

### Instruction manual

EXP-CAN/DN-ADV CANopen / DeviceNet interface expansion card



#### Contents

| Reinforced insulation                                   | 2  |
|---|----|
| Introduction  | 2  |
| Mounting  | 2  |
| Connections   |    |
| Leds  |    |
| Optional card recognization                             |    |
| 1.0 CANopen interface                                   |    |
| 1.1 CANopen functions                                   |    |
| 1.2 CANopen management                                  |    |
| 1.3 Process Data Channel Control                        | 9  |
| 1.4 SDO management                                      |    |
| 1.5 Alarms  |    |
| 1.6 Configuration example                               |    |
| 1.7 Connecting the master control panel to ADV200 nodes |    |
| 2.0 Operation according to the DS402 profile            | 23 |
| 3.0 DeviceNet Interface                                 | 25 |
| 3.1 General description of DeviceNet                    |    |
| 3.2 DeviceNet function                                  |    |
| 3.3 Object description                                  | 25 |
| 3.4 Data transfert via Explicit Messaging               |    |
| 3.5 Polling function                                    |    |
| 3.6 Devicenet Interface configuration                   |    |
| 3.7 Alarms<br>3.8 Process Data Channel Control          |    |
| 3.9 Configuration example                               |    |
| 5 I   |    |
| References  |    |



Use only the supplied screws!

#### **Reinforced insulation**

PELV (Protective Extra Low Voltage) EN 61800-5-1.

#### Introduction

This manual describes the EXP-CAN/DN-ADV option card aimed at connecting the ADV200 series Drives to CANopen or DeviceNet networks. It is possible to use only one field bus expansion card per Drive.

This manual is addressed to desing engineers and technicians responsible for the maintenance and the commissioning of CANopen and DeviceNet systems. A CANopen and DeviceNet basic knowledge is therefore required; for further information see the following manuals:

- CANopen CAL-Base COMMUNICATION PROFILE for Industrial Systems; CiA Draft Standard 301 Version 4.2 Date 13 February 2002 by CAN in Automation e. V.
- DeviceNet Specifications. Volume 1 DeviceNet Communication Model and Protocol (Issued by ODVA).
- DeviceNet Specifications. Volume 2 DeviceNet Device Profiles and Object Library (Issued by ODVA).

#### Mounting

Refer to ADV200 Quick Start up manual, chapter "**Installation of optional** cards": the card must be inserted on slot 3.

#### Connections

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
|   |   |   |   |   |

Wire sizes: 0.2 ... 2.5 mm<sup>2</sup> (AWG 24 ... 12)

For the Bus connection use a shielded loop, type as stated by the CANopen or DeviceNet specification.

The Bus connection is provided via a shielded loop (**type as stated by the CANopen or DeviceNet specification**) to be placed far from the power cables, with a minimum distance of 20 cm. The cable shielding must be continuous and grounded at a single point.

In addition, the unipotential connection of ADV200 CAN bus nodes must be ensured with card terminal 1 (V- / CAN\_GND).

| PIN   | DeviceNet  | CANopen                | Function                      | Max    |  |  |
|---|--|------------------------|-------------------------------|--------|--|--|
| BUS terminal : allows to connect the card to the CANopen or DeviceNet network   |  |                        |                               |        |  |  |
|   | INTERNAL SUPPLY EXTERNAL SUPPLY<br>(Connection not insulated) (Connection optocoupled) |                        |                               |        |  |  |
| I       I |  |                        |                               |        |  |  |
| 1   | V-   | CAN_GND                | Ground / OV / V-              | OV     |  |  |
| 2   | 2 CAN_L CAN_L CAN_L busline (dominant low)   |                        |                               |        |  |  |
| 3   | DRAIN  | IN CAN_SHLD CAN shield |                               |        |  |  |
| 4   | CAN_H  | CAN_H                  | CAN_H busline (dominant high) |        |  |  |
| 5 V+ CAN_V+ CAN external positive supply (dedicated for supply of transceiver and optocouplers)   |  |                        |                               | 11 30V |  |  |
|   |  |                        |                               | ENL O  |  |  |

#### Important!

#### Note on terminating resistor :

Caution

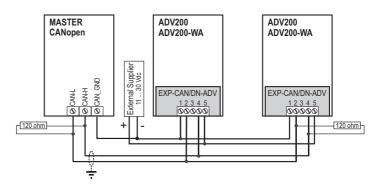
The first and last node of the CAN line must have a 120 ohm resistance between pins 2 and 4.

- if using an external power supply the mains is galvanically isolated,

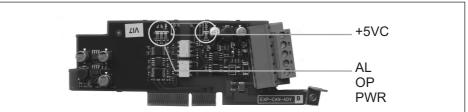
- if derived from the regulation card (terminals C3/S3, +24Vout/0V24 out) the mains is NOT galvanically isolated. The maximum available current value must not be exceeded (total max = 150 mA @ 24V), including any other expansion cards.

 - on CANopen networks in general, the CAN\_GND connection must apply to all participating nodes unless the CANopen network is completely galvanically isolated. If a node (master or slave) does not have the CAN\_GND connection, or if the connection is not used, the user must ensure maximum rejection of line noise for all network participants.

The connection among the single cards is performed with a shielded cable as shown in the following figure.



Leds



| LEDs                   |         | CANopen  | DeviceNet                    |
|------------------------|---------|--|------------------------------|
| AL (red)<br>OP (green) |         | ON: the CANopen interface is in alarm condition                | DeviceNet connection status, |
|                        |         | ON: the CANopen interface is in Operational condition          | see next table               |
| PWR                    | (green) | The led is ON when the expansion card is powered and<br>active |                              |
| +5VC                   | (green) | The led is ON when the optoinsulated CAN                       | I node is correctly powered  |

| OP                                 | AL    | Meaning   |
|------------------------------------|-------|---|
| ON                                 | ON    | Card power-up   |
| BLINK                              | BLINK | Self test and Duplicate MAC-ID check is running         |
| BLINK                              | OFF   | Master configuration and/or I/O Polling wait not active |
| ON                                 | OFF   | I/O Polling active, operative status                    |
| OFF                                | BLINK | Minor fault (DUP MAC-ID fail, bus-off, bus-loss)        |
| OFF                                | ON    | Major fault (configuration error, internal error)       |
| OFF OFF DeviceNet not configurated |       | DeviceNet not configurated                              |

#### **Optional card recognization**

|   | T+         T-         EN         LOC         ILlim         n:0         AL           MESSAGE         0 1 |                          |
|---|---|--------------------------|
|   | Option detect slot 3  |                          |
|   | Code: 0004H-4   |                          |
| 1 |   |                          |
|   | T+ T EN LOC ILim n:0 AL   | T+ T- EN LOC ILim n:O AL |
|   | 01 MONITOR  | 0 2 . 1 7 PAR : 5 3 4    |
|   | 02 DRIVE INFO   | Slot 3 card type         |
|   | 03 STARTUP WIZARD   | Can/Dnet                 |
| 2 | 04 DRIVE CONFIG   | Value 4                  |

1 - At power-on, the drive recognizes the presence of optional card in the expansion slot 3, this message is shown on the display.

2- On 02 DRIVE INFO menu, select the PAR 534  ${\rm Slot}$  3 card type to read the recognized card type.

| Value | Description | Card type      |
|-------|-------------|----------------|
| 0     | None        | -              |
| 4     | Can/Dnet    | EXP-CAN/DN-ADV |
| 255   | Unknown     | -              |

#### 1.0 CANopen interface

CANopen is a communication profile for CAL-based industrial systems. The reference document is the CANopen CAL-Base COMMUNICATION PROFILE for Industrial Systems; CiA Draft Standard 301 Version 4.2 Date 13 February 2002 by CAN in Automation e. V.

The drive also implements the DS402 profile according to the CANopen Device Profile Drives and Motion Control v4.02 specification.

The CAN protocol (ISO 11898) is CAN2.0A with an 11-bit identifier.

The integrated CANopen interface is developed as a "Minimum Capability Device". The data exchange is cyclic, the Master unit reads the Slave input data and writes the Slave output data.

#### 1.1 CANopen functions

This chapter describes the controlled functions of the CANopen communication profile.

#### Main features:

- 1) The "Minimum Boot-up" is managed; the "Extended Boot-up (CAL)" is not managed.
- 2) The SYNC function is implemented.
- 3) The PDO asynchronous assignment and RTR are managed.
- 4) The Node Guarding and HearthBeat protocols are managed.
- 5) The emergency message is managed ("EMERGENCY").
- 6) The Dynamic ID distribution function (DBT slave) is not managed.
- 7) A "Pre-Defined Master/Slave connection" is implemented to simplify the Master tasks during the initialization phase. "Inhibit-Times" (in units of 100 uS) can be modified up to a value of 1 min.
- 8) The high-resolution synchronization is not supported.
- 9) "TIME STAMP" is not managed.
- 10) On the access of the structured parameters, the OFFhex option subindex (access to the whole object) is not managed.
- In order to obtain a higher efficiency level, only the "Expedited" data transfer (max. 4 Bytes) of the SDO services is managed.

#### 1.1.1 Pre-defined Master/Slave Connection

The "Pre-defined Master/Slave connection" allows a peer-to-peer communication between one Master and 127 Slaves; the Broadcast address is zero.

#### 1.1.2 NMT Services (Network Management)

The NMT "mandatory" services are:

- Enter\_Pre-Operational\_State CS = 128
- Reset\_Node CS = 129
- Reset\_Communication CS = 130

Being that the "Minimum Boot-up" is used, also the following NMT services are managed:

- Start\_Remote\_Mode CS = 1
- Stop\_Remote\_Mode CS = 2

The COB-ID  $^{\ast}$  of an initialization NMT service is always at 0; CS is the Command Specifier defining the NMT service.

#### 1.1.3 Initialization

The ADV drive supports the Node Guarding and HeartBeat mechanism. The Node Guarding configuration can be performed through the master via the standard Object Dictionary elements (1006h, 100Ch, 100Dh) and the 1016h, 1017h objects for HeartBeat.

The drive checks the master functioning conditions through the Life Guarding. If the check fails, the drive enables the "Buss Loss" alarm. The Life Guarding threshold can be calculated as follows:

| Value                    | Condition   |
|--------------------------|---|
| 60ms<br>SYNC_PERIOD (*)  | Default. No parameterization of the Node Guarding.  |
| LIFE_TIME_FACTOR         | Use of the synchronous mode. If not stated by the master, the Life_Time_Factor default value is equal to 3. |
| NODE_GUARDING_PERIOD (*) | set by the master   |
| LIFE_TIME_FACTOR         | If not otherwise stated, the value is equal to 3  |

#### 1.1.4 Communication objects

This chapter describes the communication objects of the CANopen protocol; they are managed by the interface card.

The managed communication objects are:

- 1) 1 SDO reception Server.
- 2) 1 SDO transmission Server.
- 3) 4 reception PDOs.
- 4) 4 transmission PDOs.
- 5) 1 Emergency Object.
- 6) 1 Node Guarding Life Guarding.
- 7) 1 SYNC object.

The following table lists the used communication objects in ascending order downward, and the Message Identifier; the "Resulting COB-ID" is obtained by adding the Node-ID (card address) to the number.

| OBJECT             | MESSAGE ID                         |
|--------------------|------------------------------------|
| NODE GUARDING & HB | 1792 700h+Nodeld                   |
| 1st SDO rx         | 1536 600h+Nodeld                   |
| 1st SDO tx         | 1408 580h+Nodeld                   |
| 1st PDO tx         | 384 180h+Nodeld                    |
| 1st PDO rx         | 512 200h+Nodeld                    |
| 2nd PDO tx         | 640 280h+Nodeld                    |
| 2nd PDO rx         | 768 300h+Nodeld                    |
| 3st PDO tx         | 384 380h+Nodeld                    |
| 3st PDO rx         | 512 400h+Nodeld                    |
| 4th PDO tx         | 640 480h+Nodeld                    |
| 4th PDO rx         | 768 500h+Nodeld                    |
| EMERGENCY          | 220 80h+Nodeld                     |
| SYNC               | 128 80h                            |
| NMT                | Network Management                 |
|                    | Table 1 4 1: Communication Objects |

Table 1.4.1: Communication Objects

#### 1.1.5 Object Dictionary Elements

The Object Dictionary is accessible from a master CANopen.

The following table shows the communication objects used and accessibility with master CANopen.

| Index (hex) | Name                            |
|-------------|---------------------------------|
| 1000        | Device Type                     |
| 1001        | Error Register                  |
| 1002        | Manufacturer status register    |
| 1005        | COB-ID SYNC Message             |
| 1006        | Communication cycle period      |
| 1008        | Manufacterer Device Name        |
| 1010        | Store parameter                 |
| 1009        | Manufacturer Hardware Version   |
| 100A        | Manufacturer Software Version   |
| 100C        | Guard Time                      |
| 100D        | Life Time Factor                |
| 1014        | COB-ID Emergency                |
| 1016        | HeartBeat time consumer         |
| 1017        | HeartBeat time producer         |
| 1018        | Identity object                 |
| 1400        | 1st Receive PDO                 |
| 1401        | 2nd Receive PDO                 |
| 1402        | 3rd Receive PDO                 |
| 1403        | 4th Receive PDO                 |
| 1600        | Receive PD01 mapping parameter  |
| 1601        | Receive PD02 mapping parameter  |
| 1602        | Receive PD03 mapping parameter  |
| 1603        | Receive PD04 mapping parameter  |
| 1A00        | Transmit PD01 mapping parameter |
| 1A01        | Transmit PD02 mapping parameter |
| 1A02        | Transmit PD03 mapping parameter |
| 1A03        | Transmit PDO4 mapping parameter |
| 1800        | 1st Transmit PDO                |
| 1801        | 2nd Transmit PDO                |
| 1802        | 3rd Transmit PDO                |
| 1803        | 4th Transmit PDO                |

Table 1.5.1: Objects used by the CANopen communication profile

The objects shown in **bold** in the table allow writing of the parameters assigned with the exchange of data in the PDO.

The allocation criterion is variable, and depends on the size (in bytes) of the parameter exchanged.

#### 1.1.6 RX PDO Entries

The structure of the PDO Communication Parameter (index 1400h, 1401h) is:

1) Subindex 0 (Number of supported entries) = 2

- 2) The structure of Subindex 1 (COB-ID used by the PDO) is:
- Bit 31 (valid/invalid PDO) can be set via SDO.
- Bit 30 (RTR Remote Transmission Request) = 0 because this function is not supported.
- Bit 29 = 0 because the 11-bit ID is used (CAN 2.0A).
- Bits 11-28 are not used.
- Bit 0-10 COB-ID (see table 1.4.1).
- Cyclic-synchronous Subindex 2 (Transmission Type), or synchronous according to the master performed setting (1 if SYNC has been foreseen, 254...255 if asynchronous). If not stated, the synchronous mode is active.

#### 1.1.7 TX PDO Entries

The structure of the PDO Communication Parameter (index 1800h, 1801h) is:

- 1) Subindex 0 (Number of supported entries) = 3
- 2) The structure of Subindex 1 (COB-ID used by the PDO) is:
- Bit 31 (valid/invalid PDO) can be set via SDO.
- Bit 30 (RTR Remote Transmission Request) = 0 because this function is not supported.
- Bit 29 = 0 because the 11-bit ID is used (CAN 2.0A).
- Bits 11-28 are not used.
- Bit 0-10 COB-ID (see table 1.4.1).
- Cyclic-synchronous Subindex 2 (Transmission Type), or synchronous according to the master performed setting (1 if SYNC has been foreseen, 254...255 if asynchronous). If not stated, the synchronous mode is active.
- 4) Inhibit timee

#### 1.1.8 SDO Entries

Only the "Expedited" data transfer mode (max. 4 Bytes) is used.

- Subindex 0 (Number of supported entries) = 3 because the device is a Server of the SDO service.
- 2) The structure of the Subindex 1 and 2 (COB-ID used by the SDO) is:
- Bit 31 (valid/invalid SDO); it is equal to 1 because just the Default SDOs are used.
- Bit 30 reserved = 0.
- Bit 29 = 0 because the 11-bit ID is used (CAN 2.0A).
- Bits 11-28 are not used.
- Bit 0-10 COB-ID (see table 1.4.1).

The element "node ID of SDO's client resp. server" is not supported because just the Default SDOs are used.

#### 1.1.9 COB-ID SYNC Entries

The structure of the 32 bits contained in the COB-ID SYNC communication parameter is:

- Bit 31 = 1 because the CANopen interface card is a "consumer" of SYNC messages.
- Bit 30 = 0 because the interface card does not create SYNC messages.
- Bit 29 = 0 because the 11-bit ID is used (CAN 2.0A).
- Bits 11-28 are not used.
- Bit 0-10 COB-ID (see table 1.4.1).

#### 1.1.10 COB-ID Emergency

The structure of the 32 bits contained in the COB-ID Emergency Message communication parameter is:

- Bit 31 = 0 because the CANopen interface card is not a "consumer" of Emergency messages.
- Bit 30 = 0 because the interface card creates Emergency messages.
- Bit 29 = 0 because the 11-bit ID is used (CAN 2.0A).
- Bits 11-28 are not used.
- Bit 0-10 COB-ID (see table 1.4.1).

#### 1.2 CANopen management

The user interface of the CANopen protocol is performed via the drive parameters. The parameters are controlled via hierarchical menus. All the writing parameters referring to the field bus are active only after the drive reset. Here following is a list of drive parameters useful to control the CANopen protocol.

To activate the EXP-CAN-ADV card, set parameter PAR 4000 **Fieldbus type** to CANopen or DS402.

The following parameters are available in the COMMUNICATION->FIELDBUS CONFIG menu:

| PAR  | Nome Par             | Туре            | Default value | Attr      |
|------|----------------------|-----------------|---------------|-----------|
| 4004 | Fieldbus baudrate    | Enum            | None          | Write     |
| 4006 | Fieldbus address     | 2 byte unsigned | 0             | Write     |
| 4010 | Fieldbus M->S enable | Enum            | On            | Write     |
| 4012 | Fieldbus alarm mode  | 2 byte unsigned | 0             | Write     |
| 4014 | Fieldbus state       | Enum            | Stop          | Read only |

- Fieldbus baudrate = Sets the network baud rate. Values available for CANopen: 125k, 250k, 500k, 1M
- Fieldbus address = address of this slave node in the network, accepted values from 1 to 127
- Fieldbus M->S enable = if set to Off data in the RPDOs are not processed by the drive
- Fieldbus alarm mode = if set to 1 the drive generates Opt Bus Fault errors relating to the loss of communication (Bus Loss) even when the drive is not enabled.
- Fieldbus state = state of the communication for this node on the CANopen network: Stop, Pre-Operational, Operational.

#### **1.3 Process Data Channel Control**

This function allows to allocate the drive parameters or application variables to the Process Data Channel data.

As for the CANopen protocol, the PDC is performed via the PDO messages ((Process data Object).

The CANopen protocol uses a number of words for the Process Data Channel (abbr. PDC Process Data Channel), which can always be set.

The fieldbus Process Data Channel configuration is the following:

Data 0

Data...

Data n

The drive can both read and write the Process Data Channel data. A datum can be made both of 2 and 4 bytes. The word "data" refers to any quantity of bytes included between 0 and 16, if the byte total number required is not higher than 32.

*Example:* It is possible to have:

- from 0 to 16 data with 2 bytes
- 1 datum with 4 bytes + from 0 to 14 data with 2 bytes
- 2 data with 4 bytes + from 0 to 12 data with 2 bytes

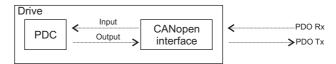
...

8 data with 4 bytes

The data exchanged via the PDC can be of two types:

- drive parameters
- variables of an MDPIc application. The use of the MDPIc variables is described in par. 1.3.1 and 1.3.2.

The master writes the data defined as PDC input and reads the data defined as PDC output.



#### 1.3.1 PDC Input Configuration (FB XXX MS Parameter)

Data exchanged in RPDOs are configured using the parameters in the COMMU-NICATION->FIELDBUS M2S menu (refer to the drive manual).

Data mapping in PDOs is performed on the basis of the data format set in **Fieldbus M->Sn sys** according to the following rules:

- PDOs are filled starting from RPDO1
- When the PDO contains 4 words it is full and the next RPDO is filled with a maximum of 4 PDOs.
- 32-bit data (long or float) cannot be split among PDOs, they must be placed inside the PDO (an alarm is generated).
- PDOs containing fewer than 4 words can be created, using Fieldbus M->Sn ipa = 0 but assigned (Fieldbus M->Sn sys other than Not Assigned, Fill16 or Fill32) after an assigned datum.

(N.B.: if assigned as Fill16 or Fill32, the datum is included in the PDO anyway)

- At the first Fieldbus M->Sn sys = Not Assigned parameter the PDOs are complete. The size of the last PDO thus depends on the data that have been assigned.
- Example: 2-word RPD01 and 2-word RPD02: Fieldbus M->S1 ipa = 610 (Ramp ref 1 src)
   Fieldbus M->S1 sys = Eu
   Fieldbus M->S2 ipa = 4452 (Word decomp src)
   Fieldbus M->S2 sys = Count 16
   Fieldbus M->S3 ipa = 0
   Fieldbus M->S3 sys = Fill 32

Fieldbus M->S4 ipa = 3660 (Compare input 1 src) Fieldbus M->S4 sys = Count32 Fieldbus M->S5 sys = Not Assigned

#### 1.3.2 PDC Output Configuration (FB XXX SM Parameter)

Data exchanged in RPDOs are configured using the parameters in the COMMU-NICATION->FIELDBUS S2M menu (refer to the drive manual).

Data mapping in PDOs is performed on the basis of the data format set in **Fieldbus M->Sn sys** according to the following rules:

- PDOs are filled starting from TPDO1
- When the PDO contains 4 words it is full and the next TPDO is filled with a maximum of 4 PDOs.
- 32-bit data (long or float) cannot be split among PDOs, they must be placed inside the PDO (an alarm is generated).
- PDOs with fewer than 4 words can be created, using Fieldbus S->Mn ipa = 0 but assigned (Fieldbus M->Sn sys other than Not Assigned, Fill16 or Fill32) after an assigned datum.
- At the first Fieldbus S->Mn sys = Not Assigned parameter the PDOs are complete. The size of the last PDO thus depends on the data that have been assigned.

#### 1.3.3 Use of the PDC in MDPIc Applications

It is possible to configure both the PDC input and output data in order to allow the data direct access via the MDPIc application code.

For read data simply set **Fieldbus M->Sn sys** to MDPLC16 or MDPLC32, leaving **Fieldbus M->Sn ipa** = 0.

The MDPLC application can now read the incoming datum directly from the **Fieldbus M->Sn mon** parameter.

Write data are configured by setting Fieldbus S->Mn ipa = [4184 + (n-1)\*10] (Dig Fieldbus S->Mn,  $1 \le n \le 16$ ).

Fieldbus S->Mn sys is automatically set to MDPLC. The application writes the datum in the Dig Fieldbus S->Mn parameter to send it to the bus.

#### 1.4 SDO management

The SDO service is always available.

The drive parameters can be accessed via the "MSPA" Manufacturer Specific Profile Area (2000hex< index <5FFFhex).

The index to be shown in the SDO command to access a drive parameter is obtained via the following rules:

SDO index = PAR + 2000h

SDO subindex = 1

The Data field must contain the value of the drive parameter.

Example:

Writing the value 1000 in the PAR 600 Dig Ramp ref 1 parameter (258hex).

The following information is required:

- The SDO index resulting from the formula is: 2000hex + 258hex = 2258h
- 2) The value to be written is 1000, corresponding to 03E8 hex.

| Ind | lex | Subindex |     |                       |                       |     |
|-----|-----|----------|-----|-----------------------|-----------------------|-----|
| 22h | 58h | 01h      | E8h | 03h                   | 00h                   | 00h |
|     |     | Subindex |     | Drive parameter value | to be assigned to SDO |     |

In case an error occurs during the parameter reading or setting, the CANopen interface sends an Abort domain transfer message; the value of Application-errorcodes has the following meanings:

| Error class | Error code | Additional<br>code (hex) | Meaning                                      |
|-------------|------------|--------------------------|--|
| 6           | 0          | 0                        | Parameter doesn't exist                      |
| 8           | 0          | 22                       | Acces failed because of present device state |
| 6           | 1          | 2                        | Read/Write only error                        |
| 8           | 0          | 0                        | Generic error                                |
| 6           | 9          | 32                       | Minimum value                                |
| 6           | 9          | 31                       | Maximum value                                |
| 5           | 4          | 0                        | SDO time_out                                 |
| 5           | 4          | 1                        | Invalid command                              |
| 3           | 9          | 30                       | Invalid value                                |

#### 1.5 Alarms

#### **Fieldbus alarms**

The bus failure is signaled via the "Opt Bus Fault" alarm. As for CANopen, the possible failure causes are:

- "Bus-off" condition of the CAN line;
- th drive has not been enabled in the "Operational" mode;
- exceeding of monitoring limit (heartbeat, node guard, communication cycle period).
- drive configuration error

This alarm becomes active only when the drive is enabled.

If ON, the PAR 4014 **Fieldbus alarm mode** parameter enables the generation of the "Field bus failure" alarm also when the drive is disabled.

| Code     | Cfg | Description  | Actions  |  |  |  |  |
|----------|-----|--|--|--|--|--|--|
| 0        |     | Bus Loss   | Check line for noise, terminations , problems with cabling |  |  |  |  |
| FF01     | *   | Fieldbus type does not match expansion card        | Verify if EXP-CAN-ADV card is properly<br>installed        |  |  |  |  |
| FF02     | *   | Wrong baudrate selected                            | Check "Fieldbus baudrate" is one of 125k, 250k, 500k, 1M   |  |  |  |  |
| FF03     | *   | Invalid address for node                           | Check "Fieldbus address"                                   |  |  |  |  |
| FF04     | *   | Error initializing CAN interface                   | Internal error, contact manufacturer                       |  |  |  |  |
| FF14FF23 | *   | Wrong object selected for mapping in channel M2S n | Check "Fieldbus M->Sn Dest                                 |  |  |  |  |

| Code     | Cfg | Description   | Actions   |  |  |  |  |
|----------|-----|---|---|--|--|--|--|
| FF24FF33 | *   | More than 1 Src pointing to M2S Channel n   | Check for multiple destinations on<br>"Fieldbus M->Sn Dest"                         |  |  |  |  |
| FF34FF43 | *   | M2S Channel n , data size is wrong<br>(16 bits on 32 bits or 32 bits on 16<br>bits parameter) | Check "Fieldbus M->Sn sys"  |  |  |  |  |
| FF44FF53 | *   | Invalid parameter in channel S2M n  | Check "Fieldbus S->Mn src"  |  |  |  |  |
| FF54FF63 | *   | S2M Channel n , data size is wrong<br>(16 bits on 32 bits or 32 bits on 16<br>bits parameter) | Check "Fieldbus S->Mn sys"  |  |  |  |  |
| FF64FF73 | *   | Wrong object selected for mapping in channel S2M n  | Check "Fieldbus S->Mn src"  |  |  |  |  |
| FF74FF83 | *   | M2S Channel n : too many words in PDC   | "Fieldbus M-Sn dest" & "Fieldubs<br>M->Sn sys" address more than 16<br>words in PDC |  |  |  |  |
| FF84FF93 | *   | S2M Channel n : too many words in PDC   | "Fieldbus S->Mn src" & "Fieldubs<br>S->Mn sys" address more than 16<br>words in PDC |  |  |  |  |
| FFB4FFC3 | *   | Internal database error on channel n  | Internal error, contact manufacturer  |  |  |  |  |
| 8110     |     | CAN msg overflow  | Too many packets for selected<br>baudrate   |  |  |  |  |
| 8130     |     | LifeGuard/HeartBeat error   | Software timeout from master  |  |  |  |  |
| FFC5     |     | Wrong NMT message length  | Check NMT packets   |  |  |  |  |
| FFC6     |     | Invalid NMT command   | Check NMT packets   |  |  |  |  |
| FFC7     |     | CAN bus off   | Check line state for problems   |  |  |  |  |
| 8100     |     | CAN bus off   | Check line state for hardware<br>problems   |  |  |  |  |

**Drive alarm messages** Drive alarms are managed by means of an Emergency message containing the error code relating to the alarm that is generated, according to the table below:

| Selection      | Code   |  |  |
|----------------|--------|--|--|
| No alarm       | 0x0000 |  |  |
| Overvoltage    | 0x3210 |  |  |
| Undervoltage   | 0x3220 |  |  |
| Ground fault   | 0x2110 |  |  |
| Overcurrent    | 0x2310 |  |  |
| Desaturation   | 0x2130 |  |  |
| MultiUndervolt | 0xFF06 |  |  |
| MultiOvercurr  | 0xFF07 |  |  |
| MultiDesat     | 0xFF08 |  |  |
| Heatsink OT    | 0x4210 |  |  |
| HeatsinkS OTUT | 0x4310 |  |  |
| Intakeair OT   | 0x4130 |  |  |
| Motor OT       | 0xFF0C |  |  |
| Drive overload | 0x8311 |  |  |

| Selection      | Code   |
|----------------|--------|
| Motor overload | 0x7121 |
| Bres overload  | 0x7112 |
| Phaseloss      | 0xFF10 |
| Opt Bus fault  | 0xFF11 |
| Opt 1 IO fault | 0xFF12 |
| Opt Enc fault  | 0x3130 |
| External fault | 0x9000 |
| Speed fbk loss | 0x7310 |
| Overspeed      | 0x8400 |
| Plc1 fault     | 23     |
| Plc2 fault     | 24     |
| Plc3 fault     | 25     |
| Plc4 fault     | 26     |
| Plc5 fault     | 27     |
| Plc6 fault     | 28     |
| Plc7 fault     | 29     |
| Plc8 fault     | 30     |
| Emg stop alarm | 31     |
| Watchdog       | 32     |
| Trap error     | 33     |
| System error   | 34     |
| User error     | 35     |
| Power down     | 36     |
| Speed ref loss | 37     |
| Not Used1      | 38     |
| Opt 2 IO fault | 39     |
| Not Used2      | 40     |
| Not Used3      | 41     |
| Not Used4      | 42     |
| Not Used5      | 43     |
| Not Used6      | 44     |
| Param error    | 45     |

#### 1.6 Configuration example

This chapter provides an example of how to configure the parameters of ADV200 drives so that they can be read and written by a CANopen master via the processing channels (PDO). See the chapter 1.4 for the configuration channels (SDO). The paragraph 1.6.1 provides the information required on a CANopen master controlling a machine. The paragraph 1.6.2 contains basic information for programming the ADV200 drive starting from the factory settings.

#### 1.6.1 CANopen Master

This section contains an example of data exchange seen from the master side. This is the data normally contained in the machine specifications in the case of applications controlled by a CANopen master.

#### 1.6.1.1 Description of Master -> Slave PDO Communication

There are two parameters to be written via the processing channels. The first is a control word, in which the single bits contain certain commands (e.g. enable, start, etc.). The second processing channel contains the ramp reference 1 (RampRef1) in rpm.

| Position       | Description  | Format      | Unit of Measure |
|----------------|--------------|-------------|-----------------|
| Word1 M -> S   | Control word | 16 bit Word |                 |
| Word2 M -> S   | Ramp Ref 1   | Int 16 bit  | rpm             |
| Word3 M -> S   |              |             |                 |
| Word4 M -> S   |              |             |                 |
| Word5 M -> S   |              |             |                 |
| Word6 M -> S   |              |             |                 |
| Word7 M -> S   |              |             |                 |
|                |              |             |                 |
|                |              |             |                 |
| Word16 $M > S$ |              |             |                 |

#### CANopen PDO: Master -> Drive (max 16 word)

#### CONTROL WORD

| Bit | Description  | Remarks                                      |
|-----|--------------|--|
| 0   | EnableCmd    | Enable command from CANopen master           |
| 1   | StartCmd     | Start command from CANopen master            |
| 2   | Free         |  |
| 3   | Free         |  |
| 4   | Free         |  |
| 5   | Free         |  |
| 6   | Free         |  |
| 7   | Free         |  |
| 8   | Digital Out3 | Digital output 3 command from CANopen master |
| 9   | Digital Out4 | Digital output 4 command from CANopen master |
| 10  | Free         |  |
| 11  | Free         |  |
| 12  | Free         |  |
| 13  | Free         |  |
| 14  | Free         |  |
| 15  | Free         |  |

#### 1.6.1.2 Description of Slave -> Master PDO Communication

The CAN master reads three parameters from the drive. The first contains a status word in which the single bits carry information about the status of the drive (e.g. DriveOk). The second parameter is the actual speed in rpm. The third parameter contains the value of analog input 2.

| Position      | Description    | Format      | Unit of Measure |
|---------------|----------------|-------------|-----------------|
| Word1 S -> M  | Status Word    | 16 bit Word | BitWide         |
| Word2 S -> M  | Actual Speed   | Int 16 bit  | rpm             |
| Word3 S -> M  | Analog Input 2 | Int 16 bit  |                 |
| Word4 S -> M  |                |             |                 |
| Word5 S -> M  |                |             |                 |
| Word6 S -> M  |                |             |                 |
| Word7 S -> M  |                |             |                 |
|               |                |             |                 |
|               |                |             |                 |
| Word16 S -> M |                |             |                 |

| 0.444   |           |   |        | ,    |    |       |
|---------|-----------|---|--------|------|----|-------|
| CANopen | PDO Slave | > | Master | (max | 16 | Word) |

#### STATUS WORD

| Bit | Description     | Remarks                       |
|-----|-----------------|-------------------------------|
| 0   | EnableState     | Drive enabled                 |
| 1   | Drive Ok        | Drive Ok                      |
| 2   | Speed is zero   | Zero speed threshold          |
| 3   | Free            |                               |
| 4   | Free            |                               |
| 5   | Free            |                               |
| 6   | Free            |                               |
| 7   | Free            |                               |
| 8   | Digital Input 4 | ADV200 digital input 4 status |
| 9   | Digital Input 5 | ADV200 digital input 5 status |
| 10  | Free            |                               |
| 11  | Free            |                               |
| 12  | Free            |                               |
| 13  | Free            |                               |
| 14  | Free            |                               |
| 15  | Free            |                               |

#### 1.6.2 ADV200 Configuration

The example given in this section is based on the assumption that the parameters of the ADV200 drive are the factory settings (**Default parameter** command).

#### 1.6.2.1 FIELDBUS CONFIG

The example assumes that the drive is node 12 and the CANopen communication baudrate is 500k.

The drive must be reset to make all fieldbus settings and configurations effective.



Configure the fieldbus menu parameters as shown below:

| ADV200_1_X_OTestCanOpen01V28_07_08.par [FIELD8US CONFIG] - E@syDrives   |    |       |                   |                |               |      |      |             |      |                   |
|---|----|-------|-------------------|----------------|---------------|------|------|-------------|------|-------------------|
| <u>x 6 2 5</u> 7 8 W 2 4 6 11 <i>f</i> 11 <b>C</b> 10 <b>2</b> <i>f</i> 3 <b>1</b> <i>f</i> 3 11 <b>C</b> 10 <b>2</b> <i>f</i> 3 <b>1</b> <i>f</i> 3 11 <i></i> |    |       |                   |                |               |      |      |             |      |                   |
| Menu  | =1 | IPA   | Parameter Name    | Value          | Default value | Unit | Type | Description | Note | Internal Name     |
| Menu selection  |    | 4000  | Fieldbus type     | CanOpen        | Off           |      | Enum |             |      | Fieldbus type     |
|   | -  | 4004  | Fieldbus baudrate | 500k           | 500k          |      | Enum |             |      | Fieldbus baudrate |
| ANALOG INPUTS   |    | 4006  | Fieldbus address  | 12             | 3             |      | Int  |             |      | Fieldbus address  |
| ANALOG OUTPUTS  |    | 4014* | Fieldbus state    | PreOperational | Stop          |      | Enum |             |      | Fieldbus state    |
| SPEED REG GAINS<br>VF PARAMETERS<br>COMMUNICATION<br>FILIDRUS CONFIC<br>FILIDRUS SCAN<br>ALARM CONFIG<br>ALARM CONFIG<br>ALARM CONFIG<br>ALARM CONFIG<br>ALARM CONFIG<br>ALARM CONFIG<br>ALARM CONFIG<br>ALARM CONFIG<br>ALARM CONFIG<br>ALARM CONFIG   | -  |       |                   |                |               |      |      |             |      |                   |
| Votes:  |    |       |                   |                |               |      |      |             |      |                   |
| 🔿 No alarms   |    |       |                   |                |               |      |      |             |      | S CONNECTED       |

The pre-operational status of the CANopen expansion card LEDs is shown in the relative column:

| Led          | \$  | Status = Pre-operational          | Status = Operational |
|--------------|-----|-----------------------------------|----------------------|
| AL (Red)     | ON  | alarm condition                   | OFF                  |
| OP (Green)   | OFF | Pre-Operational condition         | ON                   |
| PWR (Green)  | ON  | expansion card powered and active | ON                   |
| +5VC (Green) | ON  | CAN opto-isolated power           | ON                   |

Processing channel communication is not active in these conditions. When drive configuration is complete (see following sections) use the NMT "start node" command to activate communication by the master.

# Upon receiving this command the FieldBus State parameter moves to Operational and the green OP LED is set to "ON". Only at this point are the processing channels active.

#### 1.6.2.2 MASTER -> SLAVE channel configuration

Wdecomp is used to **configure the control word**. The Wdecomp configuration on the first M -> S word ("Export" mode) is shown below:

| 8 🗃 🗐 🗇 R 🖤 🗄                 | e   4 | r an | # 🖉 🎞 🖪 🐄         | 🗖 🛦 🕫 🕨 🖪          | 8 ?            |      |       |             |      |                   |   |
|-------------------------------|-------|--|-------------------|--------------------|----------------|------|-------|-------------|------|-------------------|---|
| lenu                          | = [   | IPA                                      | Parameter Name    | Value              | Default value  | Unit | Type  | Description | Note | Internal Name     |   |
| Menu selection                |       | 4450                                     | Dig word decomp   | 0                  | 0              |      | DWord |             |      | Dig word decomp   |   |
|                               |       | 4452                                     | Word decomp src   | Fieldbus M->S1 mon | Dig word decor |      | Enum  |             |      | Word decomp src   |   |
| + C VF PARAMETERS             |       | 1454*                                    | Bit0 decomp mon   | 0                  | 0              |      | Word  |             |      | Bit0 decomp mon   |   |
|                               |       | 1456*                                    | Bit1 decomp mon   | 0                  | 0              |      | Word  |             |      | Bit1 decomp mon   |   |
| C R5485                       |       | 1458*                                    | Bit2 decomp mon   | 0                  | 0              |      | Word  |             |      | Bit2 decomp mon   |   |
| FIELDBUS CONFIG               | - E   | 1460*                                    | Bit3 decomp mon   | 0                  | 0              |      | Word  |             |      | Bit3 decomp mon   |   |
| FIELDBUS M2S                  |       | 1462*                                    | Bit4 decomp mon   | 0                  | 0              |      | Word  |             |      | Bit4 decomp mon   |   |
| WORD COMP                     |       | 1464*                                    | Bit5 decomp mon   | 0                  | 0              |      | Word  |             |      | Bit5 decomp mon   |   |
| WORD DECOMP                   |       |  |                   | 0                  | 0              |      | Word  |             |      | Bit6 decomp mon   |   |
| 2 ALARM CONFIG                |       | 468*                                     | Bit7 decomp mon   | 0                  | 0              |      | Word  |             |      | Bit7 decomp mon   |   |
| APPLICATION     InterfaceMenu |       | 1470*                                    | Bit8 decomp mon   | 0                  | 0              |      | Word  |             |      | Bit8 decomp mon   |   |
| WIZARD                        |       | 1472*                                    | Bit9 decomp mon   | 0                  | 0              |      | Word  |             |      | Bit9 decomp mon   |   |
| 🗉 🧱 DIAGRAMS                  |       | 1474*                                    | Bit10 decomp mon  | 0                  | 0              |      | Word  |             |      | Bit10 decomp mon  |   |
| DESTINATIONS     tes:         |       | 4767                                     | Bitt 1 decomp mon | n                  | n              |      | Word  |             |      | Bitt 1 decomp mon | 1 |

Now simply connect the single Wdecomp bits. For Commands the drive must be set to "**Remote**" and "**Digital**" mode, as explained in the ADV200 manual. Configure the first two bits in the commands menu as shown below:

| Š 🗃 🗟 🚉 🖪 🖤 🕄                                     |    | ц 🗳   | <u> </u>            | a 🛃 🕾 🖻 🗗           |                   |      |      |             |      |                     |
|---|----|-------|---------------------|---------------------|-------------------|------|------|-------------|------|---------------------|
| lenu :  | 3  | IPA   | Parameter Name      | Value               | Default value     | Unit | Type | Description | Note | Internal Name       |
| Menu selection                                    |    | 1000  | Commands remote sel | Digital             | Terminal          |      | Enum |             |      | Commands remote s   |
|   | 1  | 1002  | Commands local sel  | Keypad              | Keypad            |      | Enum |             |      | Commands local sel  |
| - CI SPEED MONITOR FUNC                           |    | 1004  | Enable/disable mode | Stop/FS&Spd=0       | Stop/FS&Spd=0     |      | Enum |             |      | Enable/disable mode |
| DIGITAL INPUTS                                    |    | 1006  | Speed 0 disable dly | 1000                | 1000              | ms   | Word |             |      | Speed 0 disable dly |
| DIGITAL OUTPUTS                                   |    | 1008  | Stop key mode       | Inactive            | Inactive          |      | Enum |             |      | Stop key mode       |
| - 0 ANALOG INPUTS                                 |    | 1010  | Commands safe start | Off                 | Off               |      | Bool |             |      | Commands safe star  |
|   | нÍ | 1012  | Dig local/remote    | Remote              | Remote            |      | Enum |             |      | Dig local/remote    |
| - I ENCODER CONFIG                                |    | 1014  | Local/remote src    | Dig local/remote    | Dig local/remot   |      | Enum |             |      | Local/remote src    |
| SPEED REG GAINS                                   |    | 1016  | Terminal Start src  | FR start mon        | FR start mon      |      | Enum |             |      | Terminal Start src  |
| - 👸 REGULATOR PARAM                               |    | 1018  | Digital Enable src  | Bit0 decomp mon     | Null              |      | Enum |             |      | Digital Enable src  |
| - C TORQUE CONFIG                                 |    | 1020  | Digital Start src   | Bit1 decomp mon     | Null              |      | Enum |             |      | Digital Start src   |
| H- 1 FUNCTIONS                                    |    | 1022  | FastStop src        | Null                | Null              |      | Enum |             |      | FastStop src        |
| COMMUNICATION                                     |    | 1024* | Enable cmd mon      | 0                   | 0                 |      | Word |             |      | Enable cmd mon      |
| 👩 R5485   |    | 1026* | Start cmd mon       | 0                   | 0                 |      | Word |             |      | Start cmd mon       |
| FIELDBUS CONFIG                                   |    | 1028* | FastStop cmd mon    | 0                   | 0                 |      | Word |             |      | FastStop cmd mon    |
| FIELDBUS S2M                                      | 11 | 1040  | FR mode             | Normal              | Normal            |      | Enum |             |      | FR mode             |
| WORD COMP   |    | 1042  | FR forward src      | Digital input 1 mon | Digital input 1 n |      | Enum |             |      | FR forward src      |
| WORD DECOMP                                       |    | 1044  | FR reverse src      | Digital input 2 mon | Digital input 2 n |      | Enum |             |      | FR reverse src      |
| ALARM CONFIG                                      |    | 1046  | FR *stop src        | Null                | Null              |      | Enum |             |      | FR *stop src        |
| InterfaceMenu                                     | 11 | 1048* | FR start mon        | 0                   | 0                 |      | Word |             |      | FR start mon        |
| WIZARD  | 1  | 1050* | FR reverse mon      | 0                   | 0                 |      | Word |             |      | FR reverse mon      |
| DIAGRAMS     DESTINATIONS     FIELDBLIS WORDS MAP | -  | 1052* | FR cmd mon          | 0                   | 0                 |      | Word |             |      | FR cmd mon          |

Configure bits 8 and 9 of the "Command word" as shown below (Digital Outputs menu):

| 🕯 📾 🛃 💭 R W 😇                 | 1   | - 67 | 601 🖋 👬 🖪 🖼          | 🗖 📐 🕫 🛸 🗖       | ) <b>3 8</b> ?   |      |      |             |      |                      |   |
|-------------------------------|-----|------|----------------------|-----------------|------------------|------|------|-------------|------|----------------------|---|
| enu                           | E F | IPA  | Parameter Name       | Value           | Default value    | Unit | Type | Description | Note | Internal Name        | _ |
| Menu selection                | ][  | 1310 | Digital output 1 src | Drive OK        | Drive OK         |      | Enum |             |      | Digital output 1 src |   |
| 🕂 🖞 DRIVE CONFIG              | ][  | 1312 | Digital output 2 src | Drive ready     | Drive ready      |      | Enum |             |      | Digital output 2 src |   |
| REFERENCES                    | ſ   | 1314 | Digital output 3 src | Bit8 decomp mon | Speed is 0 delay |      | Enum |             |      | Digital output 3 src |   |
| MULTI REFERENCE               | .Г  | 1316 | Digital output 4 src | Bit9 decomp mon | Refis 0 delay    |      | Enum |             |      | Digital output 4 src |   |
| MOTORPOTENTIOMETER            | 11  | 1330 | Dig out 1 inversion  | off             | off              |      | Bool |             |      | Dig out 1 inversion  |   |
| - 🖞 JOG FUNCTION              | Ir  | 1332 | Dig out 2 inversion  | Off             | Off              |      | Bool |             |      | Dig out 2 inversion  |   |
| COMMANDS                      | lŀ  | 1334 | Dig out 3 inversion  | Off             | Off              |      | Bool |             |      | Dig out 3 inversion  |   |
| DIGITAL INPUTS                | IF  | 1336 | Dig out 4 inversion  | off             | off              |      | Bool |             |      | Dig out 4 inversion  |   |
| DIGITAL OUTPUTS               | ١٢  | 1410 | Dig output 1X src    | Null            | Null             |      | Enum |             |      | Dig output 1X src    |   |
| ANALOG INPUTS                 | ١٢  | 1412 | Dig output 2X src    | Null            | Null             |      | Enum |             |      | Dig output 2X src    |   |
| MOTOR DATA                    | IF  | 1414 | Dig output 3X src    | Null            | Null             |      | Enum |             |      | Dig output 3X src    |   |
| ENCODER CONFIG                | 11  | 1416 | Dig output 4X src    | Null            | Null             |      | Enum |             |      | Dig output 4X src    |   |
| SPEED REG GAINS               | ۱ŀ  | 1418 | Dig output 5X src    | Null            | Null             |      | Enum |             |      | Dig output 5X src    |   |
| - 👸 REGULATOR PARAM           | lŀ  | 1420 | Dig output 6X src    | Null            | Null             |      | Enum |             |      | Dig output 6X src    |   |
| TORQUE CONFIG                 | 11  | 1422 | Dig output 7X src    | Null            | Null             |      | Enum |             |      | Dig output 7X src    |   |
| FILL FUNCTIONS                | ١ŀ  | 1424 | Dig output 8X src    | Null            | Null             |      | Enum |             |      | Dig output 8X src    |   |
| COMMUNICATION                 | ۱ŀ  | 1430 | Dig out 1X inversion | Off             | Off              |      | Bool |             |      | Dig out 1X inversion |   |
| - 👸 ALARM CONFIG              | Ir  | 1432 | Dig out 2X inversion | off             | off              |      | Bool |             |      | Dig out 2X inversion |   |
| APPLICATION     InterfaceMenu | IF  | 1434 | Dig out 3X inversion | off             | Off              |      | Bool |             |      | Dig out 3X inversion |   |
| WIZARD                        | ۱Ľ  | 1436 | Dig out 4X inversion | Off             | Off              |      | Bool |             |      | Dig out 4X inversion |   |
| E DIAGRAMS                    | lŀ  | 1438 | Dig out 5X inversion | Off             | Off              |      | Bool |             |      | Dig out 5X inversion |   |
| DESTINATIONS                  | IF  | 1440 | Dig out 6X inversion | orr             | off              |      | Bool |             |      | Dig out 6X inversion |   |
| 1 FIELDBUS WORDS MAP          | łĽ  | 1442 | Dig out 7X inversion | Off             | 0ff              |      | Bool |             |      | Dig out 7X inversion |   |

The second word is configured in the "References" menu:

|      |  | 🖻 🔥 🖉 🖻 🖻   |   |   |   |   |   |   |
|------|--|---|---|---|---|---|---|---|
| IPA  | Parameter Name   | Value   | Default value   | Unit  | Type D  | Description   | Note  | Internal Name   |
| 600  | Dig ramp ref 1   | 0   | 0   | rpm   | Int   |   |   | Dig ramp ref 1  |
| 602  | Dig ramp ref 2   | 0   | 0   | rpm   | Int   |   |   | Dig ramp ref 2  |
| 604  | Dig ramp ref 3   | 0   | 0   | rpm   | Int   |   |   | Dig ramp ref 3  |
| 610  | Ramp ref 1 src   | Fieldbus M->S2 mon  | Multi ref out mon   |   | Enum  |   |   | Ramp ref 1 src  |
| 612  | Ramp ref 2 src   | Dig ramp ref 2  | Dig ramp ref 2  |   | Enum  |   |   | Ramp ref 2 src  |
| 614  | Ramp ref 3 src   | Mpot output mon   | Mpot output mon   |   | Enum  |   |   | Ramp ref 3 src  |
| 616  | Ramp ref invert src  | FR reverse mon  | FR reverse mon  |   | Enum  |   |   | Ramp ref invert src   |
| 620* | Ramp ref 1 mon   | 0   | 0   | rpm   | Int   |   |   | Ramp ref 1 mon  |
| 622* | Ramp ref 2 mon   | 0   | 0   | rpm   | Int   |   |   | Ramp ref 2 mon  |
| 624* | Ramp ref 3 mon   | 0   | 0   | rpm   | Int   |   |   | Ramp ref 3 mon  |
| 630  | Reference skip set   | 0   | 0   | rpm   | Int   |   |   | Reference skip set  |
| 632  | Reference skip band  | 0   | 0   | rpm   | Int   |   |   | Reference skip band   |
| 640  | Dig speed ref 1  | 0   | 0   | rpm   | Int   |   |   | Dig speed ref 1   |
| 642  | Dig speed ref 2  | 0   | 0   | rpm   | Int   |   |   | Dig speed ref 2   |
| 650  | Speed ref 1 src  | Dig speed ref 1   | Dig speed ref 1   |   | Enum  |   |   | Speed ref 1 src   |
| 652  | Speed ref 2 src  | Dig speed ref 2   | Dig speed ref 2   |   | Enum  |   |   | Speed ref 2 src   |
| 654  | Speed ref invert src   | Null  | Null  |   | Enum  |   |   | Speed ref invert src  |
| 660* | Speed ref 1 mon  | 0   | 0   | rpm   | Int   |   |   | Speed ref 1 mon   |
| 662* | Speed ref 2 mon  | 0   | 0   | rpm   | Int   |   |   | Speed ref 2 mon   |
| 670  | Speed reftop lim   | 1500  | 0   | rpm   | Int   |   |   | Speed ref top lim   |
| 672  | Speed ref bottom lim   | -1500   | 0   | rpm   | Int   |   |   | Speed ref bottom lim  |
| 680  | Full scale speed   | 1500  | 0   | rpm   | Int   |   |   | Full scale speed  |
|      | 600<br>602<br>604<br>610<br>612<br>614<br>616<br>620*<br>622*<br>624*<br>630<br>632<br>640<br>642<br>650<br>652<br>654<br>652<br>654<br>660*<br>670<br>672 | 800         Digramp ref 1           602         Digramp ref 2           604         Digramp ref 3           610         Ramp ref 1 arc           611         Ramp ref 2 arc           612         Ramp ref 2 arc           613         Ramp ref 2 arc           614         Ramp ref 3 arc           615         Ramp ref 7 arc           616         Ramp ref 7 arc           6227         Ramp ref 2 mon           6247         Ramp ref 2 mon           6247         Ramp ref 3 mon           630         Reference skip ard           631         Reference skip ard           632         Reference skip ard           640         Dig speed ref 1 arc           652         Speed ref 1 arc           653         Speed ref 1 arc           654         Speed ref 1 arc           655         Speed ref 1 arc           656         Speed ref 2 arc           657         Speed ref 2 arc           658         Speed ref 2 arc           654         Speed ref 2 arc           657         Speed ref 2 arc           658         Speed ref 2 arc           657         Speed ref 1 arc | 500         Digrampitef1         0           602         Digrampitef1         0           603         Digrampitf3         0           610         Rampitef1 size         Fieldbus Mi+52 mon           611         Rampitf3 size         Might object with mon           616         Rampitf1 mon         0           6207         Rampitf1 mon         0           6227         Rampitf2 mon         0           6227         Rampitf2 mon         0           6327         Rampitf2 mon         0           6328         Reference skip bad         0           640         Dig speed ref1         0           650         Speed ref1 arc         Oig speed ref1           652         Speed ref2 arc         0 speed ref1           652         Speed ref2 arc         0 speed ref1           652         Speed ref2 arc         0 speed ref1           6547         Speed ref2 arc         0           6547         Speed ref2 | 800         Dig ramp ref 1         0         0           802         Dig ramp ref 2         0         0           801         Dig ramp ref 3         0         0           802         Dig ramp ref 3         0         0           803         Dig ramp ref 3         0         0           804         Dig ramp ref 2         Dig ramp ref 2         Dig ramp ref 2           805         Ramp ref 3 arc         Mpot output mon         Mpot output mon           807         Ramp ref 3 arc         Mpot output mon         0           8227         Ramp ref 2         0         0         0           8227         Ramp ref 2         0         0         0           8227         Ramp ref 2         0         0         0           8228         Ramp ref 2         0         0         0           829         Ramp ref 3         0         0         0           820         Dig specif 2         0         0         0           833         Reference skip bard         0         0         0           840         Dig specif 1         0         0         0           8530         Specei ref 1 xrc         Di | B00         Digramp ref1         0         0         rpm           600         Digramp ref2         0         0         rpm           601         Digramp ref2         0         0         rpm           610         Barap ref3         0         rpm         rpm           611         Ramp ref3 arc         Fieldbus M=>S2 mon         Mult ref out mon         rpm           616         Ramp ref2 arc         Digramp ref2         Digramp ref2         Digramp ref2           616         Ramp ref3 arc         Mod output mon         Mod output mon         Mod output mon           6207         Ramp ref3 mon         0         rpm         rpm           6227         Ramp ref3 mon         0         rpm         rpm           633         Reference skip bad         0         rpm         rpm           634         Dig speed ref1         Dig speed ref1         Dig speed ref1         Dig speed ref1           640         Dig speed ref1         Dig speed ref2         Dig speed ref2         E         E           653         Speed ref1 rem to         Dig speed ref2         Dig speed ref2         E         E           654         Speed ref1 rem to         0         rpm <td< td=""><td>B00         Dig ramp ref1         0         0         rpm         Int           602         Dig ramp ref2         0         0         rpm         Int           602         Dig ramp ref3         0         0         rpm         Int           610         Dig ramp ref3         0         0         rpm         Int           610         Dig ramp ref3         0         0         rpm         Int           610         Dig ramp ref3         0         0         rpm         Int           611         Ramp ref3 rec         Dig ramp ref2         Dig ramp ref4         Dig ramp re</td><td>B00         Dig ramp ref1         0         0         rpm         Int           602         Dig ramp ref2         0         0         rpm         Int           602         Dig ramp ref3         0         0         rpm         Int           610         Dig ramp ref2         Dig ramp ref2         Dig ramp ref4         Dig ramp ref4         Dig ramp ref4           616         Ramp ref7 arcs         Dig ramp ref7         Dig ramp ref7         Dig ramp ref7         Dig ramp ref7           617         Ramp ref7 arcs         Dig ramp ref7         Dig ramp ref7</td><td>B000         Digramp ref1         0         0         rpm         Int           6000         Digramp ref2         0         0         rpm         Int        </td></td<> | B00         Dig ramp ref1         0         0         rpm         Int           602         Dig ramp ref2         0         0         rpm         Int           602         Dig ramp ref3         0         0         rpm         Int           610         Dig ramp ref3         0         0         rpm         Int           610         Dig ramp ref3         0         0         rpm         Int           610         Dig ramp ref3         0         0         rpm         Int           611         Ramp ref3 rec         Dig ramp ref2         Dig ramp ref4         Dig ramp re | B00         Dig ramp ref1         0         0         rpm         Int           602         Dig ramp ref2         0         0         rpm         Int           602         Dig ramp ref3         0         0         rpm         Int           610         Dig ramp ref2         Dig ramp ref2         Dig ramp ref4         Dig ramp ref4         Dig ramp ref4           616         Ramp ref7 arcs         Dig ramp ref7         Dig ramp ref7         Dig ramp ref7         Dig ramp ref7           617         Ramp ref7 arcs         Dig ramp ref7         Dig ramp ref7 | B000         Digramp ref1         0         0         rpm         Int           6000         Digramp ref2         0         0         rpm         Int |

After sending a save command and re-starting the drive, check that the M -> S channels have been configured correctly as shown (Html page):

| ADV200_1_X_0TestCanOpen01V<br>File View Parameters Target Serv      |    |          | ar [ M25 ] | ] - E@syDrives                         |                  | _ 🗆 3 |
|---|----|----------|------------|--|------------------|-------|
| * 📽 🖬 🛐 🗊 R 🖤 🕮   | 1  |          | # 9        | •••••••••••••••••••••••••••••••••••••• | 2                |       |
| Menu  | -  |          |            |  |                  |       |
| Menu selection  | 11 | Fiel     | dbus       | s M->S Words Mapp                      | ing              |       |
| REFERENCES  | ]  | Number   | ofpara     | meters exchanged: 2                    | Esc Key          |       |
| MOTORPOTENTIOMETER  |    |          | -          |  |                  |       |
| JOG FUNCTION     SPEED MONITOR FUNC     COMMANDS     DIGITAL INPUTS |    | Number   | of word    | ls mapped: 2                           |                  |       |
| - 0 DIGITAL OUTPUTS   |    | Word     | Ipa        | Parameter name                         | Format exchange  |       |
| - 💭 ANALOG INPUTS   |    | 1        | 4452       | Word decomp src                        | Count 16 - 16bit |       |
| MOTOR DATA  |    | 2        | 610        | Ramp ref 1 src                         | Eu - 16bit       |       |
| - MOTOR DATA  |    |          | -          | -                                      | -                |       |
| SPEED REG GAINS   |    | <u> </u> | -          | -                                      | -                |       |
| REGULATOR PARAM   |    | L ·      | <u> </u>   | -                                      |                  |       |
| TORQUE CONFIG   |    | <u> </u> | <u> </u>   | -                                      | -                |       |
| VF PARAMETERS   |    | <u> </u> | <u> </u>   | -                                      | -                |       |
| FUNCTIONS   |    | <u> </u> | <u> </u>   |  | -                |       |
| COMMUNICATION   |    | <u> </u> | <u> </u>   | -                                      | · ·              |       |
| - 👸 ALARM CONFIG  |    | <u> </u> | <u> </u>   | -                                      | · ·              |       |
| - (7) APPLICATION   |    | <u> </u> | <u> </u>   | -                                      | -                |       |
| InterfaceMenu   |    | · .      | -          | -                                      |                  |       |
| WIZARD  |    | <u> </u> | -          | -                                      | -                |       |
| 🕀 🔚 DIAGRAMS  |    | <u> </u> | •          | -                                      | -                |       |
| DESTINATIONS     FIELDBUS WORDS MAP                                 |    | <u> </u> | -          | -                                      | -                |       |
| E E FIELDOUS WORDS MAP  |    | <u> </u> | •          | -                                      | -                |       |
| S2M   |    | d        |            |  |                  | Þ     |
| lotes:  |    |          |            |  |                  |       |
| No alarms   |    |          |            |  |                  |       |

The same information is also available in the Fieldbus M2S menu:

| * <b>☞∎</b> ∰ <u>♂</u> ∎₩ฃ                       | ₩ 🗗   | 🗰 🖋 👬 🖪 🐄           | 🛛 🔥 🖉 🛸 🛛       | 3 <i>5</i> 27 |      |       |             |      |                     |   |
|--|-------|---------------------|-----------------|---------------|------|-------|-------------|------|---------------------|---|
| mu :   | IPA   | Parameter Name      | Value           | Default value | Unit | Type  | Description | Note | Internal Name       | Г |
| Menu selection                                   | 4020* | Fieldbus M->S1 dest | Word decomp src | Notused       |      | Enum  |             |      | Fieldbus M->S1 dest |   |
| REFERENCES                                       | 4022  | Fieldbus M->S1 sys  | Count 16        | Not assigned  |      | Enum  |             |      | Fieldbus M->S1 sys  |   |
| MULTI REFERENCE                                  | 4024* | Fieldbus M->S1 mon  | 0               | 0             |      | Long  |             |      | Fieldbus M->S1 mon  |   |
| MOTORPOTENTIOMETER                               | 4026  | Fieldbus M->S1 div  | 1               | 1             |      | Float |             |      | Fieldbus M->S1 div  |   |
| JOG FUNCTION                                     | 4030* | Fieldbus M->S2 dest | Ramp ref 1 src  | Notused       |      | Enum  |             |      | Fieldbus M->S2 dest |   |
| - 0 SPEED MONITOR FUNC                           | 4032  | Fieldbus M->S2 sys  | Eu              | Not assigned  |      | Enum  |             |      | Fieldbus M->S2 sys  |   |
| COMMANDS   | 4034* | Fieldbus M->S2 mon  | 0               | 0             |      | Long  |             |      | Fieldbus M->S2 mon  |   |
| DIGITAL OUTPUTS                                  | 4036  | Fieldbus M->S2 div  | 1               | 1             |      | Float |             |      | Fieldbus M->S2 div  |   |
| ANALOG INPUTS                                    | 4040* | Fieldbus M->S3 dest | Notused         | Notused       |      | Enum  |             |      | Fieldbus M->S3 dest |   |
| ANALOG OUTPUTS     MOTOR DATA     ENCODER CONEIG | 4042  | Fieldbus M->S3 sys  | Not assigned    | Not assigned  |      | Enum  |             |      | Fieldbus M->S3 sys  |   |
|  | 4044* | Fieldbus M->S3 mon  | 0               | 0             |      | Long  |             |      | Fieldbus M->S3 mon  |   |
| SPEED REG GAINS                                  | 4046  | Fieldbus M->S3 div  | 1               | 1             |      | Float |             |      | Fieldbus M->S3 div  |   |
| - REGULATOR PARAM                                | 4050* | Fieldbus M->S4 dest | Not used        | Notused       |      | Enum  |             |      | Fieldbus M->S4 dest |   |
| - 🖞 TORQUE CONFIG                                | 4052  | Fieldbus M->S4 sys  | Not assigned    | Not assigned  |      | Enum  |             |      | Fieldbus M->S4 sys  |   |
| + C VF PARAMETERS                                | 4054* | Fieldbus M->S4 mon  | 0               | 0             |      | Long  |             |      | Fieldbus M->S4 mon  |   |
|  | 4056  | Fieldbus M->S4 div  | 1               | 1             |      | Float |             |      | Fieldbus M->S4 div  |   |
| 0 R5485  | 4060* | Fieldbus M->S5 dest | Not used        | Notused       |      | Enum  |             |      | Fieldbus M->S5 dest |   |
| - 👸 FIELDBUS CONFIG                              | 4062  | Fieldbus M->S5 sys  | Not assigned    | Not assigned  |      | Enum  |             |      | Fieldbus M->S5 sys  |   |
| FIELDBUS M2S                                     | 4064* | Fieldbus M->S5 mon  | 0               | 0             |      | Long  |             |      | Fieldbus M->S5 mon  |   |
| WORD COMP  | 4066  | Fieldbus M->S5 div  | 1               | 1             |      | Float |             |      | Fieldbus M->S5 div  |   |
| WORD DECOMP                                      | 4070* | Fieldbus M->S6 dest | Notused         | Notused       |      | Enum  |             |      | Fieldbus M->S6 dest |   |
| - 🖞 ALARM CONFIG                                 | 4072  | Fieldbus M->S6 sys  | Not assigned    | Not assigned  |      | Enum  |             |      | Fieldbus M->S6 sys  |   |
| APPLICATION     InterfaceMenu                    | 4074* | Fieldbus M->S6 mon  | 0               | 0             |      | Long  |             |      | Fieldbus M->S6 mon  |   |
| Interfacemenu -                                  | 4076  | Fieldhus M->S6 div  | 1               | 1             |      | Float |             |      | Fieldhus M->S6 div  |   |

#### 1.6.2.3 SLAVE -> MASTER channel configuration

These channels are configured in the Fieldbus S2M menu. Use Wcomp to configure the first channel.

S2M configuration is shown below:

| 📽 🖬 🛃 🖉 R W   | 2 | u 🗗  | 🗰 🖋 👬 🖾 🐄          | 🛛 📥 🖉 🛸 🖸          | a e ?         |      |       |             |      |                    |   |
|---|---|------|--------------------|--------------------|---------------|------|-------|-------------|------|--------------------|---|
| enu :   | = | IPA  | Parameter Name     | Value              | Default value | Unit | Type  | Description | Note | Internal Name      | Г |
| Menu selection  |   | 4180 | Fieldbus S->M1 src | Word comp mon      | Null          |      | Enum  |             |      | Fieldbus S->M1 src |   |
| MainMenu  | - | 4182 | Fieldbus S->M1 sys | Count 16           | Not assigned  |      | Enum  |             |      | Fieldbus S->M1 sys |   |
| MONITOR   |   | 4184 | Dig Fieldbus S->M1 | 0                  | 0             |      | Long  |             |      | Dig Fieldbus S->M1 |   |
| DRIVE CONFIG  |   | 4186 | Fieldbus S->M1 mul | 1                  | 1             |      | Float |             |      | Fieldbus S->M1 mul |   |
| - P REFERENCES  |   | 4190 | Fieldbus S->M2 src | Motor speed        | Null          |      | Enum  |             |      | Fieldbus S->M2 src |   |
| - 🖞 RAMPS   |   | 4192 | Fieldbus S->M2 sys | Eu                 | Not assigned  |      | Enum  |             |      | Fieldbus S->M2 sys |   |
| MULTI REFERENCE                                       |   | 4194 | Dig Fieldbus S->M2 | 0                  | 0             |      | Long  |             |      | Dig Fieldbus S->M2 |   |
| 10G FUNCTION  |   | 4196 | Fieldbus S->M2 mul | 1                  | 1             |      | Float |             |      | Fieldbus S->M2 mul |   |
|   |   | 4200 | Fieldbus S->M3 src | Analog input 2 mon | Null          |      | Enum  |             |      | Fieldbus S->M3 src |   |
| 0 COMMANDS<br>0 DIGITAL INPUTS<br>0 DIGITAL OLITPLITS |   | 4202 | Fieldbus S->M3 sys | Count 16           | Not assigned  |      | Enum  |             |      | Fieldbus S->M3 sys |   |
|   |   | 4204 | Dig Fieldbus S->M3 | 0                  | 0             |      | Long  |             |      | Dig Fieldbus S->M3 |   |
| ANALOG INPLITS  |   | 4206 | Fieldbus S->M3 mul | 1                  | 1             |      | Float |             |      | Fieldbus S->M3 mul |   |
| ANALOG OUTPUTS  |   | 4210 | Fieldbus S->M4 src | Null               | Null          |      | Enum  |             |      | Fieldbus S->M4 src |   |
| - 🖞 MOTOR DATA  |   | 4212 | Fieldbus S->M4 sys | Not assigned       | Not assigned  |      | Enum  |             |      | Fieldbus S->M4 sys |   |
|   |   | 4214 | Dig Fieldbus S->M4 | 0                  | 0             |      | Long  |             |      | Dig Fieldbus S->M4 |   |
| REGULATOR PARAM                                       |   | 4216 | Fieldbus S->M4 mul | 1                  | 1             |      | Float |             |      | Fieldbus S->M4 mul |   |
| TORQUE CONFIG   |   | 4220 | Fieldbus S->M5 src | Null               | Null          |      | Enum  |             |      | Fieldbus S->M5 src |   |
| VF PARAMETERS   |   | 4222 | Fieldbus S->M5 sys | Not assigned       | Not assigned  |      | Enum  |             |      | Fieldbus S->M5 sys |   |
| FUNCTIONS     COMMUNICATION                           |   | 4224 | Dig Fieldbus S->M5 | 0                  | 0             |      | Long  |             |      | Dig Fieldbus S->M5 |   |
| RS485   |   | 4226 | Fieldbus S->M5 mul | 1                  | 1             |      | Float |             |      | Fieldbus S->M5 mul |   |
| FIELDBUS CONFIG                                       |   | 4230 | Fieldbus S->M6 src | Null               | Null          |      | Enum  |             |      | Fieldbus S->M6 src |   |
|   |   | 4232 | Fieldbus S->M6 sys | Not assigned       | Not assigned  |      | Enum  |             |      | Fieldbus S->M6 sys |   |
| - C FIELDBUS S2M                                      |   | 4234 | Dig Fieldbus S->M6 | 0                  | 0             |      | Long  |             |      | Dig Fieldbus S->M6 |   |
| WORD COMP   | • | 4236 | Fieldhus S->M6 mul | 1                  | 1             |      | Float |             |      | Fieldhus S->M6 mul |   |

Wcomp configuration is shown below:

| <b>ジョ</b> 割 ② R W   | 12 |       | 🗰 🖋 👬 🖾 🖄       |                              |               | 11-24 |       | L Barristern |      | Later at Maria                 |
|---|----|-------|-----------------|------------------------------|---------------|-------|-------|--------------|------|--------------------------------|
| Menu selection  | _  | IPA   | Parameter Name  | Value                        | Default value | Unit  | Туре  | Description  | Note | Internal Name                  |
| MULTI REFERENCE   | -  | 4400  | Word bit0 src   | Enable state mon<br>Drive OK | Null          |       | Enum  |              |      | Word bit0 src<br>Word bit1 src |
| MOTORPOTENTIOMETER  |    | 4402  | Word bit1 src   |                              | Null          |       | Enum  |              |      |                                |
| O JOG FUNCTION  |    |       | Word bit2 src   | Speed is 0                   |               |       | Enum  |              |      | Word bit2 src                  |
| C SPEED MONITOR FUNC  |    | 4406  | Word bit3 src   | Null                         | Null          |       | Enum  |              |      | Word bit3 src                  |
| COMMANDS  |    | 4408  | Word bit4 src   | Null                         | Null          |       | Enum  |              |      | Word bit4 src                  |
| DIGITAL OUTPUTS   |    |       | Word bit5 src   | Null                         | Null          |       | Enum  |              |      | Word bit5 src                  |
| ANALOG INPUTS   |    | 4412  | Word bit6 src   | Null                         | Null          |       | Enum  |              |      | Word bit6 src                  |
| MALOG OUTPUTS<br>MOTOR DATA<br>ENCODER CONFIG<br>SPEED REG GAINS  |    | 4414  | Word bit7 src   | Null                         | Null          |       | Enum  |              |      | Word bit7 src                  |
|   |    |       | Word bit8 src   | Digital input 4 mon          | Null          |       | Enum  |              |      | Word bit8 src                  |
|   |    | 4418  | Word bit9 src   | Digital input 5 mon          | Null          |       | Enum  |              |      | Word bit9 src                  |
| REGULATOR PARAM   |    | 4420  | Word bit10 src  | Null                         | Null          |       | Enum  |              |      | Word bit10 src                 |
| TORQUE CONFIG   |    | 4422  | Word bit11 src  | Null                         | Null          |       | Enum  |              |      | Word bit11 src                 |
| C VF PARAMETERS   |    | 4424  | Word bit12 src  | Null                         | Null          |       | Enum  |              |      | Word bit12 src                 |
| C FUNCTIONS   |    | 4426  | Word bit1 3 src | Null                         | Null          |       | Enum  |              |      | Word bit13 src                 |
| COMMUNICATION   |    | 4428  | Word bit1 4 src | Null                         | Null          |       | Enum  |              |      | Word bit14 src                 |
| FIELDBUS CONFIG   |    | 4430  | Word bit15 src  | Null                         | Null          |       | Enum  |              |      | Word bit15 src                 |
| - 🎢 FIELDBUS M2S  |    | 4432* | Word comp mon   | 0006h                        | 0000h         |       | DWord |              |      | Word comp mon                  |
| PIELDBUS S2M     WORD COMP     WORD DECOMP     WORD DECOMP     ALARM CONFIG     InterfaceMenu     WIZARD     DIAGRAMS | _  |       |                 |                              |               |       |       |              |      |                                |

Save and then re-start the drive to check the correct configuration of the Slave -> Master channels in the same way:

|  | 삔뽀 ☞ !      | <b>11</b> 🖉   | ii 🖪 🖻 🔬 🗢 🖻 🖻  | 3 E ?  |          |
|--|-------------|---|---|--|----------|
| Menu<br>Menu selection<br>CommanDs<br>Digital UNPUTS<br>Digital OUPUTS<br>ANALOG OUPUTS<br>MATOR DATA<br>ENCODER CONFIG<br>ENCODER CONFIG  | •<br>Number | of paran  | S->M Words Mapp<br>neters exchanged: 3<br>s mapped: 3   | Esc Key  | <u>^</u> |
| REGILATOR PARAM     REGILATOR PARAM     TORQUE CONFIG     VE PARAMETERS     VE | Word 1 2 3  | Ipa           4432           260           1550           -   - | Parameter name<br>Word comp mon<br>Motor speed<br>Analog input 2 mon<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>-<br>- | Format exchange           Count 10 - 160it           Eu - 160it           Count 16 - 160it           - |          |

#### 1.6.2.4 Communication check

Some notes/suggestions for checking communication.

- PDO communication is only active in "Operational". Check the status using Easydrive or the expansion card LED.
- For Master -> Slave communication in the FIELDBUS M2S menu you can check the value received by the communication channel (e.g. for the first channel it is the Fieldbus M->S1 Mon parameter).
- For EU (engineering unit) communication, remember that the value read on the Mon parameters of FIELDBUS MS2 is in internal units (see conversion tables on chapter 5.0 SYSTEM INTERNAL VARIABLES, "ADV200, Write the applications with the MDPIc" manual available on www.weg.net).

#### 1.6.2.5 Configuration Errors

In the event of a channel configuration error an "Option bus fault" alarm condition occurs when the drive is switched on

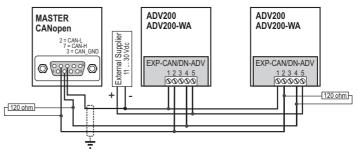
with an error code indicating the channel that generated the alarm. The expansion card manual contains a table listing all the error codes.

With WEG\_eXpress software configuration simply log-on to the relative HTML page as shown below:

| 🕯 🖬 🛃 😂 🖪 🖤 🖻   | te Help<br>맨데 | 60 🖉 ii                    |   | E ?              |   |
|---|---------------|----------------------------|---|------------------|---|
| Mu selection<br>C comMutos<br>Diatra, INPUTS<br>O Diatra, INPUTS<br>O Diatra, ComPUTS<br>O WANA, OG CUTPUTS<br>O WANA, OG CUTPUTS<br>O WOTO COATA | Numbe         | er of param<br>er of words | S->M Words Mappi<br>heters exchanged: 4<br>smapped: 4<br>Error - SubCode: FF57h - S2M | Esc Key          | m: Opt Bus fault<br>e is wrong (18 bits on 32 bits or 32 bils on 16 bits parameter) |
| - C SPEED REG GAINS   | Word          | Ine                        | Parameter name  | Format exchange  |   |
| - TORQUE CONFIG   | 1             | 4432                       | Word comp mon   | Count 16 - 16hit |   |
| - 👘 VF PARAMETERS   | 2             | 260                        | Motor speed   | Eu - 16bit       |   |
| C FUNCTIONS   | 1             | 1550                       | Analog input 2 mon  | Count 16 - 16bit |   |
| E COMMUNICATION   | 4             | 4214                       | Dig Fieldbus S->M4  | Count 16 - 16bit |   |
| -C R5465  |               | 1 . 1                      |   |                  |   |
| FIELDBUS CONFIG   | 1 ·           | i . i                      |   | i                |   |
| FIELDBUS S2M  | · ·           | î · î                      |   | i . (            |   |
|   | · ·           | î · î                      |   | î · í            |   |
|   | · ·           | 1 • 1                      |   | i                |   |
| WORD COMP   |               | 1 - 1                      |   |                  |   |
| WORD COMP   | 1 ·           |                            |   |                  |   |
| WORD DECOMP   | i F÷          | † · †                      | -   | · ·              |   |
| WORD DECOMP   |               | ÷—                         | -   |                  |   |
| ALARM CONFIG<br>ALARM CONFIG<br>APPLICATION<br>InterfaceMenu  | · ·           | <u>i · i</u>               |   |                  |   |
| WORD DECOMP     ALARM CONFIG     APPLICATION     InterfaceMenu     WIZARD     DIAGRAMS  |               |                            |   | · · ·            |   |
| ALARM CONFIG     ALARM CONFIG     ALARM CONFIG     ALARM CONFIG     ThrefaceMenu     ThrefaceMenu     JIAGRAMS     DESTINATIONS                   |               |                            | •   |                  |   |
| WORD DECOMP     ALARM CONFIG     APPLICATION     InterfaceMenu     WIZARD     DIAGRAMS  |               |                            |   |                  |   |

#### 1.7 Connecting the master control panel to ADV200 nodes

The following example shows a connection between the Control panel CANopen Master and ADV200 slave nodes.



This is a 3-wire connection: CAN line and unipotential cable on terminals 1 (V- / CAN\_GND) of the EXP-CAN/DN-ADV cards, 0 V power supply and pin 3 (CAN\_GND) CAN connector on Vedo terminal.

The shielding must be continuous along the entire CAN line.

The shielding is grounded at a single point, in proximity of the CAN line power supply. This supply can also be used to power the Vedo terminal. It is NOT advisable to use this power supply for other purposes, especially auxiliary circuits with relays.

The CAN line shielding may also be grounded at two or more points, for example if the CAN line nodes are distributed in separate electrical panels, provided the ground connections are made correctly.

#### 2.0 Operation according to the DS402 profile

If the **Fieldbus type** parameter is set to DS402 the drive works with the standard profile for Drives & Motion Control Ver 2.0 and contains Device Identity 192H (402) in object 1000h.

The ADV200 drive supports Velocity Mode.

In the default configuration the drive is automatically set to use PDOs No. 6 (DS402 section 7.2.1 & 7.2.2), mapped onto RPDO1 and TPDO1 with COB-ID 200h & 180h +Nodeld

| PD0 | Object Number | Object Name          | Description   |
|-----|---------------|----------------------|---|
| 6   | 6040h         | Controlword          | controls the state machine and the nominal speed (vI) |
|     | 6042h         | Target velocity (vl) |   |
| 6   | 6041h         | Statusword           | shows status and the current speed (vI)               |
|     | 6044h         | vl control effort    |   |

The remaining PDOs can be set by the user.

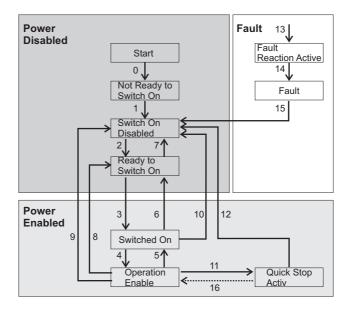
The following DS402 objects are implemented:

| Object no. |        | Description                   | Туре                        | Access | Mandatory |
|------------|--------|-------------------------------|-----------------------------|--------|-----------|
| 6040h      | VAR    | Controlword                   | UNSIGNED16                  | rw     | М         |
| 6041       | VAR    | Statusword                    | UNSIGNED16                  | ro     | М         |
| 6042h      | VAR    | vl target velocity (1)        | INTEGER16                   | rw     | М         |
| 6043h      | VAR    | vl velocity demand (1)        | INTEGER16                   | ro     | М         |
| 6044h      | VAR    | vl control effort(1)          | INTEGER16                   | ro     | М         |
| 6046h      | ARRAY  | vl velocity min max<br>amount | UNSIGNED32                  | rw     | М         |
| 6048h      | RECORD | vl velocity acceleration      | vl velocity<br>acceleration | rw     | М         |
| 6049       | RECORD | vl velocity deceleration      | vl velocity<br>deceleration | rw     | М         |
| 6060h      | VAR    | Modes of operation (2)        | INTEGER8                    | rw     | М         |
| 6061h      | VAR    | Modes of operation<br>display | INTEGER8                    | ro     | Μ         |

(1) The unit of measure for objects 6042h, 6043h, 6044h is expressed in rpm

(2) Object 6060h is only available in that it is mandatory. As the drive only supports Velocity Mode, the value of this object is not modifiable.

The device operates as a DS402 state machine (refer to CiA DSP 402 V 2.0, section 10.1.1):



#### 3.0 DeviceNet Interface

This chapter describes the connecting of ADV200 drives to DeviceNet networks. It is intended for design engineeres and technicians responsible for the maintenance, commissioning and operation of DeviceNet systems.

A basic knowledge of DeviceNet is assumed and may be found in the following manuals:

- DeviceNet Specifications. Volume 1 DeviceNet Communication Model and Protocol (Issued by ODVA).
- DeviceNet Specifications. Volume 2 DeviceNet Device Profiles and Object Library (Issued by ODVA)

#### 3.1 General description of DeviceNet

DeviceNet is a profile of communication for industrial systems based on CAN. As protocol CAN (ISO 11898) is used CAN2.0A with the 11 bit identifier. The ADV200 driver is developed as "Slave UCMM Capable Device" for operating only in "Predefined Master/Slave Connection Set".

The data transfer is carried out cyclically; the Master unit reads the data supplied by the Slaves and writes the Slave reference data; the Baud Rate supported by the SBI card are: 125 kbit, 250 kbit, 500 kbit.

The physical support is given by the RS485 serial line; a maximum of 64 Slaves can be connected to the Bus.

#### 3.2 DeviceNet function

In this chapter are described the functions of DeviceNet managed by the driver. The main characteristics of the card are:

- 1. The drive operates only as Slave in "Predifined Master/Slave Connection Set".
- 2. Within the "Predefined Master/Slave Connection Set" the driver is a "UCMM Capable Device".
- 3. The "Explicit Messaging" is managed.
- 4. The "Polling" for the fast cyclical data exchange Master/Slave is managed.
- 5. The detection mechanism of the "Duplicate MAC ID" is implemented.

Regarding the "Explicit Messaging" the fragmentation of the data frame, with a total of max. 32 byte, is managed.

#### Connection sizes

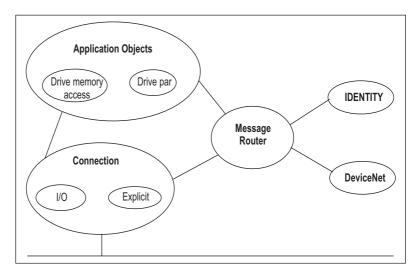
| CONNECTION INSTANCE |             | PRODUCED                   | CONSUMED |
|---------------------|-------------|----------------------------|----------|
| Polled I            | /0          | Depending on frame setting |          |
| Explicit            | : messaging | 32                         | 32       |

#### 3.3 Object description

Hereafter you find the description of the objects managed by the ADV200 drive.

#### 3.3.1 Object Model

The following figure shows the ADV200 "Object Model".



The following table shows:

- 1. The object classes of EXP-CAN-ADV card.
- 2. If the class is mandatory.
- 3. The number of instances included in every class.

See "DeviceNet Specifications" for the Standard classes.

| Object                 | Optional/Required | # of Instances    |
|------------------------|-------------------|-------------------|
| Identity               | Required          | 1                 |
| Message Router         | Required          | 1                 |
| DeviceNet              | Required          | 1                 |
| Connection             | Required          | 1 I/O, 3 Explicit |
| Parameter              | Optional          | many              |
| Drive Parameter Access | Optional          | many              |
| Drive memory Access    | Optional          | many              |

#### 3.3.2 How Objects Affect Behavior

The "Affect Behaviour" of the objects is reported in the following table:

| Object                 | Effect on Behavior                   |
|------------------------|--------------------------------------|
| Identity               | Supports Reset Service               |
| Message Router         | No effect                            |
| DeviceNet              | Port attributes configuration        |
| Connection             | Contains the number of logical ports |
| Parameter              | Drive parameters read/write          |
| Drive Parameter Access | Drive parameters read/write          |
| Drive Memory Access    | Drive parameters read/write          |

#### 3.3.3 Defining Object Interface

The object interface of the ADV200 drive is the following:

| Object                 | Interface                              |
|------------------------|--|
| Identity               | Message router                         |
| Message Router         | Explicit Messaging Connection Instance |
| DeviceNet              | Message router                         |
| Connection             | Message router                         |
| Parameter              | Message router                         |
| Drive Parameter Access | Message router                         |
| Drive memory Access    | Message router                         |

#### 3.4 Data transfert via Explicit Messaging

The data transfer via Explicit Messaging is made through two new objects: one for accessing the Drive parameters, the other to direct access the drive memory.

#### 3.4.1 Parameter Access

For read/write of Drive parameters, the Drive Parameter Access object is defined with the following properties:

- Class ID: Fh.
- Class Attribute: Revision
- Instance Attribute: This instance does not have attributes.

#### 3.4.1.1 Class Code

Class code: F hex

#### 3.4.1.2 Class attributes

| Number |          | Access<br>Rule | Name     | DeviceNet<br>Data Type | ·                       | Semantics<br>of values |
|--------|----------|----------------|----------|------------------------|-------------------------|------------------------|
| 1      | Optional | Get            | Revision | UINT                   | Revision of this object |                        |

#### 3.4.1.3 Instance Attributes

| Number   |  | Access<br>Rule |  |  | Description of<br>Attribute | Semantics<br>of values |  |
|--|--|----------------|--|--|-----------------------------|------------------------|--|
| This instance provide attributes and correspond to the PAR |  |                |  |  |                             |                        |  |

#### 3.4.1.4 Common Services

This object has no common services.

#### 3.4.1.5 Object Specific Services

| Service Code | Need in i | mplementation | Service Name         | Description of Service       |
|--------------|-----------|---------------|----------------------|------------------------------|
|              | Class     | Instance      |                      |                              |
| Ohex         | n/a       | Required      | Get_Attribute_Single | Read drive parameter value   |
| 10hex        | n/a       | Required      | Set_Attribute_Single | Writes drive parameter value |

#### 3.4.1.6 Behavior

This object is the interface between the DeviceNet and all drive parameters. The Drive parameter is accessed via the parameter index itself. *For example, reading a parameter (PAR 600 Dig Ramp ref 1):* 

- Run a Get Attribute Single from class Fh
- instance =600 (258 hex)
- class 1 attribute
- the drive responds with 4 bytes (Dword format).

For example, writing a parameter (PAR 600 Dig Ramp ref 1):

- Run a Set\_Attribute\_Single from class Fh
- instance = 600 (258 hex)
- class 1 attribute
- to set value 1000, select "Word 2 byte" (parameter format is INT, 16 bit)
- the drive does not respond if there is an error (timeout).

|            |       |  | Low byte - Low word drive parameter drive   |
|------------|-------|--|---|
| byte VALUE |       | High byte - Low word drive parameter drive   |   |
|            | VALUE | VALUE XX Low byte - High word drive parameter drive<br>High byte - High word drive parameter drive | Low byte - High word drive parameter drive  |
|            |       |  | High byte - High word drive parameter drive |

The number of bytes in the "Value" field depends on the length of drive parameter.

#### Example:

if the type of drive parameter is "Integer" the length of VALUE is 2 bytes.

#### 3.4.2 Drive Parameter Access

For read/write of Drive parameters, the Drive Parameter Access object is defined with the following properties:

- Class ID: 66h.
- Class Attribute: Revision
- Instance Attribute: This instance does not have attributes.

#### 3.4.2.1 Class Code

Class code: 66 hex

#### 3.4.2.2 Class attributes

| Number | Need in<br>implementation | Access<br>Rule | Name     | DeviceNet<br>Data Type |                         | Semantics<br>of values |
|--------|---------------------------|----------------|----------|------------------------|-------------------------|------------------------|
| 1      | Optional                  | Get            | Revision | UINT                   | Revision of this object |                        |

#### 3.4.2.3 Instance Attributes

| Num                                       | ber | Need in<br>implementation | Access<br>Rule | Name | DeviceNet<br>Data Type |  | Semantics<br>of values |
|---|-----|---------------------------|----------------|------|------------------------|--|------------------------|
| This instance does not provide attributes |     |                           |                |      |                        |  |                        |

#### 3.4.2.4 Common Services

This object has no common services.

| Service Code | Need in implementation |          | Service Name              | Description of Service                                   |
|--------------|------------------------|----------|---------------------------|--|
|              | Class                  | Instance |                           |  |
| 32hex        | n/a                    | Required | Get_Drive_Value           | Get parameter from Drive                                 |
| 33hex        | n/a                    | Required | Set_Drive_Value           | Set parameter into Drive                                 |
| 34hex        | n/a                    | Required | Get_Typed_Drive_<br>Value | Read drive parameter value indicating the data type      |
| 35hex        | n/a                    | Required | Set_Typed_Drive_<br>Value | Writes drive parameter value<br>indicating the data type |

3.4.2.5 Object Specific Services

#### 3.4.2.6 Behavior

This object is the interface between the DeviceNet network and all Drive parameters. The access to the Drive parameter is carried out by the parameter index; if the parameter does not exist or may not be accessed for any reason (for example: try to write a read only parameter) an error code will be returned.

Drive parameters in text format cannot be accessed.

In the following are repeted patterns of how the data frame of data has to be composed for reading/writing Drive parameters.

A) Write Drive Parameter

In this example the writing of a Drive parameter is shown; the cases of positive or wrong writing are distinguished.

#### A-1) Write Drive Parameter Request

The data frame for writing a drive parameter is composed as follows:

| DATA TYPE   | FIELD        | VALUE | MEANING  |  |  |
|-------------|--------------|-------|--|--|--|
| Byte        | Service Code | 33hex | Set Drive Parameter - Object Specific Service      |  |  |
| Can Nata 1) | Class ID     | 66hex | Drive Parameter Access Class Object                |  |  |
| See Note 1) | Instance ID  | XXXX  | Drive Parameter Index in format Low byte-High byte |  |  |
|             | VALUE        | xx    | Low byte-Low word drive parameter value            |  |  |
| Byte 2)     |              |       | High byte-Low word drive parameter value           |  |  |
| Dyte 2)     | VALUE        |       | Low byte-High word drive parameter value           |  |  |
|             |              |       | High byte-High word drive parameter value          |  |  |

1) Byte or Word depending on the type of allocation executed by the Master.

 The number of bytes of the "Value"-field depends on the length of the Drive parameter; i.e.: if the Drive parameter type is "Integer" the length of VALUE is 2 bytes.

A-2) Write drive parameter - Reply OK

If the Drive parameter is written correctly, the response is:

| DATA TYPE | FIELD        | VALUE    | MEANING  |
|-----------|--------------|----------|--|
| Byte      | Service Code | 33hex OR | Set Drive Parameter Reply code- Object Specific Service.     |
|           |              | 80hex    |  |
| Word      | Result       | 0000     | Result field equal to zero means writing correctly executed. |

#### A-3) Write drive parameter - Reply Error

If the writing of the drive parameter has been rejected, the response is the following:

| DATA TYPE | FIELD        | VALUE    | MEANING  |
|-----------|--------------|----------|--|
| Byte      | Service Code | 33hex OR | Set Drive Parameter Reply code- Object Specific Service. |
|           |              | 80hex    |  |
| Word      | Result       | XXXX 1   | Drive specific error code.                               |

1) For error codes see table 3.4.1.

#### B) Read Drive Parameter

In this example is shown the reading of a Drive parameter; the cases of positive or wrong reading are distinguished.

#### B-1) Read Drive Parameter Request

The data frame for the Drive parameter reading is composed as follows:

| DATA TYPE   | FIELD        | VALUE | MEANING  |
|-------------|--------------|-------|--|
| Byte        | Service Code | 32hex | Get Drive Parameter - Object Specific Service.     |
| See Note 1) | Class ID     | 66hex | Drive Parameter Access Class Object.               |
| See Note 1) | Instance ID  | XXXX  | Drive Parameter Index in format Lowbyte-High byte. |

1) Byte or Word depending on the type of allocation executed by the Master.

#### B-2) Read drive parameter - Reply OK

If the Drive parameter is read correctly, the response is:

| DATA TYPE | FIELD        | VALUE | MEANING  |
|-----------|--------------|-------|--|
| Byte      | Service Code | 32hex | Get Drive Parameter Reply code- Object Specific Service.     |
| Word      | Result       | 0     | Result field equal to zero means reading correctly executed. |
|           | VALUE        | хх    | Low byte-Low word drive parameter value.                     |
| Dute 1)   |              |       | High byte-Low word drive parameter value.                    |
| Byte 1)   |              |       | Low byte-High word drive parameter value.                    |
|           |              |       | High byte-High word drive parameter value.                   |

 The number of bytes of the Value-field depends on the length of the Drive parameter; i.e. if the Drive parameter type is "Integer" the length of VALUE is 2 bytes.

#### **B-3)** Read drive parameter - Reply Error

If Drive parameter reading is rejected, the response is the following:

| DATA TYPE | FIELD        | VALUE  | MEANING  |  |
|-----------|--------------|--------|--|--|
| Byte      | Service Code | 32hex  | Get Drive Parameter Reply code- Object Specific Service. |  |
| Word      | Result       | XXXX 1 | Drive specific error code.                               |  |

1) For error codes see table 3.4.1.

#### C) Write Typed Drive Parameter

In this example the writing of a Drive parameter is shown; the cases of positive or wrong writing are distinguished.

In this case, it is shown the parameter IPA number, the value and the data type used in the data transmission.

The optional data type conversion is automatically executed by the firmware.

#### C-1) Write Drive Parameter Request

The data frame for writing a drive parameter is composed as follows:

| DATA TYPE               | FIELD        | VALUE | MEANING  |  |
|-------------------------|--------------|-------|--|--|
| Byte                    | Service Code | 35hex | Set Drive Parameter - Object Specific Service      |  |
| See Note 1)             | Class ID     | 66hex | Drive Parameter Access Class Object                |  |
| See Note 1) Instance ID | Instance ID  | XXXX  | Drive Parameter Index in format Low byte-High byte |  |
| Byte 2)                 | DATA TYPE    | XX    | Value data type                                    |  |
|                         |              | хх    | Low byte-Low word drive parameter value            |  |
| Durte 21                | VALUE        |       | High byte-Low word drive parameter value           |  |
| Byte 3)                 |              |       | Low byte-High word drive parameter value           |  |
|                         |              |       | High byte-High word drive parameter value          |  |

1) Byte or Word depending on the type of allocation executed by the Master.

2) The coding of the possible data type is listed in table 3.4.2.

3) The number of bytes of the "Value"-field depends on the length of the Drive parameter; i.e.: if the Drive parameter type is "Integer" the length of VALUE is 2 bytes.

#### C-2) Write drive parameter - Reply OK

If the Drive parameter is written correctly, the response is:

| DATA TYPE | FIELD        | VALUE | MEANING  |  |
|-----------|--------------|-------|--|--|
| Byte      | Service Code | 33hex | Set Drive Parameter Reply code- Object Specific Service.     |  |
| Word      | Result       | 0000  | Result field equal to zero means writing correctly executed. |  |

#### C-3) Write drive parameter - Reply Error

If the writing of the drive parameter has been rejected, the response is the following:

| DATA TYPE | FIELD        | VALUE   | MEANING  |  |
|-----------|--------------|---------|--|--|
| Byte      | Service Code | 33hex   | Set Drive Parameter Reply code- Object Specific Service. |  |
| Word      | Result       | XXXX 1) | Drive specific error code.                               |  |

1) For error codes see table 3.4.1.

#### D) Read Drive Parameter

In this example is shown the reading of a Drive parameter; the cases of positive or wrong reading are distinguished.

In this case, it is shown the parameter IPA number, the value and the data type used in the data transmission.

The optional data type conversion is automatically executed by the firmware.

#### D-1) Read Drive Parameter Request

The data frame for the Drive parameter reading is composed as follows:

| DATA TYPE   | FIELD        | VALUE | MEANING  |  |
|-------------|--------------|-------|--|--|
| Byte        | Service Code | 36hex | Get Drive Parameter - Object Specific Service.     |  |
|             | Class ID     | 66hex | Drive Parameter Access Class Object.               |  |
| See Note 1) | Instance ID  | XXXX  | Drive Parameter Index in format Lowbyte-High byte. |  |
| Byte 2)     | DATA TYPE    | XX    | Value data type                                    |  |

1) Byte or Word depending on the type of allocation executed by the Master.

2) The coding of the possible data type is listed in table 3.4.2.

D-2) Read drive parameter - Reply OK

If the Drive parameter is read correctly, the response is:

| DATA TYPE | FIELD        | VALUE | MEANING  |
|-----------|--------------|-------|--|
| Byte      | Service Code | 32hex | Get Drive Parameter Reply code- Object Specific Service.     |
| Word      | Result       | 0     | Result field equal to zero means reading correctly executed. |
|           | VALUE        | хх    | Low byte-Low word drive parameter value.                     |
| Dute 1)   |              |       | High byte-Low word drive parameter value.                    |
| Byte 1)   |              |       | Low byte-High word drive parameter value.                    |
|           |              |       | High byte-High word drive parameter value.                   |

 The number of bytes of the Value-field depends on the length of the Drive parameter; i.e. if the Drive parameter type is "Integer" the length of VALUE is 2 bytes.

#### D-3) Read drive parameter - Reply Error

If Drive parameter reading is rejected, the response is the following:

| DATA TYPE | FIELD        | VALUE  | MEANING  |
|-----------|--------------|--------|--|
| Byte      | Service Code | 32hex  | Get Drive Parameter Reply code- Object Specific Service. |
| Word      | Result       | XXXX 1 | Drive specific error code.                               |

1) For error codes see table 3.4.1.

| Table 3.4.1: Erro | r codes for the | parameter access |
|-------------------|-----------------|------------------|
|-------------------|-----------------|------------------|

| Code  | Descrption                                     |  |  |
|-------|--|--|--|
| 1     | Incorrect parameter number                     |  |  |
| 9     | Maximum value exceeded                         |  |  |
| 10    | Minimum value exceeded                         |  |  |
| 11    | Value not allowed for the parameter            |  |  |
| 12,13 | Read-only parameter                            |  |  |
| 16,31 | Parameter cannot be written with drive enabled |  |  |
| 20    | Parameter loading error                        |  |  |
| 21    | Error saving parameter                         |  |  |
| 23    | Parameter timeout                              |  |  |
| Other | Generic error, request technical assistance    |  |  |

Table 3.4.2: Parameter format

| FORMAT     | VALUE | MEANING                                 |
|------------|-------|---|
| DB_T_VOID  | 0     | Ritorno al valore nel formato originale |
| DB_T_INT   | 3     | 16 bit con segno                        |
| DB_T_WORD  | 6     | 16 bit senza segno                      |
| DB_T_LONG  | 4     | 32 bit con segno                        |
| DB_T_DWORD | 7     | 32 bit senza segno                      |
| DB_T_FLOAT | 8     | Formato Float IEEE 754                  |

#### 3.5 Polling function

This type of DeviceNet-function is used for a fast cyclic exchange of Drive-parameters between Master and ADV200 drive.

The characteristics of the Polling-function are:

1. The data frame length is configurable through specific drive parameter (see

COMMUNICATION menu) and can vary from 1 to 16 word for both directions (Slave->Master and Master->Slave).

The card, as it is a Slave, during the Polling consumes Output data and produces Input data as response.

The configuration of the Drive parameters transferred via Polling is set by using configuration parameter allocated in the drive (see COMMUNICATIONS menu).

#### 3.6 Devicenet Interface configuration

The DeviceNet interface configuration is performed via the drive parameters. The parameters are controlled via hierarchical menus. All the writing parameters referring to the DeviceNet interface are active only after the drive reset. Here following is a list of drive parameters useful to control the DeviceNet interface

To activate the EXP-CAN-ADV card, set parameter PAR 4000 **Fieldbus type** to "DeviceNet".

The following parameters are available in the COMMUNICATION->FIELDBUS CONFIG menu:

| PAR  | Parameter Description | Туре            | Default value | Attr      |
|------|-----------------------|-----------------|---------------|-----------|
| 4004 | Fieldbus baudrate     | Enum            | None          | Write     |
| 4006 | Fieldbus address      | 2 byte unsigned | 0             | Write     |
| 4010 | Fieldbus M->S enable  | Enum            | On            | Write     |
| 4012 | Fieldbus alarm mode   | 2 byte unsigned | 0             | Write     |
| 4014 | Fieldbus state        | Enum            | Stop          | Read only |

- Fieldbus baudrate = Sets the network baud rate. Values available for Device-Net: 125k, 250k, 500k
- Fieldbus address = address of this slave node in the network, accepted values from 1 to 63
- Fieldbus M->S enable = if set to Off, master to slave Polling data are not managed
- Fieldbus alarm mode = if set to 1 the drive generates Opt Bus Fault errors relating to the loss of communication (Bus Loss) even when the drive is not enabled.
- Fieldbus state = state of the communication for this node on the DeviceNet network
  - Stop: No communication with the master
  - Pre-Operational: Recognition by master in progress
  - Operational: I/O Polling active

#### 3.7 Alarms

#### 3.7.1 DeviceNet Alarms

The **Opt Bus Fault** alarm indicates a bus malfunction. In case of DeviceNet, possible reasons for faults are:

- CAN line in Bus-off state;
- drive enabled in a state other than I/O Polling
- connection timeout limit exceeded.

This alarm is only activated when the drive is enabled.

If parameter PAR 4014 **Fieldbus alarm mode** is set to ON, the **Opt Bus Fault** alarm can be generated even with the drive disabled.

| Code     | Cfg | Description   | Action  |
|----------|-----|---|---|
| 0        |     | Bus Loss  | Check line for noise, terminations, problems with cabling                           |
| FF01     | *   | Fieldbus type does not match expansion card   | Verify if EXP-CAN-ADV card is properly<br>installed                                 |
| FF04     | *   | Error initializing CAN interface  | Internal error, contact manufacturer  |
| FF24FF33 | *   | More than 1 Src pointing to M2S<br>Channel n  | Check for multiple destinations on<br>"Fieldbus M->Sn Dest"                         |
| FF34FF43 | *   | M2S Channel n , data size is wrong<br>(16 bits on 32 bits or 32 bits on 16<br>bits parameter) | Check "Fieldbus M->Sn sys"  |
| FF44FF53 | *   | Invalid parameter in channel S2M n  | Check "Fieldbus S->Mn src"  |
| FF54FF63 | *   | S2M Channel n, data size is wrong<br>(16 bits on 32 bits or 32 bits on 16<br>bits parameter)  | Check "Fieldbus S->Mn sys"  |
| FF74FF83 | *   | M2S Channel n : too many words in PDC   | "Fieldbus M-Sn dest" & "Fieldubs<br>M->Sn sys" address more than 16<br>words in PDC |
| FF84FF93 | *   | S2M Channel n : too many words in PDC   | "Fieldbus S->Mn src" & "Fieldubs<br>S->Mn sys" address more than 16<br>words in PDC |
| FFB4FFC3 | *   | Internal database error on channel n  | Internal error, contact manufacturer  |
| 8110     |     | CAN msg overflow  | Too many packets for selected<br>baudrate   |
| FFC5     |     | Wrong baudrate  | Wrong baudrate  |
| FFC6     |     | Wrong MacID   | Check "Fieldbus address" is 1 to 63   |
| FFC7     |     | CAN bus off   | Check line state for problems   |
| FFC8     |     | System error on connection  | Check master for proper connection  |
| FFC9     |     | Duplicate MacID Check failed  | Check "Fieldbus address" is unique<br>in the network                                |

#### 3.7.2 Drive alarm handling

The "drive alarm status" is not foreseen.

The "drive alarm status" is not therefore given any special treatment.

The ADV200 firmware, provides a series of parameters capable of detecting the drive status. Refer to drive manual for more information.

#### 3.8 Process Data Channel Control

This function allows to allocate the drive parameters or application variables to the Process Data Channel data.

The ADV200 DeviceNet interface uses a number of words for the Process Data Channel (abbr. PDC Process Data Channel), which can always be set. The Process Data Channel configuration for the ADV200 interface is the following:

DATA 0 DATA... DATAn

The Slave can both read and write the Process Data Channel data. The DeviceNet data read by the Slave are defined as input data; the data written in DeviceNet by the Slave are defined as output data. A datum can be made both of 2 and 4 bytes. The word "data" refers to any guantity of bytes included between 0 and 16, if the byte total number required is not higher than 32.

#### Example:

It is possible to have:

- from 0 to 16 data items of 2 byte
- 1 datum of 4 bytes + from 0 to 14 data items of 2 bytes
- 2 data items of 4 bytes + from 0 to 12 data items of 2 bytes
- 8 data items of 4 bytes

The data exchanged via the PDC can be of two types: -drive parameters -variables of an MDPIc application

The composition of the PDC input and output data is defined via suitable parameters as described in the paragraphs 3.8.1 and 3.8.2.

The master cyclically writes the data defined as PDC input and cyclically reads the data defined as PDC output.

#### 3.8.1 PDC Input Configuration (SYS FB XXX MS parameter)

See section 131

## 3.8.2 PDC Output Configuration (SYS FB XXX SM Parameter)

See section 1.3.2.

#### 3.8.3 Configuration of the Virtual Digital I/Os

The ADV200 firmware, provides the Word Comp and Word Decomp functions. which allows to exchange discrete signals between the master and the slave and vice versa.

Commands can be sent to the drive using the functions of PAR 4452 Word decomp. The meaning of the single bits is programmable. It can be set on a Fieldbus M->Sn channel as Count 16.

The drive state is read in PAR 4432 Word Comp, programmable on any Fieldbus S->Mn channel as Count 16. The meaning of each single bit can be selected by the user using PAR 4400 Word Bit 0 src ... PAR 4430 Word Bit 15 src.

For a detailed description of these parameters see the drive manual.

#### 3.8.4 Use of the PDC in MDPIc Applications

Refer to section 1.3.3 Using the PDC in MDPLC applications.

#### 3.9 Configuration example

See chapter 1.6.

On paragraph 1.6.2.2 : P4000 Fieldbus Type = DeviceNet.

"Polling I/O" input / output area dimensions: with the examples reported on paragraphs 1.6.1.1 and 1.6.1.2 there are 4 bytes for writing and 6 bytes for reading.

#### References

- CiA: CAN in Automation, user international group.
- CAN : Controller Area Network.
- PDO: Process Data Object, service messages without confirmation used for the real time data transfer from/to the device.
- DBT: Distributor. It is a service element of the CAN Application Layer in the CAN Reference Model; the DBT has the task to assign COB-ID to the COBs used by the CMS.
- SDO: Service Data Object, service messages with confirmation used for the acyclic data transfer from/to the device.
- NMT: Network Management.
  - It is a service element of the CAN Application Layer in the CAN Reference Model; it initializes, configures and controls the errors of a CAN network.
- CS: Command Specifier; it defines the NMT service.
- COB-ID COB-Identifier. It identifies a COB inside the network. It also states the COB priority.