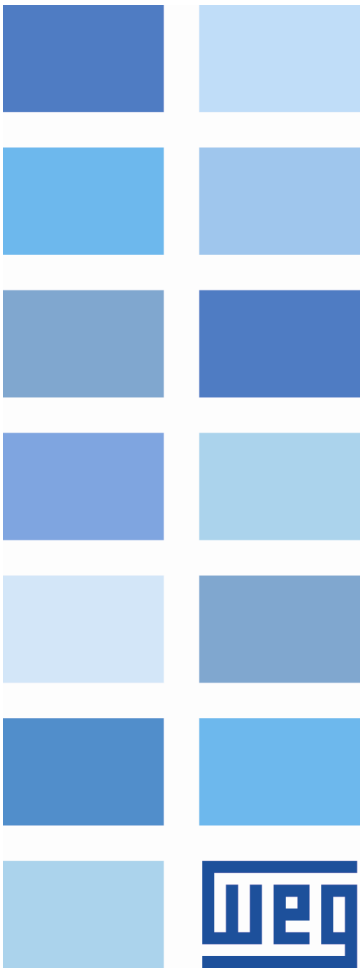
DeviceNet

CFW300

User's Guide



DeviceNet User's Guide

Series: CFW300

Language: English

Document Number: 10003961988 / 00

Build 463*

Publication Date: 03/2016

Contents

ABOUT THE MANUAL

This manual supplies the necessary information for the operation of the CFW300 frequency converter using the DeviceNet protocol. This manual must be used together with the CFW300 user's manual and programming manual.

ABBREVIATIONS AND DEFINITIONS

ASCII American Standard Code for Information Interchange
CRC Cycling Redundancy Check
CiA CAN in Automation
CIP Common Industrial Protocol
PLC Programmable Logic Controller
HMI Human-Machine Interface
ODVA Open DeviceNet Vendor Association
ro Read only (solamente de lectura)
rw Read/write (lectura y escrita)

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 DeviceNet protocol was developed based on the following specifications and documents:

Document	Version	Source
CAN Specification	2.0	CiA
Volume One - Common Industrial Protocol (CIP) Specification	3.2	ODVA
Volume Three - DeviceNet Adaptation of CIP	1.4	ODVA
Planning and Installation Manual - DeviceNet Cable System	PUB00027R1	ODVA

In order to obtain this documentation, consult ODVA, which is nowadays the organization that keeps, publishes and updates the information related to the DeviceNet network.

1 DEVICENET COMMUNICATION INTRODUCTION

In order to operate the equipment in a DeviceNet network, it is necessary to know the manner this communication is performed. Therefore, this section brings a general description of the DeviceNet protocol operation, containing the functions used by the CFW300. Refer to the protocol specification for a detailed description.

1.1 CAN

DeviceNet is a network based on CAN, i.e., it uses CAN telegrams for exchanging data in the network.

The CAN protocol is a serial communication protocol that describes the services of layer 2 of the ISO/OSI model (data link layer)[1] . This layer defines the different types of telegrams (frames), the error detection method, the validation and arbitration of messages.

1.1.1 Data Frame

CAN network data is transmitted by means of a data frame. This frame type is composed mainly by an 11 bits[2] identifier (arbitration field), and by a data field that may contain up to 8 data bytes.

Identifier	8 data bytes							
11 bits	byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7

1.1.2 Remote Frame

Besides the data frame, there is also the remote frame (RTR frame). This type of frame does not have a data field, but only the identifier. It works as a request, so that another network device transmits the desired data frame. The DeviceNet communication protocol does not use this type of frame.

1.1.3 Access to the Network

Any device in a CAN network can make an attempt to transmit a frame to the network in a certain moment. If two devices try to access the network simultaneously, the one that sends the message with the highest priority will be able to transmit. The message priority is defined by the CAN frame identifier, the smaller the value of this identifier, the higher the message priority. The telegram with the identifier 0 (zero) is the one with the highest priority.

1.1.4 Error Control

The CAN specification defines several error control mechanisms, which makes the network very reliable and with a very low undetected transmission error rate. Every network device must be able to identify the occurrence of these errors, and to inform the other elements that an error was detected.

A CAN network device has internal counters that are incremented every time a transmission or reception error is detected, and are decremented when a telegram is successfully transmitted or received. If a considerable amount of errors occurs, the device can be led to the following states:

[1] In the CAN protocol specification, the ISO11898 standard is referenced as the definition of the layer 1 of this model (physical layer).

[2] The CAN 2.0 specification defines two data frame types, standard (11 bit) and extended (29 bit). For this implementation, only the standard frames are accepted.

- **Error Active:** the internal error counters are at a low level and the device operates normally in the CAN network. You can send and receive telegrams and act in the CAN network if it detects any error in the transmission of telegrams.
- **Warning** when the counter exceeds a defined limit, the device enters the warning state, meaning the occurrence of a high error rate.
- **Error Passive** when this value exceeds a higher limit, the device enters the error passive state, and it stops acting in the network when detecting that another device sent a telegram with an error.
- **Bus Off** finally, we have the bus off state, in which the device will not send or receive telegrams any more. The device operates as if disconnected from the network.

1.1.5 CAN and DeviceNet

Only the definition of how to detect errors, create and transmit a frame, are not enough to define a meaning for the data transmitted via the network. It is necessary to have a specification that indicates how the identifier and the data must be assembled and how the information must be exchanged. Thus, the network elements can interpret the transmitted data correctly. In that sense, the DeviceNet specification defines exactly how to exchange data among the devices and how every one must interpret these data.

There are several other protocols based on CAN, as DeviceNet, CANopen, J1939, etc., which use CAN frames for the communication. However, those protocols cannot be used together in the same network.

1.2 DEVICENET NETWORK CHARACTERICTICS

Introduced in 1994, DeviceNet is an implementation of the Common Industrial Protocol (CIP) for industrial communication networks. Developed originally by Allen-Bradley, it had its technology transferred to the ODVA that, since then, keeps, publishes and promotes DeviceNet and other networks based on the CIP[3] protocol. Furthermore, it uses the Controller Area Network (CAN) protocol for the data link and access to the medium, layers 2 and 1 of the OSI/ISO model, respectively.

Used mainly for the connection of industrial controllers and I/O devices, the protocol follows the model producer-consumer, supports multiple communication modes and has priority between messages.

It is a system that can be configured to operate in master-slave architecture as well as in a distributed point-to-point architecture. Besides, it defines two kinds of messages, I/O (process data) and explicit (configuration and parameter setting). It also has mechanisms to detect duplicated addresses and for node isolation in case of critical faults.

A DeviceNet network can have up to 64 devices, addressed from 0 to 63. Any of them can be used. There is no restriction, although the 63 should be avoided because it is usually used for commissioning.

1.3 PHYSICAL LAYER

DeviceNet uses a network topology of the trunk/derivation type that allows the signal wiring as well as the power wiring to be present in the same cable. This power is supplied by a power supply connected directly to the network, which feeds the CAN transceivers of the nodes, and has the following characteristics:

- 24 Vdc;
- DC output isolated from the AC input;
- Current capacity compatible with the installed equipment.

[3] CIP actually represents a family of networks. DeviceNet, EtherNet/IP and ControlNet use CIP in the application layer. The difference among them is primordially in the data link and physical layers.

The used Baud rate depends on the size (cable length) of the network, as showed in the table 1.1

Table 1.1: *Network size x Baud rate*

Baud Rate	Network Size	Derivation	
125 Kbps	500 m	6	156 m
250 Kbps	250 m	6	78 m
500 Kbps	100 m	6	39 m

In order to avoid reflections in the line, it is recommended the installation of termination resistors at the line extremes, because the absence of them may cause intermittent errors. This resistor must have the following characteristics, according to the protocol specification:

- 121Ω;
- 0,25 W;
- 1% tolerance.

For DeviceNet, several types of connectors can be used, sealed ones as well as open ones. The definition of the type to be used depends on the application and on the equipment operation environment. The CFW300 uses a 5 wire connector, and its pinout is showed in the section 3. For a complete description of the connectors used with DeviceNet, consult the protocol specification.

1.3.1 Data Link Layer

The DeviceNet data link layer is defined by the CAN specification, which defines two possible states; dominant (logic level 0) and recessive (logic level 1). A node can bring the network to the dominant state if it transmits any information. Thus, the bus will only be in the recessive state if there where no transmitting nodes in the dominant state.

CAN uses the CSMA/NBA to access the physical medium. This means that a node, before transmitting, must verify if the bus is free. In case it is, then the node can initiate the transmission of its telegram. In case it is not, then the node must await. If more than one node access the network simultaneously, a priority mechanism takes action to decide which one will have priority over the others. This mechanism is not destructive, i.e., the message is preserved even if there is a collision between two or more telegrams.

CAN defines four types of telegrams (data, remote, overload and error). Among them, DeviceNet uses only the data frame and the error frame.

Data is moved using the data frame. This frame structure is showed in the figure 1.1.

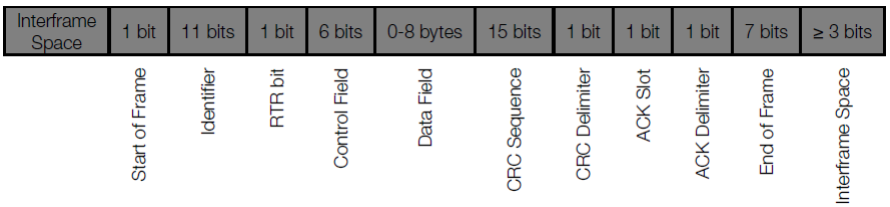


Figure 1.1: *CAN data frame*

Errors, however, are indicated by means of the error frames. CAN has a very robust error verification and confinement. This assures that a node with problems does not impair the communication in the network.

For a complete description of the errors, consult the CAN specification.

1.3.2 Network and Transport Layer

DeviceNet requires that a connection be established before data exchange with the device takes place. In order to establish this connection each DeviceNet node must implement the Unconnected Message Manager (UCMM) or the Group 2 Unconnected Port. These two allocation mechanisms use messages of the explicit type to establish a connection, which will then be used for process data exchange between one node and the other. This data exchange uses messages of the I/O type (refer to item 1.3.5).

The DeviceNet telegrams are classified in groups, which define specific functions and priorities. Those telegrams use the identifier field (11 bits) of the CAN data frame to uniquely identify each one of the messages, thus assuring the CAN priority mechanism.

A DeviceNet node can be a client, a server or both. Furthermore, clients and servers can be producers and/or consumers of messages. In a typical client node, for instance, its connection will produce requests and will consume answers. Other client or server connections will only consume messages. In other words, the protocol allows several connection possibilities among the devices.

The protocol also has a resource for detection of nodes with duplicated addresses (Mac ID). Avoiding that duplicated addresses occur is, in general, more efficient than trying to locate them later.

1.3.3 Application Layer – CIP Protocol

In the application layer, DeviceNet uses the Common Industrial Protocol (CIP). It is a protocol strictly orientated to objects, used also by ControlNet and EtherNet/IP. In other words, it is independent from the physical medium and from the data link layer. The figure 1.1 presents the structure of this protocol.

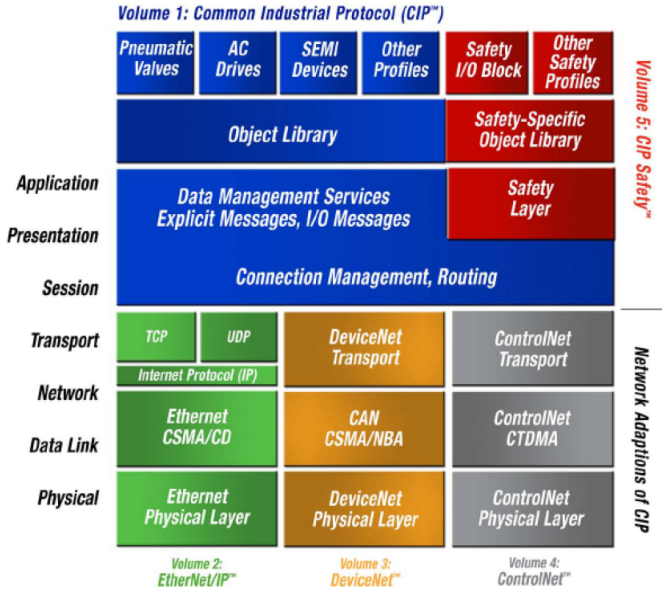


Figure 1.2: *CIP protocol structure in layers*

The CIP has two main purposes:

- Transport of I/O devices control data;
- Transport of configuration and diagnosis information of the system being controlled.

A DeviceNet node (master or slave) is then molded by a set of CIP objects, which encapsulate data and services, thus determining its behavior.

There are obligatory objects (each device must have) and optional objects. Optional objects are those that mold the device according to the category (called profile) to which they belong, as: AC/DC drive, bar code reader or pneumatic valve. For being different, each one of these will contain a group of also different objects.

For more information refer to the DeviceNet specification. It presents the complete list of devices already standardized by the ODVA, as well as the objects that compose them.

1.3.4 Configuration File (EDS)

Each device in a DeviceNet network has an EDS configuration file that contains information about the device operation and must be registered in the network configuration software, for programming of devices present in the DeviceNet Network.

The EDS configuration file is supplied together with the product, and it can also be obtained from the website http://www.weg.net. It is necessary to observe the equipment software version, in order to use an EDS file that is compatible with that version.

1.3.5 Communication Modes

The DeviceNet protocol presents two basic types of messages, I/O and explicit. Each one of them is adequate to a specific kind of data, as described below:

- I/O: it is a kind of synchronous telegram dedicated to the movement of priority data between one producer and one or more consumers. They are divided according to the data exchange method. The main types are:
  - Polled: it is a communication method where the master sends one telegram to each of the slaves of its list (scan list). As soon as receiving the request, the slave responds promptly to the request from the master. This process is repeated until all be consulted, restarting the cycle;
  - Bit-strobe: it is a communication method where the master sends to the network a telegram containing 8 data bytes. Each bit from those 8 bytes represents a slave that, if addressed, responds according to the programmed;
  - Change of State: it is a communication method where the data exchange between master and slave occurs only when changes in the monitored/controlled values happened, until a certain time limit. When this limit is reached, the transmission and reception will occur even if there were no changes. The configuration of this time variable is done in the network configuration program;
  - Cyclic: it is another communication method very similar to the previous one. The only difference stays in the production and consume of messages. In this type, every data exchange occurs in regular time intervals, whether or not they had been changed. This time period is also adjusted in the network configuration software.
- Explicit: it is a kind of general purpose telegram and without priority. It is mainly used for asynchronous tasks like the parameter settings and the configuration of the equipment.

NOTE!
the CFW300 frequency converter does not provide the bit-strobe communication method.

1.3.6 Set of Predefined Master/Slave Connections

DeviceNet uses fundamentally a point-to-point message model. However, it is quite common to use a predefined communication model based on the master/slave mechanism.

This model uses a simplified message movement of the I/O type, very common in control applications. An advantage of this method is that the necessary requests to run it are generally less than for the UCMM. Even simple devices with limited resources (memory, 8 bit processor) are capable of executing the protocol.

2 DEVICENET COMMUNICATION INTERFACE

The accessory CFW300 frequency converter 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

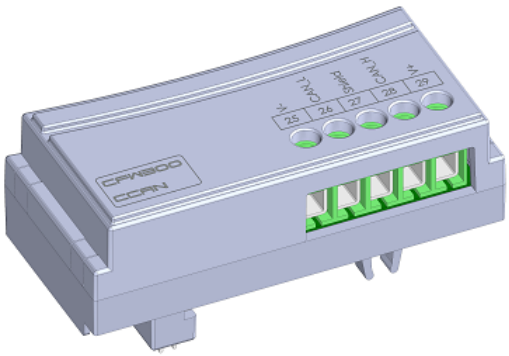


Figure 2.1: *Acessório CCAN*

- 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[4].
- 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:

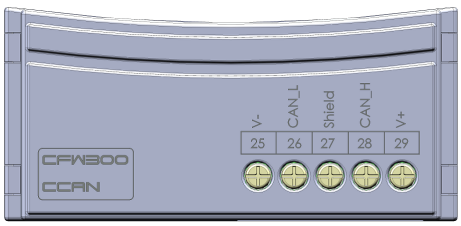


Figure 2.2: *Detalhe do conector CAN*

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

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

2.3 POWER SUPLLY

The CAN interfaces require an external power supply between pins 25 and 29 of the network connector. The data for individual consumption and input voltage are shown in the following table.

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 DeviceNet communication for the CFW300 frequency converter are made trough the HMI and parameters of the product.

3 DEVICENET NETWORK INSTALLATION

The DeviceNet 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 DeviceNet interface generally allow the configuration of the desired baud rate, ranging from 125Kbit/s até 500Kbit/s. The baud rate that can be used by equipment depends on the length of the cable used in the installation. The next table shows the baud rates and the maximum cable length that can be used in the installation, according to the protocol recommendation [5].

Table 3.1: *Supported baud rates and installation size*

Baud Rate	Cable length
125 Kbps	500 m
250 Kbps	250 m
500 Kbps	100 m

All network equipment must be programmed to use the same communication baud rate. At the CFW300 servo drive the baud rate configuration is done through the P0702 parameter.

3.2 ADDRESS IN THE DEVICENET NETWORK

Each DeviceNet network device must have an address or Node ID, and may range from 0 to 63. This address must be unique for each equipment. For CFW300 frequency converter the address configuration is done through the parameter P0701.

3.3 TERMINATION RESISTOR

The CAN bus line must be terminated with resistors to avoid line reflection, which can impair the signal and cause communication errors. The extremes of the CAN bus must have a termination resistor with a 121Ω|0.25W value, connecting the CAN_H and CAN_L signals.

3.4 CABLE

The connection of CAN_L and CAN_H signals must done with shielded twisted pair cable. The following table shows the recommended characteristics for the cable.

[5] Different products may have different maximum allowed cable length for installation.

Table 3.2: DeviceNet cable characteristics

Cable Length (m)	Resistence per meter (Ω|m)	Conductor cross section (mm2)
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 IN 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. 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.

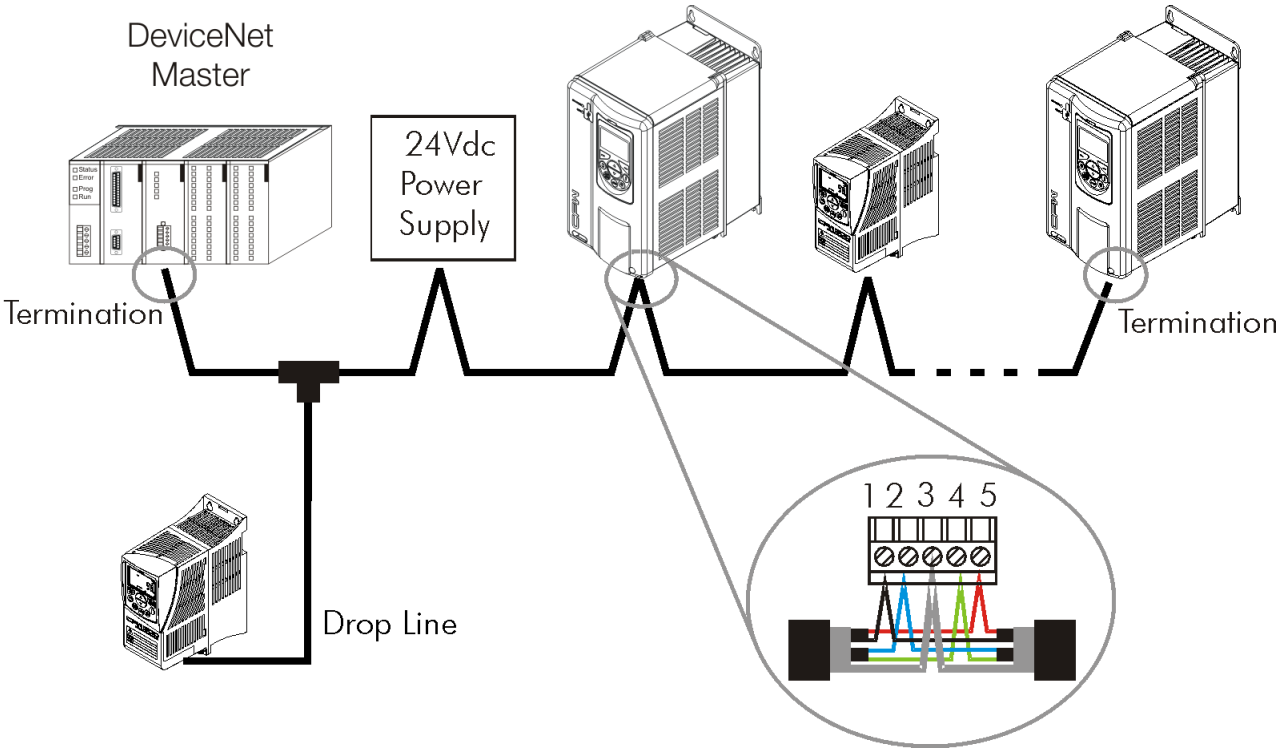


Figure 3.1: DeviceNet network installation example

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.

4 PARAMETERIZATION

Next, the CFW300 frequency converter parameters related to the DeviceNet communication will be presented.

4.1 SYMBOLS FOR THE PROPERTIES DESCRIPTION

- **RO** Read-only parameter
- **CFG** Parameter that can be changed only with a stopped motor
- **CAN** Parâmetro visível através da HMI se o produto possuir interface CAN instalada

P105 – 1ST/2ND RAMP SELECTION

P220 – LOCAL/REMOTE SELECTION SOURCE

P221 – SPEED REFERENCE SELECTION – LOCAL SITUATION

P222 – SPEED REFERENCE SELECTION – REMOTE SITUATION

P223 – FORWARD/REVERSE SELECTION – LOCAL SITUATION

P224 – RUN/STOP SELECTION – LOCAL SITUATION

P225 – JOG SELECTION – LOCAL SITUATION

P226 – FORWARD/REVERSE SELECTION – REMOTE SITUATION

P227 – RUN/STOP SELECTION – REMOTE SITUATION

P228 – JOG SELECTION – REMOTE SITUATION

Description:

These parameters are used to configure the command sources for CFW300 frequency converter local and remote situations. To control the device through the DeviceNet interface, select the options 'DeviceNet(CO/DN/DP)' available in these parameters.

The detailed description of these parameters is found in the CFW300 programming manual.

P313 – ACTION IN CASE OF COMMUNICATION ERROR

Range:	0 = Inactive 1 = Disable via Run/Stop 2 = Disable via General Enable 3 = Change to Local 4 = Change to Local keeping commands and reference 5 = Causes a Fault	**Default:** 1
Properties:	CFG	

Description:

It allows the selection of the action performed by the device, if it is controlled via network and detects a communication error.

Table 4.1: *P313 options*

Options	Description
0 = Inactive	It takes no action and the device remains in the existing status.
1 = Disable via Run/Stop	It performs a stop command and the motor stops according to the programmed deceleration ramp.
2 = Disable via General Enable	The device is disabled by removing the General Enabling and the motor coasts to stop.
3 = Change to Local	The device commands change to Local.
4 = Change to Local keeping commands and reference	The device commands change to Local, but the enabling commands and speed reference received via network are kept, if the device 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 = Causes a Fault	Instead of an alarm, the communication error causes a device fault, so that a fault reset becomes necessary in order to restore normal operation.

The device consider the following events as 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).
- Alarm A136/Fault F236: DeviceNet master in Idle mode.
- Alarm A137/Fault F237: detected timeout in one or more DeviceNet I/O connections.

The actions described in this parameter are done 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 written in this parameter be 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.

P680 – LOGICAL STATUS

Range:	0000h ... FFFFh	**Default:** -
Properties:	RO	

Description:

It allows the device status monitoring. Each bit represents a specific status:

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Function	Fault condition	Reserved	Undervoltage	LOC/REM	JOG	Speed direction	General enable active	Motor Running	Alarm condition	In configuration mode	Second ramp	Reserved	Reserved			

Table 4.2: P680 parameter bit functions

Bit	Value/Description
Bit 0 ... 3	Reserved
Bit 4 Quick stop active	**0**: The quick stop command is not active. **1**: The drive is executing the quick stop command.
Bit 5 Second ramp	**0**: The drive is configured to use the first ramp values, programmed in P100 and P101, as the motor acceleration and deceleration ramp times. **1**: The drive is configured to use the second ramp values, programmed in P102 and P103, as the motor acceleration and deceleration ramp times.
Bit 6 In configuration mode	**0**: The drive is operating normally. **1**: The drive is in the configuration mode. It indicates a special condition during which the drive cannot be enabled: ▪ Running the self-tuning routine. ▪ Running the oriented start-up routine. ▪ Running the HMI copy function. ▪ Running the flash memory card self-guided routine. ▪ There is a parameter setting incompatibility. ▪ There is no power at the device power section.
Bit 7 Alarm condition	**0**: The drive is not in alarm condition. **1**: The drive is in alarm condition.
Bit 8 Motor Running	**0**: The motor is stopped (or coast to stop). **1**: The drive is running the motor at the set point speed, or executing either the acceleration or the deceleration ramp.
Bit 9 General enable active	**0**: General Enable is not active. **1**: General Enable is active and the drive is ready to run the motor.
Bit 10 Speed direction	**0**: The motor is running in the reverse direction. **1**: The motor is running in the forward direction.
Bit 11 JOG	**0**: JOG function disabled. **1**: JOG function enabled.
Bit 12 LOC/REM	**0**: Drive in Local mode. **1**: Drive in Remote mode.
Bit 13 Undervoltage	**0**: No Undervoltage. **1**: With Undervoltage.
Bit 14 Automatic (PID)	**0**: PID in manual mode. **1**: PID in Automatic mode.
Bit 15 Fault condition	**0**: The drive is not in a fault condition. **1**: The drive has detected a fault.

P681 – MOTOR SPEED IN 13 BITS

Range:	-32768 ... 32767	**Default:** -
Properties:	RO	

Description:

It allows monitoring the motor speed. This word uses 13-bit resolution with signal to represent the motor rated frequency (P403):

P681 = 0000h (0 decimal) → motor speed = 0
P681 = 2000h (8192 decimal) → motor speed = rated frequency (P403)

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 – CONTROL WORD

Range:	0000h ... FFFFh	**Default:** 0000h
Properties:	CAN	

Description:

It is the device DeviceNet interface control word. This parameter can only be changed via DeviceNet interface. For the other sources (HMI, etc.) it behaves like a read-only parameter.

In order to have those commands executed, it is necessary to program the equipment to be controlled via DeviceNet. This programming is achieved by means of parameters P105 e P220 até P228.

Each bit of this word represents a command that can be executed.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Function	Reserved								Fault Reset	Quick Stop	Second Ramp	LOC/REM	JOG	Speed direction	General Enable	Run/Stop

Table 4.3: *parameter bit functions P684*

Bit	Values
Bit 0 Run/Stop	**0**: It stops the motor with deceleration ramp. **1**: The motor runs according to the acceleration ramp until reaching the speed reference value.
Bit 1 General Enable	**0**: It disables the drive, interrupting the supply for the motor. **1**: It enables the drive allowing the motor operation.
Bit 2 Speed Direction	**0**: To run the motor in a direction opposed to the speed reference. **1**: To run the motor in the direction indicated by the speed reference.
Bit 3 JOG	**0**: It disables the JOG function. **1**: It enables the JOG function.
Bit 4 LOC/REM	**0**: The drive goes to the Local mode. **1**: The drive goes to the Remote mode.
Bit 5 Second Ramp	**0**: The drive uses the first ramp values, programmed in P100 and P101, as the motor acceleration and deceleration ramp times. **1**: The drive is configured to use the second ramp values, programmed in P102 and P103, as the motor acceleration and deceleration ramp times.
Bit 6 Quick Stop	**0**: Quick Stop command not activated. **1**: Quick Stop command activated.
Bit 7 Fault Reset	**0**: no function. **1**: If in a fault condition, then it executes the reset.
Bit 8...15	Reserved.

P685 – SPEED REFERENCE

Range:	-32768 ... 32767	**Default:** 0
Properties:	CAN	

Description:

It allows programming the motor speed reference via the DeviceNet interface. This parameter can only be changed via DeviceNet interface. For the other sources (HMI, etc.) it behaves like a read-only parameter.

In order that the reference written in this parameter be used, it is necessary that the drive be programmed to use the speed reference via CANopen/DeviceNet/Profibus DP. This programming is achieved by means of parameters P221 and P222.

This word uses a 13-bit resolution with signal to represent motor 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

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

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 - P685- 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!
Os valores transmitidos via rede apresentam uma limitação na escala utilizada, permitindo que no máximo seja programado uma referência de velocidade de 4 vezes a frequência nominal do motor.

P695 – DIGITAL OUTPUTS SETTING

Range:	0000b... 1111b	**Default:** 0000b
Properties:	RW	

Description:

It allows the control of the digital outputs by means of the network interfaces (DeviceNet, etc.). It is not possible to change this parameter via HMI.

Each bit of this parameter corresponds to the desired value for one digital output. In order to control the correspondent digital output according to this content, it is necessary that its function be programmed for “P0695 Content” at parameters to program digital outputs function.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Function	Reserved											Reserved	Value for DO4	Value for DO3	Value for DO2	Value for DO1

Table 4.4: *Bit function for parameter P0695*

Bit	Valor/Descrição
Bit 0 Value for DO1	**0**: Open DO1. **1**: Close DO1.
Bit 1 Value for DO2	**0**: Open DO2. **1**: Close DO2.
Bit 2 Value for DO3	**0**: Open DO3 . **1**: Close DO3.
Bit 3 Value for DO4	**0**: Open DO4. **1**: Close DO4.
Bit 4...15	Reserved.

P696 - VALUE 1 FOR ANALOG OUTPUTS

P697 - VALUE 2 FOR ANALOG OUTPUTS

Range:	-32768 ... 32767	**Default:** 0
Properties:	rw	

Description:

They allow the control of the analog outputs by means of network interfaces (DeviceNet, etc.). It is not possible to change this parameter via HMI.

The value written in these parameters may control the analog output value, as long as you program the option “P0696 ... P0697 value” at the analog output function parameters.

The value must be written in a 15-bit[6] scale (7FFFh = 32767) to represent 100 % of the output desired value, i.e.:

- P0696 = 0000h (0 decimal) → analog output value = 0 %
- P0696 = 7FFFh (32767 decimal) → analog output value = 100 %

The showed example was for P0696, but the same scale is also used for the other parameters. For instance, to control the analog output 1 via DeviceNet, the following programming must be done:

- Choose a parameter from P0696 ... P0697 to be the value used by the analog output 1. For this example, we are going to select P0696.
- Program the option “P0696 value” as the function for the analog output 1 in P0254.
- Using the network interface, write in P0696 the desired value for the analog output 1, between 0 and 100 %, according to the parameter scale.

[6]For the actual output resolution, refer to the product manual.

P700 – CAN PROTOCOL

Range:	1 = CANopen 2 = DeviceNet	**Default:** 2
Properties:	CAN	

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.

P701 – CAN ADDRESS

Range:	0 a 127	**Default:** 63
Properties:	CAN	

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

Range:	0 = 1 Mbit/s / Autobaud 1 = 800 Kbit/s / Autobaud 2 = 500 Kbit/s 3 = 250 Kbit/s 4 = 125 Kbit/s 5 = 100 Kbit/s / Autobaud 6 = 50 Kbit/s / Autobaud 7 = 20 Kbit/s / Autobaud 8 = 10 Kbit/s / Autobaud	**Default:** 0
Properties:	CAN	

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.

P703 – BUS OFF RESET

Range:	0 = Manual 1 = Automatic	**Default:** 1
Properties:	CAN	

Description:

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

Table 4.5: *Options for the parameter P703*

Option	Description
0 = Manual Reset	If bus off occurs, the A134/F34 alarm will be indicated on the HMI, the action 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 Reset	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 CONTROLER STATUS

Range:	0 = Disable 1 = Autobaud 2 = CAN Enabled 3 = Warning 4 = Error Passive 5 = Bus Off 6 = No Bus Power	**Default:** -
Properties:	RO	

Description:

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

Table 4.6: *Indication of the parameter P705*

Value	Description
0 = Disable	Inactive CAN interface. It occurs when CAN protocol is not programmed at P705.
1 = Autobaud	CAN controller is trying to detect baud rate of the network (only for DeviceNet communication protocol).
2 = CAN Enable	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 = Sem Alimentação	CAN interface does not have power supply between the pins 1 and 5 of the connector.

P706 – RECEIVED CAN TELEGRAM COUNTER

Range:	0 a 9999	**Default:** -
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 – TRANSMITED CAN TELEGRAM COUNTER

Range:	0 a 9999	**Default:** -
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 ERROR COUNTER

Range:	0 a 9999	**Default:** -
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 – LOST CAN MESSAGE COUNTER

Range:	0 a 9999	**Default:** -
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.

P710 – DEVICENET I/O INSTANCES

Range:	0 = ODVA Basic Speed (2 words) 1 = ODVA Extended Speed (2 words) 2 = Manuf. Spec. 2W (2 words) 3 = Manuf. Spec. 3W (3 words) 4 = Manuf. Spec. 4W (4 words) 5 = Manuf. Spec. 5W (5 words) 6 = Manuf. Spec. 6W (6 words)	**Default:** 0
Properties:	CAN	

Description:

It allows selecting the Assembly class instance for the I/O type communication.

The CFW300 frequency converter has seven setting options. Two of them follow the ODVA AC/DC Drive Profile. The other five represent specific CFW300 frequency converter words. The tables presented next describe each of these control and monitoring words.

NOTE!
If this parameter is changed, it becomes valid only after cycling the power of the product.

0 = Data format for the ODVA Basic Speed (2 words) instances:

Called Basic Speed, these instances represent the simplest operation interface of a device according to the AC/DC Drive Profile. The data mapping is showed below.

Monitoring (Input)

Instância	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
70	0						Running1		Faulted
	1	-							
	2	Speed Actual (low byte)							
	3	Speed Actual (high byte)							

Control (Output)

Instância	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
20	0						Fault Reset		Run Fwd
	1	-							
	2	Speed Reference (low byte)							
	3	Speed Reference (high byte)							

1 = Data format for the ODVA Extended Speed (2 words) instances:

Called Extended Speed, these instances present an equipment operation interface a little bit more refined, which follows the AC/DC Drive Profile. The data mapping is showed below.

Monitoring (Input)

Instância	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
71	0	At Reference	Ref. from Net	Ctrl from Net	Ready	Running2 (Rev)	Running1 (Fwd)	Warning	Faulted
	1	Drive State							
	2	Speed Actual (low byte)							
	3	Speed Actual (high byte)							

Control (Output)

Instância	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
21	0		NetRef	NetCtrl		Fault Reset		Warning	Faulted
	1	-							
	2	Speed Actual (low byte)							
	3	Speed Actual (high byte)							

The table below presents the meaning of data for the instances 20/70 and 21/71.

Monitoring:

Bit	Valor/Descrição
Bit 0 Faulted	**0**: frequency converter is not in a fault state **1**: Some fault registered by the frequency converter. Note: The number of the fault can be read through parameter P049 – Current Fault.
Bit 1 Warning	**0**: frequency converter is not in an alarm state. **1**: Some alarm registered by the frequency converter. The number of the alarm can be read by means of parameter P048 – Current Alarm.
Bit 2 Running1 (Fwd)	**0**: The motor is not running forward. **1**: The motor is running forward.
Bit 3 Running2 (Rev)	**0**: The motor is not running in the reverse direction. **1**: The motor is running in the reverse direction.
Bit 4 Ready	**0**: frequency converter not ready to operate. **1**: frequency converter ready to operate (Ready, Enabled or Stopping states)..
Bit 5 Ctrl from Net	**0**: Drive locally controlled. **1**: Drive remotely controlled.
Bit 6 Ref. from Net	**0**: Speed reference is not being sent via DeviceNet. **1**: Indicates speed reference being sent via DeviceNet.
Bit 7 At Reference	**0**: frequency converter has not reached the programmed speed yet. **1**: frequency converter has reached the programmed speed.

- Byte 1 indicates the state of the drive:
  - 0 = Non Existent
  - 1 = Startup
  - 2 = Not Ready
  - 3 = Ready
  - 4 = Enabled
  - 5 = Stopping
  - 6 = Fault Stop
  - 7 = Faulted
- Bytes 2 (low) and 3 (high) represent the effective speed of the motor in RPM.

Control:

Bit	Valor/Descrição
Bit 0 Run Fwd	**0**: Stop motor **1**: The motor runs in the forward direction.
Bit 1 Run Rev	**0**: Stop motor **1**: The motor runs in the reverse direction.
Bit 2 Fault Reset	**0**: Not used. **1**: : If in a fault condition, it resets the frequency converter.
Bits 3 e 4	Reserved.
Bit 5 NetCtrl	**0**: frequency converter selects the local mode. **1**: frequency converter selects the remote mode.
Bit 6 NetRef	**0**: Speed reference is not being sent via network. **1**: Speed reference being sent via network.
Bits 7	Reserved.

- Bytes 2 (low) and 3 (high) represent the speed reference of the motor in RPM.

2 = Data format for the Manufacturer Specific 2W (2 words) instances:
3 = Data format for the Manufacturer Specific 3W (3 words) instances:
4 = Data format for the Manufacturer Specific 4W (4 words) instances:
5 = Data format for the Manufacturer Specific 5W (5 words) instances:
6 = Data format for the Manufacturer Specific 6W (6 words) instances:

Called Manufacturer Specific, these instances present the simplest equipment operation interface according to the CFW300 frequency converter profile. The data mapping is showed below. Besides the command and monitoring words showed above, they make it possible to program up to 8 parameters of the equipment for reading and/or writing via network, through P711 to P718 parameters.

Monitoring (Input)

Table 4.7: *Programming of the I/O words*

Instance	16 bits word	Function	Option of P710
150	#1	Status Word - P680	2
	#2	Motor Speed - P681	3
	#3	Read #3 DeviceNet	4
	#4	Read #4 DeviceNet	5
	#5	Read #5 DeviceNet	6
	#6	Read #6 DeviceNet	

Programable Constant

Control (Output)

Table 4.8: *Programming of the I/O words*

Instance	16 bits word	Function	Option of P710
100	#1	Control word - P684	2
	#2	Speed reference - P685	3
	#3	Write #3 DeviceNet	4
	#4	Write #4 DeviceNet	5
	#5	Write #5 DeviceNet	6
	#6	Write #6 DeviceNet	

Programable Constant

P711 – DEVICENET READING #3

P712 – DEVICENET READING #4

P713 – DEVICENET READING #5

P714 – DEVICENET READING #6

Range:	0 a 959	**Default:** 0
Properties:	CAN	

Description:

These parameters allow the user to program the content of input words 3 to 6 (input: slave sends to the master). Using these parameters, it is possible to program the number of other parameter whose content shall be made available in the input area of the network master.

For instance, in case it is necessary to read the motor current in amperes from the inverter, the value 3 must be programmed in some of these parameters, since the parameter P003 is the parameter that contains this information. Note that the reading value of any parameter is represented with a 16-bit word. Even if the parameter has a decimal resolution value, the value is transferred with no decimal indication. For instance, if the parameter P003 has the value 4.7, the value transferred via network will be 47.

These parameters are used only if the device is programmed in parameter P710 to use options 3 through 6. According to the selected option, it will be available up to six words for reading by the network master.

The first two input words are fixed.

NOTE!
The 0 (zero) value disables the word writing. The amount of input words, however, always remains the same as it was programmed in parameter P710.

P715 – DEVICENET WRITING #3

P716 – DEVICENET WRITING #4

P717 – DEVICENET WRITING #5

P718 – DEVICENET WRITING #6

Range:	0 a 959	**Default:** 0
Properties:	CAN	

Description:

These parameters allow the user to program the content of output words 3 to 6 (output: masters sends to the slave). Using these parameters, it is possible to program the number of other parameter whose content shall be made available in the output area of the network master.

For instance, in case it is necessary to write the acceleration in the device, the value 100 must be programmed in some of these parameters, since the parameter P100 is the parameter where this information is programmed. Note that the written value of any parameter is represented with a 16-bit word. Even if the parameter has a decimal resolution value, the value is transferred with no decimal indication. For instance, if you want to set the parameter P100 with the value 5.0s, the value 50 should be written via network.

These parameters are used only if the device is programmed in parameter P710 to use options 3 through 6. According to the selected option, it will be available up to six words for writing by the network master.

NOTE!
The 0 (zero) value disables the word writing. The amount of input words, however, always remains the same as it was programmed in parameter P710.

P719 – DEVICENET NETWORK STATUS

Range:	0 = Offline 1 = Online, Not Connected 2 = Online, Connected 3 = Timed-out Connection 4 = Connection Failure 5 = Autobaud	**Default:** -
Properties:	RO	

Description:

It indicates the status of the DeviceNet network. The next table presents a brief description of those states.

Table 4.9: *Indication of the parameter P719*

Status	Description
Offline	Device without power supply or not online. Communication cannot be established.
Online, Not Connected	Device online, but not connected. The slave has successfully completed the MacID verification procedure. This means that the configured baud rate is correct (or it has been detected correctly in case of autobaud) and that there are no other network nodes with the same address. However, there is no communication with the master yet in this stage.
Online, Connected	The device is operational and in normal conditions. The master has allocated a set of I/O type connections with the slave. In this stage the effective exchange of data by means of I/O type connections occurs.
Timed-out Connection	One or more I/O type connections have expired.
Connection Failure	It indicates that the slave was not able to enter the network due to addressing problems or due to the occurrence of bus off. Make sure the configured address is not used by other device, verify if the chosen baud rate is correct and make sure there are no installation problems.
Autobuad	The equipment is executing the autobaud mechanism.

P720 – DEVICENET MASTER STATUS

Range:	0 = RUN 1 = IDLE	**Default:** -
Properties:	RO	

Description:

It indicates the DeviceNet network master status. It may be in operation mode (Run) or in configuration mode (Idle).

When in Run, reading and writing telegrams are processed normally and updated by the master. When in Idle, only the reading telegrams from the slaves are updated by the master. Writing, in this case, remains disabled.

When communication is disabled this parameter does not represent the actual state of the master.

5 SUPPORTED OBJECT CLASSES

Any DeviceNet equipment is modeled as a set of objects. The objects are responsible for defining the function that each device will have. In other words, depending on the objects the device implements, it may be a communication adapter, an AC/DC drive, a photoelectric sensor, etc. Mandatory and optional objects are defined for each Device Profile. The CFW300 frequency converter supports all mandatory classes defined for the AC/DC Device Profile. It also supports Manufacturer Specific classes. The following sections present detailed information about these classes.

5.1 IDENTITY CLASS (01H)

Provides general information about the device identity such as: VendorID, Product Name, Serial Number, etc. The following attributes are implemented:

Table 5.1: *: Identity Class instance attributes*

Attribute	Method	Name	Default	Description
1	GET	Vendor ID	355h	Manufacturer identifier
2	GET	Product Type	2h	Product Type
3	GET	Product Code		Product Code
4	GET	Vendor Revision		Vendor Revision
5	GET	Status		Device status
6	GET	Serial Number		Serial Number
7	GET	Product Name	CFW300	Product Name

5.2 MESSAGE ROUTER CLASS (02H)

Provides information on the explicit message router object. This class does not have any attribute implemented in the CFW300.

5.3 DEVICENET CLASS (03H)

This class is responsible for maintaining the configuration and the state of the physical connections of the DeviceNet node. The following attributes are implemented:

Table 5.2: *DeviceNet Class attributes*

Atributte	Method	Name	Min./Max	Default	Description
1	GET	Revision	1 - 65535		Revision of the DeviceNet Object Class Definition upon which the implementation is based.

Table 5.3: *DeviceNet Class instance attributes*

Atributte	Method	Name	Mín./Max	Default	Description
1	GET/SET	Mac ID	0 - 63	63	Node address.
2	GET/SET	Baud Rate	0 - 2	0	Communication baud rate
3	GET/SET	Bus-Off Interrupt	0 - 1	1	Bus-off reset
4	GET/SET	Bus-Off Counter	0 - 255		Bus-off counter
5	GET/SET	Allocation Information			Information about allocation byte

5.4 ASSEMBLY CLASS (04H)

This class is responsible for grouping several attributes in only one connection. Only the attribute Data (3) is implemented in the CFW300 (5.4).

Table 5.4: *Attributes of the instances of the Assembly class*

Atribute	Method	Name	Description
3	GET/SET	Data	Data contained in the assembly object

The Assembly class contains the following instances in the CFW300:

Table 5.5: *Instances of the Assembly class*

Instance	Size	Description
20	2 words	ODVA Basic Speed Control Output.
70	2 words	ODVA Basic Speed Control Output.
21	2 words	ODVA Extended Speed Control Output.
71	2 words	ODVA Extended Speed Control Output.
23	3 words	ODVA Extended Speed/Torque Control Output.
73	3 words	ODVA Extended Speed/Torque Control Output.
100	2 - 14 words	Manufacturer Specific Output
150	2 - 14 words	Manufacturer Specific Input

5.5 CLONNECTION CLASS (05H)

This class allocates and manages the internal resources associated with both I/O and Explicit Messaging Connections. The following methods are implemented:

5.5.1 Instance 1: Explicit Message

Table 5.6: *Classe Connection – Instância 1: Explicit Message*

Attribute	Method	Name	Description
1	GET	State	Object state
2	GET	Instance Type	I/O ou explicit
3	GET	transport Class trigger	Defines the connection behavior
4	GET	Produced Connection ID	CAN ID field for transmission
5	GET	Consumed Connection ID	CAN ID field value representing received msg
6	GET	Initial Comm. Charac.	Defines message groups related to this connection
7	GET	Produced Connection Size	Maximum size (bytes) of this transmission connection
8	GET	Consumed Connection Size	Maximum size (bytes) of this reception connection
9	GET/SET	Expected Packet Rate	Defines timing associated to this connection
12	GET	Watchdog Timeout Action	Action for inactivity/watchdog timeout
13	GET	Produced Connection Path Length	Number of bytes in the producer connection
14	GET	Produced Connection Path	Specifies the path of the data producer objects
15	GET	Consumed Connection Path Length	Number of bytes in the consumer connection
16	GET	Consumed Connection Path	Specifies the path of the data consumer objects
17	GET/SET	Production Inhibit Time	Defines the minimum time between new data production

5.5.2 Instance 2: Polled

Table 5.7: Classe Connection – Instância 2: Polled

Attribute	Method	Name	Description
1	GET	State	Object state
2	GET	Instance Type	I/O ou explicit
3	GET	transport Class trigger	Defines the connection behavior
4	GET	Produced Connection ID	CAN ID field for transmission
5	GET	Consumed Connection ID	CAN ID field value representing received msg
6	GET	Initial Comm. Charac.	Defines message groups related to this connection
7	GET	Produced Connection Size	Maximum size (bytes) of this transmission connection
8	GET	Consumed Connection Size	Maximum size (bytes) of this reception connection
9	GET/SET	Expected Packet Rate	Defines timing associated to this connection
12	GET	Watchdog Timeout Action	Action for inactivity/watchdog timeout
13	GET	Produced Connection Path Length	Number of bytes in the producer connection
14	GET	Produced Connection Path	Specifies the path of the data producer objects
15	GET	Consumed Connection Path Length	Number of bytes in the consumer connection
16	GET	Consumed Connection Path	Specifies the path of the data consumer objects
17	GET/SET	Production Inhibit Time	Defines the minimum time between new data production

5.5.3 Instance 4: Change of State/Cyclic

Table 5.8: Connection class – Instance 4: Change of State/Cyclic

Attribute	Method	Name	Description
1	GET	State	Object state
2	GET	Instance Type	I/O ou explicit
3	GET	transport Class trigger	Defines the connection behavior
4	GET	Produced Connection ID	CAN ID field for transmission
5	GET	Consumed Connection ID	CAN ID field value representing received msg
6	GET	Initial Comm. Charac.	Defines message groups related to this connection
7	GET	Produced Connection Size	Maximum size (bytes) of this transmission connection
8	GET	Consumed Connection Size	Maximum size (bytes) of this reception connection
9	GET/SET	Expected Packet Rate	Defines timing associated to this connection
12	GET	Watchdog Timeout Action	Action for inactivity/watchdog timeout
13	GET	Produced Connection Path Length	Number of bytes in the producer connection
14	GET	Produced Connection Path	Specifies the path of the data producer objects
15	GET	Consumed Connection Path Length	Number of bytes in the consumer connection
16	GET	Consumed Connection Path	Specifies the path of the data consumer objects
17	GET/SET	Production Inhibit Time	Defines the minimum time between new data production

5.6 MOTOR DATA CLASS (28H)

This class stores the information on the motor connected to the product. The following attributes have been implemented:

Table 5.9: Motor Data Class attributes

Attribute	Method	Name	Min/Max	Description
1	GET	Revision	1 - 65535	Revision of the Motor Data Object Class Definition upon which the implementation is based
2	GET	Max Instance	1 - 65535	Maximum instance number

Table 5.10: *Motor Data Class instance attributes*

Attribute	Method	Name	Min/Max	Unit	Default	Description
3	Get	Motor Type	0 - 10	-	7	0 = Non Standard Motor 1 = PM DC Motor 2 = FC DC Motor 3 = PM Synchronous Motor 4 = FC Synchronous Motor 5 = Switched Reluctance Motor 6 = Wound Rotor Induction Motor 7 = Squirrel Cage Induction Motor 8 = Stepper Motor 9 = Sinusoidal PM BL Motor 10 = Trapezoidal PM BL Motor
6	Get/Set	Rated Current	0-999.9	100mA		Nominal current
7	Get/Set	rated Voltage	0-600	V		Nominal voltage

5.7 MOTOR DATA CLASS (29H)

Responsible for modeling the drive management functions. The following attributes have been implemented:

Table 5.11: *Control Supervisor Class attributes*

Attribute	Method	Name	Min/Max	Description
1	GET	Revision	1 - 65535	Revision of the Control Supervisor Object Class Definition upon which the implementation is based
2	GET	Max Instance		Maximum instance number

Table 5.12: *Atributos da instância da classe Control Supervisor*

Attribute	Method	Name	Mín/Max	Default	Description
3	Get/Set	Run 1	0 - 1	-	Run Fwd
4	Get/Set	Run 2	0 - 1	-	Run Rev
5	Get/Set	NetCtrl	0 - 1	0	0 = Local control 1 = Remote control
6	Get	State	0 - 7	-	0 = Vendor specific 1 = Startup 2 = Not Ready 3 = Ready 4 = Enable 5 = Stopping 6 = Fault Stop 7 = Fault
7	Get	Running 1	0 - 1	0	0 = Other state 1 = (Enabled e Run1) or (Stopping e Running1) or (Fault Stop e Running1)
8	Get	Running 2	0 - 1	0	0 = Other state 1 = (Enabled e Run2) or (Stopping e Running2) or (Fault Stop e Running2)
9	Get	Ready	0 - 1	0	0 = Outro estado 1 = Ready or Enabled or Stopping
10	Get	Faulted	0 - 1	0	0 = No error 1 = Error
11	Get	Warning	0 - 1	0	0 = No warnings
12	Get/Set	Fault Reset	0 - 1	0	0 = No action 0 -> 1 = Error reset
15	Get	Ctrl from Net	0 - 1	0	0 = Local control 1 = Remote control

5.8 AC/DC DRIVE CLASS (2AH)

This class is responsible for modeling the management functions of the drive. The following attributes have been implemented: contains specific information of an AC/DC Drive such as operation mode, speed and torque ranges.

Table 5.13: AC/DC Drive Class attributes

Attribute	Method	Name	Min/Max	Description
1	GET	Revision	1 - 65535	Revision of the AC/DC Drive Object Class Definition upon which the implementation is based
2	GET	Max Instance		Maximum instance number

Table 5.14: AC/DC Drive Class instance attributes

Attribute	Method	Name	Mín/Max	Default	Description
4	Get/Set	NetRef 2	0 - 1	0	0 = Local reference 1 = Remote reference
6	Get	DriveMode	1 - 2	-	1 = Speed control (open loop) 2 = Speed control (closed loop)
7	Get	Speed Actual	0 - 9999		Actual speed (best approximation)
8	Get/Set	Speed Ref	0 - 9999	0	Speed reference

NOTE!
The CFW300 will work in speed mode independently of the content of the DriveMode attribute.

5.9 MESSAGE ROUTER CLASS (2BH)

This class is responsible for managing the reception of acknowledgment messages.

Table 5.15: Acknowledge Handler Class instance attributes

Attribute	Method	Name
1	GET/Set	Acknowledge Timer
2	GET	Retry Limit
3	GET	COS Production Connection Instance

5.10 MANUFACTURER APECIFIC CLASSES

The Manufacturer Specific Classes are used for mapping all CFW300 parameters. These classes allow the user to read from and write to any parameter through the network. The Manufacturer Specific Classes use DeviceNet explicit messages. There are separate ranges for each group of parameters, as presented in Table 5.16:

Table 5.16: Manufacturer Specific Classes

Class	Name	Range
Class 100 (64h)	VENDOR CLASS F1	Parameters 000 - 099
Class 101 (65h)	VENDOR CLASS F2	Parameters 100 - 199
Class 102 (66h)	VENDOR CLASS F3	Parameters 200 - 299
Class 103 (67h)	VENDOR CLASS F4	Parameters 300 - 399
Class 104 (68h)	VENDOR CLASS F5	Parameters 400 - 499
Class 105 (69h)	VENDOR CLASS F6	Parameters 500 - 599
Class 106 (6Ah)	VENDOR CLASS F7	Parameters 620 - 699
Class 107 (6Bh)	VENDOR CLASS F8	Parameters 700 - 799
Class 108 (6Ch)	VENDOR CLASS F9	Parameters 800 - 899
Class 109 (6Dh)	VENDOR CLASS F10	Parameters 900 - 999
Class 110 (6Eh)	VENDOR CLASS F11	Parameters 1000 - 1099
Class 111 (6Fh)	VENDOR CLASS F12	Parameters 1100 - 1199

Table 5.17: Parameters of the Manufacturer Specific classes

Parameter	Class	Instance	Attribute
P0000	Class 100 (64h)	1	100
P0001	Class 100 (64h)	1	101
P0002	Class 100 (64h)	1	102
...	...	...	...
P0100	Class 101 (65h)	1	100
P0101	Class 101 (65h)	1	101
P0102	Class 101 (65h)	1	102
...	...	...	...
P0200	Class 102 (66h)	1	100
P0201	Class 102 (66h)	1	101
P0202	Class 102 (66h)	1	102
...	...	...	...
P0300	Class 103 (67h)	1	100
P0301	Class 103 (67h)	1	101
P0302	Class 103 (67h)	1	102
...	...	...	...

NOTE!

- The CFW300 uses only instance 1 for Manufacturer Specific Classes.
- In order to access the parameters through the Manufacturer Specific Classes, add the value 100 to the last two digits of any parameter. This new resulting number is known as attribute.

For instance:

Parameter 23: class 64h, instance 1, attribute 123. This path gives access to P0023.
Parameter 100: class 65h, instance 1, attribute 100. This path gives access to P0100.
Parameter 202: class 66h, instance 1, attribute 102. This path gives access to P0202.

6 FAULTS AND ALARMS

A133/F233 - CAN INTERFACE WITHOUT POWER SUPPLY

Description:

It indicates that the CAN interface does not have power supply between the pins 25 and 29 of the connector.

Actuation:

In order that it be possible to send and receive telegrams through the CAN interface, it is necessary to supply external power to the interface circuit.

If the CAN interface is connected to the power supply and the absence of power is detected, the alarm A133 – or the fault F233, depending on the P313 programming, will be signalized through the HMI. If the circuit power supply is reestablished, the CAN communication will be reinitiated. In case of alarms, the alarm indication will also be removed from the HMI.

Possible Causes/Correction:

- Measure the voltage between the pins 6 and 10 of the CAN interface connector.
- Verify if the power supply cables have not been changed or inverted.
- Make sure there is no contact problem in the cable or in the CAN interface connector.

A134/F234- BUS OFF

Description:

The bus off error in the CAN interface has been detected.

Actuation:

If the number of reception or transmission errors detected by the CAN interface is too high , the CAN controller can be taken to the bus off state, where it interrupts the communication and disables the CAN interface.

In this case the alarm A134 – or the fault F234, depending on the P0313 programming, will be signalized through the HMI. In order that the communication be reestablished, it will be necessary to cycle the power of the product, or remove the power supply from the CAN interface and apply it again, so that the communication be reinitiated.

Possibles Causes/Correction:

- Verify if there is any short-circuit between the CAN circuit transmission cables.
- Verify if the cables have not been changed or inverted.
- Verify if all the network devices use the same baud rate.
- Verify if termination resistors with the correct values were installed only at the extremes of the main bus.
- Verify if the CAN network installation was carried out in proper manner.

A136/F236 - IDLE MASTER

Description:

It is the alarm that indicates that the DeviceNet network master is in the Idle mode.

Actuation:

It acts when the CFW300 detects that the network master went to the Idle mode. In this mode, only the variables read from the slave continue being updated in the memory of the master. None of the commands sent to the slave is processed.

In this case the alarm A136 – or the fault F236, depending on the P0313 programming, will be signalized through the HMI. In case of alarms, If the master is set in the Run mode again (normal equipment operation status), the alarm indication will be removed from the HMI.

Possibles Causes/Correction:

- Adjust the switch that commands the master operation mode for execution (Run) or set the correspondent bit in the configuration word of the master software. In case of doubts, referrer to used master documentation.

A137/F237 - DEVICENET CONNECTION TIMEOUT

Description:

It is the alarm that indicates that one or more DeviceNet I/O connections have expired.

Actuation:

It occurs when, for any reason, after the cyclic communication of the master with the product is started, this communication is interrupted.

In this case the alarm A137 – or the fault F237, depending on the P0313 programming, will be signalized through the HMI. In case of alarms, if the connection with the master is reestablished, the alarm indication will be removed from the HMI.

Possibles Causes/Correction:

- Check the status of the network master.
- Check the network installation, broken cable or failed/bad contact in the network connections.

WEG Drives & Controls - Automação 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