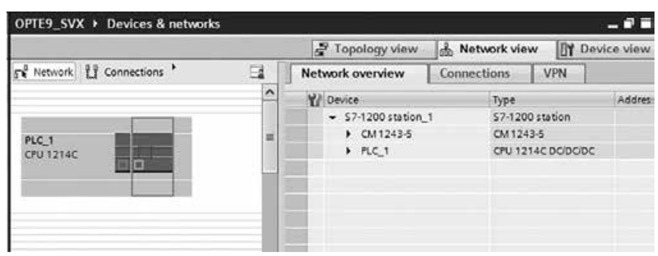


Select the 2 boxes by selecting the box next to File and next to the GSDML file, then select Install and follow the directions to install the GSDML file for the SVX drive.

Configure the PROFINET Network with the Siemens software

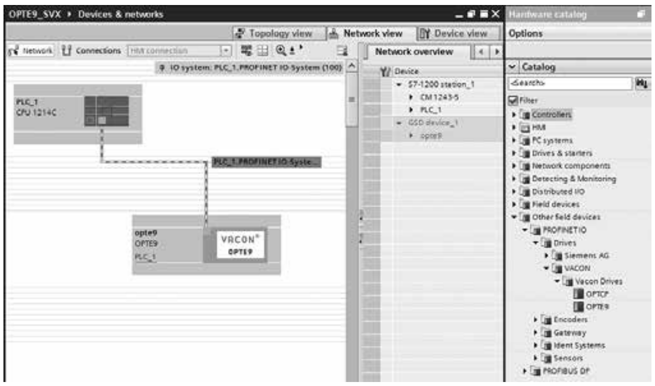
Per the following, select the Network View tab.

Figure 40.



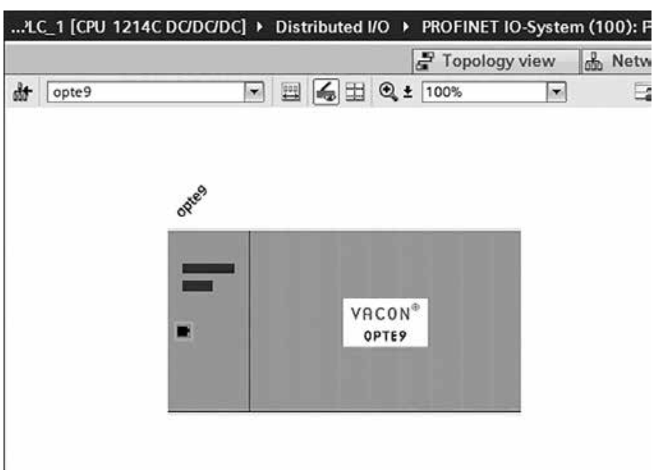
The PLC will be displayed. In the Catalog at the far right, select the arrow next to "Other field devices", then next to PROFINET IO / Drives / VACON/ Vacon Drives/OPTE9. Drag and drop the device called OPTE9 below the controller/ PROFINET master on the Network View screen. Then select the green square on the PROFINET master and drag it to the green square on the OPTE9 and release the mouse button. The following should now be displayed:

Figure 41.



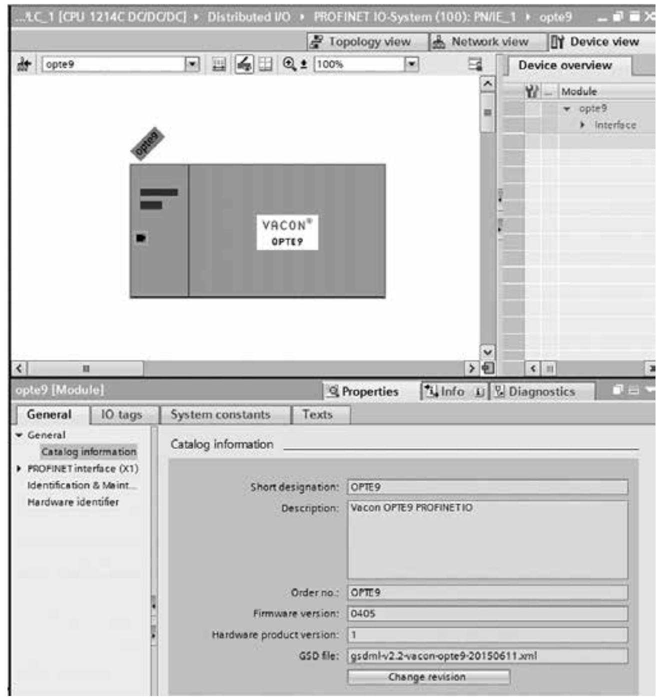
The Profibus network has been created in the offline project. Double click the OPTE9 module and the following will be displayed:

Figure 42.



Double click the OPTE9 box and its Properties will open below it as follows:

Figure 43.

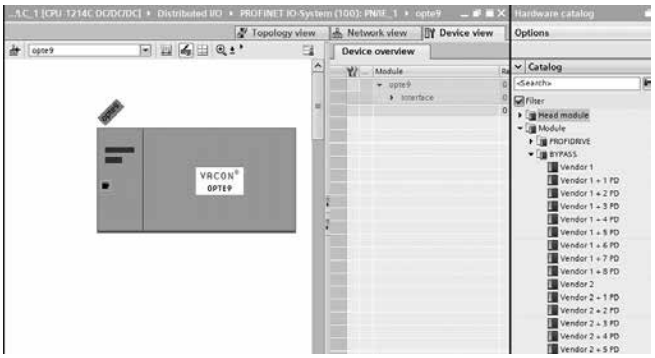


Change the PROFINET IP Address under the "PROFINET interface" category to match the address se-lected for the OPTE9.

Select the I/O for the OPTE9

For this application example, the "PROFIDRIVE/ST1 + 8 PD" will be used for the OPTE9, per the following:

Figure 44.



This I/O selection includes 10 input words and 10 output words.

Input words

Word 1: Status Word
Word 2: Speed Actual
Word 3: Process Data Out 1
Word 4: Process Data Out 2
Word 5: Process Data Out 3
Word 6: Process Data Out 4
Word 7: Process Data Out 5
Word 8: Process Data Out 6
Word 9: Process Data Out 7
Word 10: Process Data Out 8

Default Process Data Out assignments:

Process Data Out 1: Output Frequency (0.01Hz)
Process Data Out 2: Motor Speed (rpm)
Process Data Out 3: Motor Current (0.1 amps)
Process Data Out 4: Motor Torque (0.1%)
Process Data Out 5: Motor Power (0.1%)
Process Data Out 6: Motor Voltage (0.1 volts)
Process Data Out 7: DC Link Voltage (volts)
Process Data Out 8: Active Fault Code

These default values for the Process Data Out words will be used for this application example.

Output words

Word 1: Control Word
Word 2: Speed Reference.

In the Modify value column for QW64 the following 2 hexadecimal values may be used to start and stop the drive:
Start: 0x047F
Stop: 0x047E

Speed Reference range is as follows:

0 represents 0.00 Hz

4000 represents 100% Speed (CW) (60.00 Hz if the Maximum Speed is set to 60.00 Hz)

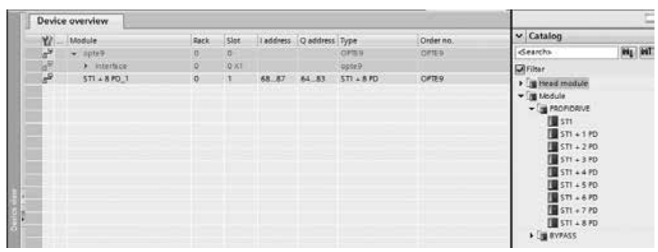
C000 represents -100% Speed (CCW) (-60.00 Hz if the Maximum Speed is set to 60.00 Hz)

The ProcessDataIn words are not used in this application example.

Map the I/O Tags for the OPTE9 option card in the Siemens Programming Software

Grab the "PROFIDRIVE/ST1 + 8 PD"" and drop it under the opte9/Interface/Port_1 as shown below:

Figure 45.



Note: I/O addresses have been allocated for the SVX drive as shown above. This includes 20 bytes or 10 words of input and output data.

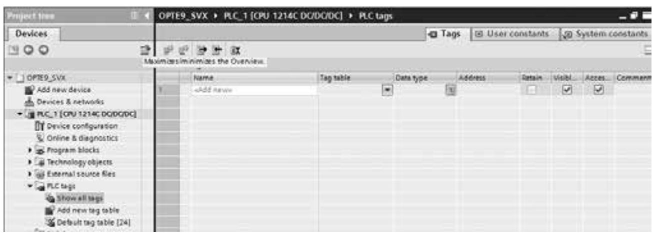
Controlling and Monitoring the SVX Drive with the Siemens PLC

In place of a user program, this example will add the I/O tags for the SVX Drive to the Tag database, then to a Watch List and control the device by modifying the Output tags and monitoring the input data.

Adding I/O tags

Under the Project tree, open "PLC tags" and select "Show all tags" and the following will be displayed:

Figure 46.



Add the I/O tags and their I/O address as assigned by the PLC, per the following:

Figure 47.

PLC tags

	Name	Tag table	Data type	Address	Retain	Visibl...	Acces...
1	Speed_Actual	Default tag table	Word	%IW70	☐	☑	☑
2	Status_Output_Frequency	Default tag table	Word	%IW72	☐	☑	☑
3	Status_Motor_Speed	Default tag table	Word	%IW74	☐	☑	☑
4	Status_Motor_Current	Default tag table	Word	%IW76	☐	☑	☑
5	Status_Motor_Torque	Default tag table	Word	%IW78	☐	☑	☑
6	Status_Motor_Power	Default tag table	Word	%IW80	☐	☑	☑
7	Status_Motor_Voltage	Default tag table	Word	%IW82	☐	☑	☑
8	Status_DC_Link_Voltage	Default tag table	Word	%IW84	☐	☑	☑
9	Status_Active_Fault_Code	Default tag table	Word	%IW86	☐	☑	☑
10	Control_Word	Default tag table	Word	%QW64	☐	☑	☑
11	Speed_Reference	Default tag table	Word	%QW66	☐	☑	☑
12	Status_Word	Default tag table	Word	%IW68	☐	☑	☑

Next, double click "Add new watch table" under "Watch and Force tables" in the Project Tree on the left per the following:

Figure 48.

The new watch table is named "SVX" for this example. Add the I/O addresses for the SWD devices as shown below. Note that the addresses for these tag names that you entered into the PLC Tags area are automatically populated as the tag names are selected. This watch table will allow testing the SVX drive over PROFINET without writing a program. This will allow monitoring the input data from the SVX drive, while also operating it.

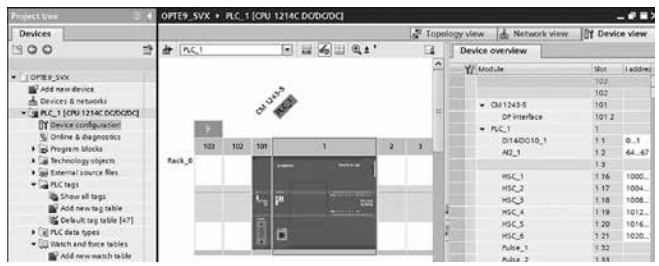
Figure 49.

	i	Name	Address	Display format	Monitor value	Modify value		Co
1		"Status_Word"	%IW68	Hex	16#3737		☐	
2		"Speed_Actual"	%IW70	Hex	16#4000		☐	
3		"Status_Output_Frequency"	%IW72	DEC	6000		☐	
4		"Status_Motor_Speed"	%IW74	DEC	1799		☐	
5		"Status_Motor_Current"	%IW76	DEC	8		☐	
6		"Status_Motor_Torque"	%IW78	DEC	8		☐	
7		"Status_Motor_Power"	%IW80	DEC	8		☐	
8		"Status_Motor_Voltage"	%IW82	DEC	2115		☐	
9		"Status_DC_Link_Voltage"	%IW84	DEC	272		☐	
10		"Status_Active_Fault_Code"	%IW86	DEC	0		☐	
11		"Control_Word"	%QW64	Hex	16#047F	16#047F	☑ !	
12		"Speed_Reference"	%QW66	Hex	16#4000	16#4000	☑ !	

Downloading the program to the Siemens S7 PLC

The project must first be compiled with no errors before it is downloaded to the PLC. In the Project Tree under PLC_1 [CPU….], double click "Device configuration" as follows to display the Device View containing the PLC.

Figure 50.

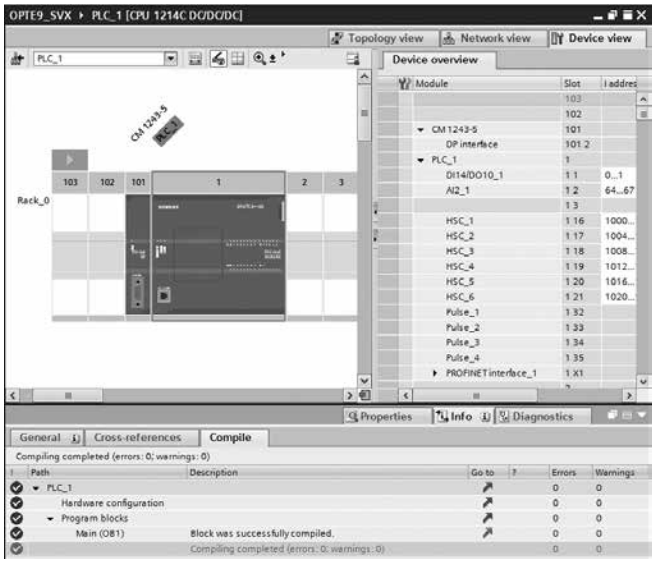


Select the PLC then click the Compile button. The compile button is just to the left of the Download button on the tool bar. Shown below are, from left to right: Compile button, download button and the upload button. As you hover over each of these buttons in the software, it will display its function.

Figure 51.

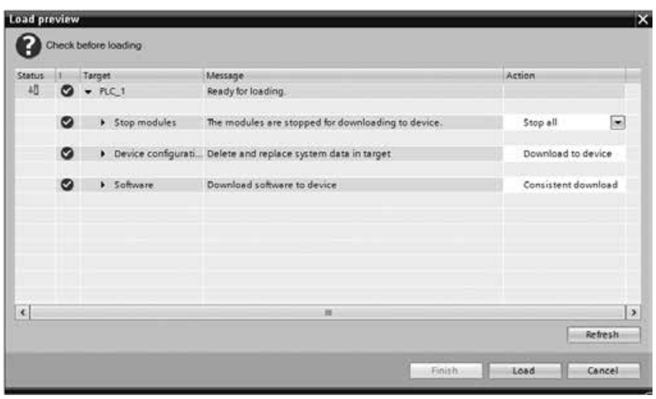
The results of the compile process will be displayed in the area below the PLC as follows:

Figure 52.



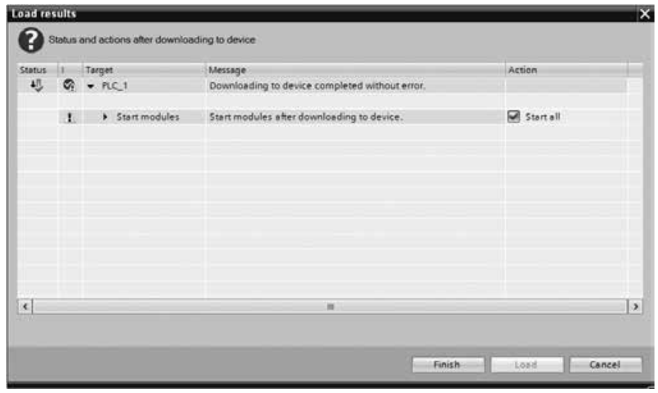
Next, select the download button to download the project to the PLC. The following window will be dis-played. If the controller was in the Run mode, it must be stopped for the download. Select "Stop all" per the following, then select the Load button.

Figure 53.



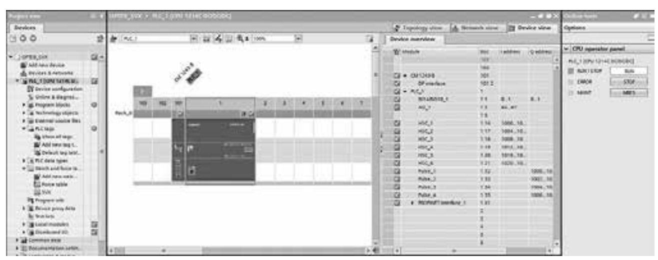
The results of the Load will be displayed in the lower portion of the project screen as shown below. "Start all" should be selected, then select the Finish button. This will complete the download and place the PLC into the Run mode per the following:

Figure 54.



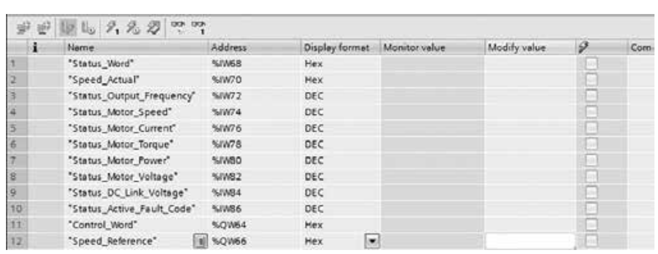
With the PLC selected, select "Go online" from the Tool Bar to go online with the project running in the PLC.
When online, the Simatic software should look like the following:

Figure 55.



In the Project Tree on the left, double click "SVX" under "Watch and force tables" to display the following:

Figure 56.

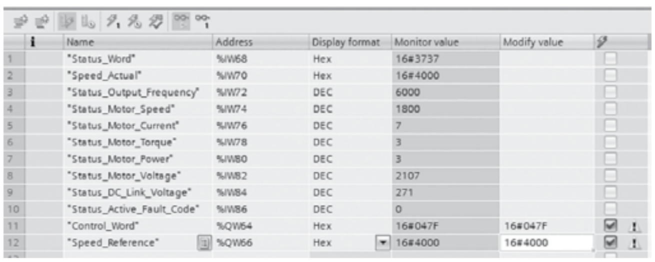


Below is the Tool Bar located above the Watch List.

Figure 57.

If the second icon from the right is selected, the Watch List will begin monitoring and displaying the I/O data as follows:

Figure 58.



	Name	Address	Display format	Monitor value	Modify value
1	"Status_Word"	%IW68	Hex	16#3737	
2	"Speed_Actual"	%IW70	Hex	16#4000	
3	"Status_Output_Frequency"	%IW72	DEC	6000	
4	"Status_Motor_Speed"	%IW74	DEC	1800	
5	"Status_Motor_Current"	%IW76	DEC	7	
6	"Status_Motor_Torque"	%IW78	DEC	3	
7	"Status_Motor_Power"	%IW80	DEC	3	
8	"Status_Motor_Voltage"	%IW82	DEC	2107	
9	"Status_DC_Link_Voltage"	%IW84	DEC	271	
10	"Status_Active_Fault_Code"	%IW86	DEC	0	
11	"Control_Word"	%QW64	Hex	16#047F	16#047F
12	"Speed_Reference"	%QW66	Hex	16#4000	16#4000

In the "Modify value" column, the data for output tags can be modified to energize the SVX drive and send a Speed Reference value.

Start: 0x047F
Stop: 0x047E

Speed Reference:

0 represents 0.00 Hz
4000 represents 100% Speed (CW) (60.00 Hz if the Maximum Speed is set to 60.00 Hz)
C000 represents -100% Speed (CCW) (-60.00 Hz if the Maximum Speed is set to 60.00 Hz)
The data can be entered/viewed in different formats by changing the Display Format for any value.

Each time values are entered or modified in the "Modify Value" column for the Output tags, the lightning bolt with a 1 under it shown below must be selected to instruct the software and the PLC to write the val-ue to the SVX drive via the OPTE9 option card.

Figure 59.

References

SVX Drive User Manual, Publication MN04001004E

SVX Drive Application Manual, Publication MN04004001E

SVX Drive PROFINET Option card (OPTE9) User Manual, Publication XXXXXX.

Additional help

In the US or Canada: please contact the Technical Resource Center at 1-877-ETN-CARE

or 1-877-326-2273 option 2, option 6.

All other supporting documentation is located on the Eaton web site at www.eaton.com/swd

14. Appendix 4 – LWIP licence

License for LWIP

Copyright (c) 2001, 2002 Swedish Institute of Computer Science.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

15. Appendix 5 – Fieldbus parametrisation

The following chapter describes briefly how to parametrise the AC drive in order for the motor to be controllable via fieldbus. These instructions are written for basic applications. For more information, consult the application-specific manual.

In order for the AC drive to accept commands from the fieldbus network, the control place of the AC drive has to be set to fieldbus. The default value of the parameter "Control Place" is usually I/O. Note that if the control unit firmware is updated, the default settings are restored. In addition, some applications may have the remote speed reference selection set by default to other than fieldbus. In these cases, the speed reference selection must be set to fieldbus, in order for the speed reference to be controlled via fieldbus.

Note: The motor control mode should be selected to support the used process and profile.

15.1 Fieldbus control and basic reference selection

The following tables list some of the parameters related to fieldbus control in case of the standard application, the Eaton MMX, for use via fieldbus. See the application specific manuals for more detailed information.

Parameters can be read and written by using the drive panel, PC Tools or fieldbus protocol. Table below contains links to chapters where the ID value reading is described.

Table 162. Links to ID value descriptions

Protocol	Chapter
Modbus TCP	See Chapter "7.1.3.5.1. application IDs" See Chapter "7.1.3.5.4. ID map"
Profinet IO	See Chapter 8.3.3 "Drive parameter access using application ID"
EtherNet/IP	See Chapter 9.2.3.1 "Vendor Parameters Object, Class 0xA0"

Table 163. Parametrization for SVX/SPX (multipurpose application)

Parameter name	ID	Value	Default	Panel tree
Motor control mode	600	0 = Frequency 1 = Speed 2 = Torque	0	P 2.6.1
Control place selection	125	3 = Fieldbus	1	P 3.1
Fieldbus Ctrl Ref.	122	9 = Fieldbus	3	P 2.1.13

15.2 Torque control parametrization

Some extra parametrisation has to be made in order to control the frequency control with torque control. The following instructions are for the Eaton SPX application, see the application-specific manual for more detailed information.

- Motor control mode (ID 600) should be configured to "Torque control" (2)

To configure the drive to use correct torque reference, select the parameter "Torque Reference Selection" to ProcessDataIn1 (9). This can be done with:

- PC-tool or panel
- Vendor Parameter Object

This Page Intentionally Left Blank

This Page Intentionally Left Blank

Eaton
1000 Eaton Boulevard
Cleveland, OH 44122
United States
Eaton.com

© 2017 Eaton
All Rights Reserved
Printed in USA
Publication No.
MN032004EN / CSSC-1611-3741
October 2017

Powering Business Worldwide

Eaton is a registered trademark.

All trademarks are property
of their respective owners.