CANopen - Network Master PLC500

Application Note







Master CANopen - Application Note

Series: PLC500 Language: English Document: 10010165104 / 00

Publication Date: 10/2022



The information below describes the reviews made in this manual.

Version	Revision	Description
-	R00	First edition

Contents

1	INT		5
	1.1		5
	1.2		5
	1.3	NETWORK ARCHITECTURE	5
2	MA	ASTER CANOPEN CONFIGURATION	7
	2.1	STARTING THE PROJECT IN CODESYS	7
3	СА		10
	3.1	NETWORK SETTINGS	10
	3.2	I/O SETTINGS	11
	3.3	DECLARATION OF ADDRESSES	11
	3.4		12
4	MC	ONITORING	13
	4.1		13
	4.2		13
	4.3		14

1 INTRODUCTION

This application manual describes how to operate the PLC500 as a network master. CANopen. For the PLC500 programmable controller settings, it is recommended to follow the steps contained in this document to configure network CANopen properly.



ATTENTION!

This application manual is intended for professionals trained in industrial networks. Devices must be installed and configured according to manufacturer's manual.

1.1 CANOPEN NETWORK COMPONENTS

For the network passive components—cables, connectors, termination resistors, power supply—it is recommended to use only components certified for industrial applications. See the product documentation for information on the proper installation of the CANopen network.

For a deeper and detailed description of the CANopen network operation and its settings, access the online help of **Codesys** at: *https://help.codesys.com*.

1.2 CANOPEN INTERFACE

Figure 1.1 shows the PLC500 with interface CANopen highlighted. The indication of the interface CANopen pins is described in Table 1.1.



Figure 1.1: Indication of the CAN connector pins on the PLC500.

Pin	Name	Lettering	Function
1	COM	-	Common of the CAN network (connected to the negative pole of the network CANopen)
2	CAN_L	L	CAN_L communication signal
3	SHIELD	S	Cable Shield
4	CAN_H	Н	CAN_H communication signal
5	NC	+	Not connected (it can receive the positive pole of network CANopen)
			Table 4.4. lateries - OAN and a serietien

Table 1.1: Interface CANopen description.

1.3 NETWORK ARCHITECTURE

Figure 1.2 shows the components and architecture of network CANopen. This example shows the connection of a PLC500 (network master) and a RUW100 (network slave). The computer programs the devices, with CODESYS in the PLC500 and the WPS for the RUW100.



Figure 1.2: Network components and architecture CANopen.



NOTE!

This application manual is about the PLC500 and the **Codesys** programming tool—we recommend using **Codesys V3.5 SP18 or higher**. If you need more information about the CANopen communication protocol, refer to its manuals.



2 MASTER CANOPEN CONFIGURATION

2.1 STARTING THE PROJECT IN CODESYS

For a network CANopen settings and definitions, you must first create the project and include the PLC500 programmable controller. In the Codesys software, create a new project and choose the directory and name for the application. Then, select the PLC500-WEG device and the desired programming language, as shown in Figure 2.1.

🖹 New Project		×		
Categories	Templates	dard Standard project w		
A project containing one device, one	application, and an empty implementation f	or PLC_PRG	Standard Project You are about to create a new standard project. This wizard will create the follow objects within this project: - One programmable device as specified below - A program PLC_PRG in the language specified below - A cyclic task which calls PLC_PRG - A reference to the newest version of the Standard library currently installed.	ving
Name Example Location C: Users User Documen	Its\CODESYS	Cancel	Device PLC500-WEG (WEG Drives & Controls) PLC_PRG in Ladder Logic Diagram (LD) OK Ca	~ ~

Figure 2.1: Project configuration in Codesys.



NOTE!

In case device PLC500 is not available in the Codesys options yet, you must download and install the configuration file. See the **Product Manual** for the necessary steps and settings.

With the PLC500 device selected, you will have a project with the available network interfaces already preset, as shown in Figure 2.2.



Figure 2.2: PLC500 interfaces.

In the next step, add the device CANopen_Manager, as shown in Fig. 2.3.



Devices		→ ₽ ×	Ú	f Add	Device				×
Example		•							
🖹 🔟 Device (PLC500-WEG))			Name	CANopen_Manager				
🗉 🗐 🖯 PLC Logic				Action					
🔤 Setup (Setup)							- ···		
I_Os (I/Os)				• Ар	pend device O Insert device O Plug d	evice (0 0	odate device	
🗉 🤠 Expansions (Expa	nsio	ns)		String	for a full text search	Vendo	r	<all vendere=""></all>	~
ETH1 (ETH1)				buing				<air vehuors=""></air>	· ·
ETH2 (ETH2)				Nam	e	١	/end	or	Versio ^
CAN (CAN)	¥	Cut		B 👔	Fieldbuses				
🕤 RS485 (RS485)	30 35	Carry		E	CANopen				
	===	сору			CANopenManager				
	B	Paste			CANopen_Manager	3	S - S	mart Software Solutions GmbH	3.5.17
	×	Delete			CANopen_Manager_SIL2	3	S - S	mart Software Solutions GmbH	3.5.17
		Refactoring	•		CANopen_Manager_SoftMotic	n 3	S - S	mart Software Solutions GmbH	3.5.17
		-							
		Properties		E E	- SAE 11939				`
2		Add Object			Biaslaus II.			la) 🗖 Diselsu sutdated useria	-
(2	Add Folder			up by category Display all versions (f	or exper	ts or	liy) Display outdated versio	ns
		Add Device		1	Name: CANopen_Manager	ьн		^	
		Disable Device			Categories: CANopenManager				
		Update Device			Version: 3.5.17.0 Order Number:				5
1	î	Edit Object			Description: CANopen Manager			v ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	
ľ	_	Edit Object With							
				Арре	nd selected device as last child of				
		Edit IO mapping		CAN					
		Import mappings from CSV		0	You can select another target node in the	navigato	or wh	ile this window is open.)	
		Export mappings to CSV							
<	_	>						Add Device	Close
Sevices POUs									

Figure 2.3: CANopen settings Step 1.

In **CANopen_Manager**, add the network slave device CANopen. Figure 2.4 shows the steps described above for this configuration. In this case, RUW100 is selected as an example of a network slave.

Devices -	д х					
Example	-					
😑 🗐 Device (PLC500-WEG)			F	Add Device		×
PLC Logic			-			
E Setup (Setup)			Na	me RUW01		1
I_Os (I/Os)				Action		
Expansions (Expansions)				Appand davice O Insert davice O Dug davice O Undate davice		
ETH1 (ETH1)						- 1
ETH2 (ETH2)			s	String for a full text search Vendor <all vendors=""></all>	`	~
CAN (CAN)			L F			51
CANopen_Manager (CANopen_Manager)	X	Cut		Name	Vendor	
	Bb	Conv		ExiumSD3_SoftMotion	Schneider Electric	
		Pasta		In anotec CANopen DS401 PD4C_SoftMotion	Nanotec Electronic	
		Paste			WEG	
		Delete		SMC14/5_SOTTMOTION	nanotec	
		Refactoring +		SVMCAN CMZ CANopen node_SoftWotion Encoder	CMZ Sistemi Elettro	
	ra.	Properties		UCan 3PhDrive SoftMation	Brunner Elektronik	
	4:=				Stariner Electronic.	-
		Add Object		<	>	
		Add Folder		Group by category 🗌 Display all versions (for experts only) 🗌 Display out	dated versions	
		Add Device		Name: BLW01		=
		Insert Device		Vendor: WEG		
		Scan for Devices		Categories: Remote Device Version: Revision=16#00000078, FileVersion=1.0		
		Disable Device		Order Number: contact WEG	1	
		Update Device		Description: Imported from CO_RUWU1_V12X.eds		
	n° .	Edit Object				= 1
		Edit Object With	4	Append selected device as last child of		
				Anopen_manager		
		Edit IO mapping		(You can select another target node in the navigator while this window is op	en.)	
		Import mappings from CSV				
<		Export mappings to CSV		Add Devi	ce Close	
Services POUs	_		-			

Figure 2.4: CANopen settings Step 2.

The first time some equipment is added as a network slave device CANopen, you must obtain the respective EDS file of the product and import it to Codesys. Install the EDS file in Codesys at: **Tools -> Device Repository** -> **Install**, and select the desired EDS file. All features can be found on the product page at https://www.weg.net.



NOTE!

Electronic Device Description (EDS) is the file needed to describe the slave device to the network CANopen master. The manufacturer must provide this file to allow the correct configuration of this device when the network slave option is available.

At this moment, the CAN interface should have the items indicated in Figure 2.5.



Figure 2.5: CAN interface working as master CANopen.

3 CANOPEN SETTINGS

3.1 NETWORK SETTINGS

The network must be configured on the master to serve the addresses defined on the network slaves. To that end, set the parameters and addresses that will be transmitted from the network master. Each slave device must be configured with a unique address and at the same baud rate; otherwise, the network will experience problems. In figure 3.1, the master device settings are configured.

CAN X CANopen_Ma	anager 🔐 RUW01		
General	General		
Log	Network	0	CAN
CANbus IEC Objects	Baud rate (kbit/s)	250 ~	
Status			
Information			

Figure 3.1: Master CANopenSettings.

The network settings for the slave device are shown in Figure 3.2. Each slave must also have its own address and be within the range of addresses available to slaves. Thus, the settings to start communication between master and slave devices are completed.

General	General	
PDOs	Node-ID 2 SDO Channels	(1/1 Active)
SDOs	Enable expert settings Optional device	
Log	Enable SYNC producing No initialization	Reset node 🗸
CANopen I/O Mapping	⊿ Guarding	
	Enable nodeguarding	Enable heartbeat producing
CANopen IEC Objects	Guard time (ms) 0	Producer time (ms) 200
Status	Life time factor	✓ Heartbeat consuming (1/4 active)
Information	■ Emergency (EMCY)	I TIME
	Enable emergency (EMCY)	Enable TIME producing
	COB-ID 0	COB-ID (Hex) 16# 100
		Enable TIME consuming
	▲ Checks at Startup	
	Check vendor ID Check product number	Check revision number

As this example uses the RUW100 device, the address and rate are directly set on the DIP switches outside the product.



NOTE!

The configuration method may be different for each device, depending on the product type and programming software. Set the slave device according to the manual provided by the manufacturer.

3.2 I/O SETTINGS

Variables and addresses set for a slave must be programmed taking into account the master of that network CANopen. As shown in Figure 3.3, after importing the network CANopen slave into Codesys, the PDO set (Process Data Object) will be available according to each device.

General	Receive PDOs (Master => Slave)		Transmit PDOs (Slave => Master)	
	🕂 Add PDO 🕂 Add Mapping 🖋 Edit 🗙 Delete 🛧 Move Up	Ŧ	🕂 Add PDO 🕂 Add Mapping 🖋 Edit 🗙 Delete 🛧 Move Up	
DOs	Name	^	Name	
00s	✓ 16#1400: Receive PDO Communication Parameter 1		✓ 16#1800: Transmit PDO Communication Parameter 1	
	Write Output 1 to 8		Read Input 1 to 8	
og	Write Output 9 to 16		Read Input 9 to 16	
	16#1401: Receive PDO Communication Parameter 2		✓ 16#1801: Transmit PDO Communication Parameter 2	
ANopen I/O Mapping	Write Output 1 to 16		Read Input 1 to 16	
	16#1402: Receive PDO Communication Parameter 3		✓ 16#1802: Transmit PDO Communication Parameter 3	
ANopen IEC Objects	Write Analogue Output 1		Read Analogue Input 1	
	Write Analogue Output 2		Read Analogue Input 2	
tatus	Write Analogue Output 3		Read Analogue Input 3	
oformation	Write Analogue Output 4		Read Analogue Input 4	
Tornation	16#1403: Receive PDO Communication Parameter 4		✓ 16#1803: Transmit PDO Communication Parameter 4	
	Write Analogue Output 5		Read Analogue Input 5	
	Write Analogue Output 6		Read Analogue Input 6	
	Write Analogue Output 7		Read Analogue Input 7	
	Write Analogue Output 8		Read Analogue Input 8	
	16#1404: Receive PDO Communication Parameter 5		16#1804: Transmit PDO Communication Parameter 5	
	16#1405: Receive PDO Communication Parameter 6	~	16#1805: Transmit PDO Communication Parameter 6	

Figure 3.3: List of PDOs available on the slave device.

For the RUW100 device, we have 32 PDOs for sending and 32 PDOs for receiving data, which can be individually configured to transmit up to 8 bytes of data. These items are described in the RUW100 **User Manual**.



NOTE!

For each application, there may be specific settings and addresses. Thus, the manufacturer must inform the availability of PDOs according to the device. It is recommended to check the specific product manual for more details.

3.3 DECLARATION OF ADDRESSES

With the transmission addresses, you can define which data are relevant for the application. In the example of Figure 3.4, we write some value to the outputs and perform a reading right away.





3.4 TERMINATION RESISTORS

Network CANopen must have resistors installed at its terminations, as they are essential for the correct operation. The configuration of this item is done only through **Codesys**. For the PLC500, go to **setup** and then **Setup Parameters**, where the **Termination Resistors** fields are, as shown in Fig. 3.5.

🔣 Setup 🗙						
Setup Parameters	Parameter	Туре	Value	Default Value	Unit	Description
	🗏 🛄 Firmware					
PLC Setup IEC Objects	Firmware version	STRING	'Not connected'	'Not connected'		PLC's firmware version
Satur I/O Mapping	🖤 🕸 Update available	STRING	'Not connected'	'Not connected'		Firmware version available in PLC for update
Secup 1/0 Happing	🛛 🕸 Update	Enumeration of BYTE	No	No		Select "Yes" to update PLC's firmware with th
	🖻 📴 Date and Time					
	🖤 < Date	STRING	'2021-12-31'	'2021-12-31'		Read PLC date value, use the format '2021-
	🖉 🖉 Time	STRING	'00:00:00'	'00:00:00'		Read PLC time value, use the format '12:59:
	😑 🚞 Termination Resistors					
	🔷 🖗 RS485	Enumeration of BYTE	Not Connected	Not Connected		RS485 termination tesistor configuration
	💮 🖗 CAN	Enumeration of BYTE	Not Connected	Not Connected		CAN termination resistor configuration

Figure 3.5: Configuration page for terminating resistors.



MONITORING 4

VARIABLES MONITORING 4.1

Figure 4.1 shows the Codesys variables view screen for the used network slave.

	CANbus 11 CANop	en_Manager RUW100 🗙	PLC_PRG						
Rede CANopen	General	Find		Filter Show all			🔹 🜵 Add FB for IO Cha	nnel * Go to Instance	
B B CLook		Valable	Manalan	Channel	A	Ture	Current Value	Descend Velue Unit	Description
= O Application [run]	PDOs	valiable	wapping	Channel	Address	iype	Current value	Prepared value Onic	Description
Ibrary Manager		I OUTPUT		Write Output 1 to 8	%QB2	USINT	1		
PLC PRG (PRG)	SDOs	10 Ka		Write Output 9 to 16	%QB3	USINT			
Task Configuration	100			Write Output 1 to 16	76QVV2	UINI			
🖶 😏 🍪 MainTask	LOG			Write Analogue Output 1	76QW5	INT			
PLC_PRG	CANopen I/O Mapping	1. A		Write Analogue Output 2	960W/5	INT			
🕞 🎲 StatusLEDs_Task				Write Analogue Output 4	%OW6	INT			
- 😯 🌆 Setup (Setup)	CANopen IEC Objects	1.5		Write Analogue Output 5	%OW7	INT			
😏 🎹 I_Os (I/Os)		i		Write Analogue Output 6	%OW8	INT			
🖲 😏 🎹 Expansions (Expansions)	Status	8-50		Write Analogue Output 7	%OW9	INT			
😑 🚱 🚮 CANbus (CANbus)	Telesanting	÷		Write Analogue Output 8	%QW10	INT			
🖹 😏 🛐 CANopen_Manager (CANopen_Manager)	Information	H V INPUT	×.	Read Input 1 to 8	%IB34	USINT	0		
- 😏 💮 RUW100 (RUW100)		B-10		Read Input 9 to 16	%IB35	USINT			
		B- 10		Read Input 1 to 16	%IW18	UINT			
		ii*≱		Read Analogue Input 1	%IW19	INT			
		iii - 🎭		Read Analogue Input 2	%IW20	INT			
		10 - 1 9		Read Analogue Input 3	%IW21	INT			
		B- 19		Read Analogue Input 4	%IW22	INT			
		iii		Read Analogue Input 5	%IW23	INT			
		18 - No		Read Analogue Input 6	%IW24	INT			
		10-10		Read Analogue Input 7	%IW25	INT			
		iii- ₩		Read Analogue Input 8	%IW26	INT			
				F	leset Mapping	Alway	s update variables Use p	arent device setting	
		* Create new variable	~	in to existing variable					

The variables declared in the network can be monitored using two methods: first, adding the variables to the Codesys program and monitoring the variable values online; second, enabling Always Update Variables at the bottom of the previous page, as indicated by the red arrow in Figure 4.2.

General	Find		Filter Show all			-	Add FB for IO	Channel → Go to Instance	
PDOs	Variable	Mapping	Channel	Address	Туре	Unit	Description		
1203	🗐 🍢		1. Digital Output 8-Bit	%QB26	USINT				
SDOs	😟 🍫		2. Digital Output 8-Bit	%QB27	USINT				
	±*>		1. Digital Output 16-Bit	%QW14	UINT				
Log	😟 🧤		1. Digital Input 8-Bit	%IB34	USINT				
	¥ø		2. Digital Input 8-Bit	%IB35	USINT				
CANopen I/O Mapping	😟 🍫		1. Digital Input 16-Bit	%IW18	UINT				
CANOPER IEC Objects									
Status							п		
Status Information						1	Г У		
Status			Reset Ma	apping A	lways upda	tevariab	les Enabled 1 (use bus cycle task if not used in any	task) 🗸

Figure 4.2: Monitoring variables in Codesys.

COMMUNICATION ERRORS 4.2

The network status can be monitored in Devices, which indicates the state of each communication step and reports the (Status). When connection problems occur, as shown in Figure 4.3, check that the cables are properly connected, the status of the CAN LED at PLC500, and then review the settings.



Figure 4.3: Indication of error in the communication with the slave device.

Connected to PLC500, access the **Status** and **Log** tabs, shown in Figure 4.4. There, Codesys will inform you which problem it is encountering to perform the communication. When the settings are correct, and the devices identify each other, we should check all other items in green.

Devices 👻 🕂 🗙	RUW01 X	
Example Example Example Example Device (PLC500-WEG) EI PLC Logic Setun (Setun)	General PDOs	CANopen : Last diagnostic message
I_OS (I/Os) Expansions (Expansions) If ETH1 (ETH1) ETH2 (ETH2) CAN (CAN) CANopen_Manager (CANopen_Manager) RUW01 (RUW01)	SDOs	Diag String:
	Log CANopen I/O Mapping	-
	CANopen IEC Objects	
<u>[1</u>]] RS485 (RS485)	Status	
	Information	
Devices POUs		ζ.

Figure 4.4: Status and Log pages contained in each of the items of the communication interfaces.

4.3 CAN INDICATION LED

The PLC500 programmable logic controller has LEDs on the front of the product that are used to indicate the interfaces. The LED color indication shown in Figure 4.5 is described in Table 4.1.



Figure 4.5: LEDs indicadores do PLC500.

DESCRIPTION
There are no communication set- tings CANopen in the application.
Communication CANopen config- ured in the application and all ele- ments running without error/fault.
Communication CANopen config- ured in the application and with some elements containing error/fault.
Communication CANopen config- ured in the application and all ele- ments running without error/fault.

Table 4.1: Operation of the CAN interface indication LED.



WEG Drives & Controls - Automation LTDA. Jaraguá do Sul - SC - Brazil Phone 55 (47) 3276-4000 - Fax 55 (47) 3276-4020 São Paulo - SP - Brazil Phone 55 (11) 5053-2300 - Fax 55 (11) 5052-4212 automacao@weg.net www.weg.net