CANopen

CFW320-CCAN

User's Guide

CANopen User's Guide

Series: CFW320

Language: English

Document: 10009156382 / 00

Build 232

Publication Date: 03/2022

The information below describes the reviews made in this manual.

Version	Revision	Description
-	R00	First edition

ABOUT THE MANUAL

This manual supplies the necessary information for the operation of the CFW320 frequency inverter using the CANopen protocol. This manual must be used together with the CFW320 user's manual and programming manual.

ABBREVIATIONS AND DEFINITIONS

ASCII American Standard Code for Information Interchange
CAN Controller Area Network
CiA CAN in Automation
CIP Common Industrial Protocol
CRC Cycling Redundancy Check
HMI Human-Machine Interface
ISO International Organization for Standardization
ODVA Open DeviceNet Vendor Association
OSI Open Systems Interconnection
PLC Programmable Logic Controller
ro Read only
rw Read/write
RTR Remote Transmission Request

NUMERICAL REPRESENTATION

Decimal numbers are represented by means of digits without suffix. Hexadecimal numbers are represented with the letter 'h' after the number. Binary numbers are represented with the letter 'b' after the number.

DOCUMENTS

The CANopen protocol was developed based on the following specifications and documents:

Document	Version	Source
CAN Specification	2.0	CiA
CiA DS 301 CANopen Application Layer and Communication Profile	4.02	CiA
CiA DRP 303-1 Cabling and Connector Pin Assignment	1.1.1	CiA
CiA DSP 303-3 CANopen Indicator Specification	1.0	CiA
CiA DSP 306 Electronic Data Sheet Specification for CANopen	1.1	CiA
CiA DSP 402 Device Profile Drives and Motion Control	2.0	CiA
Planning and Installation Manual - DeviceNet Cable System	PUB00027R1	ODVA

1 MAIN CHARACTERISTICS

Below are the main characteristics for communication of the frequency inverter CFW320 with CANopen accessory.

- Network management task (NMT).
- 2 transmission PDOs.
- 2 reception PDOs.
- Heartbeat Consumer.
- Heartbeat Producer.
- Node Guarding.
- SDO Client.
- SYNC producer/consumer.
- It is supplied with an EDS file for the network master configuration.
- Acyclic data available for parameterization.

2 CANOPEN COMMUNICATION INTERFACE

The standard CFW320 frequency inverter features a CAN interface. It can be used for communication in Devicenet protocol as a network slave. The characteristics of this interface are described below.

2.1 CAN INTERFACE CHARACTERISTICS

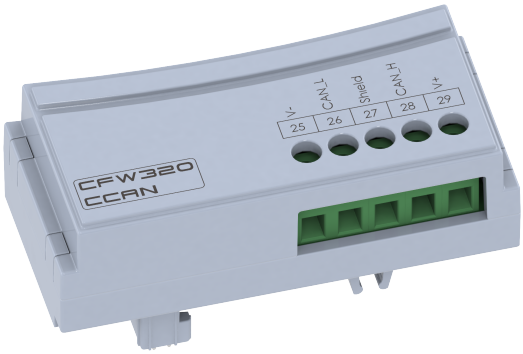


Figure 2.1: *CCAN Accessory*

- Interface galvanically insulated and with differential signal, providing more robustness against electromagnetic interference.
- External power supply of 24 V.
- It allows the connection of up to 64 devices to the same segment. More devices can be connected by using repeaters[1].
- Maximum bus length of 1000 meters.

2.2 PIN ASSIGNMENT OF THE CONNECTOR

The CAN interface has a 5-way connector with the following pin assignment:

Figure 2.2: **CAN connector detail**

Table 2.1: *Pin assignment of connector for CAN interface*

Pin	Name	Function
25	V-	Negative pole of the power supply
26	CAN_L	Communication signal CAN_L
27	Shield	Cable shield
28	CAN_H	Communication signal CAN_H
29	V+	Positive pole of the power supply

2.3 POWER SUPLLY

The CAN interfaces require an external power supply between pins 6 and 10 of the network connector. The data for individual consumption and input voltage are shown in the Table 2.2 on page 9.

[1] The maximum number of devices that can be connected to the network also depends on the protocol used.

Table 2.2: Characteristics of the supply for the CAN interface

Power Suplly (Vdc)		
Minimum	Maximum	Recomended
11	30	24
Current (mA)		
Typical		Maximum
30		50

2.4 INDICATIONS

The alarm, fault and status indications of the CANopen communication for the CFW320 frequency inverter are made trough the HMI and parameters of the product.

3 CANOPEN NETWORK INSTALLATION

The CANopen network, such as several industrial communication networks, for being many times applied in aggressive environments with high exposure to electromagnetic interference, requires that certain precautions be taken in order to guarantee a low communication error rate during its operation. Recommendations to perform the connection of the product in this network are presented next.

NOTE!
Detailed recommendations on how to perform the installation are available at document "Planning and Installation Manual" (item DOCUMENTS).

3.1 BAUD RATE

Equipments with CANopen interface generally allow the configuration of the desired baud rate, ranging from 10 kbit/s to 1 Mbit/s. The baud rate that can be used by the equipment depends on the length of the cable used in the installation. The Table 3.1 on page 10 shows the baud rates and the maximum cable length that can be used in the installation, according to the protocol recommendation.

Table 3.1: *Supported baud rates and cable length*

Baud Rate	Cable length
10 kbit/s	1000 m
20 kbit/s	1000 m
50 kbit/s	1000 m
100 kbit/s	600 m
125 kbit/s	500 m
250 kbit/s	250 m
500 kbit/s	100 m
800 kbit/s	50 m
1 Mbit/s	25 m

All network equipment must be programmed to use the same communication baud rate.

3.2 ADDRESS IN THE CANOPEN NETWORK

Each CANopen network device must have an address or Node-ID, and may range from 1 to 127. This address must be unique for each equipment.

3.3 TERMINATION RESISTOR

The use of termination resistors at the ends of the bus is essential to avoid line reflection, which can impair the signal and cause communication errors. Termination resistors of 121 Ω | 0.25 W must be connected between the signals CAN_H and CAN_L at the ends of the main bus.

3.4 CABLE

The connection of CAN_L and CAN_H signals must be done with shielded twisted pair cable. The Table 3.2 on page 10 shows the recommended characteristics for the cable.

Table 3.2: *CANopen cable characteristics*

Cable Length (m)	Resistence per Meter (mΩ/m)	Conductor Cross Section (mm^2)
0 ... 40	70	0.25 ... 0.34
40 ... 300	<60	0.34 ... 0.60
300 ... 600	<40	0.50 ... 0.60
600 ... 1000	<26	0.75 ... 0.80

It is necessary to use a twisted pair cable to provide additional 24Vdc power supply to equipments that need this signal. It is recommended to use a certified DeviceNet cable.

3.5 CONNECTION TO THE NETWORK

In order to interconnect the several network nodes, it is recommended to connect the equipment directly to the main line without using derivations. During the cable installation the passage near to power cables must be avoided, because, due to electromagnetic interference, this makes the occurrence of transmission errors possible.

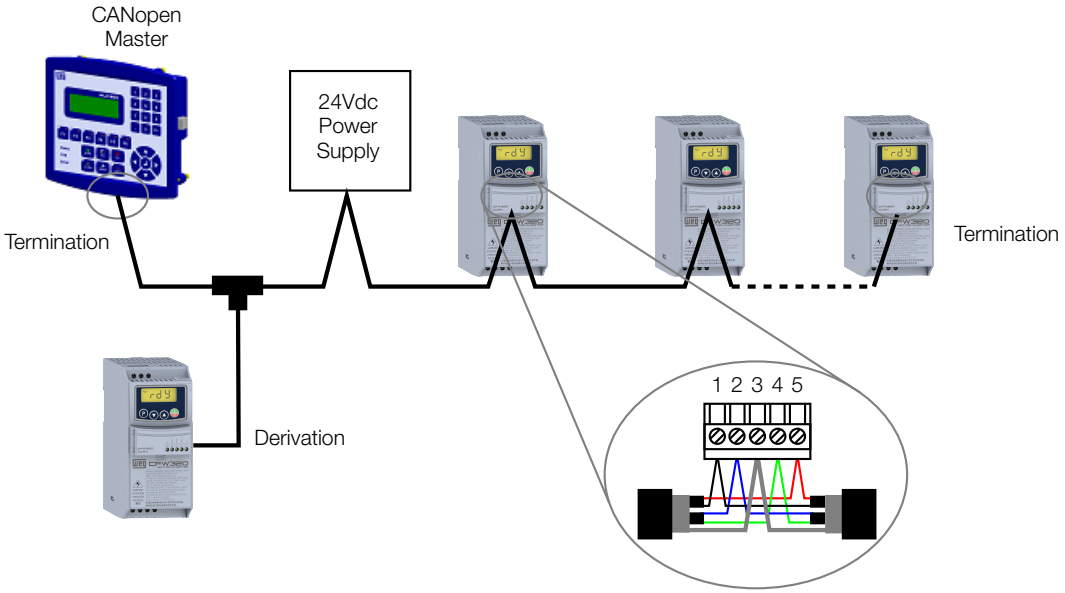


Figure 3.1: *CANopen network installation example*

In order to avoid problems with current circulation caused by difference of potential among ground connections, it is necessary that all the devices be connected to the same ground point.

To avoid voltage difference problems between the power supplies of the network devices, it is recommended that the network is fed by only one power supply and the signal is provided to all devices through the cable. If it is required more than one power supply, these should be referenced to the same point. Use the power supply to power the bus cable system only.

The maximum number of devices connected to a single segment of the network is limited to 64. Repeaters can be used for connecting a bigger number of devices.

4 PARAMETERS

4.1 COMMANDS AND COMMUNICATION STATUS

See below the parameters related to the states and commands through the communication networks available for the frequency inverter.

P313 - Action for Communic. Error

Adjustable Range:	0 = Inactive 1 = Ramp Stop 2 = General Disable 3 = Go to LOC 4 = LOC Keep Enab. 5 = Cause Fault	**Factory Setting:**	1

Description:

It allows the selection of the action to be executed by the device, if it is controlled via network and a communication error is detected.

The following events are considered communication errors:

- Alarm A133/Fault F233: CAN interface not powered.
- Alarm A134/Fault F234: bus off.
- Alarm A135/Fault F235: CANopen communication error (Node Guarding/Heartbeat).

The actions described in this parameter are executed by means of the automatic writing of the selected actions in the respective bits of the interface control words. Therefore, in order that the commands are effective, it is necessary that the device be programmed to be controlled via the used network interface (with exception of option “Causes a Fault”, which blocks the equipment even if it is not controlled by network). This programming is achieved by means of parameters P220 to P228.

Table 4.1: *P313 options*

Indication	Description
0 = Inactive	No action is taken and the drive remains in the existing status.
1 = Ramp Stop	A stop command with deceleration ramp is executed and the motor stops according to the programmed deceleration ramp.
2 = General Disable	The drive is disabled by removing the General Enabling and the motor coasts to stop.
3 = Go to LOC	The drive commands change to Local.
4 = LOC Keep Enab.	The drive commands change to Local, but the status of the enabling and speed reference commands received via network are kept, providing that the drive has been programmed to use in Local mode the commands via HMI, or 3-wire start/stop and speed reference via either HMI or electronic potentiometer.
5 = Cause Fault	Instead of an alarm, the communication error causes a drive fault, so that a drive fault reset becomes necessary in order to restore normal operation.

P680 - Logical Status

Adjustable Range:	0 to FFFF (hexa) Bit 0 = Reserved Bit 1 = Run Command Bit 2 = Fire Mode Bit 3 to 4 = Reserved Bit 5 = 2nd Ramp Bit 6 = Config. Mode Bit 7 = Alarm Bit 8 = Running Bit 9 = Enabled Bit 10 = Forward Bit 11 = JOG Bit 12 = Remote Bit 13 = Subvoltage Bit 14 = Reserved Bit 15 = Fault	**Factory Setting:**	-
Properties:	ro		

Description:

The inverter status word is unique for all the sources and can only be accessed for reading. It indicates all the relevant operating status and modes of the inverter. The function of each bit of P680 is described in Table 4.2 on page 13.

Table 4.2: *P680 bits function*

Bit	Value/Description
Bit 0 Reserved	-
Bit 1 Run Command	**0:** there was no Run command **1:** there was Run command
Bit 2 Fire Mode	**0:** fire Mode function inactive **1:** fire Mode function active
Bit 3 … 4 Reserved	-
Bit 5 2nd Ramp	**0:** 1st acceleration and deceleration ramp by P100 and P101 **1:** 2nd acceleration and deceleration ramp by P102 and P103
Bit 6 Config. Mode	**0:** inverter operating in normal conditions **1:** inverter in configuration state. It indicates a special condition in which the inverter cannot be enabled, because it has parameterization incompatibility
Bit 7 Alarm	**0:** inverter is not in alarm state **1:** inverter is in alarm state
Bit 8 Running	**0:** motor is stopped **1:** inverter is running according to reference and command
Bit 9 Enabled	**0:** inverter is disabled **1:** inverter is enabled and ready to run the motor
Bit 10 Forward	**0:** motor is running in the reverse direction **1:** motor is running in the forward direction
Bit 11 JOG	**0:** JOG function inactive **1:** JOG function active
Bit 12 Remote	**0:** inverter in Local mode **1:** inverter in Remote mode
Bit 13 Subvoltage	**0:** no undervoltage **1:** with undervoltage
Bit 14 Reserved	-
Bit 15 Fault	**0:** inverter is not in fault state **1:** some fault registered by the inverter

P681 - 13-Bit Speed

Adjustable Range:	0 to FFFF (hexa)	**Factory Setting:**	-
Properties:	ro		

Description:

It defines the 13-bit speed reference. The 13-bit Frequency Reference is a scale based on the motor rated speed (P402) or on the motor rated frequency (P403). In the inverter, parameter P403 is taken as the base to determine the frequency reference.

Thus, the 13-bit frequency value has a range of 16 bits with signal, that is, -32768 to 32767; however, the rated frequency in P403 is equivalent to the value 8192. Therefore, the maximum value in the range 32767 is equivalent to four times P403:

- P681 = 0000h (0 decimal) → motor speed = 0
- P681 = 2000h (8192 decimal) → motor speed = rated frequency

Intermediate or higher frequency values can be obtained by using this scale. E.g., for a 60Hz rated frequency motor, if the value read is 2048 (0800h), then, to obtain the value in Hz one must calculate:

8192 => 60 Hz
2048 => Frequency

$$\text{Frequency} = \frac{2048 \times 60}{8192}$$

Frequency = 15 Hz

Negative values in this parameter indicate that the motor is running in the reverse direction.

NOTE!
The values transmitted over the network have a scale limitation, allowing a maximum of 4 times the rated frequency of the motor, with saturation in 32767 (or -32768).

P684 - CO/DN/DP/ETH Control

Adjustable Range:	0 to FFFF (hexa) Bit 0 = Ramp Enable Bit 1 = General Enable Bit 2 = Run Forward Bit 3 = JOG Enable Bit 4 = Remote Bit 5 = 2nd Ramp Bit 6 = Reserved Bit 7 = Fault Reset Bit 8 to 15 = Reserved	**Factory Setting:**	-
Properties:	ro		

Description:

The inverter control word has read and write access only via network interface, but read only access is permitted for the other sources (keypad, SoftPLC). Each bit function is described as per Table 4.3 on page 15. The value of P684 is indicated in hexadecimal.

Table 4.3: P684 bits function

Bit	Value/Description
Bit 0 Ramp Enable	**0**: stops the motor by deceleration ramp **1**: run the motor according to the acceleration ramp until reaching the speed reference value
Bit 1 General Enable	**0**: disables the inverter, interrupting the power supply to the motor **1**: enables the inverter, allowing the operation of the motor
Bit 2 Run Forward	**0**: run the motor in the opposite direction of the reference signal (reverse) **1**: run the motor in the direction of the reference signal (forward)
Bit 3 JOG Enable	**0**: disable JOG function **1**: enable JOG function
Bit 4 Remote	**0**: inverter goes into Local mode **1**: inverter goes into Remote mode
Bit 5 2nd Ramp	**0**: acceleration and deceleration ramp by P100 and P101 **1**: acceleration and deceleration ramp by P102 and P103
Bit 6 Reserved	-
Bit 7 Fault Reset	**0**: no function **1**: if in fault state, reset the fault
Bit 8 ... 15 Reserved	-

P685 - CO/DN/DP/ETH Speed Ref

Adjustable Range:	0 to FFFF (hexa)	**Factory Setting:**	-
Properties:	ro		

Description:

It allows programming the motor speed reference via communication interfaces only. For other sources (HMI, etc.), it behaves as a read-only parameter.

To enable the use of the reference written in this parameter, the product must be programmed to use the speed reference via communication network. This is programming is done using parameters P221 and P222.

This word uses a 13-bit resolution with signal to represent the motor rated frequency (P403):

- P683 = 0000h (0 decimal) → speed reference = 0.
  P683 = 2000h (8192 decimal) → speed reference = rated frequency (P403).
- P685 = 0000h (0 decimal) → speed reference = 0.
  P685 = 2000h (8192 decimal) → speed reference = rated frequency (P403).

Intermediate or higher reference values can be programmed by using this scale. E.g. 60Hz rated frequency, to obtain a speed reference of 30 Hz one must calculate:

60 Hz => 8192
30 Hz => 13 bits reference

$$13 \text{ bits reference} = \frac{30 \times 8192}{60}$$

13 bits reference = 4096 => Value corresponding to 30 Hz in a 13 bit scale

This parameter also accepts negative values to revert the motor speed direction. The reference speed direction, however, depends also on the control word - P684 bit 2 setting:

- Bit 2 = 1 and P685 > 0: reference for forward direction
- Bit 2 = 1 and P685 < 0: reference for reverse direction
- Bit 2 = 0 and P685 > 0: reference for reverse direction
- Bit 2 = 0 and P685 < 0: reference for forward direction

NOTE!
The values transmitted over the network have a scale limitation, allowing a maximum of 4 times the rated frequency of the motor, with saturation in 32767 (or -32768).

P695 - DOx Value

Adjustable Range:	0 to F (hexa) Bit 0 = DO1 Bit 1 = DO2 Bit 2 = DO3 Bit 3 = DO4	**Factory Setting:**	-
Properties:	ro		

Description:

It provides access for monitoring and controlling the inverter by using the communication interfaces. Each bit represents the value for a digital output. The value written in this parameter is used as the digital output value, providing that the function for the desired digital output be programmed for “P695 value”.

Table 4.4: *P695 bits function*

Bit	Value/Description
Bit 0 DO1	**0**: DO1 output open. **1**: DO1 output closed.
Bit 1 DO2	**0**: DO2 output open. **1**: DO2 output closed.
Bit 2 DO3	**0**: DO3 output open. **1**: DO3 output closed.
Bit 3 DO4	**0**: DO4 output open. **1**: DO4 output closed.

P696 - AOx Value 1

P697 - AOx Value 2

Adjustable Range:	0 to FFFF (hexa)	**Factory Setting:**	-
Properties:	ro		

Description:

It provides access for monitoring and controlling the inverter by using the communication interfaces.

They allow the control of the analog outputs by means of network interfaces (Serial, CAN, etc.). These parameters cannot be changed via HMI.

The value written in these parameters is used as the analog output value, providing that the function for the desired analog output be programmed for “P696 / P697 value”, at the parameters P251, P254.

The value must be written in a 15-bit scale (7FFFh = 32767) to represent 100 % of the output desired value, i.e.:

- P696 = 0000h (0 decimal) → analog output value = 0 %
- P696 = 7FFFh (32767 decimal) → analog output value = 100 %

The showed example was for P696, but the same scale is also used for the parameters P697. For instance, to control the analog output 1 via serial, the following programming must be done:

- Choose a parameter from P696, P697 to be the value used by the analog output 1. For this example, we are going to select P696.
- Program the option “P696 value” as the function for the analog output 1 in P254.
- Using the network interface, write in P696 the desired value for the analog output 1, between 0 and 100 %, according to the parameter scale.

NOTE!
If the analog output is programmed for working from -10 V to 10 V, negative values for this parameter must be used to command the output with negative voltage values, i.e., -32768 to 32767 represent a variation from -10 V to 10 V at the analog output.

4.2 CANOPEN

See below the parameters to configure and operate the CAN interface.

P700 - CAN Protocol

Adjustable Range:	1 = CANopen 2 = DeviceNet	**Factory Setting:**	2

Description:

It allows selecting the desired protocol for the CAN interface. If this parameter is changed, the change takes effect only if the CAN interface is not powered, it is in auto-baud or after the equipment is switched off and on again.

Table 4.5: *P700 options*

Indication	Description
1 = CANopen	Enables the CAN interface with CANopen protocol.
2 = DeviceNet	Enables the CAN interface with DeviceNet protocol.

P701 - CAN Address

Adjustable Range:	0 to 127	**Factory Setting:**	63

Description:

It allows programming the address used for the CAN communication. It is necessary that each element of the network has an address different from the others. The valid addresses for this parameter depend on the protocol programmed in P700:

- P700 = 1 (CANopen): valid addresses: 1 to 127.
- P700 = 2 (DeviceNet): valid addresses: 0 to 63.

If this parameter is changed, the change takes effect only if the CAN interface is not powered, auto-baud or after the equipment is switched off and on again.

P702 - CAN Baud Rate

Adjustable Range:	0 = 1 Mbps/Auto 1 = Reserved/Auto 2 = 500 Kbps 3 = 250 Kbps 4 = 125 Kbps 5 = 100 Kbps/Auto 6 = 50 Kbps/Auto 7 = 20 Kbps/Auto 8 = 10 Kbps/Auto	**Factory Setting:**	0

Description:

It allows programming the desired baud rate for the CAN interface, in bits per second. This rate must be the same for all the devices connected to the network. The supported bauld rates for the device depend on the protocol

programmed in the parameter P700:

- P700 = 1 (CANopen): It is possible to use any rate specified in this parameter, but it does not have the automatic baud rate detection function – autobaud.
- P700 = 2 (DeviceNet): only the 500, 250 and 125 Kbit/s rates are supported. Other options will enable the automatic baud rate detection function – autobaud.

If this parameter is changed, the change takes effect only if the CAN interface is not powered or after the equipment is switched off and on again. After a successful detection, the baud rate parameter (P702) changes automatically to the detected rate. In order to execute the autobaud function again, it is necessary to change the parameter P702 to one of the 'Autobaud' options.

Table 4.6: *P702 options*

Indication	Description
0 = 1 Mbps/Auto	CAN baud rate (automatic detection for DeviceNet).
1 = Reserved/Auto	Automatic detection for DeviceNet.
2 = 500 Kbps	CAN baud rate.
3 = 250 Kbps	CAN baud rate.
4 = 125 Kbps	CAN baud rate.
5 = 100 Kbps/Auto	CAN baud rate (automatic detection for DeviceNet).
6 = 50 Kbps/Auto	CAN baud rate (automatic detection for DeviceNet).
7 = 20 Kbps/Auto	CAN baud rate (automatic detection for DeviceNet).
8 = 10 Kbps/Auto	CAN baud rate (automatic detection for DeviceNet).

P703 - Bus Off Reset

Adjustable Range:	0 = Manual 1 = Automatic	**Factory Setting:**	1

Description:

It allows programming the inverter behavior when detecting a bus off error at the CAN interface.

Table 4.7: *P703 options*

Indication	Description
0 = Manual	If bus off occurs, the A134/F234 alarm will be indicated on the HMI, theaction programmed in parameter P313 will be executed and the communication will be disabled. In order that the inverter communicates again through the CAN interface, it will be necessary to cycle the power of the inverter.
1 = Automatic	If bus off occurs, the communication will be reinitiated automatically and the error will be ignored. In this case the alarm will not be indicated on the HMI and the inverter will not execute the action programmed in P313.

P705 - CAN Controller Status

Adjustable Range:	0 = Disabled 1 = Auto-baud 2 = CAN Active 3 = Warning 4 = Error Passive 5 = Bus Off 6 = No Bus Power	**Factory Setting:**	-
Properties:	ro		

Description:

It allows identifying if the CAN interface board is properly installed and if the communication presents errors.

Table 4.8: P705 options

Indication	Description
0 = Disabled	Inactive CAN interface. It occurs when CAN protocol is not programmed at P705.
1 = Auto-baud	CAN controller is trying to detect baud rate of the network (only for CANopen communication protocol).
2 = CAN Active	CAN interface is active and without errors.
3 = Warning	CAN controller has reached the warning state.
4 = Error Passive	CCAN controller has reached the error passive state.
5 = Bus Off	CAN controller has reached the bus off state.
6 = No Bus Power	CAN interface does not have power supply between the pins 25 and 29 of the connector.

P706 - RX CAN Telegrams

Adjustable Range:	0 to 9999	**Factory Setting:**	-
Properties:	ro		

Description:

This parameter works as a cyclic counter that is incremented every time a CAN telegram is received. It informs the operator if the device is being able to communicate with the network. This counter is reset every time the device is switched off, a reset is performed or the parameter maximum limit is reached.

P707 - TX CAN Telegrams

Adjustable Range:	0 to 9999	**Factory Setting:**	-
Properties:	ro		

Description:

This parameter works as a cyclic counter that is incremented every time a CAN telegram is transmitted. It informs the operator if the device is being able to communicate with the network. This counter is reset every time the device is switched off, a reset is performed or the parameter maximum limit is reached.

P708 - Bus Off Counter

Adjustable Range:	0 to 9999	**Factory Setting:**	-
Properties:	ro		

Description:

It is a cyclic counter that indicates the number of times the device entered the bus off state in the CAN network. This counter is reset every time the device is switched off, a reset is performed or the parameter maximum limit is reached.

P709 - CAN Lost Messages

Adjustable Range:	0 to 9999	**Factory Setting:**	-
Properties:	ro		

Description:

It is a cyclic counter that indicates the number of messages received by the CAN interface, but could not be

processed by the device. In case that the number of lost messages is frequently incremented, it is recommended to reduce the baud rate used in the CAN network. This counter is reset every time the device is switched off, a reset is performed or the parameter maximum limit is reached.

P721 - CANopen Comm. Status

Adjustable Range:	0 = Disabled 1 = Reserved 2 = Communic. Enabled 3 = Error Ctrl. Enable 4 = Guarding Error 5 = Heartbeat Error	**Factory Setting:**	-
Properties:	ro		

Description:

It indicates the board state regarding the CANopen network, informing if the protocol has been enabled and if the error control service is active (Node Guarding or Heartbeat).

Table 4.9: *P721 options*

Indication	Description
0 = Disabled	CANopen protocol disabled.
1 = Reserved	-
2 = Communic. Enabled	Communication enabled.
3 = Error Ctrl. Enable	Communication enabled and error control service enabled (Node Guarding/Heartbeat).
4 = Guarding Error	Node Guarding error occurred.
5 = Heartbeat Error	Heartbeat error occurred.

P722 - CANopen Node Status

Adjustable Range:	0 = Disabled 1 = Initialization 2 = Stopped 3 = Operational 4 = Preoperational	**Factory Setting:**	-
Properties:	ro		

Description:

It indicates in which state the device is. It operates as a slave of the CANopen network, and as such element it has a state machine that controls its behavior regarding the communication.

Table 4.10: *P722 options*

Indication	Description
0 = Disabled	CANopen protocol disabled.
1 = Initialization	Communication with the device is not possible during this stage, which is concluded automatically.
2 = Stopped	Only the NMT object is available.
3 = Operational	All the communication objects are available.
4 = Preoperational	It is already possible to communicate with the slave but its PDOs are not yet available for operation.

5 OPERATION IN THE CANOPEN NETWORK

5.1 ACCESS TO THE DATA

Each slave of the CANopen network has a list called object dictionary that contains all the data accessible via network. Each object of this list is identified with an index, which is used during the equipment configuration as well as during message exchanges. This index is used to identify the object being transmitted.

5.2 CYCLIC DATA

Cyclic data is the data normally used for status monitoring and equipment control. For CANopen protocol, the interface supports 2 receive PDOs and 2 transmit PDOs.

It is necessary the configuration to be made both at the slave and master.

5.3 ACYCLIC DATA

In addition to the cyclic data, the interface also provides acyclic data via SDO. Using this type of communication, you can access any equipment parameter. Access to this type of data is commonly done using instructions for reading or writing data, which should indicate the index and sub-index to the desired parameter. The Section 6.4 on page 24 describes how to address the parameters for CFW320 frequency inverter.

5.4 COMMUNICATION OBJECTS - COB

There is a specific set of objects that are responsible for the communication among the network devices. Those objects are divided according to the type of data and the way they are sent or received by a device. The following communication objects (COBs) are described by the specification:

Table 5.1: *Types of Communication Objects (COBs)*

Type of object	Description
Service Data Object (SDO)	SDO are objects responsible for the direct access to the object dictionary of a device. By means of messages using SDO, it is possible to indicate explicitly (by the object index) what data is being handled. There are two SDO types: Client SDO, responsible for doing a read or write request to a network device, and the Server SDO, responsible for taking care of that request. Since SDO are usually used for the configuration of a network node, they have less priority than other types of message.
Process Data Object (PDO)	PDO are used for accessing equipment data without the need of indicating explicitly which dictionary object is being accessed. Therefore, it is necessary to configure previously which data the PDO will be transmitting (data mapping). There are also two types of PDO: Receive PDO and Transmit PDO. They are usually utilized for transmission and reception of data used in the device operation, and for that reason they have higher priority than the SDO.
Emergency Object (EMCY)	This object is responsible for sending messages to indicate the occurrence of errors in the device. When an error occurs in a specific device (EMCY producer), it can send a message to the network. In the case that any network device be monitoring that message (EMCY consumer), it can be programmed so that an action be taken (disabling the other devices, error reset, etc.).
Syncronization Object (SYNC)	In the CANopen network, it is possible to program a device (SYNC producer) to send periodically a synchronization message for all the network devices. Those devices (SYNC consumers) will then be able, for instance, to send a certain datum that needs to be made available periodically.
Network Management (NMT)	Every CANopen network needs a master that controls the other devices (slaves) in the network. This master will be responsible for a set of services that control the slave communications and their state in the CANopen network. The slaves are responsible for receiving the commands sent by the master and for executing the requested actions. The protocol describes two types of service: device control service, with which the master controls the state of each network slave, and error control service (Node Guarding an Heartbeat), with which the device sends periodic messages to inform that the connection is active.

All the communication of the slave with the network is performed using those objects, and the data that can be accessed are the existent in the device object dictionary.

5.5 COB-ID

A telegram of the CANopen network is always transmitted by a communication object (COB). Every COB has an identifier that indicates the type of data that is being transported. This identifier, called COB-ID has an 11 bit size, and it is transmitted in the identifier field of a CAN telegram. It can be subdivided in two parts:

Function Code				Address						
bit 10	bit 9	bit 8	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0

- Function Code: indicates the type of object that is being transmitted.
- Adrress: indicates with which network device the telegram is linked.

The Table 5.2 on page 22 shows the standard values for the different communication objects. Notice that the standard value of the object depends on the slave address, with the exception of the COB-ID for NMT and SYNC, which are common for all the network elements. Those values can also be changed during the device configuration stage.

Table 5.2: *COB-ID for the different objects*

COB	Function Code (bits 10-7)	COB-ID Resultant COB-ID (function + address)
NMT	0000	0
SYNC	0001	128 (80h)
EMCY	0001	129 - 255 (81h - FFh)
PDO1 (tx)	0011	385 - 511 (181h - 1FFh)
PDO1 (rx)	0100	513 - 639 (201h - 27Fh)
PDO2 (tx)	0101	641 - 767 (281h - 2FFh)
PDO2 (rx)	0110	769 - 895 (301h - 37Fh)
PDO3 (tx)	0111	897 - 1023 (381h - 3FFh)
PDO3 (rx)	1000	1025 - 1151 (401h - 47Fh)
PDO4 (tx)	1001	1153 - 1279 (481h - 4FFh)
PDO4 (rx)	1010	1281 - 1407 (501h - 57Fh)
SDO (tx)	1011	1409 - 1535 (581h - 5FFh)
SDO (rx)	1100	1537 - 1663 (601h - 67Fh)
Node Guarding/Heartbeat	1110	1793 - 1919 (701h - 77Fh)

5.6 EDS FILE

Each device on an CANopen network has an EDS configuration file, which contains information about the device functions on the network. This file is used by a master or configuration software to program devices present at CANopen network.

The EDS file is available from WEG website (http://www.weg.net). It is important to note if the EDS configuration file is compatible with the firmware version of the CFW320 frequency inverter.

6 OBJECT DICTIONARY

The object dictionary is a list containing several equipment data which can be accessed via CANopen network. An object of this list is identified by means of a 16-bit index, and it is based in that list that all the data exchange between devices is performed.

The CiA DS 301 document defines a set of minimum objects that every CANopen network slave must have. The objects available in that list are grouped according to the type of function they execute. The objects are arranged in the dictionary in the following manner:

Table 6.1: *Object dictionary groupings*

Index	Objects	Description
0001h - 025Fh	Data type definition	Used as reference for the data type supported by the system.
1000h - 1FFFh	Communication objects	They are objects common to all the CANopen devices. They contain general information about the equipment and also data for the communication configuration.
2000h – 5FFFh	Manufacturer specific objects	In this range, each CANopen equipment manufacturer is free to define which data those objects will represent.
6000h – 9FFFh	Standardized device objects	This range is reserved to objects that describe the behavior of similar equipment, regardless of the manufacturer.

The other indexes that are not referred in this list are reserved for future use.

6.1 DICTIONARY STRUCTRE

The general structure of the dictionary has the following format:

Index	Object	Name	Type	Access

- **Index:** indicates directly the object index in the dictionary.
- **Object:** describes which information the index stores (simple variable, array, record, etc.).
- **Name:** contains the name of the object in order to facilitate its identification.
- **Type:** indicates directly the stored data type. For simple variables, this type may be an integer, a float, etc. For arrays, it indicates the type of data contained in the array. For records, it indicates the record format according to the types described in the first part of the object dictionary (indexes 0001h – 0360h).
- **Access:** informs if the object in question is accessible only for reading (ro), for reading and writing (rw), or if it is a constant (const).

For objects of the array or record type, a sub-index that is not described in the dictionary structure is also necessary.

6.2 DATA TYPE

The first part of the object dictionary (index 0001h – 025Fh) describes the data types that can be accessed at a CANopen network device. They can be basic types, as integers and floats, or compound types formed by a set of entries, as records and arrays.

6.3 COMMUNICATION PROFILE - COMMUNICATION OBJECTS

The indexes from 1000h to 1FFFh in the object dictionary correspond to the part responsible for the CANopen network communication configuration. Those objects are common to all the devices, however only a few are obligatory. A list with the objects of this range that are supported by the frequency inverter CFW320 is presented next.

Table 6.2: Object list – Communication Profile

Index	Object	Name	Type	Access
1000h	VAR	device type	UNSIGNED32	ro
1001h	VAR	error register	UNSIGNED8	ro
1005h	VAR	COB-ID SYNC	UNSIGNED32	rw
100Ch	VAR	quard time	UNSIGNED16	rw
100Dh	VAR	life time factor	UNSIGNED8	rw
1016h	ARRAY	consume heartbeat time	UNSIGNED32	rw
1017h	VAR	producer heartbeat time	UNSIGNED16	rw
1018h	RECORD	Identity Object	Identity	ro
Server SDO Parameter				
1200h	RECORD	1st Server SDO parameter	SDO Parameter	ro
Receive PDO Communication Parameter				
1400h	RECORD	1st receive PDO Parameter	PDO CommPar	rw
1401h	RECORD	2nd receive PDO Parameter	PDO CommPar	rw
Receive PDO Mapping Parameter				
1600	RECORD	1st receive PDO mapping	PDO Mapping	rw
1601	RECORD	2st receive PDO mapping	PDO Mapping	rw
Transmit PDO Communication Parameter				
1800h	RECORD	1st transmit PDO Parameter	PDO CommPar	rw
1801h	RECORD	2st transmit PDO Parameter	PDO CommPar	rw
Transmit PDO Mapping Parameter				
1A00h	RECORD	1st transmit PDO mapping	PDO Mapping	rw
1A01h	RECORD	1st transmit PDO mapping	PDO Mapping	rw

These objects can only be read and written via the CANopen network, it is not available via the keypad (HMI) or other network interface. The network master, in general, is the equipment responsible for setting up the equipment before starting the operation. The EDS configuration file brings the list of all supported communication objects.

Refer to Section 7 on page 25 for more details on the available objects in this range of the objects dictionary.

6.4 MANUFACTURER SPECIFIC – CFW320 SPECIFIC OBJECTS

For indexes from 2000h to 5FFFh, each manufacture is free to define which objects will be present, and also the type and function of each one. In the case of the CFW320, the whole list of parameters was made available in this object range. It is possible to operate the CFW320 by means of these parameters, carrying out any function that the inverter can execute. The parameters were made available starting from the index 2000h, and by adding their number to this index their position in the dictionary is obtained. The Table 6.3 on page 24 illustrates how the parameters are distributed in the object dictionary.

Table 6.3: CFW320 object list – Manufacturer Specific

Índice	Objeto	Nombre	Tipo	Acceso
2001h	VAR	P001 – Speed reference	INTEGER16	ro
2002h	VAR	P002 – Motor speed	INTEGER16	ro
2003h	VAR	P003 – Motor current	INTEGER16	ro
2004h	VAR	P004 – DC voltage	INTEGER16	ro
...	...	...	...	...
2064h	VAR	P100 – Acceleration time	INTEGER16	rw
2065h	VAR	P101 – Deceleration time	INTEGER16	rw
...	...	...	...	...

Refer to the CFW320 manual for a complete list of the parameters and their detailed description. In order to be able to program the inverter operation correctly via the CANopen network, it is necessary to know its operation through the parameters.

7 COMMUNICATION OBJECTS DESCRIPTION

This item describes in detail each of the communication objects available for the CFW320 frequency inverter. It is necessary to know how to operate these objects to be able to use the available functions for the inverter communication.

7.1 IDENTIFICATION OBJECT

There is a set of objects in the dictionary which are used for equipment identification; however, they do not have influence on their behavior in the CANopen network.

7.1.1 Objeto 1000h - Device Type

This object gives a 32-bit code that describes the type of object and its functionality.

Table 7.1: *Objeto 1000h - Device Type*

Index	Sub-index	Name	Type	Access	PDO Mapping	Value
1000h	0	Device Type	UNSIGNED32	RO	No	0

This code can be divided into two parts: 16 low-order bits describing the type of profile that the device uses, and 16 high-order bits indicating a specific function according to the specified profile.

7.1.2 Objeto 1001h - Error Register

This object indicates whether or not an error in the device occurred. The type of error registered for the frequency inverter follows what is described in the Table 7.2 on page 25.

Table 7.2: *Objeto 1001h - Error Register*

Index	Sub-index	Name	Type	Access	PDO Mapping	Value
1001h	0	Error register	UNSIGNED8	RO	yes	0

Table 7.3: *Structure of the object Error Register*

Bit	Meaning
0	Generic error
1	Currrent
2	Voltage
3	Temperature
4	Communication
5	Reservaded (always 0)
6	Reservaded (always 0)
7	Specific of the manufacturer

If the device presents any error, the equivalent bit must be activated. The first bit (generic error) must be activated with any error condition.

7.1.3 Objeto 1018h - Identity Object

It brings general information about the device.

Table 7.4: Objeto 1018h - Identity Object

Index	Sub-index	Name	Type	Access	PDO Mapping	Value
1018h	0	Number of the last sub-index	UNSIGNED8	RO	No	4
	1	Vendor ID	UNSIGNED32	RO	No	0000.0123h
	2	Product code	UNSIGNED32	RO	No	0000.0A00h
	3	Revision number	UNSIGNED32	RO	No	According to the equipment firmware version
	4	Serial number	UNSIGNED32	RO	No	Different for every CFW320

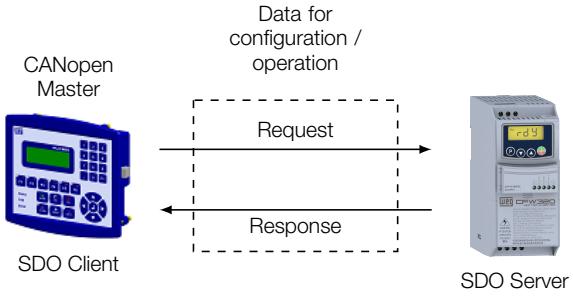
The vendor ID is the number that identifies the manufacturer at the CiA. The product code is defined by the manufacturer according to the type of product. The revision number represents the equipment firmware version. The sub-index 4 is a unique serial number for each frequency inverter CFW320 in CANopen network.

7.2 SERVICE DATA OBJECTS - SDOS

The SDOs are responsible for the direct access to the object dictionary of a specific device in the network. They are used for the configuration and therefore have low priority, since they do not have to be used for communicating data necessary for the device operation.

There are two types of SDOs: client and server. Basically, the communication initiates with the client (usually the master of the network) making a read (upload) or write (download) request to a server, and then this server answers the request.

Figure 7.1: Communication between SDO client and server



7.2.1 Objeto 1200h - SDO Server

The frequency inverter CFW320 has only one SDO of the server type, which makes it possible the access to its entire object dictionary. Through it, an SDO client can configure the communication, the parameters and the drive operation. Every SDO server has an object, of the SDO_PARAMETER type, for its configuration, having the following structure:

Table 7.5: Objeto 1200h - Servidor SDO

Index	Sub-index	Name	Type	Access	PDO Mapping	Value
1200h	0	Number of the last sub-index	UNSIGNED8	RO	No	2
	1	COB-ID Cliente - Server (rx)	UNSIGNED32	RO	No	600h + Node-ID
	2	COB-ID Servidor - Client (tx)	UNSIGNED32	RO	No	580h + Node-ID

7.2.2 SDOs Operation

A telegram sent by an SDO has an 8 byte size, with the following structure:

Identifier	8 data bytes							
11 bits	Command	Index		Subindex	Object data			
	byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7

The identifier depends on the transmission direction (rx or tx) and on the address (or Node-ID) of the destination server. For instance, a client that makes a request to a server which Node-ID is 1, must send a message with the identifier 601h. The server will receive this message and answer with a telegram which COB-ID is equal to 581h.

The command code depends on the used function type. For the transmissions from a client to a server, the following commands can be used:

Table 7.6: *Command codes for SDO client*

Command	Function	Description	Object Data
22h	Download	Write object	Not defined
23h	Download	Write object	4 byte
2Bh	Download	Write object	2 byte
2Fh	Download	Write object	1 byte
40h	Upload	Read object	Not used
60h ou 70h	Upload segment	Segmented read	Not used

When making a request, the client will indicate through its COB-ID, the address of the slave to which this request is destined. Only a slave (using its respective SDO server) will be able to answer the received telegram to the client. The answer telegram will have also the same structure of the request telegram, the commands however are different:

Table 7.7: *Command codes for SDO server*

Command	Function	Description	Object Data
60h	Download	Write object	Not used
43h	Upload	Write object	4 byte
4Bh	Upload	Write object	2 byte
4Fh	Upload	Write object	1 byte
41h	Upload segment	Initiates segmented response for read	4 byte
01h ou 0Dh	Upload segment	Last data segment for read	8 ... 2 bytes

For readings of up to four data bytes, a single message can be transmitted by the server; for the reading of a bigger quantity of bytes, it is necessary that the client and the server exchange multiple telegrams.

A telegram is only completed after the acknowledgement of the server to the request of the client. If any error is detected during telegram exchanges (for instance, no answer from the server), the client will be able to abort the process by means of a warning message with the command code equal to 80h.

NOTE!
When the SDO is used for writing in objects that represent the CFW320 parameters (objects starting from the index 2000h), this value is saved in the nonvolatile frequency inverter memory. Therefore, the configured values are not lost after the equipment is switched off or reset. For all the other objects these values are not saved automatically, so that it is necessary to rewrite the desired values.

E.g.: A client SDO requests for a CFW320 at address 1 the reading of the object identified by the index 2000h, sub-index 0 (zero), which represents an 16-bit integer. The master telegram has the following format:

Identifier	Command	Index		Subindex	Data			
601h	40h	00h	20h	00h	00h	00h	00h	00h

The CFW320 responds to the request indicating that the value of the referred object is equal to 999 [2]:

Identifier	Command	Index		Subindex	Data			
581h	4Bh	00h	20h	00h	E7h	03h	00h	00h

[2] Do not forget that for any integer type of data, the byte transfer order is from the least significant to the most significant.

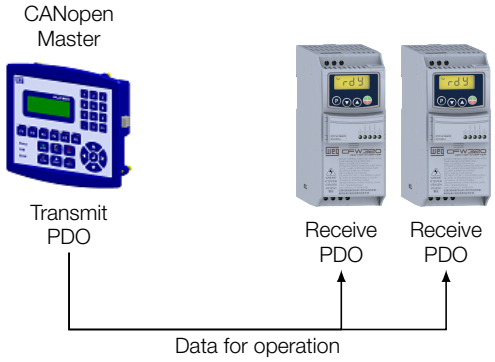
7.3 PROCESS DATA OBJECTS - PDOS

The PDOs are used to send and receive data used during the device operation, which must often be transmitted in a fast and efficient manner. Therefore, they have a higher priority than the SDOs.

In the PDOs only data are transmitted in the telegram (index and sub-index are omitted), and in this way it is possible to do a more efficient transmission, with larger volume of data in a single telegram. However it is necessary to configure previously what is being transmitted by the PDO, so that even without the indication of the index and sub-index, it is possible to know the content of the telegram.

There are two types of PDOs, the receive PDO and the transmit PDO. The transmit PDOs are responsible for sending data to the network, whereas the receive PDOs remain responsible for receiving and handling these data. In this way it is possible to have communication among slaves of the CANopen network, it is only necessary to configure one slave to transmit information and one or more slaves to receive this information.

Figure 7.2: *Communication using PDOs*



NOTE!
PDOs can only be transmitted or received when the device is in the operational state.

7.3.1 PDO Mapping Objects

In order to be able to be transmitted by a PDO, it is necessary that an object be mapped into this PDO content. In the description of communication objects (1000h – 1FFFh), the filed “PDO Mapping” informs this possibility. Usually only information necessary for the operation of the device can be mapped, such as enabling commands, device status, reference, etc. Information on the device configuration are not accessible through PDOs, and if it is necessary to access them one must use the SDOs.

For CFW320 specific objects (2000h – 5FFFh), the Table 7.8 on page 29 presents some PDO mapping objects. Read-only parameters (ro) can be used only by transmit PDOs, whereas the other parameters can be used only by receive PDOs. The CFW320 EDS file brings the list of all the objects available for the inverter, informing whether the object can be mapped or not.

Table 7.8: Examples of PDO mapping parameters

Index	Object	Name	Type	Access
2002h	VAR	P002 – Motor speed	UNSIGNED16	ro
2003h	VAR	P003 – Motor current	UNSIGNED16	ro
2005h	VAR	P005 – Motor frequency	UNSIGNED16	ro
2006h	VAR	P006 – Inverter status	UNSIGNED16	ro
2007h	VAR	P007 – Output voltage	UNSIGNED16	ro
2009h	VAR	P009 – Motor torque	INTEGER16	ro
200Ch	VAR	P012 – DI1 to DI8 status	UNSIGNED16	ro
2012h	VAR	P018 – AI1 value	INTEGER16	ro
2064h	VAR	P100 – Acceleration time	UNSIGNED16	rw
2065h	VAR	P101 – Deceleration time	UNSIGNED16	rw
22A8h	VAR	P680 – Logical status	UNSIGNED16	ro
22A9h	VAR	P681 – Motor speed in 13 bits	INTEGER16	ro
22ACh	VAR	P684 – Control CANopen/DNet	UNSIGNED16	rw
22ADh	VAR	P685 – Speed reference CANopen/DNet	INTEGER16	rw

The EDS file brings the list of all available objects informing whether the object can be mapped or not.

7.3.2 Receive PDOs

The receive PDOs, or RPDOs, are responsible for receiving data that other devices send to the CANopen network. The frequency inverter CFW320 has receive PDOs, each one being able to receive up to 8 bytes. Each RPDO has two parameters for its configuration, a PDO_COMM_PARAMETER and a PDO_MAPPING, as described next.

PDO_COMM_PARAMETER

Index	Sub index	Name	Type	Access	PDO Mapping	Value
1400h até 1401h	0	Number of the last sub-index	UNSIGNED8	RO	No	2
	1	COB-ID used by the PDO	UNSIGNED32	RW	No	200h/300h + Node-ID
	2	Transmission Type	UNSIGNED8	RW	No	254

The sub-index 1 contains the receive PDO COB-ID. Every time a message is sent to the network, this object will read the COB-ID of that message and, if it is equal to the value of this field, the message will be received by the device. This field is formed by an UNSIGNED32 with the following structure:

Table 7.9: COB-ID description

Bit	Value	Description
31 (MSB)	0	PDO is enabled
	1	PDO is disabled
30	0	RTR permitted
29	0	Identifier size = 11 bits
28 - 11	0	Not used, always 0
10 - 0 (LSB)	X	11-bit COB-ID

The bit 31 allows enabling or disabling the PDO. The bits 29 and 30 must be kept in 0 (zero), they indicate respectively that the PDO accepts remote frames (RTR frames) and that it uses an 11-bit identifier. Since the CFW320 frequency inverter does not use 29-bit identifiers, the bits from 28 to 11 must be kept in 0 (zero), whereas the bits from 10 to 0 (zero) are used to configure the COB-ID for the PDO.

The sub-index 2 indicates the transmission type of this object, according to the Table 7.10 on page 30.

Table 7.10: Transmission type description

	Type of transmission	PDOs transmission				
		Cyclic	Acyclic	Synchronous	Asynchronous	RTR
0	0		•	•		
	1 - 240	•		•		
	241 - 251	Reserved				
	252			•		•
	253				•	•
	254				•	
	255				•	

- **Values 0 – 240:** any RPDO programmed in this range presents the same performance. When detecting a message, it will receive the data; however it won't update the received values until detecting the next SYNC telegram.
- **Values 252 e 253:** not allowed for receive PDOs.
- **Values 254 e 255:** they indicated that there is no relationship with the synchronization object. When receiving a message, its values are updated immediately.

PDO_MAPPING

Index	Sub index	Name	Type	Access	PDO Mapping	Value
1600h até 1601h	0	Number of mapped objects	0 = disable 1-4=number of mapped objects	RO	No	0
	1 up to	1 up to object mapped in the PDO	UNSIGNED32	RW	No	According EDS file

This parameter indicates the mapped objects in the CFW320 receive PDOs. It is possible to map up to different objects for each RPDO, provided that the total length does not exceed eight bytes. The mapping of an object is done indicating its index, sub-index [3] and size (in bits) in an UNSIGNED32, field with the following format:

UNSIGNED32		
Index (16 bits)	Sub index (8 bits)	Objects size (8 bits)

For instance, analyzing the receive PDO standard mapping, we have:

- **Sub index 0 = 2:** This PDO has two mapped objects.
- **Sub index 1 = 22AC.0010h:** the first mapped object has an index equal to 22A8h, sub-index 0 (zero), and a size of 16 bits. This object corresponds to the parameter P684 that is inverter status.
- **Sub index 2 = 22AD.0010h:** the second mapped object has an index equal to 22A9h, sub- index 0 (zero), and a size of 16 bits. This object corresponds to the parameter P685 that is motor speed.

It is possible to modify this mapping by changing the quantity or the number of mapped objects. Remembering that only 4 objects or 8 bytes can be mapped at maximum.

NOTE!
- In order to change the mapped objects in a PDO, it is first necessary to write the value 0 (zero) in the sub-index 0 (zero). In that way the values of the sub-indexes 1 to can be changed. After the desired mapping has been done, one must write again in the sub-index 0 (zero) the number of objects that have been mapped, enabling again the PDO.
- Do not forget that PDOs can only be received if the CFW320 is in the operational state.

[3] If the object is of the VAR type and does not have sub-index, the value 0 (zero) must be indicated for the sub-index.

7.3.3 Transmit PDOs

The transmit PDOs, or TPDOs, as the name says, are responsible for transmitting data for the CANopen network. The frequency inverter CFW320 has 2 transmit PDOs, each one being able to transmit up to 8 data bytes. In a manner similar to RPDOs, each TPDO has two parameters for its configuration, a PDO_COMM_PARAMETER and a PDO_MAPPING, as described next.

PDO_COMM_PARAMETER

Index	Subinde	Name	Type	Access	PDO Mapping	Value
1800h-1801h	0	Number of the last sub-index	UNSIGNED8	RO	No	5
	1	COB-ID used by the PDO	UNSIGNED32	RW	No	180h/280h + Node-ID
	2	Transmission Type	UNSIGNED8	RW	No	254
	3	Time between transmissions	UNSIGNED16	RW	No	-
	4	Reserved	UNSIGNED8	RW	No	-
	5	Event timer	UNSIGNED16	RW	No	0

The sub-index 1 contains the transmit PDO COB-ID. Every time this PDO sends a message to the network, the identifier of that message will be this COB-ID. The structure of this field is described in Table 7.9 on page 29.

The sub-index 2 indicates the transmission type of this object, which follows the Table 7.10 on page 30 description. Its working is however different for transmit PDOs:

- **Value 0:**indicates that the transmission must occur immediately after the reception of a SYNC telegram, but not periodically.
- **Values 1 – 240:**the PDO must be transmitted at each detected SYNC telegram (or multiple occurrences of SYNC, according to the number chosen between 1 and 240).
- **Value 252:**indicates that the message content must be updated (but not sent) after the reception of a SYNC telegram. The transmission of the message must be done after the reception of a remote frame (RTR frame).
- **Value 253:**the PDO must update and send a message as soon as it receives a remote frame.
- **Value 254:**The object must be transmitted according to the timer programmed in sub-index 5.
- **Value 255:**the object is transmitted automatically when the value of any of the objects mapped in this PDO is changed. It works by changing the state (Change of State). This type does also allow that the PDO be transmitted according to the timer programmed in sub-index 5.

In the sub-index 3 it is possible to program a minimum time (in multiples of 100µs) that must elapse after the a telegram has been sent, so that a new one can be sent by this PDO. The value 0 (zero) disables this function.

The sub-index 5 contains a value to enable a timer for the automatic sending of a PDO. Therefore, whenever a PDO is configured as the asynchronous type, it is possible to program the value of this timer (in multiples of 1ms), so that the PDO is transmitted periodically in the programmed time.

NOTE!
- The value of this timer must be programmed according to the used transmission rate. Very short times (close to the transmission time of the telegram) are able to monopolize the bus, causing indefinite retransmission of the PDO, and avoiding that other less priority objects transmit their data
- The minimum time allowed for this Function in the frequency inverter CFW320 is 2ms.
- It is important to observe the time between transmissions programmed in the sub-index 3, especially when the PDO is programmed with the value 255 in the sub-index 2 (Change of State).
- Do not forget that PDOs can only be received if the CFW320 is in the operational state.

PDO_MAPPING

Index	Subinde	Name	Type	Access	PDO Mapping	Value
1A00h-1A01h	0	Number of the last sub-index	UNSIGNED8	RO	No	0
	1 - 4	1 up to 4 object mapped in the PDO	UNSIGNED32	RW	No	0

The PDO MAPPING for the transmission works in similar way than for the reception, however in this case the data to be transmitted by the PDO are defined. Each mapped object must be put in the list according to the description showed next:

UNSIGNED32		
Index (16 bits)	Subindex (8 bits)	Object size (8 bits)

For instance, analyzing the standard mapping of the fourth transmit PDO, we have:

- **Sub-índice 0 = 2:** This PDO has two mapped objects.
- **Sub-índice 1 = 22A8.0010h:** the first mapped object has an index equal to 22A8h, sub-index 0 (zero), and a size of 16 bits. This object corresponds to the parameter P680 that is inverter status.
- **Sub-índice 2 = 22A9.0010h:** the second mapped object has an index equal to 22A9h, sub- index 0 (zero), and a size of 16 bits. This object corresponds to the parameter P681 that is motor speed.

It is possible to modify this mapping by changing the quantity or the number of mapped objects. Remember that a maximum of 4 objects or 8 bytes can be mapped.

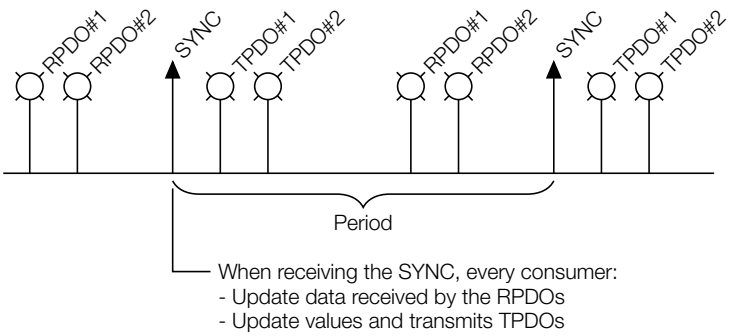
NOTE!
In order to change the mapped objects in a PDO, it is first necessary to write the value 0 (zero) in the sub-index 0 (zero). In that way the values of the sub-indexes 1 to 4 can be changed. After the desired mapping has been done, one must write again in the sub-index 0 (zero) the number of objects that have been mapped, enabling again the PDO.

7.4 SYNCHRONIZATION OBJECT - SYNC

This object is transmitted with the purpose of allowing the synchronization of events among the CANopen network devices. It is transmitted by a SYNC producer, and the devices that detect its transmission are named SYNC consumers

The frequency inverter CFW320 has the function of a SYNC consumer and, therefore, it can program its PDOs to be synchronous. Synchronous PDOs are those related to the synchronization object, thus they can be programmed to be transmitted or updated based in this object.

Figure 7.3: *SYNC*



The SYNC message transmitted by the producer does not have any data in its data field, because its purpose is to provide a time base for the other objects. There is an object in the CFW320 for the configuration of the COB-ID of the SYNC consumer.

Index	Subinde	Name	Type	Access	PDO Mapping	Value
1005h	0	COB-ID SYNC	UNSIGNED32	RW	No	80h

NOTE!
The period of the SYNC telegrams must be programmed in the producer according to the transmission rate and the number of synchronous PDOs to be transmitted. There must be enough time for the transmission of these objects, and it is also recommended that there is a tolerance to make it possible the transmission of asynchronous messages, such as EMCY, asynchronous PDOs and SDOs.

7.5 NETWORK MANAGEMENT - NMT

The network management object is responsible for a series of services that control the communication of the device in a CANopen network. For the CFW320 the services of node control and error control are available (using Node Guarding or Heartbeat).

7.5.1 Slave State Control

With respect to the communication, a CANopen network device can be described by the following state machine:

Figure 7.4: *CANopen node state diagram*

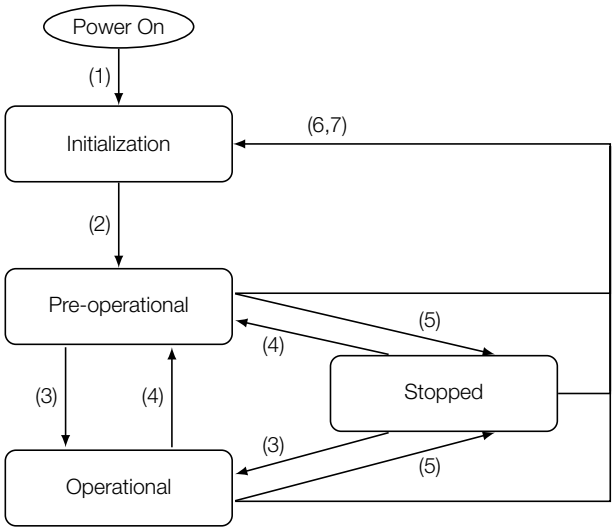


Table 7.11: *Transitions Description*

Transition	Description
1	The device is switched on and initiates the initialization (automatic)
2	Initialization concluded, it goes to the preoperational state (automatic)
3	It receives the Start Node command for entering the operational state
4	It receives the Enter Pre-Operational command, and goes to the preoperational state
5	It receives the Stop Node command for entering the stopped state
6	It receives the Reset Node command, when it executes the device complete reset
7	It receives the Reset Communication command, when it reinitializes the object values and the CANopen device communication

During the initialization the Node-ID is defined, the objects are created and the interface with the CAN network is configured. Communication with the device is not possible during this stage, which is concluded automatically. At the end of this stage the slave sends to the network a telegram of the Boot-up Object, used only to indicate that the initialization has been concluded and that the slave has entered the preoperational state. This telegram has the identifier 700h + Node-ID, and only one data byte with value equal to 0 (zero).

In the preoperational state it is already possible to communicate with the slave, but its PDOs are not yet available for operation. In the operational state all the objects are available, whereas in the stopped state only the NMT object can receive or transmit telegrams to the network. The Table 7.12 on page 34 shows the objects available for each state.

Table 7.12: Objects accessible in each state

	Initialization	Préoperational	Operational	Stopped
PDO			•	
SDO		•	•	
SYNC		•	•	
EMCY		•	•	
Boot-up	•			
NMT		•	•	•

This state machine is controlled by the network master, which sends to each slave the commands so that the desired state change be executed. These telegrams do not have confirmation, what means that the slave does only receive the telegram without returning an answer to the master. The received telegrams have the following structure:

Identifier	byte 1	byte 2
00h	Command Code	Destination Node-ID

Table 7.13: Commands for the state transition

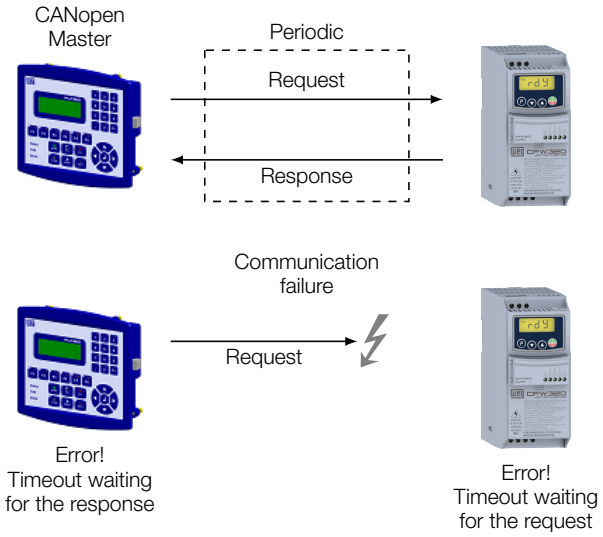
Command Code	Destination Node ID
1 = START node (transition 3) 2 = STOP node (transition 4) 128 = Enter preoperational (transition 5) 129 = Reset node (transition 6) 130 = Reset comunication (transition 7)	0 = All the slaves 1 ... 127 = Specific slave

The transitions indicated in the command code correspond to the state transitions executed by the node after receiving the command (according to the Figure 7.4 on page 33). The Reset node command makes the CFW320 execute a complete reset of the device, while the Reset communication command causes the device to reinitialize only the objects pertinent to the CANopen communication.

7.5.2 Error Control - Node Guarding

This service is used to make it possible the monitoring of the communication with the CANopen network, both by the master and the slave as well. In this type of service the master sends periodical telegrams to the slave, which responds to the received telegram. If some error that interrupts the communication occurs, it will be possible to identify this error, because the master as well as the slave will be notified by the Timeout in the execution of this service. The error events are called Node Guarding for the master and Life Guarding for the slave.

Figure 7.5: Error control service – Node Guarding



There are two objects of the dictionary for the configuration of the error detection times for the Node Guarding service:

Index	Sub index	Name	Type	Access	PDO Mapping	Value
100Ch	0	Guard Time	UNSIGNED32	RW	No	0

Index	Sub index	Name	Type	Access	PDO Mapping	Value
100Dh	0	Life Time Factor	UNSIGNED8	RW	No	0

The 100Ch object allows programming the time necessary (in milliseconds) for a fault occurrence being detected, in case the CFW320 does not receive any telegram from the master. The 100Dh object indicates how many faults in sequence are necessary until it be considered that there was really a communication error. Therefore, the multiplication of these two values will result in the total necessary time for the communication error detection using this object. The value 0 (zero) disables this function.

Once configured, the CFW320 starts counting these times starting from the first Node Guarding telegram received from the network master. The master telegram is of the remote type, not having data bytes. The identifier is equal to 700h + Node-ID of the destination slave. However the slave response telegram has 1 data byte with the following structure:

Identificador	byte 1	
	bit 7	bit 6 ... 0
700h + Node ID	Toogle	Estado do Escravo

This telegram has one single data byte. This byte contains, in the seven least significant bits, a value to indicate the slave state (4 = stopped, 5 = operational and 127 = preoperational), and in the eighth bit, a value that must be changed at every telegram sent by the slave (toggle bit).

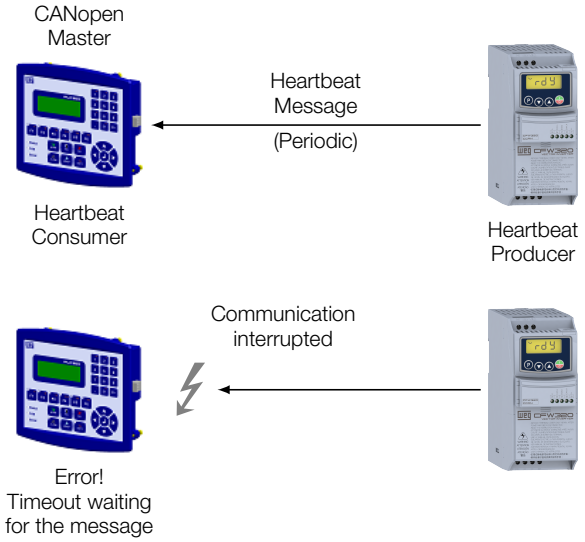
NOTE!

- This object is active even in the stopped state (see Table 7.12 on page 34).
- The value 0 (zero) in any of these two objects will disable this function.
- If after the error detection the service is enabled again, then the error indication will be removed from the HMI.
- The minimum value accepted by the frequency inverter CFW320 is 2ms, but considering the transmission rate and the number of nodes in the network, the times programmed for this function must be consistent, so that there is enough time for the transmission of the telegrams and also that the rest of the communication be able to be processed.
- For any every slave only one of the two services - Heartbeat or Node Guarding – can be enabled.

7.5.3 Error Control - Heartbeat

The error detection through the Heartbeat mechanism is done using two types of objects: the Heartbeat producer and the Heartbeat consumer. The producer is responsible for sending periodic telegrams to the network, simulating a heartbeat, indicating that the communication is active and without errors. One or more consumers can monitor these periodic telegrams, and if they cease occurring, it means that any communication problem occurred.

Figure 7.6: Error control service – Heartbeat



One device of the network can be both producer and consumer of heartbeat messages. For example, the network master can consume messages sent by a slave, making it possible to detect communication problems with the master, and simultaneously the slave can consume heartbeat messages sent by the master, also making it possible to the slave detect communication fault with the master.

The CFW320 has the producer and consumer of heartbeat services. As a consumer, it is possible to program up to 4 different producers to be monitored by the inverter.

Index	Sub index	Name	Type	Access	PDO Mapping	Value
1016h	0	Number of the last sub-index	UNSIGNED8	RO	No	4
	1 - 4	Consumer Heartbeat Time 1 – 4	UNSIGNED32	RW	No	0

At sub-indexes 1 to , it is possible to program the consumer by writing a value with the following format:

UNSIGNED32		
Reserved (8 bits)	Node-ID (8 bits)	HeartBeat time (16 bits)

- Node-ID: it allows programming the Node-ID for the heartbeat producer to be monitored.
- Heartbeat time: it allows programming the time, in 1 millisecond multiples, until the error detection if no message of the producer is received. The value 0 (zero) in this field disables the consumer.

Once configured, the heartbeat consumer initiates the monitoring after the reception of the first telegram sent by the producer. In case that an error is detected because the consumer stopped receiving messages from the heartbeat producer, the frequency inverter will turn automatically to the preoperational state and indicate .

As a producer, the frequency inverter CFW320 has an object for the configuration of that service:

Index	Sub index	Name	Type	Access	PDO Mapping	Value
1017h	0	Producer Heartbeat Time	UNSIGNED8	RW	No	0

The 1017h object allows programming the time in milliseconds during which the producer has to send a heartbeat telegram to the network. Once programmed, the inverter initiates the transmission of messages with the following format:

Identifier	byte 1	
	bit 7	bit 6 ... 0
700h + Node ID	Always 0	Slave State

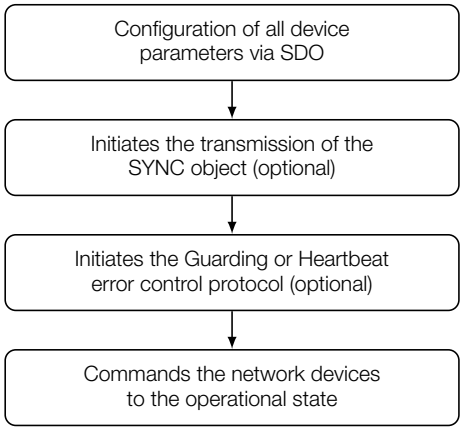
NOTE!

- This object is active even in the stopped state (see Table 7.12 on page 34).
- The value 0 (zero) in any of these two objects will disable this function.
- If after the error detection the service is enabled again, then the error indication will be removed from the HMI.
- The minimum value accepted by the frequency inverter CFW320 is 2ms, but considering the transmission rate and the number of nodes in the network, the times programmed for this function must be consistent, so that there is enough time for the transmission of the telegrams and also that the rest of the communication be able to be processed.
- For any every slave only one of the two services - Heartbeat or Node Guarding – can be enabled.

7.6 INITIALIZATION PROCEDURE

Once the operation of the objects available for the frequency inverter CFW320 is known, then it becomes necessary to program the different objects to operate combined in the network. In a general manner, the procedure for the initialization of the objects in a CANopen network follows the description of the next flowchart:

Figure 7.7: *Initialization process flowchart*



It is necessary to observe that the frequency inverter CFW320 communication objects (1000h to 1FFFh) are not stored in the nonvolatile memory. Therefore, every time the equipment is reset or switched off, it is necessary to redo the communication objects parameter setting.

8 STARTUP GUIDE

The main steps to start up the CFW320 frequency inverter in CANopen network are described below. These steps represent an example of use. Check out the specific chapters for details on the indicated steps.

8.1 INSTALLING THE ACCESSORY

1. Install the communication accessory, as indicated in the installation guide supplied with the accessory.
2. Observe the content of parameter P028. Check if the module was recognized. The detection is done automatically and does not require the user's intervention.
3. Connect the cable to the accessory, considering the recommended instructions in network installation, as described in Section 3 on page 10:
   - Use shielded cable.
   - Properly ground network equipment.
   - Avoid laying communication cables next to power cables.

8.2 CONFIGURING THE EQUIPMENT

1. Follow the recommendations described in the user manual to program the device parameters related to the motor parameterization, desired functions for the I/O signals, etc.
2. Program the command sources as desired for the application (P220 ... P228).
3. Configure communication parameters, such as protocol, address and baudrate in P700, P701 and P702.
4. Program the desired action for the equipment in case of communication fault in parameter P313.

8.3 CONFIGURING THE MASTER

The way the network configuration is done depends greatly on the used client and the configuration tool. It is essential to know the tools used to perform this activity. In general, the following steps are necessary to perform the network configuration.

1. Load the EDS file[4] to the list of devices in the network configuration tool.
2. Select CFW320 frequency inverter from the available list of devices on the network configuration tool. This can be done manually or automatically, if allowed by the tool.
3. During the configuration of the network, it is necessary to define which data will be read and written at frequency inverter CFW320 by configuring the transmission and reception PDOs as described in Section 7.3 on page 28. Among the main parameters that can be used to control the device, we can mention:
   - P680 - Status word (read)
   - P681 - Motor speed (read)
   - P684 - Control word (write)
   - P685 - Speed reference (write)
4. Configure error control using the Node Guarding or Heartbeat services as described in Section 7.5 on page 33.

Once configured, the network status P721 indicates Comm. Enabled or Error Ctrl. and the node state P722 indicates Operational. It is in this condition that PDO transmission and reception effectively occurs.

[4] The EDS file is available from WEG website (http://www.weg.net). It is important to note if the EDS configuration file is compatible with the firmware version of the CFW320 frequency inverter.

8.4 COMMUNICATION STATUS

Once the network is set up and the master programmed, it is possible to use the equipment parameters to identify some states related to communication.

- The parameters P721 and P722 indicate the status of CANopen communication.

The master of the network must also supply information about the communication with the slave.

8.5 OPERATION USING PROCESS DATA

Once the communication is established, the data mapped in the PDOs is automatically updated. Among the main parameters that can be used to control the device, we can mention:

- P680 - Status word (read)
- P681 - Motor speed (read)
- P684 - Control word (write)
- P685 - Speed reference (write)

It is important to know these parameters to program the master as desired for the application.

8.6 ACCESS TO PARAMETERS – ACYCLIC MESSAGES

Besides the cyclic communication using PDOs, the CANopen protocol also defines a kind of acyclic message via SDO, used especially in asynchronous tasks, such as parameter setting and configuration of the equipment.

The EDS file provides the full parameter list of the equipment, which can be accessed via SDO. The Section 6.4 on page 24 describes how to address the parameters of the frequency inverter CFW320 via SDO.

9 QUICK REFERENCE OF ALARMS AND FAULTS

Fault / Alarm	Description	Possible Causes
F032 Comm. Plug-in module comunication Lost	Main control cannot establish the communication link with the communication acccessory.	■ Accessory damaged. ■ Poor connection of the accessory. ■ Problem in the identification of the accessory; refer to P028.
A133 No Power Supply on the CAN Interface	It indicates that the CAN interface has no power supply between pins 25 and 29 of the connector.	■ Measure if there is voltage within the allowed range between pins 25 and 29 of the CAN interface connector. ■ Check if the power supply cables are not misconnected or inverted. ■ Check for contact problems on the cable or connector of the CAN interface.
A134 Bus Off	Bus off error detected on the CAN interface.	■ Check for short circuit on the CAN circuit transmission cable. ■ Check if the cables are not misconnected or inverted. ■ Check if all the network devices use the same baud rate. ■ Check if the termination resistors with the right specification were installed only at the end of the main bus. ■ Check if the CAN network was properly installed.
A135 Node Guarding/ Heartbeat	CANopen communication error control detected communication error using the guarding mechanism.	■ Check the times set on the master and on the slave for message exchange. In order to prevent problems due to transmission delays and time counting, it is recommended that the values set for error detection by the slave be multiples of the times set for message exchange on the master. ■ Check if the master is sending the guarding telegrams in the time set. ■ Check problems in the communication that may cause missing telegrams or transmission delays.
F233 No Power Supply on the CAN Interface	It indicates that the CAN interface has no power supply between pins V(-) and V(+) of the connector.	■ Measure if there is voltage within the allowed range between pins V(-) and V(+) of the CAN interface connector. ■ Check if the power supply cables are not misconnected or inverted. ■ Check for contact problems on the cable or connector of the CAN interface.
F234 Bus Off	Bus off error detected on the CAN interface.	■ Check for short circuit on the CAN circuit transmission cable. ■ Check if the cables are not misconnected or inverted. ■ Check if all the network devices use the same baud rate. ■ Check if the termination resistors with the right specification were installed only at the end of the main bus. ■ Check if the CAN network was properly installed.
F235 Node Guarding/ Heartbeat	CANopen communication error control detected communication error using the guarding mechanism.	■ Check the times set on the master and on the slave for message exchange. In order to prevent problems due to transmission delays and time counting, it is recommended that the values set for error detection by the slave be multiples of the times set for message exchange on the master. ■ Check if the master is sending the guarding telegrams in the time set. ■ Check problems in the communication that may cause missing telegrams or transmission delays.

Fault and alarm operation:

- Faults operate by indicating their occurrence on the HMI, in the frequency inverter status word (P006), in the present fault parameter (P049) and disabling the motor. They can only be reset with a reset command or de-energizing the frequency inverter.
- Alarms operate by indicating their occurrence on the HMI and in the present alarm parameter (P048). They are automatically reset when the alarm condition ceases existing.

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