DCS800

Firmware manual
DCS800 Drives (20 to 5200 A)

DCS800 Drive Manuals

All the documents available for the drive system DCS800 are listed below:

		Language							
	Public. number	E	D	I	ES	F	CN	RU	PL
DCS800 Quick Guide	3ADW000191	x	x	x	x	x			
DCS800 Tools & Documentation CD	3ADW000211	x							
DCS800 Converter module									
Flyer DCS800	3ADW000190	x	x		x	x			
Technical Catalogue DCS800	3ADW000192	x	x	x	x	x	x	x	
Hardware Manual DCS800	3ADW000194	x	x	x	x	x	x	x	x
Hardware Manual DCS800 update DCF503B/DCF504B	3ADW000194Z0301	x							
Firmware Manual DCS800	3ADW000193	x	x	p	x	x	x	x	x
Installation according to EMC	3ADW000032	x							
Technical Guide	3ADW000163	x							
Service Manual DCS800	3ADW000195	x	x						
12-Pulse Manual	3ADW000196	x							
CMA-2 Board	3ADW000136	p							
Flyer Hard - Parallel	3ADW000213	x							
Drive Tools									
DriveWindow 2.x - User's Manual	3BFE64560981	x							
DriveOPC 2.x - User's Manual	3BFE00073846	x							
Optical DDCS Communication Link	3AFE63988235	x							
DDCS Branching Units - User's Manual	3BFE64285513	x							
DCS800 Applications									
PLC Programming with CoDeSys	CoDeSys_V23	x	x			x			
61131 DCS800 target +tool description - Application Program	3ADW000199	x							
DCS800 Crane Drive									
DCS800 Crane Drive Manual suppl.	3AST004143	x							
DCS800 Crane Drive Product note	PDC5 EN REVA	p							
DCS800 Winder ITC									
DCS800 Winder Product note	PDC2 EN	x							
DCS800 Winder description ITC	3ADW000308	x							
Winder Questionnaire	3ADW000253z	x							
DCS800-E Panel Solution									
Flyer DCS800-E Panel solution	3ADW000210	x							
Hardware Manual DCS800-E	3ADW000224	x							
DCS800-A Enclosed Converters									
Flyer DCS800-A	3ADW000213	x							
Technical Catalogue DCS800-A	3ADW000198	x							
Installation of DCS800-A	3ADW000091	x	x						
DCS800-R Rebuild System									
Flyer DCS800-R	3ADW000007	x	x						
DCS800-R Manual	3ADW000197	x							
DCS500/DCS600 Size A5...A7, C2b, C3 and C4 Upgrade Kits	3ADW000256	x							
Extension Modules									
RAIO-01 Analog IO Extension	3AFE64484567	x							
RDIO-01 Digital IO Extension	3AFE64485733	x							
RRIA-01 Resolver Interface Module	3AFE68570760	x							
RTAC-01 Pulse Encoder Interface	3AFE64486853	x							
RTAC-03 TTL Pulse Encoder Interface	3AFE68650500	x							
AIMA R-slot extension	3AFE64661442	x							
Serial Communication									
Drive specific serial communication									
NETA Remote diagnostic interface	3AFE64605062	x							
Fieldbus Adapter with DC Drives RPBA- (PROFIBUS)	3AFE64504215	x							
Fieldbus Adapter with DC Drives RCAN-02 (CANopen)									
Fieldbus Adapter with DC Drives RCNA-01 (ControlNet)	3AFE64506005	x							
Fieldbus Adapter with DC Drives RDNA- (DeviceNet)	3AFE64504223	x							
Fieldbus Adapter with DC Drives RMBA (MODBUS)	3AFE64498851	x							
Fieldbus Adapter with DC Drives RETA (Ethernet)	3AFE64539736	x							
x -> existing p -> planned									
Status 04.2010									

Safety instructions

What this chapter contains

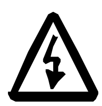
This chapter contains the safety instructions you must follow when installing, operating and servicing the drive. If ignored, physical injury or death may follow, or damage may occur to the drive, the motor or driven equipment. Read the safety instructions before you work on the unit.

To which products this chapter applies

The information is valid for the whole range of the product DCS800, the converter modules DCS800-S0x size D1 to D7, field exciter units DCF80x, etc. like the Rebuild Kit DCS800-R00-9xxx.

Usage of warnings and notes

There are two types of safety instructions throughout this manual: warnings and notes. Warnings caution you about conditions which can result in serious injury or death and/or damage to the equipment, and advise on how to avoid the danger. Notes draw attention to a particular condition or fact, or give information on a subject. The warning symbols are used as follows:



Dangerous voltage warning warns of high voltage which can cause physical injury or death and/or damage to the equipment.

General danger warning warns about conditions, other than those caused by electricity, which can result in physical injury or death and/or damage to the equipment.

Electrostatic sensitive devices warning warns of electrostatic discharge which can damage the equipment.

Installation and maintenance work

These warnings are intended for all who work on the drive, motor cable or motor. Ignoring the instructions can cause physical injury or death and/or damage to the equipment.

WARNING!

- **Only qualified electricians are allowed to install and maintain the drive!**
- Never work on the drive, motor cable or motor when main power is applied.
  Always ensure by measuring with a multimeter (impedance at least 1 Mohm) that:
  1. Voltage between drive input phases U1, V1 and W1 and the frame is close to 0 V.
  2. Voltage between terminals C+ and D- and the frame is close to 0 V.
- Do not work on the control cables when power is applied to the drive or to the external control circuits. Externally supplied control circuits may cause dangerous voltages inside the drive even when the main power on the drive is switched off.
- Do not make any insulation resistance or voltage withstand tests on the drive or drive modules.
- Isolate the motor cables from the drive when testing the insulation resistance or voltage withstand of the cables or the motor.
- When reconnecting the motor cable, always check that the C+ and D- cables are connected with the proper terminal.

Note:

- The motor cable terminals on the drive are at a dangerously high voltage when the main power is on, regardless of whether the motor is running or not.
- Depending on the external wiring, dangerous voltages (115 V, 220 V or 230 V) may be present on the relay outputs of the drive system (e.g. SDCS-IOB-2 and RDIO).
- DCS800 with enclosure extension: Before working on the drive, isolate the whole drive system from the supply.

Grounding

These instructions are intended for all who are responsible for the grounding of the drive. Incorrect grounding can cause physical injury, death and/or equipment malfunction and increase electromagnetic interference.

WARNING!

- Ground the drive, motor and adjoining equipment to ensure personnel safety in all circumstances, and to reduce electromagnetic emission and pick-up.
- Make sure that grounding conductors are adequately sized and marked as required by safety regulations.
- In a multiple-drive installation, connect each drive separately to protective earth (PE ⏚).
- Minimize EMC emission and make a 360° high frequency grounding (e.g. conductive sleeves) of screened cable entries at the cabinet lead-through plate.
- Do not install a drive equipped with an EMC filter to an ungrounded power system or a high resistance-grounded (over 30 ohms) power system.

Note:

- Power cable shields are suitable as equipment grounding conductors only when adequately sized to meet safety regulations.
- As the normal leakage current of the drive is higher than 3.5 mA AC or 10 mA DC (stated by EN 50178, 5.2.11.1), a fixed protective earth connection is required.

Printed circuit boards and fiber optic cables

These instructions are intended for all who handle the circuit boards and fiber optic cables. Ignoring the following instructions can cause damage to the equipment.

WARNING! The printed circuit boards contain components sensitive to electrostatic discharge. Wear a grounding wrist band when handling the boards. Do not touch the boards unnecessarily.

Use grounding strip:

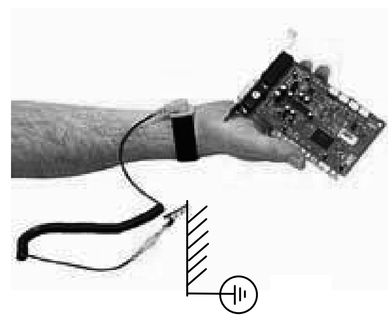


ABB order no.: 3ADV050035P0001

WARNING! Handle the fiber optic cables with care. When unplugging optic cables, always grab the connector, not the cable itself. Do not touch the ends of the fibers with bare hands as the fiber is extremely sensitive to dirt. The minimum allowed bend radius is 35 mm (1.38 in.).

Mechanical installation

These notes are intended for all who install the drive. Handle the unit carefully to avoid damage and injury.

WARNING!

- DCS800 sizes D4 ... D7: The drive is heavy. Do not lift it alone. Do not lift the unit by the front cover. Place units D4 and D5 only on its back.
  DCS800 sizes D5 ... D7: The drive is heavy. Lift the drive by the lifting lugs only. Do not tilt the unit. The unit will overturn from a tilt of about 6 degrees.
- Make sure that dust from drilling does not enter the drive when installing. Electrically conductive dust inside the unit may cause damage or lead to malfunction.
- Ensure sufficient cooling.
- Do not fasten the drive by riveting or welding.

Operation

These warnings are intended for all who plan the operation of the drive or operate the drive. Ignoring the instructions can cause physical injury or death and/or damage to the equipment.

WARNING!

- Before adjusting the drive and putting it into service, make sure that the motor and all driven equipment are suitable for operation throughout the speed range provided by the drive. The drive can be adjusted to operate the motor at speeds above and below the base speed.
- Do not control the motor with the disconnecting device (disconnecting mains); instead, use the control panel keys and , or commands via the I/O board of the drive.
- Mains connection
  You can use a disconnect switch (with fuses) to disconnect the electrical components of the drive from the mains for installation and maintenance work. The type of disconnect switch used must be as per EN 60947-3, Class B, so as to comply with EU regulations, or a circuit-breaker type which switches off the load circuit by means of an auxiliary contact causing the breaker's main contacts to open. The mains disconnect must be locked in its "OPEN" position during any installation and maintenance work.
- EMERGENCY STOP buttons must be installed at each control desk and at all other control panels requiring an emergency stop function. Pressing the STOP button on the control panel of the drive will neither cause an emergency stop of the motor, nor will the drive be disconnected from any dangerous potential.
  To avoid unintentional operating states, or to shut the unit down in case of any imminent danger according to the standards in the safety instructions it is not sufficient to merely shut down the drive via signals "RUN", "drive OFF" or "Emergency Stop" respectively "control panel" or "PC tool".
- Intended use
  The operating instructions cannot take into consideration every possible case of configuration, operation or maintenance. Thus, they mainly give such advice only, which is required by qualified personnel for normal operation of the machines and devices in industrial installations.
  If in special cases the electrical machines and devices are intended for use in non-industrial installations - which may require stricter safety regulations (e.g. protection against contact by children or similar) - these additional safety measures for the installation must be provided by the customer during assembly.

Note:

- When the control location is not set to Local (L not shown in the status row of the display), the stop key on the control panel will not stop the drive. To stop the drive using the control panel, press the LOC/REM key and then the stop key .

Table of contents

Introduction

Chapter overview

This chapter describes the purpose, contents and the intended use of this manual.

Before You Start

The purpose of this manual is to provide you with the information necessary to control and program the drive.

Study carefully the *Safety instructions* at the beginning of this manual before attempting any work on or with the drive. Read through this manual before starting-up the drive. The installation and commissioning instructions given in the *DCS800 Hardware Manual* and *DCS800 Quick Guide* must also be read before proceeding.

This manual describes the **standard** DCS800 firmware.

What this manual contains

The *Safety instructions* can be found at the beginning of this manual.

Introduction to this manual, the chapter you are currently reading, introduces you to this manual.

Start-up, this chapter describes the basic start-up procedure of the drive.

Firmware description, this chapter describes how to control the drive with **standard** firmware.

I/O configuration, this chapter describes the I/O configuration of digital and analog inputs and outputs with different hardware possibilities.

Communication, this chapter describes the communication capabilities of the drive.

Adaptive Program (AP), this chapter describes the basics of the Adaptive Program and instructs how to build a program.

Signal and parameter list, this chapter contains all signals and parameters.

DCS800 Control Panel operation, this chapter describes the handling of the DCS800 Control Panel.

Fault Tracing, this chapter describes the protections and fault tracing of the drive.

Appendix A: Firmware structure diagram

Appendix B: SDCS-CON-4 Terminal Allocation

Appendix C: Index of signal and parameters

Start-up

Chapter overview

This chapter describes the basic start-up procedure of the drive. A more detailed description of the signals and parameters involved in the procedure can be found in *section* *Signal and parameter list.*

General

The drive can be operated:

- locally from DriveWindow, DriveWindow Light or DCS800 Control Panel
- respectively remote from local I/O or overriding control.

The following start-up procedure uses DriveWindow (for further information about DriveWindow, consult its online help). However, parameters can also be changed with DriveWindow Light or the DCS800 Control Panel.

The start-up procedure includes actions that need only be taken when powering up the drive for the first time in a new installation (e.g. entering the motor data). After the start-up, the drive can be powered up without using these start-up functions again. The start-up procedure can be repeated later if the start-up data needs to be altered.

Refer to *section* *Fault tracing* in case problems should arise. In case of a major problem, disconnect mains and wait for 5 minutes before attempting any work on the drive, the motor, or the motor cables.

Start-up procedure

The *Safety Instructions* at the beginning of this manual have to be observed with extreme care during the start-up procedure!
The start-up procedure should only be carried out by a qualified electrician.

Check the mechanical and electrical installation the drive according to the *DCS800 Hardware Manual*.

Tools

For drive commissioning following software tools are mandatory:
- DriveWindow Light including commissioning wizard and DWL AP for Adaptive Program **and**
- DriveWindow for fast drive monitoring using SDCS-COM-8.

For drive commissioning following tools are mandatory in addition to standard tools:
- An oscilloscope including memory function with either galvanically isolating transformer or isolating amplifier for safe measurements.
- A clamp on current probe. In case the scaling of the DC load current needs to be checked it must be a DC clamp on current probe.
- A voltmeter.

Make sure that all equipment in use is suitable for the voltage level applied to the power part!

Checking with the power switched off

Check the settings of:
- the main breaker (e.g. overcurrent = 1.6 * I_n, short circuit current = 10 * I_n, time for thermal tripping = 10 s),
- time, overcurrent, thermal and voltage relays,
- the earth fault protection (e.g. Bender relay)

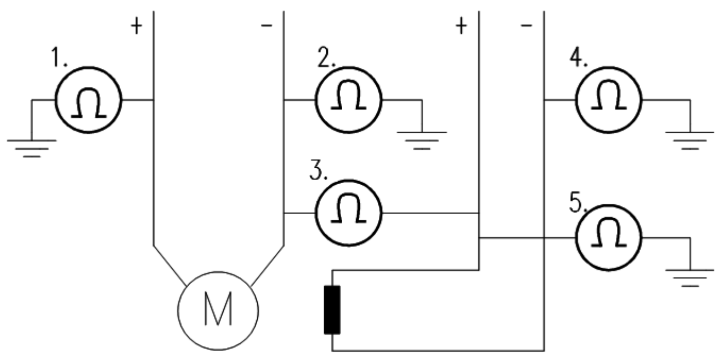
Check the insulation of the mains voltage cables or busbars between the secondary side of the dedicated transformer and the drive:
- disconnect the dedicated transformer from its incoming voltage,
- check that all circuits between the mains and the drive (e.g. control / auxiliary voltage) are disconnected,
- measure the insulation resistance between L1 - L2, L1 - L3, L2 - L3, L1 - PE, L2 - PE, L3 - PE,
- the result should be MΩs

Check the installation:
- crosscheck the wiring with the drawings,
- check the mechanical mounting of the motor and pulse encoder or analog tacho,
- make sure that the motor is connected in a correct way (armature, field, serial windings, cable shields),
- check the connections of the motor fan if existing,
- make sure that the converter fan is connected correctly especially in modules size D6 and D7 were star or delta connection is possible,

- if a pulse encoder is used make sure that pulse encoder's auxiliary voltage connection corresponds to its voltage and that the channel connection corresponds to correct direction of rotation,
- check that the shielding of the pulse encoder's cable is connected to the TE bar of the DCS800,
- if an analog tacho is used make sure that it is connected to the proper voltage input at the SDCS-CON-4:
  - X3:1 - X3:4 (90 - 270 V)
  - X3:2 - X3:4 (30 - 90 V)
  - X3:3 - X3:4 (8 - 30 V)
- for all other cables make sure that both ends of the cables are connected and they do not cause any damage or danger when power is being switched on

Measuring the insulation resistance of the motor cables and the motor:

- isolate the motor cables from the drive before testing the insulation resistance or voltage withstand of the cables or the motor,



Instructions how to measure the insulation resistance

- measure the insulation resistance between:
  1. + cables and PE,
  2. - cables and PE,
  3. armature cables and field cables,
  4. field - cable and PE,
  5. field + cable and PE,
- the result should be MΩs

Setting of Jumpers:
The boards of the DCS800 include jumpers to adapt the boards to different applications. The position of the jumpers must be checked before connecting voltage. For specific jumper settings consult the *DCS800 Hardware Manual*.

Drive data
Check following items for each drive and mark the differences in the delivery documents:

- motor, analog tacho or pulse encoder and cooling fan rating plates data,
- direction of motor rotation,
- maximum and minimum speed and if fixed speeds are used,
- speed scaling factors:
  e.g. gear ratio, roll diameter,
- acceleration and deceleration times,
- operating modes:
  e.g. stop mode, E-stop mode,
- the amount of motors connected

Checking with the power switched on

There is dangerous voltage inside the cabinet!

Switching the power on:

- prior to connecting the voltage proceed as follows:
  1. ensure that all the cable connections are checked and that the connections can't cause any danger,
  2. close all doors of enclosed converter before switching power on,
  3. be ready to trip the supply transformer if anything abnormal occurs,
  4. switch the power on

Measurements made with power on:

- check the operation of the auxiliary equipment,
- check the circuits for external interfaces on site:
  1. E-stop circuit,
  2. remote control of the main breaker,
  3. signals connected to the control system,
  4. other signals which remain to be checked

Connecting voltage to the drive:

- check from the delivery diagrams the type of boards and converters which are used in the system,
- check all time relay and breaker settings,
- close the supply disconnecting device (check the connection from the delivery diagrams),
- close all protection switches one at a time and measure for proper voltage

Commissioning a DCS800

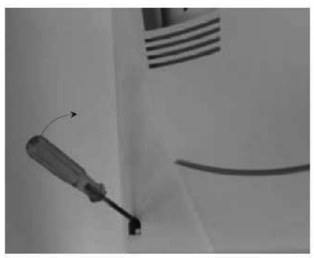
Nominal values of the converter can be found in group 4, check following signals:

- *ConvNomVolt (4.04)*, nominal AC converter voltage in V read from *TypeCode (97.01)* or *S ConvScaleVolt (97.03)*,
- *ConvNomCur (4.05)*, nominal converter DC current in A read from *TypeCode (97.01)* or *S ConvScaleCur (97.02)*,
- *ConvType (4.14)*, recognized converter type read from *TypeCode (97.01)*,
- *QuadrantType (4.15)*, recognized converter quadrant type read from *TypeCode (97.01)* or *S BlockBrdg2 (97.07)*,
- *MaxBridgeTemp (4.17)*, maximum bridge temperature in degree centigrade read from *TypeCode (97.01)* or *S MaxBrdgTemp (97.04)*

If signals are not correct adapt them, see group 97 in this manual.

Connect DCS800 to PC with DriveWindow Light

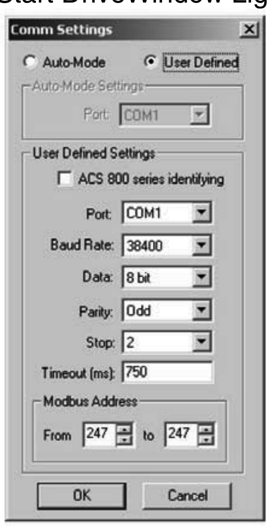
- Connect a normal serial cable from the PC COM port to X34 on the drive:



Remove the DCS800 Control Panel if present. Depress the locks to remove the cover

Connect drive (X34) to your PC COM port

- Start DriveWindow Light and check the communication settings:



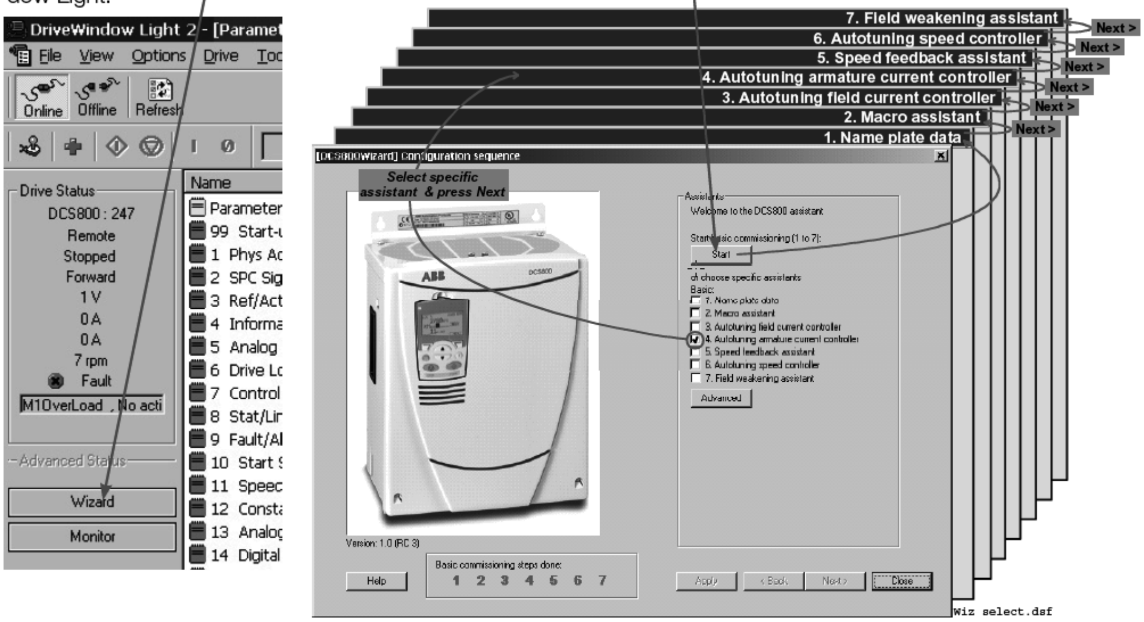
Example with COM1.

Commissioning a DCS800 with the wizard

To launch the commissioning wizard start DriveWindow Light and press the *Wizard* button:

Start the wizard in DriveWindow Light:

For basic commissioning press the *Start* button or select a specific assistant:

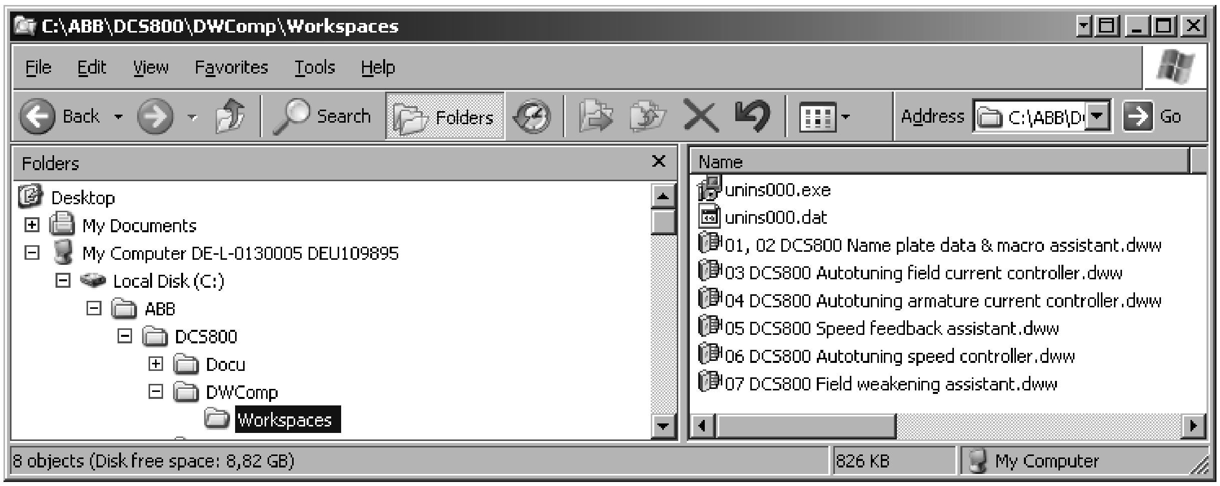


For more information about the wizard, parameters faults and alarms press the *Help* button!

Commissioning a DCS800 with DriveWindow

Requirements

1. Before starting with the commissioning, connect the drive (via Ch3 on SDCS-COM-8) with DriveWindow (via e.g. NDPA-02 and NDPC-12). All workspaces are ‘online’ workspaces, thus use *Ch3 NodeAddr (70.22)* = 1.
2. The preconfigured workspaces are available from Your local ABB agent or can be found - after the DCS800 CD (tools CD) is installed - under:



Location of workspaces

01, 02 Macro assistant / Name plate data

1. Open the workspace *01, 02 DCS800 Name plate data & macro assistant.dww*[1].
2. Set all parameters to default by means of *ApplMacro (99.08)* = **Factory** and *ApplRestore (99.07)* = **Yes**. Check with *MacroSel (8.10)*.
3. Enter the motor data, the mains (supply) data and the most important protections [*M1SpeedMin (20.01), M1SpeedMax (20.02), ArmOvrCurLev (30.09), M1OvrSpeed (30.16), Language (99.01), M1NomVolt (99.02), M1NomCur (99.03), M1BaseSpeed (99.04), NomMainsVolt (99.10)* and *M1NomFldCur (99.11)*].
4. After filling out the parameters it is - in most cases - possible to turn the motor for the first time.
5. Select an application macro by means of *ApplMacro (99.08)* = <macro> and *ApplRestore (99.07)* = **Yes**. Check with *MacroSel (8.10)*.

03 Autotuning field current controller

1. Open the workspace *03 DCS800 Autotuning field current controller.dww*[1].
2. Enter the field circuit data [*FldCtrlMode (44.01), M1NomFldCur (99.11)* and *M1UsedFexType (99.12)*].
3. Switch the drive to local mode (DriveWindow, DCS800 Control Panel or local I/O).
4. Start the autotuning by means of *ServiceMode (99.06)* = **FieldCurAuto** and set **On** within 20 s.
5. During the autotuning the main respectively field contactor will be closed, the field circuit is measured by means of increasing the field current to nominal field current and the field current control parameters are set. The armature current is not released while the autotuning is active and thus the motor should not turn.
6. When the autotuning is finished successfully, check *M1KpFex (44.02), M1TiFex (44.03)* and *M1PosLimCtrl (45.02)* - parameters set by the autotuning - for confirmation.
7. If the autotuning fails **A121 AutotuneFail** is set. For more details check *Diagnosis (9.11)* and repeat the autotuning.

04 Autotuning armature current controller

1. Open the workspace *04 DCS800 Autotuning armature current controller.dww*[1].
2. Enter the basic current limitations and the motor nominal current [*TorqMax (20.05), TorqMin (20.06), M1CurLimBrdg1 (20.12), M1CurLimBrdg2 (20.13)* and *M1NomCur (99.03)*].
   Attention:
   Do not change the default values of *M1ArmL (43.09)* and *M1ArmR (43.10)*! Changing them will falsify the results of the autotuning.
3. Switch the drive to local mode (DriveWindow, DCS800 Control Panel or local I/O).
4. Start the autotuning by means of *ServiceMode (99.06)* = **ArmCurAuto** and set **On** and **Run** within 20 s.
5. During the autotuning the main contactor will be closed, the armature circuit is measured by means of armature current bursts and the armature current control parameters are set. The field current is not released while the autotuning is active and thus the motor should not turn, but due to remanence in the field circuit about 40% of all motors will turn (create torque). These motors have to be locked.
6. When the autotuning is finished successfully, check *M1KpArmCur (43.06), M1TiArmCur (43.07), M1DiscontCurLim (43.08), M1ArmL (43.09)* and *M1ArmR (43.10)* - parameters set by the autotuning - for confirmation.
7. If the autotuning fails **A121 AutotuneFail** is set. For more details check *Diagnosis (9.11)* and repeat the autotuning.

05 Speed feedback assistant

1. Open the workspace *05 DCS800 Speed feedback assistant.dww*[1].
2. Enter the EMF speed feedback parameters and - if applicable - the parameters for pulse encoder 1, pulse encoder 2 or the analog tacho [*M1SpeedMin (20.01), M1SpeedMax (20.02), M1EncMeasMode (50.02), M1SpeedFbSel (50.03), M1EncPulseNo (50.04), M1TachoVolt1000 (50.13), M1NomVolt (99.02)* and *M1BaseSpeed (99.04)*].
3. Switch the drive to local mode (DriveWindow, DCS800 Control Panel or local I/O).
4. Start the autotuning by means of *ServiceMode (99.06)* = **SpdFbAssist** and set **On** and **Run** within 20 s.
5. The speed feedback assistant detects the kind of speed feedback - EMF, pulse encoder 1, pulse encoder 2 or analog tacho - the drive is using.
6. During the autotuning the main contactor and the field contactor - if existing - will be closed and the motor will run up to base speed [*M1BaseSpeed (99.04)*]. During the whole procedure the drive will be in EMF speed control despite the setting of *M1SpeedFbSel (50.03)*.
7. When the autotuning is finished successfully, check *M1SpeedFbSel (50.03)* - parameter set by the autotuning - for confirmation.
8. If the autotuning fails **A121 AutotuneFail** is set. For more details check *Diagnosis (9.11)* and repeat the autotuning.

Analog tacho fine tune procedure

1. In case an analog tacho is detected [*M1SpeedFbSel (50.03)* = **Tacho**] it is recommended to fine tune the analog tacho.
2. Switch the drive to local mode (DriveWindow, DCS800 Control Panel or local I/O).
3. Start the autotuning by means of *ServiceMode (99.06)* = **TachFineTune** and set **On** and **Run** within 20 s.
4. Measure the motor speed with a hand held tacho and write the value into *M1TachoAdjust (50.12)*.
5. Check *SpeedActTach (1.05)* against *SpeedRef4 (2.18)*.
6. Stop the autotuning by removing **Run** and **On** via the DriveWindow control panel.

06 Autotuning speed controller

1. Open the workspace *06 DCS800 Autotuning speed controller.dww*[1].
2. Enter the basic speed, torque and current limits, the speed filter times and the motor base speed [*M1SpeedMin (20.01), M1SpeedMax (20.02), TorqMax (20.05), TorqMin (20.06), M1CurLimBrdg1 (20.12), M1CurLimBrdg2 (20.13), SpeedErrFilt (23.06), SpeedErrFilt2 (23.11), SpeedFiltTime (50.06)* and *M1BaseSpeed (99.04)*].
   Attention:
   For better results set the filters, especially when using EMF speed feedback.
3. Switch the drive to local mode (DriveWindow, DCS800 Control Panel or local I/O).
4. Start the autotuning by means of *ServiceMode (99.06)* = **SpdCtrlAuto** and set **On** and **Run** within 20 s.
5. During the autotuning the main contactor and the field contactor - if existing - will be closed, the ramp is bypassed and torque respectively current limits

are valid. The speed controller is tuned by means of speed bursts up to base speed [*M1BaseSpeed (99.04)*] and the speed controller parameters are set.
Attention:
During the autotuning the torque and/or current limits will be reached.

6. When the autotuning is finished successfully, check *KpS (24.03)* and *TiS (24.09)* - parameters set by the autotuning - for confirmation.
7. If the autotuning fails **A121 AutotuneFail** is set. For more details check *Diagnosis (9.11)* and repeat the autotuning.

Attention:
The assistant is using the setting of *M1SpeedFbSel (50.03)*. If using setting **Encoder**, **Encoder2** or **Tacho** make sure the speed feedback is working properly!

07 Field weakening assistant

1. Open the workspace *07 DCS800 Field weakening assistant.dww*[1].
2. Enter the motor data and the field circuit data [*M1SpeedMin (20.01), M1SpeedMax (20.02), M1FldMinTrip (30.12), FldCtrlMode (44.01), M1NomVolt (99.02), M1BaseSpeed (99.04)* and *M1NomFldCur (99.11)*].
3. Switch the drive to local mode (DriveWindow, DCS800 Control Panel or local I/O).
4. Start the autotuning by means of *ServiceMode (99.06)* = **EMF FluxAuto** and set **On** and **Run** via within 20 s.
5. During the autotuning the main contactor and the field contactor - if existing - will be closed and the motor will run up to base speed [*M1BaseSpeed (99.04)*]. The EMF controller data are calculated, the flux linearization is tuned by means of a constant speed while decreasing the field current and the EMF controller respectively flux linearization parameters are set.
6. When the autotuning is finished successfully, check *KpEMF (44.09), TiEMF (44.10), FldCurFlux40 (44.12), FldCurFlux70 (44.13)* and *FldCurFlux90 (44.14)* - parameters set by the autotuning - for confirmation.
7. If the autotuning fails **A121 AutotuneFail** is set. For more details check *Diagnosis (9.11)* and repeat the autotuning.

[1]: before opening the workspaces, the drive has to be connected to DriveWindow

Manual tuning

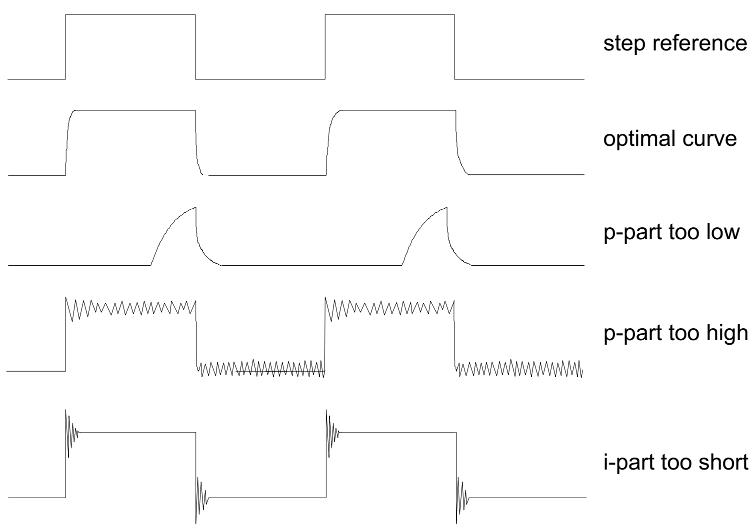
I/O configuration

To set the in- and outputs see chapter I/O configuration.

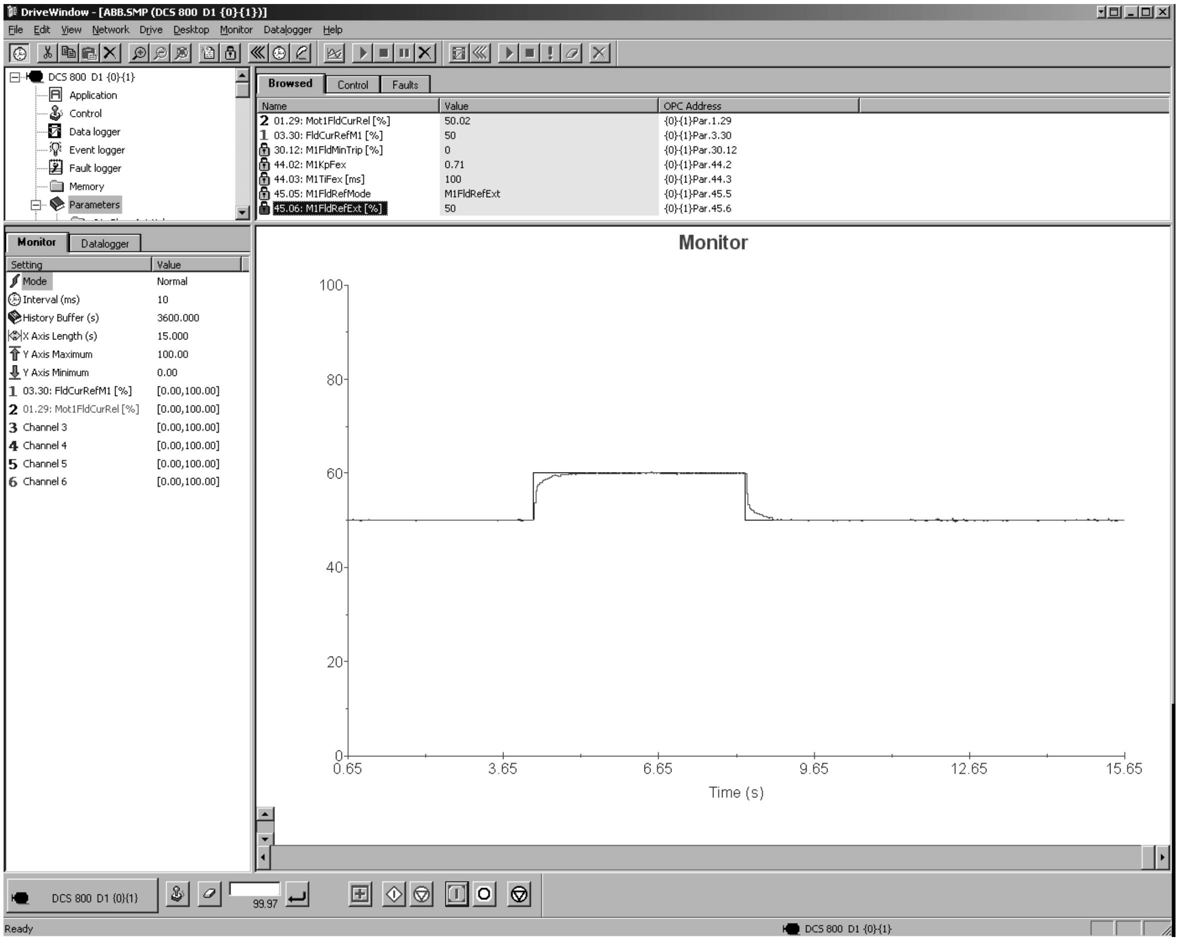
Field current controller

Manual tuning of the field current controller:

- connect DriveWindow to the drive and choose local mode,
- monitor *Mot1FldCurRef (1.29)* and *FldCurRefM1 (3.30)*,
- set *M1FldMinTrip (30.12)* = 0 %,
- set *M1FldRefMode (45.05)* = **M1FldRefExt**,
- give **On** via DriveWindow,
- use *M1FldRefExt (45.06)* to step the field current controller,
- tune the field current controller by means of *M1KpFex (44.02)* and *M1TiFex (44.03)*,
  - steps size: about 2 % - 5 % of nominal field current (do not hit any limits during the step and the step response, e.g. max. field current, or supply voltage),
  - step response time: 50 ms - 60 ms (count only from 10 % to 90 %),
  - where to step: 30 %, 60 % and 80 % of nominal field current,



Field current controller step responses



DriveWindow manual tuning field current controller

- set *M1FldRefExt (45.06)* = 0 %,
- remove **On** via DriveWindow,
- set *M1FldMinTrip (30.12)* and *M1FldRefMode (45.05)* back to their original settings

Armature current controller

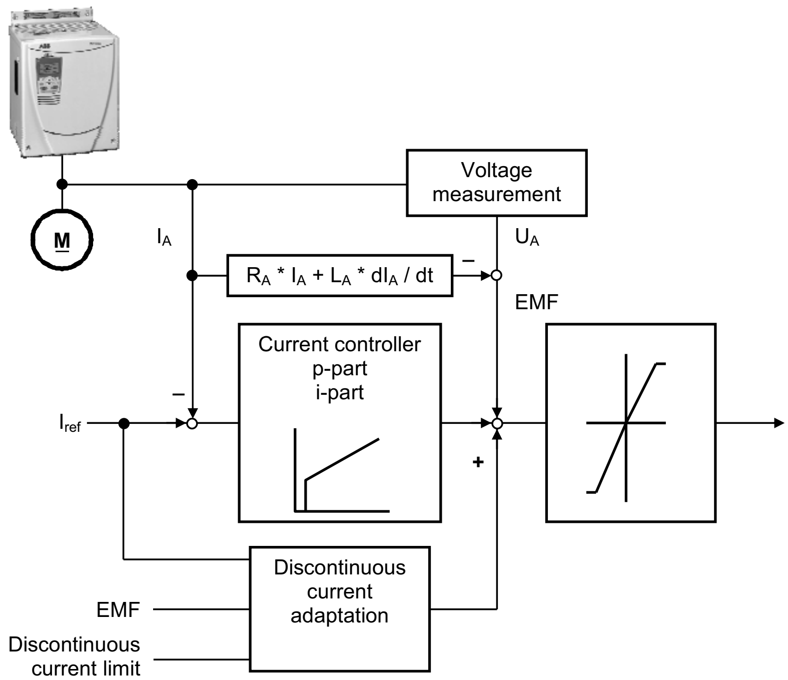
Control principle

To keep a PI-controller as fast as possible idealistically the integral part should stay at zero. The worst case is that the integral part is running into the limits and thus needs a long time to recover. To prevent this and to achieve an integral part as small as possible two feed forwards are used for the current controller:

1. During discontinuous current the signal from the current controller is boosted by means of the discontinuous current adaptation, depending on discontinuous current limit, current reference and EMF. The discontinuous current limit has to be determent during the commissioning.
2. Additionally the EMF itself is used as feed forward. Unfortunately it is not possible to measure the EMF directly. It has to be calculated by means of following formula:

$$EMF = U_A - (R_A * I_A + L_A * \frac{dI_A}{dt})$$

The values for the resistance (R_A) and the inductance (L_A) of the motor have to be determent during the commissioning.



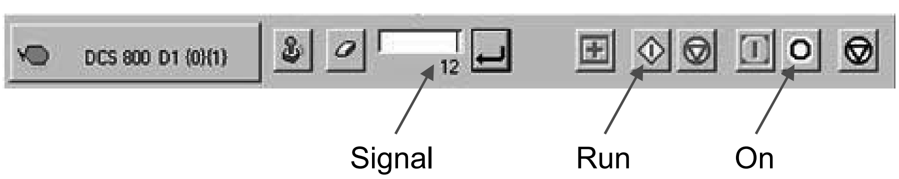
Control principle armature current controller

Manual tuning

Thus the manual tuning of the armature current controller has to be splitted into three parts:

1. determine resistance and inductance of the motor,
2. determine discontinuous current limit of the motor,
3. manual tuning of the armature current controller (p- and i-part)

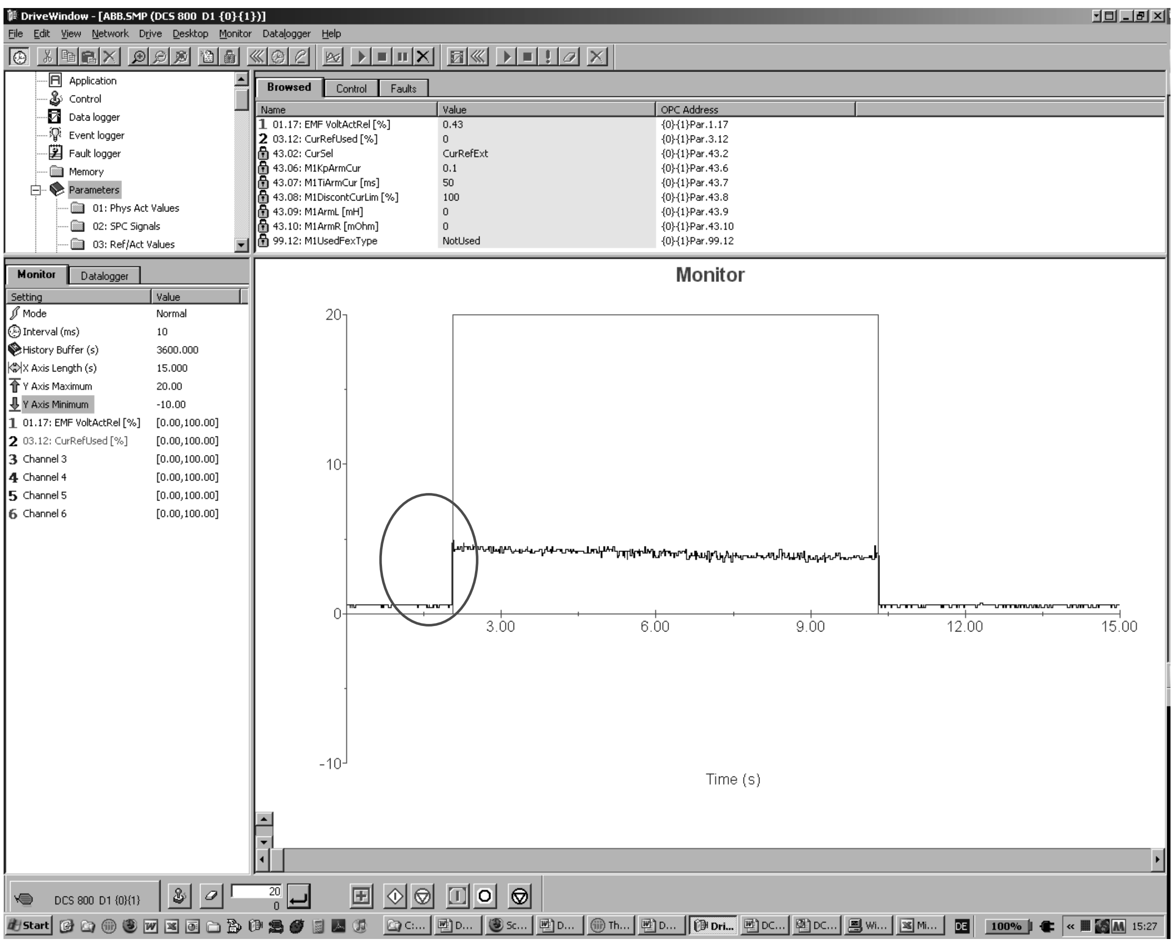
DriveWindow information:



DriveWindow information

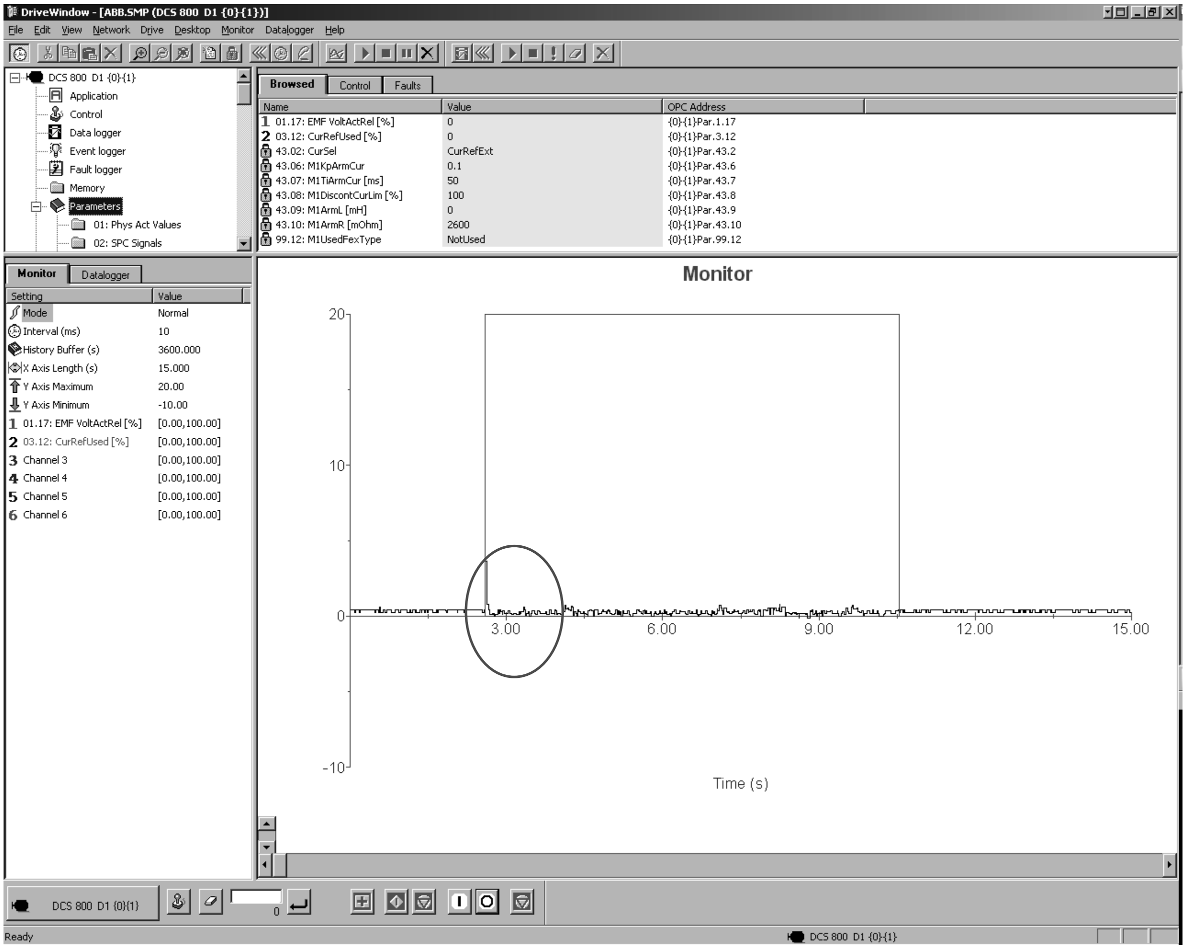
Part 1, determine resistance and inductance of the motor:

- connect DriveWindow to the drive and choose local mode,
- monitor *EMF VoltActRel (1.17)* and *CurRefUsed (3.12)*,
- set *CurSel (43.02)* = **CurRefExt**,
- set *M1KpArmCur (43.06)*, *M1TiArmCur (46.07)*, *M1DiscontCurLim (46.08)*, *M1ArmL (43.09)* and *M1ArmR (46.10)* to default,
- set *M1UsedFexType (99.12)* = **NotUsed**,
- give **On** and **Run** via DriveWindow,
- use DriveWindow to step the armature current controller and watch the EMF,
- make sure the motor is not turning (**Attention**: let the drive run only for a short time),



Before tuning of M1ArmL (43.09) and M1ArmR (46.10)

- tune *M1ArmR (46.10)* until the EMF is as close as possible to zero and dose not change it’s value during the current step,



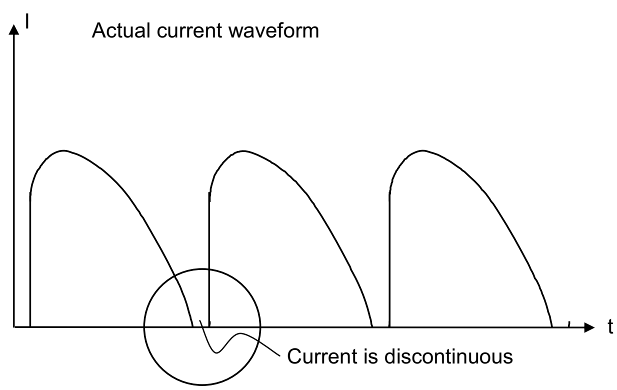
After tuning of M1ArmR (46.10)

– It is not possible to tune *M1ArmL (43.09)* manually.
  Thus set *M1ArmL (43.09)* = **0**!

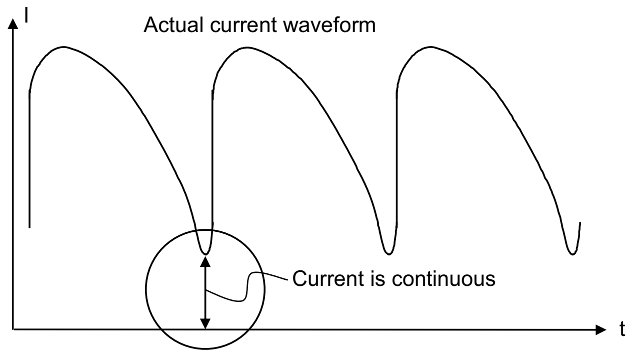
- remove **On** and **Run** via DriveWindow,
- set *CurSel (43.02)* and *M1UsedFexType (99.12)* back to their original settings

Part 2, determine discontinuous current limit of the motor:

- connect an oscilloscope to the fixed AO I-act (X4:9 / 10 on the SDCS-CON-4 or X4:5 / 6 on the SDCS-IOB-3),
- connect DriveWindow to the drive and choose local mode,
- set *CurSel (43.02)* = **CurRefExt**,
- set *M1DiscontCurLim (46.08)* to default,
- set *M1UsedFexType (99.12)* = **NotUsed**,
- give **On** and **Run** via DriveWindow,
- use DriveWindow to increase the armature current reference,
- make sure the motor is not turning (**Attention**: let the drive run only for a short time),
- watch the current bubbles and increase the current reference until the current is continuous,



Discontinuous current

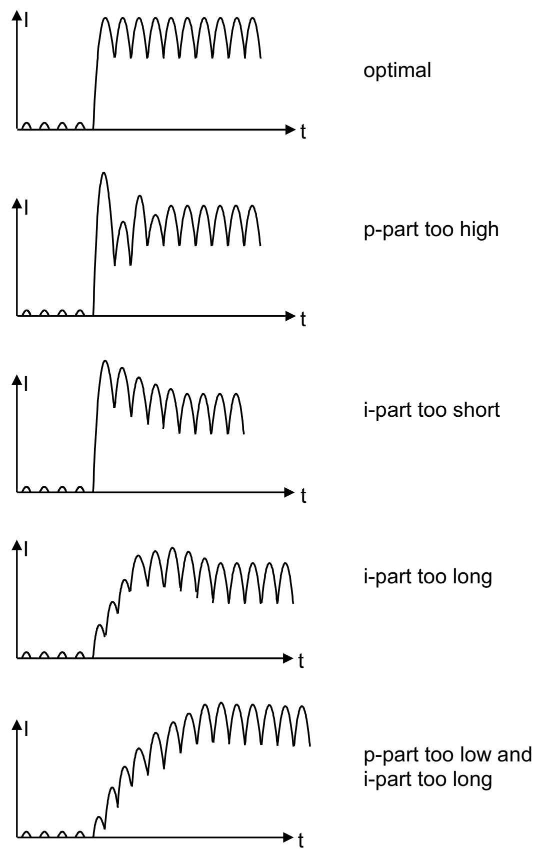


Continuous current

- remove **On** and **Run** via DriveWindow,
- set *CurSel (43.02)* and *M1UsedFexType (99.12)* back to their original settings,
- copy the current reference used in DriveWindow and paste it into *M1DiscontCurLim (46.08)*

Part 3, manual tuning of the armature current controller:

- connect an oscilloscope to the fixed AO I-act (X4:9 / 10 on the SDCS-CON-4 or X4:5 / 6 on the SDCS-IOB-3),
- connect DriveWindow to the drive and choose local mode,
- set *CurSel (43.02)* = **CurRefExt**,
- set *M1UsedFexType (99.12)* = **NotUsed**,
- give **On** and **Run** via DriveWindow,
- use DriveWindow to step the armature current controller,
- make sure the motor is not turning (**Attention**: let the drive run only for a short time),
- tune the armature current controller by means of *M1KpArmCur (43.06)* and *M1TiArmCur (46.07)*,



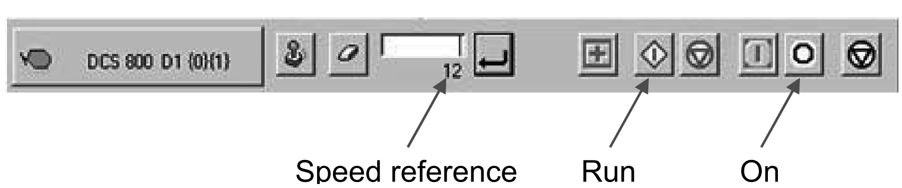
Armature current controller step responses

- remove **On** and **Run** via DriveWindow,
- set *CurSel (43.02)* and *M1UsedFexType (99.12)* back to their original settings

Analog tacho

In case an analog tacho is used for speed feedback it has to be tuned.

DriveWindow information:

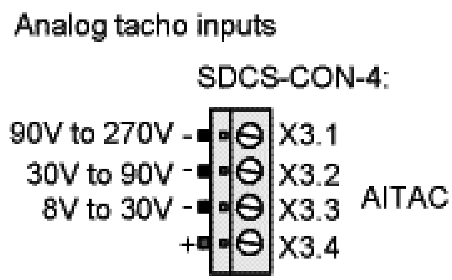


DriveWindow information

Manual tuning

Manual tuning of the analog tacho:

- set speed and analog tacho parameters,
  - o *M1SpeedMin (20.01)*,
  - o *M1SpeedMax (20.02)*,
  - o *M1OvrSpeed (30.16)*,
  - o *M1BaseSpeed (99.04)* and
  - o tacho voltage at 1000 rpm with *M1TachoVolt1000 (50.13)*,
- the maximum tacho speed is calculated automatically and shown in *M1TachoMaxSpeed (88.25)*,
- the needed tacho connection is calculated automatically and shown in *TachoTerminal (4.25)*,



Analog tacho connections

- check the tacho connections and change them accordingly,
- set *M1TachoTune (88.27)* = 1.000 (default),
- make sure that the drive is in **EMF control** - *M1SpeedFbSel (50.03)* = **EMF**,
- give **On** and **Run** via DriveWindow,
- use DriveWindow to set a constant speed reference,
- measure speed actual at the motor shaft using a hand held tacho,
- rescale *M1TachoTune (88.27)* in small steps, e.g. +/- 0.005 until the speed actual measured at the shaft and the speed actual measured with the analog tacho match, see *SpeedActTach (1.05)*,
- remove **On** and **Run** via DriveWindow

Speed controller

Basics

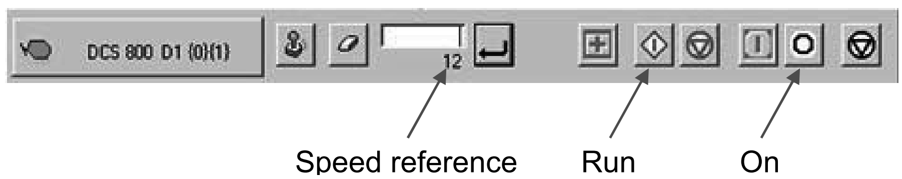
When tuning the drive, change one parameter at a time, then monitor the effect on the step response and possible oscillations. The effect of each parameter change must be checked over a wide speed range and not just at one point. The set speed controller values mainly depend on:

- the relationship between the motor power and the attached masses,
- backlashes and natural frequencies of the attached mechanics (filtering)

The step response tests must be carried out at different speeds, from minimum up to maximum speed, at several different points. The whole speed range must also be tested carefully, e.g. at 25 % - 30 % of maximum speed (step has to be in base

speed range) and 80 % of maximum speed (step has to be in field weakening area) in order to find any oscillation points.
A suitable speed step is about 2 % of maximum speed. A too large step reference or incorrect values of the speed controller might force the drives into torque / current limits, damage the mechanical parts (e.g. gear boxes) or cause tripping of the drive.

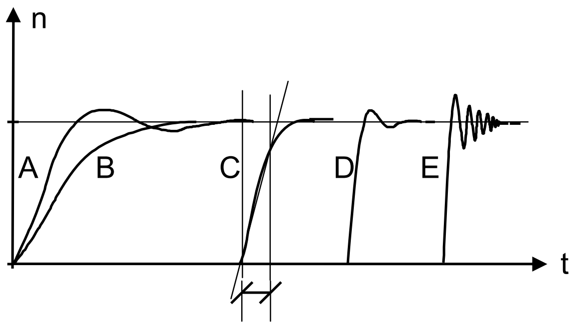
DriveWindow information:



DriveWindow information

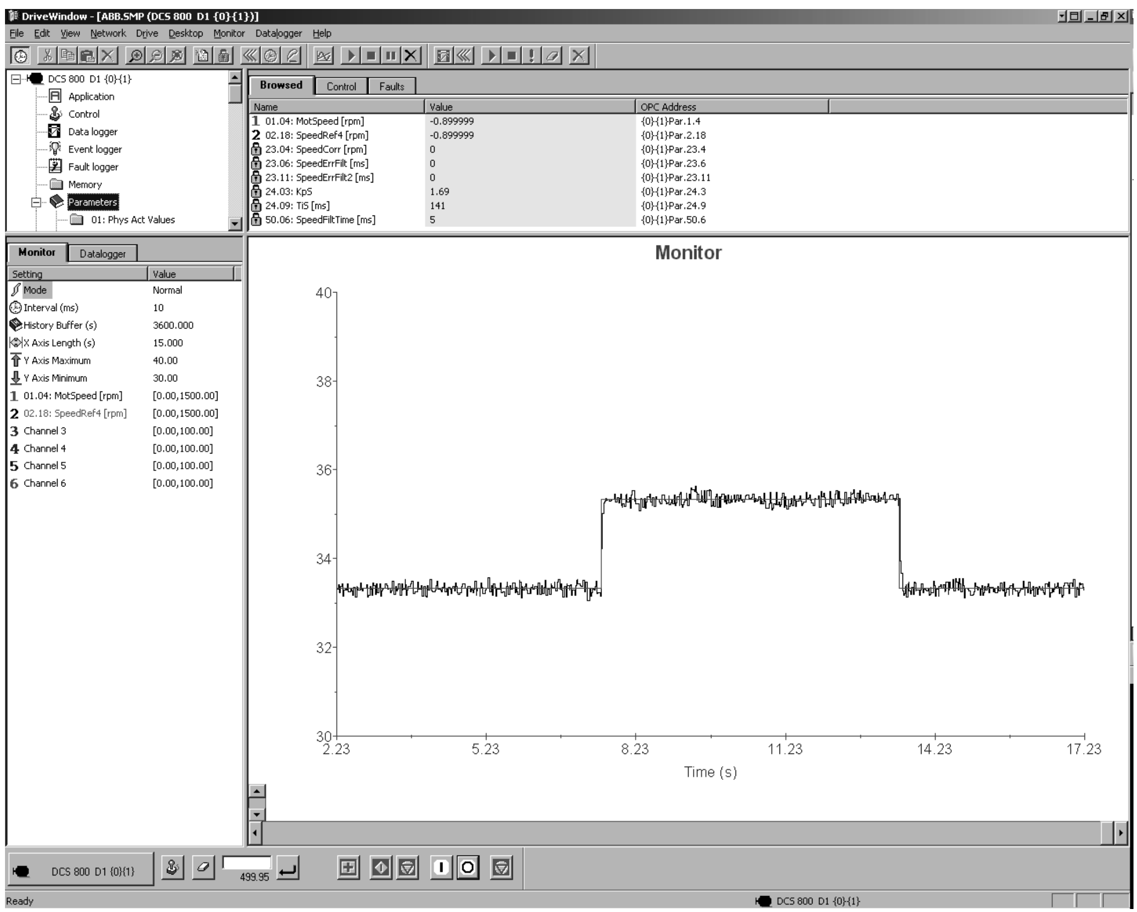
Manual tuning

Manual tuning of the speed controller:
- connect DriveWindow to the drive and choose local mode,
- monitor *MotSpeed (1.04)* and *SpeedRef4 (2.18)*,
- give **On** and **Run** via DriveWindow,
- use DriveWindow to set a constant speed reference,
- use *SpeedCorr (23.04)* to step the speed controller,
- tune the speed controller by means of *KpS (24.03)* and *TiS (24.09)*,
  - steps size: 2 % of maximum speed (do not hit any limits during the step and the step response, e.g. torque or current limits),
  - disable the i-part by setting *TiS (24.09)* = 0 ms,
  - increase *KpS (24.03)* until the step response shows an overshoot,
  - decrease *KpS (24.03)* about 30 %,
  - adjust *TiS (24.09)* in such a way, that there is no overshoot or only a slight overshoot, depending on the application (the function of the i-part is to reduce as quickly as possible the difference between speed reference and speed actual),
  - step response time: 100 ms (count only from 10 % to 90 %) in cold mills and 60 ms in rod and bar mills,
  - where to step: 25 % - 30 % of maximum speed (step has to be in base speed range) and 80 % of maximum speed (step has to be in field weakening area),
  - filter time •n: e.g. 5 ms - 10 ms [see *SpeedErrFilt (23.06)* and *SpeedErrFilt2 (23.11)*] or
  - filter time speed actual: e.g. 5 ms - 10 ms [see *SpeedFiltTime (50.06)*],



A: undercompensated, p-part too small and i-part too short
B: undercompensated, p-part too small
C: normal
D: normal, when a low impact speed drop is required
E: overcompensated, p-part too large and i-part too short

Speed controller step responses



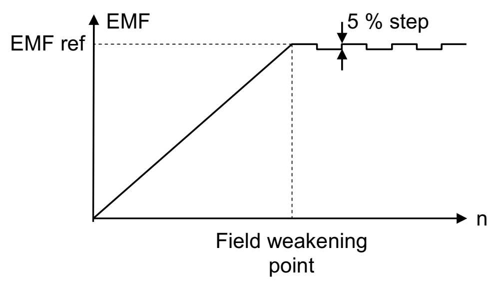
DriveWindow manual tuning speed controller

- set *SpeedCorr (23.04)* = 0 %,
- remove **On** and **Run** via DriveWindow

EMF controller

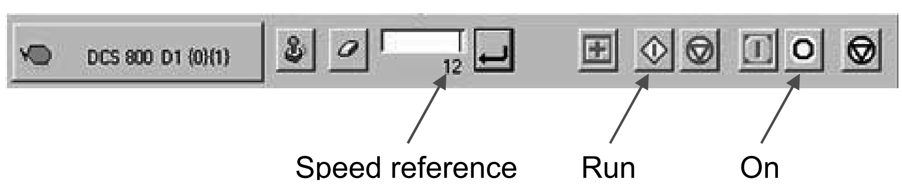
Basics

In case the motor needs to be used in the field weakening area the EMF controller has to be tuned. The EMF controller needs to have a quick response. Usually 2 to 3 times slower than the field current controller.
The tuning has to be done in the field weakening area, because the EMF controller is blocked in the base speed range.



EMF reference for manual tuning EMF controller

DriveWindow information:

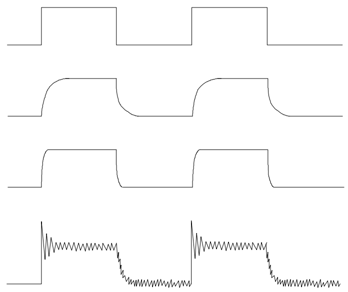


DriveWindow information

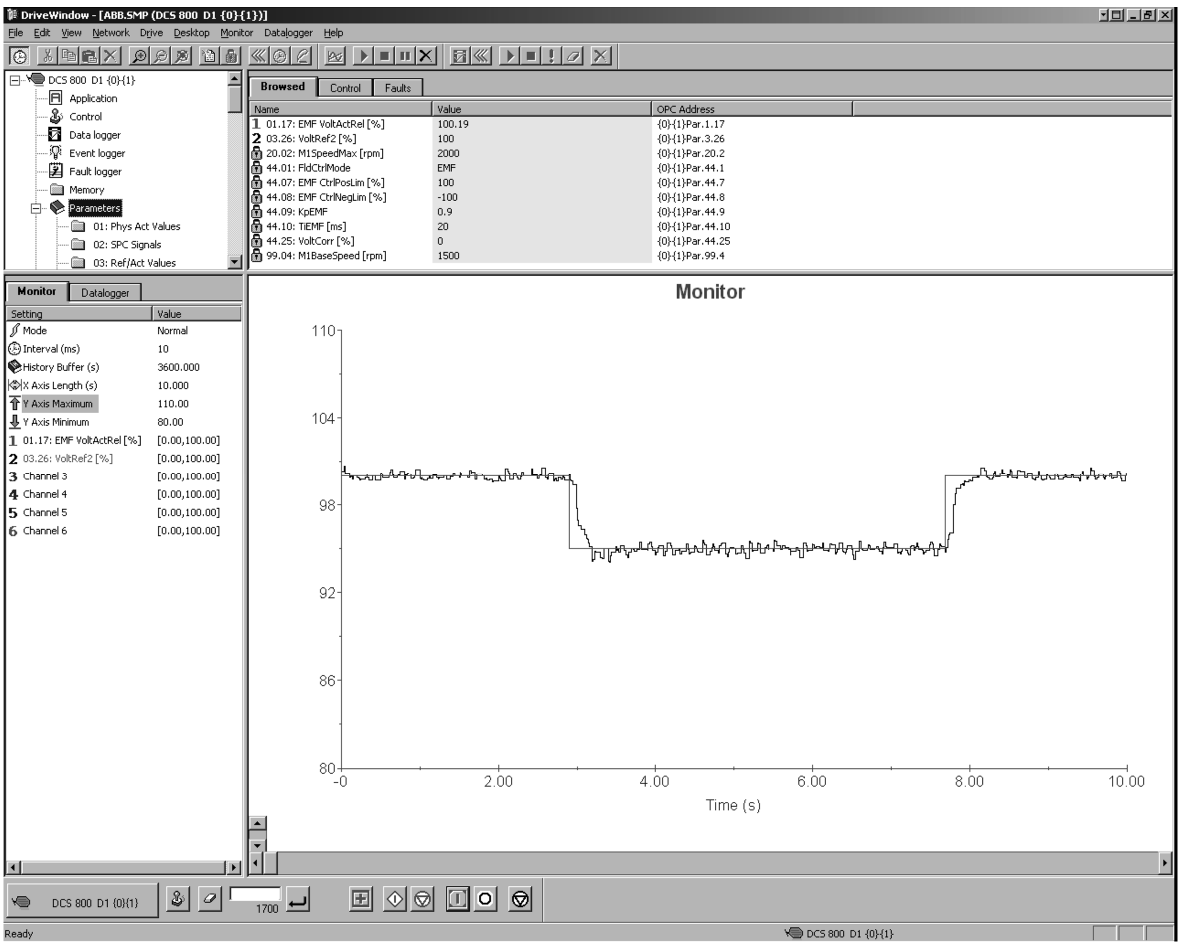
Manual tuning

Manual tuning of the EMF controller:

- connect DriveWindow to the drive and choose local mode,
- monitor *EMF VoltActRel (1.17)* and *VoltRef2 (3.26)*,
- set *FldCtrlMode (44.01)* = **EMF**,
- set *EMF CtrlPosLim (44.07)* = 100 %,
- set *EMF CtrlNegLim (44.08)* = -100 %,
- give **On** and **Run** via DriveWindow,
- use DriveWindow to set a constant speed reference in the field weakening area,
- use *VoltCorr (44.25)* to step the EMF controller,
- tune the EMF controller by means of *KpEMF (44.09)* and *TiEMF (44.10)*,
    - steps size: 2 % - 5 % (do not hit any limits during the step and the step response),
    - step response time: 2 - 3 times slower than the field current controller,
    - where to step: in the field weakening area,



EMF controller step responses



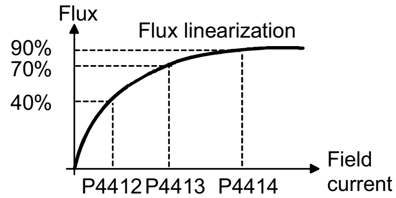
DriveWindow manual tuning EMF controller

- set *VoltCorr (44.25)* = 0 %,
- remove **On** and **Run** via DriveWindow.
- set *FldCtrlMode (44.01)*, *EMF CtrlPosLim (44.07)* and *EMF CtrlNegLim (44.08)* back to their original settings

Flux linearization

Basics

In case the motor needs to be used in the field weakening area the flux linearization has to be set. The flux linearization is needed because of the non-linear relation of flux and field current due to saturation effects of the field winding.



Flux of DC-motor versus field current

The magnetization of the motor starts to saturate at a certain field current and thus the flux does not increase linearly. For this reason the field current cannot be directly used to calculate the flux inside the motor.
In base speed area EMF and speed are directly proportional because the flux is kept constant:

$$n = \frac{k * EMF}{\Phi}$$

k = constant
Φ = Flux

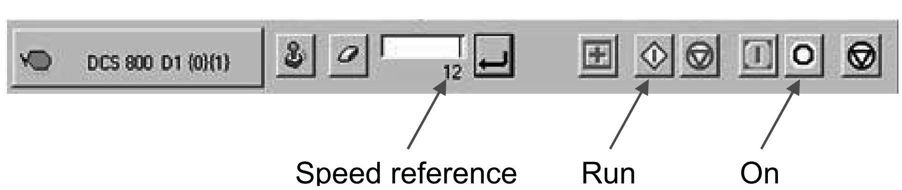
Example:
If the nominal armature voltage is 440 V and the motor is running at half speed with full flux, then the armature voltage is about 220 V. Now the flux is reduced to 50 % at constant speed, then the armature voltage drops to about 110 V.
Since the EMF is directly proportional to the flux it is possible to define a relationship between the field current and the flux by means of measuring the armature voltage without load (= EMF).

Thus the main idea of the flux linearization is to find field currents which produces desired EMF-voltage at a certain speed. The flux linearization is done by means of a function block defined by 3 values:

- field current at 40 % flux, *FldCurFlux40 (44.12)*,
- field current at 70 % flux, *FldCurFlux70 (44.13)*,
- field current at 90 % flux, *FldCurFlux90 (44.14)*

The intermediate values are interpolated. During commissioning all 3 parameters must be set, if the flux linearization is needed.

DriveWindow information:



DriveWindow information

Manual tuning

Manual tuning of the flux linearization:

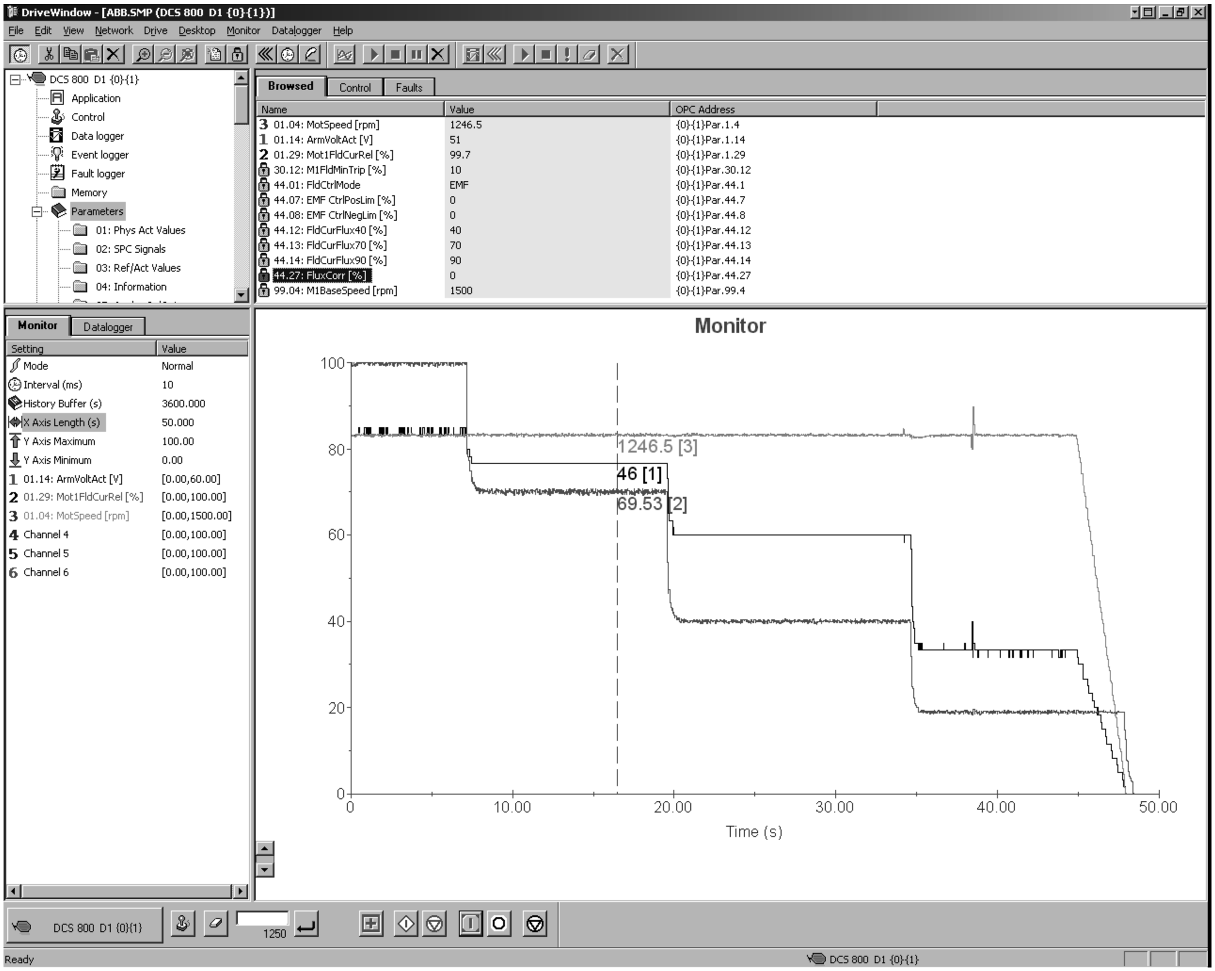
- connect DriveWindow to the drive and choose local mode,
- make sure the speed feedback device is either encoder or analog tacho - *M1SpeedFbSel (50.03)* = **Encoder** or **Tacho** - and **not** EMF!
- monitor *MotSpeed (1.04)*, *ArmVoltAct (1.14)* and *Mot1FldCurRel (1.29)*,
- set *M1FldMinTrip (30.12)* = 10 %,
- set *FldCtrlMode (44.01)* = **EMF**,
- set *EMF CtrlPosLim (44.07)* = 0 %,
- set *EMF CtrlNegLim (44.08)* = 0 %,
- set *FldCurFlux40 (44.12)*, *FldCurFlux70 (44.13)* and *FldCurFlux90 (44.14)* to default,
- give **On** and **Run** via DriveWindow,

- use DriveWindow to run the motor at e.g. half base speed,
- make sure, that the motor is running without load,
- read *ArmVoltAct (1.14)*, e.g. the measured value is 220 V,

- reduce the flux with *FluxCorr (44.27)* until *ArmVoltAct (1.14)* reaches 90 % of the 1st measurement,
- read the value of *Mot1FldCurRel (1.29)*, keep it in mind and write it into *FldCurFlux90 (44.14)* after this procedure is finished,

- reduce the flux with *FluxCorr (44.27)* until *ArmVoltAct (1.14)* reaches 70 % of the 1st measurement,
- read the value of *Mot1FldCurRel (1.29)*, keep it in mind and write it into *FldCurFlux70 (44.13)* after this procedure is finished,

- reduce the flux with *FluxCorr (44.27)* until *ArmVoltAct (1.14)* reaches 40 % of the 1st measurement,
- read the value of *Mot1FldCurRel (1.29)*, keep it in mind and write it into *FldCurFlux40 (44.12)* after this procedure is finished,



DriveWindow manual tuning flux linearization

- set *FluxCorr (44.27)* = 0 %,
- remove **On** and **Run** via DriveWindow,
- set *FldCurFlux90 (44.14)*, *FldCurFlux70 (44.13)* and *FldCurFlux40 (44.12)* to the determined values,
- set *M1FldMinTrip (30.12)*, *FldCtrlMode (44.01)*, *EMF CtrlPosLim (44.07)* and *EMF CtrlNegLim (44.08)* back to their original settings

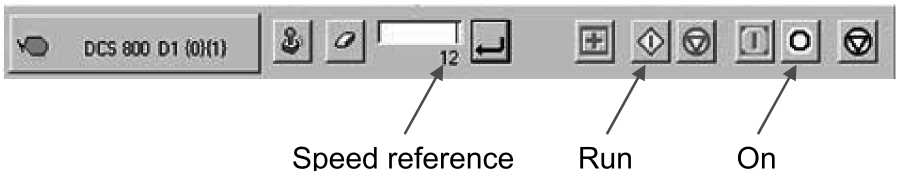
Thyristor diagnosis

Basics

Thyristor diagnosis basically provides two possibilities:

1. check all thyristors of the drive for proper function or
2. check individual firing pulses

DriveWindow information:

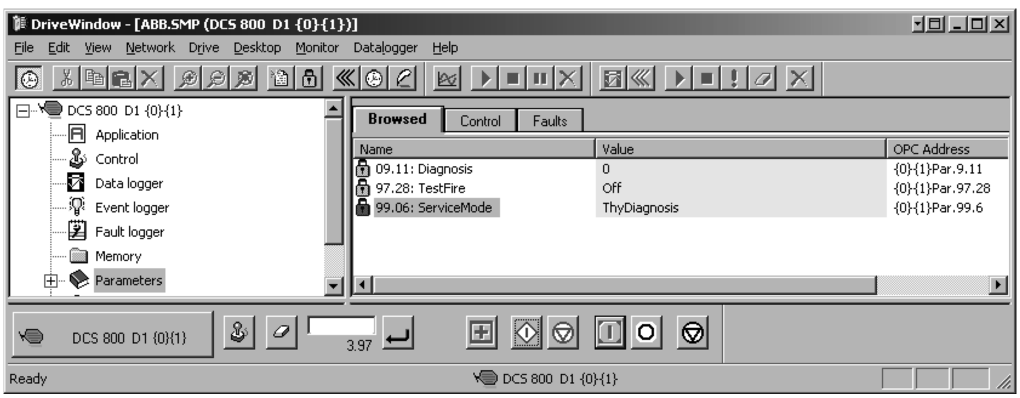


DriveWindow information

Check all thyristors

Thyristor diagnosis for all thyristors:

- connect DriveWindow to the drive and choose local mode,
- set *ServiceMode (99.06)* = **ThyDiagnosis**,
- set *TestFire (97.28)* = **Off**,
- give **On** and **Run** via DriveWindow,



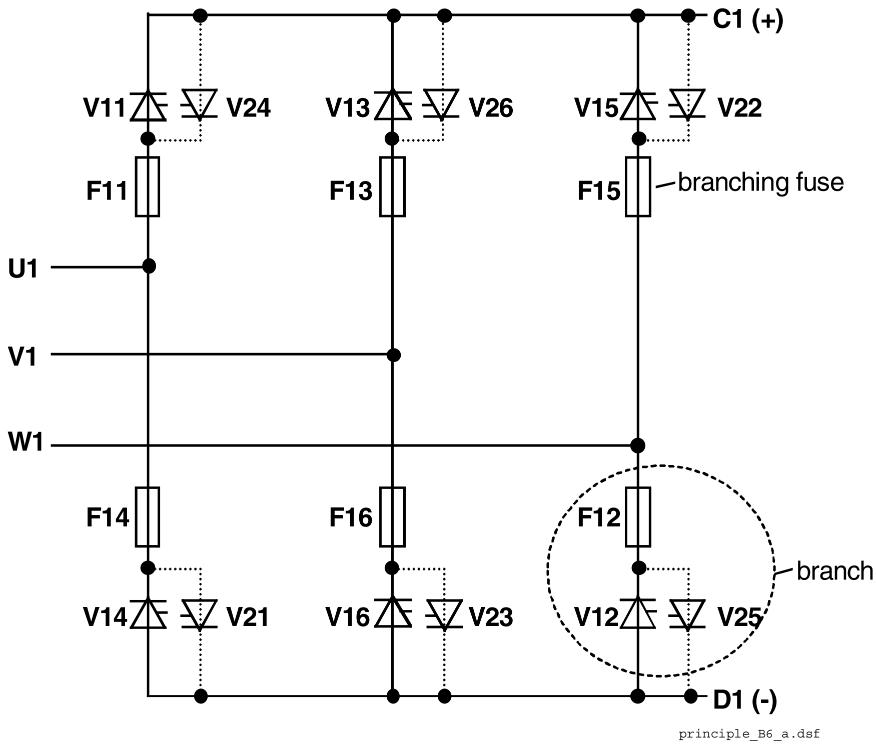
The main contactor is closed and the thyristor diagnosis is started. After the thyristor diagnosis is finished:

- the result is written into *Diagnosis (9.11),*
- the *ServiceMode (99.06)* is automatically set back to **NormalMode** and
- the drive is automatically switched off.

Check individual firing pulses

Check individual firing pulses:

- make sure, that the main contactor cannot close (e.g. disconnect the digital output controlling the main contactor) or that the mains voltage is off (e.g. high voltage breaker is open),
- connect a current clamp to one of the firing pulse cables,
- connect DriveWindow to the drive and choose local mode,
- set *ServiceMode (99.06)* = **ThyDiagnosis**,
- set *TestFire (97.28)* = **V11, …, V26** depending individual firing pulse to be checked,



- give **On** and **Run** via DriveWindow, the main contactor should not pick up,
- make sure, that the mains voltage is zero,
- check the firing pulse with the current clamp,
- remove **On** and **Run** via DriveWindow,
- set *ServiceMode (99.06)* back to **NormalMode**,

TestFire (97.28) is automatically set back to **Off**.

Firmware description

Chapter overview

This chapter describes how to control the drive with **standard** firmware.

Identification of the firmware versions

The DCS800 is controlled by the SCDS-CON-4. The firmware version and type can be checked from:

- *FirmwareVer (4.01)* and
- *FirmwareType (4.02)*

The DDCS communication is handled by the SDCS-COM-8. The firmware revision can be checked from:

- *Com8SwVersion (4.11)*

The firmware revisions of the field exciters can be checked from:

- *Mot1FexSwVer (4.08)* and
- *Mot2FexSwVer (4.09)*

Start / stop sequences

General

The drive is controlled by control words [*MainCtrlWord (7.01)* or *UsedMCW (7.04)*]. The *MainStatWord (8.01)* provides the hand shake and interlocking for the overriding control.
The overriding control uses the *MainCtrlWord (7.01)* or hardware signals to command the drive. The actual status of the drive is displayed in the *MainStatWord (8.01)*.
The marks (e.g. ➊) describe the order of the commands according to Profibus standard. The overriding control can be:

- AC 800M via DDCS communication,
- serial communication (e.g. Profibus),
- hardware signals - see *CommandSel (10.01)* = **Local I/O**,
- master-follower communication,
- Adaptive Program or
- application program.

Switch on sequence

Bit	15 ... 11	RemoteCmd 10	Inching2 09	Inching1 08	Reset 07	RampInZero 06	RampHold 05	RampOutZero 04	Run 03	Off2N 02	Off2N 01	On 00	**Dec.**	**Hex.**
Reset		1	x	x	1	x	x	x	x	x	x	x	**1270**	**04F6**
Off (before On)		1	0	0	0	x	x	x	0	1	1	0	**1142**	**0476**
On (main cont. On)		1	0	0	0	x	x	x	0	1	1	1	**1143**	**0477**
Run (with reference)		1	0	0	0	1	1	1	1	1	1	1	**1151**	**047F**
E-Stop		1	x	x	x	1	1	1	1	0	1	1	**1147**	**047B**
Start inhibit		1	x	x	x	x	x	x	x	x	0	x	**1140**	**0474**

Examples for the MainCtrlWord (7.01)

Start the drive

The start sequence given below is only valid for *MainContCtrlMode (21.16)* = **On**.

Attention:
All signals have to be maintained. **On**- and **Run** [*MainCtrlWord (7.01)* bit 0 and 1] commands are only taken over with their rising edges.

Overriding Control *MainCtrlWord (7.01)*		Drive *MainStatWord (8.01)*
		When the drive is ready to close the main contactor **RdyOn** state is set
	❶	⇦ **RdyOn** = 1; (bit 0)
The overriding control commands **On**		
On = 1; (bit 0) ⇨	❷	
		The drive closes the main contactor, the field contactor and the contactors for converter and motor fans. After the mains voltage and all acknowledges are checked and the field current is established, the drive sets state **RdyRun**.
	❸	⇦ **RdyRun** = 1; (bit 1)
The overriding control commands **Run**		
Run = 1; (bit 3) ⇨	❹	
		The drive releases the ramp, all references, all controllers and sets state **RdyRef**
	❺	⇦ **RdyRef** = 1; (bit 2)
Now the drive follows the speed or torque references		

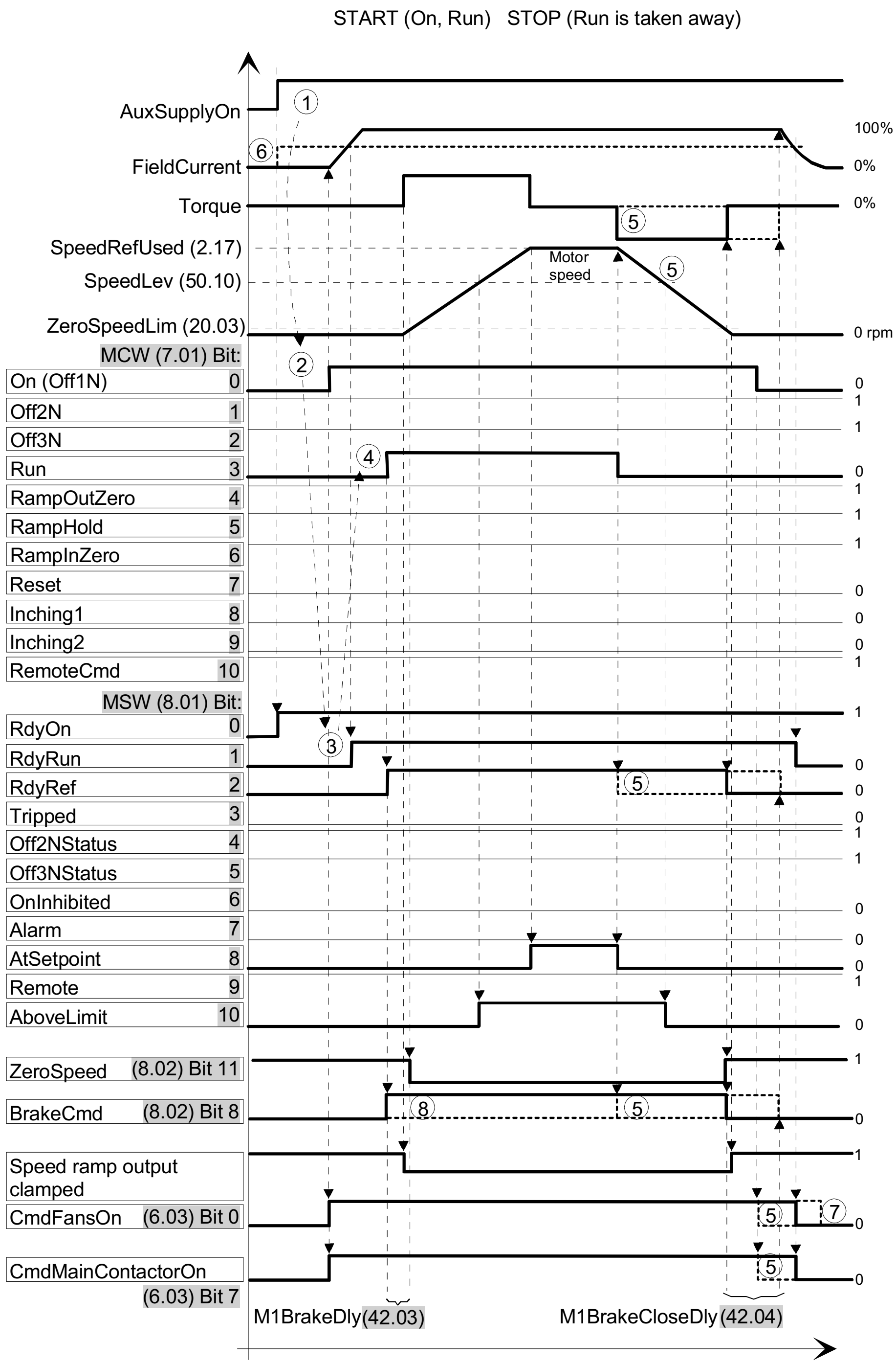
Note:
To give **On** and **Run** at the same time set *OnOff1 (10.15)* = *StartStop (10.16)*.

Stop the drive

The drive can be stopped in two ways, either by taking away the **On** command directly which opens all contactors as fast as possible after stopping the drive according to *Off1Mode (21.02)* or by means of the following sequence:

Overriding Control *MainCtrlWord (7.01)*		Drive *MainStatWord (8.01)*
The overriding control removes **Run**		
Run = 0; (bit 3) ⇨	❶	
		In speed control mode, the drive stops according to *StopMode (21.03)*. In torque control mode, the torque reference is reduced to zero according to *TorqRefA FTC (25.02)* respectively *TorqRampDown (25.06)*, depending on the used torque reference channel (A or B). When zero speed or zero torque is reached the state **RdyRef** is removed.
	❷	⇦ **RdyRef** = 0; (bit 2)
The overriding control can keep the **On** command if the drive has to be started up again The overriding control removes **On**		
On = 0; (bit 0) ⇨	❸	
		All contactors are opened - the fan contactors stay in according to *FanDly (21.14)* - and the state **RdyRun** is removed
	❹	⇦ **RdyRun** = 0; (bit 1)

Besides in *MainStatWord (8.01)*, the drive's state is shown in *DriveStat (8.08)*.



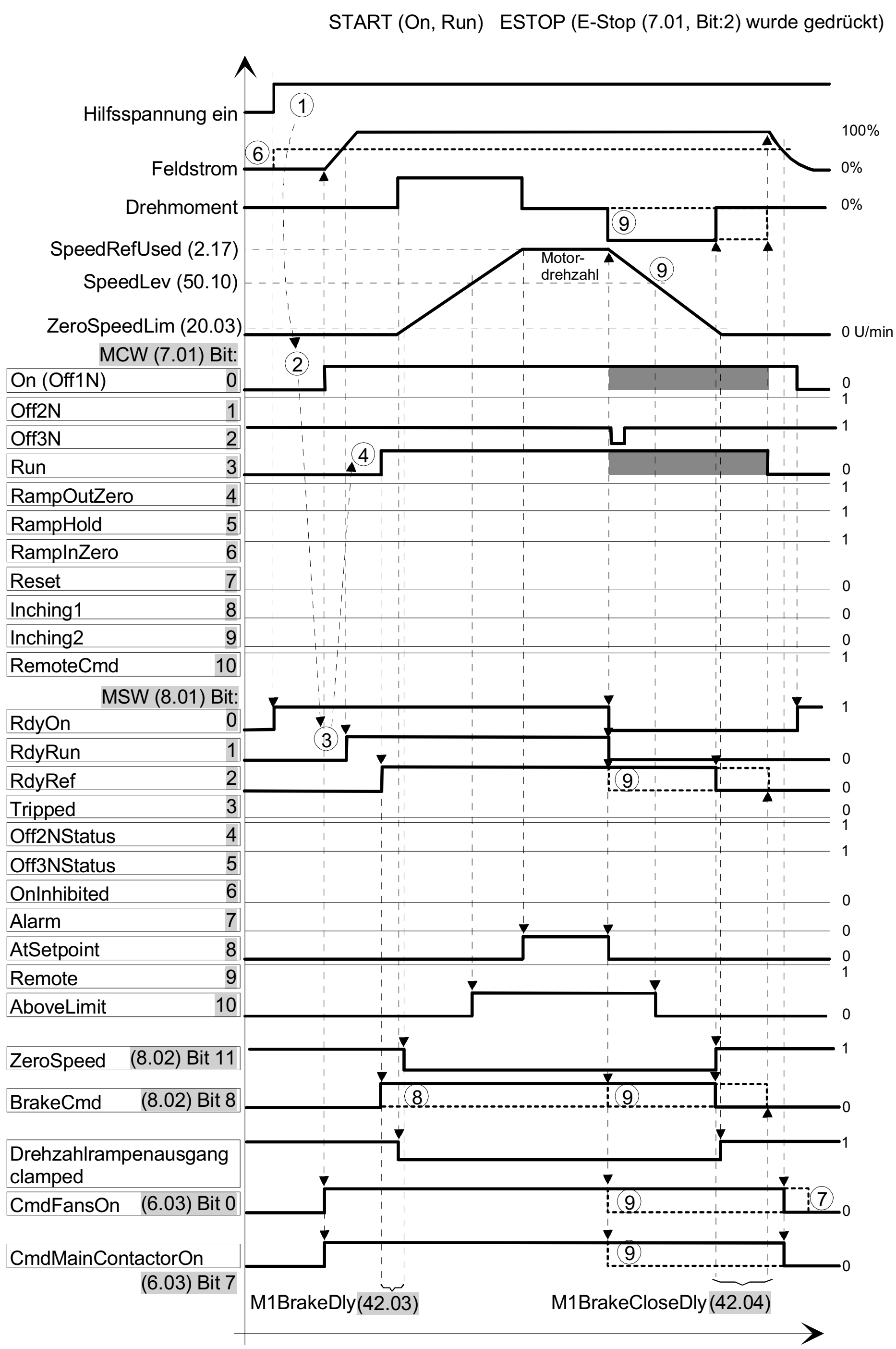
5 Behaviour depends on Off1Mode (21.02) and StopMode (21.03)

6 Behaviour depends on FldHeatSel (21.18) and M1FldMinTrip (30.12)

7 Behaviour depends on FanDly (21.14)

8 Behaviour depends on M1BrakeCtrl (42.01)

Start stop seq.dsf



5 Verhalten abhängig von Off1Mode (21.02) und StopMode (21.03)

6 Verhalten abhängig von FldHeatSel (21.18) und M1FldMinTrip (30.12)

7 Verhalten abhängig von FanDly (21.14)

8 Verhalten abhängig von BrakeEStopMode (42.09)

9 Verhalten abhängig von EStopMode (21.04)

Nicht relevant

Start stop seq_b.dsf

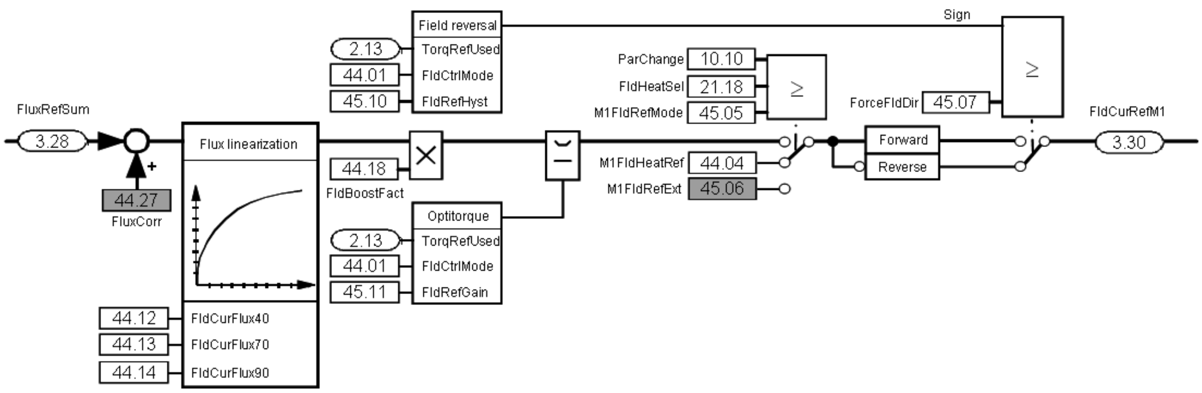
Field excitation

General

Depending on the application the DCS800 has the capability to use several different kinds of field exciters or combinations of them. The differences of the field exciters and their functions are explained here.

Field Reversal

Changing the field current direction is needed when the armature converter has only one bridge (2-quadrant). Field reversal is changing the direction of the field current. Thus the direction of the speed is changing and it is possible to regenerate energy back into the mains. For example to decelerate a large inertia.
To initiate the field reversal the sign of *TorqRefUsed (2.13)* is taken and defines the desired direction of the field current. Armature converters with two anti-parallel bridges (4-quadrant) do not require field reversal.



Overview field reversal and optitorque

Field control

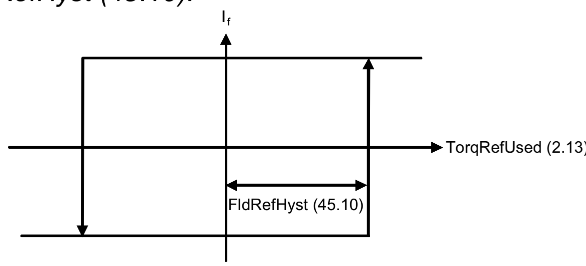
Field reversal is activated by means of *FldCtrlMode (44.01)*:

Mode	Functionality	Armature converter
Fix	constant field (no field weakening), EMF controller blocked, field reversal blocked, optitorque blocked, default	2-Q or 4-Q
EMF	field weakening active, EMF controller released, field reversal blocked, optitorque blocked	2-Q or 4-Q
Fix/Rev	constant field (no field weakening), EMF controller blocked, **field reversal active**, optitorque blocked	2-Q
EMF/Rev	field weakening active, EMF controller released, **field reversal active**, optitorque blocked	2-Q
Fix/Opti	constant field (no field weakening), EMF controller blocked, field reversal blocked, **optitorque active**	2-Q or 4-Q
EMF/Opti	field weakening active, EMF controller released, field reversal blocked, **optitorque active**	2-Q or 4-Q
Fix/Rev/Opti	constant field (no field weakening), EMF controller	2-Q

	blocked, **field reversal active, optitorque active**	
EMF/Rev/Opti	field weakening active, EMF controller released, **field reversal active, optitorque active**	2-Q

Field reference hysteresis

To prevent field reversal from continuous toggling due to a too small torque reference, a torque reference hysteresis is available. The hysteresis is symmetrical and is set by *FldRefHyst (45.10)*:



Field reference hysteresis

Force field current direction

With *ForceFldDir (45.07)* it is possible to force and clamp the field current direction. This gives the user the possibility to control the field current direction or change it in case of need. Thus unnecessary field current changes at low torque are prevented and it is also possible to release field reversal for certain occasions, e.g. jogging or E-stop.

Reversal time

The physical reversal time can be reduced by increasing the input voltage of the field exciter and using Optitorque.
Please note that the output voltage of the field exciter is limited by means of *M1PosLimCtrl (45.02)* or *M2PosLimCtrl (45.16).* This can also increase the physical reversal time.

Bumpless transition

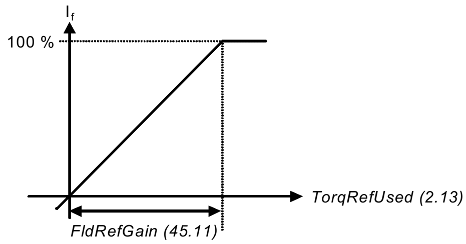
The output of the speed ramp is updated by means of the actual speed to ensure a bumpless transition, if *RevDly (43.14)* is greater than 25 ms and *RevMode (43.16)* = **Soft**.

Optitorque

Due to high inductances of motors, the field reversal takes a relatively long time. In certain cases this time can be reduced by means of optitorque - see *FldCtrlMode (44.01)*. In case the process requires only a small torque during field reversal, the field current is decreased and the armature current is increased prior to the field current change. This speeds up the field reversal. The rate of the field current reduction depends on the process. E.g. if the speed direction is changed rather slowly, the required torque may also be quite small. This allows the reduction of the field current. Thus by means of optitorque it is possible to shorten the field reversal time.

Field current reference gain

In optitorque mode the field current will be reduced proportionally to *TorqRefUsed (2.13)*. The relation between *TorqRefUsed (2.13)* and field current is defined by *FldRefGain (45.11)*:



Field current reference gain

For example with *FldRefGain (45.11)* = 20 %, 100 % field current is generated at *TorqRefUsed (2.13)* = 20 %.

Field current monitoring

Field minimum trip

During normal operation the field current is compared with *M1FldMinTrip (30.12)*. The drive trips with **F541 M1FexLowCur** [*FaultWord3 (9.03)* bit 8] if the field current drops below this limit and is still undershot when *FldMinTripDly (45.18)* is elapsed.
During field reversal the situation is different. *M1FldMinTrip (30.12)* is disabled for *FldCtrlMode (44.01)* = **Fix/Opti**, **EMF/Opti**, **Fix/Rev/Opti** or **EMF/Rev/Opti**. In this case the trip level is automatically set to 50 % of *FldCurRefM1 (3.30).* The drive trips with **F541 M1FexLowCur** [*FaultWord3 (9.03)* bit 8] if 50 % of *FldCurRefM1 (3.30)* is still undershot when *FldMinTripDly (45.18)* is elapsed.

Flux reversal

If actual flux and armature voltage of the motor cannot follow the field current during field reversal it is necessary to delay the active field direction. *FluxRevMonDly (45.08)* is the maximum allowed time within *Mot1FldCurRel (1.29)* and the internal motor flux doesn't correspond to each other during field reversal. During this time **F522 SpeedFb** [*FaultWord2 (9.02)* bit 5] is disabled.

Field reversal hysteresis

The sign of *Mot1FldCurRel (1.29)* is used to generate the field reversal acknowledge. To avoid signal noise problems a small hysteresis - defined by means of *FldRevHyst (45.09)* - is needed.

Field reversal active

While the field reversal is in progress - see *CurCtrlStat2 (6.04)*, bit 11,
- the current controller is blocked,
- the I-part of the speed controller frozen,
- the output of the speed ramp is updated by means of the actual speed, if *RevDly (43.14)* is greater than 25 ms and *RevMode (43.16)* = **Soft**

Field Heating

Overview

Field heating (also referred to as “field warming and field economy”) is used for a couple of reasons.
Previous generations of DC-drives used voltage-controlled field supplies, meaning that the only thing the field supply could directly control was the field voltage. For DC-motors to maintain optimal torque it is important to maintain the field current. Ohm's law (U = R*I) tells us that voltage equals resistance multiplied by current. So as long as resistance remains constant, current is proportional to voltage. But field resistance increases with temperature. Therefore, a cold motor would have a

higher field current than a warm motor, even though voltage remained unchanged. To keep the resistance and thus the current constant, the field was left on to keep it warm. Then the voltage-controlled field supply works just fine.
The new generation of drives, including all field supplies used with the DCS800, are current controlled. Thus the field supply directly controls field current. This means that field heating may no longer be necessary when the DCS800 is employed.
Another reason field heating is used is to keep moisture out of the motor.
Following parameters are used to turn on and control field heating:

- *FldHeatSel (21.18)*,
- *M1FldHeatRef (44.04)*

Modes of operation

There are basically two modes of operation. In both modes, the field current will be at a reduced level, determined by *M1FldHeatRef (44.04)*.
FldHeatSel (21.18) = **On**:

- Field heating is on, as long as **On** = 0 [*UsedMCW (7.04)* bit 0], **Off2N** = 1 [*UsedMCW (7.04)* bit 1] and **Off3N** = 1 [*UsedMCW (7.04)* bit 2].
  In general, field heating will be on as long as the OnOff input is not set and no Coast Stop or E-stop is pending.

Condition	**On** [*UsedMCW (7.04)* bit 0]	**Off2N** [*UsedMCW (7.04)* bit 1]*	**Result**
Power up	0	1	reduced field current**
Start drive	1	1	normal field current
Normal stop	1 → 0	1	normal field current, then reduced** after stop
Coast Stop while running	1	1 → 0	field is turned off as motor coasts to stop and cannot turned back on again as long as Coast Stop is pending

*see *Off2 (10.08)*
**the field current will be at the level set by means of *M1FldHeatRef (44.04)* while motor is stopped

FldHeatSel (21.18) = **OnRun**:

- Field heating is on as long as **On** = 1, **Run** = 0 [*UsedMCW (7.04)* bit 3], **Off2N** = 1 and **Off3N** = 1.
  In general, field heating will be on as long as the OnOff input is set, the Start/Stop input is not set and no Coast Stop or E-stop is pending.

On [*UsedMCW (7.04)* bit 0]	**Run** [*UsedMCW (7.04)* bit 3]	**Off2N** [*UsedMCW (7.04)* bit 1]*	**Result**
0	x	x	field is turned off
1	0	1	reduced field current**
1	1	1	normal field current
1	1 → 0	1	normal field current, then reduced** after stop

1	x	1 → 0	field is turned off as motor coasts to stop and cannot turned back on again as long as Coast Stop is pending

*see *Off2 (10.08)*
**the field current will be at the level set by means of *M1FldHeatRef (44.04)* while motor is stopped

E-stop

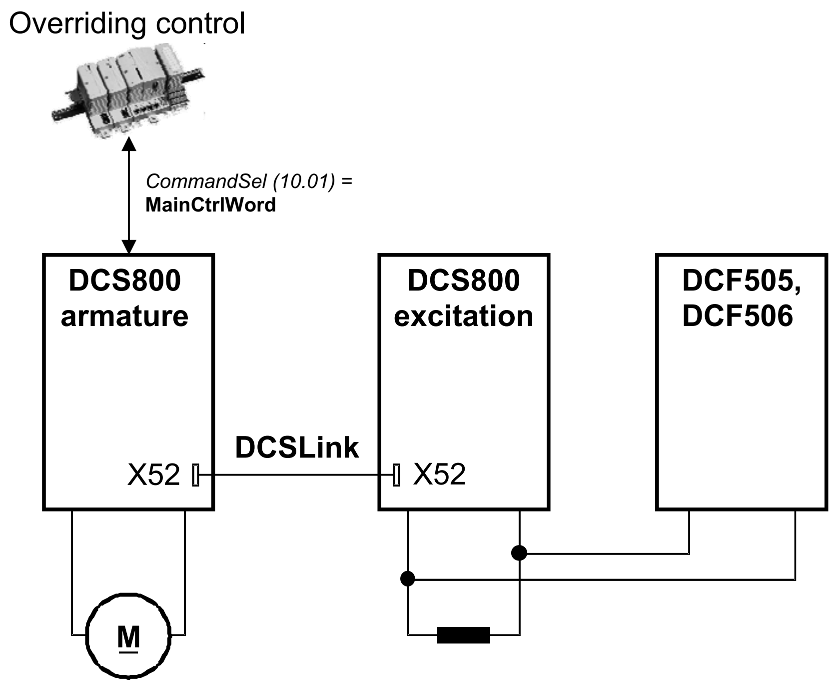
In both modes of operation, if the E-stop - see *E Stop (10.09)* - is pending the field will be turned off. It cannot be turned back on again as long as the E-stop is pending. If the E-stop is cleared while in motion, the motor will be stopped according to *E StopMode (21.04)* and then field and drive will be turned off.

Field exciter mode

General

The standard DCS800 module can be operated as large field exciter by simply setting parameters. It is either controlled by a DCS800 armature converter or can be configured as stand alone field exciter.
The field exciter mode uses the standard armature current controller as field current controller. Thus the current of the converter [*ConvCurAct (1.16)*] equals the field current of the motor. For these configurations an overvoltage protection (DCF505 or DCF506) is mandatory.

Large field exciter controlled by a DCS800 armature converter

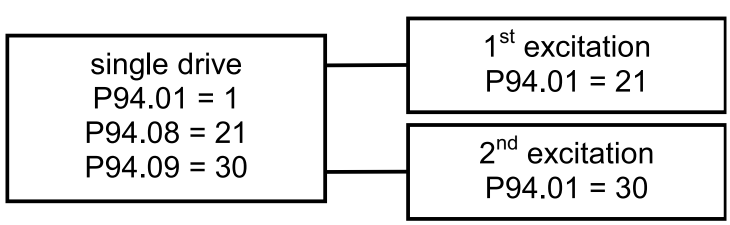


Communication in field exciter mode

Large field exciters are fully controlled via the DCSLink:

DCSLinkNodeID (94.01) = 1, default
M1FexNode (94.08) = 21, default
M2FexNode (94.09) = 30, default

Single drive with one or two large field exciters:



In the large field exciters set *OperModeSel (43.01)* = **FieldConv** and *CommandSel (10.01)* = **FexLink** as source for the control word (**OnOff1**, **StartStop** and **Reset**). The reference is selected by *CurSel (43.02)* = **FexCurRef**. In the armature converter the field current is set by means of *M1NominalFldCur (99.11)* and in the large field exciter the current is set by means of *M1NomCur (99.03).*
To close the field contactor use *CurCtrlStat1 (6.03)* bit 7.

Parameters to be set in the DCS800 armature converter:

Parameter	Armature converter	Comments
M1FldMinTrip (30.12)	xxx %	sets level for **F541 M1FexLowCur**
FldCtrlMode (44.01)	1 = **EMF**	EMF controller released, field weakening active - depending on the application
FldMinTripDly (45.18)	2000 ms (def.)	delays **F541 M1FexLowCur**
DCSLinkNodeID (94.01)	1	
FexTimeOut (94.07)	100 ms (def.)	causes **F516 M1FexCom**
M1FexNode (94.08)	21 (def.)	Use the same node number as in *DCSLinkNodeID (94.01)* of the field exciter
M1NomFldCur (99.11)	xxx A	I_{FN} = xxx A, rated field current
M1UsedFexType (99.12)	8 = **DCS800-S01**, 9 = **DCS800-S02**	

Parameters to be set in large field exciters:

Before starting with the commissioning set all parameters to default by means of *ApplMacro (99.08)* = **Factory** and *ApplRestore (99.07)* = **Yes**. Check with *MacroSel (8.10)*.

Parameter	Field converter	Comments
CommandSel (10.01)	4 = **FexLink**	
MotFanAck (10.06)	0 = **NotUsed**	
OvrVoltProt (10.13)	2 = **DI2**	depending on hardware connection to DCF506
ArmOvrVoltLev (30.08)	500 %	to suppress **F503 ArmOverVolt** if this does not help, increase *M1NomVolt (99.02)*
OperModeSel (43.01)	1 = **FieldConv**	
CurSel (43.02)	8 = **FexCurRef**	
M1DiscontCurLim (43.08)	0 %	
RevDly (43.14)	50 ms	
FldCtrlMode (44.01)	0 = **Fix** (def.)	

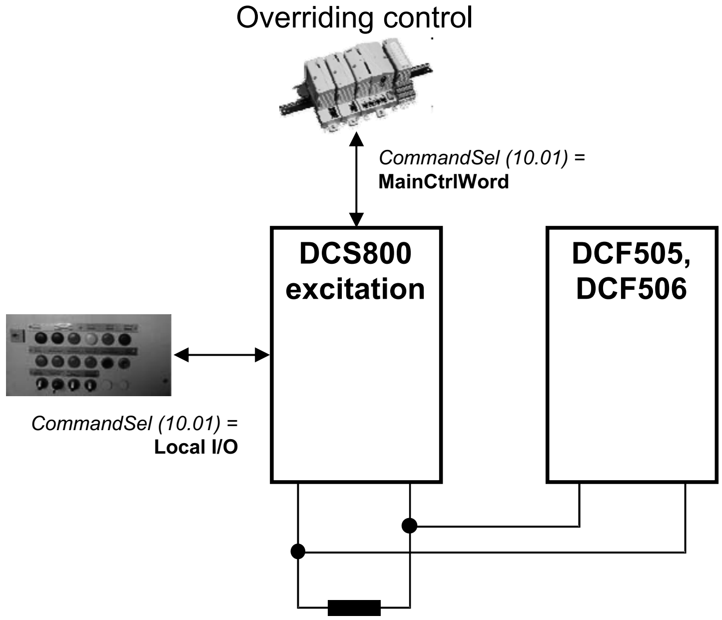
DCSLinkNodeID (94.01)	21 (def.)	Use the same node number as in *M1FexNode (94.08)* of the armature module
DevLimPLL (97.13)	20 °	to suppress blocking of current controller see *CuCtrlStat2 (6.04)* bit 13
M1NomVolt (99.02)	xxx V	U_{FN} = xxx V, rated field voltage
M1NomCur (99.03)	xxx A	I_{FN} = xxx A, rated field current
NomMainsVolt (99.10)	xxx V	U_{NetN} = xxx V; nominal supply voltage (AC)
M1UsedFexType (99.12)	0 = **NotUsed**	

Field current autotuning for large field exciters:

The field current autotuning has to be started directly in the large field exciter:

Parameter	**Field converter**	**Comments**
ServiceMode (99.06)	2 = **FieldCurAuto**	Give the **On** and **Run** command within 20 s
M1KpArmCur (43.06)	xxx	Is set by field current autotuning
M1TiArmCur (43.07)	xxx	Is set by field current autotuning
M1DiscontCurLim (43.08)	0 %	Is set to zero by field current autotuning

Stand alone field exciter



Stand alone field exciter

In the stand alone field exciters set *OperModeSel (43.01)* = **FieldConv** and *CommandSel (10.01)* = **Local I/O** or **MainCtrlWord** as source for the control word (**OnOff1**, **StartStop** and **Reset**). The reference is selected by *CurSel (43.02)* = **CurRefExt** or **AI1** to **AI6**. The field exciter mode uses the standard armature current controller as field current controller. Thus the field current is set by means of *M1NomCur (99.03).*
To close the field contactor use *CurCtrlStat1 (6.03)* bit 7.

Parameters to be set in the stand alone field exciter:

Before starting with the commissioning set all parameters to default by means of *ApplMacro (99.08)* = **Factory** and *ApplRestore (99.07)* = **Yes**. Check with *MacroSel (8.10)*.

Parameter	**Field converter**	**Comments**
CommandSel (10.01)	0 = **Local I/O** (def.), 1 = **MainCtrlWord**	
MotFanAck (10.06)	0 = **NotUsed**	
OvrVoltProt (10.13)	2 = **DI2**	depending on hardware connection to DCF506
ArmOvrVoltLev (30.08)	500 %	to suppress **F503 ArmOverVolt** if this does not help, increase *M1NomVolt (99.02)*
OperModeSel (43.01)	1 = **FieldConv**	

CurSel (43.02)	1 = **CurRefExt**, 2 = **AI1**, 3 = **AI2**, 4 = **AI3**, 5 = **AI4**, 6 = **AI5**, 7 = **AI6**	depending on the connection
CurRefExt (43.03)	xxx %	e.g. written to by overriding control
M1DiscontCurLim (43.08)	0 %	
RevDly (43.14)	50 ms	
FldCtrlMode (44.01)	0 = **Fix** (def.)	
DevLimPLL (97.13)	20 °	to suppress blocking of current controller see *CuCtrlStat2 (6.04)* bit 13
M1NomVolt (99.02)	xxx V	U_{FN} = xxx V, rated field voltage
M1NomCur (99.03)	xxx A	I_{FN} = xxx A, rated field current
NomMainsVolt (99.10)	xxx V	U_{NetN} = xxx V; nominal supply voltage (AC)
M1UsedFexType (99.12)	0 = **NotUsed**	

Field current autotuning for stand alone field exciter:

The field current autotuning has to be started directly in the stand alone field exciter:

Parameter	**Field converter**	**Comments**
ServiceMode (99.06)	2 = **FieldCurAuto**	Give the **On** and **Run** command within 20 s
M1KpArmCur (43.06)	xxx	Is set by field current autotuning
M1TiArmCur (43.07)	xxx	Is set by field current autotuning
M1DiscontCurLim (43.08)	0 %	Is set to zero by field current autotuning

DC-breaker, DC-contactor

General

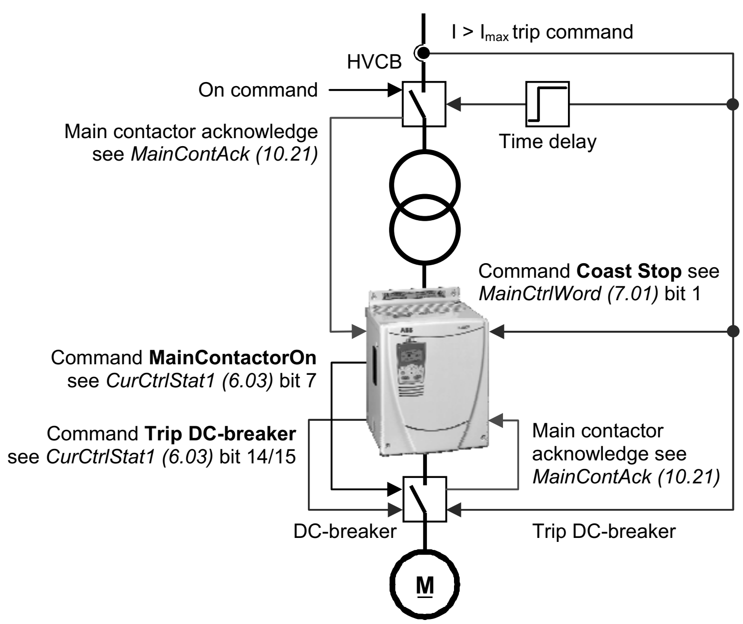
The DC-breaker is used to protect the DC-motor or - in case of too low mains voltage or voltage dips - the generating bridge of the drive from overcurrent. In case of an overcurrent the DC-breaker is forced open by its own tripping spring. DC-breakers have different control inputs and trip devices:

- an On / Off coil with a typical time delay of 100 to 200 ms,
- a high speed tripping coil (e.g. Secheron = CID) to trip the DC-breaker within 2 ms from e.g. the drive,
- an internal tripping spring which is released by overcurrent and set mechanically

There are different ways how to control the DC-breaker depending on the available hardware and the customers on / off philosophy. Following are the most common examples.

Attention:
If a DC breaker is used and DC voltage measurement is taken inside the converter module (D1 – D4 modules and D5 – D7 in default configuration) then deselect the automatic offset compensation by setting *OffsetUDC (97.24)* = 0

HVCB controlled externally, DC-breaker controlled by the drive



HVCB controlled externally, DC-breaker controlled by the drive

In the above example the **H**igh **V**oltage **C**ircuit **B**reaker (HVCB) is controlled externally (e.g. by the operator). The result is checked by means of *MainContAck (10.21)*. In case the main contactor acknowledge is missing **F524 MainContAck** [*FaultWord2 (9.02)* bit 7] is set. Usually HVCB are equipped with an overcurrent relay, which can trip the HVCB. To protect the drive a 50 ms to 100 ms pre-triggered trip command must be connected to **Off2** (Coast Stop) [*MainCtrlWord (7.01)* bit 1]. Additionally the trip command from the HVCB should also trip the DC-breaker.
DC-breaker is controlled by the drive. The drive closes and opens the DC-breaker with the command **MainContactorOn**. The result is checked by means of *MainContAck (10.21)*. In case the main contactor acknowledge is missing **F524 MainContAck** [*FaultWord2 (9.02)* bit 7] is set.
The DC-breaker can be tripped actively by the command **Trip DC-breaker**.

DC-contactor US version

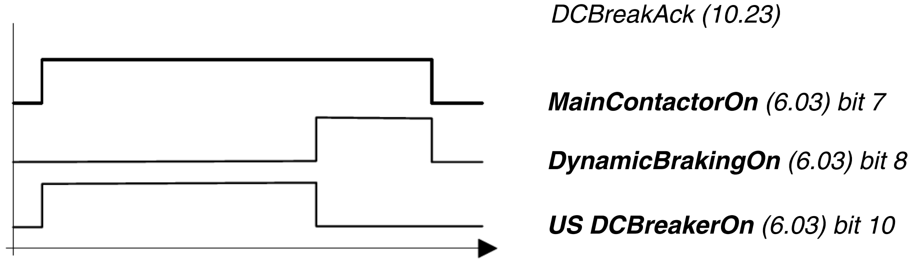
If using a DC contactor, you must connect an auxiliary contact to a digital input of your choice and set para. *MainContAck* accordingly. Set the following parameters:

MainContAck (10.21)	=	**DI1** (or any input you choose for the DC cont. auxiliary contact)
DO8BitNo (14.16)	=	10
MainContCtrlMode (21.16)	=	**DCcontact** (3)

Set these parameters AFTER macros are loaded but BEFORE the drive is commissioned.

Digital output 8 (DO8) must be used to turn the DC-contactor on and off.
DC-contactor US:

DC-contactor US K1.1 is a special designed contactor with 2x NO contacts for C1 and D1 connection and 1x NC contact for connection of Dynamic Brake resistor RB.
The contactor should be controlled by *CurCtrlStat1 (6.03)* bit 10.
The acknowledge can be connected to parameter: *MainContAck (10.21)*
DCBreakAck (10.23)



If using Dynamic Braking, the drive allows you to select the stopping method under three different situations. Parameters 21.02, 21.03 and 21.04 select the stopping method for loss of the OnOff, run command (StartStop, Jog1, Jog2, etc.), and E-Stop input, respectively.
Each can be set to:

- **RampStop**
- **CoastStop**
- **TorqueLimit**
- **DynBraking**

In order to command the drive to perform a DB stop, one or more of these parameters must be set to DynBraking. Most users will want the drive to ramp stop when OnOff or a run command (StartStop, Jog1, Jog2, etc.) input is cleared, and dynamically brake when the E-Stop input is cleared. In that case, use the following settings:

- *Off1Mode (21.02)* = **RampStop**
- *StopMode (21.03)* = **RampStop**
- *E StopMode (21.04)* = **DynBraking**

However, any case is allowed and the final decision is left to the user.

Other parameters control stops during faults.
See:

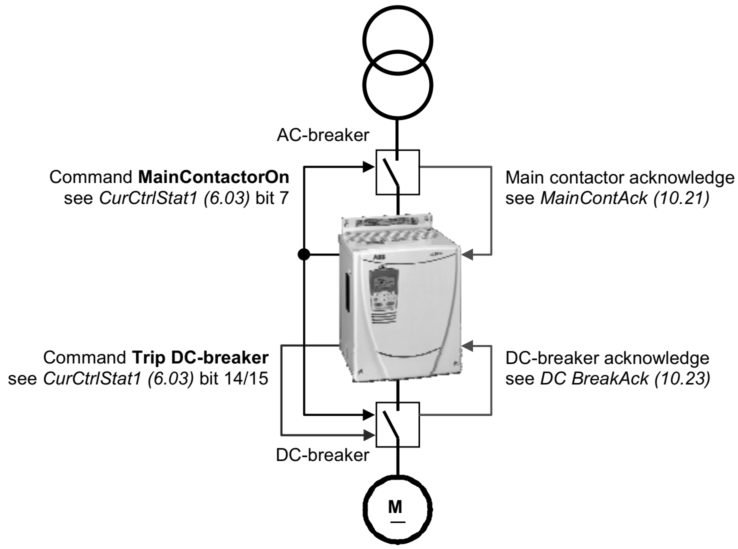
LocalLossCtrl (30.27) *ComLossCtrl (30.28)*
FaultStopMode (30.30) *SpeedFbFltMode (30.36)*

If using EMF feedback with dynamic braking, set:

- *DynBrakeDly (50.11)* = t
  Where: t = the time (sec) it normally takes the motor to stop during dynamic braking

Attention:
If the motor voltage measurement is connected to the motor terminals (D5 – D7 with modified SDCS-PIN-51) then set:
MainContCtrl (21.16) = **On**

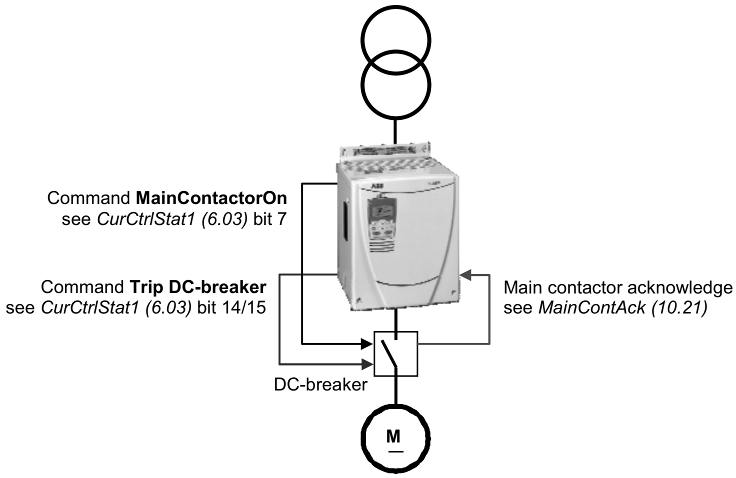
AC- and DC-breaker controlled by the drive



AC- and DC-breaker controlled by the drive

In the above example both, the AC- and the DC-breaker are controlled by the drive. The drive closes and opens both breakers with the command **MainContactorOn**. The result is checked by means of *MainContAck (10.21)* and *DC BreakAck (10.23)*. In case the main contactor acknowledge is missing **F524 MainContAck** [*FaultWord2 (9.02)* bit 7] is set. In case the DC-breaker acknowledge is missing **A103 DC BreakAck** [*AlarmWord1 (9.06)* bit 2] is set,   is forced to 150° and single firing pulses are given.
The DC-breaker can be tripped actively by the command **Trip DC-breaker**.

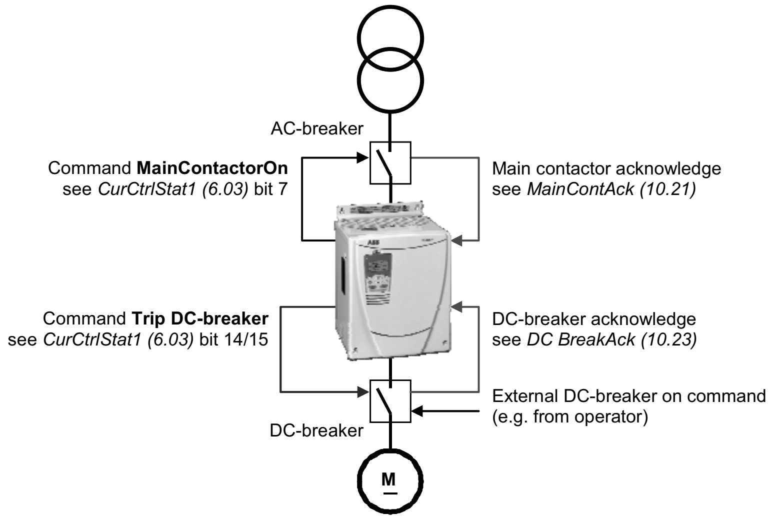
No AC-breaker, DC-breaker controlled by the drive



No AC-breaker, DC-breaker controlled by the drive

In the above example no AC-breaker is used and the DC-breaker is controlled by the drive. The drive closes and opens the DC-breaker with the command **MainContactorOn**. The result is checked by means of *MainContAck (10.21)*. In case the main contactor acknowledge is missing **F524 MainContAck** [*FaultWord2 (9.02)* bit 7] is set.
The DC-breaker can be tripped actively by the command **Trip DC-breaker**.

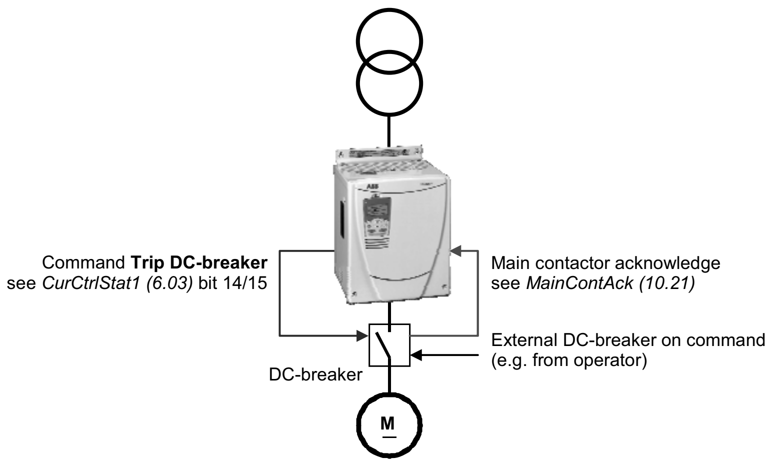
AC-breaker controlled by the drive, DC-breaker controlled externally



AC-breaker controlled by the drive, DC-breaker controlled externally

In the above example the AC-breaker is controlled by the drive. The drive closes and opens the AC-breaker with the command **MainContactorOn**. The result is checked by means of *MainContAck (10.21)*. In case the main contactor acknowledge is missing **F524 MainContAck** [*FaultWord2 (9.02)* bit 7] is set.
The DC-breaker is controlled externally (e.g. by the operator). The result is checked by means of *DC BreakAck (10.23)*. In case the DC-breaker acknowledge is missing **A103 DC BreakAck** [*AlarmWord1 (9.06)* bit 2] is set, is forced to 150° and single firing pulses are given.
The DC-breaker can be tripped actively by the command **Trip DC-breaker**.

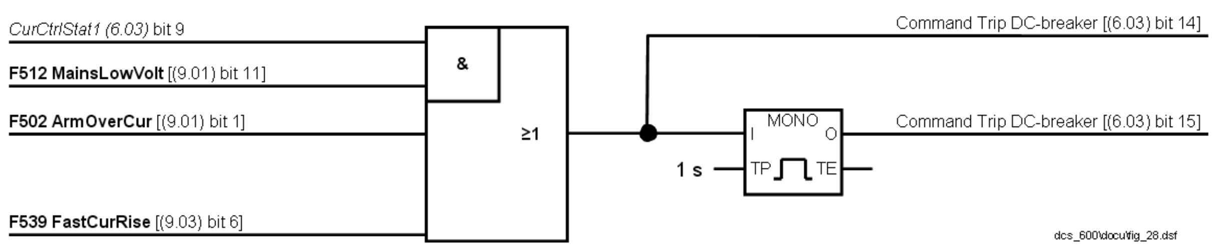
No AC-breaker, DC-breaker controlled externally



No AC-breaker, DC-breaker controlled externally

In the above example no AC-breaker is used and the DC-breaker is controlled externally (e.g. by the operator). The result is checked by means of *MainContAck (10.21)*. In case the main contactor acknowledge is missing **F524 MainContAck** [*FaultWord2 (9.02)* bit 7] is set.
The DC-breaker can be tripped actively by the command **Trip DC-breaker**.

Command Trip DC-breaker



Command **Trip DC-breaker**

The firmware sets the:
- command **Trip DC-breaker** (continuous signal) [*CurCtrlStat1 (6.03)* bit 14] and
- command **Trip DC-breaker** (4 s pulse signal) [*CurCtrlStat1 (6.03)* bit 15]

by means of
- **F512 MainsLowVolt** [FaultWord1 *(9.01)* bit 11] in regenerative mode,
- **F502 ArmOverCur** [FaultWord1 *(9.01)* bit 1] or
- **F539 FastCurRise** [FaultWord3 *(9.03)* bit 6] (see chapter Motor protection)

In case a digital output - see group 14 - is assigned to one of the two signals, it is updated immediately after detecting the fault and thus actively tripping the DC-breaker.

Dynamic braking

General

The drive can be stopped by dynamic braking. The principle is to transfer the power of the machine inertia into a braking resistor. Therefore the armature circuit has to be switched over from the drive to a braking resistor. Additionally flux and field current have to be maintained.

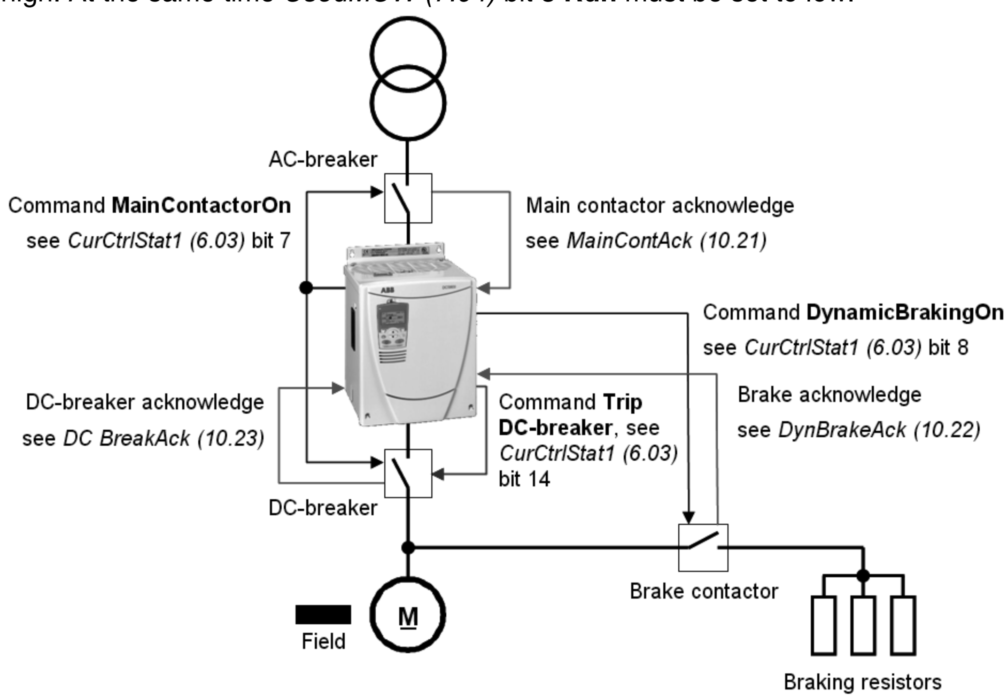
Operation

Activation

Dynamic braking can be activated by all stop modes, in cases of a fault or due to communication breaks:

- *Off1Mode (21.02)* when *UsedMCW (7.04)* bit 0 **On** is set to low,
- *StopMode (21.03)* when *UsedMCW (7.04)* bit 3 **Run** is set to low,
- *E StopMode (21.04)* when *UsedMCW (7.04)* bit 2 **Off3N** is set to low,
- *FaultStopMode (30.30)* in case of a trip level 4 fault,
- *SpeedFbFltMode (30.36)* in case of a trip level 3 fault,
- *LocalLossCtrl (30.27)* when local control is lost,
- *ComLossCtrl (30.28)* when communication is lost,
- *Ch0 ComLossCtrl (70.05)* when communication is lost and
- *Ch2 ComLossCtrl (70.15)* when communication is lost.

In addition dynamic braking can be forced by setting *AuxCtrlWord (7.02)* bit 5 to high. At the same time *UsedMCW (7.04)* bit 3 **Run** must be set to low.



Application example of dynamic braking

Function

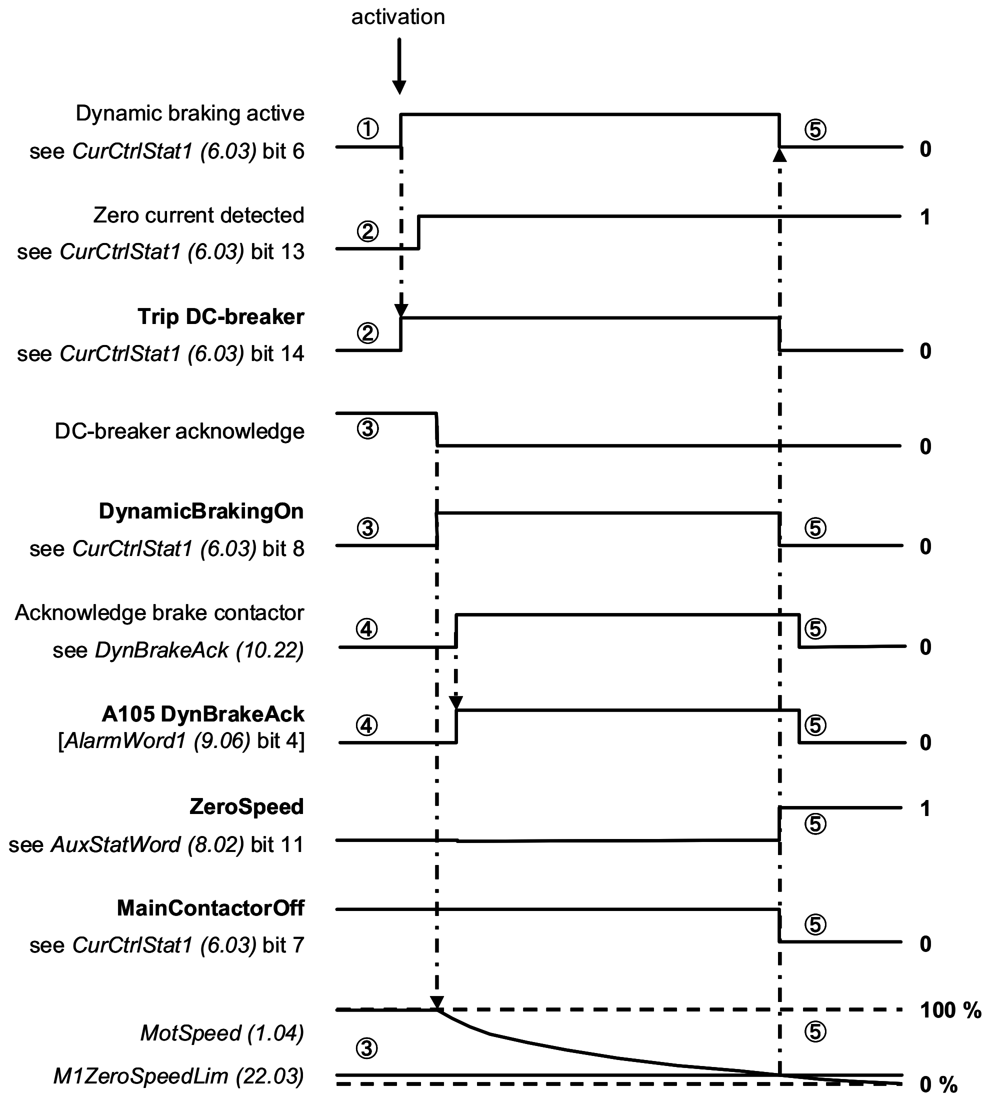
During dynamic braking the field current is maintained by keeping the field exciter activated. It is recommended to supply external / internal field exciters via a short time UPS to make sure that the field is maintained during mains failure.
OnBoard field exciters (D1 to D4) will be supplied via the main contactor, thus

CurCtrlStat1 (6.03) bit 7 stays high (**MainContactorOn)** until zero speed is reached.

① The activation of dynamic braking immediately sets *CurCtrlStat1 (6.03)* bit 6 to high (dynamic braking active**)**.
② Dynamic braking forces the armature current to zero and opens the DC-breaker by setting *CurCtrlStat1 (6.03)* bit 14 to high (**Trip DC-breaker)**.
③ After the armature current is zero and the DC-breaker acknowledge is gone *CurCtrlStat1 (6.03)* bit 8 is set to high (**DynamicBrakingOn)**. This signal is connected to a digital output (see group 14) and used to close the brake contactor. As soon as the brake contactor is closed dynamic braking starts and decreases the speed.
④ With *DynBrakeAck (10.22)* it is possible to select a digital input for the brake resistor acknowledge. This input sets **A105 DynBrakeAck** [*AlarmWord1 (9.06)* bit 4] as long as the acknowledge is present. Thus the drive cannot be started or re-started while dynamic braking is active, except *FlyStart (21.10)* = **FlyStartDyn**.

Deactivation

⑤ Dynamic braking is deactivated as soon as zero speed is reached and *AuxStatWord (8.02)* bit 11 **ZeroSpeed** is set to high.
In case of dynamic braking with EMF feedback [*M1SpeedFbSel (50.03)* = **EMF**] there is no valid information about the motor speed and thus no zero speed information. To prevent an interlocking of the drive after dynamic braking the speed is assumed zero after *DynBrakeDly (50.11)* is elapsed:



Dynamic braking sequence

For usage of US style DC-breakers see *MainContCtrlMode (21.16)*.

Position counter

General

The position counter is used for position measurements. It can be synchronized, that is preset, with an initial value. The counter output value and its initial value are 32-bit signed values. The 32-bit position value is sent to and received as two 16-bit values. Thus the low word dose not possess a sign.

Counting procedure

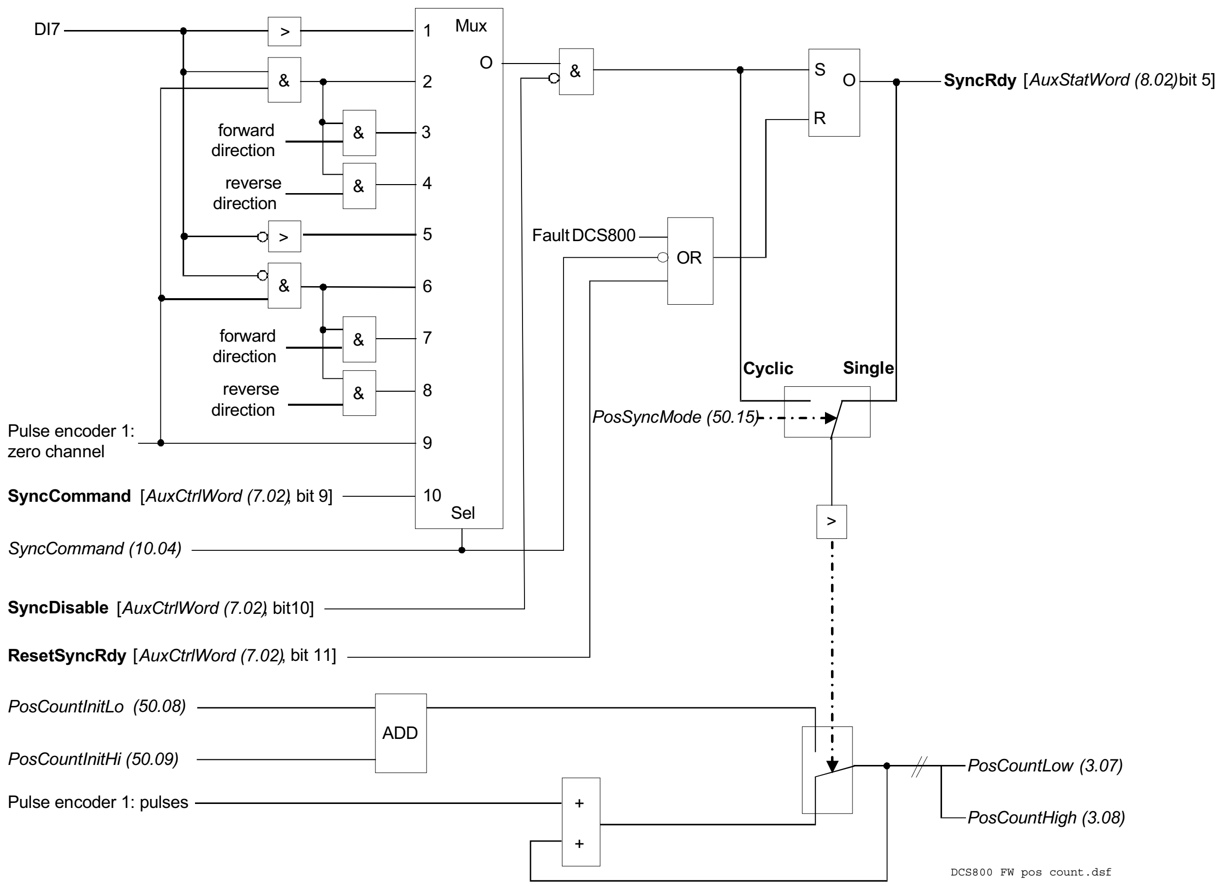
The position counting is only possible when using an encoder, see *M1SpeedFbSel (50.03)*. Its measurement mode is selected by means of *M1EncMeasMode (50.02)* and *PosCountMode (50.07)*. Counting is increasing when the motor is rotating forward and decreasing when the motor is rotating backward. A loss free algorithm is used in order to avoid an increasing error due to rounding errors.

Synchronization

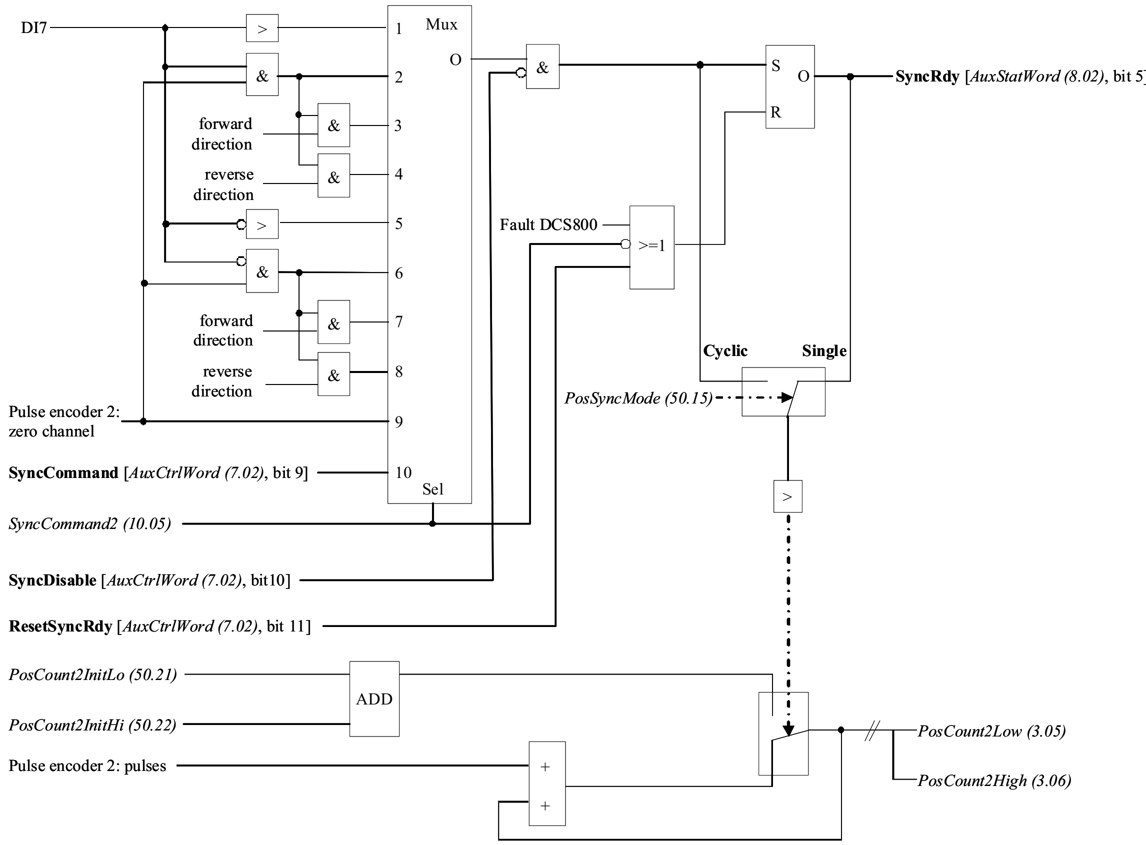
The position counter can be synchronized with an initial value. This initial value is set by means of *PosCountInitLo (50.08)* and *PosCountInitHi (50.09)*.
At the synchronization event the position counter output - *PosCountLow (3.07)* and *PosCountHigh (3.08)* - is preset with the initial value and **SyncRdy** [*AuxStatWord (8.02)*, bit 5] is set:

PosCountInitLo (50.08) ⇨ *PosCountLow (3.07)*
PosCountInitHi (50.09) ⇨ *PosCountHigh (3.08)*

The synchronization command is chosen by means of *SyncCommand (10.04)*. It can either be **SyncCommand** [*AuxCtrlWord (7.02)*, bit 9] or hardware. The fastest synchronization is achieved by the encoder zero pulse. Synchronization by DI7 is delayed due to its scan time and additional hardware filter times.
The synchronization can be inhibited by setting **SyncDisable** [*AuxCtrlWord (7.02)*, bit 10].
SyncRdy [*AuxStatWord (8.02)*, bit 5] can be reset by means of **ResetSyncRdy** [*AuxCtrlWord (7.02)*, bit 11].
With *PosSyncMode (50.15)* either single or cyclic synchronization is selected. With single synchronization, the next synchronization event must be released with **ResetSyncRdy** [*AuxCtrlWord (7.02)*, bit 11].



Pulse encoder 1 position counter logic



Pulse encoder 2 position counter logic

I/O configuration

Chapter overview

This chapter describes the I/O configuration of digital and analog inputs and outputs with different hardware possibilities.

Digital inputs (DI's)

The basic I/O board is the SDCS-CON-4 with 8 standard DI's. All 8 standard DI's can be replaced with SDCS-IOB-2 and extended by means of one or two RDIO-01 digital I/O extension modules. Thus the maximum number of DI's is 14.

The hardware source is selected by:
- *DIO ExtModule1 (98.03)* for DI9 to DI11
- *DIO ExtModule2 (98.04)* for DI12 to DI14 and
- *IO BoardConfig (98.15)*

Note:
The maximum amount of digital I/O extension modules is two regardless if an AIMA-01 board is used.

SDCS-CON-4 / SDCS-IOB-2

On the SDCS-CON-4 the standard DI's are filtered and not isolated. On the SDCS-IOB-2 the standard DI's are filtered and isolated. Selectable hardware filtering time (DI7 and DI8 on the SDCS-IOB-2):
- 2 ms or 10 ms (jumper S7 and S8)

Input voltages:
- 24 VDC to 48 VDC, 115 VAC or 230 VAC depending on the hardware
- for more details see *DCS800 Hardware Manual*

Scan time for DI1 to DI6:
- 5 ms

Scan time for DI7 and DI8:
- 3.3 ms / 2.77 ms (synchronized with mains frequency)

1st and 2nd RDIO-01

The extension DI's are isolated and filtered. Selectable hardware filtering time:
- 2 ms or 5 ms to 10 ms

Input voltages:
- 24 VDC to 250 VDC, 110 VAC to 230 VAC
- for more details see *RDIO-01 User's Manual*

Scan time for DI9 to DI14:
- 5 ms connected at SDCS-CON-4
- 14 ms connected via SDCS-COM-8

Attention:
To ensure proper connection and communication of the RDIO-01 boards with the SDCS-CON-4 use the screws included in the scope of delivery.

Configuration

All DI’s can be read from *DI StatWord (8.05)*:

bit	DI	configurable	default setting
0	1	yes	*ConvFanAck (10.20)*
1	2	yes	*MotFanAck (10.06)*
2	3	yes	*MainContAck (10.21)*
3	4	yes	*Off2 (10.08)*
4	5	yes	*E Stop (10.09)*
5	6	yes	*Reset (10.03)*
6	7	yes	*OnOff1 (10.15)*
7	8	yes	*StartStop (10.16)*
8	9	yes	-
9	10	yes	-
10	11	yes	-
11	12	no	not selectable
12	13	no	not selectable
13	14	no	not selectable

Configurable = yes:
The DI’s can be connected to several converter functions and it is possible to invert the DI’s - *DI1Invert (10.25)* to *DI11Invert (10.35)*. In addition the DI’s can be used by Adaptive Program, application program or overriding control.

Configurable = no:
The DI’s can only be used by Adaptive Program, application program or overriding control.

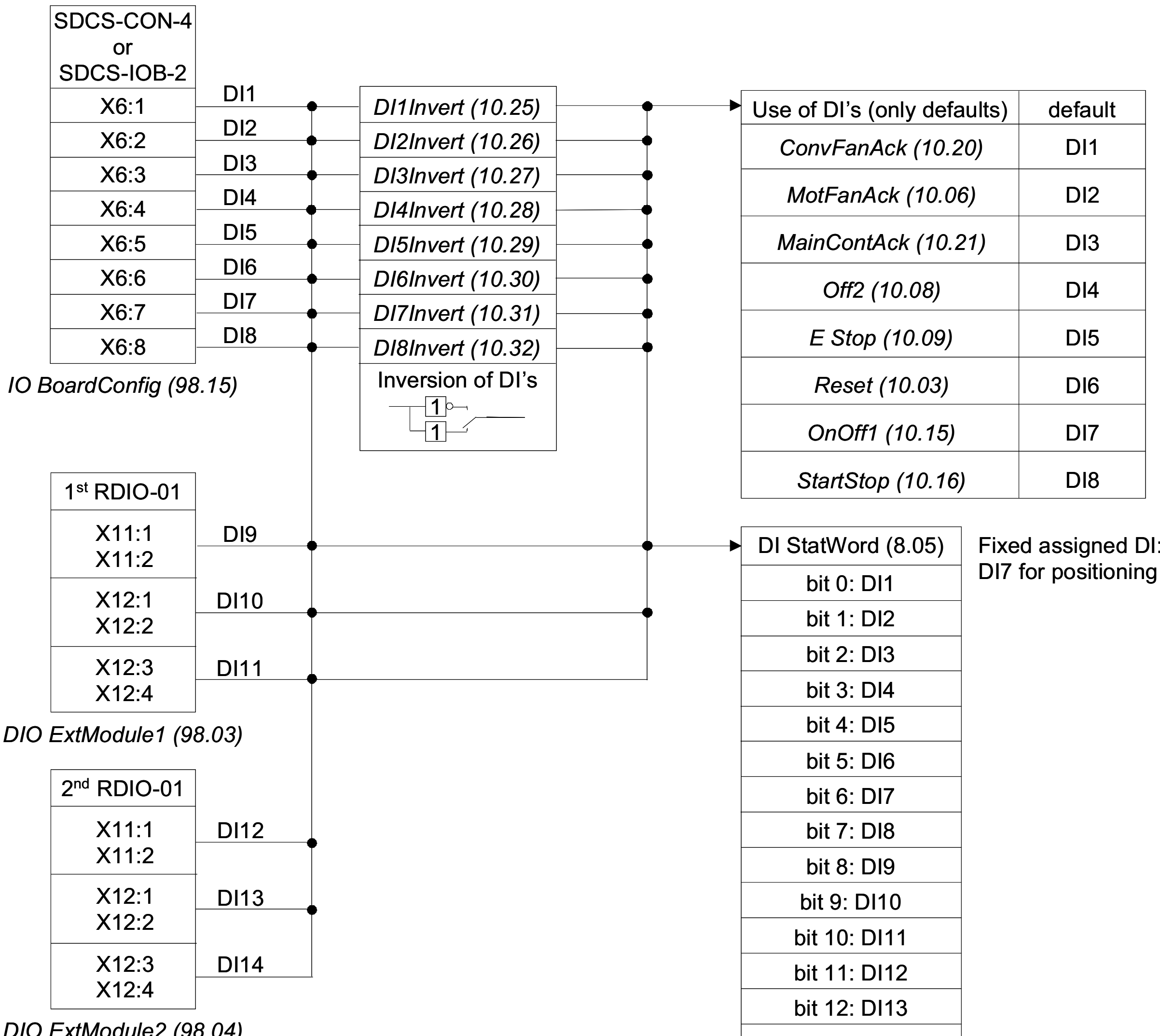
Configurable DI’s are defined by means of following parameters:

- *Direction (10.02)*
- *Reset (10.03)*
- *SyncCommand (10.04)*
- *MotFanAck (10.06)*
- *HandAuto (10.07)*
- *Off2 (10.08)*
- *E Stop (10.09)*
- *ParChange (10.10)*
- *OvrVoltProt (10.13)*
- *OnOff1 (10.15)*
- *StartStop (10.16)*
- *Jog1 (10.17)*
- *Jog2 (10.18)*
- *ConvFanAck (10.20)*
- *MainContAck (10.21)*
- *DynBrakeAck (10.22)*
- *DC BreakAck (10.23)*
- *Ref1Mux (11.02)*
- *Ref2Mux (11.12)*
- *MotPotUp (11.13)*
- *MotPotDown (11.14)*
- *MotPotMin (11.15)*
- *Ramp2Select (22.11)*
- *Par2Select (24.29)*
- *TorqMux (26.05)*
- *ResCurDetectSel (30.05)*
- *ExtFaultSel (30.31)*
- *ExtAlarmSel (30.32)*
- *M1KlixonSel (31.08)*
- *M1BrakeAckSel (42.02)*
- *FldBoostSel (44.17)*
- *M2KlixonSel (49.38)*
- *ZeroCurDetect (97.18)*
- *ResetAhCounter (97.21)*

Following restrictions apply:

- The position counter synchronization is fixed assigned to input DI7, if activated via *SyncCommand (10.04)*

– DI12 to DI14 are only available in the *DI StatWord (8.05)*, thus they can only be used by Adaptive Program, application program or overriding control



Structure of DI's

Digital outputs (DO’s)

The basic I/O board is the SDCS-CON-4 with 7 standard DO’s. Standard DO8 is located on the SDCS-PIN-4 for units size D1 - D4 or SDCS-POW-4 for units size D5 - D7. All 8 standard DO’s can be replaced with SDCS-IOB-2 and extended by means of one or two RDIO-01 digital I/O extension modules. Thus the maximum number of DO’s is 12.

The hardware source is selected by:

- *DIO ExtModule1 (98.03)* for DO9 and DO10
- *DIO ExtModule2 (98.04)* for DO11 and DO12
- *IO BoardConfig (98.15)*

Note:
The maximum amount of digital I/O extension modules is two regardless if an AIMA-01 board is used.

SDCS-CON-4 / SDCS-IOB-2

On the SDCS-CON-4 the standard DO’s are relay drivers. DO8 is located on the SDCS-PIN-4 and is isolated by means of a relay. If the SDCS-IOB-2 is being used DO6 and DO7 are isolated by means of optocouplers, while the others (DO1 to DO5 and DO8) are isolated by means of relays.

Output values SDCS-CON-4:

- DO1 to DO7 max. 50 mA / 22 VDC at no load
- for more details see *DCS800 Hardware Manual*

Output values SDCS-PIN-4:

- DO8 max. 3 A / 24 VDC, max. 0.3 A / 115 VDC / 230 VDC or max. 3 A / 230 VAC
- for more details see *DCS800 Hardware Manual*

Output values SCDS-IOB-2:

- DO6 and DO7: max. 50 mA / 24 VDC
- all others: max. 3 A / 24 VDC, max. 0.3 A / 115 VDC / 230 VDC or max. 3 A / 250 VAC
- for more details see *DCS800 Hardware Manual*

Cycle time for DO1 to DO8:

- 5 ms

1st and 2nd RDIO-01

The extension DO’s are isolated by means of relays.

Output values:

- max. 5 A / 24 VDC, max. 0.4 A / 120 VDC or max. 1250 VA / 250 VAC
- for more details see *RDIO-01 User’s Manual*

Cycle time for DO9 to DO12:

- 5 ms connected at SDCS-CON-4
- 14 ms connected via SDCS-COM-8

Attention:
To ensure proper connection and communication of the RDIO-01 boards with the SDCS-CON-4 use the screws included in the scope of delivery.

Configuration

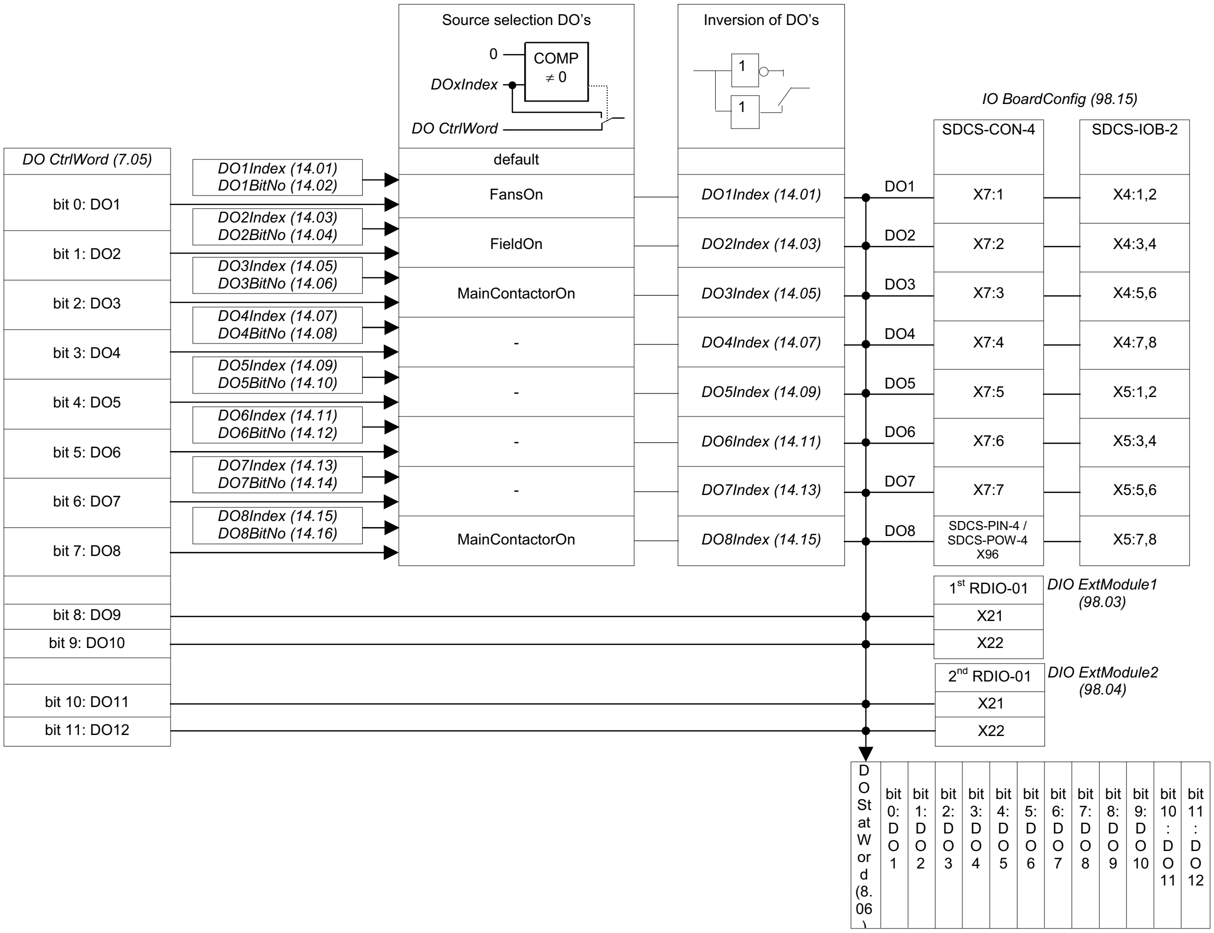
All DO's can be read from *DO StatWord (8.06)*:

bit	DO	configurable	default setting	
0	1	yes	**FansOn**; *CurCtrlStat1 (6.03)*	bit0
1	2	yes	**FieldOn**; *CurCtrlStat1 (6.03)*	bit5
2	3	yes	**MainContactorOn**; *CurCtrlStat1 (6.03)*	bit7
3	4	yes	-	
4	5	yes	-	
5	6	yes	-	
6	7	yes	-	
7	8	yes	**MainContactorOn**; *CurCtrlStat1 (6.03)*	bit7
8	9	no	not selectable	
9	10	no	not selectable	
10	11	no	not selectable	
11	12	no	not selectable	

Configurable = yes:
The DO's can be connected to any integer or signed integer of the drive by means of group 14. It is possible to invert the DO's by simply negate *DO1Index (14.01)* to *DO8Index (14.15)*. In addition the DO's can be used by Adaptive Program, application program or overriding control if the corresponding *DOxIndex (14.xx)* is set to zero - see *DO CtrlWord (7.05)*.

Configurable = no:
The DO's can only be used by Adaptive Program, application program or overriding control - see *DO CtrlWord (7.05)*.

Note:
DO8 is only available as relay output on the SDCS-PIN-4, if no SDCS-IOB-2 is used.



Structure of DO's

Analog inputs (AI's)

The basic I/O board is the SDCS-CON-4 with 4 standard AI's. All 4 standard AI's can be replaced with SDCS-IOB-3 and extended by means of one or two RAIO-01 analog I/O extension modules. Thus the maximum number of AI's is 8.

The hardware source is selected by:
- *AIO ExtModule (98.06)* for AI5 and AI6
- *AIO MotTempMeas (98.12)* for AI7 and AI8
- *IO BoardConfig (98.15)*

Note:
The maximum amount of analog I/O extension modules is two regardless if an AIMA-01 board is used.

SDCS-CON-4

Hardware setting:
- switching from voltage input to current input by means of jumper S2 and S3
- for more details see *DCS800 Hardware Manual*

Input range AI1 and AI2 set by parameter:
- ±10 V, 0 V to 10 V, 2 V to 10 V, 5 V offset, 6 V offset
- ±20 mA, 0 mA to 20 mA, 4 mA to 20 mA, 10 mA offset, 12 mA offset

Input range AI3 and AI4 set by parameter:
- ±10 V, 0 V to 10 V, 2 V to 10 V, 5 V offset, 6 V offset

Resolution:
- 15 bits + sign

Scan time for AI1 and AI2:
- 3.3 ms / 2.77 ms (synchronized with mains frequency)

Scan time for AI3 and AI4:
- 5 ms

Additional functions:
- motor temperature measurement for a PTC connected to AI2 - see *section Motor protection*

SDCS-IOB-3

Hardware setting:
- switching from voltage input to current input by means of jumper S1
- the hardware gain for AI2 and AI3 can be increased by 10 with jumpers S2 and S3, thus the input range changes e.g. from ±10 V to ±1 V
- for more details see *DCS800 Hardware Manual*

Input range AI1 to AI4 set by parameter:
- ±10 V, 0 V to 10 V, 2 V to 10 V, 5 V offset, 6 V offset
- ±20 mA, 0 mA to 20 mA, 4 mA to 20 mA, 10 mA offset, 12 mA offset

Resolution:
- 15 bits + sign

Scan time for AI1 and AI2:
- 3.3 ms / 2.77 ms (synchronized with mains frequency)

Scan time for AI3 and AI4:
- 5 ms

Additional functions:
- motor temperature measurement for PT100 or PTC connected to AI2 and AI3 - see *section* Motor protection
- residual current detection monitor input via AI4 - see *section* Motor protection

1st RAIO-01

Hardware setting:
- input range and switching from voltage to current by means of a DIP switch,
- for more details see *RAIO-01 User's Manual*

Input range AI5 and AI6 set by parameter:
- ±10 V, 0 V to 10 V, 2 V to 10 V, 5 V offset, 6 V offset
- ±20 mA, 0 mA to 20 mA, 4 mA to 20 mA, 10 mA offset, 12 mA offset

Resolution:
- 11 bits + sign

Scan time for AI5 and AI6:
- 10 ms connected at SDCS-CON-4
- 14 ms connected via SDCS-COM-8

Additional functions:
- all AI's are galvanically isolated

Attention:
To ensure proper connection and communication of the RAIO-01 board with the SDCS-CON-4 use the screws included in the scope of delivery.

2nd RAIO-01

Hardware setting:
- AI7 and AI8 are only used for motor temperature measurement, thus set 0 V to 2 V for 1 PT100 respectively 0 V to 10 V for 2 or 3 PT100 using the DIP switch
- for more details see *RAIO-01 User's Manual*

Resolution:
- 11 bits + sign

Scan time for AI7 and AI8:
- 10 ms connected at SDCS-CON-4
- 14 ms connected via SDCS-COM-8

Additional functions:
- all AI's are galvanically isolated
- motor temperature measurement for PT100 connected to AI7 and AI8 - see *section* Motor protection,

Attention:
To ensure proper connection and communication of the RAIO-01 board with the SDCS-CON-4 use the screws included in the scope of delivery.

Configuration

The value of AI1 to AI6 and AITacho can be read from group 5.

AI	configurable	default setting
1	yes	-
2	yes	-
3	yes	-
4	yes	-
5	yes	-
6	yes	-
7	temperature	-
8	temperature	-

Configurable = yes:
The AI's can be connected to several converter functions and it is possible to scale them by means of group 13. In addition the AI's can be read by Adaptive Program, application program or overriding control.

Configurable = temperature:
The AI's can only be used by the motor temperature measurement - see *M1TempSel (31.05)* and *M2TempSel (49.35)*.

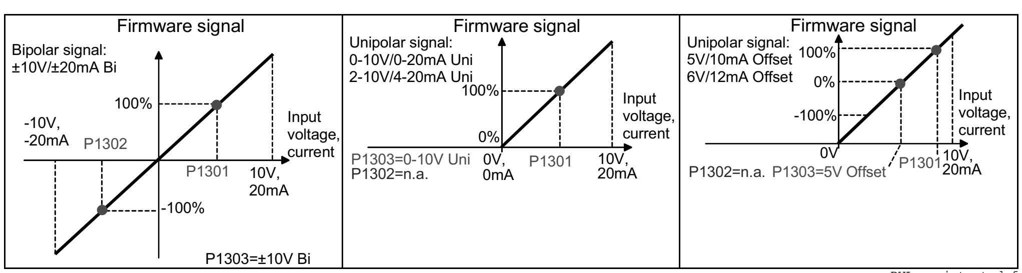
Configurable AI's are defined by means of following parameters:

- *Ref1Sel (11.03)*
- *Ref2Sel (11.06)*
- *TorqUsedMaxSel (20.18)*
- *TorqUsedMinSel (20.19)*
- *TorqRefA Sel (25.10)*
- *TorqCorrect (26.15)*
- *ResCurDetectSel (30.05)*
- *M1TempSel (31.05)*
- *M1StrtTorqRefSel (42.07)*
- *CurSel (43.02)*
- *M2TempSel (49.35)*
- *M2StrtTorqRefSel (49.44)*

Following restrictions apply:

- the residual current detection input is fixed assigned to AI4, if activated via *ResCurDetectSel (30.05)*
- the motor temperature measurement is fixed assigned to AI2 and AI3 respectively AI7 and AI8, if activated via *M1TempSel (31.05)* respectively *M2TempSel (49.35)*

Scaling

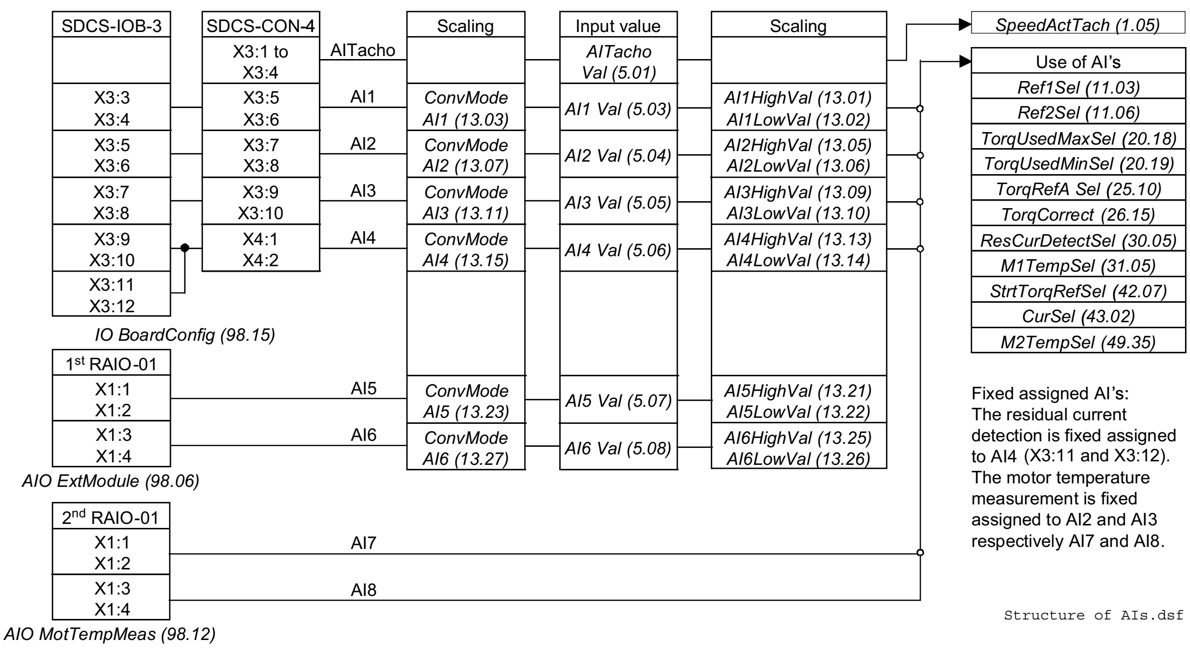


It is possible to scale AI1 to AI6 with 3 parameters each:

- the range of each AI is set by means of a jumper - distinguishing between current and voltage - and *ConvModeAI1 (13.03)* to *ConvModeAI6 (13.27)*
- +100 % of the input signal connected to an AI is scaled by means of *AI1HighVal (13.01)* to *AI6HighVal (13.25)*
- -100 % of the input signal connected to an AI is scaled by means of *AI1LowVal (13.02)* to *AI6LowVal (13.26)*

  Example:

  In case the min. / max. voltage (±10 V) of AI1 should equal ±250 % of *TorqRefExt (2.24)*, set:
  TorqRefA Sel (25.10) = **AI1**
  ConvModeAI1 (13.03) = ±**10V Bi**
  AI1HighVal (13.01) = 4000 mV
  AI1LowVal (13.02) = -4000 mV



Structure of AI's

Analog outputs (AO's)

The basic I/O board is the SDCS-CON-4 with 3 standard AO's. Two AO's are programmable, the third one is fixed and used to display the actual armature current taken directly from the burden resistors. All 3 standard AO's can be replaced with SDCS-IOB-3 and extended by means of one or two RAIO-01 analog I/O extension modules. Thus the maximum number of AO's is 7.

The hardware source is selected by:

- *AIO ExtModule (98.06)* for AO3 and AO4
- *AIO MotTempMeas (98.12)* for AO5 and AO6
- *IO BoardConfig (98.15)*

Note:
The maximum amount of analog I/O extension modules is two regardless if an AIMA-01 board is used.

SDCS-CON-4 / SDCS-IOB-3

Output range AO1 and AO2 set by parameter:

- ±10 V, 0 V to 10 V, 2 V to 10 V, 5 V offset, 6 V offset

Output range fixed AO I-act:

- 8 V equals the minimum of 325 % *M1NomCur (99.03)* or 230 % *ConvNomCur (4.05)*
- see also *IactScaling (4.26)*
- for more details see *DCS800 Hardware Manual*

Resolution:

- 11 bits + sign

Cycle time for AO1 and AO2:

- 5 ms

Cycle time fixed AO I-act:

- directly taken from hardware

Additional functions:

- the gain of the fixed AO I-act can be adjusted by means of R110 on the SDCS-IOB-3

1st RAIO-01

Output range AO3 and AO4 set by parameter:

- 0 mA to 20 mA, 4 mA to 20 mA, 10 mA offset, 12 mA offset

Resolution:

- 12 bits

Cycle time for AO3 and AO4:

- 5 ms connected at SDCS-CON-4
- 14 ms connected via SDCS-COM-8

Additional functions:

- all AO's are galvanically isolated

Attention:
To ensure proper connection and communication of the RAIO-01 board with the SDCS-CON-4 use the screws included in the scope of delivery.

2nd RAIO-01

Hardware settings:

- AO5 and AO6 are only used for motor temperature measurement, no additional setting needed
- for more details see *RAIO-01 User's Manual*

Resolution:

- 12 bits

Cycle time for AO5 and AO6:

- 5 ms connected at SDCS-CON-4
- 14 ms connected via SDCS-COM-8

Additional functions:

- all AO's are galvanically isolated
- motor temperature measurement for PT100 connected to AO5 and AO6 - see *section* <u>Motor protection</u>

Attention:
To ensure proper connection and communication of the RAIO-01 board with the SDCS-CON-4 use the screws included in the scope of delivery.

Configuration

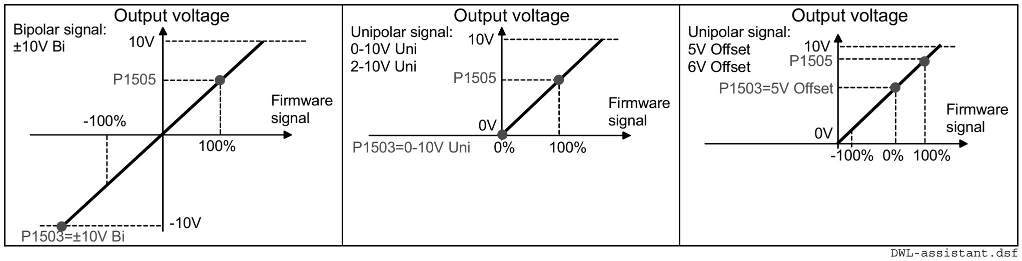
The value of AO1 and AO2 can be read from group 5.

AO	configurable	default setting
1	yes	-
2	yes	-
3	yes	-
4	yes	-
5	temperature	-
6	temperature	-
Curr	fixed	not selectable

Configurable = yes:
The AO's can be connected to any integer or signed integer of the drive by means of group 15. It is possible to invert the AO's by simply negate *IndexAO1 (15.01)* to *IndexAO4 (15.16)*. In addition the AO's can be used by Adaptive Program, application program or overriding control if the corresponding *IndexAOx (15.xx)* is set to zero - see *CtrlWordAO1 (15.02)* to *CtrlWordAO4 (15.17)*.

Configurable = temperature:
The AO's can only be used by the motor temperature measurement - see *M1TempSel (31.05)* and *M2TempSel (49.35)*.

Scaling

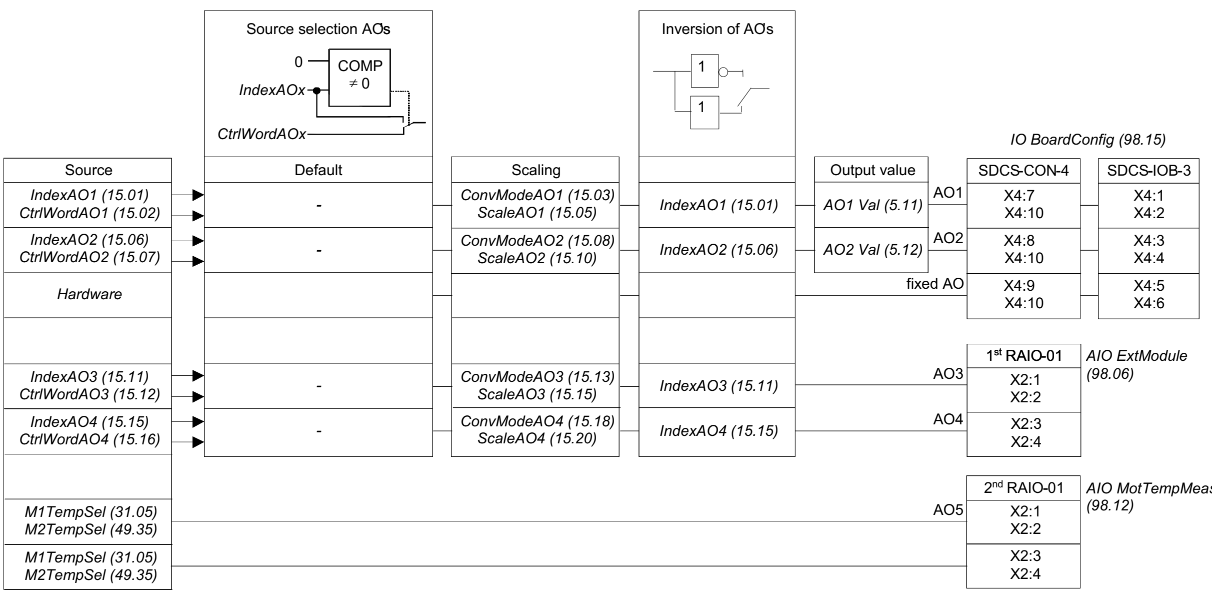


It is possible to scale AO1 to AO4 with 2 parameters each:

- the range of each AO is set by means of *ConvModeAO1 (15.03)* to *ConvModeAO4 (15.18)*
- if the range is set to bipolar or unipolar signals with offset, ±100 % of the input signal connected to an AO is scaled by means of *ScaleAO1 (15.06)* to *ScaleAO4 (15.20)*
- If the range is set to unipolar signals without offset, only +100 % of the input signal connected to an AO is scaled by means of *ScaleAO1 (15.06)* to *ScaleAO4 (15.20).* The smallest value is always zero.
- It is possible to invert the AO's by simply negate *IndexAO1 (15.01)* to *IndexAO4 (15.16)*

  Example:

  In case the min. / max. voltage (±10 V) of AO1 should equal ±250 % of *TorqRefUsed (2.13)*, set:
  IndexAO1 (15.01) = 213
  ConvModeAO1 (15.03) = ±**10V Bi**
  ScaleAO1 (15.05) = 4000 mV



Structure of AO's

Communication

Chapter overview

This chapter describes the communication capabilities of the drive.

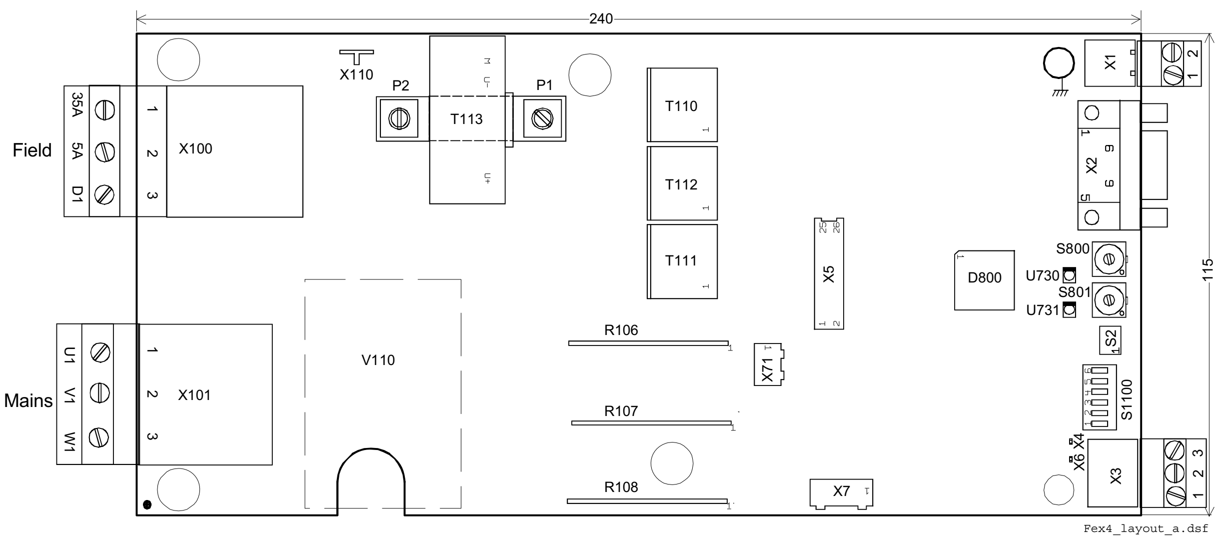
DCSLink with SDCS-DSL-4

General

The DCSLink is a multi-purpose twisted pair bus for the DCS800. All functions using the same hardware and can be used at the same time. The DCSLink can be used for excitation, master-follower, drive-to-drive communication and 12-pulse.

Excitation, commissioning a FEX-4

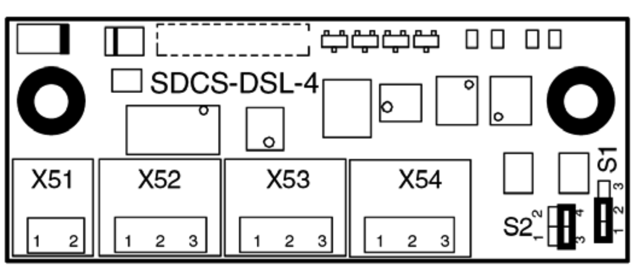
Layout FEX-4



X1:	24 V supply
X1:1	24 V DC
X1:2	0 V DC

X3:	DSL Link
X3:1	GND B
X3:2	CAN L
X3:3	CAN H

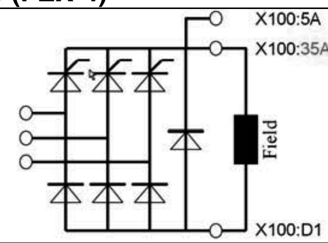
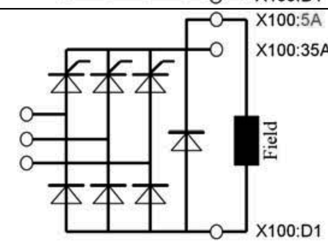
Layout SDCS-DSL-4



Set the FEX-4 type

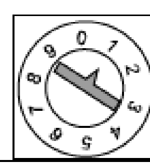
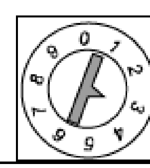
The FEX-4 can be used in 4 different applications:

- **FEX-425-Int** (as internal field exciter of a D5 module with up to 25 A)
- **DCF803-0016** (as external field exciter with up to 16 A)
- **DCF803-0035** (as external field exciter with up to 35 A) and
- **FEX-4 Term5A** (as internal or external field exciter with max. 5 A)

Firmware (armature converter)	Hardware (FEX-4)
M1UsedFexType (99.12) = **FEX-425-Int**, **DCF803-0016** or **DCF803-0035**	
M1UsedFexType (99.12) = **FEX-4 Term5A**	

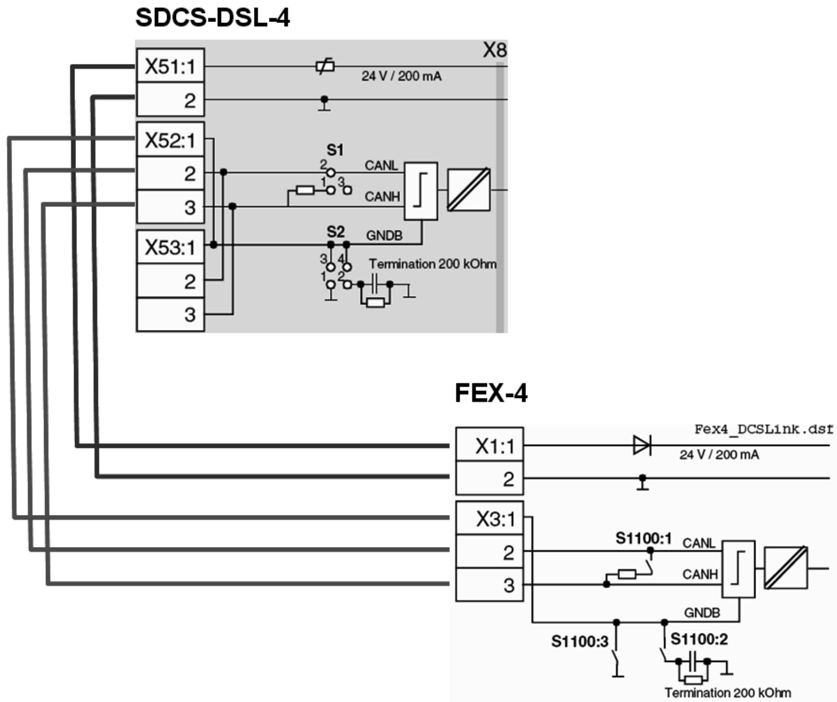
Set the node numbers, transmission speed and the communication supervision

In all bus systems unique node ID numbers are required and have to be set in the armature converter and the FEX-4. Two stations with the same node ID number are not allowed.
For example set the armature converter node ID number to 1 and the FEX-4 node ID number to 13.
The communication supervision is activated in the armature converter.
Also the transmission speed of all converters has to match:

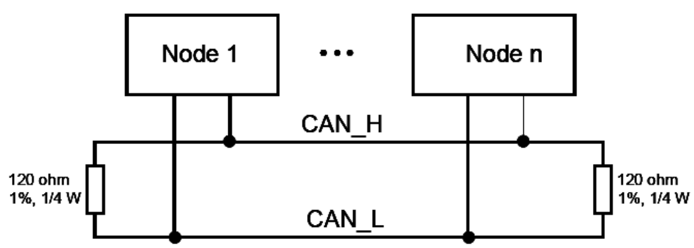
Firmware (armature converter)	Hardware (FEX-4)			
DCSLinkNodeID (94.01) = 1	-			
BaudRate (94.02) = **500 kBit/s**	**S1100:4**	**S1100:5**	**S1100:6**	**kBit/s**
	OFF	**OFF**	**ON**	**500**
FexTimeOut (94.07) = 100 ms	-			
M1FexNode (94.08) = 13	**S801**		**S800**	
	1		**3**	
	S801 		S800 	

Set the DCSLink

Cable connection:



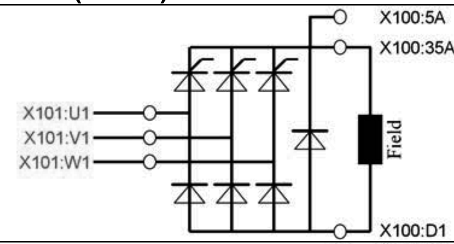
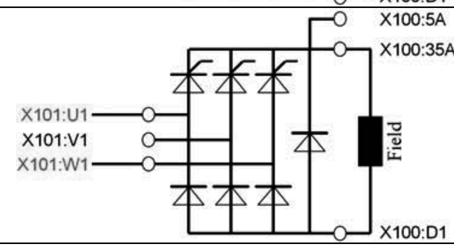
Bus- and ground termination:
The DCSLink is a bus system using twisted pair cables. Therefore bus termination is mandatory at the two physical ends of the bus.



Hardware (SDCS-DSL-4)	Hardware (FEX-4)
jumper **S1** = **1-2** if bus termination is needed	jumper **S1100:1** = **ON** if bus termination is needed
jumper **S2** sets the ground termination	jumper **S1100:2** and **S1100:3** set the ground termination

Set the supply of the FEX-4

The FEX-4 can be either supplied by 1-phase or by 3-phases:

Firmware (armature converter)	Hardware (FEX-4)
M1OperModeFex4 (45.22) = **3-phase**	
M1OperModeFex4 (45.22) = **1-phase**	

Checking the FEX-4

There are several signals to check the FEX-4 installation:

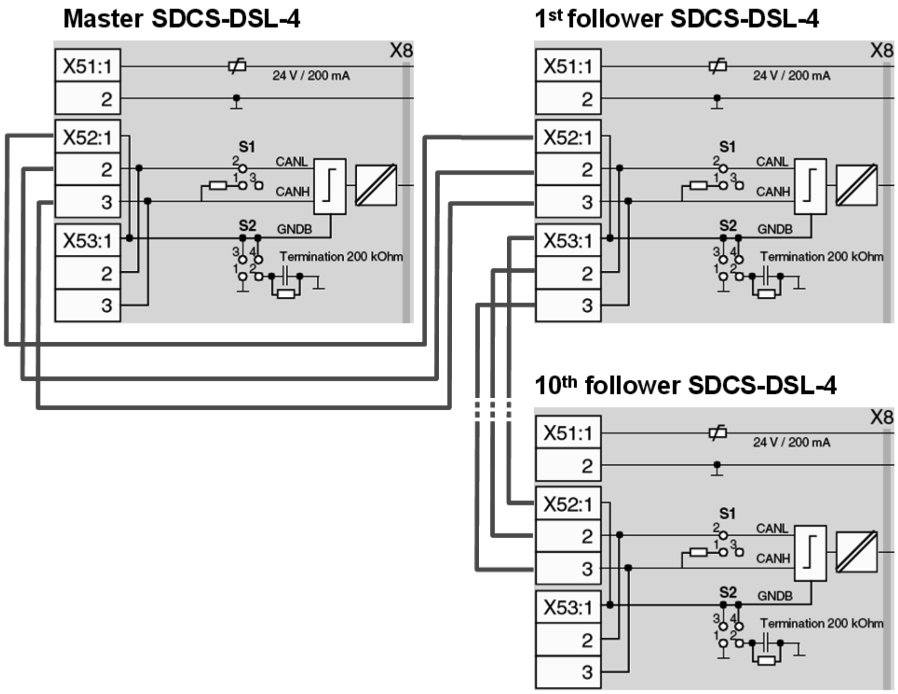
Firmware (armature converter)		Hardware (FEX-4)	
Mot1FexType (4.06)	shows the FEX-4 type as chosen with *M1UsedFexType (99.12)*	yellow (U731) or green (U730) LED is blinking:	waiting for DCSLink communication
DCSLinkStat1 (4.18) or *DCSLinkStat2 (4.19)*	show the status of the field exciter node as chosen with *M1FexNode (94.08)*	yellow (U731) or green (U730) LED is steady:	DCSLink communication is OK

For further information consult the *DCS800 Hardware Manual.*

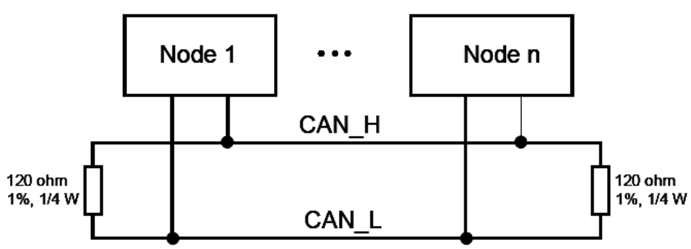
Master-follower, commissioning

Set the DCSLink hardware

Cable connection:



Bus- and ground termination:
The DCSLink is a bus system using twisted pair cables. Therefore bus termination is mandatory at the two physical ends of the bus.



In the above example termination is mandatory at the master and the 10^{th} follower.

SDCS-DSL-4
jumper **S1** = **1-2** sets the bus termination
jumper **S2** sets the ground termination

Set the node ID numbers and transmission speed

In all bus systems unique node ID numbers are required and have to be set in the master and all followers. Two stations with the same node ID number are not allowed.
For example set the masters node ID number to 1 and add one for each follower.
Also the transmission speed of all converters has to match:

Firmware master	Firmware first follower	Firmware 10th follower
DCSLinkNodeID (94.01) = 1	*DCSLinkNodeID (94.01)* = 2	*DCSLinkNodeID (94.01)* = 11
BaudRate (94.02) = **500kBit/s**	*BaudRate (94.02)* = **500kBit/s**	*BaudRate (94.02)* = **500kBit/s**

Activate the mailboxes

The master-follower communication utilizes 4 mailboxes for data transfer. Thus data transfer to any device / node in the system is possible.
Positive mailbox node ID numbers only transmit data, negative only receive data.
To get communication mailbox node ID pairs (e.g. 5 and -5) are needed:

Firmware master	Firmware first follower	Firmware 10th follower
MailBox1 (94.12) = 5	*MailBox1 (94.12)* = -5	*MailBox1 (94.12)* = -5

Attention:
Positive mailbox node ID numbers must be unique. Negative mailbox node ID numbers can be used by several mailboxes.
The master mailbox one for example is set to 5 and thus transmitting data. Mailbox one of the followers is set to -5 and thus receiving data.

Activate the communication supervision

The communication supervision is activated by means of *MailBoxCycle1 (94.13).*
The function of *MailBoxCycle1 (94.13)* is depending on the setting of *MailBox1 (94.12).*
If *MailBox1 (94.12)* is positive:

- data will be transmitted.
- *MailBoxCycle1 (94.13)* sets the transmitting and receiving intervals.
- if *MailBoxCycle1 (94.13)* is set to 3 ms the transmit and receiving intervals are synchronized with mains frequency, either 3.3 ms or 2.77 ms.
- values from 1 - 2 ms are too fast and will generate a fault.
- the communication is inactive, if *MailBoxCycle1 (94.13)* is set to 0 ms.

If MailBox1 (94.12) is negative:

- data will be received.
- *MailBoxCycle1 (94.13)* sets the communication timeout. This is the time delay before a drive-to-drive or master-follower communication break is declared. Depending on the setting of *ComLossCtrl (30.28)* either **F544 P2PandMFCom** [*FaultWord3 (9.03)* bit 11] or **A112 P2PandMFCom** [*AlarmWord1 (9.06)* bit 11] is set.
- the communication fault and alarm are inactive, if *MailBoxCycle1 (94.13)* is set to 0 ms.

Attention:
The communication timeout has to be set at least twice as long as the corresponding mail box cycle time parameter:

Firmware master	Firmware first follower	Firmware 10th follower
MailBoxCycle1 (94.13) = 100	*MailBoxCycle1 (94.13)* = 200	*MailBoxCycle1 (94.13)* = 200

Send and receive values

Each mailbox can transmit / receive up to 4 values depending on the sign of the mailbox node ID number. The master-follower communication usually needs to send 3 values from the master to the followers, thus the follower is completely controlled by the master:

Master parameters (source)

TrmtRecVal1.1 (94.14) = **701** or **704**	*MainCtrlWord (7.01)* or *UsedMCW (7.04)*
TrmtRecVal1.2 (94.15) = **217**	*SpeedRefUsed (2.17)*
TrmtRecVal1.3 (94.16) = **210**	*TorqRef3 (2.10)*
TrmtRecVal1.4 (94.17) = 0	not used

Follower parameters (sinks)

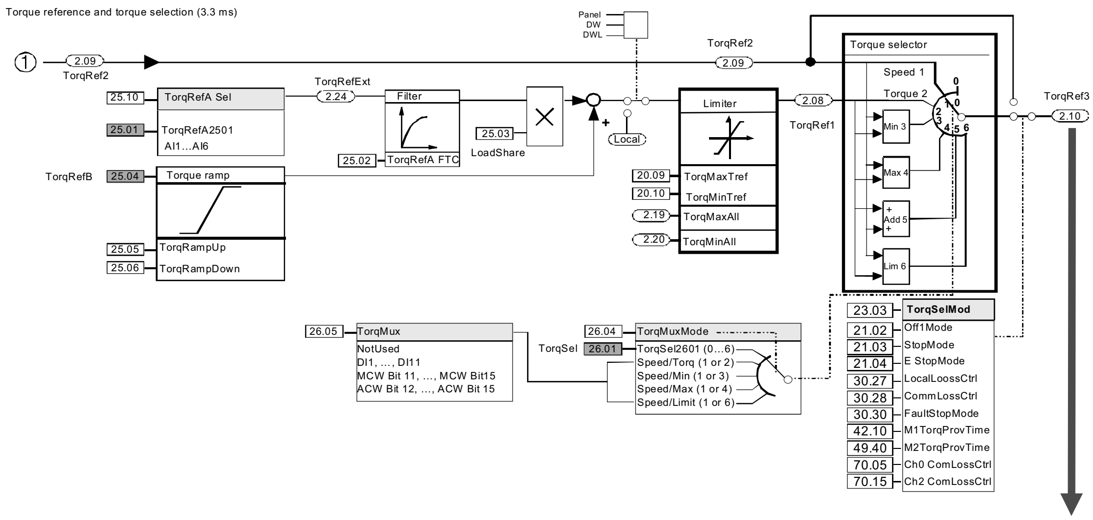
TrmtRecVal1.1 (94.14) = **701**	*MainCtrlWord (7.01)*
TrmtRecVal1.2 (94.15) = **2301**	*SpeedRef (23.01)*
TrmtRecVal1.3 (94.16) = **2501**	*TorqRefA (25.01)*
TrmtRecVal1.4 (94.17) = 0	not used
CommandSel (10.01) = **MainCtrlWord**	
TorqSel (26.01) = **Torque** or **Add**	

Master signal *TorqRef3 (2.10)* is send via master parameter *TrmtRecVal1.3 (94.16)* to
follower signal *TorqRefA (25.01)* via follower parameter *TrmtRecVal1.3 (94.16)*.

Firmware structure

Master:

MailBox1 (94.12) = 5, configures the masters first mailbox to transmit data:

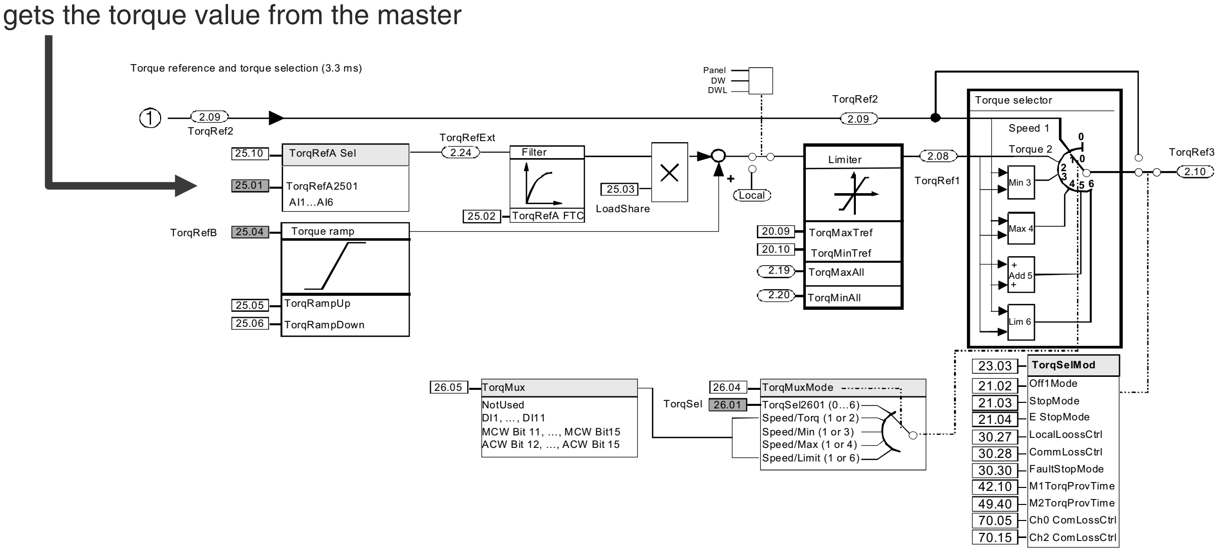


Master parameter *TrmtRecVal1.3 (94.16)* = 210
sends the torque value to the follower

Follower:

MailBox1 (94.12) = -5, configures followers first mailbox to receive data

Follower parameter *TrmtRecVal1.3 (94.16)* = 2501
gets the torque value from the master



For further information consult the *DCS800 Hardware Manual.*

Additional settings

Field weakening:
In case of field weakening all followers must have a speed feedback via encoder, tacho or *MotSpeed (1.04)* - see *M1SpeedFbSel (50.03)* = **External**.
Note:
When connecting the output of one encoder to two drives a splitter has to be used.

Connection to overriding control:
In case followers are connected to an overriding control make sure, that the overriding control is not writing on the same signals (via group 51 and / or group 90) as the master (via the master-follower link). There is always a problem when two sources writing on one sink. Be very carefully with e.g. *MainCtrlWord (7.01)*, *SpeedRef (23.01)*, *TorqRefA (25.01)*, ...

E-stop:
In case of an E-stop the master must be in control of all followers. Thus set:
- *E Stop (10.09)* = **NotUsed** and
- *TorqSelMod (26.03)* = **Fix**

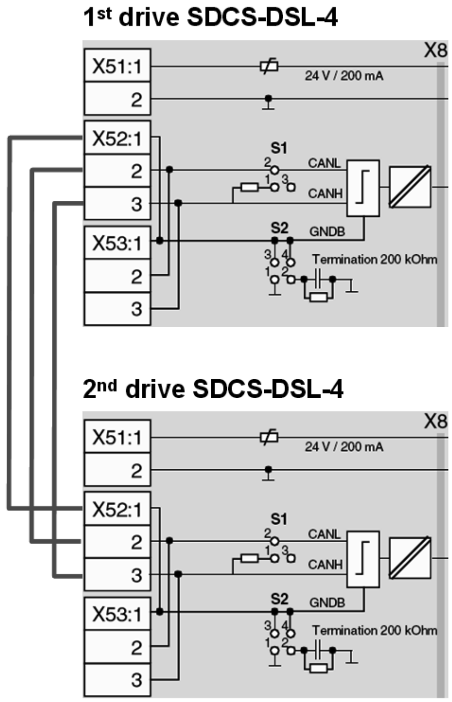
in all followers.

Feedback from the followers to the master:
The feedback from the followers to the master has to be set up manually using drive-to-drive communication and Adaptive Program or application program.

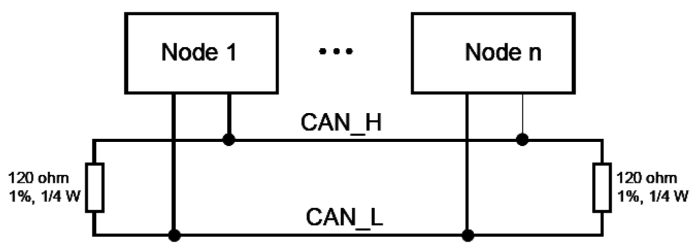
Drive-to-drive communication

Set the DCSLink hardware

Cable connection:



Bus- and ground termination:
The DCSLink is a bus system using twisted pair cables. Therefore bus termination is mandatory at the two physical ends of the bus.



In the above example termination is mandatory at drive 1 and drive 2.

SDCS-DSL-4
jumper **S1** = **1-2** sets the bus termination
jumper **S2** sets the ground termination

Set the node ID numbers and transmission speed

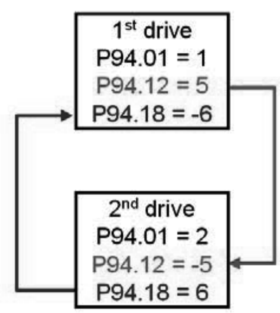
In all bus systems unique node ID numbers are required and have to be set in the master and all followers. Two stations with the same node ID number are not allowed.
For example set the 1st drives node ID number to 1 and the 2nd drives node ID number to 2.
Also the transmission speed of all converters has to match:

Firmware 1st drive	**Firmware 2nd drive**
DCSLinkNodeID (94.01) = 1	*DCSLinkNodeID (94.01)* = 2
BaudRate (94.02) = **500kBit/s**	*BaudRate (94.02)* = **500kBit/s**

Activate the mailboxes

The drive-to-drive communication utilizes 4 mailboxes for data transfer. Thus data transfer to any device / node in the system is possible.
Positive mailbox node ID numbers only transmit data, negative only receive data.
To get communication mailbox node ID pairs (e.g. 5 / -5 and 6 / -6) are needed:

Firmware 1st drive	**Firmware 2nd drive**
MailBox1 (94.12) = 5	*MailBox1 (94.12)* = -5
MailBox2 (94.18) = -6	*MailBox2 (94.18)* = 6



Attention:
Positive mailbox node ID numbers must be unique. Negative mailbox node ID numbers can be used by several mailboxes.

Activate the communication supervision

The communication supervision is activated by means of *MailBoxCycle1 (94.13).* The function of *MailBoxCycle1 (94.13)* is depending on the setting of *MailBox1 (94.12).*
If *MailBox1 (94.12)* is positive:

- data will be transmitted.
- *MailBoxCycle1 (94.13)* sets the transmitting and receiving intervals.
- if *MailBoxCycle1 (94.13)* is set to 3 ms the transmit and receiving intervals are synchronized with mains frequency, either 3.3 ms or 2.77 ms.
- values from 1 - 2 ms are too fast and will generate a fault.
- the communication is inactive, if *MailBoxCycle1 (94.13)* is set to 0 ms.

If MailBox1 (94.12) is negative:

- data will be received.

- *MailBoxCycle1 (94.13)* sets the communication timeout. This is the time delay before a drive-to-drive or master-follower communication break is declared. Depending on the setting of *ComLossCtrl (30.28)* either **F544 P2PandMFCom** [*FaultWord3 (9.03)* bit 11] or **A112 P2PandMFCom** [*AlarmWord1 (9.06)* bit 11] is set.
- the communication fault and alarm are inactive, if *MailBoxCycle1 (94.13)* is set to 0 ms.

Attention:
The communication timeout has to be set at least twice as long as the corresponding mail box cycle time parameter:

Firmware 1st drive	**Firmware 2nd drive**
MailBoxCycle1 (94.13) = 100	*MailBoxCycle1 (94.13)* = 200
MailBoxCycle2 (94.19) = 200	*MailBoxCycle2 (94.19)* = 100

Send and receive values

Each mailbox can transmit / receive up to 4 values depending on the sign of the mailbox node ID number.

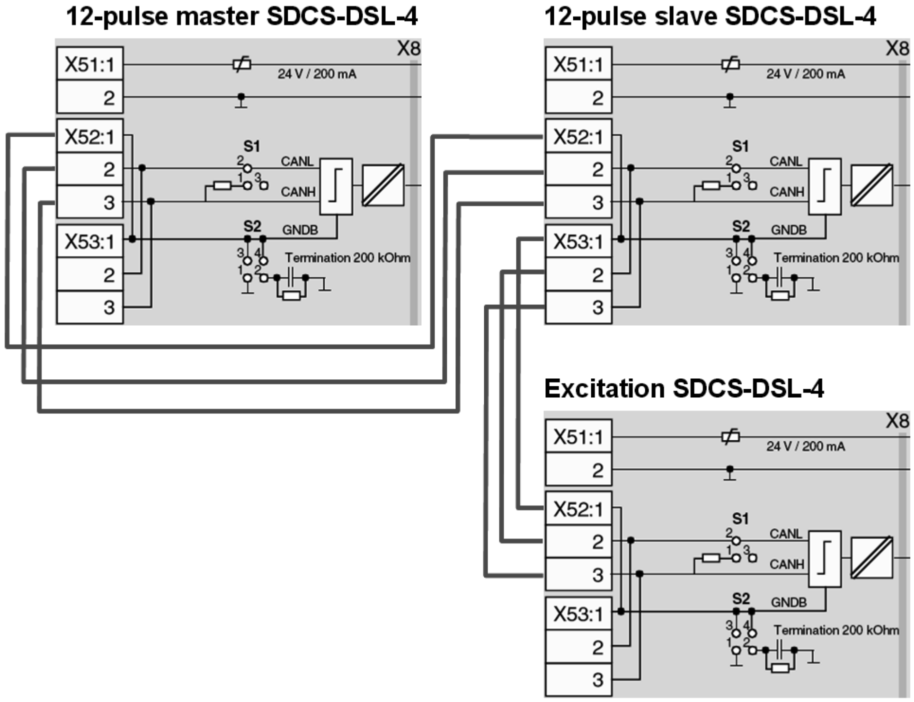
1st mailbox
TrmtRecVal1.1 (94.14)
TrmtRecVal1.2 (94.15)
TrmtRecVal1.3 (94.16)
TrmtRecVal1.4 (94.17)

2nd mailbox
TrmtRecVal2.1 (94.20)
TrmtRecVal2.2 (94.21)
TrmtRecVal2.3 (94.22)
TrmtRecVal2.4 (94.23)

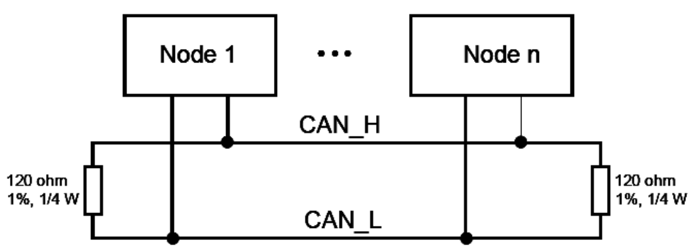
12-pulse

Set the DCSLink hardware

Cable connection:



Bus- and ground termination:
The DCSLink is a bus system using twisted pair cables. Therefore bus termination is mandatory at the two physical ends of the bus.



In the above example termination is mandatory at the 12-pulse master and the excitation.

SDCS-DSL-4
jumper **S1** = **1-2** sets the bus termination
jumper **S2** sets the ground termination

Set the node numbers, transmission speed and the communication supervision

In all bus systems unique node ID numbers are required and have to be set in the 12-pulse master, 12-pulse slave and the excitation. Two stations with the same node ID number are not allowed.
For example set the 12-pulse master node ID number to 1, the 12-pulse slave node ID number to 31 and the excitation node ID number to 21.
The 12-pulse and excitation communication supervision is activated in the 12-pulse master.
Also the transmission speed of all converters has to match:

Firmware 12-pulse master	**Firmware 12-pulse slave**	**Firmware excitation**
DCSLinkNodeID (94.01) = 1	*DCSLinkNodeID (94.01)* = 31	*DCSLinkNodeID (94.01)* = 21
BaudRate (94.02) = **500kBit/s**	*BaudRate (94.02)* = **500kBit/s**	*BaudRate (94.02)* = **500kBit/s**
12P TimeOut (94.03) = 100 ms	-	-
12P SlaNode (94.04) = 31	-	-
FexTimeOut (94.07) = 100 ms	-	-
M1FexNode (94.08) = 21	-	-

DDCS channels with SDCS-COM-8

General

The following table describes the usage of the DDCS channels of the SDCS-COM-8 board.

Channel	Standard usage	SDCS-COM-81	SDCS-COM-82
Ch0	Overriding control or NETA-01 connection	10 Mb (e.g. FCI, AC 800M)	5 Mb (fieldbus adapter)
Ch1	I/O extensions via AIMA board	5 Mb	5 Mb
Ch2	Master-follower link	10 Mb	10 Mb
Ch3	DriveWindow or NETA-01 connection	10 Mb	10 Mb

The communication protocol of Ch0 to Ch3 is DDCS (Distributed Drives Communication System). The Ch0 of the SDCS-COM-8 supports either DDCS or DriveBus, see *Ch0 DriveBus (71.01)*. Both, the DDCS and DriveBus link between the overriding control and the drive, using data sets for information exchange. Each data set is a package of three words (signals or parameters). If a data set is received by the drive the corresponding data set is automatically transmitted to the overriding control as response:

Drive	Received data	Transmitted data
	→→→ data set 10	data set 11 →→→
	→→→ data set 12	data set 13 →→→

The data received from the overriding control affects only the RAM (not FPROM) memory in the drive.

Integer scaling on the DDCS link

Communication between the drive and the overriding control uses 16 bit integer values. The overriding control has to use the information given in integer scaling to be able to change values of parameters properly.
Example1:
If *TorqMaxSPC (20.07)* is written to from the overriding control an integer value of 100 corresponds to 1 % torque.
Example2:
If *SpeedRef (23.01)* is written to from the overriding control 20.000 equals the speed (in rpm) shown in *SpeedScaleAct (2.29)*.

1.08	**MotTorq (motor torque)** Motor torque in percent of *MotNomTorque (4.23)*: – Filtered by means of a 6th order FIR filter (sliding average filter), filter time is 1 mains voltage period. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**

Ch0 communication to overriding control

ABB overriding control

The communication between the overriding control and the SDCS-COM-8 via Ch0 uses data sets. The data sets are connected to the firmware by read- and write pointers - see sections Received data set table and Transmitted data set table. Received and transmitted values are set according to groups 90 to 93. Received data sets are typically connected to *MainCtrlWord (7.01)* and *SpeedRef (23.01)*, whereas transmitted data sets are connected to *MainStatWord (8.01)* and *MotSpeed (1.04)*.

Parameter setting example

The following table lists the parameters which need to be defined when setting up the communication between the drive and ABB overriding control.

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
Ch0 NodeAddr (70.01)	0 - 254	Ch0 node address
Ch0 LinkControl (70.02)	10	Ch0 LED light intensity
Ch0 BaudRate (70.03)	**4 Mbits/s**	for ABB overriding control
Ch0 TimeOut (70.04)	100	Time delay for communication loss detection
Ch0 ComLossCtrl (70.05)	**RampStop**	Reaction to communication loss detection
Ch0 HW Config (70.06)	**Ring** or **Star**	Ch0 topology selection
CH0 DsetBaseAddr (70.24)	10	use either data set range 1 to 16 or data set range 10 to 25
CommModule (98.02)	**COM-8/AC800x**	
Ch0 DriveBus (71.01)	**No** or **Yes**	Ch0 communication mode selection

DCS800 parameter setting for ABB overriding control

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Received data set table

Send from the overriding control to the drive (typical).

Addresses for data received from the overriding control						
Data set number	**Data set index**	**Update time**	**COM-8 ⇒ CON-4**	**Selection parameter**	**Default value**	**Parameter name (default values)**
(70.24) + 0	1	2 ms	*1 ms*	*(90.01)*	701	*MainCtrlWord*
	2	2 ms	*1 ms*	*(90.02)*	2301	*SpeedRef*
	3	2 ms	*1 ms*	*(90.03)*	2501	*TorqRefA*
(70.24) + 2	1	2 ms	*1 ms*	*(90.04)*	702	*AuxCtrlWord*
	2	2 ms	*1 ms*	*(90.05)*	703	*AuxCtrlWord2*
	3	2 ms	*1 ms*	*(90.06)*		
(70.24) + 4	1	2 ms	*1 ms*	*(90.07)*		
	2	2 ms	*1 ms*	*(90.08)*		
	3	2 ms	*1 ms*	*(90.09)*		
(70.24) + 6	1	2 ms	*1 ms*	*(90.10)*		
	2	2 ms	*1 ms*	*(90.11)*		
	3	2 ms	*1 ms*	*(90.12)*		
(70.24) + 8	1	10 ms	*20 ms*	*(90.13)*		
	2	10 ms	*20 ms*	*(90.14)*		
	3	10 ms	*20 ms*	*(90.15)*		
(70.24) + 10	1	10 ms	*20 ms*	*(90.16)*		
	2	10 ms	*20 ms*	*(90.17)*		
	3	10 ms	*20 ms*	*(90.18)*		
(70.24) + 12	1	10 ms	*20 ms*	*(91.01)*		
	2	10 ms	*20 ms*	*(91.02)*		
	3	10 ms	*20 ms*	*(91.03)*		
(70.24) + 14	1	10 ms	*20 ms*	*(91.04)*		
	2	10 ms	*20 ms*	*(91.05)*		
	3	10 ms	*20 ms*	*(91.06)*		

Note:
The update time is the time within the drive is reading values from the data sets. Since the drive is a communication slave, the actual cycle time depends on the cycle time of the communication master.

Transmitted data set table

Send from the drive to the overriding control (typical).

Addresses for data transmitted to the overriding control						
Data set number	**Data set index**	**Update time**	**CON-4 ⇒ COM-8**	**Selection parameter**	**Default value**	**Parameter name (default values)**
(70.24) + 1	1	2 ms	*1 ms*	*(92.01)*	801	*MainStatWord*
	2	2 ms	*1 ms*	*(92.02)*	104	*MotSpeed*
	3	2 ms	*1 ms*	*(92.03)*	209	*TorqRef2*
(70.24) + 3	1	2 ms	*1 ms*	*(92.04)*	802	*AuxStatWord*
	2	2 ms	*1 ms*	*(92.05)*	101	*MotSpeedFilt*
	3	2 ms	*1 ms*	*(92.06)*	108	*MotTorq*
(70.24) + 5	1	2 ms	*1 ms*	*(92.07)*	901	*FaulWord1*
	2	2 ms	*1 ms*	*(92.08)*	902	*FaulWord2*
	3	2 ms	*1 ms*	*(92.09)*	903	*FaulWord3*
(70.24) + 7	1	2 ms	*1 ms*	*(92.10)*	904	*FaulWord4*
	2	2 ms	*1 ms*	*(92.11)*	906	*AlarmWord1*
	3	2 ms	*1 ms*	*(92.12)*	907	*AlarmWord2*
(70.24) + 9	1	10 ms	*20 ms*	*(92.13)*	908	*AlarmWord3*
	2	10 ms	*20 ms*	*(92.14)*	803	*LimWord*
	3	10 ms	*20 ms*	*(92.15)*	805	*DI StatWord*
(70.24) + 11	1	10 ms	*20 ms*	*(92.16)*	806	*DO StatWord*
	2	10 ms	*20 ms*	*(92.17)*	124	*BridgeTemp*
	3	10 ms	*20 ms*	*(92.18)*	122	*Mot1TempMeas*
(70.24) + 13	1	10 ms	*20 ms*	*(93.01)*		
	2	10 ms	*20 ms*	*(93.02)*		
	3	10 ms	*20 ms*	*(93.03)*		
(70.24) + 15	1	10 ms	*20 ms*	*(93.04)*		
	2	10 ms	*20 ms*	*(93.05)*		
	3	10 ms	*20 ms*	*(93.06)*		

Note:
The update time is the time within the drive is reading values from the data sets. Since the drive is a slave, the actual communication cycle time depends on the master's cycle time.

Fieldbus communication (N-type)

The communication between the N-type fieldbus adapter and the SDCS-COM-8 uses data sets. The data set base address is set by means of *CH0 DsetBaseAddr (70.24)* = 1. The communication for the fieldbus adapters is activated by means of *CommModule (98.02)* = **COM-8/Nxxx**. The contents of the fieldbus data sets is set by means of the same pointers as for the ABB overriding control data sets - see sections Received data set table and Transmitted data set table. Received and transmitted values are set according to groups 90 to 93. Also the update times are the same.

Ch1 I/O devices

All optional I/O devices are connected via AIMA-01 board to Ch1. The SDCS-COM-8 is the master in the communication link. Each device has an individual address, set with switches on the I/O device. Before use, each I/O device must be activated by means of a parameter in group 98.
See also:
I/O Module Adapter AIMA-01; User's Manual

Ch2 Master-follower link

General

The master-follower link is designed for applications in which the system is operated by several drives and the shafts are coupled to each other via gearing, chains, belts etc. The master controls all followers via a fiber optic serial communication link. Pulse encoders are recommended for the master and all followers.

The master is typically speed controlled and the other drives follow the master's torque or speed reference. In general, torque control or window control of the followers should be used when the motor shafts of the master and the followers drives are fixed coupled to each other via gearing, chains, belts etc. and no speed differences between the drives is possible.

Link configuration

Ch2 on the SDCS-COM-8 board is used for the master-follower link between the drives. Ch2 is configurable by *Ch2 MaFoMode (70.09)* either to be master or follower in the communication in broadcast mode. Typically the speed controlled process master drive is configured also to be the communication master.

Master

The master mode is selected by *Ch2 MaFoMode (70.09)*. The torque reference source address is defined in the master by *Ch2 MasSig3 (70.12)* to be sent via broadcast to the followers. Also two other signals can be sent through the link if required. Their addresses are defined by *Ch2 MasSig1 (70.10)* and *Ch2 MasSig2 (70.11)*. Typical / default addresses are:

Signal addresses in the master		
Update time	**Parameter name and index of the default values**	**Master drive selection parameters**
2 ms	*MainCtrlWord (7.01)* or *UsedMCW (7.04)*	*Ch2 MasSig1 (70.10)*
2 ms	*SpeedRefUsed (2.17)*	*Ch2 MasSig2 (70.11)*
2 ms	*TorqRef3 (2.10)*	*Ch2 MasSig3 (70.12)*

Above parameters are not valid in the follower. The master cyclically sends *Ch2 MasSig1 … 3* in one DDCS message as broadcast every 2 ms.

Followers

The follower mode is selected by *Ch2 MaFoMode (70.09)*. To control start and stop from the master set *CommandSel (10.01)* = **MainCtrlWord**. The connections are selected by *Ch2 FolSig1 (70.18)*, *Ch2 FolSig2 (70.19)* and *Ch2 FolSig3 (70.20)* according to the following table:

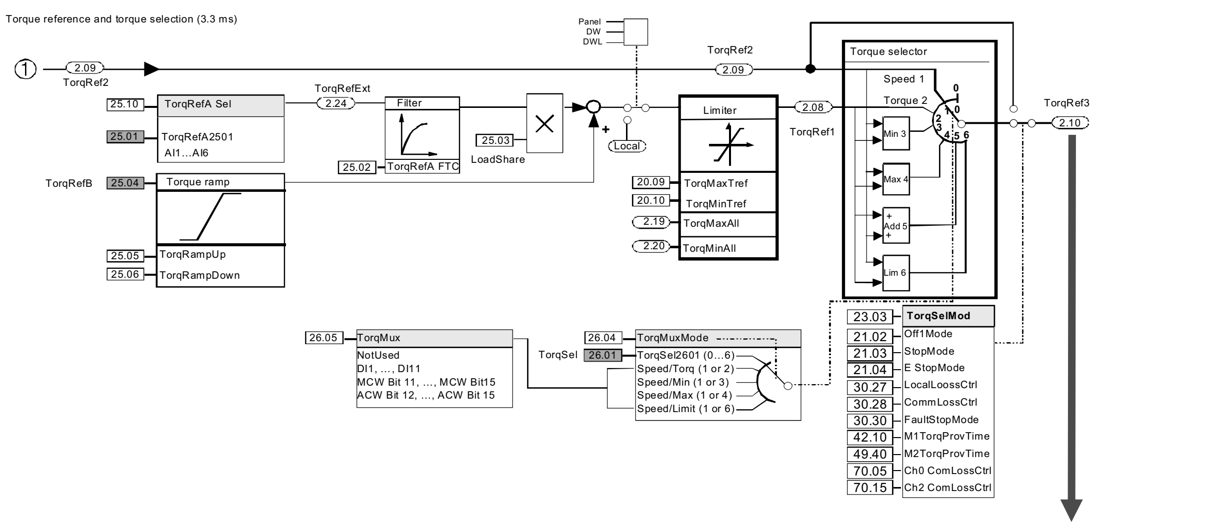
Signal addresses in the follower		
Update time	**Parameter name and index of the default values**	**Follower drive selection parameters**
2 ms	*MainCtrlWord (7.01)*	*Ch2 FolSig1 (70.18)*
2 ms	*SpeedRef (23.01)*	*Ch2 FolSig2 (70.19)*
2 ms	*TorqRefA (25.01)*	*Ch2 FolSig3 (70.20)*

Above parameters are not valid in the master. The follower cyclically reads *Ch2 FolSig1 … 3* every 2 ms.

Note:
In default setting master signal *TorqRef3 (2.10)* is send via master parameter *Ch2 MasSig3 (70.12)* to follower signal *TorqRefA (25.01)* via follower parameter *Ch2 FolSig3 (70.20)*.

Firmware structure

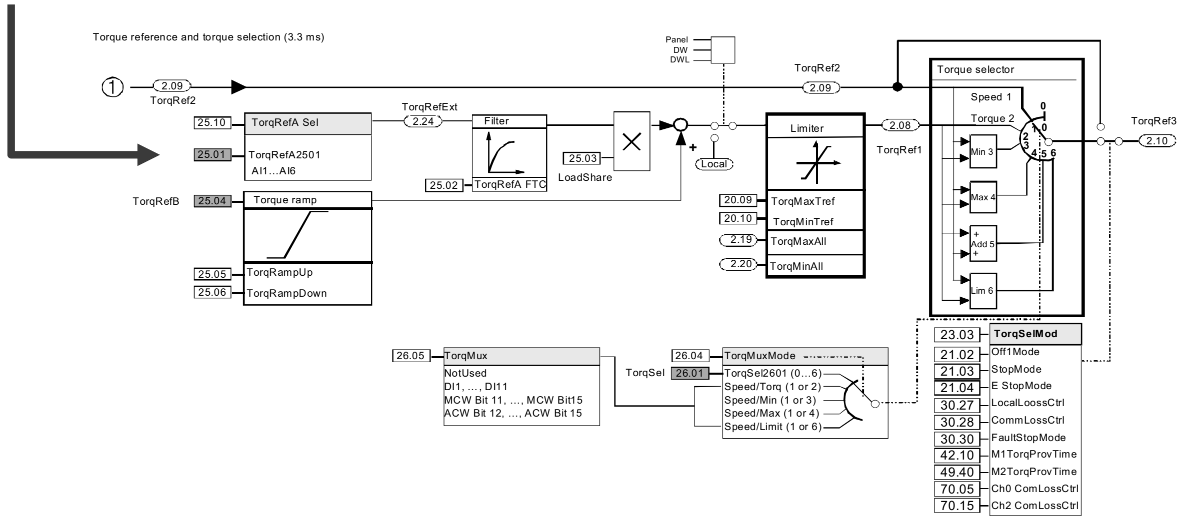
Master:
Ch2 MaFoMode (70.09) = **Master**, activates read pointer *Ch2 MasSig1 (70.10)*, *Ch2 MasSig2 (70.11)* and *Ch2 MasSig3 (70.12)*



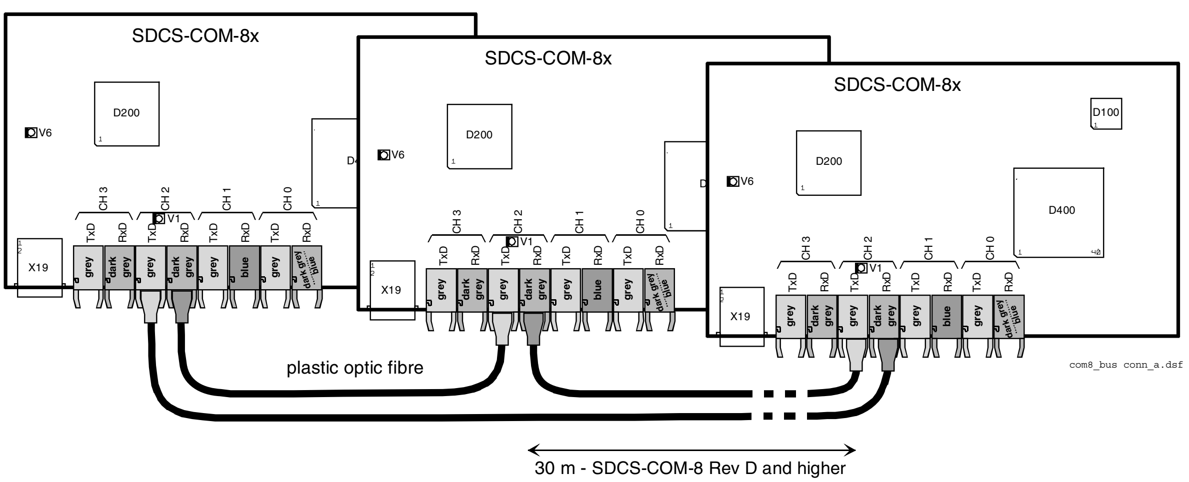
Master parameter *Ch2 MasSig3 (70.12)* = 210
sends the torque value to the follower

Follower:
Ch2 MaFoMode (70.09) = **Follower**, activates write pointer *Ch2 FolSig1 (70.18)*, *Ch2 FolSig2 (70.19)* and *Ch2 FolSig3 (70.20)*

Follower parameter *Ch2 FolSig3 (70.20)* = 2501
gets the torque value from the master



Master-follower firmware structure



Master-follower fiber optic cable connection (see also DCS800 Hardware Manual)

Toggle between speed- and torque control

In some application, both speed- and torque control of the followers are required, e.g. if it is necessary to accelerate all drives along the same speed ramp up to a certain speed before the torque control can be started. In those cases, a flying switch over between speed- and torque controls is required. The switch over can be done by e.g. the overriding control using *TorqSel (26.01).* See also *TorqMux (26.05)* and *TorqMuxMode (26.04).*

Follower diagnostics

All the followers receive the torque reference via *TorqRefA (25.01)*. All followers are able to detect communication breaks, after the first valid message is received. The action due to a communication break is defined by *Ch2 TimeOut (70.14)* and *Ch2 ComLossCtrl (70.15)*. Feedback for all alarms and faults from the followers must be handled by the overriding control through the Ch0 on the SDCS-COM-8 board.

Master-follower link specification

Size of the link: One master and maximum ten followers are allowed. If more than ten followers are required, a local ABB agent should be consulted.

Configuration: Link is configurable by the overriding control using *Ch2 MaFoMode (70.09)*. This makes possible to change between master and follower by the overriding control without changes in the hardware.

Transmission rate: 4 Mbit/s

Total performance of the link: 2 ms (between master and followers)

Protocol: Distributed Drives Communication System, DDCS

Ch3 commissioning and maintenance tools

DriveWindow

DriveWindow can be connected to Ch3 in either ring (max. 5 drives) or star connection using NDBU-xx branching units, see *Ch3 HW Config (70.21)*. The node numbers - *Ch3 NodeAddr (70.32)* - must be set for each drive individually before starting the communication through the connection. This setting has to be made by a point to point connection using the DCS800 Control Panel, DriveWindow or DriveWindow Light. The new node address becomes valid after the next SDCS-COM-8 power-up. The SDCS-COM-8 Ch3 has been configured to be a slave in the communication point of view. With *DeviceName (99.09)* and DriveWindow it is possible to fill in a string (name) with a maximum of 12 characters for individual drive identification. See also:

Configuration Instructions NDBU-85/95; 3ADW000100,
Optical DDCS Communication Link; 3BFE64285513 and
DDCS Cabling and Branching; 3AFE63988235

Ethernet communication for monitoring with Ethernet adapter NETA-01

General

This chapter gives information using the Ethernet adapter NETA-01 together with the DCS800.

NETA-01 - DCS800

The Ethernet communication for monitoring with the drive requires the options NETA-01 and SDCS-COM-8.
The NETA-01 is connected to the SDCS-COM-8 usually via Ch3. Ch0 can be used as well.
Following browser based remote monitoring functions are released for DC-drives:

- Parameters — Read and write parameters
- Signals — Read signals
- Fault logger — Show fault logger
  Clear fault logger
  Save faults to a file in the NETA-01
  Download saved fault logger files via FTP
- Data logger — Select values and set all trigger conditions
  Upload samples and show as values or as graphs
  Save samples as files in the NETA-01
  Download saved data logger files via FTP
- Status word — *MainStatWord (8.01)* is shown after clicking on the lamp
  Note:
  Bit 11 (EXT_CTRL_LOC) and bit 12 (RUN_ENABLE) are not used for DC-drives

Note:
Data set communication and motor control (e.g. local control of the drives via NETA-01) are not released for the DCS800.

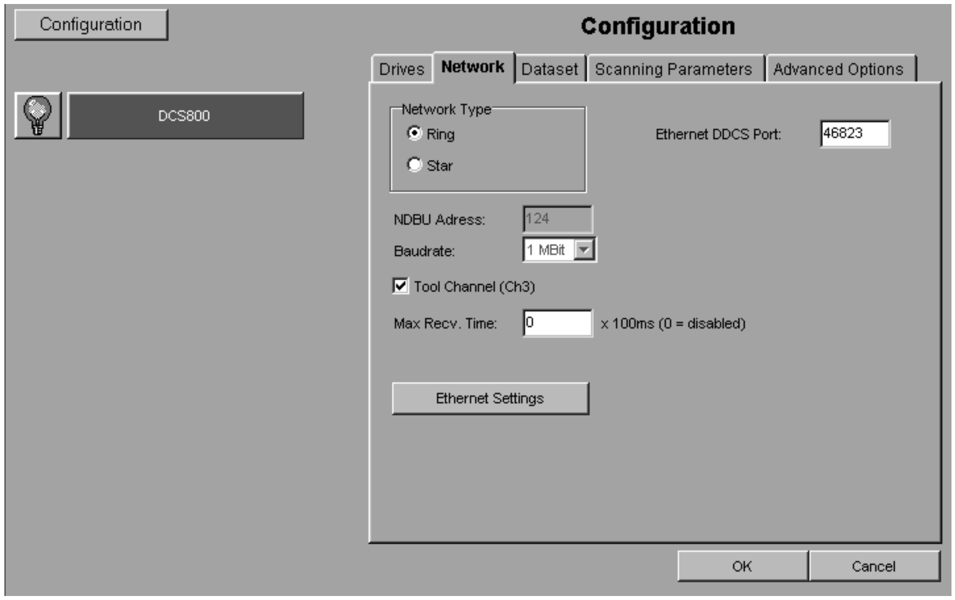
Related documentation

User's Manual Ethernet Adapter Module NETA-01.

The quoted page numbers correspond to the User's Manual.

NETA-01 configuration

The NETA-01 homepage can be called by using a browser (e.g. internet explorer).
Note:
Before connecting the NETA-01 via Ch3 with the DCS800 check, that *Tool Channel (Ch3)* of the NETA-01 configuration is ticked otherwise group 51 (Fieldbus) will be overwritten.
Note:
When connecting the NETA-01 with the DCS800 make sure to use Ch3 (tool channel) on the SDCS-COM-8, otherwise group 51 (Fieldbus) will be overwritten. Ch0 can be used too, but then group 51 (Fieldbus) will be overwritten and cannot be used for other serial communication.



More details about the NETA-01 configuration see page 55 of the User's Manual.

Mechanical and electrical installation

The adapter module is mounted onto a standard mounting rail outside the drive.

Drive configuration

The DCS800 needs no special settings when using Ch3 concerning the released functions.

Firmware compatibility:

SDCS-CON-4: firmware version 1.8 or higher, see *FirmwareVer (4.01)*
SDCS-COM-8: firmware version 1.3 or higher, see *Com8SwVersion (4.11)*

CANopen communication with fieldbus adapter RCAN-01

General

This chapter gives additional information using the CANopen adapter RCAN-01 together with the DCS800.

RCAN-01 - DCS800

The CANopen communication with the drive requires the option RCAN-01.

Related documentation

User's Manual CANopen Adapter Module RCAN-01.

The quoted page numbers correspond to the User's Manual.

Overriding control configuration

Supported operation mode is **PDO21** (see page 43 and 44).

EDS file

The EDS file for RCAN-01 and DCS800 is available. Please ask Your local ABB agent for the newest one concerning the current DCS800 firmware.

Mechanical and electrical installation

If not already done so insert RCAN-01 into slot 1 of the drive.

Drive configuration

The CANopen adapter is activated by means of *CommModule (98.02)*.
Please note that the DCS800 works with the operation mode **PDO21** (see page 43 and 44).

Parameter setting example 1 using group 51

Communication via group 51 is using 4 data words in each direction. The following table shows the parameter setting using group 51.

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Fieldbus**	

ModuleType (51.01)	**CANopen***	
Node ID (51.02)	1**	set node address as required
Baudrate (51.03)	8**	8 = 1 MBits/s
PDO21 Cfg (51.04)	1	0 = Configuration via CANopen objects 1 = Configuration via RCAN-01 adapter parameters
RX-PDO21-Enable (51.05)	769	This value has to be calculated with 300 Hex = 768 + *Node ID (51.02)*. Here 768 + 1 = 769

RX-PDO21-TxType (51.06)	255	255 = **Asynchronous** (see page 83)
RX-PDO21-1stObj (51.07)	8197	2005 Hex = 8197 = **Transparent Control Word** (see page 62)
RX-PDO21-1stSubj (51.08)	0	
RX-PDO21-2ndObj (51.09)	8198	2006 Hex = 8198 = **Transparent Reference Speed** (see page 62)
RX-PDO21-2ndSubj (51.10)	0	
RX-PDO21-3rdObj (51.11)	16409	This value has to be calculated with 4000 Hex = 16384 + parameter group number. E.g. with *TorqRefA (25.01)* follows 16384 + 25 = 16409 (see page 64)
RX-PDO21-3rdSubj (51.12)	1	This value has to be taken from the parameters index. E.g. with *TorqRefA (25.01)* follows 1 (see page 64)
RX-PDO21-4thObj (51.13)	16391	This value has to be calculated with 4000 Hex = 16384 + parameter group number. E.g. with *AuxCtrlWord (7.02)* follows 16384 + 7 = 16391 (see page 64)
RX-PDO21-4thSubj (51.14)	2	This value has to be taken from the parameters index. E.g. with *AuxCtrlWord (7.02)* follows 2 (see page 64)
TX-PDO21-Enable (51.15)	641	This value has to be calculated with 280 Hex = 640 + *Node ID (51.02)*. Here 640 + 1 = 641
TX-PDO21-TxType (51.16)	255	255 = **Asynchronous** (see page 83)
TX-PDO21-EvTime (51.17)	10	10 = 10 ms
TX-PDO21-1stObj (51.18)	8199	2007 Hex = 8199 = **Transparent Status Word** (see page 62)
TX-PDO21-1stSubj (51.19)	0	
TX-PDO21-2ndObj (51.20)	8200	2008 Hex = 8200 = **Transparent Actual Speed** (see page 62)
TX-PDO21-2ndSubj (51.21)	0	

TX-PDO21-3rdObj (51.22)	16386	This value has to be calculated with 4000 Hex = 16384 + parameter group number. E.g. with *TorqRef2 (2.09)* follows 16384 + 2 = 16386 (see page 64)
TX-PDO21-3rdSubj (51.23)	9	This value has to be taken from the parameters index. E.g. with *TorqRef2 (2.09)* follows 9 (see page 64)
TX-PDO21-4thObj (51.24)	16392	This value has to be calculated with 4000 Hex = 16384 + parameter group number. E.g. with *AuxStatWord (8.02)* follows 16384 + 8 = 16392 (see page 64)
TX-PDO21-4thSubj (51.25)	2	This value has to be taken from the parameters index. E.g. with *AuxStatWord (8.02)* follows 2 (see page 64)
TransparentIProfil (51.26)	1	1 = **Transparent**
FBA PAR REFRESH (51.27)	**DONE**, default	If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.

* Read-only or automatically detected by CANopen adapter
** The values can be automatically set via the rotary switches of the RCAN-01

DCS800 parameter setting using group 51

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Further information

RX and TX parameters 51.07, …, 51.14 and 51.18, …, 51.25 are directly connected to the desired DCS800 parameters. Take care, that the used parameters are deleted from group 90 and 92 to prevent data trouble.

Parameter setting example 2 using groups 90 and 92

Communication via groups 90 and 92 is using 4 data words in each direction. The following table shows the parameter setting using groups 90 and 92.

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Fieldbus**	

DsetXVal1 (90.01)	701, default	*MainCtrlWord (7.01)*; output data word 1 (control word) 1st data word from overriding control to drive
DsetXVal2 (90.02)	2301, default	*SpeedRef (23.01)*; output data word 2 (speed reference) 2nd data word from overriding control to drive
DsetXVal3 (90.03)	2501, default	*TorqRefA (25.01)*; output data word 3 (torque reference) 3rd data word from overriding control to drive
DsetXplus2Val1 (90.04)	702, default	*AuxCtrlWord (7.02)*; output data word 4 (auxiliary control word) 4th data word from overriding control to drive
DsetXplus1Val1 (92.01)	801, default	*MainStatWord (8.01)*; input data word 1 (status word) 1st data word from drive to overriding control
DsetXplus1Val2 (92.02)	104, default	*MotSpeed (1.04)*; input data word 2 (speed actual) 2nd data word from drive to overriding control
DsetXplus1Val3 (92.03)	209, default	*TorqRef2 (2.09)*; input data word 3 (torque reference) 3rd data word from drive to overriding control
DsetXplus3Val1 (92.04)	802, default	*AuxStatWord (8.02)*; input data word 4 (auxiliary status word) 4th data word from drive to overriding control

ModuleType (51.01)	**CANopen***	
Node ID (51.02)	1**	set node address as required
Baudrate (51.03)	8**	8 = 1 MBits/s
PDO21 Cfg (51.04)	1	0 = Configuration via CANopen objects 1 = Configuration via RCAN-01 adapter parameters

RX-PDO21-Enable (51.05)	769	This value has to be calculated with 300 Hex = 768 + *Node ID (51.02)*. Here 768 + 1 = 769
RX-PDO21-TxType (51.06)	255	255 = **Asynchronous** (see page 83)
RX-PDO21-1stObj (51.07)	16384	4000 Hex = 16384 = **Control Word** (see page 63); Data set 1 word 1
RX-PDO21-1stSubj (51.08)	1	1 Hex = 1 = **Control Word** (see page 63); Data set 1 word 1
RX-PDO21-2ndObj (51.09)	16384	4000 Hex = 16384 = **Reference 1** (see page 63); Data set 1 word 2
RX-PDO21-2ndSubj (51.10)	2	2 Hex = 2 = **Reference 1** (see page 63); Data set 1 word 2
RX-PDO21-3rdObj (51.11)	16384	4000 Hex = 16384 = **Reference 2** (see page 63); Data set 1 word 3
RX-PDO21-3rdSubj (51.12)	3	3 Hex = 3 **Reference 2** (see page 63); Data set 1 word 3
RX-PDO21-4thObj (51.13)	16384	4000 Hex = 16384 = **Reference 3** (see page 63); Data set 3 word 1
RX-PDO21-4thSubj (51.14)	7	7 Hex = 7 **Reference 3** (see page 63); Data set 3 word 1
TX-PDO21-Enable (51.15)	641	This value has to be calculated with 280 Hex = 640 + *Node ID (51.02)*. Here 640 + 1 = 641
TX-PDO21-TxType (51.16)	255	255 = **Asynchronous** (see page 83)
TX-PDO21-EvTime (51.17)	10	10 = 10 ms
TX-PDO21-1stObj (51.18)	16384	4000 Hex = 16384 = **Status Word** (see page 63); Data set 2 word 1
TX-PDO21-1stSubj (51.19)	4	4 Hex = 4 = **Status Word** (see page 63); Data set 2 word 1
TX-PDO21-2ndObj (51.20)	16384	4000 Hex = 16384 = **Actual Value 1** (see page 63); Data set 2 word 2
TX-PDO21-2ndSubj (51.21)	5	5 Hex = 5 = **Actual Value 1** (see page 63); Data set 2 word 2

TX-PDO21-3rdObj (51.22)	16384	4000 Hex = 16384 = **Actual Value 2** (see page 63); Data set 2 word 3
TX-PDO21-3rdSubj (51.23)	6	6 Hex = 6 = **Actual Value 2** (see page 63); Data set 2 word 3
TX-PDO21-4thObj (51.24)	16384	4000 Hex = 16384 = **Actual Value 3** (see page 63); Data set 4 word 1
TX-PDO21-4thSubj (51.25)	10	A Hex = 10 = **Actual Value 3** (see page 63); Data set 4 word 1
TransparentIProfil (51.26)	1	1 = **Transparent**
FBA PAR REFRESH (51.27)	**DONE**, default	If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.

* Read-only or automatically detected by CANopen adapter
** The values can be automatically set via the rotary switches of the RCAN-01

DCS800 parameter setting using groups 90 and 92

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Switch on sequence

Please see the example at the end of this chapter.

ControlNet communication with fieldbus adapter RCNA-01

General

This chapter gives additional information using the ControlNet adapter RCNA-01 together with the DCS800.

RCNA-01 - DCS800

The ControlNet communication with the drive requires the option RCNA-01.

Related documentation

User's Manual ControlNet Adapter Module RCNA-01.

The quoted page numbers correspond to the User's Manual.

Overriding control configuration

Please refer to the Scanner documentation for information how to configure the system for communication with RCNA-01.

EDS file

The EDS file for RCNA-01 and DCS800 is available. Please ask Your local ABB agent for the newest one concerning the current DCS800 firmware.

Mechanical and electrical installation

If not already done so insert RCNA-01 into slot 1 of the drive (see page 17).

Drive configuration

The ControlNet adapter is activated by means of *CommModule (98.02)*.
Please note that the DCS800 works with the instances **User transparent assembly** and **Vendor specific assembly**.
The instances **Basic speed control** and **Extended speed control** (instance 20 / 70 and 21 / 71) are supported since firmware version 2.x. With these instances it is not possible to use the full flexibility of the DCS800.
For more information see User's Manual.

Parameter setting example 1 using ABB Drives assembly

ABB Drives assembly is using 2 data words in each direction. The following table shows the parameter setting using this profile.

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Fieldbus**	

DsetXVal1 (90.01)	701, default	*MainCtrlWord (7.01)*; output data word 1 (control word) 1st data word from overriding control to drive

DsetXVal2 (90.02)	2301, default	*SpeedRef (23.01)*; output data word 2 (speed reference) 2nd data word from overriding control to drive
DsetXplus1Val1 (92.01)	801, default	*MainStatWord (8.01)*; input data word 1 (status word) 1st data word from drive to overriding control
DsetXplus1Val2 (92.02)	104, default	*MotSpeed (1.04)*; input data word 2 (speed actual) 2nd data word from drive to overriding control

ModuleType (51.01)	**CONTROLNET***	
Module macid (51.02)	4**	set node address as required
Module baud rate (51.03)	2**	2 = 500 kBits/s
HW/SW option (51.04)	0	0 = **Hardware** 1 = **Software**
Stop function (51.05)	NA	not applicable when using **ABB Drives assembly**
Output instance (51.06)	100	100 = **ABB Drives assembly**
Input instance (51.07)	101	101 = **ABB Drives assembly**
Output I/O par 1 (51.08) to *Input I/O par 9 (51.25)*	NA	not applicable when using **ABB Drives assembly**
VSA I/O size (51.26)	NA	not applicable when using **ABB Drives assembly**
FBA PAR REFRESH (51.27)	**DONE**, default	If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.

* Read-only or automatically detected by ControlNet adapter.
** If *HW/SW option (51.04)* = 0 (**Hardware**), the values are automatically set via the rotary switches of the RCNA-01.

DCS800 parameter setting using ***ABB Drives assembly***

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Parameter setting example 2 using Vendor specific assembly

Vendor specific assembly can run with up to 9 data words in each direction. The following table shows the parameter setting using this profile.

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Fieldbus**	
ModuleType (51.01)	**CONTROLNET***	
Module macid (51.02)	4**	set node address as required
Module baud rate (51.03)	5	5 = 5 MBits/s
HW/SW option (51.04)	0	0 = **Hardware** 1 = **Software**
Stop function (51.05)	NA	not applicable when using **Vendor specific assembly**
Output instance (51.06)	102	102 = **Vendor specific assembly**
Input instance (51.07)	103	103 = **Vendor specific assembly**
Output I/O par 1 (51.08) to *Input I/O par 9 (51.25)*	1 - 18	Set these values according table: Setting of parameter groups 51, 90 and 92 depending on desired data words and according to the desired numbers of data words
VSA I/O size (51.26)	1 - 9	Defines the length of the **Vendor specific assembly** in pairs of data words. E.g. a parameter value of 4 means 4 word as output and 4 words as input.
FBA PAR REFRESH (51.27)	**DONE**, default	If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.

* Read-only or automatically detected by ControlNet adapter

** If *HW/SW option (51.04)* = 0 (**Hardware**), the values are automatically set via the rotary switches of the RCNA-01

*DCS800 parameter setting using **Vendor specific assembly***

Note:

± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Setting of parameter groups 51, 90 and 92

Parameter group 51				Direction	ABB	Parameter group 90 and 92			
	name	set value		PLC<->Drive	Datasets		name	def. value	
51.08	Output I/O par 1	=	**1***	⇨	1,1	90.01	DsetXVal1	=	701
51.09	Output I/O par 2	=	**2***	⇨	1,2	90.02	DsetXVal2	=	2301
51.10	Output I/O par 3	=	**3**	⇨	1,3	90.03	DsetXVal3	=	2501
51.11	Output I/O par 4	=	**7**	⇨	3,1	90.04	DsetXplus2Val1	=	702
51.12	Input I/O par 1	=	**4***	⇦	2,1	92.01	DsetXplus1Val1	=	801
51.13	Input I/O par 2	=	**5***	⇦	2,2	92.02	DsetXplus1Val2	=	104
51.14	Input I/O par 3	=	**6**	⇦	2,3	92.03	DsetXplus1Val3	=	209
51.15	Input I/O par 4	=	**10**	⇦	4,1	92.04	DsetXplus3Val1	=	802
51.16	Output I/O par 5	=	**8**	⇨	3,2	90.05	DsetXplus2Val2	=	703
51.17	Output I/O par 6	=	**9**	⇨	3,3	90.06	DsetXplus2Val3	=	0
51.18	Output I/O par 7	=	**13**	⇨	5,1	90.07	DsetXplus4Val1	=	0
51.19	Output I/O par 8	=	**14**	⇨	5,2	90.08	DsetXplus4Val2	=	0
51.20	Output I/O par 9	=	**15**	⇨	5,3	90.09	DsetXplus4Val3	=	0
51.21	Input I/O par 5	=	**11**	⇦	4,2	92.05	DsetXplus3Val2	=	101
51.22	Input I/O par 6	=	**12**	⇦	4,3	92.06	DsetXplus3Val3	=	108
51.23	Input I/O par 7	=	**16**	⇦	6,1	92.07	DsetXplus5Val1	=	901
51.24	Input I/O par 8	=	**17**	⇦	6,2	92.08	DsetXplus5Val2	=	902
51.25	Input I/O par 9	=	**18**	⇦	6,3	92.09	DsetXplus5Val3	=	903

*For proper communication shown values have to be used

Setting of parameter groups 51, 90 and 92 depending on desired data words

Further information

Output and input parameters 51.08, …, 51.25 can also be connected directly to the desired DCS800 parameters. In this case please take care that the RCNA-01 adapter gets the changed values and also take care, that the used parameters are deleted from group 90 to prevent data trouble.

Switch on sequence

Please see the example at the end of this chapter.

DeviceNet communication with fieldbus adapter RDNA-01

General

This chapter gives additional information using the DeviceNet adapter RDNA-01 together with the DCS800.

RDNA-01 - DCS800

The DeviceNet communication with the drive requires the option RDNA-01.

Related documentation

User's Manual DeviceNet Adapter Module RDNA-01.

The quoted page numbers correspond to the User's Manual.

Overriding control configuration

Supported assemblies with DCS800 are **ABB Drives assembly** (Output instance: 100; Input instance: 101) and **User specific assembly** (Output instance: 102; Input instance: 103) (see page 35).
The assemblies **Basic speed control** and **Extended speed control** (20 / 70 and 21 / 71) are supported since DCS800 firmware version 2.x.

EDS file

The EDS file for RDNA-01 and DCS800 is available. Please ask Your local ABB agent for the newest one concerning the current DCS800 firmware.

Mechanical and electrical installation

If not already done so insert RDNA-01 into slot 1 of the drive (see page 21).

Drive configuration

The DeviceNet adapter is activated by means of *CommModule (98.02)*.
Please note that the DCS800 works with the instances **ABB Drives assembly** and **User specific assembly**.
The instances **Basic speed control** and **Extended speed control** (20 / 70 and 21 / 71) are supported since firmware version 2.x. With these instances it is not possible to use the full flexibility of the DCS800.
For more information see User's Manual.

Parameter setting example 1 using ABB Drives assembly

ABB Drives assembly is using 2 data words in each direction. The following table shows the parameter setting using this profile.

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Fieldbus**	

DsetXVal1 (90.01)	701, default	*MainCtrlWord (7.01)*; output data word 1 (control word) 1st data word from overriding control to drive
DsetXVal2 (90.02)	2301, default	*SpeedRef (23.01)*; output data word 2 (speed reference) 2nd data word from overriding control to drive
DsetXplus1Val1 (92.01)	801, default	*MainStatWord (8.01)*; input data word 1 (status word) 1st data word from drive to overriding control
DsetXplus1Val2 (92.02)	104, default	*MotSpeed (1.04)*; input data word 2 (speed actual) 2nd data word from drive to overriding control

ModuleType (51.01)	**DEVICENET***	
Module macid (51.02)	4**	set node address as required
Module baud rate (51.03)	2**	2 = 500 kBits/s
HW/SW option (51.04)	0	0 = **Hardware** 1 = **Software**
Stop function (51.05)	NA	not applicable when using **ABB Drives assembly**
Output instance (51.06)	100	100 = **ABB Drives assembly**
Input instance (51.07)	101	101 = **ABB Drives assembly**
Output I/O par 1 (51.08) to *Input I/O par 9 (51.25)*	NA	not applicable when using **ABB Drives assembly**
VSA I/O size (51.26)	NA	not applicable when using **ABB Drives assembly**
FBA PAR REFRESH (51.27)	**DONE**, default	If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.

* Read-only or automatically detected by DeviceNet adapter
** If *HW/SW option (51.04)* = 0 (**Hardware**), the values are automatically set via the DIP switches of the RDNA-01

*DCS800 parameter setting using **ABB Drives assembly***

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Parameter setting example 2 using User specific assembly

User specific assembly can run with up to 9 data words in each direction. The following table shows the parameter setting using this profile.

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Fieldbus**	

ModuleType (51.01)	**DEVICENET***	
Module macid (51.02)	4**	set node address as required
Module baud rate (51.03)	2**	2 = 500 kBits/s
HW/SW option (51.04)	0	0 = **Hardware** 1 = **Software**
Stop function (51.05)	NA	not applicable when using **User specific assembly**
Output instance (51.06)	102	102 = **User specific assembly**
Input instance (51.07)	103	103 = **User specific assembly**
Output I/O par 1 (51.08) to *Input I/O par 9 (51.25)*	1 - 18	Set these values according table: Setting of parameter groups 51, 90 and 92 depending on desired data words and according to the desired numbers of data words
VSA I/O size (51.26)	1 - 9	Defines the length of the **User specific assembly** in pairs of data words. E.g. a parameter value of 4 means 4 word as output and 4 words as input.
FBA PAR REFRESH (51.27)	**DONE**, default	If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.

* Read-only or automatically detected by DeviceNet adapter

** If *HW/SW option (51.04)* = 0 (**Hardware**), the values are automatically set via the DIP switches of the RDNA-01

*DCS800 parameter setting using **User specific assembly***

Note:

± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Setting of parameter groups 51, 90 and 92

Parameter group 51				Direction	ABB	Parameter group 90 and 92			
	name	set value		PLC<->Drive	Datasets		name		def. value
51.08	Output I/O par 1	=	**1***	→	1,1	90.01	DsetXVal1	=	701
51.09	Output I/O par 2	=	**2***	→	1,2	90.02	DsetXVal2	=	2301
51.10	Output I/O par 3	=	**3**	→	1,3	90.03	DsetXVal3	=	2501
51.11	Output I/O par 4	=	**7**	→	3,1	90.04	DsetXplus2Val1	=	702
51.12	Input I/O par 1	=	**4***	←	2,1	92.01	DsetXplus1Val1	=	801
51.13	Input I/O par 2	=	**5***	←	2,2	92.02	DsetXplus1Val2	=	104
51.14	Input I/O par 3	=	**6**	←	2,3	92.03	DsetXplus1Val3	=	209
51.15	Input I/O par 4	=	**10**	←	4,1	92.04	DsetXplus3Val1	=	802
51.16	Output I/O par 5	=	**8**	→	3,2	90.05	DsetXplus2Val2	=	703
51.17	Output I/O par 6	=	**9**	→	3,3	90.06	DsetXplus2Val3	=	0
51.18	Output I/O par 7	=	**13**	→	5,1	90.07	DsetXplus4Val1	=	0
51.19	Output I/O par 8	=	**14**	→	5,2	90.08	DsetXplus4Val2	=	0
51.20	Output I/O par 9	=	**15**	→	5,3	90.09	DsetXplus4Val3	=	0
51.21	Input I/O par 5	=	**11**	←	4,2	92.05	DsetXplus3Val2	=	101
51.22	Input I/O par 6	=	**12**	←	4,3	92.06	DsetXplus3Val3	=	108
51.23	Input I/O par 7	=	**16**	←	6,1	92.07	DsetXplus5Val1	=	901
51.24	Input I/O par 8	=	**17**	←	6,2	92.08	DsetXplus5Val2	=	902
51.25	Input I/O par 9	=	**18**	←	6,3	92.09	DsetXplus5Val3	=	903

*For proper communication shown values have to be used

Setting of parameter groups 51, 90 and 92 depending on desired data words

Further information

Output and input parameters 51.08, …, 51.25 can also be connected directly to the desired DCS800 parameters. In this case please take care that the RDNA-01 adapter gets the changed values and also take care, that the used parameters are deleted from group 90 to prevent data trouble.

Switch on sequence

Please see the example at the end of this chapter.

Ethernet/IP communication with fieldbus adapter RETA-01

General

This chapter gives additional information using the Ethernet adapter RETA-01 together with the DCS800.

RETA-01 - DCS800

The Ethernet/IP communication with the drive requires the option RETA-01.

Related documentation

User's Manual Ethernet Adapter Module RETA-01.

The quoted page numbers correspond to the User's Manual.

EDS file

The EDS file for RETA-01 and DCS800 is available. Please ask Your local ABB agent for the newest one concerning the current DCS800 firmware.

Mechanical and electrical installation

If not already done so insert RETA-01 into slot 1 of the drive.

Drive configuration

The Ethernet adapter is activated by means of *CommModule (98.02)*.
Please note that the DCS800 works with the instances 102 / 103, if *Protocol (51.16)* is set to **2 (Ethernet/IP ABB Drives communication profile)**.
The instances 100 / 101, 20 / 70 and 21 / 71 are supported since firmware version 2.x, if *Protocol (51.16)* is set to **1 (Ethernet/IP AC/DC communication profile)**.
With these instances it is not possible to use the full flexibility of the DCS800.
For more information see User's Manual.

Parameter setting example using Ethernet/IP ABB Drives communication profile

Ethernet/IP ABB Drives communication profile uses up to 4 data words in each direction by default. The internal connection from and to the DCS800 has to be done by means of parameter group 51.

Ethernet/IP ABB Drives communication profile uses up to 12 data words in each direction. The configuration has to be done via fieldbus link configuration using Vendor Specific Drive I/O Object (Class 91h).

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Fieldbus**	

DsetXVal1 (90.01)	701, default	*MainCtrlWord (7.01)*; output data word 1 (control word) 1st data word from overriding control to drive

DsetXVal2 (90.02)	2301, default	*SpeedRef (23.01)*; output data word 2 (speed reference) 2nd data word from overriding control to drive
DsetXplus1Val1 (92.01)	801, default	*MainStatWord (8.01)*; input data word 1 (status word) 1st data word from drive to overriding control
DsetXplus1Val2 (92.02)	104, default	*MotSpeed (1.04)*; input data word 2 (speed actual) 2nd data word from drive to overriding control

ModuleType (51.01)	**ETHERNET TCP***	
Comm rate (51.02)	0	Auto-negotiate; automatic, set baud rate as required
DHCP (51.03)	0	DHCP disabled; IP address setting from following parameters
IP address 1 (51.04)	192**	e.g. IP address: 192.168.0.1
IP address 2 (51.05)	168**	
IP address 3 (51.06)	0**	
IP address 4 (51.07)	1**	
Subnet mask 1 (51.08)	255	e.g. subnet mask: 255.255.255.0
Subnet mask 2 (51.09)	255	
Subnet mask 3 (51.10)	255	
Subnet mask 4 (51.11)	0	
GW address 1 (51.12)	0	e.g. gateway address: 0.0.0.0
GW address 2 (51.13)	0	
GW address 3 (51.14)	0	
GW address 4 (51.15)	0	
Protocol (51.16)	2	1 = **Ethernet/IP AC/DC communication profile** 2 = **Ethernet/IP ABB Drives communication profile**
Modbus timeout (51.17)	22	0 = no monitoring 1 = 100 ms 22 = 2200 ms
Stop function (51.18)	0	0 = Ramp stop
Output 1 (51.19)	1	data word 1; setting via parameter 90.01
Output 2 (51.20)	2	data word 2; setting via parameter 90.02
Output 3 (51.21)	3	data word 3; setting via parameter 90.03

Output 4 (51.22)	7	data word 4; setting via parameter 90.04
Input 1 (51.23)	4	data word 1; setting via parameter 92.01
Input 2 (51.24)	5	data word 2; setting via parameter 92.02
Input 3 (51.25)	6	data word 3; setting via parameter 92.03
Input 4 (51.26)	10	data word 4; setting via parameter 92.04
FBA PAR REFRESH (51.27)	**DONE**, default	If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.

* Read-only or automatically detected by Ethernet adapter
** If all DIP switches (S1) are OFF; the IP address is set according to parameters 51.04, …, 51.07. In case at least one DIP switch is on, the last byte of the IP address [*IP address 4 (51.07)*] is set according to the DIP switches (see page 42).

DCS800 parameter setting using ***Ethernet/IP ABB Drives communication profile***

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Up to 4 data words

The content of Input/Output 1 to 4 can be configured with the RETA-01 configuration parameters. Please see table RETA-01 Ethernet/IP configuration parameters, which contains all the necessary basic settings.

Up to 12 data words

The DCS800 supports up to 12 data words in each direction. The first configuration of the RETA-01 adapter has to be done according to the table RETA-01 Ethernet/IP configuration parameters, which contains all the necessary basic settings.
The additional desired data words have to be configured via the fieldbus network using Vendor Specific Drive I/O Object (Class 91h). The adapter will automatically save the configuration.
The table RETA-01 Ethernet/IP configuration parameters shows the index configuration numbers and the corresponding data words (via data sets).

Please note: The grayed index is also addressed via group 51, please set the outputs and inputs to the same configuration numbers as shown in the table RETA-01 Ethernet/IP configuration parameters.
Example:

Task: The 5th data word of the telegram (index05) should be connected to *AuxCtrlWord (7.03)*.

To do: *AuxCtrlWord (7.03)* is the default content of *DsetXplus2Val2 (90.05).* The corresponding index configuration number of *DsetXplus2Val2 (90.05)* is 8. So the configuration has to be done using the following values in the IP address (all values are in hex):

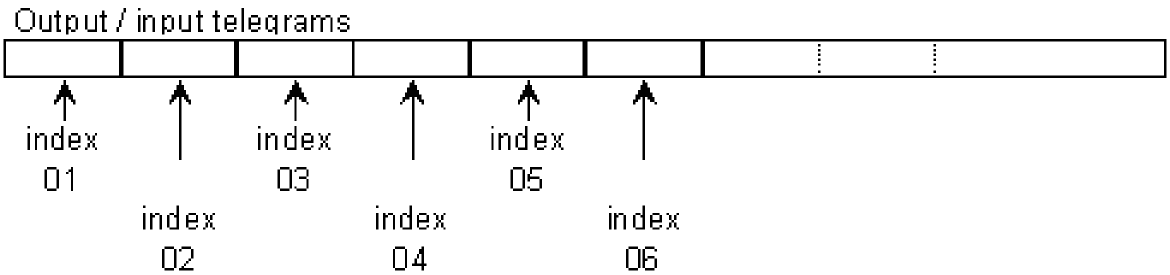
service	**0x10**	(write single)	class	**0x91**	(drive IO map function)
instance	**0x01**	(output)	attribute	**5**	(index05)
data	**08 00**	(2 char hex value)			

DCS800

RETA-01 Class 91h	index configuration no.	ABB Datasets	Parameter group 90 and 92 no.	name	def. value
Instance 1 (Output) — PLC ==> Drive					
index 01	= 1	1.1	90.01	DsetXVal1	= 701
index 02	= 2	1.2	90.02	DsetXVal2	= 2301
index 03	= 3	1.3	90.03	DsetXVal3	= 2501
index 04	= 7	3.1	90.04	DsetXplus2Val1	= 702
index 05	= 8	3.2	90.05	DsetXplus2Val2	= 703
index 06	= 9	3.3	90.06	DsetXplus2Val3	= 0
index 07	= 13	5.1	90.07	DsetXplus4Val1	= 0
index 08	= 14	5.2	90.08	DsetXplus4Val2	= 0
index 09	= 15	5.3	90.09	DsetXplus4Val3	= 0
index 10	= 19	7.1	90.10	DsetXplus6Val1	= 0
index 11	= 20	7.2	90.11	DsetXplus6Val2	= 0
index 12	= 21	7.3	90.12	DsetXplus6Val3	= 0
Instance 2 (Input) — PLC <== Drive					
index 01	= 4	2.1	92.01	DsetXplus1Val1	= 801
index 02	= 5	2.2	92.02	DsetXplus1Val2	= 104
index 03	= 6	2.3	92.03	DsetXplus1Val3	= 209
index 04	= 10	4.1	92.04	DsetXplus3Val1	= 802
index 05	= 11	4.2	92.05	DsetXplus3Val2	= 101
index 06	= 12	4.3	92.06	DsetXplus3Val3	= 108
index 07	= 16	6.1	92.07	DsetXplus5Val1	= 901
index 08	= 17	6.2	92.08	DsetXplus5Val2	= 902
index 09	= 18	6.3	92.09	DsetXplus5Val3	= 903
index 10	= 22	8.1	92.10	DsetXplus7Val1	= 904
index 11	= 23	8.2	92.11	DsetXplus7Val2	= 906
index 12	= 24	8.3	92.12	DsetXplus7Val3	= 907

RETA-01 Ethernet/IP configuration parameters

After configuration the packed telegram is defined:



Switch on sequence

Please see the example at the end of this chapter.

Modbus (RTU) communication with fieldbus adapter RMBA-01

General

This chapter gives additional information using the Modbus adapter RMBA-01 together with the DCS800.

RMBA-01 - DCS800

The Modbus communication with the drive requires the option RMBA-01.
The protocol Modbus RTU (**R**emote **T**erminal **U**nit using serial communication) is supported.

Related documentation

User's Manual Modbus Adapter Module RMBA-01.

The quoted page numbers correspond to the User's Manual.

Mechanical and electrical installation

If not already done so insert RMBA-01 into a slot of the drive. Slot 1 has to be used, if the Modbus should control the drive.

Drive configuration

The Modbus adapter is activated by means of *CommModule (98.02)* and *ModBusModule2 (98.08)*.
The serial communication parameters of the RMBA-01 adapter have to be set by means of group 52.
Up to 12 data words in each direction are possible.

Parameter setting example …

The Modbus adapter can be either used to control the drive with the overriding control system or only for monitoring purposes together with another fieldbus which is responsible for the control. Therefore different parameter settings are necessary.

… when controlling a drive

In data set mode (cyclic communication) the drive will be controlled from the overriding control using the Modbus.
Up to 12 data words in each direction are possible. The following table shows the parameter settings.

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Modbus**	
ModBusModule2 (98.08)	**Slot1**	

StationNumber (52.01)	1, …, 247	desired station number
BaudRate (52.02)	5	5 = 9600 Baud
Parity (52.03)	4	4 = Even

DsetXVal1 (90.01)	701, default	*MainCtrlWord (7.01)*; output data word 1 (control word) 1st data word from overriding control to drive (40001 => data word 1.1)
DsetXVal2 (90.02)	2301, default	*SpeedRef (23.01)*; output data word 2 (speed reference) 2nd data word from overriding control to drive (40002 => data word 1.2)
DsetXVal3 (90.03)	2501, default	*TorqRefA (25.01)*; output data word 3 (torque reference) 3rd data word from overriding control to drive (40003 => data word 1.3)
up to, …,		
DsetXplus6Val3 (90.12)	0, default	not connected; output data word 12 (not connected) 12th data word from overriding control to drive (40021 <= data word 7.3)

DsetXplus1Val1 (92.01)	801, default	*MainStatWord (8.01)*; input data word 1 (status word) 1st data word from drive to overriding control (40004 <= data word 2.1)
DsetXplus1Val2 (92.02)	104, default	*MotSpeed (1.04)*; input data word 2 (speed actual) 2nd data word from drive to overriding control (40005 <= data word 2.2)
DsetXplus1Val3 (92.03)	209, default	*TorqRef2 (2.09)*; input data word 3 (torque reference) 3rd data word from drive to overriding control (40006 <= data word 2.3)
up to, …,		
DsetXplus7Val3 (92.12)	907, default	*Alarmword2 (9.07)*; input data word 12 (alarm word 2) 12th data word from drive to overriding control (40024 <= data word 8.3)

DCS800 parameter setting using a Modbus controlling the drive

Note:
New settings of group 52 take effect only after the next power up of the adapter.

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

… when used for monitoring only

For monitoring only read commands are supported.
Up to 24 data words for monitoring are possible, because the 12 data words written to by the overriding control (see group 90) can also be read. The following table shows the parameter settings.

Drive parameters	**Settings**	**Comments**
CommModule (98.02)	**FldBusModbus**	**FldBusModbus** means controlling the drive by means of another R-type fieldbus adapter - see description of *CommModule (98.02)*
ModBusModule2 (98.08)	**Slot2** or **Slot3**	depends on the location of the adapter
StationNumber (52.01)	1, …, 247	desired station number
BaudRate (52.02)	5	5 = 9600 Baud
Parity (52.03)	4	4 = Even
DsetXVal1 (90.01)	701, default	*MainCtrlWord (7.01)*; output data word 1 (control word) 1^{st} data word from overriding control to drive (40001 => data word 1.1)
DsetXVal2 (90.02)	2301, default	*SpeedRef (23.01)*; output data word 2 (speed reference) 2^{nd} data word from overriding control to drive (40002 => data word 1.2)
DsetXVal3 (90.03)	2501, default	*TorqRefA (25.01)*; output data word 3 (torque reference) 3^{rd} data word from overriding control to drive (40003 => data word 1.3)
up to, …,		
DsetXplus6Val3 (90.12)	0, default	not connected; output data word 12 (not connected) 12^{th} data word from overriding control to drive (40021 <= data word 7.3)

DsetXplus1Val1 (92.01)	801, default	*MainStatWord (8.01)*; input data word 1 (status word) 1st data word from drive to overriding control (40004 <= data word 2.1)
DsetXplus1Val2 (92.02)	104, default	*MotSpeed (1.04)*; input data word 2 (speed actual) 2nd data word from drive to overriding control (40005 <= data word 2.2)
DsetXplus1Val3 (92.03)	209, default	*TorqRef2 (2.09)*; input data word 3 (torque reference) 3rd data word from drive to overriding control (40006 <= data word 2.3)
up to, …,		
DsetXplus7Val3 (92.12)	907, default	*Alarmword2 (9.07)*; input data word 12 (alarm word 2) 12th data word from drive to overriding control (40024 <= data word 8.3)

DCS800 parameter setting using a Modbus monitoring the drive

Note:
New settings of group 52 take effect only after the next power up of the adapter.

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Setting of PLC, parameter groups 90 and 92

Set in PLC	Direction PLC<->Drive	ABB Datasets	Parameter group 90 and 92	name		def. value
40001	⇒	1,1	90.01	DsetXVal1	=	701
40002	⇒	1,2	90.02	DsetXVal2	=	2301
40003	⇒	1,3	90.03	DsetXVal3	=	2501
40004	⇐	2,1	92.01	DsetXplus1Val1	=	801
40005	⇐	2,2	92.02	DsetXplus1Val2	=	104
40006	⇐	2,3	92.03	DsetXplus1Val3	=	209
40007	⇒	3,1	90.04	DsetXplus2Val1	=	702
40008	⇒	3,2	90.05	DsetXplus2Val2	=	703
40009	⇒	3,3	90.06	DsetXplus2Val3	=	0
40010	⇐	4,1	92.04	DsetXplus3Val1	=	802
40011	⇐	4,2	92.05	DsetXplus3Val2	=	101
40012	⇐	4,3	92.06	DsetXplus3Val3	=	108
40013	⇒	5,1	90.07	DsetXplus4Val1	=	0
40014	⇒	5,2	90.08	DsetXplus4Val2	=	0
40015	⇒	5,3	90.09	DsetXplus4Val3	=	0
40016	⇐	6,1	92.07	DsetXplus5Val1	=	901
40017	⇐	6,2	92.08	DsetXplus5Val2	=	902
40018	⇐	6,3	92.09	DsetXplus5Val3	=	903
40019	⇒	7,1	90.10	DsetXplus6Val1	=	0
40020	⇒	7,2	90.11	DsetXplus6Val2	=	0
40021	⇒	7,3	90.12	DsetXplus6Val3	=	0
40022	⇐	8,1	92.10	DsetXplus7Val1	=	904
40023	⇐	8,2	92.11	DsetXplus7Val2	=	906
40024	⇐	8,3	92.12	DsetXplus7Val3	=	907

Setting of PLC, parameter groups 90 and 92 depending on desired data words

Switch on sequence

Please see the example at the end of this chapter.

Modbus/TCP communication with fieldbus adapter RETA-01

General

This chapter gives additional information using the Ethernet adapter RETA-01 together with the DCS800.

RETA-01 - DCS800

The Modbus/TCP communication with the drive requires the option RETA-01. The protocol Modbus TCP (Ethernet) is supported.

Related documentation

User's Manual Ethernet Adapter Module RETA-01.

The quoted page numbers correspond to the User's Manual.

Mechanical and electrical installation

If not already done so insert RETA-01 into slot 1 of the drive.

Drive configuration

The Ethernet adapter is activated by means of *CommModule (98.02)*.
Please note that the DCS800 works with **Modbus/TCP**, if *Protocol (51.16)* is set to 0 (**Modbus/TCP**).

Parameter setting example using Modbus/TCP

Modbus/TCP is using 4 data words in each direction. The following table shows the parameter setting using this protocol.

Drive parameters	Settings	Comments
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Fieldbus**	

DsetXVal1 (90.01)	701, default	*MainCtrlWord (7.01)*; output data word 1 (control word) 1st data word from overriding control to drive
DsetXVal2 (90.02)	2301, default	*SpeedRef (23.01)*; output data word 2 (speed reference) 2nd data word from overriding control to drive
DsetXplus1Val1 (92.01)	801, default	*MainStatWord (8.01)*; input data word 1 (status word) 1st data word from drive to overriding control
DsetXplus1Val2 (92.02)	104, default	*MotSpeed (1.04)*; input data word 2 (speed actual) 2nd data word from drive to overriding control

ModuleType (51.01)	**ETHERNET TCP***	

Comm rate (51.02)	0	Auto-negotiate; automatic, set baud rate as required
DHCP (51.03)	0	DHCP disabled; IP address setting from following parameters
IP address 1 (51.04)	192**	e.g. IP address: 192.168.0.1
IP address 2 (51.05)	168**	
IP address 3 (51.06)	0**	
IP address 4 (51.07)	1**	
Subnet mask 1 (51.08)	255	e.g. subnet mask: 255.255.255.0
Subnet mask 2 (51.09)	255	
Subnet mask 3 (51.10)	255	
Subnet mask 4 (51.11)	0	
GW address 1 (51.12)	0	e.g. gateway address: 0.0.0.0
GW address 2 (51.13)	0	
GW address 3 (51.14)	0	
GW address 4 (51.15)	0	
Protocol (51.16)	0	0 = **Modbus/TCP**
Modbus timeout (51.17)	22	0 = no monitoring 1 = 100 ms 22 = 2200 ms
Stop function (51.18)	NA	not applicable when using **Modbus/TCP**
Output 1 (51.19)	1	data word 1; setting via parameter 90.01
Output 2 (51.20)	2	data word 2; setting via parameter 90.02
Output 3 (51.21)	3	data word 3; setting via parameter 90.03
Output 4 (51.22)	7	data word 4; setting via parameter 90.04
Input 1 (51.23)	4	data word 1; setting via parameter 92.01
Input 2 (51.24)	5	data word 2; setting via parameter 92.02
Input 3 (51.25)	6	data word 3; setting via parameter 92.03
Input 4 (51.26)	10	data word 4; setting via parameter 92.04
FBA PAR REFRESH (51.27)	**DONE**, default	If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.

* Read-only or automatically detected by Ethernet adapter
** If all DIP switches (S1) are OFF; the IP address is set according to parameters 51.04, …, 51.07. In case at least one DIP switch is on, the last byte of the IP address [*IP address 4 (51.07)*] is set according to the DIP switches (see page 42).

*DCS800 parameter setting using **Modbus/TCP** protocol*

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Switch on sequence

Please see the example at the end of this chapter.

Profibus communication with fieldbus adapter RPBA-01

General

This chapter gives additional information using the Profibus adapter RPBA-01 together with the DCS800.

RPBA-01 - DCS800

The Profibus communication with the drive requires the option RPBA-01.

Related documentation

User's Manual PROFIBUS DP Adapter Module RPBA-01.

The quoted page numbers correspond to the User's Manual.

Overriding control configuration

Supported operation mode is **VENDOR SPECIFIC** for ABB Drives (see page 19 and 20).
The RPBA-01 uses data consistent communication, meaning that the whole data frame is transmitted during a single program cycle. Some overriding controls handle this internally, but others must be programmed to transmit data consistent telegrams.

Mechanical and electrical installation

If not already done so insert RPBA-01 into slot 1 of the drive (see page 21).

Drive configuration

The Profibus adapter is activated by means of *CommModule (98.02)* (see page 22).
Please note that the DCS800 works only with the ABB Drives profile.

Parameter setting example 1 using PPO Type 1

ABB Drives profile (Vendor-specific) with **PPO Type 1** (DP-V0) (see page 25).
The first two data words (PZD1 OUT, PZD2 OUT) from the overriding control to the drive are fixed connected as control word and speed reference at the Profibus side and cannot be changed.
The first two data words (PZD1 IN, PZD2 IN) from the drive to the overriding control are fixed connected as status word and speed actual at the Profibus side and cannot be changed.

Drive parameters	**Settings**	**Comments**
CommandSel (10.01)	**MainCtrlWord**	
Ref1Sel (11.03)	**SpeedRef2301**	
CommModule (98.02)	**Fieldbus**	

DsetXVal1 (90.01)	701, default	*MainCtrlWord (7.01)*; PZD1 OUT (control word) 1st data word from overriding control to drive

DsetXVal2 (90.02)	2301, default	*SpeedRef (23.01)*; PZD2 OUT (speed reference) 2nd data word from overriding control to drive
DsetXplus1Val1 (92.01)	801, default	*MainStatWord (8.01)*; PZD1 IN (status word) 1st data word from drive to overriding control
DsetXplus1Val2 (92.02)	104, default	*MotSpeed (1.04)*; PZD2 IN (speed actual) 2nd data word from drive to overriding control

ModuleType (51.01)	**PROFIBUS DP***	
Node address (51.02)	4	set node address as required
Baud rate (51.03)	1500*	
PPO-type (51.04)	**PPO1***	
…		
DP Mode (51.21)	0	
FBA PAR REFRESH (51.27)	**DONE**, default	If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.

* Read-only or automatically detected by Profibus adapter

DCS800 parameter setting using ***PPO Type 1***

Note:
± 20.000 speed units (decimal) for speed reference [*SpeedRef (23.01)*] and speed actual [*MotSpeed (1.04)*] corresponds to the speed shown in *SpeedScaleAct (2.29)*. That speed is set by means of *M1SpeedScale (50.01)* respectively *M1SpeedMin (20.01)* or *M1SpeedMax (20.02).*

Parameter setting example 2 using PPO types 2, 4 and 5

The first two data words (PZD1 OUT, PZD2 OUT) from the overriding control to the drive are fixed connected as control word and speed reference at the Profibus side and cannot be changed.
The first two data words (PZD1 IN, PZD2 IN) from the drive to the overriding control are fixed connected as status word and speed actual at the Profibus side and cannot be changed.

Further data words are to be connected to desired parameters respectively signals by means of parameters in group 51:

- *PZD3 OUT (51.05)* means 3rd data word from overriding control to drive,
- *PZD3 IN (51.06)* means 3rd data word from Drive to overriding control
  to
- *PZD10 OUT (51.18)* means 10th data word from overriding control to drive,
- *PZD10 IN (51.19)* means 10th data word from drive to overriding control

or by means of setting parameters in group 90 and group 92.

Communication via group 51

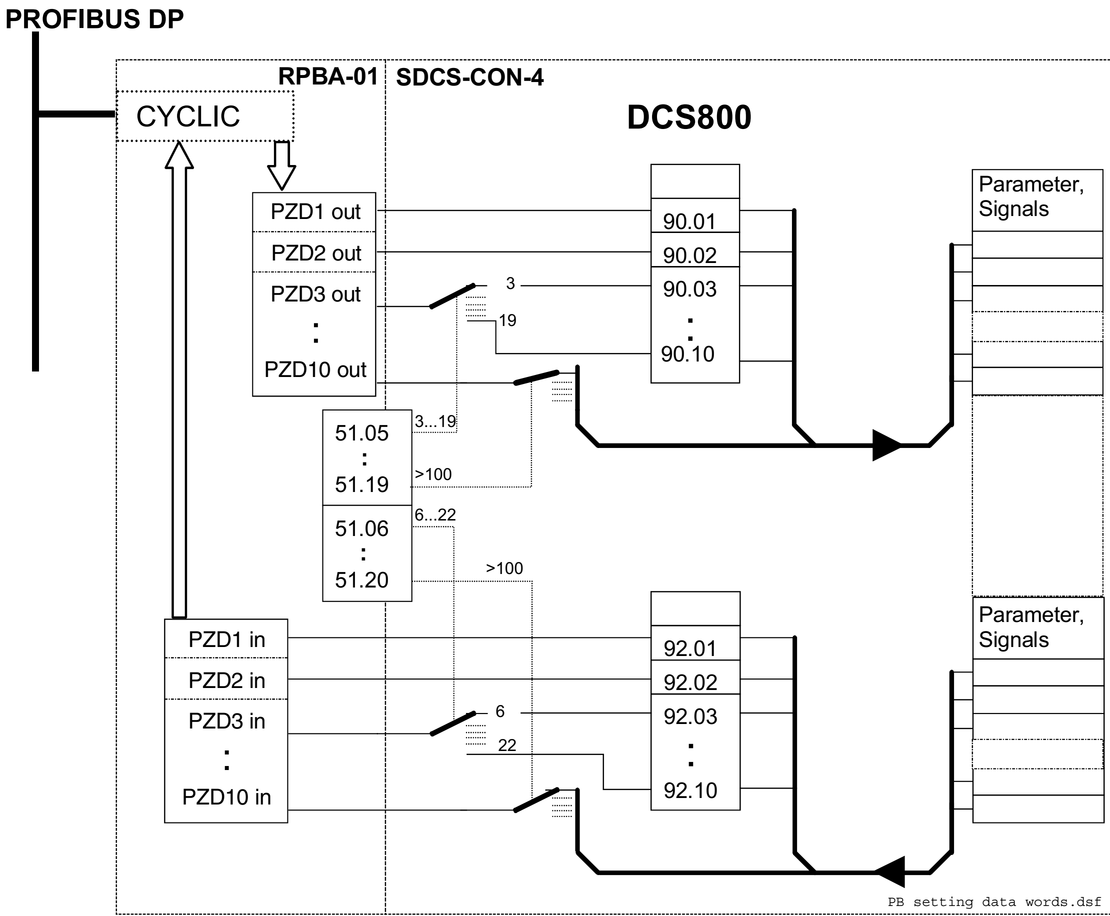
E.g. the 3rd data word from overriding control to drive should be the torque reference and the 3rd data word from the drive to the overriding control should be the actual motor torque. Therefore following settings have to be made:

- *PZD3 OUT (51.05)* = 2501 [*TorqRefA (25.01)*] and
- *PZD3 IN (51.06)* = 107 [*MotTorqFilt (1.07)*].

After changing parameters in group 51 please don't forget to reset the RPBA-01 adapter by means of *FBA PAR REFRESH (51.27)* = **RESET**. Now the corresponding parameters in group 90 and group 92 are disabled.

Attention:

Make sure, that the used parameters, like *TorqRefA (25.01)* are removed from groups 90 and 91.



Setting of data words using only group 51 or using group 90 and group 92

Communication via group 90 and group 92

The other possibility - perhaps more familiar - is to connect via group 90 and group 92.

Again the 3rd data word from overriding control to drive should be the torque reference and the 3rd data word from the drive to the overriding control should be the actual motor torque. Therefore following settings have to be made (values see table below):

- *PZD3 OUT (51.05)* = 3 and
- *PZD3 IN (51.06)* = 6.

After changing parameters in group 51 please don't forget to reset the RPBA-01 adapter by means of *FBA PAR REFRESH (51.27)* = **RESET**. Now the corresponding parameters in group 90 and group 92 are enabled. Following settings have to be made now:

- *DsetXVal3 (90.03)* = 2501 [*TorqRefA (25.01)*] and
- *DsetXplus1Val3 (92.03)* = 107 [*MotTorqFilt (1.07)*].

			Parameter group 51	name		**set value**	Direction PLC<->Drive	ABB Datasets	Parameter group 90 and 92	name		def. value
PPO 5	PPO 2, PPO 4	PPO 1, PPO 3		fixed connection			→	1.1	90.01	DsetXVal1	=	701
				fixed connection			←	2.1	92.01	DsetXplus1Val1	=	801
				fixed connection			→	1.2	90.02	DsetXVal2	=	2301
				fixed connection			←	2.2	92.02	DsetXplus1Val2	=	104
			51.05	PZD3 OUT	=	**3**	→	1.3	90.03	DsetXVal3	=	2501
			51.06	PZD3 IN	=	**6**	←	2.3	92.03	DsetXplus1Val3	=	209
			51.07	PZD4 OUT	=	**7**	→	3.1	90.04	DsetXplus2Val1	=	702
			51.08	PZD4 IN	=	**10**	←	4.1	92.04	DsetXplus3Val1	=	802
			51.09	PZD5 OUT	=	**8**	→	3.2	90.05	DsetXplus2Val2	=	703
			51.10	PZD5 IN	=	**11**	←	4.2	92.05	DsetXplus3Val2	=	101
			51.11	PZD6 OUT	=	**9**	→	3.3	90.06	DsetXplus2Val3	=	0
			51.12	PZD6 IN	=	**12**	←	4.3	92.06	DsetXplus3Val3	=	108
			51.13	PZD7 OUT	=	**13**	→	5.1	90.07	DsetXplus4Val1	=	0
			51.14	PZD7 IN	=	**16**	←	6.1	92.07	DsetXplus5Val1	=	901
			51.15	PZD8 OUT	=	**14**	→	5.2	90.08	DsetXplus4Val2	=	0
			51.16	PZD8 IN	=	**17**	←	6.2	92.08	DsetXplus5Val2	=	902
			51.17	PZD9 OUT	=	**15**	→	5.3	90.09	DsetXplus4Val3	=	0
			51.18	PZD9 IN	=	**18**	←	6.3	92.09	DsetXplus5Val3	=	903
			51.19	PZD10 OUT	=	**19**	→	7.1	90.10	DsetXplus6Val1	=	0
			51.20	PZD10 IN	=	**22**	←	8.1	92.10	DsetXplus7Val1	=	904

Setting of data words using group 90 and group 92

Switch on sequence

Bit	15 ... 11	RemoteCmd 10	Inching2 09	Inching1 08	Reset 07	RampInZero 06	RampHold 05	RampOutZero 04	Run 03	Off2N 02	Off2N 01	On 00	**Dec.**	**Hex.**
Reset		1	x	x	1	x	x	x	x	x	x	x	**1270**	**04F6**
Off (before On)		1	0	0	0	x	x	x	0	1	1	0	**1142**	**0476**
On (main cont. On)		1	0	0	0	x	x	x	0	1	1	1	**1143**	**0477**
Run (with reference)		1	0	0	0	1	1	1	1	1	1	1	**1151**	**047F**
E-Stop		1	x	x	x	1	1	1	1	0	1	1	**1147**	**047B**
Start inhibit		1	x	x	x	x	x	x	x	x	0	x	**1140**	**0474**

Examples for the MainCtrlWord (7.01)

Data set table

A lot of fieldbus communications use the data set table to transmit data words. The next table shows the configuration number of each data word and the corresponding pointer:

Data set no.	Configuration no.	Parameter (pointer) from PLC to DCS800	Parameter (pointer) from DCS800 to PLC
1.1	1	90.01	
1.2	2	90.02	
1.3	3	90.03	
2.1	4		92.01
2.2	5		92.02
2.3	6		92.03
3.1	7	90.04	
3.2	8	90.05	
3.3	9	90.06	
4.1	10		92.04
4.2	11		92.05
4.3	12		92.06
5.1	13	90.07	
5.2	14	90.08	
5.3	15	90.09	
6.1	16		92.07
6.2	17		92.08
6.3	18		92.09

Data set no.	Configuration no.	Parameter (pointer) from PLC to DCS800	Parameter (pointer) from DCS800 to PLC
7.1	19	90.10	
7.2	20	90.11	
7.3	21	90.12	
8.1	22		92.10
8.2	23		92.11
8.3	24		92.12
9.1	25	90.13	
9.2	26	90.14	
9.3	27	90.15	
10.1	28		92.13
10.2	29		92.14
10.3	30		92.15
11.1	31	90.16	
11.2	32	90.17	
11.3	33	90.18	
12.1	34		92.16
12.2	35		92.17
12.3	36		92.18

Data set no.	Configuration no.	Parameter (pointer) from PLC to DCS800	Parameter (pointer) from DCS800 to PLC
13.1	37	91.01	
13.2	38	91.02	
13.3	39	91.03	
14.1	40		93.01
14.2	41		93.02
14.3	42		93.03
15.1	43	91.04	
15.2	44	91.05	
15.3	45	91.06	
16.1	46		93.04
16.2	47		93.05
16.3	48		93.06

Configuration numbers of each data word and its corresponding pointer

Adaptive Program (AP)

Chapter overview

This chapter describes the basics of the Application Program and instructs how to build an application. All needed parameters can be found in the groups 83 to 86.

What is the Adaptive Program

Conventionally, the user can control the operation of the drive by parameters. Each parameter has a fixed set of choices or a setting range. The parameters make adapting of the drive easy, but the choices are limited. It is not possible to customize the drive any further. AP makes customizing possible without the need of a special programming tool or language:

- AP is using function blocks,
- DWL AP is the programming and documentation tool.

The maximum size of AP is 16 function blocks. The program may consist of several separate functions.

Features

The Adaptive Program of DCS800 provides the following features:

- 16 function blocks
- more than 20 block types
- password protection
- 4 different cycle times selectable
- shift functions for function blocks
- debug functions
  - output forcing
  - breakpoint
  - single step
  - single cycle
- additional output write pointer parameter for each block (group 86)
- 10 additional user constants (group 85) used as data container

How to build the program

The programmer connects a function block to other blocks through a Block Parameter Set. The sets are also used for reading values from the firmware and transferring data to the firmware. Each Block Parameter Set consists of six parameters in group 84 and a write pointer in group 86.

The programmer connects a function block to other blocks through a Block Parameter Set. The sets are also used for reading values from the firmware and transferring data to the firmware. Each Block Parameter Set consists of six parameters in group 84 and a write pointer in group 86. The figure below shows the use of Block Parameter Set 1 in the firmware (parameters 84.04 to 84.09 and 86.01):

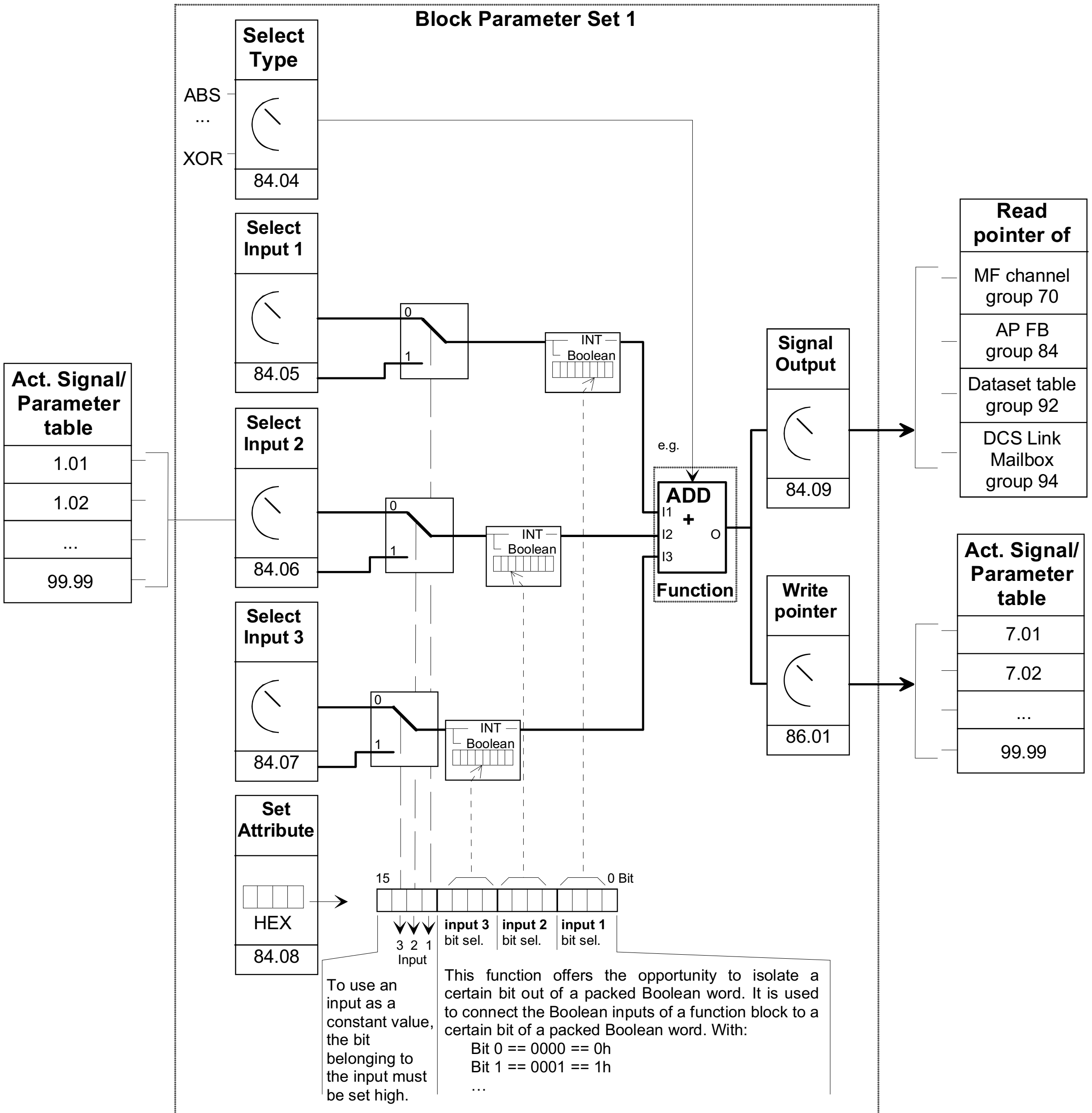
- *Block1Type (84.04)* selects the function block type.
- *Block1In1 (84.05)* selects the source of IN1. A negative value means that the source will be inverted.
- *Block1In2 (84.06)* selects the source of IN2. A negative value means that the source will be inverted.
- *Block1In3 (84.07)* selects the source of IN3. A negative value means that the source will be inverted.
- *Block1Attrib (84.08)* defines the attributes of the inputs.
- *Block1Output (84.09)* provides the value of the function block output, which can be used further for other input selections. The user cannot edit this parameter value.
- The output value is also available in write pointer *Block1Out (86.01)*. *Block1Out (86.01)* contains the destination parameter, into which the value is written.

How to connect the Application Program with the firmware

The outputs of the Adaptive Program need to be connected to the firmware. For that purpose there are two possibilities:

- The outputs, e.g. *Block1Output (84.09),* can be selected for further functions.
- The output values are available in the write pointers, e.g. *Block1Out (86.01).* These parameters contain the destination parameters, into which the values are written.

Block Parameter Set for block 1



Example:
Add a constant value and an external additional reference to the speed reference:
1. Set 84.04 = 2 (selection of ADD function)
2. Set 84.05 = xx.xx (selection of the speed reference for Input 1)
3. Set 84.06 = xx.xx (selection of an external ref for Input 2)
4. Set 84.07 = 1500 (constant value for Input 3)
5. Set 84.08 = 4000h (because Input 3 = constant ⇒ Bit 14=1 ⇒ 4000h)
6. Set 86.01 = xx.xx (write processed value to destination parameter for further processing)
7. 84.09: contains the processed value

How to control the execution of the program

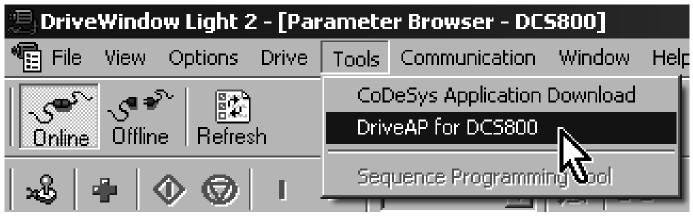
The Adaptive Program executes the function blocks in numerical order according to the block number 1, …, 16. All blocks use the same time level. This cannot be changed by the user. The user can:

- select the operation mode of the program (stop, start, editing, single cycling, single stepping)
- adjust the execution time level of the program and
- activate or de-activate blocks.

DWL AP

General

Another way to create applications is with DWL AP. It is a program plugged into DriveWindow Light and can be opened with *Tools* and *DriveAP for DCS800*:



Important keys and buttons

DWL AP is controlled by means of following keys and buttons:

Keys and buttons	Function
Ctrl + left mouse button on a box or function block	Change / insert function blocks, connect in- and outputs in **Edit** mode
Shift + left mouse button on the red cross	View actual values in **Start** mode
Cancel	Abort the action
Help	Open the online help

Program modes

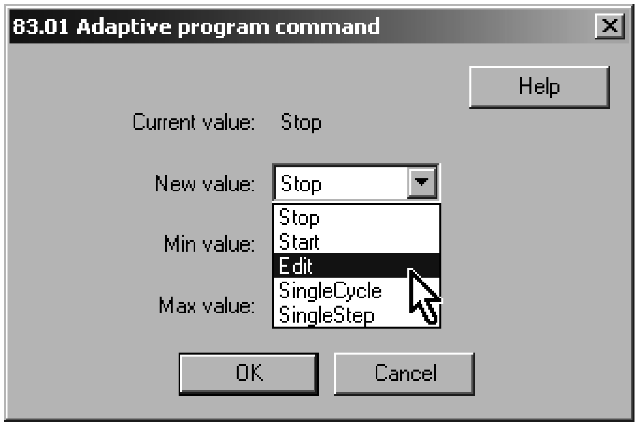
There are 5 modes for the Adaptive Program, see *AdapProgCmd (83.01)*:

- **Stop:** the Adaptive Program is not running and cannot be edited,
- **Start:** the Adaptive Program is running and cannot be edited,
- **Edit:** the Adaptive Program is not running and can be edited,
- **SingleCycle** and **SingleStep** are used for testing.

Change to Edit mode

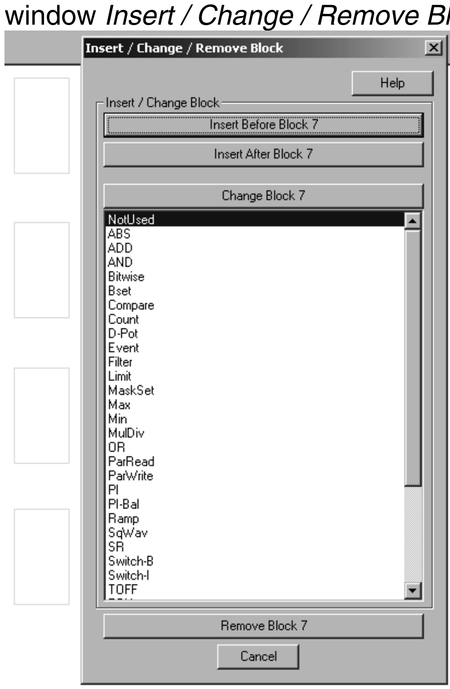
Use *Ctrl + left mouse button* on 83.01 Adaptive Program Control

and set to **Edit**:

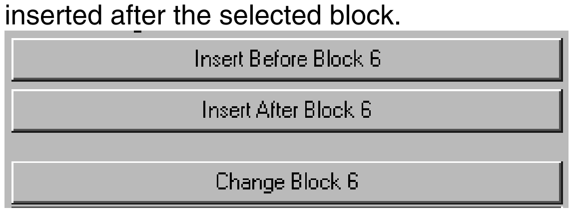


Insert function blocks

Use *Ctrl + left mouse button* on one of the yellow boxes. This opens the pop-up window *Insert / Change / Remove Block*:



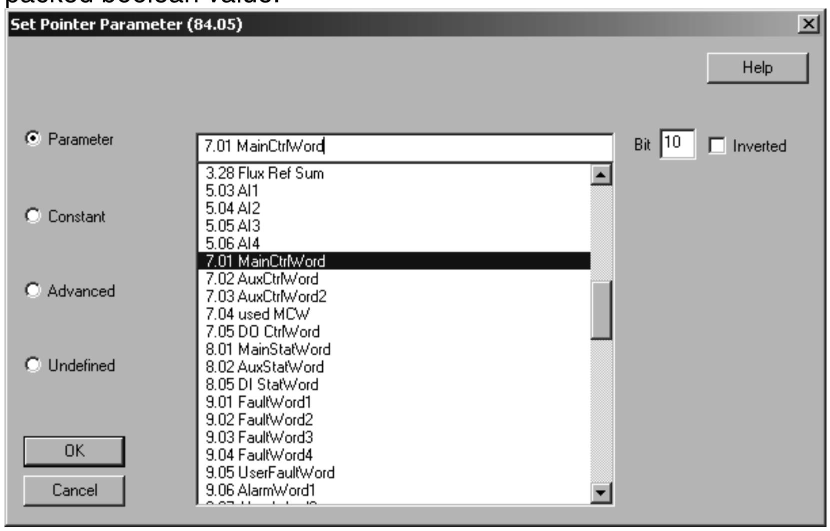
In this manner it is possible to insert up to 16 function blocks from the list to the desktop. With the button *Change Block xx* the selected block will be changed. The button *Insert Before Block xx* means that the new block will be inserted before the selected block. Button *Insert After Block xx* means that the new block will be inserted after the selected block.



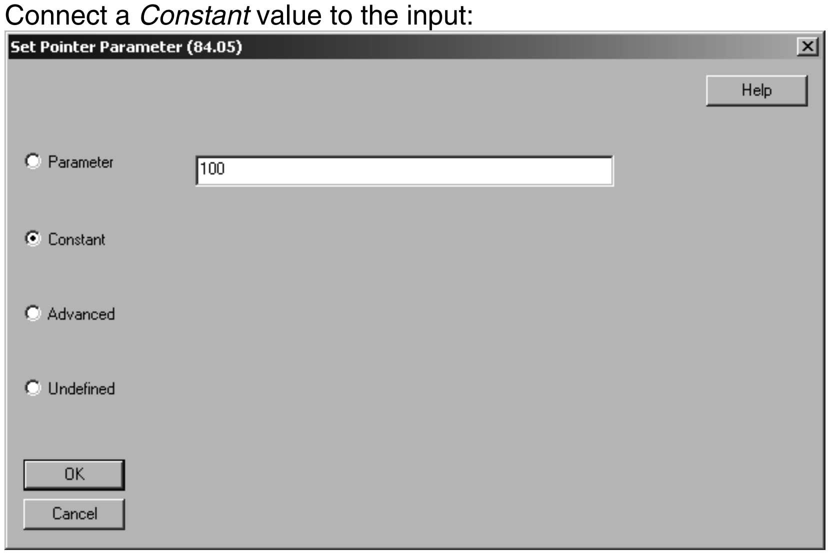
Connect function blocks

Function blocks can be connected to other blocks or to firmware parameters. To connect use *Ctrl + left mouse button* on the red cross at the input. This opens the pop-up window *Set Pointer Parameter*. This window provides several connection possibilities:

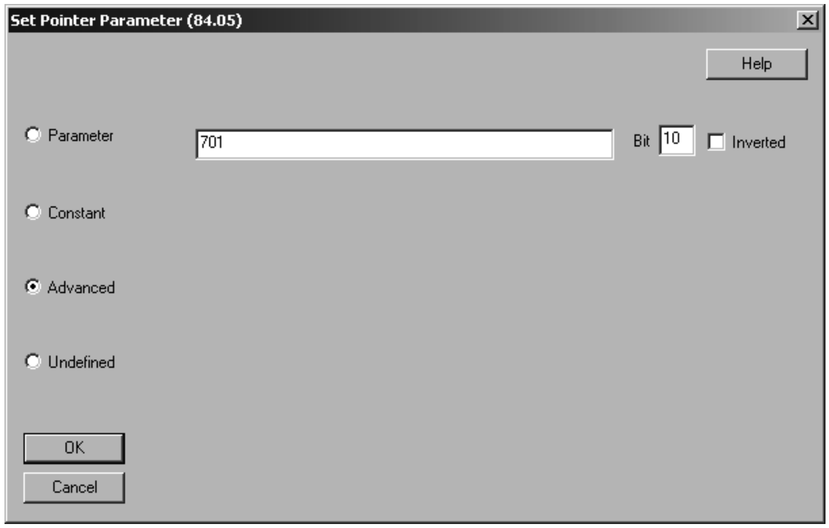
– Connect a *Parameter* from the list and set the bit in case of connecting a packed boolean value:



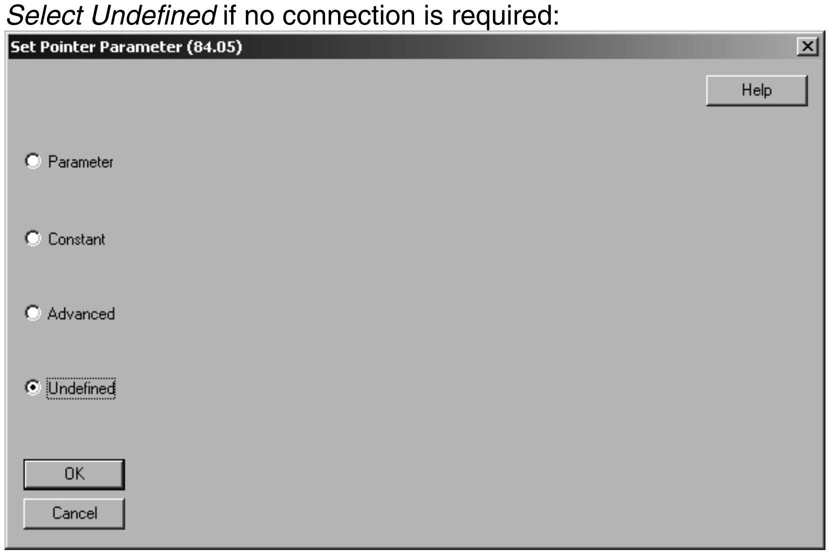
– Connect a *Constant* value to the input:



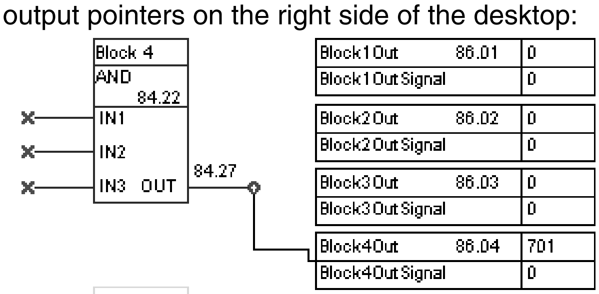
– In *Advanced* mode choose the parameter with group * 100 + index, e.g. *MainCtrlWord (7.01)* == 701:



– *Select Undefined* if no connection is required:

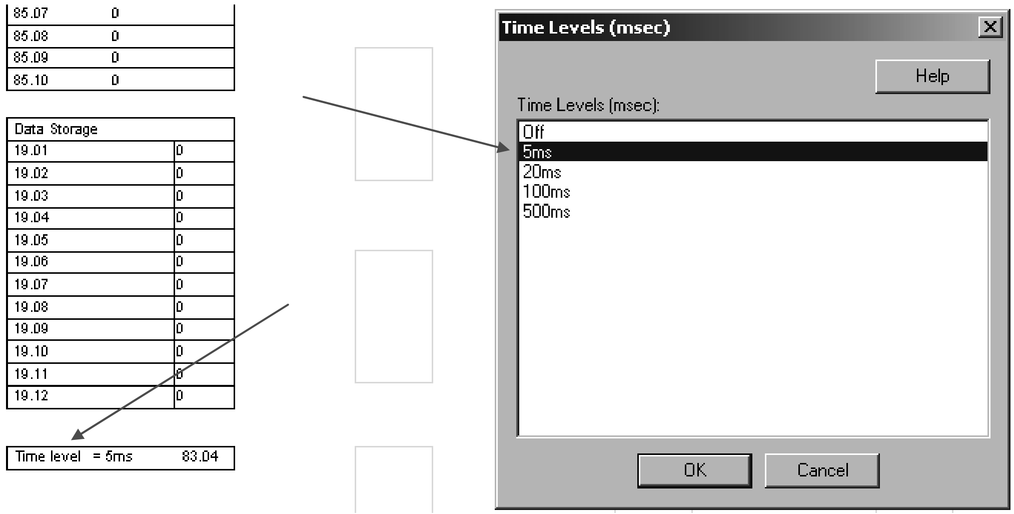


– Connections of outputs to firmware parameters can be done by means of the output pointers on the right side of the desktop:



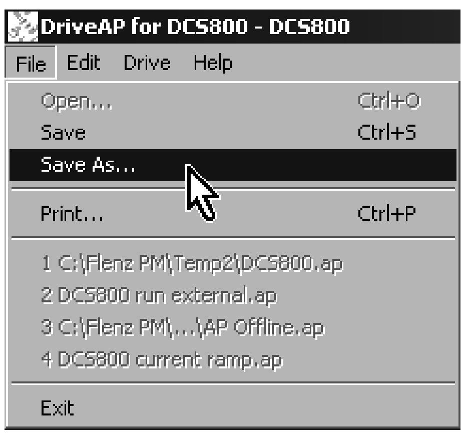
If an output of a function block should be connected with an input of a function block simply select the output's parameter at the input.

Set the *Time level*



Saving AP applications

It is possible to save AP applications as **.ap* files :



Function blocks

General rules

The use of block input 1 (BlockxIn1) is compulsory (it must not be left unconnected). Use of input 2 (BlockxIn2) and input 3 (BlockxIn3) is voluntary for the most blocks. As a rule of thumb, an unconnected input does not affect the output of the block.
The Attribute Input (BlockxAttrib) is to set with the attributes, like declaration of constant and bits, of all three inputs. DWL AP does this automatically.
The constant attribute defines a block constant which can only be changed or modified in EDIT mode.

Block inputs

The blocks use two input formats:

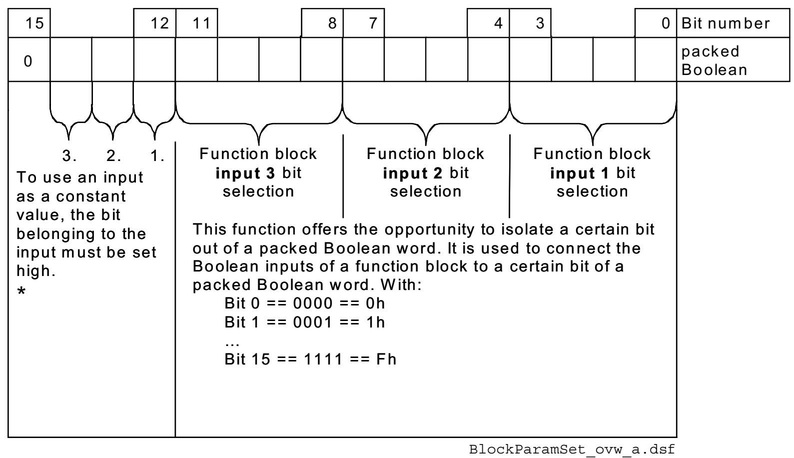
- integer or
- boolean

The used format depends on the function block type. For example, the ADD block uses integer inputs and the OR block boolean inputs.

Note:
The inputs of the block are read when the execution of the block starts, not simultaneously for all blocks!

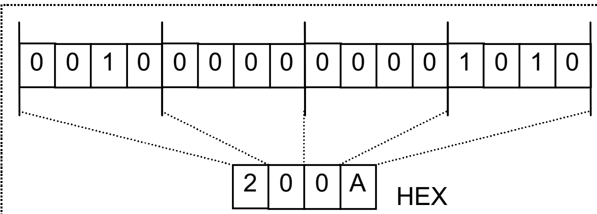
Block input attributes

Block inputs gets the parameter of signal source or user constants (e.g. 85.01). Depending on the used block function and depending on the desired function the attributes of all three inputs are to be set as integer, constant or as selection of a bit of a 16-bit word source.
Therefore it is used a 16-bit word, which is defined as following:



* this type of constant defines a Block Constant, which can only be modified in EDIT mode.

Example:



Example of attribute parameter, with
BlockxIn1 as boolean, bit 10
BlockxIn2 as constant
BlockxIn3 as integer

→ Bits converted into hex, the value 200A (H) is to be set into parameter BlockxAttrib.

Parameter value as an integer input

How the block handles the input

The block reads the selected value in as an integer.

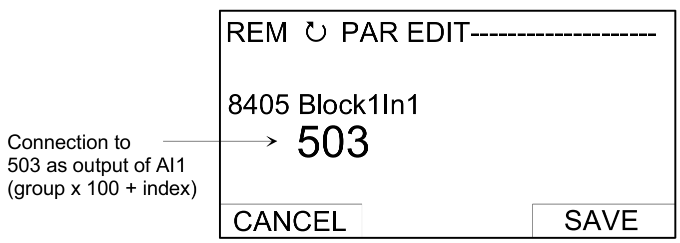
Note:
The parameter selected as an input should be an integer value. The internal scaling for each parameter can be found in chapter Parameters.

How to select the input

- Scroll to the input selection parameter of the block and switch to edit mode (Enter).
- Set the address, from which the input value is to be read, with group * 100 + index, e.g. *AccTime1 (22.01)* = 2201. A negative address (e.g. -2201) will act an inversion of the connected value.

The figure below shows the DCS800 Control Panel display when the input BlockxIn1 (with e.g. x = 1 for 1. block) selection parameter is in edit mode.

Display of panel



Example:
AI1 is supplied with a voltage source of 5.8 V. AI1 is connected to the block as follows:

- Scroll to *Block1In1 (84.05)* and shift to edit mode (Enter). Set to 503, because the value of AI1 is shown in group 5 with index 3 - *AI1 Val (05.03)* == 05 * 100 + 3 = 503.
- The value at the input of the block is 5800, since the integer scaling of *AI1 Val (05.03)* is 1000 == 1 V see chapter Parameters.

Constant as an integer input

How to set and connect the input

•Option 1

- Scroll to the input selection parameter of the block and switch to edit mode (Enter).
- Give the constant value to this input parameter (arrow keys).
- Accept by Enter.
- Scroll to attribute parameter, e.g. Block1Attrib (4.08).
- Set the bit for constant attribute of this input in Block1Attrib (4.08).
- Accept by Enter.

The constant may have a value from -32768 to 32767. The constant cannot be changed while the Application Program is running. The figures below shows the DCS800 Control Panel display when *Block1In2 (84.06)* is in edit mode and the constant field is visible:

Display of panel

REM ↻ PAR EDIT--------------------

8406 Block1In2

Value of the desired constant → -10000

CANCEL SAVE

Display of panel

REM ↻ PAR EDIT--------------------

8408 Block1Attrib

Setting of constant value of Block1In2 input → 2000 hex

CANCEL SAVE

Option 2

- User constants 85.01 to 85.10 are reserved for the Adaptive Program and can be used for custom setting. Parameters 19.01 to 19.12 can be used in the same way, but are not stored in the flash.
- Connect the user constant to a block as usual by the input selection parameter.

The user constants can be changed while the Adaptive Program is running. They may have values from -32767 to 32767.

Parameter value as a boolean input

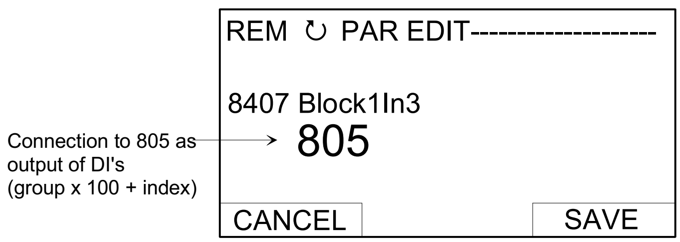
How the block handles the input

The block:

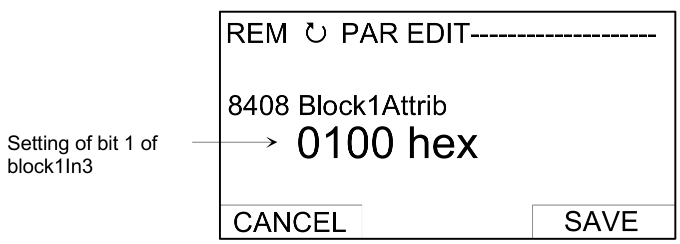
- reads the selected value as an integer,
- uses the bit defined by the bit field as the boolean input and
- interprets bit value 1 as true and 0 as false.

Example:
The figure below shows the value of *Block1In3 (84.07)* when the input is connected to DI2. All digital inputs are available in *DI StatWord (8.05).* Bit 0 corresponds to DI1 and bit 1 to DI2.

Display of panel



Display of panel



Note:
The parameter selected as an input should have a packed boolean value (binary data word).

Constant as a boolean input

How to set and connect the input

- Scroll to the input selection parameter of the block and switch to edit mode (Enter).
- Give the constant. If boolean value true is needed, set the constant to 1. If boolean value false is needed, set to 0.
- Accept by Enter.
- Scroll to attribute parameter (BlockxAttrib).
- Set the bit for constant attribute of this input in BlockxAttrib parameter.
- Accept by Enter.

String input

How to select the input

With the EVENT block the text from fault, alarm or notice lists will be selected. To change the text DriveWindow and SDCS-COM-8 are required.

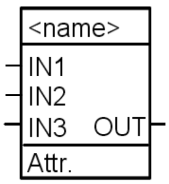
Function blocks

General

Each of the 16 function blocks has three input parameters IN1 to IN3, which can be connected to the firmware, outputs of other function blocks or constants. Boolean values are interpreted like this:

- 1 as true and
- 0 as false.

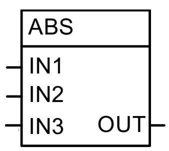
A 4th parameter is used for the attributes of the inputs. The attribute has to be edited manually, if the functions blocks are edited with the DCS800 Control Panel, DriveWindow or DriveWindow Light. The attribute is set automatically when DWL AP is used. The output OUT can connected with the inputs of function blocks. To write output values into firmware parameters connect the necessary output pointer (group 86) to the desired parameter.



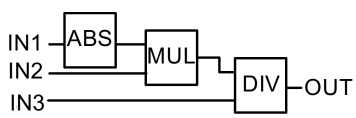
ABS

Type Arithmetical function

Illustration



Operation OUT is the absolute value of IN1 multiplied by IN2 and divided by IN3.
OUT = |IN1| * IN2 / IN3



Connections

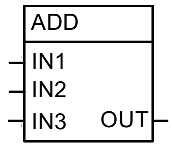
IN1, IN2 and IN3:	16 bit integer (15 bits + sign)
OUT:	16 bit integer (15 bits + sign)

ADD

Type

Arithmetical function

Illustration



Operation

OUT is the sum of the inputs.
OUT = IN1 + IN2 + IN3

Connections

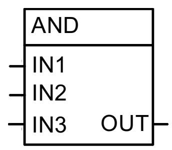
IN1, IN2 and IN3: 16 bit integer (15 bits + sign)
OUT: 16 bit integer (15 bits + sign)

AND

Type

Logical function

Illustration



Operation

OUT is true if all connected inputs are true. Otherwise the OUT is false. Truth table:

IN1	**IN2**	**IN3**	**OUT** (binary)	**OUT** (value on display)
0	0	0	False (All bits 0)	0
0	0	1	False (All bits 0)	0
0	1	0	False (All bits 0)	0
0	1	1	False (All bits 0)	0
1	0	0	False (All bits 0)	0
1	0	1	False (All bits 0)	0
1	1	0	False (All bits 0)	0
1	1	1	True (All bits 1)	-1

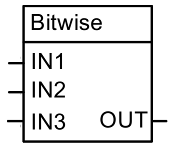
Connections

IN1, IN2 and IN3: boolean
OUT: 16 bit integer (packed boolean)

Bitwise

Type Logical function

Illustration

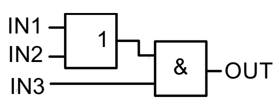


Operation The block compares bits of three 16 bit word inputs and forms the output bits as follows:
OUT = (IN1 OR IN2) AND IN3.
Example:
Single bit:

IN1	IN2	IN3	OUT
0	0	0	0
0	1	0	0
1	0	0	0
1	1	0	0
0	0	1	0
0	1	1	1
1	0	1	1
1	1	1	1

Example:
Whole word:

Input [word]		bits 15															0		Output [word]
20518	=> IN1	0	**1**	0	**1**	0	0	0	0	0	0	**1**	0	0	**1**	**1**	0		
4896	=> IN2	0	0	0	**1**	0	0	**1**	**1**	0	0	0	0	0	0	0	0		
17972	=> IN3	0	**1**	0	0	0	**1**	**1**	0	0	0	**1**	**1**	0	**1**	0	0		
		0	**1**	0	0	0	0	**1**	0	0	0	**1**	0	0	**1**	0	0	=> OUT	16932



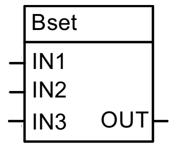
Connections

IN1, IN2 and IN3: 16 bit integer (packed boolean)
OUT: 16 bit integer (packed boolean)

Bset

Type Logical function

Illustration



Operation With Bset it is possible to set the value of a certain bit in a word. Connect the word to be processed at IN1. Define the number of the bit to be changed at IN2. Define the desired bit value at IN3 (1 for true and 0 for false). OUT is the result of the operation.

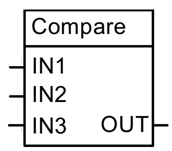
Connections

IN1:	16-bit integer (packed boolean); word to be processed e.g. *MainCtrlWord (7.01)*
IN2:	0 … 15; bit to be changed
IN3:	boolean; desired bit value
OUT:	16-bit integer (packed boolean), result

Compare

Type Arithmetical function

Illustration

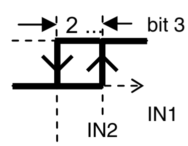


Operation Output bits 0, 1 and 2 (bits 4 ... 15 are not used):

- If IN1 > IN2 ⇒ OUT = 001 OUT bit 0 is true,
- if IN1 = IN2 ⇒ OUT = 010 OUT bit 1 is true and
- if IN1 < IN2 ⇒ OUT = 100 OUT bit 2 is true.

Output bit 3:

- If IN1 > IN2, OUT = 1ddd	OUT bit 3 is true and remains true until IN1 < (IN2 - IN3), after which bit 3 is false.



Output bit 4...15: not used

OUT integer value, which is shown on display, is the sum of the bits:

bit 3	**bit 2**	**bit 1**	**bit 0**	**OUT** (value on display)
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	1	0	0	4
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	1	0	0	12

Connections

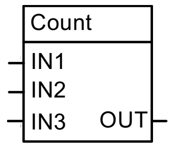
IN1, IN2 and IN3:	16 bit integer values (15 bits + sign)
OUT:	16 bit integer (packed boolean)

Count

Type

Arithmetical function

Illustration



Operation

The counter counts the rising edges of IN1. Rising edges at IN2 reset the counter. IN3 limits OUT. IN3 > 0: OUT increases to the set limit. IN3 < 0: OUT increases up to the absolute maximum value (32768). When the maximum value is reached the output will be set to 0 and the counter starts counting from zero.

Connections

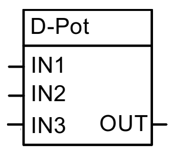
IN1:	boolean; counts rising edges
IN2:	boolean; reset input (high active)
IN3:	16 bit integer (15 bit + sign); limit
OUT:	15 bit integer (15 bit + sign); shows the counted value

D-Pot

Type

Arithmetical function

Illustration



Operation

IN1 increases OUT. IN2 decreases OUT. The absolute value of IN3 is the ramp time in ms which is needed to increase OUT from 0 to 32767. With positive IN3 the output range is limited from 0 to 32767. With negative IN3 the output range is between -32767 and +32767. If both IN1 and IN2 are true, IN2 overwrites IN1.

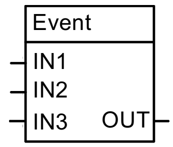
Connections

IN1:	boolean; ramp up
IN2:	boolean; ramp down
IN3:	16 bit integer (15 bit + sign); ramp time scale
OUT:	16 bit integer (15 bit + sign); ramp value

Event

Type Display function

Illustration



Operation IN1 triggers the event. IN2 selects the fault, alarm or notice. IN3 is the event delay in ms.

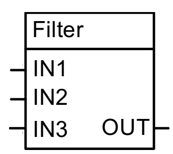
IN1	Activation input (boolean)			
	0 -> 1	trigger event		
	0	block deactivated		
IN2	Selection of the message to be displayed. There exist 15 different messages, which are selected by using numbers. The default message is shown in the brackets. It can be changed by means of string parameters.			
	Alarms	Faults	Notices	String parameters
	301 (APAlarm1)	601 (APFault1)	801 (………..)	*String1 (85.11)*
	302 (APAlarm2)	602 (APFault2)	802 (………..)	*String2 (85.12)*
	303 (APAlarm3)	603 (APFault3)	803 (………..)	*String3 (85.13)*
	304 (APAlarm4)	604 (APFault4)	804 (………..)	*String4 (85.14)*
	305 (APAlarm5)	605 (APFault5)	805 (………..)	*String5 (85.15)*

Connections
IN1: boolean
IN2: Text of alarm, fault or notice. Must be defined via *String1 (85.11)* to *String5 (85.15)* and connected to IN2
IN3: 16 bit integer
OUT: not used

Filter

Type Arithmetical function

Illustration



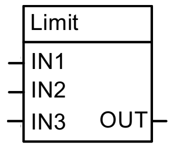
Operation OUT is the filtered value of IN1. IN2 is the filter time in ms.
OUT = IN1 (1 - $e^{-t/IN2}$)
Note:
The internal calculation uses 32 bits accuracy to avoid offset errors.

Connections
IN1: 16 bit integer (15 bits + sign); value to be filtered
IN2: 16 bit integer (15 bits + sign); filter time in ms
IN3: not used
OUT: 16 bit integer (15 bits + sign); filtered value

Limit

Type Logical function

Illustration



Operation The value, connected to IN1 will be limited with IN2 as upper limit and IN3 as lower limit. OUT is the limited input value. OUT stays 0, if IN3 is >= IN2.

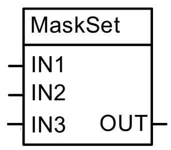
Connections

IN1: 16 bit integer (15 bits + sign); value to be limited
IN2: 16 bit integer (15 bits + sign); upper limit
IN3: 16 bit integer (15 bits + sign); lower limit
OUT: 16 bit integer (15 bits + sign); limited value

MaskSet

Type Logical function

Illustration



Operation The block sets or resets the bits in IN1 and IN2.
Example:

IN3 = set

IN1	IN2	IN3	OUT
0	0	True	0
1	0	True	**1**
1	1	True	**1**
0	1	True	**1**

IN3 = reset

IN1	IN2	IN3	OUT
0	0	False	0
1	0	False	**1**
1	1	False	0
0	1	False	0

Example:
Whole word with IN3 = set

Input [word]		bits 15															0		Output [word]
26214	=> IN1	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0		
-13108	=> IN2	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0		
		1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	=> OUT	-4370

Whole word with IN3 = reset

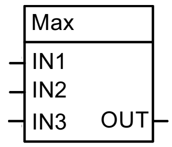
Connections

IN1: 16 bit integer (packed boolean); word input
IN2: 16 bit integer (packed boolean); word input
IN3: boolean; set / reset IN2 in IN1
OUT: 16 bit integer (packed boolean); result

Max

Type Arithmetical function

Illustration



Operation OUT is the highest input value.
OUT = MAX (IN1, IN2, IN3)
Note:
An open input will ignored.

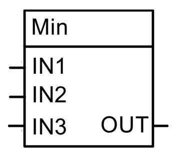
Connections

IN1, IN2 and IN3:	16 bit integer (15 bits + sign)
OUT:	16 bit integer (15 bits + sign)

Min

Type Arithmetical function

Illustration



Operation OUT is the lowest input value.
OUT = MIN (IN1, IN2, IN3)

Note: An open input will be set to as zero.

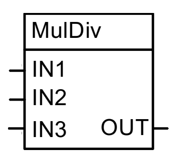
Connections

Input IN1, IN2 and IN3:	16 bit integer values (15 bits + sign)
Output OUT:	16 bit integer (15 bits + sign)

MulDiv

Type Arithmetical function

Illustration



Operation OUT is the IN1 multiplied with IN2 and divided by IN3.
OUT = (IN1 * IN2) / IN3

Connections

Input IN1, IN2 and IN3:	16 bit integer values (15 bits + sign)
Output OUT:	16 bit integer (15 bits + sign)

NotUsed

Type -

Illustration

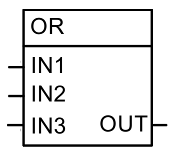
Operation Block is not enabled and not working, default

Connections -

OR

Type Logical function

Illustration



Operation OUT is true if any of the connected inputs is true. Otherwise the OUT is false. Truth table:

IN1	**IN2**	**IN3**	**OUT** (binary)	**OUT** (value on display)
0	0	0	False (All bits 0)	0
0	0	1	True (All bits 1)	-1
0	1	0	True (All bits 1)	-1
0	1	1	True (All bits 1)	-1
1	0	0	True (All bits 1)	-1
1	1	0	True (All bits 1)	-1
1	1	1	True (All bits 1)	-1

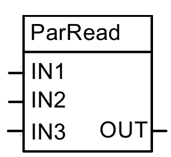
Connections

IN1, IN2 and IN3: boolean values
OUT: 16 bit integer value (packed boolean)

ParRead

Type Parameter function

Illustration



Operation OUT shows the value of a parameter, which is defined with IN1 as group and IN2 as index.
Example:
Reading *AccTime1 (22.01)*:
IN1 = 22 and IN2 = 01

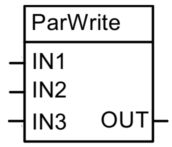
Connections

IN1: 16 bit integer (15 bits + sign); group
IN2: 16 bit integer (15 bits + sign); index
IN3: not used
OUT: 16 bit integer (15 bits + sign); parameter value

ParWrite

Type
Parameter function

Illustration



Operation
Value of IN1 is written into a parameter defined by IN2 as group * 100 + index, e.g. MainCtrlWord (7.01) == 701. The block will be activated with a change of IN1. IN3 determines if the value is saved in the flash.
Attention:
Cyclic saving of values in the flash will damage it! Do not set IN3 constantly to true!
OUT gives the error code, if parameter access is denied.
Example:
Set AccTime1 (22.01) = 150, not saving into flash:
IN1 = 150, desired value
IN2 = 2201, this must be a defined as a constant and not as a parameter
IN3 = false

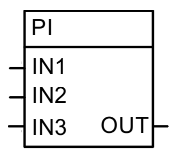
Connections

IN1:	16 bit integer (15 bits + sign); desired value
IN2:	16 bit integer (15 bits + sign); group * 100 + index
IN3:	boolean; true = save in flash, false = don't save in flash
OUT:	16 bit integer (packed boolean); error code

PI

Type
Arithmetical controller

Illustration



Operation
OUT is IN1 multiplied by (IN2 / 100) plus integrated IN1 multiplied by (IN3 / 100).

$$O = I1 * I2/100 + \left(I3/100\right) * \int I1$$

Note:
The internal calculation uses 32 bits accuracy to avoid offset errors.

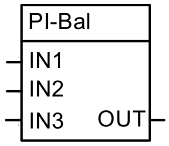
Connections

IN1:	16 bit integer (15 bit + sign); error (e.g. speed error)
IN2:	16 bit integer (15 bit + sign); p-part (30 == 0.3, 100 == 1)
IN3:	16 bit integer (15 bit + sign); i-part (250 == 2.5, 5,000 == 50)
OUT:	16 bit integer (15 bits + sign); the range is limited from -20,000 to +20,000

PI-Bal

Type Arithmetical function

Illustration



Operation The PI-Bal block initializes the PI block. The PI-Bal block must follow directly behind the PI block and can only be used together with the PI block.
When IN1 is true, the PI-Bal block writes the value of IN2 directly into OUT of the PI block. When IN1 is false, the PI-Bal block releases OUT of the PI block. Normal operation continues starting with the set output value - bumpless transition.

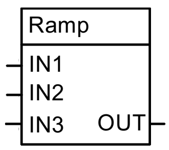
Connections

IN1:	boolean; true = balance PI block, false = no balancing
IN2:	16 bit integer (15 bits + sign); balance value
IN3:	not used
OUT:	affects PI block

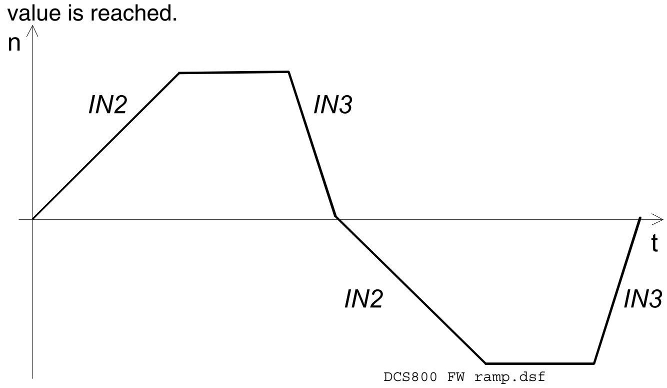
Ramp

Type Arithmetical function

Illustration



Operation IN1 is the input. IN2 and IN3 are the times. OUT increases or decreases until the input value is reached.



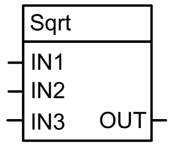
Connections

IN1:	16 bit integer (15 bit + sign); ramp input
IN2:	16 bit integer (15 bit + sign); ramp up time in ms (related to 20,000)
IN3:	16 bit integer (15 bit + sign); ramp down time in ms, (related to 20,000)
OUT:	16 bit integer (15 bit + sign); ramp output

Sqrt

Type
Arithmetical function

Illustration



Operation
OUT is the square root of IN1 * IN2. With IN3 = true IN1 and IN2 are read as absolute values:

$$OUT = \sqrt{|IN1| * |IN2|}$$

With IN3 = false OUT is set to zero if IN1 * IN2 is negative:

$$OUT = \sqrt{IN1 * IN2}; \quad if\ IN1 * IN2 \geq 0$$

$$OUT = 0 \qquad\qquad\qquad if\ IN1 * IN2 < 0$$

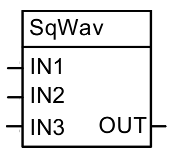
Connections

IN1:	16 bit integer (15 bits + sign)
IN2:	16 bit integer (15 bits + sign)
IN3:	boolean
OUT:	16 bit integer

SqWav

Type
Arithmetical function

Illustration



Operation
OUT alternates between the value of IN3 and zero (0), if the block is enabled with IN1 = true. The period is set with IN2 in ms.

Connections

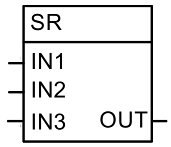
IN1:	boolean; true = enable SqWav, false = disable SqWav
IN2:	16 bit integer; cycle time in ms
IN3:	16 bit integer (15 bits + sign); height of square wave
OUT:	16 bit integer (15 bits + sign); square wave

SR

Type

Logical function

Illustration



Operation

Set/reset block. IN1 (S) sets OUT. IN2 (R) or IN3 (R) reset OUT. If IN1, IN2 and IN3 are false, the current value remains at OUT. The SR is reset dominant. Truth table:

IN1	**IN2**	**IN3**	**OUT** (binary)	**OUT** (value on display)
0	0	0	no change	no change
0	0	1	false (all bits 0)	0
0	1	0	false (all bits 0)	0
0	1	1	false (all bits 0)	0
1	0	0	true (all bits 1)	-1
1	0	1	false (all bits 0)	0
1	1	0	false (all bits 0)	0
1	1	1	false (all bits 0)	0

Connections

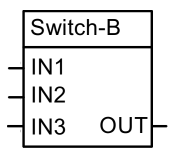
IN1, IN2 and IN3: boolean
OUT: 16 bit integer (15 bits + sign)

Switch-B

Type

Logical function

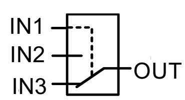
Illustration



Operation

OUT is equal to IN2 if IN1 is true. OUT is equal to IN3 if IN1 is false.

IN1	**OUT**
0	= IN3
1	= IN2



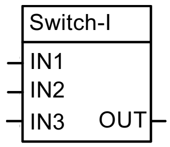
Connections

IN1: boolean (only bit 0 is valid)
IN2 and IN3: boolean
OUT: 16 bit integer (packed boolean)

Switch-I

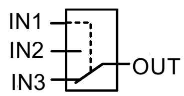
Type Arithmetical function

Illustration



Operation OUT is equal to IN2 if IN1 is true and equal to IN3 if IN1 is false.

IN1			OUT
0			= IN3
1			= IN2



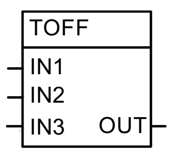
Connections

IN1: boolean (only bit 0 is valid)
IN2 and IN3: 16 bit integer (15 bits + sign)
OUT: 16 bit integer (15 bits + sign)

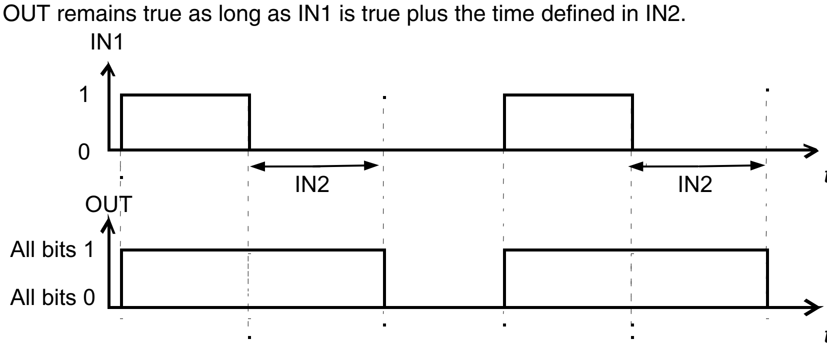
TOFF

Type Logical function

Illustration



Operation OUT is true when IN1 is true. OUT is false when IN1 has been false for a time >= IN2. OUT remains true as long as IN1 is true plus the time defined in IN2.



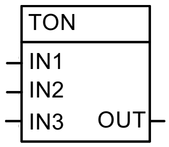
Connections

IN1: boolean, input
IN2: 16 bit integer; delay time in ms (IN3 = false) or s (IN3 = true)
IN3: boolean; determines unit of time
OUT: 16 bit integer (packed boolean); result with values on display: True = -1, false = 0

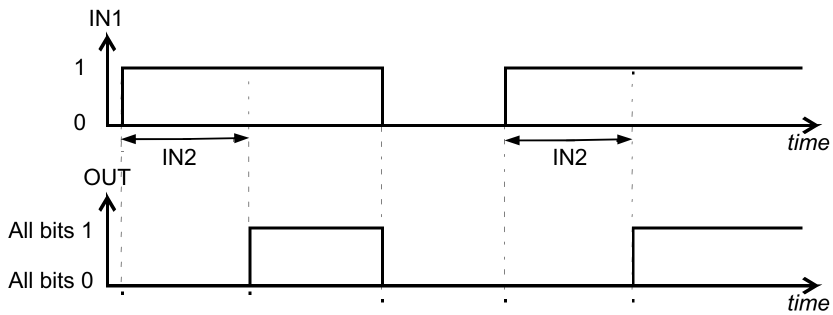
TON

Type Logical function

Illustration



Operation OUT is true when IN1 has been true for a time equal or longer than IN2.



Values on display: True = -1, false = 0
With IN3 = False the delay time of IN2 is scaled in ms,
with IN3 = True the delay time of IN2 is scaled in s

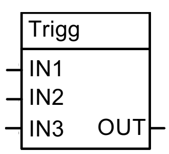
Connections

Input IN1 and IN3:	boolean value
Input IN2:	16 bit integer value (15 bits + sign)
Output OUT:	16 bit integer value (packed boolean)

Trigg

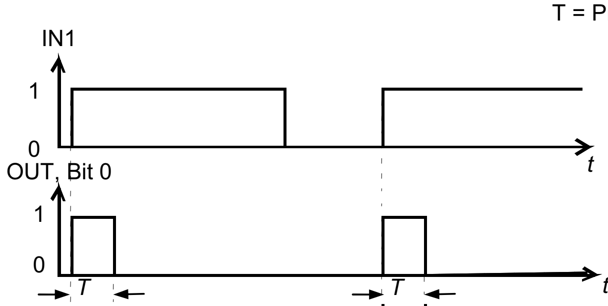
Type Logical function

Illustration



Operation The rising edge of IN1 sets OUT bit 0 for one program cycle.
The rising edge of IN2 sets OUT bit 1 for one program cycle.
The rising edge of IN3 sets OUT bit 2 for one program cycle.

T = Program cycle



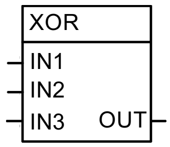
Connections

IN1, IN2 and IN3:	boolean
OUT:	16 bit integer (packed boolean)

XOR

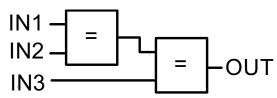
Type Logical function

Illustration



Operation OUT is true if one input is true, otherwise OUT is false. Truth table:

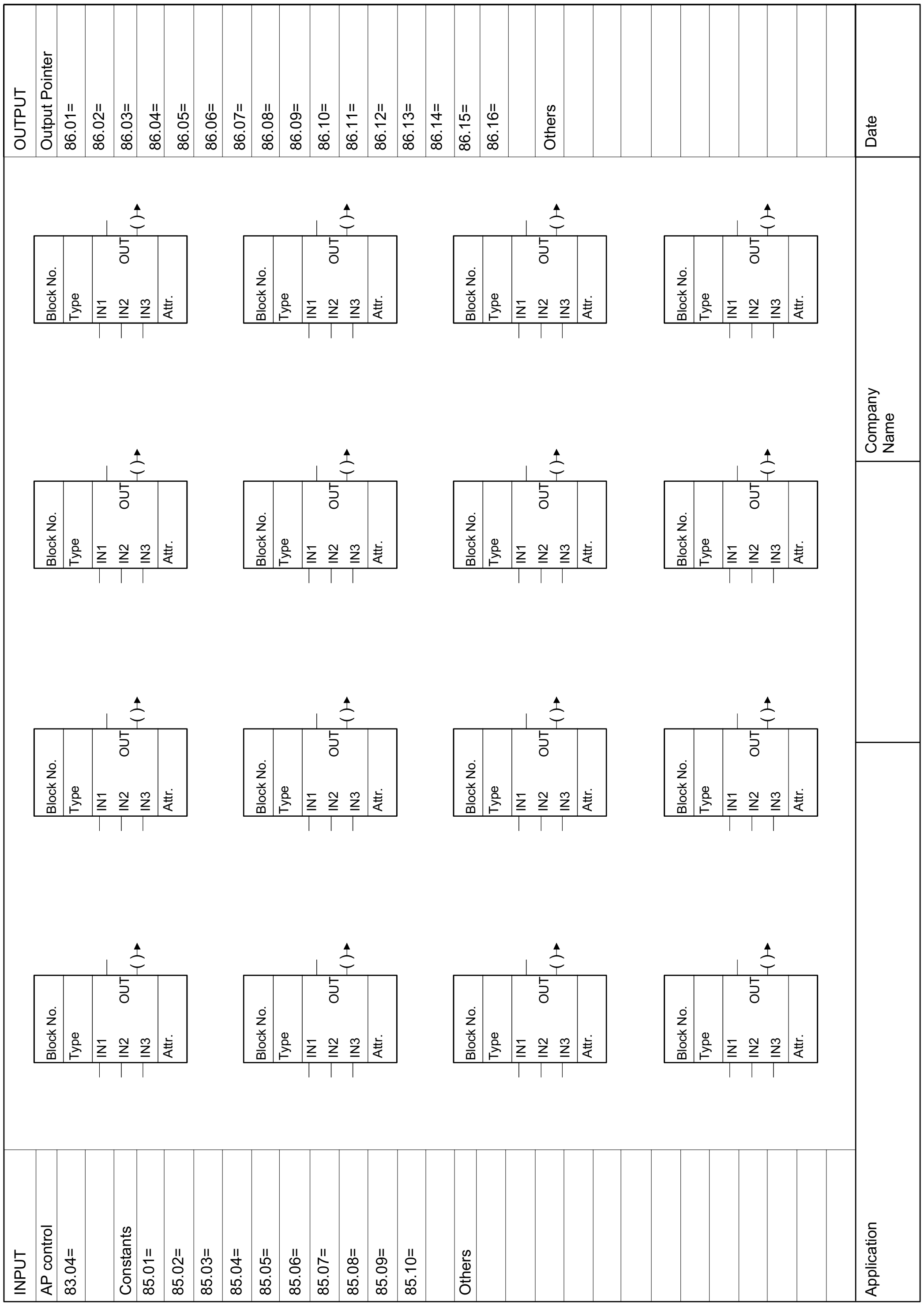
IN1	**IN2**	**IN3**	**OUT** (binary)	**OUT** (value on display)
0	0	0	false (all bits 0)	0
0	0	1	true (all bits 1)	-1
0	1	0	true (all bits 1)	-1
0	1	1	false (all bits 0)	0
1	0	0	true (all bits 1)	-1
1	0	1	false (all bits 0)	0
1	1	0	false (all bits 0)	0
1	1	1	true (all bits 1)	-1



Connections IN1, IN2 and IN3: boolean
OUT: 16 bit integer value (packed boolean)

Diagram

Blank block diagram sheet on which the Adaptive Program can be documented.



Signal and parameter list

Signals and parameters

This chapter contains all signals and parameters.

Signal groups list

Signals are measured and calculated actual values of the drive. This includes the control-, status-, limit-, fault- and alarm words. The drive's signals can be found in groups 1 to 9. None of the values inside these groups is stored in the flash and thus volatile.

Note:
All signals in group 7 can be written to by means of DWL, DCS800 Control Panel, Adaptive Program, application program or overriding control.

The following table gives an overview of all signal groups:

Group	Description	Comment
1	Physical actual values	
2	Speed controller signals	
3	Reference actual values	
4	Information	self identification
5	Analog I/O	
6	Drive logic signals	
7	Control words	command words
8	Status / limit words	detection on operation and limits
9	Fault / alarm words	diagnosis information

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
1.08	**MotTorq (motor torque)** Motor torque in percent of *MotNomTorque (4.23)*: – Filtered by means of a 6th order FIR filter (sliding average filter), filter time is 1 mains voltage period. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
2.17	**SpeedRefUsed (used speed reference)** Used speed reference selected with: – *Ref1Mux (11.02)* and *Ref1Sel (11.03)* or – *Ref2Mux (11.12)* and *Ref2Sel (11.06)* **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C

Sample of signals

All signals are read-only. However the overriding control can write to the control words, but it only affects the RAM.

Min., max., def.:
Minimum, maximum and default values are not valid for groups 1 to 9.

Unit:
Shows the physical unit of a signal, if applicable. The unit is displayed in the DCS800 Control Panel and PC tools.

E/C:
By means of *USI Sel (16.09)* it is possible to change between compact **(C)** and extended **(E)** signal and parameter list. The compact list contains only signals and parameters used for a typical commissioning.

Group.Index:
Signal and parameter numbers consists of group number and its index.

Integer Scaling:
Communication between the drive and the overriding control uses 16 bit integer values. The overriding control has to use the information given in integer scaling to read the value of the signal properly.
Example1:
If *MotTorq (1.08)* is read from the overriding control an integer value of 100 corresponds to 1 % torque.
Example2:
If *SpeedRefUsed (2.17)* is read from the overriding control 20.000 equals the speed (in rpm) shown in *SpeedScaleAct (2.29)*.

Type:
The data type is given with a short code:
I = 16-bit integer value (0, …, 65536)
SI = 16-bit signed integer value (-32768, …, 32767)
C = text string (ENUM)

Volatile:
Y = values are NOT stored in the flash, they will be lost when the drive is de-energized
N = values are stored in the flash, they will remain when the drive is de-energized

Parameter groups list

This chapter explains the function and valid values or selections for all parameters. They are arranged in groups by their function. The following table gives an overview of all parameter groups:

Group	Description
10	Start / stop select
11	Speed reference inputs
12	Constant speeds
13	Analog inputs
14	Digital outputs
15	Analog outputs
16	System control inputs
19	Data storage
20	Limits
21	Start / stop
22	Speed ramp
23	Speed reference
24	Speed control
25	Torque reference
26	Torque reference handling
30	Fault functions
31	Motor 1 temperature
34	DCS800 Control Panel display
40	PID control
42	Brake control
43	Current control
44	Field excitation
45	Field converter settings
47	12-pulse operation
49	Shared motion
50	Speed measurement
51	Fieldbus
52	Modbus
60…69	Application program parameters
70	DDCS control
71	Drivebus
83	Adaptive Program control
84	Adaptive Program
85	User constants
86	Adaptive Program outputs
88	Internal
90	Receiving data sets addresses 1
91	Receiving data sets addresses 2
92	Transmit data sets addresses 1
93	Transmit data sets addresses 2
94	DCSLink control
97	Measurement
98	Option modules
99	Start-up data

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
20.07	**TorqMaxSPC (maximum torque speed controller)** Maximum torque limit - in percent of *MotNomTorque (4.23)* - at the output of the speed controller: – *TorqRef2 (2.09)* **Note:** The used torque limit depends also on the converter's actual limitation situation (e.g. other torque limits, current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0	325	325	%	E
23.01	**SpeedRef (speed reference)** Main speed reference input for the speed control of the drive. Can be connected to *SpeedRefUsed (2.17)* via: – *Ref1Mux (11.02)* and *Ref1Sel (11.03)* or – *Ref2Mux (11.12)* and *Ref2Sel (11.06)* Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Int. Scaling: (2.29) Type: SI Volatile: Y**	-10000	10000	0	rpm	C

Sample of parameters

Parameter changes by DCS800 Control Panel, DriveWindow or DriveWindow Light are stored in the flash. Changes made by the overriding control are only stored in the RAM.

Min., max., def.:
Minimum and maximum value or selection of parameter.
Default value or default selection of parameter.

Unit:
Shows the physical unit of a parameter, if applicable. The unit is displayed in the DCS800 Control Panel and PC tools.

E/C:
By means of *USI Sel (16.09)* it is possible to change between compact **(C)** and extended **(E)** signal and parameter list. This influences parameter display of DCS800 Control Panel. The compact list contains only signals and parameters used for a typical commissioning.

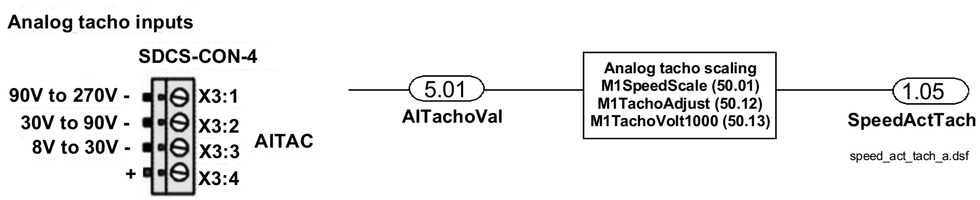
Group.Index:
Signal and parameter numbers consists of group number and its index.

Integer Scaling:
Communication between the drive and the overriding control uses 16 bit integer values. The overriding control has to use the information given in integer scaling to change the value of the parameter properly.
Example1:
If *TorqMaxSPC (20.07)* is written to from the overriding control an integer value of 100 corresponds to 1 %.
Example2:
If *SpeedRef (23.01)* is written to from the overriding control 20.000 equals the speed (in rpm) shown in *SpeedScaleAct (2.29)*.

Type:
The data type is given with a short code:
I = 16-bit integer value (0, …, 65536)
SI = 16-bit signed integer value (-32768, …, 32767)
C = text string (ENUM)

Volatile:
Y = values are NOT stored in the flash, they will be lost when the drive is de-energized
N = values are stored in the flash, they will remain when the drive is de-energized

Signal and parameter list

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 1	**Physical actual values**					
1.01	**MotSpeedFilt (filtered motor speed)** Filtered actual speed feedback: – Choose motor speed feedback with *M1SpeedFbSel (50.03)* – Filtered with 1 s and – *SpeedFiltTime (50.06)* **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
1.02	**SpeedActEMF (speed actual EMF)** Actual speed calculated from EMF. **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
1.03	**SpeedActEnc (speed actual encoder 1)** Actual speed measured with pulse encoder 1. **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
1.04	**MotSpeed (motor speed)** Actual motor speed: – Choose motor speed feedback with *M1SpeedFbSel (50.03)*. If *M1SpeedFbSel (50.03)* is set to **External** the signal is updated by Adaptive Program, application program or overriding control. – *SpeedFiltTime (50.06)* **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
						
1.05	**SpeedActTach (speed actual tacho)** Actual speed measured with analog tacho. **Note:** This value is only valid, if an analog tacho is connected! **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
1.06	**MotCur (motor current)** Relative actual motor current in percent of *M1NomCur (99.03)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
1.07	**MotTorqFilt (filtered motor torque)** Relative filtered motor torque in percent of *MotNomTorque (4.23)*: – Filtered by means of a 6th order FIR filter (sliding average filter), filter time is 1 mains voltage period plus – *TorqActFiltTime (97.20)* **Note:** The cycle time is 20 ms **Note:** The value is calculated the following way: $MotTorqFilt\,(1.07) = \frac{FluxRefFldWeak\,(3.24) * MotCur\,(1.06)}{100}$ *with* $FluxRefFldWeak\,(3.24) = FluxMax * \frac{M1BaseSpeed\,(99.04)}{\lvert MotSpeed\,(1.04)\rvert};\ for\ n > M1BaseSpeed\,(99.04)$ *or* $FluxRefFldWeak\,(3.24) = FluxMax = 100\%;\ for\ n \leq M1BaseSpeed\,(99.04)\ or\ M1UsedFexType\,(99.12) = NotUsed$ **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
1.08	**MotTorq (motor torque)** Motor torque in percent of *MotNomTorque (4.23)*: – Filtered by means of a 6th order FIR filter (sliding average filter), filter time is 1 mains voltage period. **Note:** The cycle time is 20 ms **Note:** The value is calculated the following way: $MotTorq\,(1.08) = \frac{FluxRefFldWeak\,(3.24) * MotCur\,(1.06)}{100}$ *with* $FluxRefFldWeak\,(3.24) = FluxMax * \frac{M1BaseSpeed\,(99.04)}{\lvert MotSpeed\,(1.04)\rvert};\ for\ n > M1BaseSpeed\,(99.04)$ *or* $FluxRefFldWeak\,(3.24) = FluxMax = 100\%;\ for\ n \leq M1BaseSpeed\,(99.04)\ or\ M1UsedFexType\,(99.12) = NotUsed$ **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
1.09	**CurRipple (current ripple)** Relative current ripple monitor output in percent of *M1NomCur (99.03).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
1.10	**CurRippleFilt (filtered current ripple)** Relative filtered current ripple monitor output in percent of *M1NomCur (99.03):* – Filtered with 200 ms **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
1.11	**MainsVoltActRel (relative actual mains voltage)** Relative actual mains voltage in percent of *NomMainsVolt (99.10).* **Int. Scaling: 100 == 1 % Type: I Volatile: Y**	-	-	-	%	C
1.12	**MainsVoltAct (actual mains voltage)** Actual mains voltage: – Filtered with 10 ms **Int. Scaling: 1 == 1 V Type: I Volatile: Y**	-	-	-	V	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
1.13	**ArmVoltActRel (relative actual armature voltage)** Relative actual armature voltage in percent of *M1NomVolt (99.02)*. **Note:** the value is also influenced by *AdjUDC (97.23)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
1.14	**ArmVoltAct (actual armature voltage)** Actual armature voltage: – Filtered with 10 ms **Note:** the value is also influenced by *AdjUDC (97.23)* **Int. Scaling: 1 == 1 V Type: SI Volatile: Y**	-	-	-	V	C
1.15	**ConvCurActRel (relative actual converter current [DC])** Relative actual converter current in percent of *ConvNomCur (4.05).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
1.16	**ConvCurAct (actual converter current [DC])** Actual converter current: – Filtered with 10 ms **Int. Scaling: 1 == 1 A Type: SI Volatile: Y**	-	-	-	A	C
1.17	**EMF VoltActRel (relative actual EMF)** Relative actual EMF in percent of *M1NomVolt (99.02):* *EMF VoltActRel (1.17).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
1.18	**Unused**					
1.19	**Unused**					
1.20	**Mot1TempCalc (motor 1 calculated temperature)** – Motor 1 calculated temperature from motor thermal model in percent - see *M1AlarmLimLoad (31.03)* and *M1FaultLimLoad (31.04)*. Used for motor overtemperature protection. – *M1AlarmLimLoad (31.03)* – *M1FaultLimLoad (31.04)* **Int. Scaling: 100 == 1 % Type: I Volatile: Y**	-	-	-	%	E
1.21	**Mot2TempCalc (motor 2 calculated temperature)** – Motor 2 calculated temperature from motor thermal model in percent - see *M2AlarmLimLoad (49.33)* and *M2FaultLimLoad (49.34)*. Used for motor overtemperature protection. – *M2AlarmLimLoad (49.33)* – *M2FaultLimLoad (49.34)* **Int. Scaling: 100 == 1 % Type: I Volatile: Y**	-	-	-	%	E
1.22	**Mot1TempMeas (motor 1 measured temperature)** Motor 1 measured temperature. Used for motor overtemperature protection: – Unit depends on setting of *M1TempSel (31.05):* 0 = **NotUsed** - 1 = **1 to 6 PT100** °C 2 = **PTC** Ω **Int. Scaling: 1 == 1 °C / 1 Ω / 1 Type: I Volatile: Y**	-	-	-	°C / Ω / -	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
1.23	**Mot2TempMeas (motor 2 measured temperature)** Motor 2 measured temperature. Used for motor overtemperature protection: – Unit depends on setting of *M2TempSel (49.35):* 0 = **NotUsed** - 1 = **1 to 6 PT100** °C 2 = **PTC** Ω **Int. Scaling: 1 == 1 °C / 1 Ω / 1 Type: I Volatile: Y**	-	-	-	°C / Ω / -	E
1.24	**BridgeTemp (actual bridge temperature)** Actual bridge temperature in degree centigrade. **Int. Scaling: 1 == 1 °C Type: I Volatile: Y**	-	-	-	°C	C
1.25	**CtrlMode (control mode)** Used control mode: – *see TorqSel (26.01)* 0 = **NotUsed** - 1 = **SpeedCtrl** speed control 2 = **TorqCtrl** torque control 3 = **CurCtrl** current control 4 = **VoltCtrl** voltage control, if *CtrlModeSel (43.08)* = **PowerSupply2** **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
1.26	**Unused**					
1.27	**Unused**					
1.28	**Unused**					
1.29	**Mot1FldCurRel (motor 1 relative actual field current)** Motor 1 relative field current in percent of *M1NomFldCur (99.11).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
1.30	**Mot1FldCur (motor 1 actual field current)** Motor 1 field current: – Filtered with 500 ms **Int. Scaling: 10 == 1 A Type: SI Volatile: Y**	-	-	-	A	C
1.31	**Mot2FldCurRel (motor 2 relative actual field current)** Motor 2 relative field current in percent of *M2NomFldCur (49.05).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
1.32	**Mot2FldCur (motor 2 actual field current)** Motor 2 field current: – Filtered with 500 ms **Int. Scaling: 10 == 1 A Type: SI Volatile: Y**	-	-	-	A	E
1.33	**ArmCurActSl (12-pulse slave actual armature current)** Actual armature current of 12-pulse slave: – Valid in 12-pulse master only – Valid for 12-pulse parallel only **Int. Scaling: 1 == 1 A Type: SI Volatile: Y**	-	-	-	A	E
1.34	**Unused**	-	-	-	-	E
1.35	**ArmCurAll (12-pulse parallel master and slave actual armature current)** Sum of actual armature current for 12-pulse master and 12-pulse slave: – Filtered with 10 ms – Valid in 12-pulse master only – Valid for 12-pulse parallel only **Int. Scaling: 1 == 1 A Type: SI Volatile: Y**	-	-	-	A	E
1.36	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
1.37	**DC VoltSerAll (12-pulse serial master and slave actual DC voltage)** Sum of actual armature voltage for 12-pulse master and 12-pulse slave: – Valid in 12-pulse master only – Valid for 12-pulse serial/sequential only **Int. Scaling: 1 == 1 V Type: SI Volatile: Y**	-	-	-	V	E
1.38	**MainsFreqAct (internal mains frequency)** Calculated and internally controlled mains frequency. Output of PLL controller. See also: – *DevLimPLL (97.13)* – *KpPLL (97.14)* – *TfPLL (97.15)* **Int. Scaling: 100 == 1 Hz Type: I Volatile: Y**	-	-	-	Hz	C
1.39	**AhCounter (ampere-hour counter)** Ampere hour counter. **Int. Scaling: 100 == 1kAh Type: I Volatile: Y**	-	-	-	kAh	E
1.40	**Unused**					
1.41	**ProcSpeed (process speed)** Calculated process/line speed: – Scaled with *WinderScale (50.17)* **Int. Scaling: 10 == 1 m/min Type: SI Volatile: Y**	-	-	-	m/min	E
1.42	**SpeedActEnc2 (speed actual encoder 2)** Actual speed measured with pulse encoder 2. **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
Group 2	**Speed controller signals**					
2.01	**SpeedRef2 (speed reference 2)** Speed reference after limiter: – *M1SpeedMin (20.01)* – *M1SpeedMax (20.02)* **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
2.02	**SpeedRef3 (speed reference 3)** Speed reference after speed ramp and jog input. **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
2.03	**SpeedErrNeg (Δn)** Δn = speed actual - speed reference. **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
2.04	**TorqPropRef (proportional part of torque reference)** P-part of the speed controller's output in percent of *MotNomTorque (4.23)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
2.05	**TorqIntegRef (integral part of torque reference)** I-part of the speed controller's output in percent of *MotNomTorque (4.23)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
2.06	**TorqDerRef (derivation part of torque reference)** D-part of the speed controller's output in percent of *MotNomTorque (4.23)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E

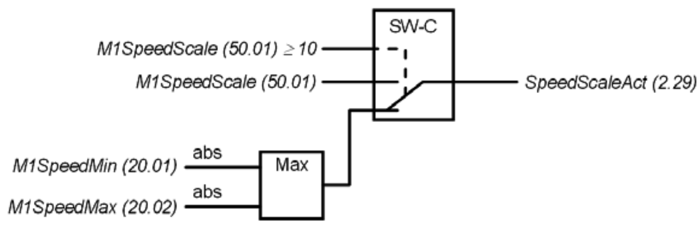
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
2.07	**TorqAccCompRef (torque reference for acceleration compensation)** Acceleration compensation output in percent of *MotNomTorque (4.23)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.08	**TorqRef1 (torque reference 1)** Relative torque reference value in percent of *MotNomTorque (4.23)* after limiter for the external torque reference: – *TorqMaxTref (20.09)* – *TorqMinTref (20.10)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.09	**TorqRef2 (torque reference 2)** Output value of the speed controller in percent of *MotNomTorque (4.23)* after limiter: – *TorqMaxSPC (20.07)* – *TorqMinSPC (20.08)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.10	**TorqRef3 (torque reference 3)** Relative torque reference value in percent of *MotNomTorque (4.23)* after torque selector: – *TorqSel (26.01)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.11	**TorqRef4 (torque reference 4)** = *TorqRef3 (2.10)* + *LoadComp (26.02)* in percent of *MotNomTorque (4.23).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.12	**Unused**					
2.13	**TorqRefUsed (used torque reference)** Relative final torque reference value in percent of *MotNomTorque (4.23)* after torque limiter: – *TorqMax (20.05)* – *TorqMin (20.06)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.14	**TorqCorr (torque correction)** Relative additional torque reference in percent of *MotNomTorque (4.23)*: – *TorqCorrect (26.15)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.16	**dv_dt (dv/dt)** Acceleration/deceleration (speed reference change) at the output of the speed reference ramp. **Int. Scaling: (2.29)/s Type: SI Volatile: Y**	-	-	-	rpm/s	C
2.17	**SpeedRefUsed (used speed reference)** Used speed reference selected with: – *Ref1Mux (11.02)* and *Ref1Sel (11.03)* or – *Ref2Mux (11.12)* and *Ref2Sel (11.06)* **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
2.18	**SpeedRef4 (speed reference 4)** = *SpeedRef3 (2.02)* + *SpeedCorr (23.04).* **Int. Scaling: (2.29) Type: SI Volatile: Y**	-	-	-	rpm	C
2.19	**TorqMaxAll (torque maximum all)** Relative calculated positive torque limit in percent of *MotNomTorque (4.23)*. Calculated from the smallest maximum torque limit, field weakening and armature current limits: – *TorqUsedMax (2.22)* – *FluxRefFldWeak (3.24)* and – *M1CurLimBrdg1 (20.12)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
2.20	**TorqMinAll (torque minimum all)** Relative calculated negative torque limit in percent of *MotNomTorque (4.23)*. Calculated from the largest minimum torque limit, field weakening and armature current limits: – *TorqUsedMax (2.22)* – *FluxRefFldWeak (3.24)* and – *M1CurLimBrdg2 (20.13)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.21	**Unused**					
2.22	**TorqUsedMax (used torque maximum)** Relative positive torque limit in percent of *MotNomTorque (4.23)*. Selected with: – *TorqUsedMaxSel (20.18)* Connected to torque limiter after *TorqRef4 (2.11)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.23	**TorqUsedMin (used torque minimum)** Relative negative torque limit in percent of *MotNomTorque (4.23)*. Selected with: – *TorqUsedMinSel (20.19)* Connected to torque limiter after *TorqRef4 (2.11)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.24	**TorqRefExt (external torque reference)** Relative external torque reference value in percent of *MotNomTorque (4.23)* after torque reference A selector: – *TorqRefA (25.01)* and – *TorqRefA Sel (25.10)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
2.25	**Unused**					
2.26	**TorqLimAct (actual used torque limit)** Shows parameter number of the actual active torque limit: 0 = **0** no limitation active 1 = **2.19** *TorqMaxAll (2.19)* is active, includes current limits and field weakening 2 = **2.20** *TorqMinAll (2.20)* is active, includes current limits and field weakening 3 = **2.22** *TorqUsedMax (2.22)* selected torque limit is active 4 = **2.23** *TorqUsedMin (2.23)* selected torque limit is active 5 = **20.07** *TorqMaxSPC (20.07)* speed controller limit is active 6 = **20.08** *TorqMinSPC (20.08)* speed controller limit is active 7 = **20.09** *TorqMaxTref (20.09)* external reference limit is active 8 = **20.10** *TorqMinTref (20.10)* external reference limit is active 9 = **20.22** *TorqGenMax (20.22)* regenerating limit is active 10 = **2.08** *TorqRef1 (2.08)* limits *TorqRef2 (2.09)*, see also *TorqSel (26.01)* **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C
2.27	**Unused**					
2.28	**Unused**					

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>2.29</td><td>

SpeedScaleAct (actual used speed scaling)
The value of *SpeedScaleAct (2.29)* equals 20.000 internal speed units.
Currently used speed scaling in rpm for *MotSel (8.09)* = **Motor1**:

- 20.000 speed units == *M1SpeedScale (50.01)*, in case *M1SpeedScale (50.01)* ≥ 10
- 20.000 speed units == maximum absolute value of *M1SpeedMin (20.01)* and *M1SpeedMax (20.02),* in case *M1SpeedScale (50.01)* < 10
  or mathematically:
- If *(50.01)* ≥ 10 then 20.000 == *(50.01)* in rpm
- If *(50.01)* < 10 then 20.000 == Max [|*(20.01)*|, |*(20.02)*|] in rpm



Currently used speed scaling in rpm for *MotSel (8.09)* = **Motor2**:

- 20.000 speed units == *M2SpeedScale (49.22)*, in case *M2SpeedScale (49.22)* ≥ 10
- 20.000 speed units == maximum absolute value of *M2SpeedMin (49.19)* and *M2SpeedMax (49.20),* in case *M2SpeedScale (49.22)* < 10
  or mathematically:
- If *(49.22)* ≥ 10 then 20.000 == *(49.22)* in rpm
- If *(49.22)* < 10 then 20.000 == Max [|*(49.19)*|, |*(49.22)*|] in rpm

Int. Scaling: 1 == 1 rpm Type: SI Volatile: Y
</td><td>-</td><td>-</td><td>-</td><td>rpm</td><td>C</td></tr>
<tr><td>2.30</td><td>

SpeedRefExt1 (external speed reference 1)
External speed reference 1 after reference 1 multiplexer:

- *Ref1Mux (11.02)*

Int. Scaling: (2.29) Type: SI Volatile: Y
</td><td>-</td><td>-</td><td>-</td><td>rpm</td><td>C</td></tr>
<tr><td>2.31</td><td>

SpeedRefExt2 (external speed reference 2)
External speed reference 2 after reference 2 multiplexer:

- *Ref2Mux (11.12)*

Int. Scaling: (2.29) Type: SI Volatile: Y
</td><td>-</td><td>-</td><td>-</td><td>rpm</td><td>C</td></tr>
<tr><td>2.32</td><td>

SpeedRampOut (speed ramp output)
Speed reference after ramp
Int. Scaling: (2.29) Type: SI Volatile: Y
</td><td></td><td></td><td></td><td>rpm</td><td>C</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 3	**Reference actual values**					
3.01	**DataLogStatus (status data logger)** 0 = **NotInit** data logger not initialized 1 = **Empty** data logger is empty 2 = **Running** data logger is running (activated) 3 = **Triggered** data logger is triggered but not filled jet 4 = **Filled** data logger is triggered and filled (data can be uploaded) **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
3.02	**Unused**					
3.03	**SquareWave (square wave)** Output signal of the square wave generator: – *Pot1 (99.15)*, – *Pot2 (99.16)*, – *SqrWavePeriod (99.17)*, – *SqrWaveIndex (99.18)* and – *TestSignal (99.19)* **Int. Scaling: 1==1 Type: SI Volatile: Y**	-	-	-	-	E
3.04	**Unused**					
3.05	**PosCount2Low (position counter low value encoder 2)** Position counter low word pulse encoder 2: – *PosCount2InitLo (50.21)* – Unit depends on setting of *PosCountMode (50.07)*: 0 = **PulseEdges** 1 == 1 pulse edge 1 = **Scaled** 0 == 0° and 65536 == 360° 2 = **Rollover** 0 == 0° and 65536 == 360° **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
3.06	**PosCount2High (position counter high value encoder 2)** Position counter high word pulse encoder 2: – *PosCount2InitHi (50.22)* – Unit depends on setting of *PosCountMode (50.07)*: 0 = **PulseEdges** 1 == 65536 pulse edges 1 = **Scaled** 1 == 1 revolution 2 = **Rollover** always 0 **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
3.07	**PosCountLow (position counter low value encoder 1)** Position counter low word pulse encoder 1: – *PosCountInitLo (50.08)* – Unit depends on setting of *PosCountMode (50.07)*: 0 = **PulseEdges** 1 == 1 pulse edge 1 = **Scaled** 0 == 0° and 65536 == 360° 2 = **Rollover** 0 == 0° and 65536 == 360° **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
3.08	**PosCountHigh (position counter high value encoder 1)** Position counter high word pulse encoder 1: – *PosCountInitHi (50.09)* – Unit depends on setting of *PosCountMode (50.07)*: 0 = **PulseEdges** 1 == 65536 pulse edges 1 = **Scaled** 1 == 1 revolution 2 = **Rollover** always 0 **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
3.09	**PID Out (output PID controller)** PID controller output value in percent of the used PID controller input (see group 40). **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	-	E
3.10	**Unused**					
3.11	**CurRef (current reference)** Relative current reference in percent of *M1NomCur (99.03)* after adaption to field weakening. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
3.12	**CurRefUsed (used current reference)** Relative current reference in percent of *M1NomCur (99.03)* after current limitation: – *M1CurLimBrdg1 (20.12)* – *M1CurLimBrdg2 (20.13)* – *MaxCurLimSpeed (43.17)* to *(43.22)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
3.13	**ArmAlpha (armature α, firing angle)** Firing angle (α). **Int. Scaling: 1 == 1 ° Type: I Volatile: Y**	-	-	-	°	C
3.14	**Unused**					
3.15	**ReactCur (reactive current)** Relative actual reactive motor current in percent of *M1NomCur (99.03)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
3.16	**Unused**					
3.17	**ArmAlphaSl (12-pulse slave armature α, firing angle)** Firing angle (α) of 12-pulse slave converter: – Valid in 12-pulse master only **Int. Scaling: 1 == 1 ° Type: I Volatile: Y**	-	-	-	°	E
3.18	**Unused**					
3.19	**Unused**					
3.20	**PLL In (phase locked loop input)** Actual measured mains voltage cycle (period) time. Is used as input of the PLL controller. The value should be: – 1/50 Hz = 20 ms = 20,000 – 1/60 Hz = 16.7 ms = 16,667 See also: – *DevLimPLL (97.13)* – *KpPLL (97.14)* – *TfPLL (97.15)* **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E
3.21	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
3.22	**CurCtrlIntegOut (integral part of current controller output)** I-part of the current controller's output in percent of *M1NomCur (99.03).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
3.23	**CurActPeak (relative actual armature peak current)** Relative actual armature peak current in percent of *M1NomCur (99.03)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
3.24	**FluxRefFldWeak (flux reference for field weakening)** Relative flux reference for speeds above the field weakening point (base speed) in percent of nominal flux. For proper scaling, setting of *CtrlModeSel (43.05)* = **PowerSupply1** divides the value of *FluxRefFldWeak (3.24)* by 2. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
3.25	**VoltRef1 (EMF voltage reference 1)** Selected relative EMF voltage reference in percent of *M1NomVolt (99.02)*: – *EMF RefSel (46.03)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	C
3.26	**VoltRef2 (EMF voltage reference 2)** Relative EMF voltage reference in percent of *M1NomVolt (99.02)* after ramp and limitation (input to EMF controller): – *VoltRefSlope (46.06)* – *VoltPosLim (46.07)* – *VoltNegLim (46.08)* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
3.27	**FluxRefEMF (flux reference after EMF controller)** Relative EMF flux reference in percent of nominal flux after EMF controller. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
3.28	**FluxRefSum (sum of flux reference)** *FluxRefSum (3.28)* = *FluxRefEMF (3.27)* + *FluxRefFldWeak (3.24)* in percent of nominal flux. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
3.29	**Unused**					
3.30	**FldCurRefM1 (motor 1 field current reference)** Relative motor 1 field current reference in percent of *M1NomFldCur (99.11).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E
3.31	**FldCurRefM2 (motor 2 field current reference)** Relative motor 2 field current reference in percent of *M2NomFldCur (49.05).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-	-	-	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 4	**Information**					
4.01	**FirmwareVer (firmware version)** Name of the loaded firmware version. The format is: **yyy** or **-yyy** with: **yyy** = consecutively numbered version and **-yyy** = single phase firmware for demo units. **Int. Scaling: - Type: C Volatile: Y**	-	-	-	-	C
4.02	**FirmwareType (firmware type)** Type of the loaded firmware version. The format is: 80 = Standard firmware 87 = Heating firmware **Int. Scaling: - Type: C Volatile: Y**					
4.03	**ApplicName (name of application program)** Name of the running application program: 0 = **NoMemCard** no Memory Card plugged in 1 = **Inactive** A Memory Card is plugged in, but the application program is inactive. Use *ParApplSave (16.06)* = **EableAppl** to activate the application program. 2 = **NoApplic** the Memory Card is empty (no application program available) 3 = <application name> name of the running application program **Int. Scaling: - Type: C Volatile: Y**	-	-	-	-	C
4.04	**ConvNomVolt (converter nominal AC voltage measurement circuit)** Adjustment of AC voltage measuring channels (SDCS-PIN-4 or SDCS-PIN-51). Read from *TypeCode (97.01)* or set with *S ConvScaleVolt (97.03)*: – Read from *TypeCode (97.01)* if *S ConvScaleVolt (97.03)* = 0 – Read from *S ConvScaleVolt (97.03)* if *S ConvScaleVolt (97.03)* ≠ 0 **Int. Scaling: 1 == 1 V Type: I Volatile: Y**	-	-	-	V	C
4.05	**ConvNomCur (converter nominal DC current measurement circuit)** Adjustment of DC current measuring channels (SDCS-PIN-4 or SDCS-PIN-51). Read from *TypeCode (97.01)* or set with *S ConvScaleCur (97.02)*: – Read from *TypeCode (97.01)* if *S ConvScaleCur (97.02)* = 0 – Read from *S ConvScaleCur (97.02)* if *S ConvScaleCur (97.02)* ≠ 0 **Int. Scaling: 1 == 1 A Type: I Volatile: Y**	-	-	-	A	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
4.06	**Mot1FexType (motor 1 type of field exciter)** Motor 1 field exciter type. Read from *M1UsedFexType (99.12)*: 0 = **NotUsed** no or third party field exciter connected 1 = **OnBoard** integrated 1-Q field exciter (for sizes D1 - D4 only), default 2 = **FEX-425-Int** internal 1-Q 25 A field exciter (for size D5 only) used for field currents from 0.3 A to **25 A** (terminals X100.1 and X100.3) 3 = **DCF803-0035** external 1-Q 35 A field exciter used for field currents from 0.3 A to **35 A** (terminals X100.1 and X100.3) 4 = **DCF803-0050** external 1-Q 50 A field exciter (DCF803-0050 or DCF503B-0050) 5 = **DCF804-0050** external 4-Q 50 A field exciter (DCF804-0050 or DCF504B-0050) 6 = **DCF803-0060** external 1-Q 60 A field exciter; not implemented yet 7 = **DCF804-0060** external 4-Q 60 A field exciter; not implemented yet 8 = **DCS800-S01** external 2-Q 3-phase field exciter 9 = **DCS800-S02** external 4-Q 3-phase field exciter 10 = **DCF803-0016** external 1-Q 16 A field exciter used for field currents from 0.3 A to **16 A** (terminals X100.1 and X100.3) 11 = reserved to 14 = reserved 15 = **ExFex AITAC** third party field exciter, acknowledge via AITAC 16 = **ExFex AI1** third party field exciter, acknowledge via AI1 17 = **ExFex AI2** third party field exciter, acknowledge via AI2 18 = **ExFex AI3** third party field exciter, acknowledge via AI3 19 = **ExFex AI4** third party field exciter, acknowledge via AI4 20 = **FEX-4-Term5A** internal 2-Q 25 A field exciter (FEX-425-Int), external 2-Q 16 A field exciter (DCF803-0016) or external 2-Q 35 A field exciter (DCF803-0035) used for field currents from 0.3 A to **5 A** (terminals X100.2 and X100.3) 21 = **VariFexType** see *DCS800 MultiFex motor control (3ADW000309)* 22 = **Exc-Appl-1** see *DCS800 Series wound motor control (3ADW000311)* **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C

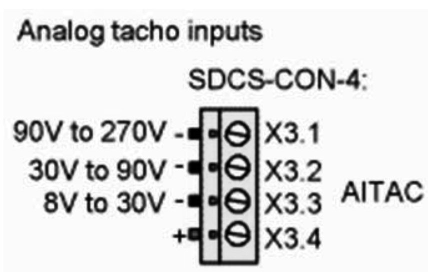
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
4.07	**Mot2FexType (motor 2 type of field exciter)** Motor 2 field exciter type. Read from *M2UsedFexType (49.07)*: 0 = **NotUsed** no or third party field exciter connected 1 = **OnBoard** integrated 1-Q field exciter (for sizes D1 - D4 only), default 2 = **FEX-425-Int** internal 1-Q 25 A field exciter (for size D5 only) used for field currents from 0.3 A to **25 A** (terminals X100.1 and X100.3) 3 = **DCF803-0035** external 1-Q 35 A field exciter used for field currents from 0.3 A to **35 A** (terminals X100.1 and X100.3) 4 = **DCF803-0050** external 1-Q 50 A field exciter (DCF803-0050 or DCF503B-0050) 5 = **DCF804-0050** external 4-Q 50 A field exciter (DCF804-0050 or DCF504B-0050) 6 = **DCF803-0060** external 1-Q 60 A field exciter; not implemented yet 7 = **DCF804-0060** external 4-Q 60 A field exciter; not implemented yet 8 = **DCS800-S01** external 2-Q 3-phase field exciter 9 = **DCS800-S02** external 4-Q 3-phase field exciter 10 = **DCF803-0016** external 1-Q 16 A field exciter used for field currents from 0.3 A to **16 A** (terminals X100.1 and X100.3) 11 = reserved to 14 = reserved 15 = **ExFex AITAC** third party field exciter, acknowledge via AITAC 16 = **ExFex AI1** third party field exciter, acknowledge via AI1 17 = **ExFex AI2** third party field exciter, acknowledge via AI2 18 = **ExFex AI3** third party field exciter, acknowledge via AI3 19 = **ExFex AI4** third party field exciter, acknowledge via AI4 20 = **FEX-4-Term5A** internal 2-Q 25 A field exciter (FEX-425-Int), external 2-Q 16 A field exciter (DCF803-0016) or external 2-Q 35 A field exciter (DCF803-0035) used for field currents from 0.3 A to **5 A** (terminals X100.2 and X100.3) 21 = reserved 22 = **Exc-Appl-1** see *DCS800 Series wound motor control (3ADW000311)* **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
4.08	**Mot1FexSwVer (motor 1 firmware version of field exciter)** Motor 1 field exciter firmware version. The format is: **yyy** with: **yyy** = consecutively numbered version. This signal is set during initialization of the drive. New values are shown after the next power-up. **Int. Scaling: - Type: C Volatile: Y**	-	-	-	-	C
4.09	**Mot2FexSwVer (motor 2 firmware version of field exciter)** Motor 2 field exciter firmware version. The format is: **yyy** with: **yyy** = consecutively numbered version. This signal is set during initialization of the drive. New values are shown after the next power-up. **Int. Scaling: - Type: C Volatile: Y**	-	-	-	-	E
4.10	**Unused**					
4.11	**Com8SwVersion (firmware version of SDCS-COM-8)** SDCS-COM-8 firmware version. The format is: **yyy** with: **yyy** = consecutively numbered version. This signal is set during initialization of the drive. New values are shown after the next power-up. **Int. Scaling: Type: C Volatile: Y**					E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
4.12	**ApplicVer (application version)** Version of the loaded application program. The format is: **yyy** with: **yyy** = consecutively numbered version. **Int. Scaling: - Type: C Volatile: Y**	-	-	-	-	C
4.13	**DriveLibVer (drive library version)** Version of the loaded function block library. The format is: **yyy** with: **yyy** = consecutively numbered version. **Int. Scaling: - Type: C Volatile: Y**	-	-	-	-	C
4.14	**ConvType (converter type)** Recognized converter type. Read from *TypeCode (97.01):* 0 = **None** when *TypeCode (97.01)* = **None** 1 = **D1** D1 converter 2 = **D2** D2 converter 3 = **D3** D3 converter 4 = **D4** D4 converter 5 = **D5** D5 converter 6 = **D6** D6 converter 7 = **D7** D7 converter 8 = **ManualSet** set by user, if *S ConvScaleCur (97.02)* and / or *S ConvScaleVolt (97.03)* have been changed for e.g. rebuild kits **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C
4.15	**QuadrantType (quadrant type of converter; 1 or 2 bridges)** Recognized converter quadrant type. Read from *TypeCode (97.01)* or set with *S BlockBrdg2 (97.07)*: – Read from *TypeCode (97.01)* if *S BlockBrdg2 (97.07)* = 0 – Read from *S BlockBrdg2 (97.07)* if *S BlockBrdg2 (97.07)* ≠ 0 0 = **BlockBridge2** bridge 2 blocked (== 2-Q operation) 1 = **RelBridge2** bridge 2 released (== 4-Q operation), default **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C
4.16	**ConvOvrCur (converter overcurrent [DC] level)** Converter current tripping level. This signal is set during initialization of the drive. New values are shown after the next power-up. **Int. Scaling: 1 == 1 A Type: I Volatile: Y**	-	-	-	A	C
4.17	**MaxBridgeTemp (maximum bridge temperature)** Maximum bridge temperature in degree centigrade. Read from *TypeCode (97.01)* or set with *S MaxBrdgTemp (97.04)*: – Read from *TypeCode (97.01)* if *S MaxBrdgTemp (97.04)* = 0 – Read from *S MaxBrdgTemp (97.04)* if *S MaxBrdgTemp (97.04)* ≠ 0 The drive trips with **F504 ConvOverTemp** [*FaultWord1 (9.01)* bit 3], when *MaxBridgeTemp (4.17)* is reached. **A104 ConvOverTemp** [*AlarmWord1 (9.06)* bit 3] is set, when the actual converter temperature is approximately 5°C below *MaxBridgeTemp (4.17)*. **Int. Scaling: 1 == 1 °C Type: I Volatile: Y**	-	-	-	°C	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
4.18	**DCSLinkStat1 (DCSLink status 1 of field exciter nodes)** Status of DCSLink for field exciter nodes 1 to 16: Bit Name Value Comment B0 **Node1** 1 DCSLink node1 active and OK 0 DCSLink node1 not active or faulty B1 **Node2** 1 DCSLink node2 active and OK 0 DCSLink node2 not active or faulty B2 **Node3** 1 DCSLink node3 active and OK 0 DCSLink node3 not active or faulty B3 **Node4** 1 DCSLink node4 active and OK 0 DCSLink node4 not active or faulty B4 **Node5** 1 DCSLink node5 active and OK 0 DCSLink node5 not active or faulty B5 **Node6** 1 DCSLink node6 active and OK 0 DCSLink node6 not active or faulty B6 **Node7** 1 DCSLink node7 active and OK 0 DCSLink node7 not active or faulty B7 **Node8** 1 DCSLink node8 active and OK 0 DCSLink node8 not active or faulty B8 **Node9** 1 DCSLink node9 active and OK 0 DCSLink node9 not active or faulty B9 **Node10** 1 DCSLink node10 active and OK 0 DCSLink node10 not active or faulty B10 **Node11** 1 DCSLink node11 active and OK 0 DCSLink node11 not active or faulty B11 **Node12** 1 DCSLink node12 active and OK 0 DCSLink node12 not active or faulty B12 **Node13** 1 DCSLink node13 active and OK 0 DCSLink node13 not active or faulty B13 **Node14** 1 DCSLink node14 active and OK 0 DCSLink node14 not active or faulty B14 **Node15** 1 DCSLink node15 active and OK 0 DCSLink node15 not active or faulty B15 **Node16** 1 DCSLink node16 active and OK 0 DCSLink node16 not active or faulty **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C

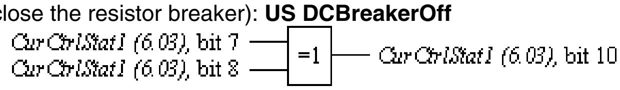
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
4.19	**DCSLinkStat2 (DCSLink status 2 of field exciter nodes)** Status of DCSLink for field exciter nodes 17 to 32: Bit Name Value Comment B0 **Node17** 1 DCSLink node17 active and OK 0 DCSLink node17 not active or faulty B1 **Node18** 1 DCSLink node18 active and OK 0 DCSLink node18 not active or faulty B2 **Node19** 1 DCSLink node19 active and OK 0 DCSLink node19 not active or faulty B3 **Node20** 1 DCSLink node20 active and OK 0 DCSLink node20 not active or faulty -- B4 **Node21** 1 DCSLink node21 active and OK 0 DCSLink node21 not active or faulty B5 **Node22** 1 DCSLink node22 active and OK 0 DCSLink node22 not active or faulty B6 **Node23** 1 DCSLink node23 active and OK 0 DCSLink node23 not active or faulty B7 **Node24** 1 DCSLink node24 active and OK 0 DCSLink node24 not active or faulty -- B8 **Node25** 1 DCSLink node25 active and OK 0 DCSLink node25 not active or faulty B9 **Node26** 1 DCSLink node26 active and OK 0 DCSLink node26 not active or faulty B10 **Node27** 1 DCSLink node27 active and OK 0 DCSLink node27 not active or faulty B11 **Node28** 1 DCSLink node28 active and OK 0 DCSLink node28 not active or faulty -- B12 **Node29** 1 DCSLink node29 active and OK 0 DCSLink node29 not active or faulty B13 **Node30** 1 DCSLink node30 active and OK 0 DCSLink node30 not active or faulty B14 **Node31** 1 DCSLink node31 active and OK 0 DCSLink node31 not active or faulty B15 **Node32** 1 DCSLink node32 active and OK 0 DCSLink node32 not active or faulty **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
4.20	**Ext IO Stat (external IO status)** Status of external I/O: Bit / Value / Comment B0 1 first RAIO-xx detected, see *AIO ExtModule (98.06)* 0 first RAIO-xx not existing or faulty B1 1 second RAIO-xx detected, see *AIO MotTempMeas (98.12)* 0 second RAIO-xx not existing or faulty B2 1 RRIA-xx detected 0 RRIA-xx not existing or faulty B3 1 RTAC-xx detected 0 RTAC-xx not existing or faulty ---------- B4 1 first RDIO-xx detected, see *DIO ExtModule1 (98.03)* 0 first RDIO-xx not existing or faulty B5 1 second RDIO-xx detected, see *DIO ExtModule2 (98.04)* 0 second RDIO-xx not existing or faulty B6 1 - 0 - B7 1 - 0 - ---------- B8 1 - 0 - B9 1 - 0 - B10 1 SDCS-DSL-4 detected, see *DCSLinkNodeID (94.01)* 0 SDCS-DSL-4 not existing or faulty B11 1 SDCS-IOB-2x detected, see *IO BoardConfig (98.15)* 0 SDCS-IOB-2x not existing or faulty ---------- B12 1 SDCS-IOB-3 detected, see *IO BoardConfig (98.15)* 0 SDCS-IOB-3 not existing or faulty B13 1 SDCS-COM-8 detected, see *CommModule (98.02)* and group 70 0 SDCS-COM-8 not existing or faulty B14 1 RMBA-xx (Modbus) detected, see *CommModule (98.02)* and *ModBusModule2 (98.08)* 0 RMBA-xx (Modbus) not existing or faulty B15 1 SDCS-MEM-8 (Memory Card) detected 0 SDCS-MEM-8 (Memory Card) not existing or faulty **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
4.21	**CPU Load (load of processor)** The calculating power of the processor is divided into two parts: – *CPU Load (4.21)* shows the load of the firmware and – *ApplLoad (4.22)* shows the load of the application. Neither should reach 100%. **Int. Scaling: 10 == 1 % Type: I Volatile: Y**	-	-	-	%	C
4.22	**ApplLoad (load of application)** The calculating power of the processor is divided into two parts: – *CPU Load (4.21)* shows the load of the firmware and – *ApplLoad (4.22)* shows the load of the application. Neither should reach 100%. **Int. Scaling: 10 == 1 % Type: I Volatile: Y**	-	-	-	%	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
4.23	**MotTorqNom (motor nominal torque)** Calculated nominal motor torque. **Note:** the value is calculated the following way: $MotTorqNom\,(4.23) = \frac{60}{2*\pi} * \frac{[M1NomVolt(99.02) - M1MotCur(99.03) * M1ArmR(43.10)] * M1NomCur\,(99.03)}{M1BaseSpeed\,(99.04)}$ Values above 65000 can not be displayed **Int. Scaling: 1 == 1 Nm Type: I Volatile: Y**	-	-	-	Nm	C
4.24	**ProgressSignal (progress signal for auto tunings)** Progress signal for auto tunings used for Startup Assistants. **Int. Scaling: 1 == 1 % Type: I Volatile: Y**	-	-	-	%	E
4.25	**TachoTerminal (tacho terminal to be used)** Depending on the analog tacho output voltage - e.g. 60 V at 1000 rpm - and the maximum speed of the drive system - which is the maximum of *SpeedScaleAct (2.29)*, *M1OvrSpeed (30.16)* and *M1BaseSpeed (99.04)* - different inputs connections at the SDCS-CON-4 have to be used:  *TachoTerminal (4.25)* shows which terminal has to be used depending on the setting of *M1TachoVolt1000 (50.13)* and the actual maximum speed of the drive system: 0 = **NotUsed** if *M1TachoVolt1000 (50.13)* = 0 V, no analog tacho used or not set jet 1 = **X3:3 8-30V** result if *M1TachoVolt1000 (50.13)* ≥ 1 V 2 = **X3:2 30-90V** result if *M1TachoVolt1000 (50.13)* ≥ 1 V 3 = **X3:1 90-120V** result if *M1TachoVolt1000 (50.13)* ≥ 1 V 4 = **Auto** result if *M1TachoVolt1000 (50.13)* = -1 V after the tacho gain was successfully measured by means of the speed feedback assistant **Note:** *TachoTerminal (4.25)* is also valid for motor 2 depending on setting of *ParChange (10.10)* and *MacroChangeMode (16.05)*. **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
4.26	**IactScaling (scaling of the fixed actual current output I-act)** Scaling of analog output for the actual output current in Ampere per 10 V output voltage. See terminals SDCS-CON-4 X4:9 and SDCS-IOB-3 X4:5. **Note:** The scaling can also be adjusted by means of R110 when using a SDCS-IOB-3. **Int. Scaling: 1 == 1 A Type: SI Volatile: Y**	-	-	-	A	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 5	**Analog I/O**					
5.01	**AITacho Val (analog input for tacho)** Measured actual voltage at analog tacho input. The integer scaling may differ, depending on the connected hardware and jumper setting. **Note:** A value of 11 V equals 1.25 * *M1OvrSpeed (30.16)* **Int. Scaling: 1000 == 1 V Type: SI Volatile: Y**	-	-	-	V	C
5.02	**Unused**					
5.03	**AI1 Val (analog input 1 value)** Measured actual voltage at analog input 1. The integer scaling may differ, depending on the connected hardware and jumper settings. **Int. Scaling: 1000 == 1 V Type: SI Volatile: Y**	-	-	-	V	C
5.04	**AI2 Val (analog input 2 value)** Measured actual voltage at analog input 2. The integer scaling may differ, depending on the connected hardware and jumper settings. **Int. Scaling: 1000 == 1 V Type: SI Volatile: Y**	-	-	-	V	C
5.05	**AI3 Val (analog input 3 value)** Measured actual voltage at analog input 3. The integer scaling may differ, depending on the connected hardware and jumper settings. **Int. Scaling: 1000 == 1 V Type: SI Volatile: Y**	-	-	-	V	E
5.06	**AI4 Val (analog input 4 value)** Measured actual voltage at analog input 4. The integer scaling may differ, depending on the connected hardware and jumper settings. **Int. Scaling: 1000 == 1 V Type: SI Volatile: Y**	-	-	-	V	E
5.07	**AI5 Val (analog input 5 value)** Measured actual voltage at analog input 5. The integer scaling may differ, depending on the connected hardware and DIP-switch settings. Available only with RAIO extension module see *AIO ExtModule (98.06).* **Int. Scaling: 1000 == 1 V Type: SI Volatile: Y**	-	-	-	V	E
5.08	**AI6 Val (analog input 6 value)** Measured actual voltage at analog input 6. The integer scaling may differ, depending on the connected hardware and DIP-switch settings. Available only with RAIO extension module see *AIO ExtModule (98.06).* **Int. Scaling: 1000 == 1 V Type: SI Volatile: Y**	-	-	-	V	E
5.09	**Unused**					
5.10	**Unused**					
5.11	**AO1 Val (analog output 1 value)** Measured actual voltage at analog output 1. **Int. Scaling: 1000 == 1 V Type: SI Volatile: Y**	-	-	-	V	C
5.12	**AO2 Val (analog output 2 value)** Measured actual voltage at analog output 2. **Int. Scaling: 1000 == 1 V Type: SI Volatile: Y**	-	-	-	V	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 6	**Drive logic signals**					
6.01	**SystemTime (converter system time)** Shows the time of the converter in minutes. The system time can be either set by means of *SetSystemTime (16.11)* or via the DCS800 Control Panel. **Int. Scaling: 1 == 1 min Type: I Volatile: Y**	-	-	-	min	C
6.02	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
6.03	**CurCtrlStat1 (1st current controller status)** 1st current controller status word: Bit Value Comment B0 1 command **FansOn** 0 command **FansOff**; See also trip levels in paragraph *Fault signals* of this manual B1 1 one mains phase missing 0 no action B2 1 - 0 - B3 1 motor heating function active 0 motor heating function not active --- B4 1 field direction reverse 0 field direction forward B5 1 command to switch excitation on: **FieldOn** 0 command to switch excitation off: **FieldOff** B6 1 dynamic braking active / started 0 dynamic braking not active B7 1 command to close main contactor: **MainContactorOn** 0 command to open main contactor: **MainContactorOff** --- B8 1 command to close contactor for dynamic braking resistor (armature current is zero): **DynamicBrakingOn** 0 command to open contactor for dynamic braking resistor: **DynamicBrakingOff** B9 1 drive is generating 0 drive is motoring B10 1 command to close the US style changeover DC-breaker (close the DC-breaker, open the resistor breaker): **US DCBreakerOn** 0 command to open the US style changeover DC-breaker (open the DC-breaker, close the resistor breaker): **US DCBreakerOff**  B11 1 firing pulses active (on) 0 firing pulses blocked --- B12 1 continuous current 0 discontinuous current B13 1 zero current detected 0 current not zero B14 1 command **Trip DC-breaker** (continuous signal) 0 no action B15 1 command **Trip DC-breaker** (1 s pulse) 0 no action **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
6.04	**CurCtrlStat2 (2nd current controller status)** 2nd current controller status word. The current controller will be blocked, *CurRefUsed (3.12)* is forced to zero and *ArmAlpha (3.13)* is forced to the value of *ArmAlphaMax (20.14)* if any of the bits is set (0 == OK): Bit / Value / Meaning B0 1 overcurrent, **F502 ArmOverCur** [*FaultWord1 (9.01)* bit 1] 0 no action B1 1 mains overvoltage (AC), **F513 MainsOvrVolt** [*FaultWord1 (9.01)* bit 12] 0 no action B2 1 mains undervoltage (AC), **F512 MainsLowVolt** [*FaultWord1 (9.01)* bit 11] 0 no action B3 1 waiting for reduction of EMF to match the mains voltage [see *RevVoltMargin (44.21)*] 0 no action ---------- B4 1 **F533 12PRevTime** [*FaultWord3 (9.03)* bit 0], **F534 12PCurDiff** [*FaultWord3 (9.03)* bit 1] or **F557 ReversalTime** [*FaultWord4 (9.04)* bit 8] 0 no action B5 1 *OperModeSel (43.01)* = **12P…..**: partner blocked) *OperModeSel (43.01)* = **FieldExciter**: Overvoltage protection active (freewheeling) 0 no action B6 1 motor 1 field exciter selftest faulty, **F529 M1FexNotOK** [*FaultWord2 (9.02)* bit 12] 0 motor 1 field exciter selftest OK B7 1 motor 1 field exciter not ready, **F537 M1FexRdyLost** [*FaultWord3 (9.03)* bit 4] 0 motor 1 field exciter ready ---------- B8 1 motor 2 field exciter selftest faulty, **F530 M2FexNotOK** [*FaultWord2 (9.02)* bit 13] 0 motor 2 field exciter selftest OK B9 1 motor 2 field exciter not ready, **F538 M2FexRdyLost** [*FaultWord3 (9.03)* bit 5] 0 motor 2 field exciter ready B10 1 waiting for zero current 0 no action B11 1 field reversal active, armature current controller is blocked 0 no action ---------- B12 1 - 0 - B13 1 current controller not released, because *DevLimPLL (97.13)* is reached 0 no action B14 1 mains not in synchronism (AC), **F514 MainsNotSync** [*FaultWord1 (9.01)* bit 13] 0 no action B15 1 Current controller not released. This bit is set in case of a relevant fault (Fxxx) or an alarm (Axxx) of alarm level 3. 0 no action **Note:** A set bit does not necessarily lead to a fault message it depends also on the status of the drive. **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
6.05	**SelBridge (selected bridge)** Selected (current-conducting) bridge: 0 = **NoBridge** no bridge selected 1 = **Bridge1** bridge 1 selected (motoring bridge) 2 = **Bridge2** bridge 2 selected (generating bridge) **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
6.06	**Unused**					
6.07	**Unused**					
6.08	**Unused**					
6.09	**CtrlStatMas (12-pulse master control status)** 12-pulse master control status: Bit Value Comment B0 1 command **On** to 12-pulse slave 0 no action B1 1 command **Off2N** (Emergency Off / Coast Stop) to 12-pulse slave (low active) 0 no action B2 1 motor heating function active 0 motor heating function not active B3 1 command **Run** to 12-pulse slave 0 no action ---------- B4 1 command field exciter **On** 0 command field exciter **Off** B5 1 dynamic braking 0 no action B6 1 12-pulse serial operation, see *OperModeSel (43.01)* 0 12-pulse parallel operation, see *OperModeSel (43.01)* B7 1 command **Reset** to 12-pulse slave 0 no action ---------- B8 1 - 0 - B9 1 - 0 - B10 1 waiting for reduction of EMF to match the mains voltage [see *RevVoltMargin (44.21)*] 0 no action B11 1 autotuning armature current controller active 0 no action ---------- B12 1 zero current detected + *RevDly (43.14)* is elapsed 0 no action B13 1 command to change direction of current (bridge change over active) 0 no action B14 1 *CurCtrlStat2 (6.04)* > 0 (current controller is blocked) 0 no action B15 1 *CurRefUsed (3,12)* negative 0 *CurRefUsed (3.12)* positive – The control bits B3 to B6 (**Reset**, **On**, **Run** and **Off2N**) are only valid in the 12-pulse slave, if in the 12-pulse slave *CommandSel (10.01)* = **12P Link** – Valid in 12-pulse master and slave **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
6.10	**CtrlStatSla (12-pulse slave control status)** 12-pulse slave control status: Bit Value Comment B0 1 - 0 - B1 1 - 0 - B2 1 - 0 - B3 1 slave is **Tripped** 0 no action ---------- B4 1 - 0 - B5 1 - 0 - B6 1 12-pulse serial operation, see *OperModeSel (43.01)* 0 12-pulse parallel operation, see *OperModeSel (43.01)* B7 1 - 0 - ---------- B8 1 - 0 - B9 1 - 0 - B10 1 - 0 - B11 1 - 0 - ---------- B12 1 - 0 - B13 1 bridge change over active 0 no action B14 1 *CurCtrlStat2 (6.04)* > 0 (current controller is blocked) 0 no action B15 1 *CurRefUsed (3,12)* negative 0 *CurRefUsed (3.12)* positive − Valid in 12-pulse master and slave **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E
6.11	**Unused**					
6.12	**Mot1FexStatus (motor 1 field exciter status)** Motor 1 field exciter status: 0 = **NotUsed** no field exciter connected 1 = **OK** field exciter and communication OK 2 = **ComFault** **F516 M1FexCom** [*FaultWord1 (9.01)* bit 15], communication faulty 3 = **FexFaulty** **F529 M1FexNotOK** [*FaultWord2 (9.02)* bit 12], field exciter selftest faulty 4 = **FexNotReady** **F537 M1FexRdyLost** [*FaultWord3 (9.03)* bit 4], field exciter not ready 5 = **FexUnderCur** **F541 M1FexLowCur** [*FaultWord3 (9.03)* bit 8], field exciter undercurrent 6 = **FexOverCur** **F515 M1FexOverCur** [*FaultWord1 (9.01)* bit 14], field exciter overcurrent 7 = **WrongSetting** check setting of *M1UsedFexType (99.12)* and *M2UsedFexType (49.07)* **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
6.13	**Mot2FexStatus (motor 2 field exciter status)** Motor 1 field exciter status: 0 = **NotUsed** no field exciter connected 1 = **OK** field exciter and communication OK 2 = **ComFault** **F519 M2FexCom** [*FaultWord2 (9.02)* bit 2], communication faulty 3 = **FexFaulty** **F530 M2FexNotOK** [*FaultWord2 (9.02)* bit 13], field exciter selftest faulty 4 = **FexNotReady** **F538 M2FexRdyLost** [*FaultWord3 (9.03)* bit 5], field exciter not ready 5 = **FexUnderCur** **F542 M2FexLowCur** [*FaultWord3 (9.03)* bit 9], field exciter undercurrent 6 = **FexOverCur** **F518 M2FexOverCur** [*FaultWord2 (9.02)* bit 1], field exciter overcurrent 7 = **WrongSetting** check setting of *M1UsedFexType (99.12)* and *M2UsedFexType (49.07)* **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 7	**Control words**					
	All signals in this group - except *UsedMCW (7.04)* - can be written to my means of DWL, DCS800 Control Panel, Adaptive Program, application program or overriding control.					
7.01	**MainCtrlWord (main control word, MCW)** The main control word contains all drive depending commands and can be written to by Adaptive Program, application program or overriding control: Bit / Name / Value / Comment B0 **On (Off1N)** 1: Command to **RdyRun** state. With *MainContCtrlMode (21.16)* = **On**: Contactors are closed, field exciter and fans are started. With *MainContCtrlMode (21.16)* = **On&Run**: **RdyRun** flag in *MainStatWord (8.01)* is forced to 1 0: Command to **Off** state. Stopping via *Off1Mode (21.02)*. B1 **Off2N** 1: No **Off2** (Emergency Off / Coast Stop) 0: Command to **OnInhibit** state. Stop by coasting. The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. **Off2N** has priority over **OffN3** and **On**. B2 **Off3N** 1: No **Off3** (E-stop) 0: Command to **OnInhibit** state. Stopping via *E StopMode (21.04)*. **Off3N** has priority over **On**. B3 **Run** 1: Command to **RdyRef** state. The firing pulses are released and the drive is running with the selected speed reference. 0: Command to **RdyRun** state. Stop via *StopMode (21.03)*. -- B4 **RampOutZero** 1: no action 0: speed ramp output is forced to zero B5 **RampHold** 1: no action 0: freeze (hold) speed ramp B6 **RampInZero** 1: no action 0: speed ramp input is forced to zero	-	-	-	-	C

Index	Signal / Parameter name				min.	max.	def.	unit	E/C
	B7	**Reset**	1 0	acknowledge fault indications with the positive edge no action	-	-	-	-	C
	B8	**Inching1**	1	constant speed defined by *FixedSpeed1 (23.02)*, active only with *CommandSel (10.01)* = **MainCtrlWord** and **RampOutZero** = **RampHold** = **RampInZero** = **Run** = 0; **Inching2** overrides **Inching1** alternatively *Jog1 (10.17)* can be used					
			0	no action					
	B9	**Inching2**	1	constant speed defined by *FixedSpeed2 (23.03)*, active only with *CommandSel (10.01)* = **MainCtrlWord** and **RampOutZero** = **RampHold** = **RampInZero** = **Run** = 0; **Inching2** overrides **Inching1** alternatively *Jog2 (10.18)* can be used					
			0	no action					
	B10	**RemoteCmd**	1	overriding control enabled (overriding control has to set this value to 1)					
			0	The last *UsedMCW (7.04)* and the last references [*SpeedRef (23.01)*, *AuxSpeedRef (23.13)*, *TorqRefA (25.01)* and *TorqRefB (25.04)*] are retained. On control place change - see *CommandSel (10.01*) - the drive is stopped. The aux. control bits (B11 to B15) are not affected.					
	B11	aux. control	x	used by Adaptive Program, application program or overriding control to control various functions selected by parameters					
	B12	aux. control	x	used by Adaptive Program, application program or overriding control to control various functions selected by parameters					
	B13	aux. control	x	used by Adaptive Program, application program or overriding control to control various functions selected by parameters					
	B14	aux. control	x	used by Adaptive Program, application program or overriding control to control various functions selected by parameters					
	B15	aux. control	x	used by Adaptive Program, application program or overriding control to control various functions selected by parameters					
	Int. Scaling: 1 == 1	**Type:**	**I**	**Volatile: Y**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
7.02	**AuxCtrlWord (auxiliary control word 1, ACW1)** The auxiliary control word 1 can be written to by Adaptive Program, application program or overriding control:	-	-	-	-	C

Bit	Name	Value	Comment
B0	**RestartDataLog**	1	restart data logger
		0	no action
B1	**TrigDataLog**	1	trigger data logger (see note)
		0	no action
B2	**RampBypass**	1	bypass speed ramp (speed ramp output is forced to value of speed ramp input)
		0	no action
B3	**BalRampOut**	1	speed ramp output is forced to *BalRampRef (22.08)*
		0	no action
B4	**LimSpeedRef4**	1	*SpeedRef4 (2.18)* is not limited
		0	*SpeedRef4 (2.18)* is limited by *M1SpeedMax (20.02)* / *M1SpeedMin (20.01)* respectively by *M2SpeedMax (49.19)* / *M2SpeedMin (49.20)*
B5	**DynBrakingOn**	1	force dynamic braking independent from *Off1Mode (21.02)*, *StopMode (21.03)* or *E StopMode (21.04)*
		0	no action
B6	**HoldSpeedCtrl**	1	freeze (hold) the I-part of the speed controller
		0	no action
B7	**WindowCtrl**	1	release window control
		0	block window control
B8	**BalSpeedCtrl**	1	speed controller output is forced to *BalRef (24.11)*
		0	no action
B9	**SyncCommand**	1	positioning: synchronizing command from overriding control for pulse encoder 1 or pulse encoder 2 or both pulse encoders depending if *SyncCommand (10.04)* and / or *SyncCommand2 (10.05)* is set to **SyncCommand**
		0	no action
B10	**SyncDisable**	1	positioning: block synchronizing command
		0	no action
B11	**ResetSyncRdy**	1	positioning: reset **SyncRdy** [*AuxStatWord (8.02*) bit 5]
		0	no action
B12	aux. control	x	used by, Adaptive Program, application program or overriding control to control various functions selected by parameters
B13	aux. control	x	used by, Adaptive Program, application program or overriding control to control various functions selected by parameters
B14	aux. control	x	used by, Adaptive Program, application program or overriding control to control various functions selected by parameters
B15	aux. control	x	used by, Adaptive Program, application program or overriding control to control various functions selected by parameters

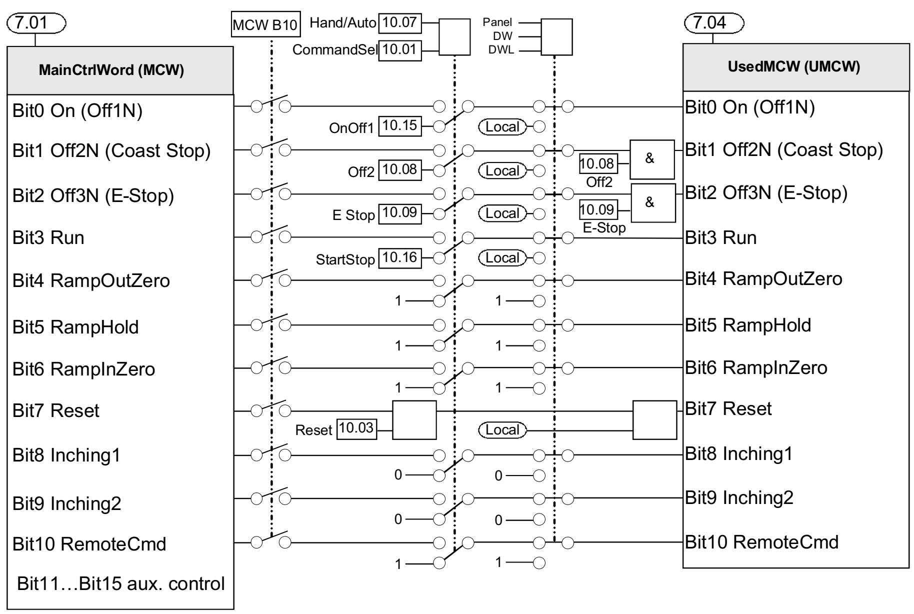
Note:
The data logger contains six channels with 1024 samples each.

Int. Scaling: 1 == 1 Type: I Volatile: Y

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
7.03	**AuxCtrlWord2 (auxiliary control word 2, ACW2)** The auxiliary control word 2 can be written to by Adaptive Program, application program or overriding control:	-	-	-	-	C

Bit	Name	Value	Comment
B0	reserved	1	
		0	
B1	reserved	1	
		0	
B2	reserved	1	
		0	
B3	reserved	1	
		0	
B4	**DisableBridge1**	1	bridge 1 blocked
		0	bridge 1 released
B5	**DisableBridge2**	1	bridge 2 blocked
		0	bridge 2 released
B6	**SupprArmCurDev**	1	**A114 ArmCurDev** [*AlarmWord1 (9.06)* bit 12] blocked, usually used for non motoric applications
		0	**A114 ArmCurDev** [*AlarmWord1 (9.06)* bit 12] released
B7	**ForceAlphaMax**	1	force single firing pulses and set firing angle (**α**) to *ArmAlphaMax (20.14)*
		0	normal firing pulses released
B8	**DriveDirection**	1	drive direction reverse (see note1), changes the signs of *MotSpeed (1.04)* and *CurRef (3.11)*
		0	drive direction forward (see note1)
B9	reserved	1	
		0	
B10	**DirectSpeedRef**	1	speed ramp output is overwritten and forced to *DirectSpeedRef (23.15)*
		0	speed ramp is active
B11	**TorqProvOK**	1	Selected motor torque proving is OK. This bit to be set by Adaptive Program, application program or overriding control [see also *M1TorqProvTime (42.10)*].
		0	Selected motor torque proving is inactive. This bit is to be set by Adaptive Program, application program or overriding control.
B12	**ForceBrake**	1	selected motor, the brake remains closed (applied) (see note2)
		0	selected motor, the brake is controlled by the internal brake logic in group 42 (Brake control)
B13	**ResetTorqMem**	1	reset torque memory (valid only if *M1StrtTorqRefSel (42.07)* = **Memory**)
		0	
B14	reserved	1	
		0	
B15	**ResetPIDCtrl**	1	reset and hold PID-controller
		0	release PID controller

Note1:
Changes of **DriveDirection** become active only in drive state **RdyRun**. Changing the speed direction of a running drive (**RdyRef** state) by means of **DriveDirection** is not possible.

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	Note2: If **ForceBrake** is set the brake remains closed (applied). If the **Run** [*MainCtrlWord (7.01)* bit 3] command is given to a drive in state **RdyOn** or **RdyRef** [*MainStatWord (8.01)* bit 0 and 1], the brake logic will be started up to the point of the brake open command. A drive in state **Running** [*MainStatWord (8.01)* bit 2] will be stopped by ramp, the brake will be closed (applied), but the drive will remain in state **Running**. **Int. Scaling: 1== 1 Type: I Volatile: Y**					
7.04	**UsedMCW (used main control word, UMCW)** Internal used (selected) main control word is read only and contains all drive depending commands. The selection is depending on the drives local/remote control setting, *CommandSel (10.01)* and *HandAuto (10.07)*. The bit functionality of bit 0 to bit 10 is the same as the in the *MainCtrlWord (7.01)*. Not all functions are controllable from local control or local I/O mode. B0 see *MainCtrlWord (7.01)* to B10 see *MainCtrlWord (7.01)* B11 reserved to B15 reserved  **Attention:** The *UsedMCW (7.04)* is write protected, thus it is not possible to write on the used main control word by means of Master-follower, Adaptive Program, application program or overriding control. **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

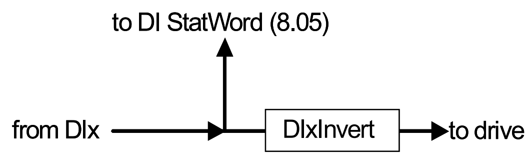
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
7.05	**DO CtrlWord (digital output control word, DOCW)** The DO control word 1 can be written to by Adaptive Program, application program or overriding control. To connect bits of the *DO CtrlWord (7.05)* with **DO1** to **DO8** use the parameters in group 14 (Digital outputs). **DO9** to **DO12** are directly sent to the extension I/O. Thus they are only available for Adaptive Program, application program or overriding control. Bit — Name — Comment B0 — **DO1** — this bit has to be send to the digital output via the parameters of group 14 (Digital outputs) B1 — **DO2** — this bit has to be send to the digital output via the parameters of group 14 (Digital outputs) B2 — **DO3** — this bit has to be send to the digital output via the parameters of group 14 (Digital outputs) B3 — **DO4** — this bit has to be send to the digital output via the parameters of group 14 (Digital outputs) -------------------- B4 — **DO5** — this bit has to be send to the digital output via the parameters of group 14 (Digital outputs) B5 — **DO6** — this bit has to be send to the digital output via the parameters of group 14 (Digital outputs) B6 — **DO7** — this bit has to be send to the digital output via the parameters of group 14 (Digital outputs) B7 — **DO8** — this bit has to be send to the digital output via the parameters of group 14 (Digital outputs) -------------------- B8 — **DO9** — this bit is written directly to DO1 of the extension IO defined by *DIO ExtModule1 (98.03)* B9 — **DO10** — this bit is written directly to DO2 of the extension IO defined by *DIO ExtModule1 (98.03)* B10 — **DO11** — this bit is written directly to DO1 of the extension IO defined by *DIO ExtModule2 (98.04)* B11 — **DO12** — this bit is written directly to DO2 of the extension IO defined by *DIO ExtModule2 (98.04)* -------------------- B12 to B15 — reserved **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C
7.06	**RFE CtrlWord (control word resonance frequency eliminator, RFECW)** The **R**esonance **F**requency **E**liminator control word can be written to by Adaptive Program, application program or overriding control: Bit — Name — Value — Comment B0 — **FilterRelease** — 1 — release the RFE filter with a static 1 0 — block the RFE filter with a static 0 B1 — **BalFilter** — 1 — Balance the RFE filter after a parameter change. Use a pulse of ≥ 10 ms (_⎍_). 0 — no action B2 to B15 — reserved **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 8	**Status / limit words**					
8.01	**MainStatWord (main status word, MSW)** Main status word: Bit / Name / Value / Comment B0 **RdyOn** 1 ready to switch on 0 not ready to switch on B1 **RdyRun** 1 ready to generate torque 0 not ready to generate torque B2 **RdyRef** 1 operation released **(Running)** 0 operation blocked B3 **Tripped** 1 fault indication 0 no fault --- B4 **Off2NStatus** 1 **Off2** not active 0 **Off2** (**OnInhibit** state) active B5 **Off3NStatus** 1 **Off3** not active 0 **Off3** (**OnInhibit** state) active B6 **OnInhibited** 1 **OnInhibited** state is active after a: – fault – Emergency Off / Coast Stop **(Off2)** – E-stop **(Off3)** – **OnInhibited** via digital input *Off2 (10.08)* or *E Stop (10.09)* 0 **OnInhibit** state not active B7 **Alarm** 1 alarm indication 0 no alarm --- B8 **AtSetpoint** 1 setpoint - *SpeedRef4 (2.18)* - and actual value - *MotSpeed (1.04)* - in the tolerance zone 0 setpoint - *SpeedRef4 (2.18)* - and actual value - *MotSpeed (1.04)* - out of the tolerance zone B9 **Remote** 1 remote control 0 local control B10 **AboveLimit** 1 speed greater than defined in *SpeedLev (50.10)* 0 speed lower or equal than defined *SpeedLev (50.10)* B11 reserved --- B12 reserved to B15 reserved **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
8.02	**AuxStatWord (auxiliary status word, ASW)** Auxiliary status word: Bit Name Value Comment B0 **DataLogReady** 1 contents of data logger is readable 0 contents of data logger is not readable B1 **OutOfWindow** 1 actual speed is out of window defined by *WinWidthPos (23.08)* and *WinWidthNeg (23.09)* 0 actual speed is inside the defined window B2 **E-StopCoast** 1 E-stop function has failed, see *E StopDecMin (21.05), E StopDecMax (21.06)* and *DecMonDly (21.07)* 0 no action B3 **User1** 1 macro **User1** active, see *ApplMacro (99.08)* 0 macro **User1** not active ---------- B4 **User2** 1 macro **User2** active, see *ApplMacro (99.08)* 0 macro **User2** not active B5 **SyncRdy** 1 positioning: synchronization is done either for pulse encoder 1 or pulse encoder 2 or both pulse encoders depending on the setting of *SyncCommand (10.04)* and *SyncCommand2 (10.05)*, enabled only if *PosSyncMode (50.15)* = **Single** 0 positioning: synchronizing not done B6 **Fex1Ack** 1 motor 1 field exciter acknowledged 0 no action B7 **Fex2Ack** 1 motor 2 field exciter acknowledged 0 no action ---------- B8 **BrakeCmd** 1 selected motor, command to open (lift) the brake is given, see group 42 (Brake control) 0 selected motor, command to close (apply) the brake is given B9 **Limiting** 1 drive is in a limit, see *LimWord (8.03)* 0 drive is not in a limit, B10 **TorqCtrl** 1 drive is torque controlled 0 no action B11 **ZeroSpeed** 1 actual motor speed is in the zero speed limit defined by *M1ZeroSpeedLim (20.03)* or *M2ZeroSpeedLim (49.04)* 0 actual motor speed is out of the zero speed limit ---------- B12 **EMFSpeed** 1 *M1SpeedFbSel (50.03)* = **EMF** 0 no action B13 **FaultOrAlarm** 1 fault or alarm indication 0 no fault or alarm indication B14 **DriveDirectionNeg** 1 negative drive direction active - controlled by bit 8 of *AuxCtrlWord2 (7.03)* 0 positive drive direction active - controlled by bit 8 of *AuxCtrlWord2 (7.03)* B15 **AutoReclosing** 1 auto reclosing logic is active 0 no action **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
8.03	**LimWord (limit word, LW)** Limit word: Bit active limit B0 *TorqMax (20.05)* or *TorqMaxAll (2.19)* B1 *TorqMin (20.06)* or *TorqMinAll (2.20)* B2 *TorqMaxSPC (20.07)* or *TorqMaxAll (2.19)* B3 *TorqMinSPC (20.08)* or *TorqMinAll (2.20)* -- B4 *TorqMaxTref (20.09)* B5 *TorqMinTref (20.10)* B6 *M1SpeedMax (20.02)* or *M2SpeedMax (49.20)* B7 *M1SpeedMin (20.01)* or *M2SpeedMin (49.19)* -- B8 *M1CurLimBrdg1 (20.12)* or *M2CurLimBrdg1 (49.12)* B9 *M1CurLimBrdg2 (20.13)* or *M2CurLimBrdg2 (49.13)* B10 reserved B11 reserved -- B12 reserved to B15 reserved **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E
8.04	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
8.05	**DI StatWord (digital inputs status word, DISW)** Digital input word, shows the value of the digital inputs before inversion [DI1Invert (10.25), …, DI11Invert (10.35)]:	-	-	-	-	C

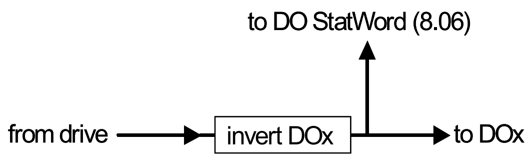


Bit	Name	Comment / default setting
B0	**DI1**	*ConvFanAck (10.20)*, actual setting depends on macro
B1	**DI2**	*MotFanAck (10.06)*, actual setting depends on macro
B2	**DI3**	*MainContAck (10.21)*, actual setting depends on macro
B3	**DI4**	*Off2 (10.08)*, actual setting depends on macro
B4	**DI5**	*E Stop (10.09)*, actual setting depends on macro
B5	**DI6**	*Reset (10.03)*, actual setting depends on macro
B6	**DI7**	*OnOff (10.15)*, actual setting depends on macro
B7	**DI8**	*StartStop (10.16)*, actual setting depends on macro
B8	**DI9**	DI1 of the extension IO defined by *DIO ExtModule1 (98.03)*
B9	**DI10**	DI2 of the extension IO defined by *DIO ExtModule1 (98.03)*
B10	**DI11**	DI3 of the extension IO defined by *DIO ExtModule1 (98.03)*
B11	**DI12**	DI1 of the extension IO defined by *DIO ExtModule2 (98.04).* Only available for Adaptive Program, application program or overriding control.
B12	**DI13**	DI2 of the extension IO defined by *DIO ExtModule2 (98.04).* Only available for Adaptive Program, application program or overriding control.
B13	**DI14**	DI3 of the extension IO defined by *DIO ExtModule2 (98.04).* Only available for Adaptive Program, application program or overriding control.
B14	reserved	
B15	reserved	

Int. Scaling: 1 == 1 **Type:** **I** **Volatile: Y**

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
8.06	**DO StatWord (digital outputs status word, DOSW)**	-	-	-	-	C

Digital output word, shows the value of the digital outputs after inversion:



Bit	Name	Comment / default setting
B0	**DO1**	*DO1Index (14.01)* = 603 and *DO1BitNo (14.02)* = 15, **FansOn**, actual setting depends on macro
B1	**DO2**	*DO2Index (14.03)* = 603 and *DO2BitNo (14.04)* = 5, **FieldOn**, actual setting depends on macro
B2	**DO3**	*DO3Index (14.05)* = 603 and *DO3BitNo (14.06)* = 7, **MainContactorOn**, actual setting depends on macro
B3	**DO4**	*DO4Index (14.07)* = 0 and *DO4BitNo (14.08)* = 0, Not connected, actual setting depends on macro
B4	**DO5**	*DO5Index (14.09)* = 0 and *DO5BitNo (14.10)* = 0, Not connected, actual setting depends on macro
B5	**DO6**	*DO6Index (14.11)* = 0 and *DO6BitNo (14.12)* = 0, Not connected, actual setting depends on macro
B6	**DO7**	*DO7Index (14.13)* = 0 and *DO7BitNo (14.14)* = 0, Not connected, actual setting depends on macro
B7	**DO8**	*DO8Index (14.15)* = 603 and *DO8BitNo (14.16)* = 7, **MainContactorOn**, actual setting depends on macro
B8	**DO9**	DO1 of the extension IO defined by *DIO ExtModule1 (98.03),* written to by *DO CtrlWord (7.05)* bit 8
B9	**DO10**	DO2 of the extension IO defined by *DIO ExtModule1 (98.03),* written to by *DO CtrlWord (7.05)* bit 9
B10	**DO11**	DO1 of the extension IO defined by *DIO ExtModule2 (98.04),* written to by *DO CtrlWord (7.05)* bit 10
B11	**DO12**	DO2 of the extension IO defined by *DIO ExtModule2 (98.04),* written to by *DO CtrlWord (7.05)* bit 11
B12 to B15	reserved	

Int. Scaling: 1 == 1 Type: I Volatile: Y

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
8.07	**Unused**					
8.08	**DriveStat (drive status)**	-	-	-	-	C

Drive status:

0 = **OnInhibited**	drive is in **OnInhibit** state
1 = **ChangeToOff**	drive is changing to **Off**
2 = **Off**	drive is **Off**
3 = **RdyOn**	drive is ready on
4 = **RdyRun**	drive is ready run
5 = **Running**	drive is **Running**
6 = **Stopping**	drive is **Stopping**
7 = **Off3**	drive is in **Off3** state (E-stop)
8 = **Off2**	drive is in **Off2** state (Emergency Off or Coast Stop)
9 = **Tripped**	drive is **Tripped**

Int. Scaling: 1 == 1 Type: C Volatile: Y

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
8.09	**MotSel (selected motor)** Select motor and field exciter: 0 = **Motor1** motor 1 and field exciter 1 are selected 1 = **Motor2** motor 2 and field exciter 2 are selected See *ParChange (10.10)* **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	E
8.10	**MacroSel (selected macro)** Currently selected macro: 0 = **None** default 1 = **Factory** factory (default) parameter set 2 = **User1** **User1** parameter set 3 = **User2** **User2** parameter set 4 = **Standard** standard parameter set 5 = **Man/Const** manual / constant speed 6 = **Hand/Auto** hand (manual) / automatic 7 = **Hand/MotPot** hand (manual) / motor potentiometer 8 = reserved reserved 9 = **MotPot** motor potentiometer 10 = **TorqCtrl** torque control 11 = **TorqLimit** torque limit 12 = **DemoStandard** demo standard 13 = **2WreDCcontUS** 2 wire with US style DC-breaker 14 = **3WreDCcontUS** 3 wire with US style DC-breaker 15 = **3WreStandard** 3 wire standard See ApplMacro *(99.08)* **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C
8.11	**RFE StatWord (status word resonance frequency eliminator)** **R**esonance **F**requency **E**liminator control word Bit Name Value Comment B0 **FiltParCalcAct** 1 internal parameters are being calculated, filter algorithm is skipped 0 no action B1 **ParUdpReq** 1 parameter update request after parameter change 0 no action B2 **FiltReleased** 1 RFE filter is released 0 RFE filter is blocked B3 **ParChange** 1 parameter have changed 0 no action -------------------------------- B4 reserved to B15 reserved **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 9	**Fault / alarm words**					
9.01	**FaultWord1 (fault word 1)** Fault word 1: Bit / Fault text / Fault code and trip level / Comment B0 **AuxUnderVolt** F501 1 auxiliary undervoltage B1 **ArmOverCur** F502 3 armature overcurrent, *ArmOvrCurLev (30.09)* B2 **ArmOverVolt** F503 3 armature overvoltage, *ArmOvrVoltLev (30.08)* B3 **ConvOverTemp** F504 2 converter overtemperature, *ConvTempDly (97.05)*, shutdown temperature see *MaxBridgeTemp (4.17)* --- B4 **ResCurDetect** F505 1 residual current detection, *ResCurDetectSel (30.05)*, *ResCurDetectLim (30.06)*, *ResCurDetectDel (30.07)* B5 **M1OverTemp** F506 2 motor 1 measured overtemperature, *M1FaultLimTemp (31.07)* or *M1KlixonSel (31.08)* B6 **M1OverLoad** F507 2 motor 1 calculated overload (thermal model), *M1FaultLimLoad (31.04)* B7 **I/OBoardLoss** F508 1 I/O board not found or faulty, *DIO ExtModule1 (98.03)*, *DIO ExtModule2 (98.04)*, *AIO ExtModule (98.06)*, *AIO MotTempMeas (98.12), IO BoardConfig (98.15)* --- B8 **M2OverTemp** F509 2 motor 2 measured overtemperature, *M2FaultLimTemp (49.37)* or *M2KixonSel (49.38)* B9 **M2OverLoad** F510 2 motor 2 calculated overload (thermal model), *M2FaultLimLoad (49.34)* B10 **ConvFanCur** F511 4 converter fan current, *ConvTempDly (97.05)* B11 **MainsLowVolt** F512 3 mains low (under-) voltage, *PwrLossTrip (30.21), UNetMin1 (30.22)*, *UNetMin2 (30.23)* --- B12 **MainsOvrVolt** F513 1 mains overvoltage, actual mains voltage is > 1.3 * *NomMainsVolt (99.10)* for longer than 10 s B13 **MainsNotSync** F514 3 mains not in synchronism B14 **M1FexOverCur** F515 1 motor 1 field exciter overcurrent, *M1FldOvrCurLev (30.13)* B15 **M1FexCom** F516 1 motor 1 field exciter communication loss, *FexTimeOut (94.07), DCSLinkNodeID (94.01)*, *M1FexNode (94.08)* **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.02	**FaultWord2 (fault word 2)** Fault word 2: Bit / Fault text / Fault code and trip level / Comment B0 **ArmCurRipple** F517 3 armature current ripple, *CurRippleMode (30.18), CurRippleLim (30.19)* B1 **M2FexOverCur** F518 1 motor 2 field exciter overcurrent, *M2FldOvrCurLev (49.09)* B2 **M2FexCom** F519 1 motor 2 field exciter communication loss *FexTimeOut (94.07), DCSLinkNodeID (94.01), M2FexNode (94.09)* B3 reserved F520 - no action ---------- B4 **FieldAck** F521 1 selected motor: field acknowledge, check fault message of or at field exciter B5 **SpeedFb** F522 3 selected motor: speed feedback, *SpeedFbFltSel (30.17), SpeedFbFltMode (30.36), M1SpeedFbSel (50.03)* B6 **ExtFanAck** F523 4 external fan acknowledge missing *MotFanAck (10.06)* B7 **MainContAck** F524 3 main contactor acknowledge missing, *MainContAck (10.21)* ---------- B8 **TypeCode** F525 1 type code mismatch, *TypeCode (97.01)* B9 **ExternalDI** F526 1 external fault via binary input, *ExtFaultSel (30.31)* B10 **ConvFanAck** F527 4 converter fan acknowledge missing, *ConvFanAck (10.20)* B11 **FieldBusCom** F528 5 fieldbus communication loss, *ComLossCtrl (30.28), FB TimeOut (30.35), CommModule (98.02)* ---------- B12 **M1FexNotOK** F529 1 motor 1 field exciter not okay B13 **M2FexNotOK** F530 1 motor 2 field exciter not okay B14 **MotorStalled** F531 3 selected motor: motor stalled, *StallTime (30.01), StallSpeed (30.02), StallTorq (30.03)* B15 **MotOverSpeed** F532 3 selected motor: motor overspeed, *M1OvrSpeed (30.16)* **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.03	**FaultWord3 (fault word 3)** Fault word 3: Bit / Fault text / Fault code and trip level / Comment B0 **12PRevTime** F533 3 12-pulse reversal timeout, *12P RevTimeOut (47.05)* B1 **12PCurDiff** F534 3 12-pulse current difference, *DiffCurLim (47.02), DiffCurDly (47.03)* B2 **12PulseCom** F535 3 12-pulse communication loss, *12P TimeOut (94.03), DCSLinkNodeID (94.01), 12P SlaNode (94.04)* B3 **12PSlaveFail** F536 4 12-pulse slave failure, this fault message trips the 12-pulse master and appears only in the 12-pulse master --- B4 **M1FexRdyLost** F537 1 motor 1 field exciter lost ready-for-operation message while working B5 **M2FexRdyLost** F538 1 motor 2 field exciter lost ready-for-operation message while working B6 **FastCurRise** F539 1 fast current rise, *ArmCurRiseMax (30.10)* B7 **COM8Faulty** F540 1 SDCS-COM-8 faulty --- B8 **M1FexLowCur** F541 1 motor 1 field exciter low (under-) current, *M1FldMinTrip (30.12), FldMinTripDly (45.18)* B9 **M2FexLowCur** F542 1 motor 2 field exciter low (under-) current, *M2FldMinTrip (49.08), FldMinTripDly (45.18)* B10 **COM8Com** F543 5 SDCS-COM-8 communication loss, *Ch0 ComLossCtrl (70.05)*, *Ch0 TimeOut (70.04)*, *Ch2 ComLossCtrl (70.15)*, *Ch2 TimeOut (70.14)* B11 **P2PandMFCom** F544 5 Peer to peer and master-follower communication loss, *ComLossCtrl (30.28)*, *MailBoxCycle1 (94.13)*, *MailBoxCycle2 (94.19)*, *MailBoxCycle3 (94.25)*, *MailBoxCycle4 (94.31)* --- B12 **ApplLoadFail** F545 1 application load failure, see *Diagnosis (9.11)* B13 **LocalCmdLoss** F546 5 local command loss, *LocalLossCtrl (30.27)* B14 **HwFailure** F547 1 hardware failure, see *Diagnosis (9.11)* B15 **FwFailure** F548 1 firmware failure, see *Diagnosis (9.11)* **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.04	**FaultWord4 (fault word 4)** Fault word 4: Bit / Fault text / Fault code and trip level / Comment B0 **ParComp** F549 1 parameter compatibility, the parameter causing the fault can be identified in *Diagnosis (9.11)* B1 **ParMemRead** F550 1 reading the actual parameter set or a user parameter set from either parameter flash or Memory Card failed (checksum fault) B2 **AIRange** F551 4 analog input range, *AI Mon4mA (30.29)* B3 **MechBrake** F552 3 selected motor: mechanical brake, *M1BrakeAckSel (42.02), M1BrakeFltTime (42.05), BrakeFaultFunc (42.06), M1BrakeLongTime (42.12)* --- B4 **TachPolarity** F553 3 selected motor: tacho respectively pulse encoder polarity B5 **TachoRange** F554 3 Overflow of AITacho input B6 reserved F555 reserved for PID-controller B7 **TorqProving** F556 3 selected motor: torque proving, *M1TorqProvTime (42.10),* the Adaptive Program, application program or overriding control providing the acknowledge signal **TorqProvOK** [*AuxCtrlWord2 (7.03)* bit 11] --- B8 **ReversalTime** F557 3 reversal time, *ZeroCurTimeOut (97.19), RevDly (43.14)* B9 reserved F558 no action B10 reserved F559 no action B11 **APFault1** F601 1 Adaptive Program fault 1 --- B12 **APFault2** F602 1 Adaptive Program fault 2 B13 **APFault3** F603 1 Adaptive Program fault 3 B14 **APFault4** F604 1 Adaptive Program fault 4 B15 **APFault5** F605 1 Adaptive Program fault 5 **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>9.05</td><td>UserFaultWord (user defined fault word 1)
User defined fault word. All names are defined by the user via application program:
Bit | Fault text | Fault code and trip level | Comment
B0 UserFault1 F610 1
B1 UserFault2 F611 1
B2 UserFault3 F612 1
B3 UserFault4 F613 1

B4 UserFault5 F614 1
B5 UserFault6 F615 1
B6 UserFault7 F616 1
B7 UserFault8 F617 1

B8 UserFault9 F618 1
B9 UserFault10 F619 1
B10 UserFault11 F620 1
B11 UserFault12 F621 1

B12 UserFault13 F622 1
B13 UserFault14 F623 1
B14 UserFault15 F624 1
B15 UserFault16 F625 1
Int. Scaling: 1 == 1 Type: I Volatile: Y</td><td></td><td></td><td></td><td></td><td>E</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.06	**AlarmWord1 (alarm word 1)** Alarm word 1: Bit / Alarm text / Alarm code and alarm level / Comment B0 **Off2ViaDI** A101 1 **Off2** (Emergency Off / Coast Stop) pending via digital input, Off2 (10.08) B1 **Off3ViaDI** A102 1 **Off3** (E-stop) pending via digital input, *E Stop (10.09)* B2 **DC BreakAck** A103 3 selected motor: DC-breaker acknowledge missing, *DC BreakAck (10.23)* B3 **ConvOverTemp** A104 2 converter overtemperature, shutdown temperature see *MaxBridgeTemp (4.17)*. The converter overtemperature alarm will already appear at approximately 5°C below the shutdown temperature. ------------------------------ B4 **DynBrakeAck** A105 1 selected motor: dynamic braking acknowledge is still pending, *DynBrakeAck (10.22)* B5 **M1OverTemp** A106 2 motor 1 measured overtemperature, *M1AlarmLimTemp (31.06)* B6 **M1OverLoad** A107 2 motor 1 calculated overload (thermal model), *M1AlarmLimLoad (31.03)* B7 reserved A108 4 no action ------------------------------ B8 **M2OverTemp** A109 2 motor 2 measured overtemperature, *M2AlarmLimTemp (49.36)* B9 **M2OverLoad** A110 2 motor 2 calculated overload (thermal model), *M2AlarmLimLoad (49.33)* B10 **MainsLowVolt** A111 3 mains low (under-) voltage, *PwrLossTrip (30.21), UNetMin1 (30.22)*, *UNetMin2 (30.23)* B11 **P2PandMFCom** A112 4 Drive-to-drive and master-follower communication loss, *ComLossCtrl (30.28), MailBoxCycle1 (94.13)*, *MailBoxCycle2 (94.19)*, *MailBoxCycle3 (94.25)*, *MailBoxCycle4 (94.31)* ------------------------------ B12 **COM8Com** A113 4 SDCS-COM-8 communication loss, *Ch0 ComLossCtrl (70.05)*, *Ch0 TimeOut (70.04), Ch2 ComLossCtrl (70.15)*, *Ch2 TimeOut (70.14)* B13 **ArmCurDev** A114 3 armature current deviation B14 **TachoRange** A115 4 Overflow of AITacho input or *M1OvrSpeed (30.16)* respectively *M2OvrSpeed (49.21)* have been changed B15 **BrakeLongFalling** A116 4 selected motor: mechanical brake, *M1BrakeAckSel (42.02), BrakeFaultFunc (42.06), M1BrakeLongTime (42.12)* **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.07	**AlarmWord2 (alarm word 2)** Alarm word 2: Bit / Alarm text / Alarm code and alarm level / Comment B0 **ArmCurRipple** A117 4 armature current ripple, *CurRippleMode (30.18, CurRippleLim (30.19)* B1 **FoundNewAppl** A118 1 found new application on Memory Card, activate application on Memory Card by means of *ParApplSave (16.06)* = **EableAppl** B2 **ApplDiff** A119 1 application on drive and Memory Card are different, activate application on Memory Card by means of *ParApplSave (16.06)* = **EableAppl** B3 **OverVoltProt** A120 3 overvoltage protection active, *OvrVoltProt (30.13)* -- B4 **AutotuneFail** A121 4 autotuning failure, *Diagnosis (9.11)* B5 **MechBrake** A122 4 selected motor: mechanical brake, *BrakeFaultFunc (42.06), M1StrtTorqRefSel (42.07)*, *M2StrtTorqRefSel (49.44)* B6 **FaultSuppres** A123 4 at least one fault message is mask B7 **SpeedScale** A124 4 speed scaling out of range, *M1SpeedScale (50.01)* and *M1BaseSpeed (99.04)*, the parameter causing the alarm can be identified in *Diagnosis (9.11)* -- B8 **SpeedFb** A125 4 selected motor: speed feedback, *M1SpeedFbSel (50.03)*, *SpeedFbFltMode (30.36)*, *SpeedFbFltSel (30.17)* B9 **ExternalDI** A126 4 external alarm via binary input, *ExtAlarmSel (30.32)* B10 **AIRange** A127 4 analog input range, *AI Mon4mA(30.29)* B11 **FieldBusCom** A128 4 fieldbus communication loss, *ComLossCtrl (30.28)* -- B12 **ParRestored** A129 4 The parameters found in flash were found invalid at power-up (checksum fault). The parameters were restored from the parameter backup. B13 **LocalCmdLoss** A130 4 local command loss, *LocalLossCtrl (30.27)* B14 **ParAdded** A131 4 A new firmware with a different amount of parameters was downloaded. The new parameters are set to their default values. The parameters causing the alarm can be identified in *Diagnosis (9.11)*. B15 **ParConflict** A132 4 parameter setting conflict, the parameter causing the alarm can be identified in *Diagnosis (9.11)* **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.08	**AlarmWord3 (alarm word 3)** Alarm word 3: Bit — Alarm text — Alarm code and alarm level — Comment B0 **RetainInv** A133 - retain data invalid B1 **ParComp** A134 4 parameter compatibility, the parameter causing the alarm can be identified in *Diagnosis (9.11)* B2 **ParUpDwnLoad** A135 4 The checksum verification failed during up- or download of parameters. Please try again. B3 **NoAPTaskTime** A136 4 Adaptive Program task for not set in *TimeLevSel (83.04)* --- B4 **SpeedNotZero** A137 1 Re-start of drive is not possible. Speed zero [see *M1ZeroSpeedLim (20.03)* or *M2ZeroSpeedLim (49.04)*] has not been reached [only in case *FlyStart (21.10)* = **StartFrom0**]. In case of a trip set **On** = **Run** = 0 to reset the alarm. B5 **Off2FieldBus** A138 1 **Off2** (Emergency Off / Coast Stop) pending via fieldbus, Off2 (10.08) B6 **Off3FieldBus** A139 1 **Off3** (E-stop) pending via fieldbus, *E Stop (10.09)* B7 **IllgFieldBus** A140 4 the fieldbus parameters in group 51 (fieldbus) are not set according to the fieldbus adapter or the device has not been selected --- B8 **COM8FwVer** A141 4 invalid combination of SDCS-CON-4 firmware and SDCS-COM-8 firmware B9 **MemCardMiss** A142 1 Memory Card missing B10 **MemCardFail** A143 1 checksum failure or wrong Memory Card B11 **APAlarm1** A301 4 Adaptive Program alarm 1 --- B12 **APAlarm2** A302 4 Adaptive Program alarm 2 B13 **APAlarm3** A303 4 Adaptive Program alarm 3 B14 **APAlarm4** A304 4 Adaptive Program alarm 4 B15 **APAlarm5** A305 4 Adaptive Program alarm 5 **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.09	**UserAlarmWord (user defined alarm word 1)** User defined alarm word. All names are defined by the user via application program: Bit Alarm text Alarm code and alarm level Comment B0 **UserAlarm1** A310 4 B1 **UserAlarm2** A311 4 B2 **UserAlarm3** A312 4 B3 **UserAlarm4** A313 4 --- B4 **UserAlarm5** A314 4 B5 **UserAlarm6** A315 4 B6 **UserAlarm7** A316 4 B7 **UserAlarm8** A317 4 --- B8 **UserAlarm9** A318 4 B9 **UserAlarm10** A319 4 B10 **UserAlarm11** A320 4 B11 **UserAlarm12** A321 4 --- B12 **UserAlarm13** A322 4 B13 **UserAlarm14** A323 4 B14 **UserAlarm15** A324 4 B15 **UserAlarm16** A325 4 **Int. Scaling: 1 == 1 Type: I Volatile: Y**					E
9.10	**SysFaultWord (system fault word)** Operating system faults from SDCS-COM-8 board: Bit Fault text Fault code F B0 Factory macro parameter file error default parameters are invalid B1 User macro parameter file error one of the User macros is invalid B2 Non Volatile operating system error AMCOS fault, please contact Your local ABB agent B3 File error in flash problems when writing to the flash memory, please try again --- B4 Internal time level T2 overflow (100 µs) timeout of task T2, if happens frequently please contact Your local ABB agent B5 Internal time level T3 overflow (1 ms) timeout of task T3, if happens frequently please contact Your local ABB agent B6 Internal time level T4 overflow (50 ms) timeout of task T4, if happens frequently please contact Your local ABB agent B7 Internal time level T5 overflow (1 s) timeout of task T5, if happens frequently please contact Your local ABB agent --- B8 State overflow timeout of task State, if happens frequently please contact Your local ABB agent B9 Application window ending overflow application on SDCS-COM-8 faulty B10 Application program overflow application on SDCS-COM-8 faulty B11 Illegal instruction crash of CPU due to EMC or hardware problems --- B12 Register stack overflow overflow due to EMC or firmware bug B13 System stack overflow overflow due to EMC or firmware bug B14 System stack underflow underflow due to crash of CPU or firmware bug B15 reserved - **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.11	**Diagnosis (diagnosis)** **Attention:** *Diagnosis (9.11)* is set to zero by means of **Reset**. Displays diagnostics messages: 0 = no message **Firmware:** 1 = default setting of parameters wrong 2 = parameter flash image too small for all parameters 3 = reserved 4 = illegal write attempt on a signal or write-protected parameter, e.g. writing on *UsedMCW (7.04)* with master-follower. 5 = reserved 6 = wrong type code 7 = an un-initialized interrupted has occurred 8, 9 = reserved 10 = wrong parameter value **Autotuning:** 11 = autotuning aborted by fault or removing the **Run** command [UsedMCW (7.04) bit 3] 12 = autotuning timeout, **Run** command [UsedMCW (7.04) bit 3] is not set in time 13 = motor is still turning, no speed zero indication 14 = field current not zero 15 = armature current not zero 16 = armature voltage measurement circuit open (e.g. not connected) or interrupted, check also current and torque limits 17 = armature circuit and/or armature voltage measurement circuit wrongly connected 18 = no load connected to armature circuit 19 = invalid nominal armature current setting; armature current *M1MotNomCur (99.03)* is set to zero 20 = field current does not decrease when the excitation is switched off 21 = field current actual doesn't reach field current reference; no detection of field resistance; field circuit open (e.g. not connected) respectively interrupted 22 = no writing of control parameters of speed controller 23 = tacho adjustment faulty or not OK or the tacho voltage is too high during autotuning 24 = tuning of speed controller, speed feedback assistant or tacho fine tuning not possible due to speed limitation - see e.g. *M1SpeedMin (20.01)* and *M1SpeedMax (20.02)* 25 = Tuning of speed controller, speed feedback assistant or tacho fine tuning not possible due to voltage limitation. During the tuning of the speed controller, the speed feedback assistant or the tacho fine tuning base speed [*M1BaseSpeed (99.04)*] might be reached. Thus full armature voltage [*M1NomVolt (99.02)*] is necessary. In case the mains voltage is too low to provide for the needed armature voltage the autotuning procedure is canceled. Check and adapt if needed: Mains voltage *M1NomVolt (99.02)* *M1BaseSpeed (99.04)* 26 = field weakening not allowed, see *M1SpeedFbSel (50.03)* and *FldCtrlMode (44.01)* 27 = discontinuous current limit could not be determined due to low current limitation in *M1CurLimBrdg1 (20.12)* or M1CurLimBrdg2 *(20.13)* 28 = filed current autotuning wrongly started in armature converter, please use the field exciter 29 = no field exciter selected, see *M1UsedFexType (99.12)* 30 = reserved 30 = DCS800 Control Panel up- or download not started 32 = DCS800 Control Panel data not up- or downloaded in time 33 = reserved 34 = DCS800 Control Panel up -or download checksum faulty 35 = DCS800 Control Panel up- or download software faulty 36 = DCS800 Control Panel up- or download verification failed 37 - 40 reserved 41 = The flash is written to cyclic by Adaptive Program (e.g. block ParWrite) or application program. Cyclic saving of values in the flash will damage it! Do not write cyclic on the flash! 42 - 49 reserved	0	65535	0	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	Hardware: 50 = parameter flash faulty (erase) 51 = parameter flash faulty (program) 52 = check connector X12 on SDCS-CON-4 and connector X12 and X22 on SDCS-PIN-4/51 53 - 69 reserved **A132 ParConflict** (alarm parameter setting conflict): 70 = no field reversal possible due to *ForceFldDir (45.07)* = **ExtReverse** 71 = flux linearization parameters not consistent 72 = reserved 73 = armature data not consistent. Check if: – *M1NomCur (99.03)* is set to zero, – *M1NomVolt (99.02)* and *M1NomCur (99.03)* are fitting with the drive. In case they are much smaller than the drive the internal calculation of *M1ArmL (43.09)* and *M1ArmR (43.10)* can cause an internal overflow. Set *M1ArmL (43.09)* and *M1ArmR (43.10)* to zero. For *M1ArmL (43.09)* following limitation is valid: $\frac{(43.09)*4096*(99.03)}{1000*(99.02)} \leq 32767$ For *M1ArmR (43.10)* following limitation is valid: $\frac{(43.10)*4096*(99.03)}{1000*(99.02)} \leq 32767$ 74 - 76 reserved					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	77 = Encoder 1 parameters for motor 1 not consistent. Check: *SpeedScaleAct (2.29)* *M1EncMeasMode (50.02)* *M1EncPulseNo (50.04)* At scaling speed - see *SpeedScaleAct (2.29)* - the pulse frequency must be greater than 600 Hz according to following formula: $f \geq 600\,Hz = \frac{ppr * evaluation * speed\ scaling}{60\,s}$ $f \geq 600\,Hz = \frac{(50.04) * (50.02) * (2.29)}{60\,s}$ E.g. the speed scaling must be ≥ 9 rpm for a pulse encoder with 1024 pulses and A+-/B+- evaluation. 78 = Encoder 1 parameters for motor 2 not consistent. Check: *SpeedScaleAct (2.29)* *M2EncMeasMode (49.23)* *M2EncPulseNo (49.25)* At scaling speed - see *SpeedScaleAct (2.29)* - the pulse frequency must be greater than 600 Hz according to following formula: $f \geq 600\,Hz = \frac{ppr * evaluation * speed\ scaling}{60\,s}$ $f \geq 600\,Hz = \frac{(49.25) * (49.23) * (2.29)}{60\,s}$ E.g. the speed scaling must be ≥ 9 rpm for a pulse encoder with 1024 pulses and A+-/B+- evaluation. 79 = Encoder 2 parameters not consistent. Check: *SpeedScaleAct (2.29)* *Enc2MeasMode (50.18)* *Enc2PulseNo (50.19)* At scaling speed - see *SpeedScaleAct (2.29)* - the pulse frequency must be greater than 600 Hz according to following formula: $f \geq 600\,Hz = \frac{ppr * evaluation * speed\ scaling}{60\,s}$ $f \geq 600\,Hz = \frac{(50.19) * (50.18) * (2.29)}{60\,s}$ E.g. the speed scaling must be ≥ 9 rpm for a pulse encoder with 1024 pulses and A+-/B+- evaluation. **Autotuning:** 80 = speed does not reach setpoint (EMF control) 81 = motor is not accelerating or wrong tacho polarity (tacho / encoder) 82 = not enough load (too low inertia) for the detection of speed controller parameters 83 = drive not in speed control mode, see *TorqSel (26.01)*, *TorqSelMod (26.03)*, *TorqMuxMode (26.04)* 84 - 89 reserved					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	Thyristor diagnosis: 90 = shortcut caused by V1 91 = shortcut caused by V2 92 = shortcut caused by V3 93 = shortcut caused by V4 94 = shortcut caused by V5 95 = shortcut caused by V6 96 = thyristor block test failed 97 = shortcut caused by V15 or V22 98 = shortcut caused by V16 or V23 99 = shortcut caused by V11 or V24 100 = shortcut caused by V12 or V25 101 = shortcut caused by V13 or V26 102 = shortcut caused by V14 or V21 103 = motor connected to ground 104 = armature winding is not connected 105 - 120 reserved **AI monitoring:** 121 = AI1 below 4 mA 122 = AI2 below 4 mA 123 = AI3 below 4 mA 124 = AI4 below 4 mA 125 = AI5 below 4 mA 126 = AI6 below 4 mA 127 = AITAC below 4 mA 128 - 149 reserved **Option modules:** 150 = fieldbus module missing see *CommModule (98.02)* 151 = SDCS-COM-8 for DDCS- respectively fieldbus communication missing see *CommModule (98.02)* 152 = SDCS-COM-8 for master-follower communication missing see group 70 153 = reserved 154 = RMBA-xx module missing see group 98 155 = RAIO-xx in option slot on SDCS-CON-4 missing see group 98 156 = RAIO-xx in option slot on AIMA missing see group 98 157 = RDIO-xx in option slot on SDCS-CON-4 missing see group 98 158 = RDIO-xx in option slot on AIMA missing see group 98 159 = RTAC-xx in option slot on SDCS-CON-4 missing see group 98 160 = RTAC-xx in option slot on AIMA missing see group 98 161 = reserved 162 = SDCS-IOB-2x respectively SDCS-IOB-3 connection does not match selection in *IO BoardConfig (98.15)* 163 = SDCS-DSL-4 missing see group 94 (needed for DCSLink) 164 = SDCS-DSL-4 missing see group 52 (needed for Modbus) **A134 ParComp** (alarm parameter compatibility conflict): 10000 … 19999 = the parameter with the compatibility conflict can be identified by means of the last 4 digits **ParNoCyc** (notice parameter not cyclic): 20000 … 29999 = the not cyclic parameter, which is being written to by means of a pointer parameter [e.g. *DsetXVal1 (90.01)*], can be identified by means of the last 4 digits **F548 FwFailure** (fault firmware failure): 20000 … 29999 = the read only parameter, which is being written to by means of a pointer parameter [e.g. *DsetXVal1 (90.01)*], Adaptive Program or application program, can be identified by means of the last 4 digits					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	Thyristor diagnosis: 30000 = possibly trigger pulse channels are mixed up 31xdd = V1 or V11 not conducting 32xdd = V2 or V12 not conducting 33xdd = V3 or V13 not conducting 34xdd = V4 or V14 not conducting 35xdd = V5 or V15 not conducting 36xdd = V6 or V16 not conducting x = 0: only a single thyristor in bridge 1 is not conducting (e.g. 320dd means V2 respectively V12 is not conducting) x = 1 … 6: additionally a second thyristor in bridge 1 is no conducting (e.g. 325dd means V2 and V5 respectively V12 and V15 are not conducting) dd = don't care, the numbers of this digits do not carry any information about the thyristors of the first bridge. Example: – 36030: means V16 in bridge 1 and V23 in bridge 2 are not conducting 3dd1y = V21 not conducting 3dd2y = V22 not conducting 3dd3y = V23 not conducting 3dd4y = V24 not conducting 3dd5y = V25 not conducting 3dd6y = V26 not conducting y = 0: only a single thyristor in bridge 2 is not conducting (e.g. 3dd20 means V22 is not conducting) y = 1 … 6: additionally a second thyristor in bridge 2 is no conducting (e.g. 3dd25 means V22 and V25 are not conducting) dd = don't care, the numbers of this digits do not carry any information about the thyristors of the second bridge. Example: – 36030: means V16 in bridge 1 and V23 in bridge 2 are not conducting **A124 SpeedScale** (alarm speed scaling): 40000 … 49999 = the parameter with the speed scaling conflict can be identified by means of the last 4 digits **F549 ParComp** (fault parameter compatibility conflict): 50000 … 59999= the parameter with the compatibility conflict can be identified by means of the last 4 digits **F545 ApplLoadFail** (ControlBuilder DCS800 application programming): 64110 = task not configured 64112 = attempt to run an illegal copy of a program 64113 = retain data invalid caused by SDCS-CON-4 hardware problem 64125 = 5 ms task halted (e.g. task contains an endless loop) 64126 = 10 ms task halted (e.g. task contains an endless loop) 64127 = 20 ms task halted (e.g. task contains an endless loop) 64128 = 50 ms task halted (e.g. task contains an endless loop) 64129 = 100 ms task halted (e.g. task contains an endless loop) 64130 = 200 ms task halted (e.g. task contains an endless loop) 64131 = 500 ms task halted (e.g. task contains an endless loop) 64132 = 1000 ms task halted (e.g. task contains an endless loop) 64133 = application program is using an unsupported DCS800 Drive library version **Int. Scaling: 1 == 1 Type: I Volatile: Y**					
9.12	**LastFault (last fault)** Displays the last fault: **F<Fault code> <FaultName>** (e.g. **F2 ArmOverCur**) **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C
9.13	**2ndLastFault (2nd last fault)** Displays the 2nd last fault: **F<Fault code> <FaultName>** (e.g. **F2 ArmOverCur**) **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.14	**3rd LastFault (3rd last fault)** Displays the 3rd last fault: **F<Fault code> <FaultName>** (e.g. **F2 ArmOverCur**) **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C
9.15	**Unused**					
9.16	**Unused**					
9.17	**M1FexAlarmWord (motor 1 field exciter alarm word)** Motor 1 field exciter alarm word : Bit Alarm text Alarm code Comment B0 reserved B1 reserved B2 reserved B3 reserved --- B4 reserved B5 reserved B6 reserved B7 reserved --- B8 reserved B9 reserved B10 reserved B11 reserved --- B12 reserved B13 reserved B14 reserved B15 reserved **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E
9.18	**M1FexFaultWord (motor 1 field exciter fault word)** Motor 1 field exciter fault word : Bit Fault text Fault code Comment B0 reserved B1 reserved B2 reserved B3 reserved --- B4 reserved B5 reserved B6 reserved B7 reserved --- B8 reserved B9 reserved B10 reserved B11 reserved --- B12 reserved B13 reserved B14 reserved B15 reserved **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
9.19	**M2FexAlarmWord (motor 2 field exciter alarm word)** Motor 2 field exciter alarm word : Bit Alarm text Alarm code Comment B0 reserved B1 reserved B2 reserved B3 reserved --- B4 reserved B5 reserved B6 reserved B7 reserved --- B8 reserved B9 reserved B10 reserved B11 reserved --- B12 reserved B13 reserved B14 reserved B15 reserved **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E
9.20	**M2FexFaultWord (motor 2 field exciter fault word)** Motor 2 field exciter fault word : Bit Fault text Fault code Comment B0 reserved B1 reserved B2 reserved B3 reserved --- B4 reserved B5 reserved B6 reserved B7 reserved --- B8 reserved B9 reserved B10 reserved B11 reserved --- B12 reserved B13 reserved B14 reserved B15 reserved **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>Group 10</td><td colspan="6">Start / stop select</td></tr>
<tr><td>10.01</td><td>CommandSel (command selector)
UsedMCW (7.04) selector:
0 = Local I/O Drive is controlled via local I/O.
Reset (10.03) = DI6; UsedMCW (7.04) bit 7, default
OnOff1 (10.15) = DI7; UsedMCW (7.04) bit 0, default and
StartStop (10.16) = DI8; UsedMCW (7.04) bit 3, default
1 = MainCtrlWord drive is controlled via MainCtrlWord (7.01)
2 = Key Automatic switchover from MainCtrlWord to Local I/O in case of F528 FieldBusCom [FaultWord2 (9.02) bit 11]. It is still possible to control the drive via local I/O. OnOff1 (10.15) = DI7; UsedMCW (7.04) bit 0, default and StartStop (10.16) = DI8; UsedMCW (7.04) bit 3, default. The used speed reference is set by means of FixedSpeed1 (23.02).
3 = 12PLink Drive is controlled from 12-pulse master (OnOff1, StartStop, Off2N and Reset). Only available when OperModeSel (43.01) = 12P ParaSla or 12P SerSla.
4 = FexLink Drive is controlled from field exciter master (OnOff1, StartStop and Reset). Only available when OperModeSel (43.01) = FieldExciter.
Note:
Local control mode has higher priority than the selection made with CommandSel (10.01).
Note:
The commands Off2 (10.08), E Stop (10.09) and Reset (10.03) are always active (in case they are assigned) regardless of CommandSel (10.01) setting.
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>Local I/O</td><td>FexLink</td><td>Local I/O</td><td>-</td><td>C</td></tr>
<tr><td>10.02</td><td>Direction (direction of rotation)
Binary signal for Direction. Direction (10.02) allows to change the direction of rotation by negating the speed reference in remote operation:
0 = NotUsed default
1 = DI1 1 = Reverse, 0 = Forward
2 = DI2 1 = Reverse, 0 = Forward
3 = DI3 1 = Reverse, 0 = Forward
4 = DI4 1 = Reverse, 0 = Forward
5 = DI5 1 = Reverse, 0 = Forward
6 = DI6 1 = Reverse, 0 = Forward
7 = DI7 1 = Reverse, 0 = Forward
8 = DI8 1 = Reverse, 0 = Forward
9 = DI9 1 = Reverse, 0 = Forward, only available with digital extension board
10 = DI10 1 = Reverse, 0 = Forward, only available with digital extension board
11 = DI11 1 = Reverse, 0 = Forward, only available with digital extension board
12 = MCW Bit11 1 = Reverse, 0 = Forward, MainCtrlWord (7.01) bit 11
13 = MCW Bit12 1 = Reverse, 0 = Forward, MainCtrlWord (7.01) bit 12
14 = MCW Bit13 1 = Reverse, 0 = Forward, MainCtrlWord (7.01) bit 13
15 = MCW Bit14 1 = Reverse, 0 = Forward, MainCtrlWord (7.01) bit 14
16 = MCW Bit15 1 = Reverse, 0 = Forward, MainCtrlWord (7.01) bit 15
17 = ACW Bit12 1 = Reverse, 0 = Forward, AuxCtrlWord (7.02) bit 12
18 = ACW Bit13 1 = Reverse, 0 = Forward, AuxCtrlWord (7.02) bit 13
19 = ACW Bit14 1 = Reverse, 0 = Forward, AuxCtrlWord (7.02) bit 14
20 = ACW Bit15 1 = Reverse, 0 = Forward, AuxCtrlWord (7.02) bit 15
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>NotUsed</td><td>ACW Bit15</td><td>NotUsed</td><td>-</td><td>C</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.03	**Reset (Reset command)** Binary signal for **Reset**, *UsedMCW (7.04)* bit 7: 0 = **NotUsed** 1 = **DI1** **Reset** by rising edge (0 → 1) 2 = **DI2** **Reset** by rising edge (0 → 1) 3 = **DI3** **Reset** by rising edge (0 → 1) 4 = **DI4** **Reset** by rising edge (0 → 1) 5 = **DI5** **Reset** by rising edge (0 → 1) 6 = **DI6** **Reset** by rising edge (0 → 1), default 7 = **DI7** **Reset** by rising edge (0 → 1) 8 = **DI8** **Reset** by rising edge (0 → 1) 9 = **DI9** **Reset** by rising edge (0 → 1), only available with digital extension board 10 = **DI10** **Reset** by rising edge (0 → 1), only available with digital extension board 11 = **DI11** **Reset** by rising edge (0 → 1), only available with digital extension board 12 = **MCW Bit11** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** **Reset** by rising edge (0 → 1), *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** **Reset** by rising edge (0 → 1), *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** **Reset** by rising edge (0 → 1), *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** **Reset** by rising edge (0 → 1), *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	DI6	-	C
10.04	**SyncCommand (synchronization command for position counter encoder 1)** Activation of synchronization for pulse encoder 1 and setting of the binary input signal. At the synchronization event [*AuxCtrlWord (7.02)* bit 9 **SyncCommand**] the position counter is initialized with following values: – *PosCountInitLo (50.08)* is written into *PosCountLow (3.07)* and – *PosCountInitHi (50.09)* is written into *PosCountHigh (3.08)*. At the same time *AuxStatWord (8.02*) bit 5 **SyncRdy** is set to 1. The synchronization can be inhibited by setting *AuxCtrlWord (7.02)* bit 10 **SyncDisable** to 1. The synchronization event is selected by: 0 = **NotUsed** default 1 = **DI7+** rising edge (0 → 1) taken from DI7 2 = **DI7Hi&Z** DI7 = 1 and rising edge (0 → 1) taken from zero channel pulse encoder 3 = **DI7Hi&Z Fwd** DI7 = 1 and rising edge (0 → 1) taken from zero channel pulse encoder, motor rotating forward 4 = **DI7Hi&Z Rev** DI7 = 1 and rising edge (0 → 1) taken from zero channel pulse encoder, motor rotating reverse 5 = **DI7-** falling edge (1 → 0) taken from DI7 6 = **DI7Lo&Z** DI7 = 0 and rising edge (0 → 1) taken from zero channel pulse encoder 7 = **DI7Lo&Z Fwd** DI7 = 0 and rising edge (0 → 1) taken from zero channel pulse encoder, motor rotating forward 8 = **DI7Lo&Z Rev** DI7 = 0 and rising edge (0 → 1) taken from zero channel pulse encoder, motor rotating reverse 9 = **Z** rising edge (0 → 1) taken from zero channel pulse encoder 10 = **SyncCommand** rising edge (0 → 1) taken from *AuxCtrlWord (7.02*) bit 9 **Note:** Forward rotation means that encoder channel A pulses lead channel B pulses by 90° (electrical). Reverse rotation means that encoder channel B pulses lead channel A pulses by 90° (electrical). **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	SyncCommand	NotUsed	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.05	**SyncCommand2 (synchronization command for position counter encoder 2)** Activation of synchronization for pulse encoder 2 and setting of the binary input signal. At the synchronization event [*AuxCtrlWord (7.02)* bit 9 **SyncCommand**] the position counter is initialized with following values: – *PosCount2InitLo (50.21)* is written into *PosCount2Low (3.05)* and – *PosCount2InitHi (50.22)* is written into *PosCount2High (3.06)*. At the same time *AuxStatWord (8.02)* bit 5 **SyncRdy** is set to 1. The synchronization can be inhibited by setting *AuxCtrlWord (7.02)* bit 10 **SyncDisable** to 1. The synchronization event is selected by: 0 = **NotUsed** default 1 = **DI7+** rising edge (0 → 1) taken from DI7 2 = **DI7Hi&Z** DI7 = 1 and rising edge (0 → 1) taken from zero channel pulse encoder 3 = **DI7Hi&Z Fwd** DI7 = 1 and rising edge (0 → 1) taken from zero channel pulse encoder, motor rotating forward 4 = **DI7Hi&Z Rev** DI7 = 1 and rising edge (0 → 1) taken from zero channel pulse encoder, motor rotating reverse 5 = **DI7-** falling edge (1 → 0) taken from DI7 6 = **DI7Lo&Z** DI7 = 0 and rising edge (0 → 1) taken from zero channel pulse encoder 7 = **DI7Lo&Z Fwd** DI7 = 0 and rising edge (0 → 1) taken from zero channel pulse encoder, motor rotating forward 8 = **DI7Lo&Z Rev** DI7 = 0 and rising edge (0 → 1) taken from zero channel pulse encoder, motor rotating reverse 9 = **Z** rising edge (0 → 1) taken from zero channel pulse encoder 10 = **SyncCommand** rising edge (0 → 1) taken from *AuxCtrlWord (7.02)* bit 9 **Note:** Forward rotation means that encoder channel A pulses lead channel B pulses by 90° (electrical). Reverse rotation means that encoder channel B pulses lead channel A pulses by 90° (electrical). **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	SyncCommand	NotUsed	-	E
10.06	**MotFanAck (motor fan acknowledge)** The drive trips with **F523 ExtFanAck** [*FaultWord2 (9.02)* bit 6] if a digital input for an external fan is selected and the acknowledge is missing for 10 seconds: 0 = **NotUsed** no reaction 1 = **DI1** 1= **acknowledge** OK, 0 = no **acknowledge** 2 = **DI2** 1= **acknowledge** OK, 0 = no **acknowledge**, default 3 = **DI3** 1= **acknowledge** OK, 0 = no **acknowledge** 4 = **DI4** 1= **acknowledge** OK, 0 = no **acknowledge** 5 = **DI5** 1= **acknowledge** OK, 0 = no **acknowledge** 6 = **DI6** 1= **acknowledge** OK, 0 = no **acknowledge** 7 = **DI7** 1= **acknowledge** OK, 0 = no **acknowledge** 8 = **DI8** 1= **acknowledge** OK, 0 = no **acknowledge** 9 = **DI9** 1= **acknowledge** OK, 0 = no **acknowledge,** only available with digital extension board 10 = **DI10** 1= **acknowledge** OK, 0 = no **acknowledge,** only available with digital extension board 11 = **DI11** 1= **acknowledge** OK, 0 = no **acknowledge,** only available with digital extension board **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI11	DI2	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.07	**HandAuto (Hand/Auto command)** Binary signal to switch between **Hand** (**Local I/O**) and **Auto** (**MainCtrlWord**) control. Thus the selection made by *CommandSel (10.01)* is overwritten: 0 = **NotUsed** default 1 = **DI1** 1 = **Auto**, 0 = **Hand** 2 = **DI2** 1 = **Auto**, 0 = **Hand** 3 = **DI3** 1 = **Auto**, 0 = **Hand** 4 = **DI4** 1 = **Auto**, 0 = **Hand** 5 = **DI5** 1 = **Auto**, 0 = **Hand** 6 = **DI6** 1 = **Auto**, 0 = **Hand** 7 = **DI7** 1 = **Auto**, 0 = **Hand** 8 = **DI8** 1 = **Auto**, 0 = **Hand** 9 = **DI9** 1 = **Auto**, 0 = **Hand,** only available with digital extension board 10 = **DI10** 1 = **Auto**, 0 = **Hand**, only available with digital extension board 11 = **DI11** 1 = **Auto**, 0 = **Hand**, only available with digital extension board 12 = **MCW Bit11** 1 = **Auto**, 0 = **Hand,** *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** 1 = **Auto**, 0 = **Hand,** *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** 1 = **Auto**, 0 = **Hand,** *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** 1 = **Auto**, 0 = **Hand,** *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** 1 = **Auto**, 0 = **Hand,** *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** 1 = **Auto**, 0 = **Hand,** *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** 1 = **Auto**, 0 = **Hand,** *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** 1 = **Auto**, 0 = **Hand,** *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** 1 = **Auto**, 0 = **Hand,** *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	C
10.08	**Off2 (Off2 command, electrical disconnect)** Binary signal for **Off2** (Emergency Off / Coast Stop), *UsedMCW (7.04)* bit 1. For fastest reaction use fast digital inputs DI7 or DI8: 0 = **NotUsed** 1 = **DI1** 1= no **Off2**, 0 = **Off2** active 2 = **DI2** 1= no **Off2**, 0 = **Off2** active 3 = **DI3** 1= no **Off2**, 0 = **Off2** active 4 = **DI4** 1= no **Off2**, 0 = **Off2** active, default 5 = **DI5** 1= no **Off2**, 0 = **Off2** active 6 = **DI6** 1= no **Off2**, 0 = **Off2** active 7 = **DI7** 1= no **Off2**, 0 = **Off2** active 8 = **DI8** 1= no **Off2**, 0 = **Off2** active 9 = **DI9** 1= no **Off2**, 0 = **Off2** active, only available with digital extension board 10 = **DI10** 1= no **Off2**, 0 = **Off2** active, only available with digital extension board 11 = **DI11** 1= no **Off2**, 0 = **Off2** active, only available with digital extension board 12 = **MCW Bit11** 1= no **Off2**, 0 = **Off2** active, *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** 1= no **Off2**, 0 = **Off2** active, *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** 1= no **Off2**, 0 = **Off2** active, *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** 1= no **Off2**, 0 = **Off2** active, *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** 1= no **Off2**, 0 = **Off2** active, *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** 1= no **Off2**, 0 = **Off2** active, *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** 1= no **Off2**, 0 = **Off2** active, *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** 1= no **Off2**, 0 = **Off2** active, *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** 1= no **Off2**, 0 = **Off2** active, *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	DI4	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.09	**E Stop (emergency stop command)** Binary signal for **Off3** (E-Stop), *UsedMCW (7.04)* bit 2. For fastest reaction use fast digital inputs DI7 or DI8: 0 = **NotUsed** 1 = **DI1** 1= no **E Stop**, 0 = **E Stop** active 2 = **DI2** 1= no **E Stop**, 0 = **E Stop** active 3 = **DI3** 1= no **E Stop**, 0 = **E Stop** active 4 = **DI4** 1= no **E Stop**, 0 = **E Stop** active 5 = **DI5** 1= no **E Stop**, 0 = **E Stop** active, default 6 = **DI6** 1= no **E Stop**, 0 = **E Stop** active 7 = **DI7** 1= no **E Stop**, 0 = **E Stop** active 8 = **DI8** 1= no **E Stop**, 0 = **E Stop** active 9 = **DI9** 1= no **E Stop**, 0 = **E Stop** active, only available with digital extension board 10 = **DI10** 1= no **E Stop**, 0 = **E Stop** active, only available with digital extension board 11 = **DI11** 1= no **E Stop**, 0 = **E Stop** active, only available with digital extension board 12 = **MCW Bit11** 1= no **E Stop**, 0 = **E Stop** active, *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** 1= no **E Stop**, 0 = **E Stop** active, *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** 1= no **E Stop**, 0 = **E Stop** active, *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** 1= no **E Stop**, 0 = **E Stop** active, *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** 1= no **E Stop**, 0 = **E Stop** active, *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** 1= no **E Stop**, 0 = **E Stop** active, *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** 1= no **E Stop**, 0 = **E Stop** active, *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** 1= no **E Stop**, 0 = **E Stop** active, *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** 1= no **E Stop**, 0 = **E Stop** active, *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	DI5	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.10	**ParChange (parameter change)** Binary signal to release either **Motor1/User1** or **Motor2/User2**. The choice to release **Motor1/2** (shared motion) or macros **User1/2** is defined by means of *MacroChangeMode (16.05):* 0 = **NotUsed** default 1 = **DI1** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0) 2 = **DI2** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0) 3 = **DI3** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0) 4 = **DI4** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0) 5 = **DI5** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0) 6 = **DI6** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0) 7 = **DI7** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0) 8 = **DI8** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0) 9 = **DI9** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), only available with digital extension board 10 = **DI10** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), only available with digital extension board 11 = **DI11** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), only available with digital extension board 12 = **MCW Bit11** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** switch to **Motor2/User2** by rising edge (0 → 1), switch to **Motor1/User1** by falling edge (1 → 0), *AuxCtrlWord (7.02)* bit 15 **Note:** The macro (**User1**/**User2**) selection made by *ParChange (10.10*) overrides the selection made with *ApplMacro (99.08*). It takes about 2 s, until the new parameter values are active. **Note:** If **User1** is active *AuxStatWord (8.02)* bit 3 is set. If **User2** is active *AuxStatWord (8.02)* bit 4 is set. **Note:** In case macro **User1** or **User2** is loaded by means of *ParChange (10.10)* it is not saved into the flash and thus not valid after the next power on.	NotUsed	ACW Bit15	NotUsed	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	Note: When changing parameters in a user macro first call the macro with *ApplMacro (99.08*), then change the parameters and save them with *ApplMacro (99.08*). **Note:** The motor (**Motor1**/**Motor2**) selection can be made in drive state **RdyOn** and **RdyRun**. It takes about 20 ms, to switch between values. **Note:** *ParChange (10.10)* itself is not overwritten. **Int. Scaling: 1 == 1 Type: C Volatile: N**					
10.11	**Unused**					
10.12	**Unused**					
10.13	**OvrVoltProt (over voltage protection triggered)** As soon as the overvoltage protection unit is triggered **A120 OverVoltProt** [*AlarmWord2 (9.07)* bit 3] is set: 0 = **NotUsed** default 1 = **DI1** 1 = **triggered,** 0 = not **triggered** 2 = **DI2** 1 = **triggered,** 0 = not **triggered** 3 = **DI3** 1 = **triggered,** 0 = not **triggered** 4 = **DI4** 1 = **triggered,** 0 = not **triggered** 5 = **DI5** 1 = **triggered,** 0 = not **triggered** 6 = **DI6** 1 = **triggered,** 0 = not **triggered** 7 = **DI7** 1 = **triggered,** 0 = not **triggered** 8 = **DI8** 1 = **triggered,** 0 = not **triggered** 9 = **DI9** 1 = **triggered,** 0 = not **triggered** 10 = **DI10** 1 = **triggered,** 0 = not **triggered** 11 = **DI11** 1 = **triggered,** 0 = not **triggered** **Note:** *OvrVoltProt (10.13)* is only released when drive is in field exciter mode. – *OperModeSel (43.01)* = **FieldConv** **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI8	NotUsed	-	E
10.14	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.15	**OnOff1 (On/Off1 command)** Binary signal for **OnOff1**, *UsedMCW (7.04)* bit 0: 0 = **NotUsed** 1 = **DI1** **On** by rising edge (0 → 1), 0 = **Off1** 2 = **DI2** **On** by rising edge (0 → 1), 0 = **Off1** 3 = **DI3** **On** by rising edge (0 → 1), 0 = **Off1** 4 = **DI4** **On** by rising edge (0 → 1), 0 = **Off1** 5 = **DI5** **On** by rising edge (0 → 1), 0 = **Off1** 6 = **DI6** **On** by rising edge (0 → 1), 0 = **Off1** 7 = **DI7** **On** by rising edge (0 → 1), 0 = **Off1**, default 8 = **DI8** **On** by rising edge (0 → 1), 0 = **Off1** 9 = **DI9** **On** by rising edge (0 → 1), 0 = **Off1**, only available with digital extension board 10 = **DI10** **On** by rising edge (0 → 1), 0 = **Off1**, only available with digital extension board 11 = **DI11** **On** by rising edge (0 → 1), 0 = **Off1**, only available with digital extension board 12 = **MCW Bit11** **On** by rising edge (0 → 1), 0 = **Off1**, *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** **On** by rising edge (0 → 1), 0 = **Off1**, *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** **On** by rising edge (0 → 1), 0 = **Off1**, *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** **On** by rising edge (0 → 1), 0 = **Off1**, *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** **On** by rising edge (0 → 1), 0 = **Off1**, *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** **On** by rising edge (0 → 1), 0 = **Off1**, *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** **On** by rising edge (0 → 1), 0 = **Off1**, *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** **On** by rising edge (0 → 1), 0 = **Off1**, *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** **On** by rising edge (0 → 1), 0 = **Off1**, *AuxCtrlWord (7.02)* bit 15 21 = **DI7DI8** **On** and **Start** by rising edge (0 → 1) of DI7, **Stop** and **Off1** by falling edge (1 → 0) of DI8. Following settings apply: *OnOff1 (10.15)* = *StartStop (10.16)* = **DI7DI8**. **Note:** To give **On** and **Run** at the same time set *OnOff1 (10.15)* = *StartStop (10.16)*. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI7DI8	DI7	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.16	**StartStop (Start/Stop command)** Binary signal for **StartStop**, *UsedMCW (7.04)* bit 3: 0 = **NotUsed** 1 = **DI1** **Start** by rising edge (0 → 1), 0 = **Stop** 2 = **DI2** **Start** by rising edge (0 → 1), 0 = **Stop** 3 = **DI3** **Start** by rising edge (0 → 1), 0 = **Stop** 4 = **DI4** **Start** by rising edge (0 → 1), 0 = **Stop** 5 = **DI5** **Start** by rising edge (0 → 1), 0 = **Stop** 6 = **DI6** **Start** by rising edge (0 → 1), 0 = **Stop** 7 = **DI7** **Start** by rising edge (0 → 1), 0 = **Stop** 8 = **DI8** **Start** by rising edge (0 → 1), 0 = **Stop**, default 9 = **DI9** **Start** by rising edge (0 → 1), 0 = **Stop**, only available with digital extension board 10 = **DI10** **Start** by rising edge (0 → 1), 0 = **Stop**, only available with digital extension board 11 = **DI11** **Start** by rising edge (0 → 1), 0 = **Stop**, only available with digital extension board 12 = **MCW Bit11** **Start** by rising edge (0 → 1), 0 = **Stop**, *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** **Start** by rising edge (0 → 1), 0 = **Stop**, *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** **Start** by rising edge (0 → 1), 0 = **Stop**, *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** **Start** by rising edge (0 → 1), 0 = **Stop**, *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** **Start** by rising edge (0 → 1), 0 = **Stop**, *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** **Start** by rising edge (0 → 1), 0 = **Stop**, *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** **Start** by rising edge (0 → 1), 0 = **Stop**, *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** **Start** by rising edge (0 → 1), 0 = **Stop**, *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** **Start** by rising edge (0 → 1), 0 = **Stop**, *AuxCtrlWord (7.02)* bit 15 21 = **DI7DI8** **On** and **Start** by rising pulse (0 → 1) of DI7, **Stop** and **Off1** by falling pulse (1 → 0) of DI8. Following settings apply: *OnOff1 (10.15)* = *StartStop (10.16)* = **DI7DI8**. **Note:** To give **On** and **Run** at the same time set *OnOff1 (10.15)* = *StartStop (10.16)*. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI7DI8	DI8	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.17	**Jog1 (jogging 1 command)** Binary signal for **Jog1**. Selects speed reference set in *FixedSpeed1 (23.02):* 0 = **NotUsed** default 1 = **DI1** 1= **Jog1** active**,** 0 = no **Jog1** 2 = **DI2** 1= **Jog1** active**,** 0 = no **Jog1** 3 = **DI3** 1= **Jog1** active**,** 0 = no **Jog1** 4 = **DI4** 1= **Jog1** active**,** 0 = no **Jog1** 5 = **DI5** 1= **Jog1** active**,** 0 = no **Jog1** 6 = **DI6** 1= **Jog1** active**,** 0 = no **Jog1** 7 = **DI7** 1= **Jog1** active**,** 0 = no **Jog1** 8 = **DI8** 1= **Jog1** active**,** 0 = no **Jog1** 9 = **DI9** 1= **Jog1** active**,** 0 = no **Jog1,** only available with digital extension board 10 = **DI10** 1= **Jog1** active**,** 0 = no **Jog1,** only available with digital extension board 11 = **DI11** 1= **Jog1** active**,** 0 = no **Jog1,** only available with digital extension board 12 = **MCW Bit11** 1= **Jog1** active**,** 0 = no **Jog1,** *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** 1= **Jog1** active**,** 0 = no **Jog1,** *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** 1= **Jog1** active**,** 0 = no **Jog1,** *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** 1= **Jog1** active**,** 0 = no **Jog1,** *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** 1= **Jog1** active**,** 0 = no **Jog1,** *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** 1= **Jog1** active**,** 0 = no **Jog1,** *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** 1= **Jog1** active**,** 0 = no **Jog1,** *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** 1= **Jog1** active**,** 0 = no **Jog1,** *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** 1= **Jog1** active**,** 0 = no **Jog1,** *AuxCtrlWord (7.02)* bit 15 **Note:** *Jog2 (10.18)* overrides *Jog1 (10.17)* **Note:** *CommandSel (10.01*) = **Local I/O**: – The drive has to be in state **RdyRun** (**RdyRef** is still zero). When **Jog1** command is given the drives sets automatically **RampOutZero** = **RampHold** = **RampInZero** = 0 [see *MainCtrlWord (7.01)*] and goes into state **Running** and turns with speed set in *FixedSpeed1 (23.02).* *CommandSel (10.01)* = **MainCtrlWord**: – The drive has to be in state **RdyRun** (**RdyRef** is still zero). **RampOutZero**, **RampHold** and **RampInZero** have to be set to zero [see *MainCtrlWord (7.01)*]. When **Jog1** command is given the drive goes into state **Running** and turns with speed set in *FixedSpeed1 (23.02)* alternatively **Inching1** [see *MainCtrlWord (7.01)*] can be used. **Note:** Acceleration and deceleration time for jogging is selected by *JogAccTime (22.12)* and *JogDecTime (22.13).* **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.18	**Jog2 (jogging 2 command)** Binary signal for **Jog2**. Selects speed reference set in *FixedSpeed2 (23.03):* Selection see *Jog1 (10.17).* **Note:** *Jog2 (10.18)* overrides *Jog1 (10.17)* **Note:** *CommandSel (10.01*) = **Local I/O**: – The drive has to be in state **RdyRun** (**RdyRef** is still zero). When **Jog2** command is given the drives sets automatically **RampOutZero** = **RampHold** = **RampInZero** = 0 [see *MainCtrlWord (7.01)*] and goes into state **Running** and turns with speed set in *FixedSpeed2 (23.03).* *CommandSel (10.01)* = **MainCtrlWord**: – The drive has to be in state **RdyRun** (**RdyRef** is still zero). **RampOutZero**, **RampHold** and **RampInZero** have to be set to zero [see *MainCtrlWord (7.01)*]. When **Jog2** command is given the drive goes into state **Running** and turns with speed set in *FixedSpeed2 (23.03)* alternatively **Inching2** [see *MainCtrlWord (7.01)*] can be used. **Note:** Acceleration and deceleration time for jogging is selected by *JogAccTime (22.12)* and *JogDecTime (22.13).* **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	C
10.19	**Unused**					
10.20	**ConvFanAck (converter fan acknowledge)** The drive trips with **F527 ConvFanAck** [*FaultWord2 (9.02)* bit 10] if a digital input for the converter fan is selected and the acknowledge is missing for 10 seconds. As soon as the acknowledge is missing **A104 ConvOverTemp** [*AlarmWord1 (9.06)* bit 3] is set. The alarm is reset automatically if the converter fan acknowledge is coming back before the 10 seconds are elapsed: 0 = **NotUsed** no reaction 1 = **DI1** 1= **acknowledge** OK, 0 = no **acknowledge**, default 2 = **DI2** 1= **acknowledge** OK, 0 = no **acknowledge** 3 = **DI3** 1= **acknowledge** OK, 0 = no **acknowledge** 4 = **DI4** 1= **acknowledge** OK, 0 = no **acknowledge** 5 = **DI5** 1= **acknowledge** OK, 0 = no **acknowledge** 6 = **DI6** 1= **acknowledge** OK, 0 = no **acknowledge** 7 = **DI7** 1= **acknowledge** OK, 0 = no **acknowledge** 8 = **DI8** 1= **acknowledge** OK, 0 = no **acknowledge** 9 = **DI9** 1= **acknowledge** OK, 0 = no **acknowledge,** only available with digital extension board 10 = **DI10** 1= **acknowledge** OK, 0 = no **acknowledge,** only available with digital extension board 11 = **DI11** 1= **acknowledge** OK, 0 = no **acknowledge,** only available with digital extension board **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI11	DI1	-	C
10.21	**MainContAck (main contactor acknowledge)** The drive trips with **F524 MainContAck** [*FaultWord2 (9.02)* bit 7] if a digital input for the main contactor is selected and the acknowledge is missing for 10 seconds: Selection see *ConvFanAck (10.20).* **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI11	DI3	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.22	**DynBrakeAck (dynamic braking acknowledge)** The drive sets **A105 DynBrakeAck** [*AlarmWord1 (9.06)* bit 4] if a digital input for dynamic braking is selected and the acknowledge (dynamic braking active) is still present when **On** [*UsedMCW (7.04)* bit 3] is set: Selection see *ConvFanAck (10.20).* **A105 DynBrakeAck** [*AlarmWord1 (9.06)* bit 4] should prevent the drive to be started while dynamic braking is active. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI11	NotUsed	-	C
10.23	**DC BreakAck (DC breaker acknowledge)** The drive sets **A103 DC BreakAck** [*AlarmWord1 (9.06)* bit 2] if a digital input for the DC-breaker is selected and the acknowledge is missing: Selection see *ConvFanAck (10.20).* The motor will coast if **A103 DC BreakAck** [*AlarmWord1 (9.06)* bit 2] is set. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI11	NotUsed	-	E
10.24	**Unused**					
10.25	**DI1Invert (invert digital input 1)** Inversion selection for digital input 1: 0 = **Direct** 1 = **Inverted** **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	C
10.26	**DI2Invert (invert digital input 2)** Inversion selection for digital input 2: 0 = **Direct** 1 = **Inverted** **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	C
10.27	**DI3Invert (invert digital input 3)** Inversion selection for digital input 3: 0 = **Direct** 1 = **Inverted** **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	C
10.28	**DI4Invert (invert digital input 4)** Inversion selection for digital input 4: 0 = **Direct** 1 = **Inverted** **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	C
10.29	**DI5Invert (invert digital input 5)** Inversion selection for digital input 5: 0 = **Direct** 1 = **Inverted** **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	C
10.30	**DI6Invert (invert digital input 6)** Inversion selection for digital input 6: 0 = **Direct** 1 = **Inverted** **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	C
10.31	**DI7Invert (invert digital input 7)** Inversion selection for digital input 7: 0 = **Direct** 1 = **Inverted** **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
10.32	**DI8Invert (invert digital input 8)** Inversion selection for digital input 8: 0 = **Direct** 1 = **Inverted** **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	C
10.33	**DI9Invert (invert digital input 9)** Inversion selection for digital input 9: 0 = **Direct** only available with digital extension board 1 = **Inverted** only available with digital extension board **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	E
10.34	**DI10Invert (invert digital input 10)** Inversion selection for digital input 10: 0 = **Direct** only available with digital extension board 1 = **Inverted** only available with digital extension board **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	E
10.35	**DI11Invert (invert digital input 11)** Inversion selection for digital input 11: 0 = **Direct** only available with digital extension board 1 = **Inverted** only available with digital extension board **Int. Scaling: 1 == 1 Type: C Volatile: N**	Direct	Inverted	Direct	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 11	**Speed reference inputs**					
11.01	**Unused**					
11.02	**Ref1Mux (speed reference 1 selector/multiplexer)** Speed reference 1 selector: 0 = **Open** switch for speed ref. 1 is fixed open 1 = **Close** switch for speed ref 1 is fixed closed, default 2 = **DI1** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0 3 = **DI2** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0 4 = **DI3** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0 5 = **DI4** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0 6 = **DI5** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0 7 = **DI6** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0 8 = **DI7** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0 9 = **DI8** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0 10 = **DI9** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; only available with digital extension board 11= **DI10** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; only available with digital extension board 12 = **DI11** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; only available with digital extension board 13 = **MCW Bit11** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 11 14 = **MCW Bit12** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 12 15 = **MCW Bit13** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 13 16 = **MCW Bit14** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 14 17 = **MCW Bit15** 1= switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 15 18 = **ACW Bit12** 1 = switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; *AuxCtrlWord (7.02)* bit 12 19 = **ACW Bit13** 1 = switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; *AuxCtrlWord (7.02)* bit 13 20 = **ACW Bit14** 1 = switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; *AuxCtrlWord (7.02)* bit 14 21 = **ACW Bit15** 1 = switch is **closed**, speed ref 1 is active; 0 = switch is **open**, speed ref = 0; *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	Open	ACW Bit15	Close	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
11.03	**Ref1Sel (speed reference 1 select)** Speed reference 1 value: 0 = **SpeedRef2301** *SpeedRef (23.01),* default 1 = **AuxSpeedRef** *AuxSpeedRef (23.13)* 2 = **AI1** analog input AI1 3 = **AI2** analog input AI2 4 = **AI3** analog input AI3 5 = **AI4** analog input AI4 6 = **AI5** analog input AI5 7 = **AI6** analog input AI6 8 = **FixedSpeed1** *FixedSpeed1 (23.02)* 9 = **FixedSpeed2** *FixedSpeed2 (23.03)* 10 = **MotPot** motor pot controlled by *MotPotUp (11.13), MotPotDown (11.14)* and *MotPotMin (11.15)* 11 = **AuxRef-AI1** *AuxSpeedRef (23.13)* minus value of AI1 12 = reserved reserved 13 = **MinAI2AI4** minimum of AI2 and AI4 14 = **MaxAI2AI4** maximum of AI2 and AI4 15 = **AI1Direct+** Fast speed reference input using analog input AI1. *SpeedRefExt1 (2.30)* is written directly onto the speed error summation. Thus the speed ramp is bypassed. The signal is forced to zero if **RampOutZero** = 0 or **RampInZero** = 0 [see *MainCtrlWord (70.1)*]. 16 = **AI2Direct+** Fast speed reference input using analog input AI2. *SpeedRefExt1 (2.30)* is written directly onto the speed error summation point. Thus the speed ramp is bypassed. The signal is forced to zero if **RampOutZero** = 0 or **RampInZero** = 0 [see *MainCtrlWord (70.1)*]. 17 = **Enc2Direct+** Fast speed reference input using pulse encoder 2. *SpeedRefExt1 (2.30)* is written directly onto the speed error summation point. Thus the speed ramp is bypassed. The signal is forced to zero if **RampOutZero** = 0 or **RampInZero** = 0 [see *MainCtrlWord (70.1)*]. 18 = **SpeedRef2315** Fast speed reference input using *DirectSpeedRef (23.15)*. *SpeedRefExt1 (2.30)* is written directly onto the speed error summation point. Thus the speed ramp is bypassed. The signal is forced to zero if **RampOutZero** = 0 or **RampInZero** = 0 [see *MainCtrlWord (70.1)*]. **Int. Scaling: 1 == 1 Type: C Volatile: N**	SpeedRef2301	Enc2Direct+	SpeedRef2301	-	C
11.04	**Unused**					
11.05	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
11.06	**Ref2Sel (speed reference 2 select)** Speed reference 2 value: 0 = **SpeedRef2301** *SpeedRef (23.01),* default 1 = **AuxSpeedRef** *AuxSpeedRef (23.13)* 2 = **AI1** analog input AI1 3 = **AI2** analog input AI2 4 = **AI3** analog input AI3 5 = **AI4** analog input AI4 6 = **AI5** analog input AI5 7 = **AI6** analog input AI6 8 = **FixedSpeed1** *FixedSpeed1 (23.02)* 9 = **FixedSpeed2** *FixedSpeed2 (23.03)* 10 = **MotPot** motor pot controlled by *MotPotUp (11.13), MotPotDown (11.14)* and *MotPotMin (11.15)* 11 = **AI2-AI3** AI2 minus AI3 12 = **AI2+AI3** AI2 plus AI3 13 = **AI1*AI2** AI1 multiplied with AI2 14 = **AI2*AI3** AI2 multiplied with AI3 15 = **MinAI2AI4** minimum of AI2 and AI4 16 = **MaxAI2AI4** maximum of AI2 and AI4 17 = **Encoder2** pulse encoder 2 **Int. Scaling: 1 == 1 Type: C Volatile: N**	SpeedRef2301	Encoder2	SpeedRef2301	-	E
11.07	**Unused**					
11.08	**Unused**					
11.09	**Unused**					
11.10	**Unused**					
11.11	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
11.12	**Ref2Mux (speed reference 2 selector/multiplexer)** Speed reference 2 selector: 0 = **Invert1102** Invert speed ref. 1 selection; implements a change over switch together with speed ref 2 selection. E.g. if speed ref. 1 selection switch is open the switch for speed ref. 2 is closed and vice versa. 1 = **Open** switch for speed ref. 2 is fixed open, default 2 = **Close** switch for speed ref 2 is fixed closed 3 = **DI1** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0 4 = **DI2** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0 5 = **DI3** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0 6 = **DI4** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0 7 = **DI5** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0 8 = **DI6** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0 9 = **DI7** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0 10 = **DI8** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0 11 = **DI9** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; only available with digital extension board 12= **DI10** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; only available with digital extension board 13 = **DI11** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; only available with digital extension board 14 = **MCW Bit11** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 11 15 = **MCW Bit12** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 12 16 = **MCW Bit13** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 13 17 = **MCW Bit14** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 14 18 = **MCW Bit15** 1= switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; *MainCtrlWord (7.01)* bit 15 19 = **ACW Bit12** 1 = switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; *AuxCtrlWord (7.02)* bit 12 20 = **ACW Bit13** 1 = switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; *AuxCtrlWord (7.02)* bit 13 21 = **ACW Bit14** 1 = switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; *AuxCtrlWord (7.02)* bit 14 22 = **ACW Bit15** 1 = switch is **closed**, speed ref 2 is active; 0 = switch is **open**, speed ref = 0; *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	Invert1102	ACW Bit15	Open	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
11.13	**MotPotUp (motor pot up)** With the motor pot up function the motor speed is increased by means of the selected binary input. The acceleration is limited by *AccTime1 (22.01)*. *MotPotDown (11.14)* overrides *MotPotUp (11.13)*: 0 = **NotUsed** default 1 = **DI1** 1= **increase** speed, 0 = **hold** speed 2 = **DI2** 1= **increase** speed, 0 = **hold** speed 3 = **DI3** 1= **increase** speed, 0 = **hold** speed 4 = **DI4** 1= **increase** speed, 0 = **hold** speed 5 = **DI5** 1= **increase** speed, 0 = **hold** speed 6 = **DI6** 1= **increase** speed, 0 = **hold** speed 7 = **DI7** 1= **increase** speed, 0 = **hold** speed 8 = **DI8** 1= **increase** speed, 0 = **hold** speed 9 = **DI9** 1= **increase** speed, 0 = **hold** speed**,** only available with digital extension board 10 = **DI10** 1= **increase** speed, 0 = **hold** speed**,** only available with digital extension board 11 = **DI11** 1= **increase** speed, 0 = **hold** speed**,** only available with digital extension board 12 = **MCW Bit11** 1= **increase** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** 1= **increase** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** 1= **increase** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** 1= **increase** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** 1= **increase** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** 1= **increase** speed, 0 = **hold** speed**,** *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** 1= **increase** speed, 0 = **hold** speed**,** *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** 1= **increase** speed, 0 = **hold** speed**,** *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** 1= **increase** speed, 0 = **hold** speed**,** *AuxCtrlWord (7.02)* bit 15 **Note:** The speed reference is selected by means of *Ref1Sel (11.03)* = **MotPot** respectively *Ref2Sel (11.06)* = **MotPot**. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
11.14	**MotPotDown (motor pot down)** With the motor pot down function the motor speed is decreased by means of the selected binary input. The deceleration is limited by *DecTime1 (22.02)* until zero speed respectively *MotPotMin (11.15)* is reached. *MotPotDown (11.14)* overrides *MotPotUp (11.13)*: 0 = **NotUsed** default 1 = **DI1** 1= **decrease** speed, 0 = **hold** speed 2 = **DI2** 1= **decrease** speed, 0 = **hold** speed 3 = **DI3** 1= **decrease** speed, 0 = **hold** speed 4 = **DI4** 1= **decrease** speed, 0 = **hold** speed 5 = **DI5** 1= **decrease** speed, 0 = **hold** speed 6 = **DI6** 1= **decrease** speed, 0 = **hold** speed 7 = **DI7** 1= **decrease** speed, 0 = **hold** speed 8 = **DI8** 1= **decrease** speed, 0 = **hold** speed 9 = **DI9** 1= **decrease** speed, 0 = **hold** speed**,** only available with digital extension board 10 = **DI10** 1= **decrease** speed, 0 = **hold** speed**,** only available with digital extension board 11 = **DI11** 1= **decrease** speed, 0 = **hold** speed**,** only available with digital extension board 12 = **MCW Bit11** 1= **decrease** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** 1= **decrease** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** 1= **decrease** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** 1= **decrease** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** 1= **decrease** speed, 0 = **hold** speed**,** *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** 1= **decrease** speed, 0 = **hold** speed**,** *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** 1= **decrease** speed, 0 = **hold** speed**,** *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** 1= **decrease** speed, 0 = **hold** speed**,** *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** 1= **decrease** speed, 0 = **hold** speed**,** *AuxCtrlWord (7.02)* bit 15 **Note:** The speed reference is selected by means of *Ref1Sel (11.03)* = **MotPot** respectively *Ref2Sel (11.06)* = **MotPot**. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
11.15	**MotPotMin (motor pot minimum)** The motor pot minimum function releases the minimum speed level. The minimum speed level is defined by *FixedSpeed1 (23.02)*. When the drive is started the motor accelerates to *FixedSpeed1 (23.02).* It is not possible to set the speed below *FixedSpeed1 (23.02)* by means of the motor pot function: 0 = **NotUsed** default 1 = **DI1** 1= **released**, 0 = **blocked** 2 = **DI2** 1= **released**, 0 = **blocked** 3 = **DI3** 1= **released**, 0 = **blocked** 4 = **DI4** 1= **released**, 0 = **blocked** 5 = **DI5** 1= **released**, 0 = **blocked** 6 = **DI6** 1= **released**, 0 = **blocked** 7 = **DI7** 1= **released**, 0 = **blocked** 8 = **DI8** 1= **released**, 0 = **blocked** 9 = **DI9** 1= **released**, 0 = **blocked,** only available with digital extension board 10 = **DI10** 1= **released**, 0 = **blocked,** only available with digital extension board 11 = **DI11** 1= **released**, 0 = **blocked,** only available with digital extension board 12 = **MCW Bit11** 1= **released**, 0 = **blocked,** *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** 1= **released**, 0 = **blocked,** *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** 1= **released**, 0 = **blocked,** *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** 1= **released**, 0 = **blocked,** *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** 1= **released**, 0 = **blocked,** *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** 1= **released**, 0 = **blocked,** *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** 1= **released**, 0 = **blocked,** *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** 1= **released**, 0 = **blocked,** *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** 1= **released**, 0 = **blocked,** *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 12	**Constant speeds**					
12.01	**unused**					
12.02	**ConstSpeed1 (constant speed 1)** Defines constant speed 1 in rpm. The constant speed can be connected by Adaptive Program or application program. Internally limited from: $-(2.29) * \frac{32767}{20000} rpm \ \ to \ \ (2.29) * \frac{32767}{20000} rpm$ **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	0	rpm	E
12.03	**ConstSpeed2 (constant speed 2)** Defines constant speed 2 in rpm. The constant speed can be connected by Adaptive Program or application program. Internally limited from: $-(2.29) * \frac{32767}{20000} rpm \ \ to \ \ (2.29) * \frac{32767}{20000} rpm$ **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	0	rpm	E
12.04	**ConstSpeed3 (constant speed 3)** Defines constant speed 3 in rpm. The constant speed can be connected by Adaptive Program or application program. Internally limited from: $-(2.29) * \frac{32767}{20000} rpm \ \ to \ \ (2.29) * \frac{32767}{20000} rpm$ **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	0	rpm	E
12.05	**ConstSpeed4 (constant speed 4)** Defines constant speed 4 in rpm. The constant speed can be connected by Adaptive Program or application program. Internally limited from: $-(2.29) * \frac{32767}{20000} rpm \ \ to \ \ (2.29) * \frac{32767}{20000} rpm$ **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	0	rpm	E

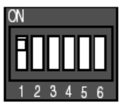
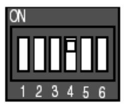
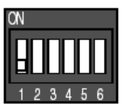
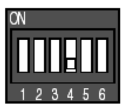
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 13	**Analog inputs**					
13.01	**AI1HighVal (analog input 1 high value)** +100 % of the input signal connected to analog input 1 is scaled to the voltage in *AI1HighVal (13.01)*. Example: – In case the min. / max. voltage (±10 V) of analog input 1 should equal ±250 % of *TorqRefExt (2.24)*, set: *TorqRefA Sel (25.10)* = **AI1** *ConvModeAI1 (13.03)* = ±**10 V Bi**, *AI1HighVal (13.01)* = 4000 mV and *AI1LowVal (13.02)* = -4000 mV **Note:** To use current please set the jumper (SDCS-CON-4 or SDCS-IOB-3) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: I Volatile: N**	-10000	10000	10000	mV	C
13.02	**AI1LowVal (analog input 1 low value)** -100 % of the input signal connected to analog input 1 is scaled to the voltage in *AI1LowVal (13.02)*. **Note:** *AI1LowVal (13.02)* is only valid if *ConvModeAI1 (13.03)* = ±**10 V Bi**. **Note:** To use current please set the jumper (SDCS-CON-4 or SDCS-IOB-3) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: SI Volatile: N**	-10000	10000	-10000	mV	C
13.03	**ConvModeAI1 (conversion mode analog input 1)** The distinction between voltage and current is done via jumpers on the SDCS-CON-4 or SDCS-IOB-3 board: 0 = ±**10V Bi** -10 V to 10 V / -20 mA to 20 mA bipolar input, default 1 = **0V-10V Uni** 0 V to 10 V / 0 mA to 20 mA unipolar input 2 = **2V-10V Uni** 2 V to 10 V / 4 mA to 20 mA unipolar input 3 = **5V Offset** 5 V / 10 mA offset in the range 0 V to 10 V / 0 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) 4 = **6V Offset** 6 V / 12 mA offset in the range 2 V to 10 V / 4 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) **Int. Scaling: 1 == 1 Type: C Volatile: N**	±10V Bi	6V Offset	±10V Bi	-	C
13.04	**FilterAI1 (filter time analog input 1)** Analog input 1 filter time. The hardware filter time is ≤ 2ms. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	C
13.05	**AI2HighVal (analog input 2 high value)** +100 % of the input signal connected to analog input 2 is scaled to the voltage in *AI2HighVal (13.05)*. **Note:** To use current please set the jumper (SDCS-CON-4 or SDCS-IOB-3) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: I Volatile: N**	-10000	10000	10000	mV	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
13.06	**AI2LowVal (analog input 2 low value)** -100 % of the input signal connected to analog input 2 is scaled to the voltage in *AI2LowVal (13.06).* **Note:** *AI2LowVal (13.06)* is only valid if *ConvModeAI2 (13.07)* = ±**10V Bi**. **Note:** To use current please set the jumper (SDCS-CON-4 or SDCS-IOB-3) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: SI Volatile: N**	-10000	10000	-10000	mV	C
13.07	**ConvModeAI2 (conversion mode analog input 2)** The distinction between voltage and current is done via jumpers on the SDCS-CON-4 or SDCS-IOB-3 board: 0 = ±**10V Bi** -10 V to 10 V / -20 mA to 20 mA bipolar input, default 1 = **0V-10V Uni** 0 V to 10 V / 0 mA to 20 mA unipolar input 2 = **2V-10V Uni** 2 V to 10 V / 4 mA to 20 mA unipolar input 3 = **5V Offset** 5 V / 10 mA offset in the range 0 V to 10 V / 0 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) 4 = **6V Offset** 6 V / 12 mA offset in the range 2 V to 10 V / 4 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) **Int. Scaling: 1 == 1 Type: C Volatile: N**	±10V Bi	6V Offset	±10V Bi	-	C
13.08	**FilterAI2 (filter time analog input 2)** Analog input 2 filter time. The hardware filter time is ≤ 2ms. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	C
13.09	**AI3HighVal (analog input 3 high value)** +100 % of the input signal connected to analog input 3 is scaled to the voltage in *AI3HighVal (13.09).* **Note:** To use current please set the jumper (SDCS-IOB-3) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: I Volatile: N**	-10000	10000	10000	mV	E
13.10	**AI3LowVal (analog input 3 low value)** -100 % of the input signal connected to analog input 3 is scaled to the voltage in *AI3LowVal (13.10).* **Note:** *AI3LowVal (13.10)* is only valid if *ConvModeAI3 (13.11)* = ±**10V Bi**. **Note:** To use current please set the jumper (SDCS-IOB-3) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: SI Volatile: N**	-10000	10000	-10000	mV	E
13.11	**ConvModeAI3 (conversion mode analog input 3)** Analog input 3 on the SDCS-CON-4 is only working with voltage. The distinction between voltage and current is done via jumpers on the SDCS-IOB-3 board: 0 = ±**10V Bi** -10 V to 10 V / -20 mA to 20 mA bipolar input, default 1 = **0V-10V Uni** 0 V to 10 V / 0 mA to 20 mA unipolar input 2 = **2V-10V Uni** 2 V to 10 V / 4 mA to 20 mA unipolar input 3 = **5V Offset** 5 V / 10 mA offset in the range 0 V to 10 V / 0 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) 4 = **6V Offset** 6 V / 12 mA offset in the range 2 V to 10 V / 4 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) **Int. Scaling: 1 == 1 Type: C Volatile: N**	±10V Bi	6V Offset	±10V Bi	-	E
13.12	**FilterAI3 (filter time analog input 3)** Analog input 3 filter time. The hardware filter time is ≤ 2 ms. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	E

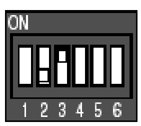
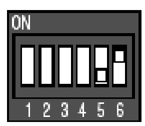
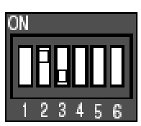
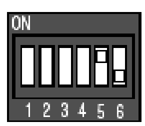
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
13.13	**AI4HighVal (analog input 4 high value)** +100 % of the input signal connected to analog input 4 is scaled to the voltage in *AI4HighVal (13.13).* **Note:** To use current please set the jumper (SDCS-IOB-3) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: I Volatile: N**	-10000	10000	10000	mV	E
13.14	**AI4LowVal (analog input 4 low value)** -100 % of the input signal connected to analog input 4 is scaled to the voltage in *AI4LowVal (13.14).* **Note:** *AI3LowVal (13.14)* is only valid if *ConvModeAI4 (13.15)* = ±**10V Bi**. **Note:** To use current please set the jumper (SDCS-IOB-3) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: SI Volatile: N**	-10000	10000	-10000	mV	E
13.15	**ConvModeAI4 (conversion mode analog input 4)** Analog input 4 on the SDCS-CON-4 is only working with voltage. The distinction between voltage and current is done via jumpers on the SDCS-IOB-3 board: 0 = ±**10V Bi** -10 V to 10 V / -20 mA to 20 mA bipolar input, default 1 = **0V-10V Uni** 0 V to 10 V / 0 mA to 20 mA unipolar input 2 = **2V-10V Uni** 2 V to 10 V / 4 mA to 20 mA unipolar input 3 = **5V Offset** 5 V / 10 mA offset in the range 0 V to 10 V / 0 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) 4 = **6V Offset** 6 V / 12 mA offset in the range 2 V to 10 V / 4 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) **Int. Scaling: 1 == 1 Type: C Volatile: N**	±10V Bi	6V Offset	±10V Bi	-	E
13.16	**FilterAI4 (filter time analog input 4)** Analog input 4 filter time. The hardware filter time is ≤ 2 ms. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	E
13.17	**Reserved**					
13.18	**Reserved**					
13.19	**Reserved**					
13.20	**Unused**					
13.21	**AI5HighVal (analog input 5 high value)** +100 % of the input signal connected to analog input 5 is scaled to the voltage in *AI5HighVal (13.21).* **Note:** To use current please set the DIP-switches (RAIO-01) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: I Volatile: N**	-10000	10000	10000	mV	E
13.22	**AI5LowVal (analog input 5 low value)** -100 % of the input signal connected to analog input 5 is scaled to the voltage in *AIO5LowVal (13.22).* **Note:** *AI5LowVal (13.22)* is only valid if *ConvModeAI5 (13.23)* = ±**10V Bi**. **Note:** To use current please set the DIP-switches (RAIO-01) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: SI Volatile: N**	-10000	10000	-10000	mV	E

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>13.23</td><td>

ConvModeAI5 (conversion mode analog input 5)
The distinction between bipolar and unipolar respectively voltage and current is done via DIP-switches on the RAIO-01 board:
0 = **±10V Bi** -10 V to 10 V / -20 mA to 20 mA bipolar input, default
1 = **0V-10V Uni** 0 V to 10 V / 0 mA to 20 mA unipolar input
2 = **2V-10V Uni** 2 V to 10 V / 4 mA to 20 mA unipolar input
3 = **5V Offset** 5 V / 10 mA offset in the range 0 V to 10 V / 0 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.)
4 = **6V Offset** 6 V / 12 mA offset in the range 2 V to 10 V / 4 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.)
Bipolar and unipolar:

<table>
<tr><th colspan="2">DIP switch setting</th><th rowspan="2">Input signal type</th></tr>
<tr><th>Analogue input AI1</th><th>Analogue input AI2</th></tr>
<tr><td>
</td><td>
</td><td>±0(4)…20 mA
±0(2)…10 V
±0…2 V</td></tr>
<tr><td>
</td><td>
</td><td>0(4)…20 mA
0(2)…10 V
0…2 V
(Default)</td></tr>
</table>

Voltage and current:

<table>
<tr><th rowspan="2">Input signal type</th><th colspan="2">DIP switch settings</th></tr>
<tr><th>Analogue input 1</th><th>Analogue input 2</th></tr>
<tr><td>Current signal
±0(4)…20 mA
(Default)</td><td>
</td><td>
</td></tr>
<tr><td>Voltage signal
±0(2)…10 V</td><td>
</td><td>
</td></tr>
</table>

Int. Scaling: 1 == 1 Type: C Volatile: N
</td><td>±10V Bi</td><td>6V Offset</td><td>±10V Bi</td><td>-</td><td>E</td></tr>
<tr><td>13.24</td><td>**Unused**</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>13.25</td><td>

AI6HighVal (analog input 6 high value)
+100 % of the input signal connected to analog input 6 is scaled to the voltage in *AI6HighVal (13.25).*
Note:
To use current please set the DIP-switches (RAIO-01) accordingly and calculate 20 mA to 10 V.
Int. Scaling: 1 == 1 mV Type: I Volatile: N
</td><td>-10000</td><td>10000</td><td>10000</td><td>mV</td><td>E</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
13.26	**AI6LowVal (analog input 6 low value)** -100 % of the input signal connected to analog input 6 is scaled to the voltage in *AIO6LowVal (13.26).* **Note:** *AI6LowVal (13.26)* is only valid if *ConvModeAI6 (13.27)* = ±**10V Bi**. **Note:** To use current please set the DIP-switches (RAIO-01) accordingly and calculate 20 mA to 10 V. **Int. Scaling: 1 == 1 mV Type: SI Volatile: N**	-10000	10000	-10000	mV	E
13.27	**ConvModeAI6 (conversion mode analog input 6)** The distinction between bipolar and unipolar respectively voltage and current is done via DIP-switches on the RAIO-01 board: 0 = ±**10V Bi** -10 V to 10 V / -20 mA to 20 mA bipolar input, default 1 = **0V-10V Uni** 0 V to 10 V / 0 mA to 20 mA unipolar input 2 = **2V-10V Uni** 2 V to 10 V / 4 mA to 20 mA unipolar input 3 = **5V Offset** 5 V / 10 mA offset in the range 0 V to 10 V / 0 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) 4 = **6V Offset** 6 V / 12 mA offset in the range 2 V to 10 V / 4 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) **Int. Scaling: 1 == 1 Type: C Volatile: N**	+10V Bi	6V Offset	+10V Bi	-	E
Group 14	**Digital outputs**					
14.01	**DO1Index (digital output 1 index)** Digital output 1 is controlled by a selectable bit - see *DO1BitNo (14.02)* - of the source (signal/parameter) selected with this parameter. The format is **-xxyy**, with: **-** = invert digital output, **xx** = group and **yy** = index. Examples: – If *DO1Index (14.01)* = 801 (main status word) and *DO1BitNo (14.02)* = 1 (**RdyRun**) digital output 1 is high when the drive is **RdyRun**. – If *DO1Index (14.01)* = -801 (main status word) and *DO1BitNo (14.02)* = 3 (**Tripped**) digital output 1 is high when the drive is not faulty. Digital output 1 default setting is: command **FansOn** *CurCtrlStat1 (6.03)* bit 0. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	603	-	C
14.02	**DO1BitNo (digital output 1 bit number)** Bit number of the signal/parameter selected with *DO1Index (14.02*). **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	0	-	C
14.03	**DO2Index (digital output 2 index)** Digital output 2 is controlled by a selectable bit - see *DO2BitNo (14.04)* - of the source (signal/parameter) selected with this parameter. The format is **-xxyy**, with: **-** = invert digital output, **xx** = group and **yy** = index. Digital output 2 default setting is: command **FieldOn** *CurCtrlStat1 (6.03)* bit 5. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	603	-	C
14.04	**DO2BitNo (digital output 2 bit number)** Bit number of the signal/parameter selected with *DO2Index (14.03*). **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	5	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
14.05	**DO3Index (digital output 3 index)** Digital output 3 is controlled by a selectable bit - see *DO3BitNo (14.06)* - of the source (signal/parameter) selected with this parameter. The format is **-xxyy**, with: **-** = invert digital output, **xx** = group and **yy** = index. Digital output 3 default setting is: command **MainContactorOn** *CurCtrlStat1 (6.03)* bit 7. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	603	-	C
14.06	**DO3BitNo (digital output 3 bit number)** Bit number of the signal/parameter selected with *DO3Index (14.05*). **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	7	-	C
14.07	**DO4Index (digital output 4 index)** Digital output 4 is controlled by a selectable bit - see *DO4BitNo (14.08)* - of the source (signal/parameter) selected with this parameter. The format is **-xxyy**, with: **-** = invert digital output, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	C
14.08	**DO4BitNo (digital output 4 bit number)** Bit number of the signal/parameter selected with *DO4Index (14.07*). **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	0	-	C
14.09	**DO5Index (digital output 5 index)** Digital output 5 is controlled by a selectable bit - see *DO5BitNo (14.10)* - of the source (signal/parameter) selected with this parameter. The format is **-xxyy**, with: **-** = invert digital output, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	C
14.10	**DO5BitNo (digital output 5 bit number)** Bit number of the signal/parameter selected with *DO5Index (14.09*). **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	0	-	C
14.11	**DO6Index (digital output 6 index)** Digital output 6 is controlled by a selectable bit - see *DO6BitNo (14.12)* - of the source (signal/parameter) selected with this parameter. The format is **-xxyy**, with: **-** = invert digital output, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	C
14.12	**DO6BitNo (digital output 6 bit number)** Bit number of the signal/parameter selected with *DO6Index (14.11*). **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	0	-	C
14.13	**DO7Index (digital output 7 index)** Digital output 7 is controlled by a selectable bit - see *DO7BitNo (14.14)* - of the source (signal/parameter) selected with this parameter. The format is **-xxyy**, with: **-** = invert digital output, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	C
14.14	**DO7BitNo (digital output 7 bit number)** Bit number of the signal/parameter selected with *DO7Index (14.13*). **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	0	-	C
14.15	**DO8Index (digital output 8 index)** Digital output 8 is controlled by a selectable bit - see *DO8BitNo (14.16)* - of the source (signal/parameter) selected with this parameter. The format is **-xxyy**, with: **-** = invert digital output, **xx** = group and **yy** = index. Digital output 8 default setting is: command **MainContactorOn** *CurCtrlStat1 (6.03)* bit 7 **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	603	-	C
14.16	**DO8BitNo (digital output 8 bit number)** Bit number of the signal/parameter selected with *DO8Index (14.15*). **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	7	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 15	**Analog outputs**					
15.01	**IndexAO1 (analog output 1 index)** Analog output 1 is controlled by a source (signal/parameter) selected with *IndexAO1 (15.01).* The format is **-xxyy**, with: **-** = negate analog output, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	C
15.02	**CtrlWordAO1 (control word analog output 1)** Analog output 1 can be written to via *CtrlWordAO1 (15.02)* using Adaptive Program, application program or overriding control if *IndexAO1 (15.01)* is set to zero. Further description see *group 19 Data Storage*. **Int. Scaling: 1 == 1 Type: SI Volatile: Y**	-32768	32767	0	-	C
15.03	**ConvModeAO1 (convert mode analog output 1)** Analog output 1 signal offset: 0 = ±**10V Bi** -10 V to 10 V bipolar output, default 1 = **0V-10V Uni** 0 V to 10 V unipolar output 2 = **2V-10V Uni** 2 V to 10 V unipolar output 3 = **5V Offset** 5 V offset in the range 0 V to 10 V for testing or indication of bipolar signals (e.g. torque, speed, etc.) 4 = **6V Offset** 6 V offset in the range 2 V to 10 V for testing or indication of bipolar signals (e.g. torque, speed, etc.) 5 = **0V-10V Abs** absolute 0 V to 10 V unipolar output (negative values are shown positive) **Int. Scaling: 1 == 1 Type: C Volatile: N**	±10V Bi	0V-10V Abs	±10V Bi	-	C
15.04	**FilterAO1 (filter analog output 1)** Analog output 1 filter time. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	C
15.05	**ScaleAO1 (scaling analog output 1)** 100 % of the signal/parameter selected with *IndexAO1 (15.01)* is scaled to the voltage in *ScaleAO1 (15.05).* Example: – In case the min. / max. voltage (±10 V) of analog output 1 should equal ±250 % of *TorqRefUsed (2.13),* set: *IndexAO1 (15.01)* = 213, *ConvModeAO1 (15.03)* = ±**10V Bi** and *ScaleAO1 (15.05)* = 4000 mV **Int. Scaling: 1 == 1 mV Type: I Volatile: N**	0	10000	10000	mV	C
15.06	**IndexAO2 (analog output 2 index)** Analog output 2 is controlled by a source (signal/parameter) selected with *IndexAO2 (15.06).* The format is **-xxyy**, with: **-** = negate analog output, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	C
15.07	**CtrlWordAO2 (control word analog output 2)** Analog output 2 can be written to via *CtrlWordAO2 (15.07)* using Adaptive Program, application program or overriding control if *IndexAO2 (15.06)* is set to zero. Further description see *group 19 Data Storage*. **Int. Scaling: 1 == 1 Type: SI Volatile: Y**	-32768	32767	0	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
15.08	**ConvModeAO2 (convert mode analog output 2)** Analog output 2 signal offset: 0 = **±10V Bi** -10 V to 10 V bipolar output, default 1 = **0V-10V Uni** 0 V to 10 V unipolar output 2 = **2V-10V Uni** 2 V to 10 V unipolar output 3 = **5V Offset** 5 V offset in the range 0 V to 10 V for testing or indication of bipolar signals (e.g. torque, speed, etc.) 4 = **6V Offset** 6 V offset in the range 2 V to 10 V for testing or indication of bipolar signals (e.g. torque, speed, etc.) 5 = **0V-10V Abs** absolute 0 V to 10 V unipolar output (negative values are shown positive) **Int. Scaling: 1 == 1 Type: C Volatile: N**	±10V Bi	0V-10V Abs	±10V Bi	-	C
15.09	**FilterAO2 (filter analog output 2)** Analog output 2 filter time. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	C
15.10	**ScaleAO2 (scaling analog output 2)** 100 % of the signal/parameter selected with *IndexAO2 (15.06)* is scaled to the voltage in *ScaleAO2 (15.10)*. **Int. Scaling: 1 == 1 mV Type: I Volatile: N**	0	10000	10000	mV	C
15.11	**IndexAO3 (analog output 3 index)** Analog output 3 is controlled by a source (signal/parameter) selected with *IndexAO3 (15.11).* The format is **-xxyy**, with: **-** = negate analog output, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999			E
15.12	**CtrlWordAO3 (control word analog output 3)** Analog output 3 can be written to via *CtrlWordAO3 (15.12)* using Adaptive Program, application program or overriding control if *IndexAO3 (15.11)* is set to zero. Further description see *group 19 Data Storage*. **Int. Scaling: 1 == 1 Type: SI Volatile: Y**	-32768	32767	0	-	E
15.13	**ConvModeAO3 (convert mode analog output 3)** Analog output 3 signal offset: 0 = **0mA-20mA Uni** 0 mA to 20 mA unipolar output 1 = **4mA-20mA Uni** 4 mA to 20 mA unipolar output, default 2 = **10mA Offset** 10 mA offset in the range 0 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) 3 = **12mA Offset** 12 mA offset in the range 4 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) 4 = **0mA-20mA Abs** absolute 0 mA to 20 mA unipolar output (negative values are shown positive) **Int. Scaling: 1 == 1 Type: C Volatile: N**	4mA-20mA Uni	0mA-20mA Abs	4mA-20mA Uni	-	E
15.14	**FilterAO3 (filter analog output 3)** Analog output 3 filter time. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	E
15.15	**ScaleAO3 (scaling analog output 3)** 100 % of the signal/parameter selected with *IndexAO3 (15.11)* is scaled to the current in *ScaleAO3 (15.15)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	20	20	mA	E
15.16	**IndexAO4 (analog output 4 index)** Analog output 4 is controlled by a source (signal/parameter) selected with *IndexAO4 (15.16).* The format is **-xxyy**, with: **-** = negate analog output, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999			E

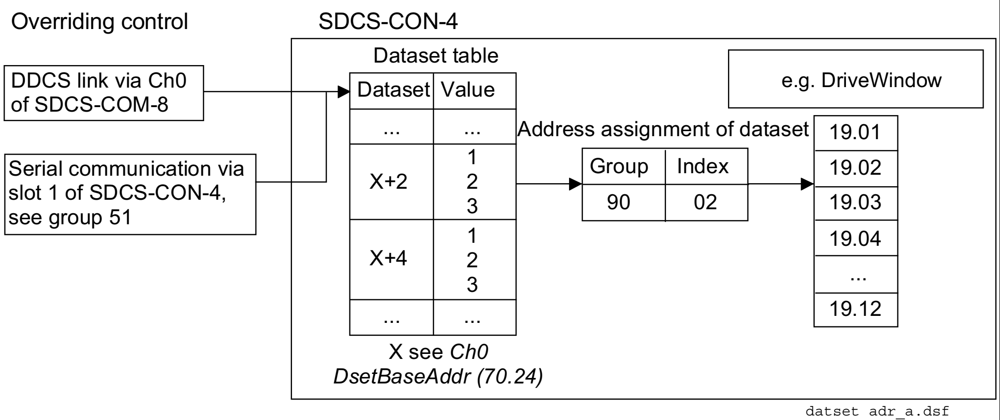
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
15.17	**CtrlWordAO4 (control word analog output 4)** Analog output 4 can be written to via *CtrlWordAO4 (15.17)* using Adaptive Program, application program or overriding control if *IndexAO4 (15.17)* is set to zero. Further description see *group 19 Data Storage*. **Int. Scaling: 1 == 1 Type: SI Volatile: Y**	-32768	32767	0	-	E
15.18	**ConvModeAO4 (convert mode analog output 4)** Analog output 4 signal offset: 0 = **0mA-20mA Uni** 0 mA to 20 mA unipolar output 1 = **4mA-20mA Uni** 4 mA to 20 mA unipolar output, default 2 = **10mA Offset** 10 mA offset in the range 0 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) 3 = **12mA Offset** 12 mA offset in the range 4 mA to 20 mA for testing or indication of bipolar signals (e.g. torque, speed, etc.) 4 = **0mA-20mA Abs** absolute 0 mA to 20 mA unipolar output (negative values are shown positive) **Int. Scaling: 1 == 1 Type: C Volatile: N**	4mA-20mA Uni	0mA-20mA Abs	4mA-20mA Uni	-	E
15.19	**FilterAO4 (filter analog output 4)** Analog output 4 filter time. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	E
15.20	**ScaleAO4 (scaling analog output 4)** 100 % of the signal/parameter selected with *IndexAO4 (15.16)* is scaled to the current in *ScaleAO4 (15.20).* **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	20	20	mA	E
Group 16	**System control inputs**					
16.01	**Unused**					
16.02	**ParLock (parameter lock)** The user can lock all parameters by means of *ParLock (16.02)* and *SysPassCode (16.03):* – To lock parameters set *SysPassCode (16.03)* to the desired value and change *ParLock (16.02)* from **Open** to **Locked**. – Unlocking of parameters is only possible if the proper pass code (the value which was present during locking) is used. To open parameters set *SysPassCode (16.03)* to the proper value and change *ParLock (16.02)* from **Locked** to **Open**. After the parameters are locked or opened the value in *SysPassCode (16.03)* is automatically changed to 0: 0 = **Open** parameter change possible, default 1 = **Locked** parameter change not possible **Int. Scaling: 1 == 1 Type: C Volatile: N**	Open	Locked	Open	-	E
16.03	**SysPassCode (system pass code)** The *SysPassCode (16.03*) is a number between 1 and 30,000 to lock all parameters by means of *ParLock (16.02)*. After using **Open** or **Locked** *SysPassCode (16.03)* is automatically set back to zero. **Attention:** Do not forget the pass code! **Int. Scaling: 1 == 1 Type: I Volatile: Y**	0	30000	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
16.04	**LocLock (local lock)** Local control can be disabled by setting *LocLock (16.04)* to **True**. If *LocLock (16.04)* is released in local control, it becomes valid after the next changeover to remote control. No pass code is required to change *LocLock (16.04):* 0 = **False** local control released, default 1 = **True** local control blocked **Int. Scaling: 1 == 1 Type: C Volatile: N**	False	True	False	-	C
16.05	**MacroChangeMode (macro change mode)** The choice to release **Motor1/2** (shared motion) or macros **User1/2** is defined by means of *MacroChangeMode (16.05):* 0 = **User1/2** change between parameter sets **User1** and **User2,** default 1 = **Motor1/2** change between **Motor1** and **Motor2**, shared motion (parameters for motor 2 see group 49) *ParChange (10.10)* selects the binary signal to release either **Motor1/User1** or **Motor2/User2.** **Int. Scaling: 1 == 1 Type: C Volatile: N**	User1/2	Motor1/2	User1/2	-	E
16.06	**ParApplSave (save/load parameters and enable/disable application programs)** If parameters are written to cyclic, e.g. from an overriding control, they are only stored in the RAM and not in the flash. By means of *ParApplSave (16.06),* all parameter values are saved from the RAM into the flash. *ParApplSave (16.06)* is also used to save/load a parameter set on/from the memory card and to enable/disable application programs: 0 = **Done** parameters are saved or all other actions are finished, default 1 = **Save** saves the actual used parameters into the flash 3 = **SaveToMemC** saves a complete parameter set - actual used parameters, **User1** and **User2** - from control board to memory card 4 = **LoadFromMemC** loads a complete parameter set - actual used parameters, **User1** and **User2** - from memory card to control board 4 = **EableAppl** enables the application program 5 = **DisableAppl** disables the application program 6 = **DeleteAppl** To un-protect **DeleteAppl** set *ServiceMode (99.06)* = **DeleteAppl**. Deletes the application and the complete parameter set - actual used parameters, **User1** and **User2** - stored on the memory card. Also all user defined parameters will be erased from the actual parameter set. Parameter sets **User1** or **User2** stored in the drive itself will not be influenced. In case an application will be loaded anew all user defined parameters are set to default. This procedure can also be used to repair a memory card. After an action (e.g. save, load, …) is finished *ParApplSave (16.06)* is changed back to **Done**. This will take max. 1 second. **Note:** Do not use the parameter save function unnecessarily **Note:** Parameters changed by DCS800 Control Panel or commissioning tools are immediately saved into the flash. **Int. Scaling: 1 == 1 Type: C Volatile: Y**	Done	DisableAppl	Done	-	E
16.07	**Unused**					
16.08	**Unused**					

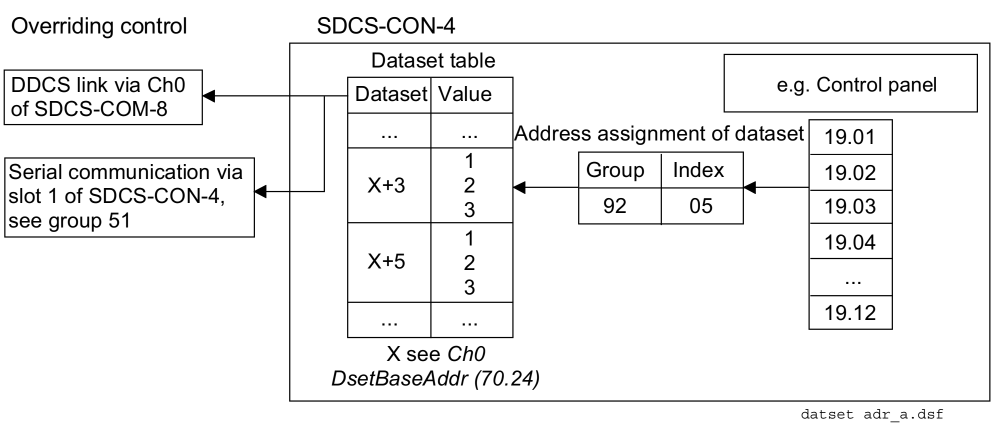
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
16.09	**USI Sel (selector for user interface)** The user interface for the DCS800 Control Panel (**Compact/Extended** parameter list) can be selected by *USI Sel (16.09):* 0 = **Compact** short parameter list (C), default 1 = **Extended** long parameter list (E) **Note:** *USI Sel (16.09)* works only for the DCS800 Control Panel. DriveWindow and DriveWindow Light always show the extended parameter list. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Compact	Extended	Compact		C
16.10	**Unused**					
16.11	**SetSystemTime (set the drive's system time)** Sets the time of the converter in minutes. The system time can be either set by means of *SetSystemTime (16.11)* or via the DCS800 Control Panel. **Int. Scaling: 1 == 1 min Type: I Volatile: Y**	0	64000	0	min	E
16.12	**Unused**					
16.13	**Unused**					
16.14	**ToolLinkConfig (tool link configuration)** The communication speed of the serial communication for the commissioning tool and the application program tool can be selected with *ToolLinkConfig (16.14)*: 0 = **9600** 9600 Baud 1 = **19200** 19200 Baud 2 = **38400** 38400 Baud, default 3 = **57600** 57600 Baud 4 = **115200** 115200 Baud If *ToolLinkConfig (16.14)* is changed its new value is taken over after the next power up. **Int. Scaling: 1 == 1 Type: C Volatile: N**	9600	115200	38400		E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 19	**Data storage**					

This parameter group consists of unused parameters for linking, testing and commissioning purposes.
Example1:
A value can be send from the overriding control to the drive via groups 90 or 91 to individual parameters in group 19. The parameters of group 19 can be read with the DCS800 Control Panel, the commissioning tools, the Adaptive Program and application program.

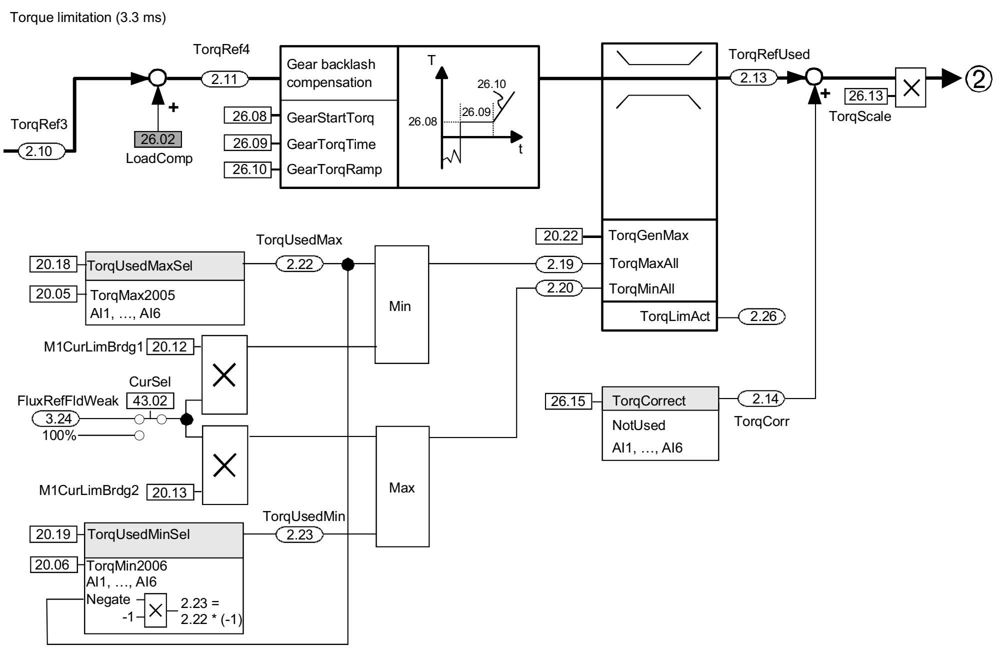


Example2:
A value can be send from the drive to the overriding control from individual parameters in group 19 via groups 92 or 93 The parameters of group 19 can be written to with the DCS800 Control Panel, the commissioning tools, the Adaptive Program and application program.



Note:
This parameter group can be used as well for reading/writing analog inputs/outputs.

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
19.01	**Data1 (data container 1)** Data container 1 (see group description above). This data container is of is of the type retain. Its value will only be saved when the drive is de-energized. Thus it will not lose its value. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.02	**Data2 (data container 2)** Data container 2 (see group description above). This data container is of is of the type retain. Its value will only be saved when the drive is de-energized. Thus it will not lose its value. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.03	**Data3 (data container 3)** Data container 3 (see group description above). This data container is of is of the type retain. Its value will only be saved when the drive is de-energized. Thus it will not lose its value. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.04	**Data4 (data container 4)** Data container 4 (see group description above). This data container is of is of the type retain. Its value will only be saved when the drive is de-energized. Thus it will not lose its value. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.05	**Data5 (data container 5)** Data container 5 (see group description above) **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.06	**Data6 (data container 6)** Data container 6 (see group description above) **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.07	**Data7 (data container 7)** Data container 7 (see group description above) **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.08	**Data8 (data container 8)** Data container 8 (see group description above) **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.09	**Data9 (data container 9)** Data container 9 (see group description above) **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.10	**Data10 (data container 10)** Data container 10 (see group description above) **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.11	**Data11 (data container 11)** Data container 11 (see group description above) **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
19.12	**Data12 (data container 12)** Data container 12 (see group description above) **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 20	**Limits**					
	This parameter group consists of all user settable limits. 					
20.01	**M1SpeedMin (motor 1 minimum speed)** Motor 1 negative speed reference limit in rpm for: – *SpeedRef2 (2.01)* – *SpeedRefUsed (2.17)* Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Note:** *M1SpeedMin (20.01)* is must be set in the range of: 0.625 to 5 times of *M1BaseSpeed (99.04).* If the scaling is out of range **A124 SpeedScale** [*AlarmWord2 (9.07)* bit 7] is generated. **Note:** *M1SpeedMin (20.01)* is also applied to *SpeedRef4 (2.18*) to avoid exceeding the speed limits by means of *SpeedCorr (23.04)*. To be able to overspeed the drive (e.g. for winder) it is possible to switch off the speed limit for *SpeedRef4 (2.18)* by means of *AuxCtrlWord (7.02)* bit 4. **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	-1500	rpm	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
20.02	**M1SpeedMax (motor 1 maximum speed)** Motor 1 positive speed reference limit in rpm for: – *SpeedRef2 (2.01)* – *SpeedRefUsed (2.17)* Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Note:** *M1SpeedMax (20.02)* is must be set in the range of: 0.625 to 5 times of *M1BaseSpeed (99.04).* If the scaling is out of range **A124 SpeedScale** [*AlarmWord2 (9.07)* bit 7] is generated. **Note:** *M1SpeedMax (20.02)* is also applied to *SpeedRef4 (2.18*) to avoid exceeding the speed limits by means of *SpeedCorr (23.04).* To be able to overspeed the drive (e.g. for winder) it is possible to switch off the speed limit for *SpeedRef4 (2.18)* by means of *AuxCtrlWord (7.02)* bit 4. **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	1500	rpm	C
20.03	**M1ZeroSpeedLim (motor 1 zero speed limit)** When the **Run** command is removed [set *UsedMCW (7.04)* bit 3 to zero], the drive will stop as chosen by *StopMode (21.03).* As soon as the actual speed reaches the limit set by *M1ZeroSpeedLim (20.03)* the motor will coast independent of the setting of *StopMode (21.03).* Existing brakes are closed (applied). While the actual speed is in the limit **ZeroSpeed** [*AuxStatWord (8.02)* bit 11] is high. **Note:** In case *FlyStart (21.10)* = **StartFrom0** and if the restart command comes before zero speed is reached **A137 SpeedNotZero** [*AlarmWord3 (9.08)* bit 4] is generated. Internally limited from: $0rpm\ \ to\ \ (2.29)rpm$ **Int. Scaling: (2.29) Type: I Volatile: N**	0	1000	75	rpm	C
20.04	**Unused**					
20.05	**TorqMax (maximum torque)** Maximum torque limit - in percent of *MotNomTorque (4.23)* - for selector *TorqUsedMaxSel (20.18).* **Note:** The used torque limit depends also on the converter's actual limitation situation (e.g. other torque limits, current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0	325	100	%	C
20.06	**TorqMin (minimum torque)** Minimum torque limit - in percent of *MotNomTorque (4.23)* - for selector *TorqUsedMinSel (20.19).* **Note:** The used torque limit depends also on the converter's actual limitation situation (e.g. other torque limits, current limits, field weakening). The limit with the largest value is valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	0	-100	%	C
20.07	**TorqMaxSPC (maximum torque speed controller)** Maximum torque limit - in percent of *MotNomTorque (4.23)* - at the output of the speed controller: – *TorqRef2 (2.09)* **Note:** The used torque limit depends also on the converter's actual limitation situation (e.g. other torque limits, current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0	325	325	%	E

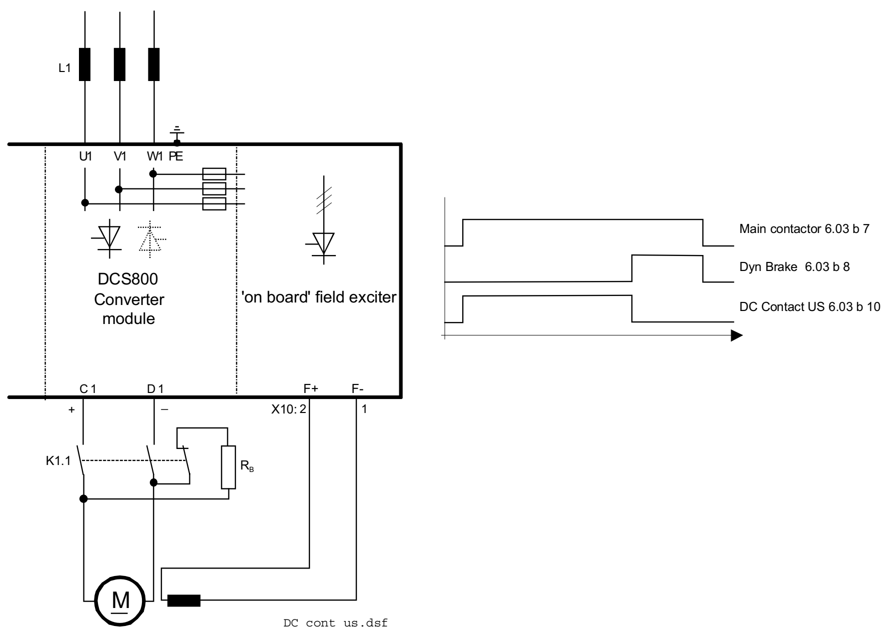
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
20.08	**TorqMinSPC (minimum torque speed controller)** Minimum torque limit - in percent of *MotNomTorque (4.23)* - at the output of the speed controller. – *TorqRef2 (2.09)* **Note:** The used torque limit depends also on the converter's actual limitation situation (e.g. other torque limits, current limits, field weakening). The limit with the largest value is valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	0	-325%	%	E
20.09	**TorqMaxTref (maximum torque of torque reference A/B)** Maximum torque limit - in percent of *MotNomTorque (4.23)* - for external references: – *TorqRefA (25.01)* – *TorqRefB (25.04)* **Note:** The used torque limit depends also on the converter's actual limitation situation (e.g. other torque limits, current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0.	325	325	%	E
20.10	**TorqMinTref (minimum torque of torque reference A/B)** Minimum torque limit - in percent of *MotNomTorque (4.23)* - for external references: – *TorqRefA (25.01)* – *TorqRefB (25.04)* **Note:** The used torque limit depends also on the converter's actual limitation situation (e.g. other torque limits, current limits, field weakening). The limit with the largest value is valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	0	-325	%	E
20.11	**Unused**					
20.12	**M1CurLimBrdg1 (motor 1 current limit of bridge 1)** Current limit bridge 1 in percent of *M1NomCur (99.03)*. Setting *M1CurLimBrdg1 (20.12)* to 0 % disables bridge 1. **Note:** The used current limit depends also on the converter's actual limitation situation (e.g. torque limits, other current limits, field weakening). The limit with the largest value is valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0	325	100	%	C
20.13	**M1CurLimBrdg2 (motor 1 current limit of bridge 2)** Current limit bridge 2 in percent of *M1NomCur (99.03).* Setting *M1CurLimBrdg2 (20.13)* to 0 % disables bridge 2. **Note:** The used current limit depends also on the converter's actual limitation situation (e.g. torque limits, other current limits, field weakening). The limit with the smallest value is valid. **Note:** *M1CurLimBrdg2 (20.13)* is internally set to 0 % if *QuadrantType (4.15)* = **2-Q** (2-Q drive). Thus do not change the default setting for 2-Q drives. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	0	-100	%	C
20.14	**ArmAlphaMax (maximum firing angle)** Maximum firing angle (α) in degrees. The maximum firing angel can be forced using *AuxCtrlWord2 (7.03)* bit 7. **Int. Scaling: 1 == 1 deg Type: SI Volatile: N**	0	165	150	deg	E
20.15	**ArmAlphaMin (minimum firing angle)** Minimum firing angle (α) in degrees. **Int. Scaling: 1 == 1 deg Type: SI Volatile: N**	0	165	15	deg	E
20.16	**Unused**					
20.17	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
20.18	**TorqUsedMaxSel (maximum used torque selector)** *TorqUsedMax (2.22)* selector: 0 = **TorqMax2005** *TorqMax (20.05),* default 1 = **AI1** analog input 1 2 = **AI2** analog input 2 3 = **AI3** analog input 3 4 = **AI4** analog input 4 5 = **AI5** analog input 5 6 = **AI6** analog input 6 **Int. Scaling: 1 == 1 Type: C Volatile: N**	TorqMax2005	AI6	TorqMax2005	-	C
20.19	**TorqUsedMinSel (minimum used torque selector)** *TorqUsedMin (2.23)* selector: 0 = **TorqMin2006** *TorqMin (20.06),* default 1 = **AI1** analog input 1 2 = **AI2** analog input 2 3 = **AI3** analog input 3 4 = **AI4** analog input 4 5 = **AI5** analog input 5 6 = **AI6** analog input 6 7 = **Negate2018** negated output of *TorqUsedMaxSel (20.18)* is used **Int. Scaling: 1 == 1 Type: C Volatile: N**	TorqMin2006	Negate	TorqMin2006	-	C
20.20	**Unused**					
20.21	**Unused**					
20.22	**TorqGenMax (maximum and minimum torque limit during regenerating)** Maximum and minimum torque limit - in percent of *MotNomTorque (4.23)* - only during regenerating. **Note:** The used torque limit depends also on the converter's actual limitation situation (e.g. other torque limits, current limits, field weakening). **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0	325	325	%	E

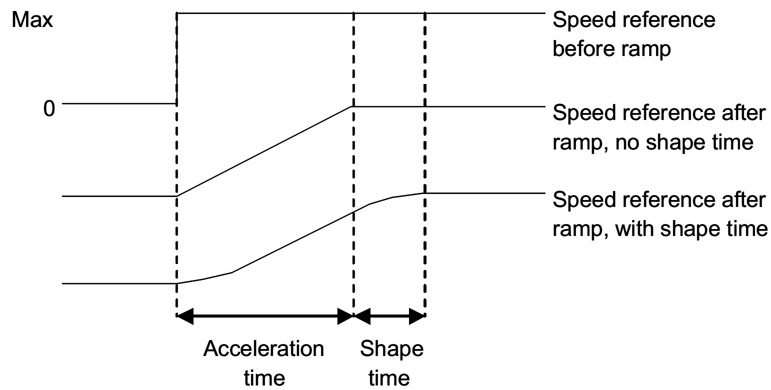
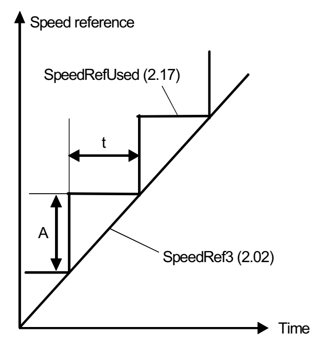
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 21	**Start / stop**					
21.01	**Unused**					
21.02	**Off1Mode (off 1 mode)** Conditions for motor deceleration when *UsedMCW (7.04)* bit 0 **On** (respectively **Off1N**) is set to low: 0 = **RampStop** The input of the drives ramp is set to zero. Thus the drive stops according to *DecTime1 (22.02)* or *DecTime2 (22.10)*. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. In case *TorqSelMod (26.03)* = **Auto** and **On** is set to low the torque selector is bypassed and the drive is forced to speed control, default. 1 = **TorqueLimit** The output of the drives ramp is set to zero. Thus the drive stops at the active torque limit. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. In case *TorqSelMod (26.03)* = **Auto** and **On** is set to low the torque selector is bypassed and the drive is forced to speed control. 2 = **CoastStop** The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. 3 = **DynBraking** dynamic braking **Note:** In case *UsedMCW (7.04)* bit 0 **On** and *UsedMCW (7.04)* bit 3 **Run** are set to low (run and on commands are taken away) at the same time or nearly contemporary *Off1Mode (21.02)* and *StopMode (21.03)* **must** have the same setting. **Int. Scaling: 1 == 1 Type: C Volatile: N**	RampStop	DynBraking	RampStop	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
21.03	**StopMode (stop mode)** Conditions for motor deceleration when *UsedMCW (7.04)* bit 3 **Run** is set to low: 0 = **RampStop** The input of the drives ramp is set to zero. Thus the drive stops according to *DecTime1 (22.02)* or *DecTime2 (22.10)*. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked. In case *TorqSelMod (26.03)* = **Auto** and **Run** is set to low the torque selector is bypassed and the drive is forced to speed control, default. 1 = **TorqueLimit** The output of the drives ramp is set to zero. Thus the drive stops at the active torque limit. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked. In case *TorqSelMod (26.03)* = **Auto** and **Run** is set to low the torque selector is bypassed and the drive is forced to speed control. 2 = **CoastStop** The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked. 3 = **DynBraking** dynamic braking **Note:** In case *UsedMCW (7.04)* bit 0 **On** and *UsedMCW (7.04)* bit 3 **Run** are set to low (run and on commands are taken away) at the same time or nearly contemporary *Off1Mode (21.02)* and *StopMode (21.03)* **must** have the same setting. **Int. Scaling: 1 == 1 Type: C Volatile: N**	RampStop	DynBraking	RampStop	-	C
21.04	**E StopMode (emergency stop mode)** Conditions for motor deceleration when *UsedMCW (7.04)* bit 2 **Off3N** (respectively E-stop) is set low: 0 = **RampStop** The input of the drives ramp is set to zero. Thus the drive stops according to *E StopRamp (22.04)*. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. In case *TorqSelMod (26.03)* = **Auto** and **Off3N** is set to low the torque selector is bypassed and the drive is forced to speed control. 1 = **TorqueLimit** The output of the drives ramp is set to zero. Thus the drive stops at the active torque limit. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. In case *TorqSelMod (26.03)* = **Auto** and **Off3N** is set to low the torque selector is bypassed and the drive is forced to speed control. 2 = **CoastStop** The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped, default. 3 = **DynBraking** dynamic braking **Note:** *E StopMode (21.04)* overrides *Off1Mode (21.02)* and *StopMode (21.03)*. **Int. Scaling: 1 == 1 Type: C Volatile: N**	RampStop	DynBraking	CoastStop	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
21.05	**E StopDecMin (emergency stop minimum deceleration rate)** During an emergency stop the deceleration of the drive is supervised. This supervision starts after the drive has received an emergency stop and the time delay defined in *DecMonDly (21.07)* is elapsed. In case the drive isn't able to decelerate within the window, defined by *E StopDecMin (21.05)* and *E StopDecMax (21.06)*, it is stopped by coasting and *AuxStatWord (8.02)* bit 2 **E-StopCoast** is set high. **Note:** The supervision is disabled in case *E StopDecMax (21.06)* or *E StopDecMin (21.05)* is set to default. **Int. Scaling: 1 == 1 rpm/s Type: I Volatile: N**	0	18000	18000	rpm/s	E
21.06	**E StopDecMax (emergency stop maximum deceleration rate)** During an emergency stop the deceleration of the drive is supervised. This supervision starts after the drive has received an emergency stop and the time delay defined in *DecMonDly (21.07)* is elapsed. In case the drive isn't able to decelerate within the window, defined by *E StopDecMin (21.05)* and *E StopDecMax (21.06)*, it is stopped by coasting and *AuxStatWord (8.02)* bit 2 **E-StopCoast** is set high. **Note:** The supervision is disabled in case *E StopDecMax (21.06)* or *E StopDecMin (21.05)* is set to default. **Int. Scaling: 1 == 1 rpm/s Type: I Volatile: N**	0	18000	18000	rpm/s	E
21.07	**DecMonDly (delay deceleration monitoring)** Time delay before the deceleration monitoring of the emergency stop starts. See also *E StopDecMin (21.05)* and *E StopDecMax (21.06).* **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	100	20	s	E
21.08	**Unused**					
21.09	**Unused**					
21.10	**FlyStart (flying start)** Selection of the desired operating response to a **Run** command [*UsedMCW (7.04))* bit 3] during braking or coasting: 0 = **StartFrom0** wait until the motor has reached zero speed [see *M1ZeroSpeedLim (20.03)*], then restart. In case the restart command comes before zero speed is reached **A137 SpeedNotZero** [*AlarmWord3 (9.08)* bit 4] is generated. 1 = **FlyingStart** start motor with its actual speed, when the drive was stopped by **RampStop**, **TorqueLimit** or **CoastStop**. Stop by **DynBraking** is not interrupted, wait until zero speed is reached, default 2 = **FlyStartDyn** start motor with its actual speed, when the drive was stopped by **RampStop**, **TorqueLimit**, **CoastStop** or **DynBraking**. **DynBraking** is interrupted. **Attention:** When using **FlyStartDyn** make sure, that the hardware (e.g. the switch disconnecting the braking resistor) is able to disconnect the current. **Int. Scaling: 1 == 1 Type: C Volatile: N**	StartFrom0	FlyingStart	FlyingStart		E
21.11	**Unused**					
21.12	**Unused**					
21.13	**Unused**					
21.14	**FanDly (fan delay)** After the drive has been switched off [*UsedMCW (7.04)* bit 0 **On** = 0], both fans (motor and converter) mustn't switched off before *FanDly (21.14)* has elapsed. If motor or converter overtemperature is pending, the delay starts after the temperature has dropped below the overtemperature limit. **Int. Scaling: 1 == 1 s Type: I Volatile: N**	0	300	0	s	E
21.15	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
21.16	**MainContCtrlMode (main contactor control mode)** *MainContCtrlMode (21.16)* determines the reaction to **On** and **Run** commands [*UsedMCW (7.04)* bits 0 and 3]: 0 = **On** main contactor closes with **On** = 1, default 1 = **On&Run** main contactor closes with **On** = **Run** = 1 2 = **OnHVCB** for high voltage AC circuit breaker configuration (for more information see chapter XXXX); not implemented yet 3 = **DCcontact** If a DC-breaker is used as a main contactor, it will be closed with **On** = 1. Additionally the armature voltage measurements are adapted to an open DC-breaker by clamping *SpeedActEMF (1.02)*, *ArmVoltActRel (1.13)*, *ArmVoltAct (1.14)* and *EMF VoltActRel (1.17)* to zero when the drive is **Off**. The clamping is released: either 100 ms after an **On** command (MCW bit 0) is given in case *DCBreakAck (10.23)* = **NotUsed** or when using the DC-breaker acknowledge with *DCBreakAck (10.23)* = **DIx** until the acknowledge signal indicates that the DC-breaker closed. **Note:** If the DC volt measurement is located at the motor terminals use 0 = **On** (Modified D5 – D7 converters) **Note:** The DC-breaker (US style) K1.1 is a special designed DC-breaker with one normally closed contact for the dynamic braking resistor R_B and two normally open contacts for C1 and D1. The DC-breaker should be controlled by *CurCtrlStart1 (6.03)* bit 10. The acknowledge signal can be connected to either *MainContAck (10.21)* or *DCBreakAck (10.23)*:  **Int. Scaling: 1 == 1 Type: C Volatile: N**	On	DCcontact	On	-	E
21.17	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
21.18	**FldHeatSel (field heat selector)** *FldHeatSel (21.18)* releases the field heating for motor 1 and motor 2: 0 = **NotUsed** field heating is off, default 1 = **On** field heating is on, as long as: **On** = 0 [*UsedMCW (7.04)* bit 0], **Off2N** = 1 [*UsedMCW (7.04)* bit 1] and **Off3N** = 1 [UsedMCW (7.04) bit 2] 2 = **OnRun** field heating is on as long as: **On** = 1, **Run** = 0 [*UsedMCW (7.04)* bit 3], **Off2N** = 1 and **Off3N** = 1 3 = **ACW Bit12** field heating is on as long as: **ACW Bit12** = 1 [*AuxCtrlWord (7.02)* bit 12] and **Run** = 0 4 = **ACW Bit13** field heating is on as long as: **ACW Bit13** = 1 [*AuxCtrlWord (7.02)* bit 13] and **Run** = 0 5 = **ACW Bit14** field heating is on as long as: **ACW Bit14** = 1 [*AuxCtrlWord (7.02)* bit 14] and **Run** = 0 6 = **ACW Bit15** field heating is on as long as: **ACW Bit15** = 1 [*AuxCtrlWord (7.02)* bit 15] and **Run** = 0 **Note:** The field heating references are set with *M1FldHeatRef (44.04)* and *M2FldHeatRef (49.06).* Field heating for the individual motor can be disabled when the belonging reference is set to zero. Field nominal currents are set with *M1NomFldCur (99.11)* and *M2NomFldCur (49.05).* **Note:** In case the field exciter is not connected via a separate field contactor following settings apply for field heating: – *MainContCtrlMode (21.16)* = **On** – *FldHeatSel (21.18)* = **OnRun** **Note:** When two motors in shared motion are used and field economy is needed for the dormant set *FldHeatSel (21.18)* = **NotUsed**. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	C
Group 22	**Speed ramp**					
22.01	**AccTime1 (acceleration time 1)** The time within the drive will accelerate from zero speed to *SpeedScaleAct (2.29):* – To expand the ramp time use *RampTimeScale (22.03)* – *AccTime1 (22.01)* can be released with *Ramp2Sel (22.11)* **Int. Scaling: 100 == 1 s Type: I Volatile: N**	0	300	20	s	C
22.02	**DecTime1 (deceleration time 1)** The time within the drive will decelerate from *SpeedScaleAct (2.29)* to zero speed: – To expand the ramp time use *RampTimeScale (22.03)* – *DecTime1 (22.02)* can be released with *Ramp2Sel (22.11)* **Int. Scaling: 100 == 1 s Type: I Volatile: N**	0	300	20	s	C
22.03	**RampTimeScale (ramp time scaling)** Multiplier for *AccTime1 (22.01) / AccTime2 (22.09)* and *DecTime1 (22.02) / DecTime2 (22.10)* to expand the ramp time. **Int. Scaling: 100 == 1 Type: I Volatile: N**	0.1	100	1	-	E
22.04	**E StopRamp (emergency stop ramp)** The time within the drive will decelerate from *SpeedScaleAct (2.29)* to zero speed. Either when emergency stop is released and *E StopMode (21.04)* = **RampStop** or as reaction to a fault of trip level 4 and *FaultStopMode (30.30)* = **RampStop**. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	3000	20	s	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
22.05	**ShapeTime (shape time)** Speed reference softening time. This function is bypassed during an emergency stop:  **Int. Scaling: 100 == 1 s Type: I Volatile: N**	0	30	0	s	E
22.06	**Unused**					
22.07	**VarSlopeRate (variable slope rate)** Variable slope is used to control the slope of the speed ramp during a speed reference change. It is active only with *VarSlopeRate (22.07)* ≠ 0. Variable slope rate and the drive's internal ramp are connected in series. Thus follows that the ramp times - *AccTime1 (22.01)* and *DecTime1 (22.02)* - have to be faster than the complete variable slope rate time. *VarSlopeRate (22.07)* defines the speed ramp time **t** for the speed reference change **A**:  **t** = cycle time of the overriding control (e.g. speed reference generation) **A** = speed reference change during cycle time **t** **Note:** In case the overriding control systems cycle time of the speed reference and *VarSlopeRate (22.07)* are equal the shape of *SpeedRef3 (2.02)* is a strait line. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	30000	0	ms	E
22.08	**BalRampRef (balance ramp reference)** The output of the speed ramp can be forced to the value defined by *BalRampRef (22.08)*. The function is released by setting *AuxCtrlWord (7.02)* bit 3 = 1. Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	0	rpm	E

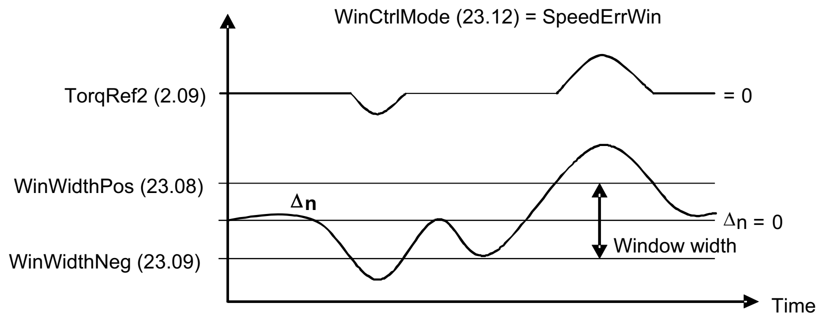
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
22.09	**AccTime2 (acceleration time 2)** The time within the drive will accelerate from zero speed to *SpeedScaleAct (2.29):* – To expand the ramp time use *RampTimeScale (22.03)* – *AccTime2 (22.09)* can be released with *Ramp2Sel (22.11)* **Int. Scaling: 100 == 1 s Type: I Volatile: N**	0	300	20	s	E
22.10	**DecTime2 (deceleration time 2)** The time within the drive will decelerate from *SpeedScaleAct (2.29)* to zero speed*:* – To expand the ramp time use *RampTimeScale (22.03)* – *DecTime2 (22.10)* can be released with *Ramp2Sel (22.11)* **Int. Scaling: 100 == 1 s Type: I Volatile: N**	0	300	20	s	E
22.11	**Ramp2Select (ramp 2 selector)** Select active ramp parameters: 0 = **Acc/Dec1** parameter set 1 [*AccTime1 (22.01)* and *DecTime1 (22.02)*] is active, default 1 = **Acc/Dec2** parameter set 2 [*AccTime2 (22.09)* and *DecTime2 (22.10)*] is active 2 = **SpeedLevel** If |*SpeedRef3 (2.02)*| ≤ |*SpeedLev (50.10)*|, then parameter set1 is active. If |*SpeedRef3 (2.02)*| > |*SpeedLev (50.10)*|, then parameter set 2 is active. 3 = **DI1** 0 = parameter set 1 is active, 1 = parameter set 2 is active 4 = **DI2** 0 = parameter set 1 is active, 1 = parameter set 2 is active 5 = **DI3** 0 = parameter set 1 is active, 1 = parameter set 2 is active 6 = **DI4** 0 = parameter set 1 is active, 1 = parameter set 2 is active 7 = **DI5** 0 = parameter set 1 is active, 1 = parameter set 2 is active 8 = **DI6** 0 = parameter set 1 is active, 1 = parameter set 2 is active 9 = **DI7** 0 = parameter set 1 is active, 1 = parameter set 2 is active 10 = **DI8** 0 = parameter set 1 is active, 1 = parameter set 2 is active 11 = **DI9** 0 = parameter set 1 is active, 1 = parameter set 2 is active, only available with digital extension board 12 = **DI10** 0 = parameter set 1 is active, 1 = parameter set 2 is active, only available with digital extension board 13 = **DI11** 0 = parameter set 1 is active, 1 = parameter set 2 is active, only available with digital extension board 14 = **MCW Bit11** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 11 15 = **MCW Bit12** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 12 16 = **MCW Bit13** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 13 17 = **MCW Bit14** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 14 18 = **MCW Bit15** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 15 19 = **ACW Bit12** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *AuxCtrlWord (7.02)* bit 12 20 = **ACW Bit13** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *AuxCtrlWord (7.02)* bit 13 21 = **ACW Bit14** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *AuxCtrlWord (7.02)* bit 14 22 = **ACW Bit15** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	Acc/Dec1	ACW Bit15	Acc/Dec1	-	E

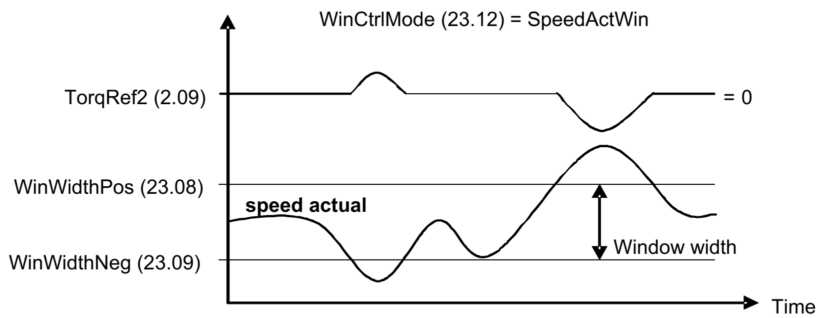
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
22.12	**JogAccTime (acceleration time jogging)** The time within the drive will accelerate from zero speed to *SpeedScaleAct (2.29)* in case of jogging: – When using jog command *Jog1 (10.17)* or *MainCtrlWord (7.01)* bit 8 speed is set by *FixedSpeed1 (23.02)* – When using jog command *Jog2 (10.18))* or *MainCtrlWord (7.01)* bit 9 speed is set by *FixedSpeed2 (23.03)* – To expand the ramp time use *RampTimeScale (22.03)* **Int. Scaling: 100 == 1 s Type: I Volatile: N**	0	300	20	s	E
22.13	**JogDecTime (deceleration time jogging)** The time within the drive will decelerate from *SpeedScaleAct (2.29)* to zero speed in case of jogging: – When using jog command *Jog1 (10.17)* or *MainCtrlWord (7.01)* bit 8 speed is set by *FixedSpeed1 (23.02)* – When using jog command *Jog2 (10.18))* or *MainCtrlWord (7.01)* bit 9 speed is set by *FixedSpeed2 (23.03)* – To expand the ramp time use *RampTimeScale (22.03)* **Int. Scaling: 100 == 1 s Type: I Volatile: N**	0	300	20	s	E
Group 23	**Speed reference**					
23.01	**SpeedRef (speed reference)** Main speed reference input for the speed control of the drive. Can be connected to *SpeedRefUsed (2.17)* via: – *Ref1Mux (11.02)* and *Ref1Sel (11.03)* or – *Ref2Mux (11.12)* and *Ref2Sel (11.06)* Internally limited from: $-(2.29)*\frac{32767}{20000}rpm \;\; to \;\; (2.29)*\frac{32767}{20000}rpm$ **Int. Scaling: (2.29) Type: SI Volatile: Y**	-10000	10000	0	rpm	E
23.02	**FixedSpeed1 (fixed speed 1)** *FixedSpeed1 (23.02)* is specifying a constant speed reference and overrides *SpeedRef2 (2.01)* at the speed ramp's input. It can be released by *Jog1 (10.17)* or *MainCtrlWord (7.01)* bit 8. The ramp times are set with *JogAccTime (22.12)* and *JogDecTime (22.13)*. Internally limited from: $-(2.29)*\frac{32767}{20000}rpm \;\; to \;\; (2.29)*\frac{32767}{20000}rpm$ **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	0	rpm	E
23.03	**FixedSpeed2 (fixed speed 2)** *FixedSpeed2 (23.03)* is specifying a constant speed reference and overrides *SpeedRef2 (2.01)* at the speed ramp's input. It can be released by *Jog2 (10.18)* or *MainCtrlWord (7.01)* bit 9. The ramp times are set with *JogAccTime (22.12)* and *JogDecTime (22.13)*. Internally limited from: $-(2.29)*\frac{32767}{20000}rpm \;\; to \;\; (2.29)*\frac{32767}{20000}rpm$ **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	0	rpm	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
23.04	**SpeedCorr (speed correction)** The *SpeedCorr (23.04)* is added to the ramped reference *SpeedRef3 (2.02).* Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Note:** Since this speed offset is added after the speed ramp, it must be set to zero prior to stopping the drive. **Int. Scaling: (2.29) Type: SI Volatile: Y**	-10000	10000	0	rpm	E
23.05	**SpeedShare (speed sharing)** Scaling factor *SpeedRefUsed (2.17).* Before speed ramp. **Int. Scaling: 10 == 1 % Type: SI Volatile: N**	-400	400	100	%	E
23.06	**SpeedErrFilt (filter for Δn)** Speed error (Δn) filter time 1. There are three different filters for actual speed and speed error (Δn): – *SpeedFiltTime (50.06)* is filtering the actual speed and should be used for filter times smaller than 30 ms. – *SpeedErrFilt (23.06)* and *SpeedErrFilt2 (23.11)* are filtering the speed error (Δn) and should be used for filter times greater than 30 ms. It is recommended to set *SpeedErrFilt (23.06) = SpeedErrFilt2 (23.11)*. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	E

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td></td><td>

Idea of Window Control:
The idea of the Window Control is to block the speed controller as long as the speed error (Δn) or speed actual remains within the window set by *WinWidthPos (23.08)* and *WinWidthNeg (23.09).* This allows the external torque reference - *TorqRef1 (2.08)* - to affect the process directly. If the speed error (Δn) or actual speed exceeds the programmed window, the speed controller becomes active and influences the process by means of *TorqRef2 (2.09)*. To release window control set *TorqSel (26.01)* = **Add** and *AuxCtrlWord (7.02)* bit 7 = 1.
This function could be called over/underspeed protection in torque control mode:





Note:
to open a window with a width of 100 rpm set *WinWidthPos (23.08)* = 50 rpm and *WinWidthNeg (23.09)* = -50 rpm.
</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>23.07</td><td>

WinIntegOn (window control integrator on)
Enables the integrator of the speed controller when window control is released:
0 = **Off** Integrator of the speed controller is blocked when window control is released
1 = **On** Integrator of the speed controller is enabled when window control is released
To release window control set *TorqSel (26.01)* = **Add** and *AuxCtrlWord (7.02)* bit 7 = 1.
Int. Scaling: 1 == 1 Type: C Volatile: N
</td><td>Off</td><td>On</td><td>Off</td><td>-</td><td>E</td></tr>
<tr><td>23.08</td><td>

WinWidthPos (positive window width)
Positive speed limit for the window control, when the speed error ($\Delta n = n_{ref} - n_{act}$) is positive.
Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$
Int. Scaling: (2.29) Type: I Volatile: N
</td><td>-10000</td><td>10000</td><td>0</td><td>rpm</td><td>E</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
23.09	**WinWidthNeg (negative window width)** Negative speed limit for the window control, when the speed error ($\Delta n = n_{ref} - n_{act}$) is negative. Internally limited from: $-(2.29) * \frac{32767}{20000} rpm \;\; to \;\; (2.29) * \frac{32767}{20000} rpm$ **Int. Scaling: (2.29) Type: I Volatile: N**	-10000	10000	0	rpm	E
23.10	**SpeedStep (speed step)** *SpeedStep (23.10)* is added to the speed error (Δn) at the speed controller's input. The given min./max. values are limited by *M1SpeedMin (20.02)* and *M1SpeedMax (20.02).* Internally limited from: $-(2.29) * \frac{32767}{20000} rpm \;\; to \;\; (2.29) * \frac{32767}{20000} rpm$ **Note:** Since this speed offset is added after the speed ramp, it must be set to zero prior to stopping the drive. **Int. Scaling: (2.29) Type: SI Volatile: Y**	-10000	10000	0	rpm	E
23.11	**SpeedErrFilt2 (2nd filter for Δn)** Speed error (Δn) filter time 2. There are three different filters for actual speed and speed error (Δn). *SpeedFiltTime (50.06)* is filtering the actual speed and should be used for filter times smaller than 30 ms. *SpeedErrFilt (23.06)* and *SpeedErrFilt2 (23.11)* are filtering the speed error (Δn) and should be used for filter times greater than 30 ms. It is recommended to set *SpeedErrFilt (23.06) = SpeedErrFilt2 (23.11)*. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	E
23.12	**WinCtrlMode (window control mode)** Window control mode: 0 = **SpeedErrWin** Standard window control, Speed error (Δn) has to be in a window defined by *WinWidthPos (23.08)* and *WinWidthNeg (23.09)*. Typically used for torque followers to limit differential speed, default. 1 = **SpeedActWin** Speed actual has to be in a window defined by *WinWidthPos (23.08)* and *WinWidthNeg (23.09).* Typically used for torque controlled test rigs to limit the no load speed. Example1: To get a window of 10 rpm width around the speed error (Δn) set: – *WinCtrlMode (23.12)* = **SpeedErrWin** – *WinWidthPos (23.08)* = 5 rpm and – *WinWidthNeg (23.09)* = -5 rpm Example2: To get a window (e.g. 500 rpm to 1000 rpm) around speed actual set: – *WinCtrlMode (23.12)* = **SpeedActWin** – *WinWidthPos (23.08)* = 1000 rpm and – *WinWidthNeg (23.09)* = 500 rpm To get a window (e.g. -50 rpm to 100 rpm) around speed actual set: – *WinCtrlMode (23.12)* = **SpeedActWin** – *WinWidthPos (23.08)* = 100 rpm and – *WinWidthNeg (23.09)* = -50 rpm **Int. Scaling: 1 == 1 Type: C Volatile: N**	SpeedErrWin	SpeedActWin	SpeedErrWin	-	E

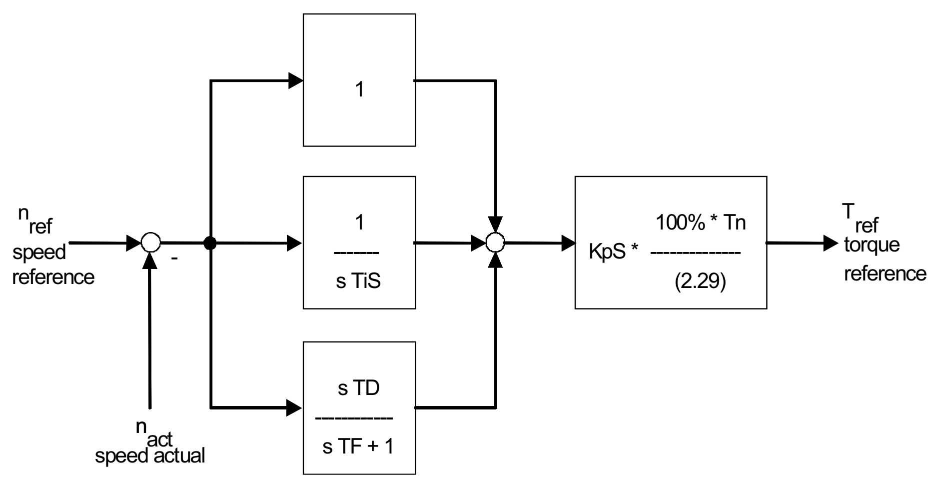
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
23.13	**AuxSpeedRef (auxiliary speed reference)** Auxiliary speed reference input for the speed control of the drive. Can be connected to *SpeedRefUsed (2.17)* via: – *Ref1Mux (11.02)* and *Ref1Sel (11.03)* or – *Ref2Mux (11.12)* and *Ref2Sel (11.06)* Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Int. Scaling: (2.29) Type: SI Volatile: Y**	-10000	10000	0	rpm	E
23.14	**Unused**					
23.15	**DirectSpeedRef (direct speed reference)** Direct speed input is connected to *SpeedRef3 (2.02)* by means of *AuxCtrlWord2 (7.03)* bit 10 = 1 and replaces the speed ramp output. Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Note:** Since this speed offset is added after the speed ramp, it must be set to zero prior to stopping the drive. **Int. Scaling: (2.29) Type: SI Volatile: Y**	-10000	10000	0	rpm	E
23.16	**SpeedRefScale (speed reference scaling)** Speed reference scaling. After *SpeedRef3 (2.02)* and before *SpeedRef4 (2.18)*. **Int. Scaling: 100 == 1 Type: I Volatile: N**	-100	100	1	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 24	**Speed control**					

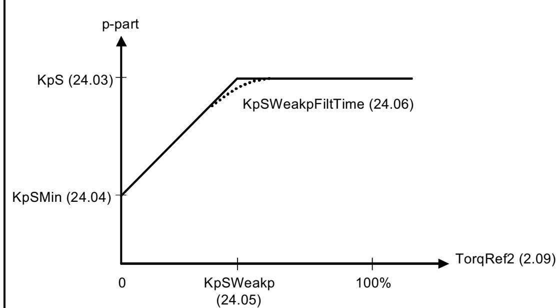
The Speed controller is based on a PID algorithm and is presented as follows:

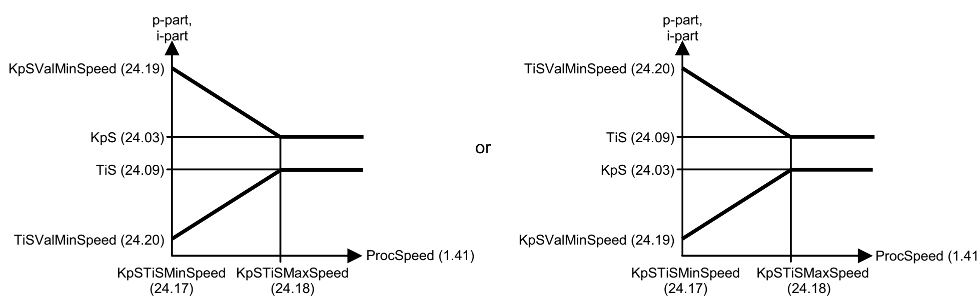
$$T_{ref(s)} = KpS * \left[\left(n_{ref(s)} - n_{act(s)} \right) * \left(1 + \frac{1}{s\,TiS} + \frac{s\,TD}{s\,TF + 1} \right) \right] * \frac{100\% * T_n}{(2.29)}$$

with:
T_{ref} = torque reference
KpS = proportional gain [*KpS (24.03)*]
N_{ref} = speed reference
N_{act} = speed actual
TiS = Integration time [*TiS (24.09)]*
TD = Derivation time [*DerivTime (24.12)]*
TF = Derivation filter time [*DerivFiltTime (24.13)]*
T_n = nominal motor torque
(2.29) = actual used speed scaling [*SpeedScaleAct (2.29)*]



Index	Signal / Parameter name	min.	max.	def.	unit	E/C
24.01	**Unused**					
24.02	**DroopRate (droop rate)** Droop is used in certain applications to archive a speed drop depending on the load. This function may become necessary for proper load sharing between drives which are linked via material (e.g. paper, steel, foil) and running with a common speed reference. The amount of speed drop caused by the load is determined by *DroopRate (24.02).* The result is a load dependent speed decrease in percent of *SpeedScaleAct (2.29).* Example: With *DroopRate (24.02)* = 3 % and *TorqIntegRef (2.05)* = 100 % (nominal motor torque) the actual speed decreases 3 % of *SpeedScaleAct (2.29).* **Int. Scaling: 10 == 1 % Type: I Volatile: N**	0	100	0	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
24.03	**KpS (p-part speed controller)** Proportional gain of the speed controller can be released by means of *Par2Select (24.29).* Example: The controller generates 15 % of motor nominal torque with *KpS (24.03)* = 3, if the speed error (Δn) is 5 % of *SpeedScaleAct (2.29).* **Int. Scaling: 100 == 1 Type: I Volatile: N**	0	325	5	-	C
	Load adaptive proportional gain:  The adaptive proportional gain of the speed controller is used to smooth out disturbances which are caused by low loads and backlash. Moderate filtering of the speed error (Δn) is typically not enough to tune the drive. The load adaptation is valid for positive and negative torque.					
24.04	**KpSMin (minimum p-part speed controller)** *KpSMin (24.04)* determines the proportional gain when the speed controller output [*TorqRef2 (2.09)*] is zero. *KpSMin (24.04)* cannot be greater than *KpS (24.03).* **Int. Scaling: 100 == 1 Type: I Volatile: N**	0	(24.03)	0	-	E
24.05	**KpSWeakp (weakening point of p-part speed controller)** The speed controller output value [*TorqRef2 (2.09)*], in percent of *MotNomTorque (4.23)*, where the gain equals *KpS (24.03)*. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	325	0	%	E
24.06	**KpSWeakpFiltTime (filter time for weakening point of p-part speed controller)** Filter time to soften the proportional gains rate of change. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	100	ms	E
24.07	**Unused**					
24.08	**Unused**					
24.09	**TiS (i-part speed controller)** Integral time of the speed controller can be released by means of *Par2Select (24.29). TiS (24.09)* defines the time within the integral part of the controller achieves the same value as the proportional part. Example: The controller generates 15 % of motor nominal torque with *KpS (24.03)* = 3, if the speed error (Δn) is 5 % of *SpeedScaleAct (2.29).* On that condition and with *TiS (24.09)* = 300 ms follows: – the controller generates 30 % of motor nominal torque, if the speed error (Δn) is constant, after 300 ms are elapsed (15 % from proportional part and 15 % from integral part). Setting *TIS (24.09)* to 0 ms disables the integral part of the speed controller and resets its integrator. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	2500	ms	C
24.10	**TiSInitValue (initial value for i-part speed controller)** Initial value of the speed controller integrator, in percent of *MotNomTorque (4.23)*. The integrator is set as soon as **RdyRef** [*MainStatWord (8.01)*] becomes valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	325	0	%	E

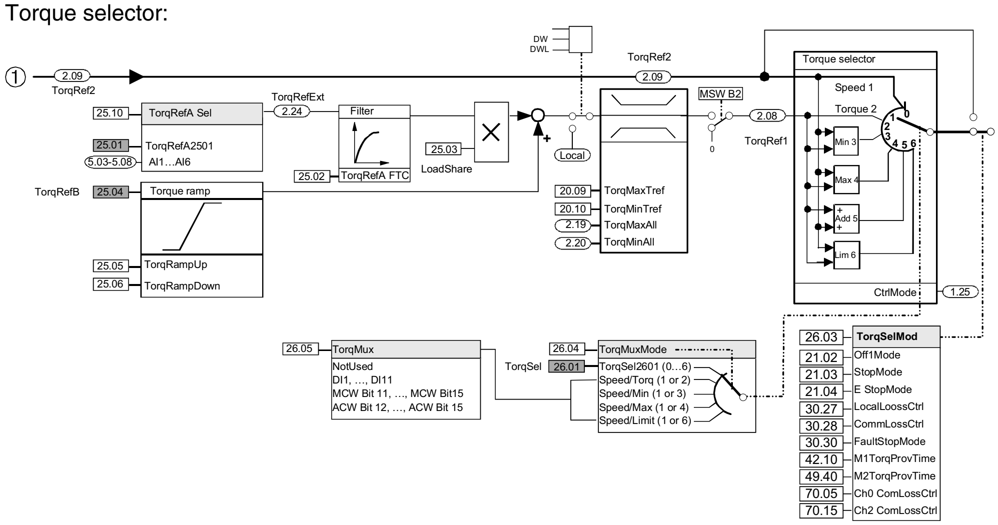
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
24.11	**BalRef (balance speed reference)** External value in percent of *MotNomTorque (4.23)*. Both, i-part and output of the speed controller are forced to *BalRef (24.11)* when *AuxCtrlWord (7.02)* bit 8 = 1. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	325	0	%	E
24.12	**DerivTime (d-part speed controller)** Speed controller derivation time. *DerivTime (24.12)* defines the time within the speed controller derives the error value. The speed controller works as PI controller, if *DerivTime (24.12)* is set to zero. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	E
24.13	**DerivFiltTime (filter time for d-part speed controller)** Derivation filter time. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	8	ms	E
24.14	**AccCompDerTime (acceleration compensation derivation time)** *AccCompDerTime (24.14)* compensates the inertia by adding the derived and weighted *SpeedRef4 (2.18)* to the speed controller output. The acceleration compensation is inactive, if *AccCompDerTime (24.14)* is set to zero. Example: *AccCompDerTime (24.14)* equals the time required to accelerate the drive to *SpeedScaleAct (2.29)* with motor nominal torque. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	1000	0	s	E
24.15	**AccCompFiltTime (filter time acceleration compensation)** Acceleration compensation filter time. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	8	ms	E
24.16	**Unused**					
	Speed adaptive proportional gain and integral time:  In certain applications it is useful to increase / decrease the proportional gain *[KpS (24.03)]* and decrease / increase the integral time [*TiS (24.09)*] at low speeds to improve the performance of the speed control. The linear increase and decrease of these parameters starts at *KpSTiSMaxSpeed (24.18)* and ends at *KpSTiSMinSpeed (24.17)* by means of *KpSValMinSpeed (24.19)* and *TiSValMinSpeed (24.20)*. The speed adaptation is valid for positive and negative speeds.					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
24.17	**KpSTiSMinSpeed (minimum speed for p- / i-part speed controller)** The speed limit below which the proportional gain and the integral time are defined by *KpSValMinSpeed (24.19)* and *TiSValMinSpeed (24.20).*The used speed is *ProcSpeed (1.41).* Internally limited from: $0rpm\ \ to\ \ (2.29) * \frac{32767}{20000} rpm$ **Int. Scaling: (2.29) Type: I Volatile: N**	0	(24.18)	0	rpm	E
24.18	**KpSTiSMaxSpeed (maximum speed for p- / i-part speed controller)** The speed limit above which the proportional gain and the integral time become constant and are defined by *KpS (24.03)* and *TiS (24.09).* The used speed is *ProcSpeed (1.41).* Internally limited from: $0rpm\ \ to\ \ (2.29) * \frac{32767}{20000} rpm$ **Int. Scaling: (2.29) Type: I Volatile: N**	(24.17)	10000	0	rpm	E
24.19	**KpSValMinSpeed (p-part speed controller value at minimum speed)** *KpSValMinSpeed (24.19)* determines the proportional gain percentage at the speed defined by parameter *KpSTiSMinSpeed (24.17)*. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	0	500	100	%	E
24.20	**TiSValMinSpeed (i-part speed controller value at minimum speed)** *TiSValMinSpeed (24.20)* determines the integral time percentage at the speed defined by parameter *KpSTiSMinSpeed (24.17)*. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	0	500	100	%	E
24.21	**ZeroFreqRFE (zero frequency resonance frequency eliminator)** Frequency of zero. The filter is located at the input of the speed controller. **Int. Scaling: 10 == 1 Hz Type: I Volatile: N**	0	150	45	Hz	E
24.22	**ZeroDampRFE (zero damping resonance frequency eliminator)** Damping of zero. **Int. Scaling: 1000 == 1 Type: I Volatile: N**	-1	1	0		E
24.23	**PoleFreqRFE (pole frequency resonance frequency eliminator)** Frequency of pole. The filter is located at the input of the speed controller. **Int. Scaling: 10 == 1 Hz Type: I Volatile: N**	0	150	40	Hz	E
24.24	**PoleDampRFE (pole damping resonance frequency eliminator)** Damping of pole. **Int. Scaling: 1000 == 1 Type: I Volatile: N**	0	1	0.25		E
24.25	**SpeedErrorScale (Δn scaling)** Scaling factor speed error (Δn). **Int. Scaling: 10 == 1 % Type: I Volatile: N**	10	400	100	%	E
24.26	**Unused**					
24.27	**KpS2 (2nd p-part speed controller)** 2nd proportional gain of the speed controller can be released by means of *Par2Select (24.29).* **Int. Scaling: 100 == 1 Type: I Volatile: N**	0	325	5	-	E
24.28	**TiS2 (2nd i-part speed controller)** 2nd integral time of the speed controller can be released by means of *Par2Select (24.29).* **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	2500	ms	E

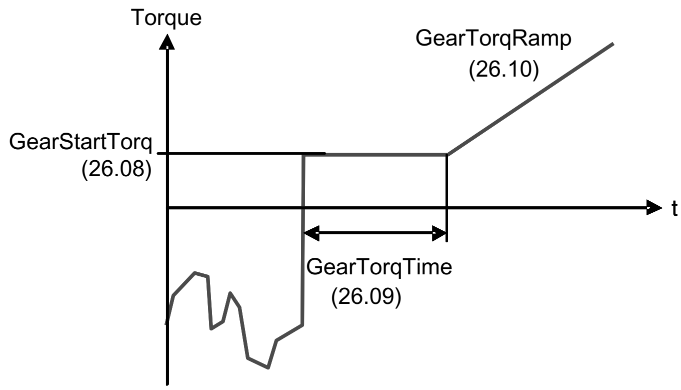
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
24.29	**Par2Select (selector for 2nd set of speed controller parameters)** Select active speed controller parameters: 0 = **ParSet1** parameter set 1 [*KpS (24.03)* and *TiS (24.09)*] is active, default 1 = **ParSet2** parameter set 2 [*KpS2 (24.27)* and *TiS2 (24.28)*] is active 2 = **SpeedLevel** If |*MotSpeed (1.04)*| $\leq$ |*SpeedLev (50.10)*|, then parameter set1 is active. If |*MotSpeed (1.04)*| > |*SpeedLev (50.10)*|, then parameter set 2 is active. 3 = **SpeedError** If |*SpeedErrNeg (2.03)*| $\leq$ |*SpeedLev (50.10)*|, then parameter set1 is active. If | *SpeedErrNeg (2.03)*| > |*SpeedLev (50.10)*|, then parameter set 2 is active. 4 = **DI1** 0 = parameter set 1 is active, 1 = parameter set 2 is active 5 = **DI2** 0 = parameter set 1 is active, 1 = parameter set 2 is active 6 = **DI3** 0 = parameter set 1 is active, 1 = parameter set 2 is active 7 = **DI4** 0 = parameter set 1 is active, 1 = parameter set 2 is active 8 = **DI5** 0 = parameter set 1 is active, 1 = parameter set 2 is active 9 = **DI6** 0 = parameter set 1 is active, 1 = parameter set 2 is active 10 = **DI7** 0 = parameter set 1 is active, 1 = parameter set 2 is active 11 = **DI8** 0 = parameter set 1 is active, 1 = parameter set 2 is active 12 = **DI9** 0 = parameter set 1 is active, 1 = parameter set 2 is active, only available with digital extension board 13 = **DI10** 0 = parameter set 1 is active, 1 = parameter set 2 is active, only available with digital extension board 14 = **DI11** 0 = parameter set 1 is active, 1 = parameter set 2 is active, only available with digital extension board 15 = **MCW Bit11** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 11 16 = **MCW Bit12** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 12 17 = **MCW Bit13** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 13 18 = **MCW Bit14** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 14 19 = **MCW Bit15** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *MainCtrlWord (7.01)* bit 15 20 = **ACW Bit12** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *AuxCtrlWord (7.02)* bit 12 21 = **ACW Bit13** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *AuxCtrlWord (7.02)* bit 13 22 = **ACW Bit14** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *AuxCtrlWord (7.02)* bit 14 23 = **ACW Bit15** 0 = parameter set 1 is active, 1 = parameter set 2 is active, *AuxCtrlWord (7.02)* bit 15 **Note:** Load and speed dependent adaptation parameters are valid regardless of the selected parameter set. **Int. Scaling: 1 == 1 Type: C Volatile: N**	ParSet1	ACW Bit15	ParSet1		E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 25	**Torque reference**					
25.01	**TorqRefA (torque reference A)** External torque reference in percent of *MotNomTorque (4.23)*. *TorqRefA (25.01)* can be scaled by *LoadShare (25.03)*. **Note:** *TorqRefA (25.01)* is only valid, if *TorqRefA Sel (25.10)* = **TorqRefA2501**. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-325	325	0	%	E
25.02	**TorqRefA FTC (torque reference A filter time)** *TorqRefA (25.01)* filter time. **Int. Scaling: 1 == 1 ms Type: SI Volatile: N**	0	10000	0	ms	E
25.03	**LoadShare (load share)** Scaling factor *TorqRefA (25.01)*. **Int. Scaling: 10 == 1 % Type: SI Volatile: N**	-400	400	100	%	E
25.04	**TorqRefB (torque reference B)** External torque reference in percent of *MotNomTorque (4.23)*. *TorqRefB (25.04)* is ramped by *TorqRampUp (25.05)* and *TorqRampDown (25.06)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-325	325	0	%	E
25.05	**TorqRampUp (torque ramp up)** Ramp time from 0 % to 100 %, of *MotNomTorque (4.23)*, for. *TorqRefB (25.04)*. **Int. Scaling: 100 = 1 s Type: I Volatile: N**	0	120	0	s	E
25.06	**TorqRampDown (torque ramp down)** Ramp time from 100 % to 0 %, of *MotNomTorque (4.23)*, for. *TorqRefB (25.04)*. **Int. Scaling: 100 = 1 s Type: I Volatile: N**	0	120	0	s	E
25.07	**Unused**					
25.08	**Unused**					
25.09	**Unused**					
25.10	**TorqRefA Sel (torque reference A selector)** Selector for *TorqRefExt (2.24)*: 0 = **TorqRefA2501** *TorqRefA (25.01),* default 1 = **AI1** analog input AI1 2 = **AI2** analog input AI2 3 = **AI3** analog input AI3 4 = **AI4** analog input AI4 5 = **AI5** analog input AI5 6 = **AI6** analog input AI6 **Int. Scaling: 1 == 1 Type: C Volatile: N**	TorqRefA2501	AI6	TorqRefA2501	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 26	**Torque reference handling**					
26.01	**TorqSel (torque selector)** Torque reference selector: 0 = **Zero** zero control, torque reference = 0 1 = **Speed** speed control, default 2 = **Torque** torque control 3 = **Minimum** minimum control: min [*TorqRef1 (2.08), TorqRef2 (2.09)*] 4 = **Maximum** maximum control: max [*TorqRef1 (2.08), TorqRef2 (2.09)*] 5 = **Add** add control: *TorqRef1 (2.08) +TorqRef2 (2.09),* used for window control 6 = **Limitation** limitation control: *TorqRef1 (2.08)* limits *TorqRef2 (2.09)*. If *TorqRef1 (2.08)* = 50%, then *TorqRef2 (2.09)* is limited to ±50%. The output of the torque reference selector is *TorqRef3 (2.10).* The currently used control mode is displayed in *CtrlMode (1.25).* If the drive is in torque control *AuxStatWord (8.02)* bit 10 is set. **Note:** *TorqSel (26.01)* is only valid, if *TorqMuxMode (26.04)* = **TorqSel2601**. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Zero	Limitation	Speed	-	E
26.02	**LoadComp (load compensation)** Load compensation - in percent of *MotNomTorque (4.23)* -added to *TorqRef3 (2.10).* The sum of *TorqRef3 (2.10)* and the *LoadComp (26.02)* results in *TorqRef4 (2.11).* **Note:** Since this torque offset is added, it must be set to zero prior to stopping the drive. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	325	0	%	E
26.03	**TorqSelMod (torque selector mode)** Mode setting for the torque selector: 0 = **Auto** the torque selector is bypassed and the drive is forced to speed control in case the mode described in: • *Off1Mode (21.02)*, • *StopMode (21.03)*, • *E StopMode (21.04)*, • *LocalLossCtrl (30.27)*, • *ComLossCtrl (30.28)*, • *FaultStopMode (30.30)*, • *M1TorqProvTime (42.10),* • *M2TorqProvTime (49.40),* • *Ch0 ComLossCtrl (70.05)* or • *Ch2 ComLossCtrl (70.15)* is active and the parameter is set to **RampStop** or **TorqueLimit**, default 1 = **Fix** the torque selector is fixed to the value set by *TorqSel (26.01), TorqMuxMode (26.04)* and *TorqMux (26.05)* **Note:** The setting of *TorqSelMod (26.03)* is especially affecting drives using torque control (e.g. master-follower). **Int. Scaling: 1 == 1 Type: C Volatile: N**	Auto	Fix	Auto	-	E

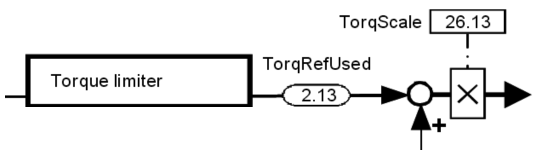
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	Torque selector:  					
26.04	**TorqMuxMode (torque multiplexer mode)** *TorqMuxMode (26.04)* selects a pair of operation modes. The change between operation modes is done by means of *TorqMux (26.05).* Torque reference multiplexer: 0 = **TorqSel2601** operation mode depends on *TorqSel (26.01)*, default 1 = **Speed/Torq** operation mode depends on *TorqMux (26.05)*: - binary input = 0 ⇒ speed control (1) - binary input = 1 ⇒ torque control (2) 2 = **Speed/Min** operation mode depends on *TorqMux (26.05):* - binary input = 0 ⇒ speed control (1) - binary input = 1 ⇒ minimum control (3) 3 = **Speed/Max** operation mode depends on *TorqMux (26.05):* - binary input = 0 ⇒ speed control (1) - binary input = 1 ⇒ maximum control (4) 4 = **Speed/Limit** operation mode depends on *TorqMux (26.05):* - binary input = 0 ⇒ speed control (1) - binary input = 1 ⇒ limitation control (6) **Int. Scaling: 1 == 1 Type: C Volatile: N**	TorqSel2601	Speed/Limit	TorqSel2601	-	E

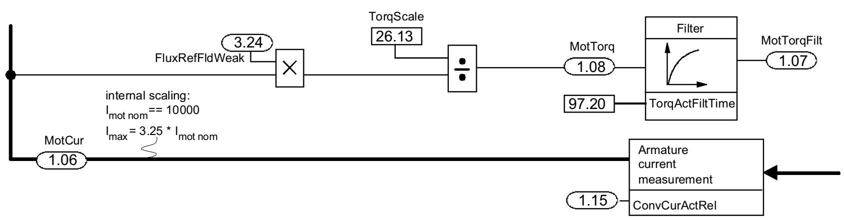
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
26.05	**TorqMux (torque multiplexer)** *TorqMux (26.05)* selects a binary input to change between operation modes. The choice of the operation modes is provided by means of *TorqMuxMode (26.04)*. Torque reference multiplexer binary input: 0 = **NotUsed** operation mode depends on *TorqSel (26.01)*, default 1 = **DI1** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)* 2 = **DI2** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)* 3 = **DI3** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)* 4 = **DI4** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)* 5 = **DI5** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)* 6 = **DI6** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)* 7 = **DI7** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)* 8 = **DI8** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)* 9 = **DI9** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, only available with digital extension board 10= **DI10** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, only available with digital extension board 11 = **DI11** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, only available with digital extension board 12 = **MCW Bit11** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** 0 = speed control, 1 = depends on *TorqMuxMode (26.04)*, *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	E
26.06	**Unused**					
26.07	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
26.08	**GearStartTorq (gearbox starting torque)** Gear backlash compensation: – *GearStartTorq (26.08)* is the reduced torque limit - in percent of *MotNomTorque (4.23)* - used after a torque direction change. The torque limit is reduced for the time defined by *GearTorqTime (26.09).*  **Int. Scaling: 100 = 1 % Type: I Volatile: N**	0	325	325	%	E
26.09	**GearTorqTime (gearbox torque time)** Gear backlash compensation: – When the torque is changing its direction, the torque limit is reduced for the time defined by *GearTorqTime (26.09)*. **Int. Scaling: 1 = 1 ms Type: I Volatile: N**	0	10000	100	ms	E
26.10	**GearTorqRamp (gearbox torque ramp)** Gear backlash compensation: – When the torque is changing its direction, the torque limit is reduced for the time defined by *GearTorqTime (26.09).* After the time has elapsed, the torque limit is increased to its normal value according to the ramp time defined by *GearTorqRamp (26.10). GearTorqRamp (26.10)* defines the time within the torque increases from zero- to *MotNomTorque (4.23)*. **Int. Scaling: 1 = 1 ms Type: I Volatile: N**	0	64000	100	ms	E
26.11	**Unused**					
26.12	**Unused**					

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>26.13</td><td>

TorqScale (torque scaling)
Scaling of *TorqRefUsed (2.13)* and *MotTorq (1.08)*:





Int. Scaling: 100 == 1 Type: I Volatile: Y
</td><td>0.1</td><td>6</td><td>1</td><td>-</td><td>E</td></tr>
<tr><td>26.14</td><td>

Unused
</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>26.15</td><td>

TorqCorrect (torque correction)
Torque correction value in percent of *MotNomTorque (4.23)*:
0 = **NotUsed** no torque correction used, default
1 = **AI1** torque correction via AI1 (fast AI)
2 = **AI2** torque correction via AI2 (fast AI)
3 = **AI3** torque correction via AI3
4 = **AI4** torque correction via AI4
5 = **AI5** torque correction via AI5
6 = **AI6** torque correction via AI6
Note:
If *TorqCorrect (26.15)* = **AI3** then AI3 is connected to *TorqCorr (2.14)* and thus added to *TorqRefUsed (2.13).*
Note:
Since this torque offset is added, it must be set to zero prior to stopping the drive.
Int. Scaling: 1 == 1 Type: C Volatile: N
</td><td>NotUsed</td><td>AI6</td><td>NotUsed</td><td>-</td><td>E</td></tr>
<tr><td>**Group 30**</td><td colspan="6">

Fault functions
</td></tr>
<tr><td>30.01</td><td>

StallTime (stall time)
The time allowed for the drive to undershoot *StallSpeed (30.02)* and exceed *StallTorq (30.03).* A triggered stall protection leads to **F531 MotorStalled** *[FaultWord2 (9.02)* bit 14].
The stall protection is inactive, if *StallTime (30.01)* is set to zero.
Int. Scaling: 1 == 1 s Type: I Volatile: N
</td><td>0</td><td>200</td><td>0</td><td>s</td><td>C</td></tr>
<tr><td>30.02</td><td>

StallSpeed (stall speed)
Actual speed limit used for stall protection.
Internally limited from: $0\,rpm\ \ to\ \ (2.29)\,rpm$
Int. Scaling: (2.29) Type: I Volatile: N
</td><td>0</td><td>1000</td><td>5</td><td>rpm</td><td>C</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
30.03	**StallTorq (stall torque)** Actual torque limit - in percent of *MotNomTorque (4.23)* - used for stall protection. **Int. Scaling: 100 = 1 % Type: I Volatile: N**	0	325	75	%	C
30.04	**Unused**					
30.05	**ResCurDetectSel (residual current detection selector)** The drive trips with **F505 ResCurDetect** [*FaultWord1 (9.01)* bit 4] if the earth current exceeds *ResCurDetectLim (30.06)* for *ResCurDetectDel (30.07)*: 0 = **NotUsed** residual current detection is blocked, default 1 = **AI4** The earth current is measured by means of a current difference sensor in combination with AI4 (X3:11 and X3:12) on the SDCS-IOB-3 board. 2 = **DI1** The earth current is measured by means of an external device (e.g. Bender relays). 3 = **DI2** The earth current is measured by means of an external device (e.g. Bender relays). 4 = **DI3** The earth current is measured by means of an external device (e.g. Bender relays). 5 = **DI4** The earth current is measured by means of an external device (e.g. Bender relays). 6 = **DI5** The earth current is measured by means of an external device (e.g. Bender relays). 7 = **DI6** The earth current is measured by means of an external device (e.g. Bender relays). 8 = **DI7** The earth current is measured by means of an external device (e.g. Bender relays). 9 = **DI8** The earth current is measured by means of an external device (e.g. Bender relays. 10 = **DI9** The earth current is measured by means of an external device (e.g. Bender relays). Only available with digital extension board 11 = **DI10** The earth current is measured by means of an external device (e.g. Bender relays. Only available with digital extension board 12 = **DI11** The earth current is measured by means of an external device (e.g. Bender relays). Only available with digital extension board **Note:** If *ResCurDetectSel (30.05)* is connected to a digital input only *ResCurDetectDel (30.07)* remains valid. The trip limit *ResCurDetectLim (30.06)* is adjusted at the external device. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI11	NotUsed	-	E
30.06	**ResCurDetectLim (residual current detection limit)** Residual current detection tripping level in amperes at the primary side of the current transformer (ratio is 400 : 1). If *ResCurDetectSel (30.05)* is connected to a digital input *ResCurDetectLim (30.06)* is deactivated, because the limit is adjusted at the external device. **Int. Scaling: 10 == 1 A Type: I Volatile: N**	0	20	4	A	E
30.07	**ResCurDetectDel (residual current detection delay)** Time delay for **F505 ResCurDetect** [*FaultWord1 (9.01)*]. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	10	ms	E
30.08	**ArmOvrVoltLev (armature overvoltage level)** The drive trips with **F503 ArmOverVolt** [*FaultWord1 (9.01)* bit 2] if *ArmOvrVoltLev (30.08)* - in percent of *M1NomVolt (99.02)* - is exceeded. It is recommended to set *ArmOvrVoltLev (30.08)* at least 20 % higher than *M1NomVolt (99.02)*. Example: With *M1NomVolt (99.02)* = 525 V and *ArmOvrVoltLev (30.08)* = 120 % the drive trips with armature voltages > 630 V. The overvoltage supervision is inactive, if *ArmOvrVoltLev (30.08)* is set to 328 % or higher. **Int. Scaling: 10 == 1 % Type: I Volatile: N**	20	500	120	%	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
30.09	**ArmOvrCurLev (armature overcurrent level)** The drive trips with **F502 ArmOverCur** [*FaultWord1 (9.01)* bit 1] if *ArmOvrCurLev (30.09)* - in percent of *M1NomCur (99.03)* - is exceeded. It is recommended to set *ArmOvrCurLev (30.09)* at least 25 % higher than *M1NomCur (99.03)*. Example: With *M1NomCur (99.03)* = 850 A and *ArmOvrCurLev (30.09)* = 250 % the drive trips with armature currents > 2125 A. **Int. Scaling: 10 == 1 % Type: I Volatile: N**	20	400	250	%	C
30.10	**ArmCurRiseMax (maximum rise armature current)** The drive trips with **F539 FastCurRise** [*FaultWord3 (9.03)* bit 6] if *ArmCurRiseMax (30.10)* - in percent of *M1NomCur (99.03)* - per 1 ms is exceeded. **Note:** This trip opens the main contactor and the DC-breaker, if present. **Int. Scaling: 100 == 1 %/ms Type: I Volatile: N**	0	325	325	%/ms	E
30.11	**Unused**					
30.12	**M1FldMinTrip (motor 1 minimum field trip)** The drive trips with **F541 M1FexLowCur** [*FaultWord3 (9.03)* bit 8] if *M1FldMinTrip (30.12)* - in percent of *M1NomFldCur (99.11)* - is still undershot when *FldMinTripDly (45.18)* is elapsed. **Note:** *M1FldMinTrip (30.12)* is not valid during field heating and field economy. In this case the trip level is automatically set to 50 % of *M1FldHeatRef (44.04)*. The drive trips with **F541 M1FexLowCur** [*FaultWord3 (9.03)* bit 8] if 50 % of *M1FldHeatRef (44.04)* is still undershot when *FldMinTripDly (45.18)* is elapsed. **Note:** *M1FldMinTrip (30.12)* is not valid for *FldCtrlMode (44.01)* = **Fix/Opti**, **EMF/Opti**, **Fix/Rev/Opti** or **EMF/Rev/Opti**. In this case the trip level is automatically set to 50 % of *FldCurRefM1 (3.30)*. The drive trips with **F541 M1FexLowCur** [*FaultWord3 (9.03)* bit 8] if 50 % of *FldCurRefM1 (3.30)* is still undershot when *FldMinTripDly (45.18)* is elapsed. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	100	50	%	E
30.13	**M1FldOvrCurLev (motor 1 field overcurrent level)** The drive trips with **F515 M1FexOverCur** [*FaultWord1 (9.01)* bit 14] if *M1FldOvrCurLev (30.13)* - in percent of *M1NomFldCur (99.11)* - is exceeded. It is recommended to set *M1FldOvrCurtLev (30.13)* at least 25 % higher than *M1NomFldCur (99.11)*. The field overcurrent fault is inactive, if *M1FldOvrCurLev (30.13)* is set to 135 %. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	135	125	%	E
30.14	**SpeedFbMonLev (speed feedback monitor level)** The drive reacts according to *SpeedFbFltSel (30.17)* or trips with **F553 TachPolarity** [*FaultWord4 (9.04)* bit 4] if the measured speed feedback [*SpeedActEnc (1.03), SpeedActTach (1.05) or SpeedActEnc2 (1.42)*] does not exceed *SpeedFbMonLev (30.14)* while the measured EMF exceeds *EMF FbMonLev (30.15)*. Internally limited from: $0 rpm \ \ to \ \ (2.29) * \frac{32767}{20000} rpm$ Example: With *SpeedFbMonLev (30.14)* = 15 rpm and *EMF FbMonLev (30.15)* = 50 V the drive trips when the EMF is > 50 V while the speed feedback is ≤ 15 rpm. **Int. Scaling: (2.29) Type: I Volatile: N**	0	10000	15	rpm	E
30.15	**EMF FbMonLev (EMF feedback monitor level)** The speed measurement monitoring function is activated, when the measured EMF exceeds *EMF FbMonLev (30.15)*. See also *SpeedFbMonLev (30.14)*. **Int. Scaling: 1 == 1 V Type: I Volatile: N**	0	2000	50	V	E

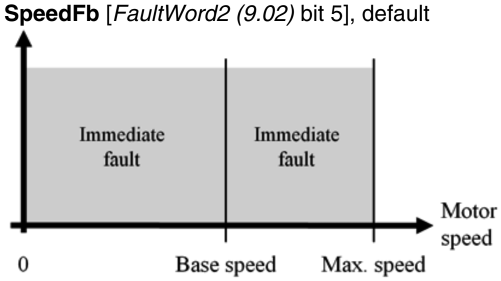
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
30.16	**M1OvrSpeed (motor 1 overspeed)** The drive trips with **F532 MotOverSpeed** [*FaultWord2 (9.02)* bit 15] if *M1OvrSpeed (30.16)* is exceeded. It is recommended to set *M1OvrSpeed (30.16)* at least 20 % higher than the maximum motor speed. Internally limited from: $0rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ The overspeed fault for motor 1 is inactive, if *M1OvrSpeed (30.16)* is set to zero. **Int. Scaling: (2.29) Type: I Volatile: N**	0	10000	1800	rpm	C

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>30.17</td><td>

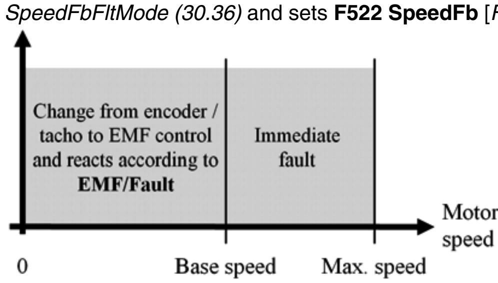
SpeedFbFltSel (speed feedback fault selector)
SpeedFbFltSel (30.17) determines the reaction to a speed feedback problem:

0 = **NotUsed** no reaction

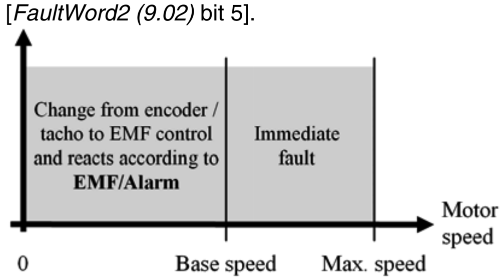
1 = **Fault** the drive trips according to *SpeedFbFltMode (30.36)* and sets **F522 SpeedFb** [*FaultWord2 (9.02)* bit 5], default



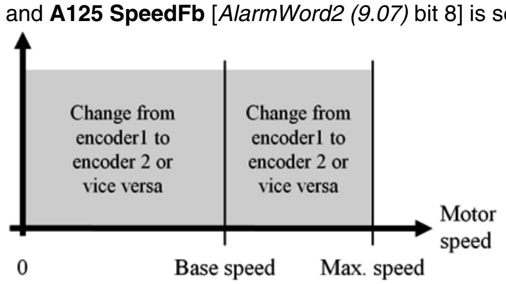
2 = **EMF/Fault** The speed feedback is switched to EMF, the drive stops according to *E StopRamp (22.11)* and sets **F522 SpeedFb** [*FaultWord2 (9.02)* bit 5]. In case speed actual is greater than base speed the drive trips according to *SpeedFbFltMode (30.36)* and sets **F522 SpeedFb** [*FaultWord2 (9.02)* bit 5].



3 = **EMF/Alarm** The speed feedback is switched to EMF and **A125 SpeedFb** [*AlarmWord2 (9.07)* bit 8] is set. In case speed actual is greater than base speed the drive trips according to *SpeedFbFltMode (30.36)* and sets **F522 SpeedFb** [*FaultWord2 (9.02)* bit 5].



4 = **Enc/Alarm** This selection is only valid if 2 pulse encoders are connected. Depending on the setting of *M1SpeeFbSel (50.03)* the speed feedback is switched from pulse encoder 1 to pulse encoder 2 or vice versa in case of a problem and **A125 SpeedFb** [*AlarmWord2 (9.07)* bit 8] is set.



Int. Scaling: 1 == 1 Type: C Volatile: N

</td><td>NotUsed</td><td>Enc/Alarm</td><td>Fault</td><td>-</td><td>E</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
30.18	**CurRippleSel (current ripple selector)** *CurRippleSel (30.18)* determines the reaction when *CurRippleLim (30.19)* is reached: 0 = **NotUsed** no reaction 1 = **Fault** the drive trips with **F517 ArmCurRipple** [*FaultWord2 (9.02)* bit 0], default 2 = **Alarm** **A117 ArmCurRipple** [*AlarmWord2 (9.07)* bit 0] is set **Note:** The current ripple function detects: – a broken fuse, thyristor or current transformer (T51, T52) – too high gain of the current controller **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	Alarm	Fault	-	E
30.19	**CurRippleLim (current ripple limit)** Threshold for *CurRippleSel (30.18),* in percent of *M1NomCur (99.03)*. Typical values when a thyristor is missing: – armature about 300 % – high inductive loads (e.g. excitation) about 90 % **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	650	150	%	E
30.20	**Unused**					
30.21	**PwrLossTrip (power loss trip)** The action taken, when the mains voltage undershoots *UNetMin2 (30.23):* 0 = **Immediately** the drive trips immediately with **F512 MainsLowVolt** [*FaultWord1 (9.01)* bit 11], default 1 = **Delayed** **A111 MainsLowVolt** [*AlarmWord1 (9.06)* bit 10] is set as long as the mains voltage recovers before *PowrDownTime (30.24)* is elapsed, otherwise **F512 MainsLowVolt** [*FaultWord1 (9.01)* bit 11] is generated **Int. Scaling: 1 == 1 Type: C Volatile: N**	Immediately	Delayed	Immediately	-	E
30.22	**UNetMin1 (mains voltage minimum 1)** First (upper) limit for mains undervoltage monitoring in percent of *NomMainsVolt (99.10).* If the mains voltage undershoots *UNetMin1 (30.22)* following actions take place: – the firing angle is set to *ArmAlphaMax (20.14)*, – single firing pulses are applied in order to extinguish the current as fast as possible, – the controllers are frozen, – the speed ramp output is updated from the measured speed and – **A111 MainsLowVolt** [*AlarmWord1 (9.06)* bit 10] is set as long as the mains voltage recovers before *PowrDownTime (30.24)* is elapsed, otherwise **F512 MainsLowVolt** [*FaultWord1 (9.01)* bit 11] is generated. **Note:** *UNetMin2 (30.23)* isn't monitored, unless the mains voltage drops below *UNetMin1 (30.22)* first. Thus for a proper function of the mains undervoltage monitoring *UNetMin1 (30.22)* has to be larger than *UNetMin2 (30.23)*. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	150	80	%	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
30.23	**UNetMin2 (mains voltage minimum 2)** Second (lower) limit for mains undervoltage monitoring in percent of *NomMainsVolt (99.10).* If the mains voltage undershoots *UnetMin2 (30.23)* following actions take place: – if *PwrLossTrip (30.21)* = **Immediately**: o the drive trips immediately with **F512 MainsLowVolt** [*FaultWord1 (9.01)* bit 11] – if *PwrLossTrip (30.21)* = **Delayed**: o field acknowledge signals are ignored, o the firing angle is set to *ArmAlphaMax (20.14)*, o single firing pulses are applied in order to extinguish the current as fast as possible, o the controllers are frozen o the speed ramp output is updated from the measured speed and o **A111 MainsLowVolt** [*AlarmWord1 (9.06)* bit 10] is set as long as the mains voltage recovers before *PowrDownTime (30.24)* is elapsed, otherwise **F512 MainsLowVolt** [*FaultWord1 (9.01)* bit 11] is generated. **Note:** *UNetMin2 (30.23)* isn't monitored, unless the mains voltage drops below *UNetMin1 (30.22)* first. Thus for a proper function of the mains undervoltage monitoring *UNetMin1 (30.22)* has to be larger than *UNetMin2 (30.23)*. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	150	60	%	C
30.24	**PowrDownTime (power down time)** The mains voltage must recover (over both limits) within *PowrDownTime (30.24).* Otherwise **F512 MainsLowVolt** [*FaultWord1 (9.01)* bit 11] will be generated. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	500	ms	C
30.25	**Unused**					
30.26	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	Overview local and communication loss:					

Device	Loss control	Time out	Related fault	Related alarm
DCS800 Control Panel	*LocalLossCtrl (30.27)*	fixed to 5 s	**F546 LocalCmdLoss**	**A130 LocalCmdLoss**
DW				
DWL				
R-type fieldbus	*ComLossCtrl (30.28)*	*FB TimeOut (30.35)*	**F528 FieldBusCom**	**A128 FieldBusCom**
DCSLink		*MailBoxCycle1 (94.13), MailBoxCycle2 (94.19), MailBoxCycle3 (94.25), MailBoxCycle4 (94.31)*	**F544 P2PandMFCom**	**A112 P2PandMFCom**
	-	*12P TimeOut (94.03)*	**F535 12PulseCom**	-
	-	*FexTimeOut (94.07)*	**F516 M1FexCom** **F519 M2FexCom**	-
SDCS-COM-8	*Ch0 ComLossCtrl (70.05)*	*Ch0 TimeOut (70.04)*	**F543 COM8Com**	**A113 COM8Com**
	Ch2 ComLossCtrl (70.15)	*Ch2 TimeOut (70.14)*		

30.27 | min.: RampStop | max.: FixedSpeed1 | def.: RampStop | unit: - | E/C: E

LocalLossCtrl (local loss control)

LocalLossCtrl (30.27) determines the reaction to a local loss (DCS800 Control Panel, DriveWindow or DriveWindow Light).

F546 LocalCmdLoss [*FaultWord3 (9.03)* bit 13] is set with:

0 = **RampStop** The input of the drives ramp is set to zero. Thus the drive stops according to *E StopRamp (22.04)*. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped.
In case *TorqSelMod (26.03)* = **Auto** and local loss is active the torque selector is bypassed and the drive is forced to speed control, default.

1 = **TorqueLimit** The output of the drives ramp is set to zero. Thus the drive stops at the active torque limit. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped.
In case *TorqSelMod (26.03)* = **Auto** and local loss is active the torque selector is bypassed and the drive is forced to speed control.

2 = **CoastStop** The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped.

3 = **DynBraking** dynamic braking

A130 LocalCmdLoss [*AlarmWord2 (9.07)* bit 13] is set with:

4 = **LastSpeed** the drive continues to run at the last speed before the warning

5 = **FixedSpeed1** the drive continuous to run with *FixedSpeed1 (23.02)*

Note:

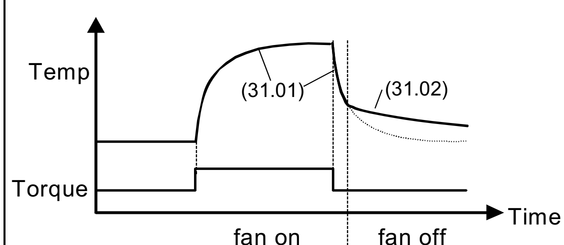
The time out for *LocalLossCtrl (30.27)* is fixed to 10 s.

Int. Scaling: 1 == 1 Type: C Volatile: N

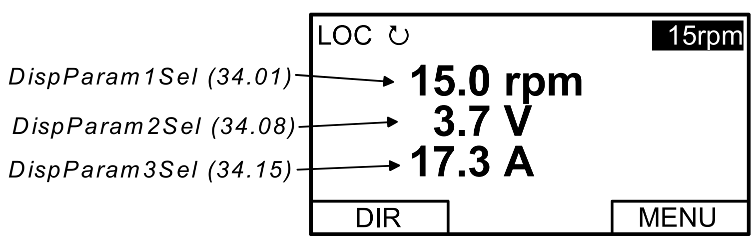
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
30.28	**ComLossCtrl (communication loss control)** *ComLossCtrl (30.28)* determines the reaction to a communication control loss (fieldbusses - R-type, DCSLink - drive-to-drive respectively master-follower) see also *CommandSel (10.01)*. Depending on the type of communication loss either **F528 FieldBusCom** [*FaultWord2 (9.02)* bit 11] or **F544 P2PandMFCom** [*FaultWord3 (9.03)* bit 11] is set with: 0 = **RampStop** The input of the drives ramp is set to zero. Thus the drive stops according to *E StopRamp (22.04)*. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. In case *TorqSelMod (26.03)* = **Auto** and communication loss is active the torque selector is bypassed and the drive is forced to speed control, default. 1 = **TorqueLimit** The output of the drives ramp is set to zero. Thus the drive stops at the active torque limit. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. In case *TorqSelMod (26.03)* = **Auto** and communication loss is active the torque selector is bypassed and the drive is forced to speed control. 2 = **CoastStop** The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. 3 = **DynBraking** dynamic braking Depending on the type of communication loss either **A128 FieldBusCom** [*AlarmWord2 (9.02)* bit 11] or **A112 P2PandMFCom** [*AlarmWord1 (9.01)* bit 11] is set with: 4 = **LastSpeed** the drive continues to run at the last speed before the warning 5 = **FixedSpeed1** the drive continuous to run with *FixedSpeed1 (23.02)* **Note:** The time out for *ComLossCtrl (30.28)* is set by: – *FB TimeOut (30.35)* for all R-type fieldbusses and – *MailBoxCycle1 (94.13)* to *MailBoxCycle4 (94.31)* for the DCSLink (drive-to-drive respectively master-follower communication). **Int. Scaling: 1 == 1 Type: C Volatile: N**	RampStop	FixedSpeed1	RampStop	-	E
30.29	**AI Mon4mA (analog input 4 mA fault selector)** *AI Mon4mA (30.29)* determines the reaction to an undershoot of one of the analog inputs under 4 mA / 2 V - if it is configured to this mode: 0 = **NotUsed** no reaction 1 = **Fault** the drive stops according to *FaultStopMode (30.30)* and trips with **F551 AIRange** [*FaultWord4 (9.04)* bit 2], default 2 = **LastSpeed** the drive continues to run at the last speed and sets **A127 AIRange** [*AlarmWord2 (9.07)* bit 10] 3 = **FixedSpeed1** the drive continues to run with *FixedSpeed1 (23.02)* and sets **A127 AIRange** [*AlarmWord2 (9.07)* bit 10] **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	FixedSpeed1	Fault	-	E

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>30.30</td><td>FaultStopMode (fault stop mode)
FaultStopMode (30.30) determines the reaction to a fault of trip level 4:
0 = RampStop The input of the drives ramp is set to zero. Thus the drive stops according to E StopRamp (22.04). When reaching M1ZeroSpeedLim (20.03) the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped.
In case TorqSelMod (26.03) = Auto and a trip of level 4 is active the torque selector is bypassed and the drive is forced to speed control, default.
1 = TorqueLimit The output of the drives ramp is set to zero. Thus the drive stops at the active torque limit. When reaching M1ZeroSpeedLim (20.03) the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped.
In case TorqSelMod (26.03) = Auto and a trip of level 4 is active the torque selector is bypassed and the drive is forced to speed control.
2 = CoastStop The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped.
3 = DynBraking dynamic braking
Note:
FaultStopMode (30.30) doesn’t apply to communication faults.
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>RampStop</td><td>DynBraking</td><td>RampStop</td><td>-</td><td>C</td></tr>
<tr><td>30.31</td><td>ExtFaultSel (external fault selector)
The drive trips with F526 ExternalDI [FaultWord2 (9.02) bit 9] if a binary input for an external fault is selected and 1:
0 = NotUsed no reaction, default
1 = DI1 1 = fault, 0 = no fault
2 = DI2 1 = fault, 0 = no fault
3 = DI3 1 = fault, 0 = no fault
4 = DI4 1 = fault, 0 = no fault
5 = DI5 1 = fault, 0 = no fault
6 = DI6 1 = fault, 0 = no fault
7 = DI7 1 = fault, 0 = no fault
8 = DI8 1 = fault, 0 = no fault
9 = DI9 1 = fault, 0 = no fault, Only available with digital extension board
10 = DI10 1 = fault, 0 = no fault, Only available with digital extension board
11 = DI11 1 = fault, 0 = no fault, Only available with digital extension board
12 = MCW Bit11 1 = fault, 0 = no fault, MainCtrlWord (7.01) bit 11
13 = MCW Bit12 1 = fault, 0 = no fault, MainCtrlWord (7.01) bit 12
14 = MCW Bit13 1 = fault, 0 = no fault, MainCtrlWord (7.01) bit 13
15 = MCW Bit14 1 = fault, 0 = no fault, MainCtrlWord (7.01) bit 14
16 = MCW Bit15 1 = fault, 0 = no fault, MainCtrlWord (7.01) bit 15
17 = ACW Bit12 1 = fault, 0 = no fault, AuxCtrlWord (7.02) bit 12
18 = ACW Bit13 1 = fault, 0 = no fault; AuxCtrlWord (7.02) bit 13
19 = ACW Bit14 1 = fault, 0 = no fault, AuxCtrlWord (7.02) bit 14
20 = ACW Bit15 1 = fault, 0 = no fault, AuxCtrlWord (7.02) bit 15
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>NotUsed</td><td>ACW Bit15</td><td>NotUsed</td><td>-</td><td>C</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
30.32	**ExtAlarmSel (external alarm selector)** The drive sets **A126 ExternalDI** [*AlarmWord2 (9.07)* bit 9] if a binary input for an external alarm is selected and 1: 0 = **NotUsed** no reaction, default 1 = **DI1** 1 = fault, 0 = no fault 2 = **DI2** 1 = fault, 0 = no fault 3 = **DI3** 1 = fault, 0 = no fault 4 = **DI4** 1 = fault, 0 = no fault 5 = **DI5** 1 = fault, 0 = no fault 6 = **DI6** 1 = fault, 0 = no fault 7 = **DI7** 1 = fault, 0 = no fault 8 = **DI8** 1 = fault, 0 = no fault 9 = **DI9** 1 = fault, 0 = no fault. Only available with digital extension board 10 = **DI10** 1 = fault, 0 = no fault. Only available with digital extension board 11 = **DI11** 1 = fault, 0 = no fault. Only available with digital extension board 12 = **MCW Bit11** 1 = fault, 0 = no fault, *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** 1 = fault, 0 = no fault, *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** 1 = fault, 0 = no fault, *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** 1 = fault, 0 = no fault, *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** 1 = fault, 0 = no fault, *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** 1 = fault, 0 = no fault, *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** 1 = fault, 0 = no fault, *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** 1 = fault, 0 = no fault, *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** 1 = fault, 0 = no fault, *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	C
30.33	**ExtFaultOnSel (external fault on selector)** *ExtFaultOnSel (30.33)* determines the reaction to an external fault: 0 = **Fault** external fault is always valid independent from drive state, default 1 = **Fault&RdyRun** external fault is only valid when drive state is **RdyRun** [*MainStatWord (8.01)* bit 1] for at least 6 s **Int. Scaling: 1 == 1 Type: C Volatile: N**	Fault	Fault&RdyR	Fault	-	E
30.34	**ExtAlarmOnSel (external alarm on selector)** *ExtAlarmOnSel (30.34)* determines the reaction to an external alarm: 0 = **Alarm** external alarm is always valid independent from drive state, default 1 = **Alarm&RdyRun** external alarm is only valid when drive state is **RdyRun** [*MainStatWord (8.01)* bit 1] for at least 6 s **Int. Scaling: 1 == 1 Type: C Volatile: N**	Alarm	Alarm&Rdy	Alarm	-	E
30.35	**FB TimeOut (fieldbus time out)** Time delay before a communication break with a fieldbus is declared. Depending on the setting of *ComLossCtrl (30.28)* either **F528 FieldBusCom** [*FaultWord2 (9.02)* bit 11] or **A128 FieldBusCom** [*AlarmWord2 (9.07)* bit 11] is set. The communication fault and alarm are inactive, if *FB TimeOut (30.35)* is set to 0 ms. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	100	ms	C
30.36	**SpeedFbFltMode (speed feedback fault mode)** *SpeedFbFltMode (30.36)* determines the reaction to a fault of trip level 3: 0 = **CoastStop** The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. 1 = **DynBraking** dynamic braking **Note:** *SpeedFbFltMode (30.36)* doesn't apply to communication faults. **Int. Scaling: 1 == 1 Type: C Volatile: N**	CoastStop	DynBraking	CoastStop	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 31	**Motor 1 temperature**					
31.01	**M1ModelTime (motor 1 model time constant)** Thermal time constant for motor 1 with fan/forced cooling. The time within the temperature rises to 63% of its nominal value. The motor thermal model is blocked, if *M1ModelTime (31.01)* is set to zero. The value of *Mot1TempCalc (1.20)* is saved at power down of the drives electronics. With the very first energizing of the drives electronics the motor's ambient temperature is set to 30°C. **WARNING!** The model does not protect the motor if it is not properly cooled e.g. due to dust and dirt. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	6400	240	s	E
31.02	**M1ModelTime2 (motor 1 model time 2 constant)** Thermal time constant for motor 1 with fan/forced cooling if motor fan is switched off.  **Attention:** For motors without fan set *M1ModelTime (31.01) = M1ModelTime2 (31.02).* **Int. Scaling: 10 == 1 % Type: I Volatile: N**	0	6400	2400	s	E
31.03	**M1AlarmLimLoad (motor 1 alarm limit load)** The drive sets **A107 M1OverLoad** [*AlarmWord1 (9.06)* bit 6] if *M1AlarmLimLoad (31.03)* - in percent of *M1NomCur (99.03)* - is exceeded. Output value for motor 1 thermal model is *Mot1TempCalc (1.20).* **Int. Scaling: 10 == 1 % Type: I Volatile: N**	10	325	102	%	E
31.04	**M1FaultLimLoad (motor 1 fault limit load)** The drive trips with **F507 M1OverLoad** [*FaultWord1 (9.01)* bit 6] if *M1FaultLimLoad (31.04)* - in percent of *M1NomCur (99.03)* - is exceeded. Output value for motor 1 thermal model is *Mot1TempCalc (1.20).* **Int. Scaling: 10 == 1 % Type: I Volatile: N**	10	325	106	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
31.05	**M1TempSel (motor 1 temperature selector)** *M1TempSel (31.05)* selects motor 1 measured temperature input. The result can be seen in *Mot1TemopMeas (1.22)*. Connection possibilities for PT100: – max. 3 PT100 for motor 1 and max. 3 PT100 for motor 2 or – up to 6 PT100 for motor 1 only. Connection possibilities PTC: – max. 1 PTC for motor 1 and max. 1 PTC for motor 2 or – up to 2 PTC for motor 1 only: 0 = **NotUsed** motor 1 temperature measurement is blocked, default 1 = **1PT100 AI2** one PT100 connected to AI2 on SDCS-IOB-3 2 = **2PT100 AI2** two PT100 connected to AI2 on SDCS-IOB-3 3 = **3PT100 AI2** three PT100 connected to AI2 on SDCS-IOB-3 4 = **4PT100 AI2/3** four PT100, 3 connected to AI2 and 1 connected to AI3 on SDCS-IOB-3 5 = **5PT100 AI2/3** five PT100, 3 connected to AI2 and 2 connected to AI3 on SDCS-IOB-3 6 = **6PT100 AI2/3** six PT100, 3 connected to AI2 and 3 connected to AI3 on SDCS-IOB-3 7 = **1PT100 AI7** one PT100 connected to AI7 on second RAIO 8 = **2PT100 AI7** two PT100 connected to AI7 on second RAIO 9 = **3PT100 AI7** three PT100 connected to AI7 on second RAIO 10 = **4PT100 AI7/8** four PT100, 3 connected to AI7 and 1 connected to AI8 on second RAIO 11 = **5PT100 AI7/8** five PT100, 3 connected to AI7 and 2 connected to AI8 on second RAIO 12 = **6PT100 AI7/8** six PT100, 3 connected to AI7 and 3 connected to AI8 on second RAIO 13 = **1PTC AI2** one PTC connected to AI2 on SDCS-IOB-3 14 = **2PTC AI2/3** two PTC, 1 connected to AI2 and 1 connected to AI3 on SDCS-IOB-3 15 = **1PTC AI2/Con** one PTC connected to AI2 on SDCS-CON-4 For more information see section *Motor protection*. **Note:** AI7 and AI8 have to be activated by means of *AIO ExtModule (98.06).* **Note:** In case only one PT100 is connected to an AI of the SDCS-IOB-3 the input range must be configured by jumpers to a gain of 10. Jumper settings for input range and constant current source see *DCS800 Hardware Manual*. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	1PTC AI2/Con	NotUsed	-	C
31.06	**M1AlarmLimTemp (motor 1 alarm limit temperature)** The drive sets **A106 M1OverTemp** [*AlarmWord1 (9.06)* bit 5] if *M1AlarmLimTemp (31.06)* is exceeded. Output value for motor 1 measured temperature is *Mot1TempMeas (1.22).* **Note:** The unit depends on *M1TempSel (31.05).* **Int. Scaling: 1 == 1 °C / 1 Ω / 1 Type: SI Volatile: N**	-10	4000	0	°C / Ω / -	C
31.07	**M1FaultLimTemp (motor 1 fault limit temperature)** The drive trips with **F506 M1OverTemp** [*FaultWord1 (9.01)* bit 5] if *M1FaultLimTemp (31.07)* is exceeded. Output value for motor 1 measured temperature is *Mot1TempMeas (1.22).* **Note:** The unit depends on *M1TempSel (31.05).* **Int. Scaling: 1 == 1 °C / 1 Ω / 1 Type: SI Volatile: N**	-10	4000	0	°C / Ω / -	C

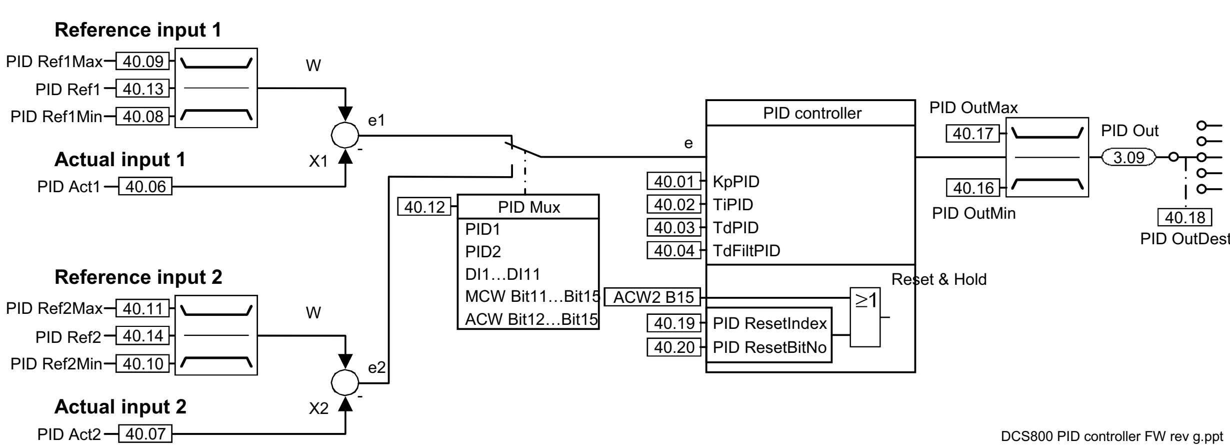
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
31.08	**M1KlixonSel (motor 1 klixon selector)** The drive trips with **F506 M1OverTemp** [*FaultWord1 (9.01)* bit 5] if a digital input selected and the klixon is open: 0 = **NotUsed** no reaction, default 1 = **DI1** 0 = fault, 1 = no fault 2 = **DI2** 0 = fault, 1 = no fault 3 = **DI3** 0 = fault, 1 = no fault 4 = **DI4** 0 = fault, 1 = no fault 5 = **DI5** 0 = fault, 1 = no fault 6 = **DI6** 0 = fault, 1 = no fault 7 = **DI7** 0 = fault, 1 = no fault 8 = **DI8** 0 = fault, 1 = no fault 9 = **DI9** 0 = fault, 1 = no fault. Only available with digital extension board 10 = **DI10** 0 = fault, 1 = no fault. Only available with digital extension board 11 = **DI11** 0 = fault, 1 = no fault. Only available with digital extension board **Note:** It is possible to connect several klixons in series. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI11	NotUsed	-	C
Group 34	**DCS800 Control Panel display**					
	Signal and parameter visualization on the DCS800 Control Panel:  Setting a display parameter to 0 results in no signal or parameter displayed. Setting a display parameter from 101 to 9999 displays the belonging signal or parameter. If a signal or parameter does not exist, the display shows “n.a.”.					
34.01	**DispParam1Sel (select signal / parameter to be displayed in the DCS800 Control Panel row 1)** Index pointer to the source of the DCS800 Control Panel first display row [e.g. 101 equals *MotSpeedFilt (1.01)*]. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	101	-	C
34.02	**Unused**					
34.03	**Unused**					
34.04	**Unused**					
34.05	**Unused**					
34.06	**Unused**					
34.07	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
34.08	**DispParam2Sel (select signal / parameter to be displayed in the DCS800 Control Panel row 2)** Index pointer to the source of the DCS800 Control Panel second display row [e.g. 114 equals *ArmVoltAct (1.14)*]. **Int. Scaling: 1 == 1 Type: I Volatile: N**					
34.09	**Unused**					
34.10	**Unused**					
34.11	**Unused**					
34.12	**Unused**					
34.13	**Unused**					
34.14	**Unused**					
34.15	**DispParam3Sel (select signal / parameter to be displayed in the DCS800 Control Panel l row 3)** *ConvCurAct (1.16)* .					
34.16	**Unused**					
34.17	**Unused**					
34.18	**Unused**					
34.19	**Unused**					
34.20	**Unused**					
34.21	**Unused**					

Group 40

PID control

Overview of the PID controller:



Index	Signal / Parameter name	min.	max.	def.	unit	E/C
40.01	**KpPID (p-part PID controller)** Proportional gain of the PID controller. Example: The controller generates 15 % output with *KpPID (40.01)* = 3, if the input is 5 %. **Int. Scaling: 100 == 1 Type: I Volatile: N**	0	325	5	-	E
40.02	**TiPID (i-part PID controller)** Integral time of the PID controller. *TiPID (40.02)* defines the time within the integral part of the controller achieves the same value as the proportional part. Example: The controller generates 15 % output with *KpPID (40.01)* = 3, if the input is 5 %. On that condition and with *TiPID (40.02)* = 300 ms follows: – the controller generates 30 % output, if the input is constant, after 300 ms are elapsed (15 % from proportional part and 15 % from integral part). **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	2500	ms	E
40.03	**TdPID (d-part PID controller)** PID controller derivation time. *TdPID (40.03)* defines the time within the PID controller derives the error value. The PID controller works as PI controller, if *TdPID (40.03)* is set to zero. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	0	ms	E
40.04	**TdFiltPID (filter time for d-part PID controller)** Derivation filter time. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	10	ms	E
40.05	**Unused**					
40.06	**PID Act1 (PID controller actual input value 1 index)** Index pointer to the source of the PID controller actual input value 1. The format is **-xxyy**, with: **-** = negate actual input value 1, **xx** = group and **yy** = index [e.g. 101 equals *MotSpeedFilt (1.01)*]. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	E
40.07	**PID Act2 (PID controller actual input value 2 index)** Index pointer to the source of the PID controller actual input value 2. The format is **-xxyy**, with: **-** = negate actual input value 2, **xx** = group and **yy** = index [e.g. 101 equals *MotSpeedFilt (1.01)*]. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	E
40.08	**PID Ref1Min (PID controller minimum limit reference input value 1)** Minimum limit of the PID controller reference input value 1 in percent of the source of *PID Ref1 (40.13)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	0	-100	%	E
40.09	**PID Ref1Max (PID controller maximum limit reference input value 1)** Maximum limit of the PID controller reference input value 1 in percent of the source of *PID Ref1 (40.13)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0	325	100	%	E
40.10	**PID Ref2Min (PID controller minimum limit reference input value 2)** Minimum limit of the PID controller reference input value 2 in percent of the source of *PID Ref2 (40.14)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	0	-100	%	E
40.11	**PID Ref2Max (PID controller maximum limit reference input value 2)** Maximum limit of the PID controller reference input value 2 in percent of the source of *PID Ref2 (40.14)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0	325	100	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
40.12	**PID Mux (PID controller reference input selector/multiplexer)** PID controller reference input selector: 0 = **PID1** reference input 1 is selected, default 1 = **PID2** reference input 2 is selected 2 = **DI1** 1= reference input 2 is selected; 0 = reference input 1 is selected 3 = **DI2** 1= reference input 2 is selected; 0 = reference input 1 is selected 4 = **DI3** 1= reference input 2 is selected; 0 = reference input 1 is selected 5 = **DI4** 1= reference input 2 is selected; 0 = reference input 1 is selected 6 = **DI5** 1= reference input 2 is selected; 0 = reference input 1 is selected 7 = **DI6** 1= reference input 2 is selected; 0 = reference input 1 is selected 8 = **DI7** 1= reference input 2 is selected; 0 = reference input 1 is selected 9 = **DI8** 1= reference input 2 is selected; 0 = reference input 1 is selected 10 = **DI9** 1= reference input 2 is selected; 0 = reference input 1 is selected; only available with digital extension board 11= **DI10** 1= reference input 2 is selected; 0 = reference input 1 is selected; only available with digital extension board 12 = **DI11** 1= reference input 2 is selected; 0 = reference input 1 is selected; only available with digital extension board 13 = **MCW Bit11** 1= reference input 2 is selected; 0 = reference input 1 is selected; *MainCtrlWord (7.01)* bit 11 14 = **MCW Bit12** 1= reference input 2 is selected; 0 = reference input 1 is selected; *MainCtrlWord (7.01)* bit 12 15 = **MCW Bit13** 1= reference input 2 is selected; 0 = reference input 1 is selected; *MainCtrlWord (7.01)* bit 13 16 = **MCW Bit14** 1= reference input 2 is selected; 0 = reference input 1 is selected; *MainCtrlWord (7.01)* bit 14 17 = **MCW Bit15** 1= reference input 2 is selected; 0 = reference input 1 is selected; *MainCtrlWord (7.01)* bit 15 18 = **ACW Bit12** 1= reference input 2 is selected; 0 = reference input 1 is selected; *AuxCtrlWord (7.02)* bit 12 19 = **ACW Bit13** 1= reference input 2 is selected; 0 = reference input 1 is selected; *AuxCtrlWord (7.02)* bit 13 20 = **ACW Bit14** 1= reference input 2 is selected; 0 = reference input 1 is selected; *AuxCtrlWord (7.02)* bit 14 21 = **ACW Bit15** 1= reference input 2 is selected; 0 = reference input 1 is selected; *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	PID1	ACW Bit15	PID1		E
40.13	**PID Ref1 (PID controller reference input value 1 index)** Index pointer to the source of the PID controller reference input value 1. The format is **-xxyy**, with: **-** = negate reference input value 1, **xx** = group and **yy** = index [e.g. 201 equals *SpeedRef2 (2.01)*]. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	E
40.14	**PID Ref2 (PID controller reference input value 2 index)** Index pointer to the source of the PID controller reference input value 2. The format is **-xxyy**, with: **-** = negate reference input value 2, **xx** = group and **yy** = index [e.g. 201 equals *SpeedRef2 (2.01)*]. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	E
40.15	**Unused**					
40.16	**PID OutMin (PID controller minimum limit output value)** Minimum limit of the PID controller output value in percent of the used PID controller input. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	0	-100	%	E
40.17	**PID OutMax (PID controller maximum limit output value)** Maximum limit of the PID controller output value in percent of the used PID controller input. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0	325	100	%	E

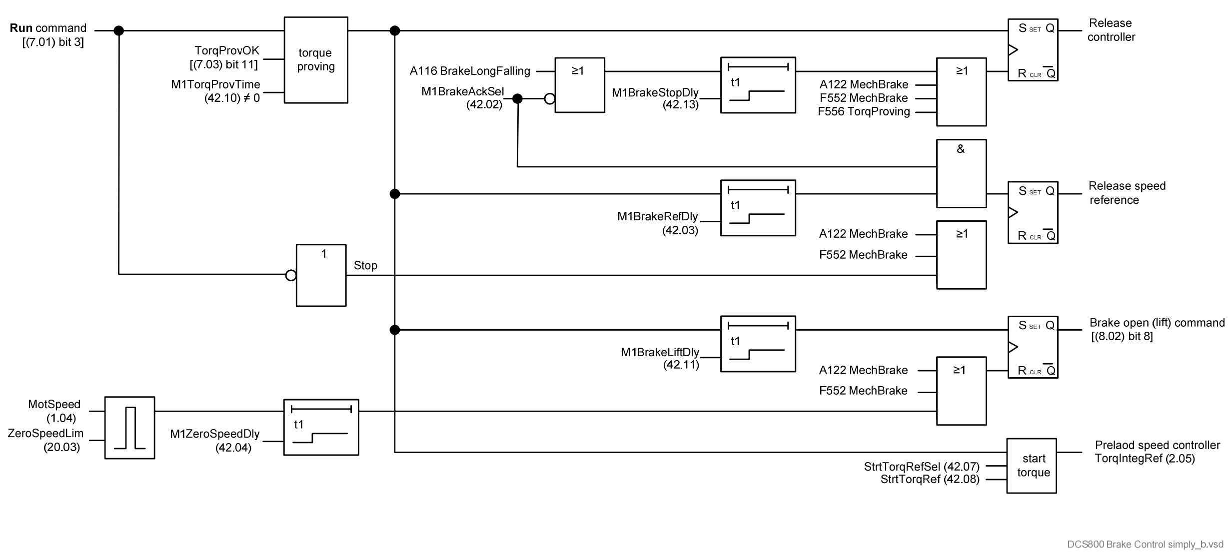
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
40.18	**PID OutDest (PID controller output value index)** Index pointer to the sink of the PID controller output value. The format is **-xxyy**, with: **-** = negate output value, **xx** = group and **yy** = index [e.g. 2301 equals *SpeedRef (23.01)*]. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	E
40.19	**PID ResetIndex (PID controller reset index)** The PID controller reset is controlled by a selectable bit - see *PID ResetBitNo (40.20)* - of the source (signal/parameter) selected with this parameter. The format is **-xxyy**, with: **-** = invert reset signal, **xx** = group and **yy** = index. Examples: – If *PID ResetIndex (40.19)* = 701 (main control word) and *PID ResetBitNo (40.20)* = 12 then the PID controller reset is active when bit 12 is high. – If *PID ResetIndex (40.19)* = -701 (main control word) and *PID ResetBitNo (40.20)* = 12 then the PID controller reset is active when bit 12 is low. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-9999	9999	0	-	E
40.20	**PID ResetBitNo (PID controller reset bit number)** Bit number of the signal/parameter selected with *PID ResetIndex (40.19)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	0	-	E
40.21	**PID Reserved (PID reserved)** reserved **Int. Scaling: 1 == 1 Type: I Volatile: N**	-	-	-	-	E

Group 42

Brake control

Brake Control is activated by means of *M1BrakeCtrl (42.01)* and controls a mechanical brake automatically with the **Run** [*MainCtrlWord (7.01)* bit 3] command. The internal logic is designed to meet the requirements of holding brakes, e.g. carriage drives or coilers, as well as the requirements for hanging load, e.g. cranes.

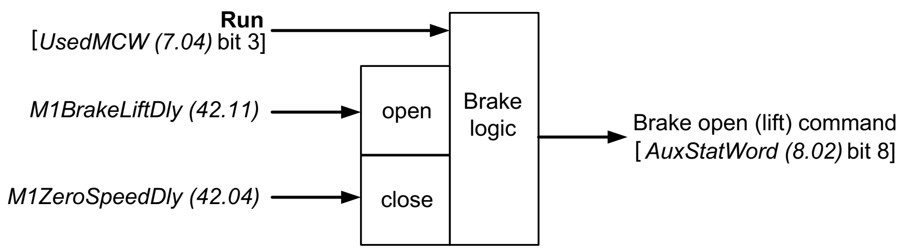
Overview brake control



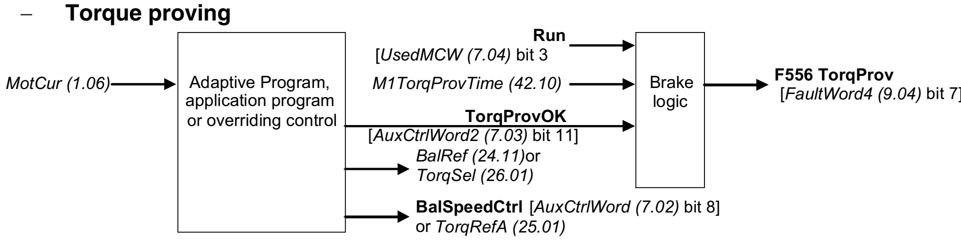
Index	Signal / Parameter name	min.	max.	def.	unit	E/C

Following functions are included:

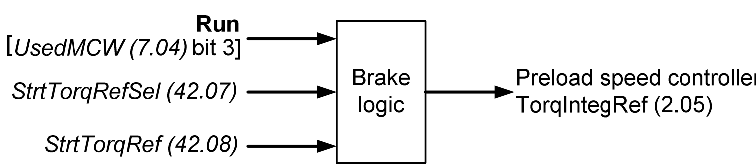
– **Mechanical open (lift), close (apply) and zero speed delays**



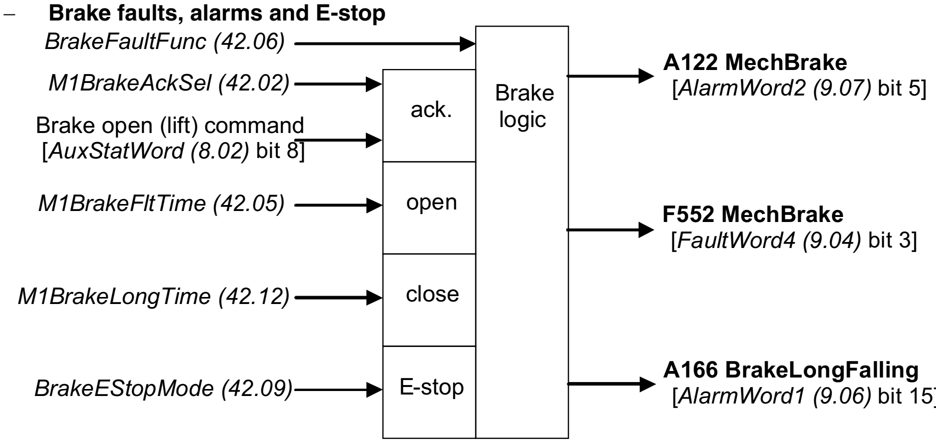
– **Torque proving**



– **Adjustable start torque**



– **Brake faults, alarms and E-stop**



Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	All speed references have to be routed via the speed ramp. With brake control **On** [*M1BrakeCtrl (42.01)*] and **RdyRef** [*MainStatWord (8.01)* bit 2] = 1 the torque proving is done, if selected. Afterwards the torque reference is set to *StrtTorqRef (42.08)* and the brake open (lift) command is given. The brake open (lift) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] is send delayed by *M1BrakeLiftDly (42.11)* to the brake. Then *M1BrakeLiftDly (42.11)* and *M1BrakeRefDly (42.03)* are started at the same time. During *M1BrakeRefDly (42.03)* the speed ramp is clamped to zero and the torque reference equals *StrtTorqRef (42.08)*. After *M1BrakeRefDly (42.03)* is elapsed and the brake acknowledge - if selected with *M1BrakeAckSel (42.02)* - is active, clamp of speed reference is removed. This function compensates for the mechanical open (lift) delay of the brake.					
	With **Run** [*UsedMCW (7.04)* bit 3] = 0 and motor speed below *M1ZeroSpeedLim (20.03)*, *M1ZeroSpeedDly (42.04)* starts to compensate for the time the drive needs to decelerate from *M1ZeroSpeedLim (20.03)* to actual speed = 0. Until *M1ZeroSpeedDly (42.04)* is elapsed the brake is kept open (lifted). After *M1ZeroSpeedDly (42.04)* is elapsed, the brake open (lift) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] is removed and the brake close (apply) delay *M1BrakeStopDelay (42.13)* is started. During *M1BrakeStopDelay (42.13)* the motor control remains active with speed reference set to zero and the speed controller stays alive. This function compensates for the mechanical close (apply) delay of the brake.					
	The brake can be forced by **ForceBrake** [*AuxCtrlWord2 (7.03)* bit 12] **ForceBrake** = 1 — If **ForceBrake** is set the brake remains closed (applied). If the **Run** [*MainCtrlWord (7.01)* bit 3] command is given to a drive in state **RdyOn** or **RdyRef** [*MainStatWord (8.01)* bit 0 and 1], the brake logic will be started up to the point of the brake open (lift) command. A drive in state **Running** [*MainStatWord (8.01)* bit 2] will be stopped by ramp, the brake will be closed (applied), but the drive will remain in state **Running**. **ForceBrake** = 0 — The brake is controlled by the internal brake logic in group 42 (Brake control).					
42.01	**M1BrakeCtrl (motor 1 brake control)** Releases the control of motor 1 brake: 0 = **NotUsed** — brake logic is blocked, default 1 = **On** — brake logic is released according to it's parameter settings 2 = **BrakeClose** — test mode, the brake logic will work, but the brake is always closed (applied) 3 = **BrakeOpen** — test mode, the brake logic will work, but the brake is always opened (lifted) **Attention**: A closed (applied) brake will open (lift) immediately! Do **not** use this mode with e.g. an unsaved crane drive! The brake open (lift) command **BrakeCmd** is readable in *AuxStatWord (8.02)* bit 8 and can be connected to the digital output controlling the brake. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	BrakeOpen	NotUsed	-	E

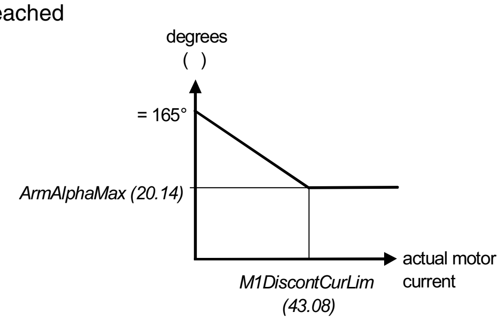
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
42.02	**M1BrakeAckSel (motor 1 brake acknowledge selector)** The drive sets either **A122 MechBrake** [*AlarmWord2 (9.07)* bit 5], **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] or **A116 BrakeLongFalling** [*AlarmWord1 (9.06)* bit 15] depending on *BrakeFaultFunc (42.06)* if a digital input is selected and the brake acknowledge fails: 0 = **NotUsed** brake acknowledge is blocked, default 1 = **DI1** 0 = brake is closed (applied), 1 = brake is open (lifted) 2 = **DI2** 0 = brake is closed (applied), 1 = brake is open (lifted) 3 = **DI3** 0 = brake is closed (applied), 1 = brake is open (lifted) 4 = **DI4** 0 = brake is closed (applied), 1 = brake is open (lifted) 5 = **DI5** 0 = brake is closed (applied), 1 = brake is open (lifted) 6 = **DI6** 0 = brake is closed (applied), 1 = brake is open (lifted) 7 = **DI7** 0 = brake is closed (applied), 1 = brake is open (lifted) 8 = **DI8** 0 = brake is closed (applied), 1 = brake is open (lifted) 9 = **DI9** 0 = brake is closed (applied), 1 = brake is open (lifted), only available with digital extension board 10 = **DI10** 0 = brake is closed (applied), 1 = brake is open (lifted), only available with digital extension board 11 = **DI11** 0 = brake is closed (applied), 1 = brake is open (lifted), only available with digital extension board 12 = **MCW Bit11** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 11* 13 = **MCW Bit12** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 12* 14 = **MCW Bit13** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 13* 15 = **MCW Bit14** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 14* 16 = **MCW Bit15** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 15* 17 = **ACW Bit12** 0 = brake is closed (applied), 1 = brake is open (lifted), *AuxCtrlWord (7.02) bit 12* 18 = **ACW Bit13** 0 = brake is closed (applied), 1 = brake is open (lifted), *AuxCtrlWord (7.02) bit 13* 19 = **ACW Bit14** 0 = brake is closed (applied), 1 = brake is open (lifted), *AuxCtrlWord (7.02) bit 14* 20 = **ACW Bit15** 0 = brake is closed (applied), 1 = brake is open (lifted), *AuxCtrlWord (7.02) bit 15* **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	E
42.03	**M1BrakeRefDly (motor 1 brake speed reference delay)** Speed reference delay. This function compensates for the mechanical open (lift) delay of the brake. During the start - **Run** [*MainCtrlWord (7.01)* bit 3] = 1 - of the drive the speed reference is clamped (ramp output is set to zero) and the speed controller output is set to start torque [see *M1StrtTorqRefSel (42.07)*] until *M1BrakeRefDly (42.03)* is elapsed. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	0.1	s	E
42.04	**M1ZeroSpeedDly (motor 1 zero speed delay)** This function compensates for the time the drive needs to decelerate from *M1ZeroSpeedLim (20.03)* to actual speed = 0. Until *M1ZeroSpeedDly (42.04)* is elapsed the brake is kept open (lifted). **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	0	s	E
42.05	**M1BrakeFltTime (motor 1 brake fault time)** Brake open (lift) acknowledge monitor. During this time the brake open (lift) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] and the brake acknowledge signal [*M1BrakeAckSel (42.02)*] can be different without causing **A122 MechBrake** [*AlarmWord2 (9.07)* bit 5] or **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] depending on *BrakeFaultFunc (42.06)*. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	1	s	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
42.06	**BrakeFaultFunc (brake fault function)** Selected motor, *BrakeFaultFunc (42.06)* determines the reaction to an invalid brake acknowledge: 0 = **Alarm** the drive sets **A122 MechBrake** [*AlarmWord2 (9.07)* bit 5] as reaction to an invalid brake open (lift) or brake close (apply) acknowledge 1 = **Fault** the drive trips with **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] as reaction to an invalid brake open (lift) or brake close (apply) acknowledge, default 3 = **Crane** The drive trips with **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] as reaction to an invalid brake open (lift) acknowledge. **A116 BrakeLongFalling** [*AlarmWord1 (9.06)* bit 15] is set as reaction to an invalid brake close (apply) acknowledge. In case of **A116 BrakeLongFalling** [*AlarmWord1 (9.06)* bit 15] the speed reference is set to zero and the speed controller is kept active until the drive is stopped by either **On** = 0 [*UsedMCW (7.04)* bit 0] or **Off2N** = 0 [*UsedMCW (7.04)* bit 1, Emergency Off / Coast Stop]. **Note:** If the brake open (lift) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] and the brake acknowledge signal [*M1BrakeAckSel (42.02)*] are different for a longer time than set in *M1BrakeFltTime (42.05)* either **A122 MechBrake** [*AlarmWord2 (9.07)* bit 5] or **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] is set depending on *BrakeFaultFunc (42.06)*. **Note:** If the brake close (apply) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] and the brake acknowledge signal [*M1BrakeAckSel (42.02)*] are different for a longer time than set in *M1BrakeLongTime (42.12)* either **A122 MechBrake** [*AlarmWord2 (9.07)* bit 5], **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] or **A116 BrakeLongFalling** [*AlarmWord1 (9.06)* bit 15] is set depending on *BrakeFaultFunc (42.06)*. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Alarm	Crane	Fault	-	E
42.07	**M1StrtTorqRefSel (motor 1 start torque reference selector)** Motor 1, start torque selector: 0 = **NotUsed** start torque function is blocked and the start torque reference is fixed zero, default 1 = **Memory** Torque memory released. The minimum value equals the absolute value of *StrtTorqRef (42.08)*. The torque memory can be reset by means of *AuxCtrlWord2 (7.03)* bit 13. 2 = **StrtTorqRef** *StrtTorqRef (42.08)* 3 = **AI1** analog input AI1 4 = **AI2** analog input AI2 5 = **AI3** analog input AI3 6 = **AI4** analog input AI4 7 = **AI5** analog input AI5 8 = **AI6** analog input AI6 **Note:** Torque memory is the presetting of the torque when starting with e.g. suspended load. The preset torque equals the actual torque stored when the brake open (lift) command is removed, if the stored torque is greater than the value in *StrtTorqRef (42.08)*. Otherwise the value in *StrtTorqRef (42.08)* is taken. After energizing the drive the value of *StrtTorqRef (42.08)* is set as torque memory. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	AI6	NotUsed	-	E
42.08	**StrtTorqRef (start torque reference)** Selected motor, start torque reference in percent of *MotNomTorque (4.23)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	325	100	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
42.09	**BrakeEStopMode (emergency stop mode brake)** Selected motor, *BrakeEStopMode (42.09)* determines the reaction when *UsedMCW (7.04)* bit 2 **Off3N** (respectively E-stop) is set low: 0 = **Disable** the brake is closed (applied) according to the standard brake control, default 1 = **Enable** the brake is closed (applied) immediately together with the E-stop command **Note:** If *BrakeEStopMode (42.09)* = **Enable** the *E StopRamp (22.04)* should be shorter than the time needed to stop the motor with the mechanical brake applied only. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Disable	Enable	Disable	-	E
42.10	**M1TorqProvTime (motor 1 torque proving time)** Brake torque proving acknowledge. The drive trips with **F556 TorqProv** [*FaultWord4 (9.04)* bit 7] if the **Run** [*MainCtrlWord (7.01)* bit 3] command is set and the acknowledge **TorqProvOK** [*AuxCtrlWord2 (7.03)* bit 11] is not set before *M1TorqProvTime (42.10)* is elapsed. The torque proving is inactive, if *M1TorqProvTime (42.10)* is set to 0. **Note:** The acknowledge signal **TorqProvOK** has to be provided by Adaptive Program, application program or overriding control and is set by means of a rising edge (0 → 1). The torque reference might be set by means of *BalRef (24.11)* or *TorqSel (26.01)* and **BalSpeedCtrl** [*AuxCtrlWord (7.02)* bit 8] or *TorqRefA (25.01)*. The reaction of the drive might be taken from *MotCur (1.06)*. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	100	0	s	E
42.11	**M1BrakeLiftDly (motor 1 brake lift delay)** Brake open (lift) delay. This function delays the brake open (lift) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] until *M1BrakeLiftDly (42.11)* is elapsed. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	0	s	E
42.12	**M1BrakeLongTime (motor 1 brake long time)** Brake close (apply) acknowledge monitor. During this time the brake close (apply) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] and the brake acknowledge signal [*M1BrakeAckSel (42.02)*] can be different without causing either **A122 MechBrake** [*AlarmWord2 (9.07)* bit 5], **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] or **A116 BrakeLongFalling** [*AlarmWord1 (9.06)* bit 15] depending on *BrakeFaultFunc (42.06)*. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	4	s	E
42.13	**M1BrakeStopDly (motor 1 brake stop delay)** Brake close (apply) delay. This function starts after the brake acknowledge - if selected with *M1BrakeAckSel (42.02)* - is zero and compensates for the mechanical close (apply) delay of the brake. During the stop - **Run** [*MainCtrlWord (7.01)* bit 3] = 0 - of the drive the speed reference is clamped (ramp output is set to zero) and the speed controller stays active until *M1BrakeStopDly (42.13)* is elapsed. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	1	s	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 43	**Current control**					
43.01	**OperModeSel (operation mode selector)** Converter mode selection: 0 = **ArmConv** 6 pulse single armature converter, default 1 = **FieldConv** field exciter mode; **Attention:** The digital input for the external overvoltage protection is assigned by means of *OvrVoltProt (10.13).* 2 = **12PParMaster** 12-pulse parallel master 3 = **12PParSlave** 12-pulse parallel slave 4 = **12PSerMaster** 12-pulse serial master 5 = **12PSerSlave** 12-pulse serial slave This parameter is write protected while **Run** [*UsedMCW (7.04)* bit 3] = 1. **Int. Scaling: 1 == 1 Type: C Volatile: N**	ArmConv	12PSerSlave	ArmConv	-	E
43.02	**CurSel (current reference selector)** *CurSel (43.02)* selector: 0 = **CurRef311** *CurRef (3.11)* calculated from torque reference as armature current reference, default 1 = **CurRefExt** *CurRefExt (43.03)* as armature current reference 2 = **AI1** analog input AI1 as armature current reference 3 = **AI2** analog input AI2 as armature current reference 4 = **AI3** analog input AI3 as armature current reference 5 = **AI4** analog input AI4 as armature current reference 6 = **AI5** analog input AI5 as armature current reference 7 = **AI6** analog input AI6 as armature current reference 8 = **FexCurRef** *FldCurRefM1 (3.30)* from armature converter via DCSLink as field current reference, only available if *OperModeSel (43.01)* = **FieldConv** 9 = **FluxRefEMF** *FluxRefEMF (3.27)* from armature converter as field current reference, only if available *OperModeSel (43.01)* = **FieldConv** 10 = **TorqRef213** *TorqRefUsed (2.13)* is directly used as armature current reference (torque = current); **Note:** The flux adaption in field weakening is inactive (means no flux dependent armature current reference) 11 = **FexCur+Ext** *FldCurRefM1 (3.30)* from armature converter via DCSLink plus *CurRefExt (43.03)* as field current reference, only available if *OperModeSel (43.01)* = **FieldConv** 12 = **CurZero** forces single firing pulses and sets *CurRefUsed (3.11)* to zero **Note:** In case *OperModeSel (43.01)* is **12PParSlave** *CurSel (43.02)* is overwritten by the current reference from the 12-pulse parallel master. **Int. Scaling: 1 == 1 Type: C Volatile: N**	CurRef311	FluxRefEMF	CurRef311	-	C
43.03	**CurRefExt (external current reference)** External current reference in percent of *M1NomCur (99.03).* **Note:** *CurRefExt (43.03)* is only valid, if *CurSel (43.02)* = **CurRefExt**. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-325	325	0		E
43.04	**CurRefSlope (current reference slope)** *CurRefSlope (43.04)* in percent of *M1NomCur (99.03)* per 1 ms. The di/dt limitation is located at the input of the current controller. **Int. Scaling: 100 == 1 %/ms Type: I Volatile: N**	0.2	40	10	%/ms	E

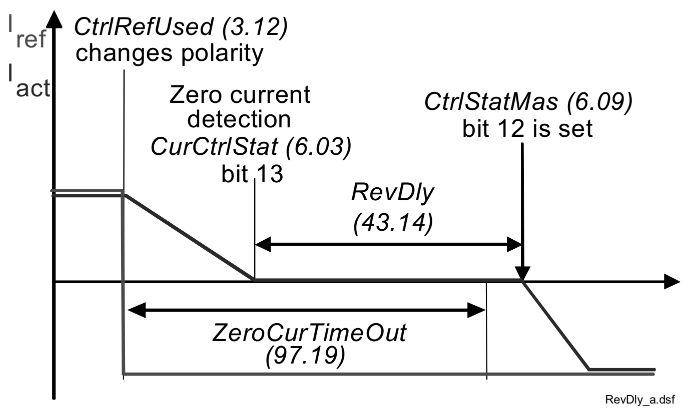
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
43.05	**CtrlModeSel (control mode selector)** Current controller mode selection: 0 = **Standard** PI-controller with RL compensation of EMF based on current actual plus feed forward, default 1 = **FeedFwdRef** PI-controller with RL compensation of EMF based on current reference plus feed forward 2 = **NoFeedFwd** PI-controller without RL compensation of EMF. No feed forward takes place, should not be used for motoric applications. 3 = **PowerSupply1** for more information see *DCS800 Power Supply Control Manual (3ADW000375)* 4 = **PowerSupply2** for more information see *DCS800 Power Supply Control Manual (3ADW000375)* **Int. Scaling: 1 == 1 Type: C Volatile: N**	Standard	PowerSupply2	Standard	-	E
43.06	**M1KpArmCur (motor 1 p-part armature current controller)** Proportional gain of the current controller. Example: The controller generates 15 % of motor nominal current [*M1NomCur (99.03)*] with *M1KpArmCur (43.06)* = 3, if the current error is 5 % of *M1NomCur (99.03)*. **Int. Scaling: 100 == 1 Type: I Volatile: N**	0	100	0.1	-	C
43.07	**M1TiArmCur (motor 1 i-part armature current controller)** Integral time of the current controller. *M1TiArmCur (43.07)* defines the time within the integral part of the controller achieves the same value as the proportional part. Example: The controller generates 15 % of motor nominal current [*M1NomCur (99.03)*] with *M1KpArmCur (43.06)* = 3, if the current error is 5 % of *M1NomCur (99.03)*. On that condition and with *M1TiArmCur (43.07)* = 50 ms follows: – the controller generates 30 % of motor nominal current, if the current error is constant, after 50 ms are elapsed (15 % from proportional part and 15 % from integral part). Setting *M1TiArmCur (43.07)* to 0 ms disables the integral part of the current controller and resets its integrator. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	50	ms	C
43.08	**M1DiscontCurLim (motor 1 discontinuous current limit)** Threshold continuous / discontinuous current in percent of *M1NomCur (99.03)*. The actual continuous / discontinuous current state can be read from *CurCtrlStat1 (6.03)* bit 12. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	325	100	%	C
43.09	**M1ArmL (motor 1 armature inductance)** Inductance of the armature circuit in mH. Used for the EMF compensation: $EMF = U_A - R_A * I_A - L_A * \frac{dI_A}{dt}$ **Attention:** Do not change the default values of *M1ArmL (43.09)* and *M1ArmR (43.10)*! Changing them will falsify the results of the autotuning. **Int. Scaling: 100 == 1 mH Type: I Volatile: N**	0	640	0	mH	C
43.10	**M1ArmR (motor 1 armature resistance)** Resistance of the armature circuit in mΩ. Used for the EMF compensation: $EMF = U_A - R_A * I_A - L_A * \frac{dI_A}{dt}$ **Attention:** Do not change the default values of *M1ArmL (43.09)* and *M1ArmR (43.10)*! Changing them will falsify the results of the autotuning. **Int. Scaling: 1 == 1 mΩ Type: I Volatile: N**	0	65500	0	mΩ	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
43.11	**PropFbSel (p-part current feedback selection)** *PropFbSel (43.11)* chooses the armature current feedback type for the p-part of the armature current controller: 0 = **PeakCur** peak current measurement is used, default 1 = **AverageCur** average current measurement is used **Int. Scaling: 1 == 1 Type: C Volatile: N**	PeakCur	AverageCur	PeakCur	-	E
43.12	**Uk (relative short circuit impedance)** For more information contact Your ABB representative. **Int. Scaling: 10 == 1 % Type: I Volatile: N**	0	15	0	%	E
43.13	**FiringLimMode (firing limit mode)** *FiringLimMode (43.13)* selects the strategy for *ArmAlphaMax (20.14):* 0 = **Fix** the firing angle limit is defined by *ArmAlphaMax (20.14)* 1 = **FixSingle** The firing angle limit is defined by *ArmAlphaMax (20.14)*. When *ArmAlphaMax (20.14)* is reached single firing pulses are fired, default 2 = **Calculated** the firing limit is reduced from 165° to *ArmAlphaMax (20.14)* depending on the actual motor current and *M1DiscontCurLim (43.08)* 3 = **CalcSingle** function same as in **Calculated**, but single pulses are fired when the limit is reached  **Note:** Single firing pulses force discontinuous current automatically to zero. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Fix	CalcSingle	FixSingle	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
43.14	**RevDly (reversal delay)**	0	600	5	ms	E
43.15	**Unused**					
43.16	**RevMode (reversal mode)**	Soft	Hard	Hard	-	E

43.14 RevDly (reversal delay)

RevDly (43.14) defines the delay time in ms for the bridge reversal after zero current has been detected - see *CurCtrlStat1 (6.03)* bit 13.



The reversal delay starts when zero current has been detected - see *CurCtrlStat1 (6.03)* bit 13 - after a command to change current direction - see *CurRefUsed (3.12)* - has been given. After a command to change the current direction the opposite current has to be reached before *ZeroCurTimeOut (97.19)* has been elapsed otherwise the drive trips with **F557 ReversalTime** [*FaultWord4 (9.04)* bit 8].

RevDly (43.14) must have the same setting for 12-pulse master and 12-pulse slave with one exception only:

- If there is no current measurement in the 12-pulse serial slave, set *RevDly (43.14)* in the 12-pulse serial slave to minimum (0 ms). Thus the 12-pulse serial slave uses the reversal command of the 12-pulse master for its own bridge changeover - see *CtrlStatMas (6.09)* bit 12. No additional reversal delay is added, since the master delays bit 12 according to its own *RevDly (43.14)*.

Note:
12P RevTimeOut (47.05) must be longer than *ZeroCurTimeOut (97.19)* and *ZeroCurTimeOut (97.19)* must be longer than *RevDly (43.14)*.

Int. Scaling: 1 == 1 ms Type: I Volatile: N

43.15 Unused

43.16 RevMode (reversal mode)

RevMode (43.16) defines the behavior of the speed ramp and speed controller during bridge and field reversal (torque reversal):

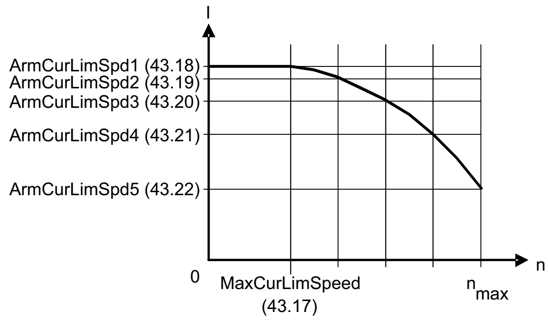
0 = **Soft** the speed ramp and speed controller are frozen during reversal --> bumpless reversal

1 = **Hard** the speed ramp and speed controller are released during reversal --> the drive follows the ramp, default

Note:
RevMode (43.16) is automatically set to **Hard** when *RevDly (43.14)* is equal or less than 25 ms.

Int. Scaling: 1 == 1 Type: C Volatile: N

Speed depending current limit:



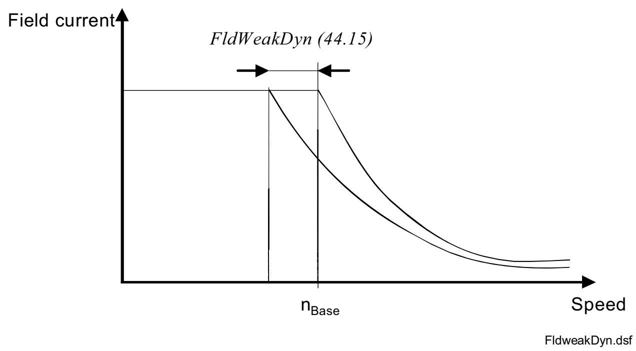
n_{max} = maximum absolute value of *M1SpeedMin (20.01)* and *M1SpeedMax (20.02)*

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
43.17	**MaxCurLimSpeed (speed limit for maximum armature current)** Minimum speed level where the armature current reduction begins. Internally limited from: $0rpm \ \ to \ \ (2.29)*\frac{32767}{20000}rpm$ **Int. Scaling: (2.29) Type: I Volatile: N**	0	10000	1500	rpm	E
43.18	**ArmCurLimSpeed1 (armature current at speed limit 1)** Armature current limit - in percent of *M1NomCur (99.03)* - at *MaxCurLimSpeed (43.17)*. Should be set to the maximum absolute value of *M1CurLimBrdg1 (20.12)* and *M1CurLimBrdg2 (20.13)*. **Note:** The used current limit depends also on the converter's actual limitation situation (e.g. torque limits, other current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	325	325	%	E
43.19	**ArmCurLimSpeed2 (armature current at speed limit 2)** Armature current limit - in percent of *M1NomCur (99.03)* - at speed: $(43.17)+\frac{1}{4}*[n_{\max}-(43.17)]$ with: n_{max} = Max [|*(20.01)*|, |*(20.02)*|] **Note:** The used current limit depends also on the converter's actual limitation situation (e.g. torque limits, other current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	325	325	%	E
43.20	**ArmCurLimSpeed3 (armature current at speed limit 3)** Armature current limit - in percent of *M1NomCur (99.03)* - at speed: $(43.17)+\frac{1}{2}*[n_{\max}-(43.17)]$ with: n_{max} = Max [|*(20.01)*|, |*(20.02)*|] **Note:** The used current limit depends also on the converter's actual limitation situation (e.g. torque limits, other current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	325	325	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
43.21	**ArmCurLimSpeed4 (armature current at speed limit 4)** Armature current limit - in percent of *M1NomCur (99.03)* - at speed: $(43.17)+\frac{3}{4}*\left[n_{max}-(43.17)\right]$ with: n_{max} = Max [|*(20.01)*|, |*(20.02)*|] **Note:** The used current limit depends also on the converter's actual limitation situation (e.g. torque limits, other current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	325	325	%	E
43.22	**ArmCurLimSpeed5 (armature current at speed limit 5)** Armature current limit - in percent of *M1NomCur (99.03)* - at n_{max} = Max [|*(20.01)*|, |*(20.02)*|]. **Note:** The used current limit depends also on the converter's actual limitation situation (e.g. torque limits, other current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	325	325	%	E
43.23	**PwrConfig (power part configuration)** *PwrConfig (43.23)* defines the configuration of the connected power part: 0 = **6-pulse** the connected power part is a B6 bridge, default 1 = reserved 2 = reserved 3 = reserved 4 = reserved **Int. Scaling: 1 == 1 Type: C Volatile: N**	6-pulse	reserved	6-pulse	-	E
43.24	**PwrSupplyRefExt (external voltage reference power supply mode)** External voltage reference for power supply mode in percent of *M1NomVolt (99.02).* For more information see *DCS800 Power Supply Control Manual (3ADW000375)*. **Note:** *PwrSupplyRefExt (43.24)* is only valid, if *ControlModeSel (43.05)* = **PowerSupply1** or **PowerSupply2**. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-150	150	0	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 44	**Field excitation**					
44.01	**FldCtrlMode (field control mode)** Motor 1 field control mode selection: 0 = **Fix** constant field (no field weakening), EMF controller blocked, field reversal blocked, optitorque blocked, default 1 = **EMF** field weakening active, EMF controller released, field reversal blocked, optitorque blocked 2 = **Fix/Rev** constant field (no field weakening), EMF controller blocked, field reversal active, optitorque blocked 3 = **EMF/Rev** field weakening active, EMF controller released, field reversal active, optitorque blocked 4 = **Fix/Opti** constant field (no field weakening), EMF controller blocked, field reversal blocked, optitorque active 5 = **EMF/Opti** field weakening active, EMF controller released, field reversal blocked, optitorque active 6 = **Fix/Rev/Opti** constant field (no field weakening), EMF controller blocked, field reversal active, optitorque active 7 = **EMF/Rev/Opti** field weakening active, EMF controller released, field reversal active, optitorque active **Note:** The field control mode for motor 2 depends on the setting of *M2RefFieldMode (45.13).* **Note:** It is not possible to go into field weakening range when *M1SpeeFbSel (50.03)* = **EMF**. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Fix	EMF/Rev/Opti	Fix	-	C
44.02	**M1KpFex (motor 1 p-part field current controller)** Proportional gain of the field current controller. Example: The controller generates 15 % of motor nominal field current [*M1NomFldCur (99.11)*] with *M1KpFex (44.02)* = 3, if the field current error is 5 % of *M1NomFldCur (99.11)*. **Int. Scaling: 100 == 1 Type: I Volatile: N**	0	325	0.2	-	C
44.03	**M1TiFex (motor 1 i-part field current controller)** Integral time of the field current controller. *M1TiFex (44.03)* defines the time within the integral part of the controller achieves the same value as the proportional part. Example: The controller generates 15 % of motor nominal field current [*M1NomFldCur (99.11)*] with *M1KpFex (44.02)* = 3, if the field current error is 5 % of *M1NomFldCur (99.11)*. On that condition and with *M1TiFex (44.03)* = 200 ms follows: – the controller generates 30 % of motor nominal field current, if the current error is constant, after 200 ms are elapsed (15 % from proportional part and 15 % from integral part). Setting *M1TiFex (44.03)* to 0 ms disables the integral part of the field current controller and resets its integrator. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	200	ms	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
44.04	**M1FldHeatRef (motor 1 field heating reference)** Field current reference - in percent of *M1NomFieldCur (99.11)* - for field heating and field economy. Field heating: Field heating is released according to *FldHeatSel (21.18)*. Field economy: Field economy is only available when 2 motors with 2 independent field exciters are connected to the drive. Field economy for motor 1 is released by means of *M1FldHeatRef (44.04)* < 100 % and activated, if: – **On** = 1 [*UsedMCW (7.04)* bit 0] for longer than 10 s, – the other motor is selected via *ParChange (10.10)*, – the other motor can be seen in *MotSel (8.09)* and – *M1FldRefMode (45.05)* = *M2FldRefMode (45.13)* = **Internal**. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	0	100	100	%	E
44.05	**Unused**					
44.06	**Unused**					
44.07	**EMF CtrlPosLim (positive limit EMF controller)** Positive limit for EMF controller in percent of nominal flux. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	0	100	10	%	E
44.08	**EMF CtrlNegLim (negative limit EMF controller)** Negative limit for EMF controller in percent of nominal flux. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	-100	0	-100	%	E
44.09	**KpEMF (p-part EMF controller)** Proportional gain of the EMF controller. Example: The controller generates 15 % of motor nominal EMF with *KpEMF (44.09)* = 3, if the EMF error is 5% of *M1NomVolt (99.02)*. **Int. Scaling: 100 == 1 Type: I Volatile: N**	0	325	0.5	-	E
44.10	**TiEMF (i-part EMF controller)** Integral time of the EMF controller. *TiEMF (44.10)* defines the time within the integral part of the controller achieves the same value as the proportional part. Example: The controller generates 15 % of motor nominal EMF with *KpEMF (44.09)* = 3, if the EMF error is 5% of *M1NomVolt (99.02)*. On that condition and with *TiEMF (44.10)* = 20 ms follows: – the controller generates 30 % of motor nominal EMF, if the EMF error is constant, after 20 ms are elapsed (15 % from proportional part and 15 % from integral part). Setting *TiEMF (44.10)* to 0 ms disables the integral part of the EMF controller and resets its integrator. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	50	ms	E
44.11	**Unused**					
44.12	**FldCurFlux40 (field current at 40% flux)** Field current at 40 % flux in percent of *M1NomFldCur (99.11)*. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	0	100	40	%	E
44.13	**FldCurFlux70 (field current at 70% flux)** Field current at 70 % flux in percent of *M1NomFldCur (99.11)*. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	0	100	70	%	E
44.14	**FldCurFlux90 (field current at 90% flux)** Field current at 90 % flux in percent of *M1NomFldCur (99.11)*. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	0	100	90	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
44.15	**FldWeakDyn (dynamic field weakening)** If the motor speed passes the field weakening point (== base speed) quickly, voltage overshoot may occur. To solve this problem the field weakening point can be lowered by means of *FldWeakDyn (44.15). FldWeakDyn (44.15)* is set in percent of *M1BaseSpeed (99.04).* **Note:** The lowered field weakening point is compensated by the EMF controller in case of constant speed or slow speed change. *EMF CtrlPosLim (44.07)* has to be set high enough to allow the EMF controller to compensate.  **Int. Scaling: 1 == 1 % Type: I Volatile: N**	80	100	100	%	E
44.16	**Unused**					
44.17	**FldBoostSel (field boost selector)** Selector for *FldBoostSel (44.17):* 0 = **NotUsed** field boost is blocked, default 1 = **Run** field boost starts with **Run** = 1 [*MainCtrlWord (7.01)* bit 3] 2 = **DI1** 1 = field boost, 0 = no field boost 3 = **DI2** 1 = field boost, 0 = no field boost 4 = **DI3** 1 = field boost, 0 = no field boost 5 = **DI4** 1 = field boost, 0 = no field boost 6 = **DI5** 1 = field boost, 0 = no field boost 7 = **DI6** 1 = field boost, 0 = no field boost 8 = **DI7** 1 = field boost, 0 = no field boost 9 = **DI8** 1 = field boost, 0 = no field boost 10 = **DI9** 1 = field boost, 0 = no field boost. Only available with digital extension board 11 = **DI10** 1 = field boost, 0 = no field boost. Only available with digital extension board 12 = **DI11** 1 = field boost, 0 = no field boost. Only available with digital extension board 13 = **MCW Bit11** 1 = field boost, 0 = no field boost, *MainCtrlWord (7.01)* bit 11 14 = **MCW Bit12** 1 = field boost, 0 = no field boost, *MainCtrlWord (7.01)* bit 12 15 = **MCW Bit13** 1 = field boost, 0 = no field boost, *MainCtrlWord (7.01)* bit 13 16 = **MCW Bit14** 1 = field boost, 0 = no field boost, *MainCtrlWord (7.01)* bit 14 17 = **MCW Bit15** 1 = field boost, 0 = no field boost, *MainCtrlWord (7.01)* bit 15 18 = **ACW Bit12** 1 = field boost, 0 = no field boost, *AuxCtrlWord (7.02)* bit 12 19 = **ACW Bit13** 1 = field boost, 0 = no field boost, *AuxCtrlWord (7.02)* bit 13 20 = **ACW Bit14** 1 = field boost, 0 = no field boost, *AuxCtrlWord (7.02)* bit 14 21 = **ACW Bit15** 1 = field boost, 0 = no field boost, *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	E

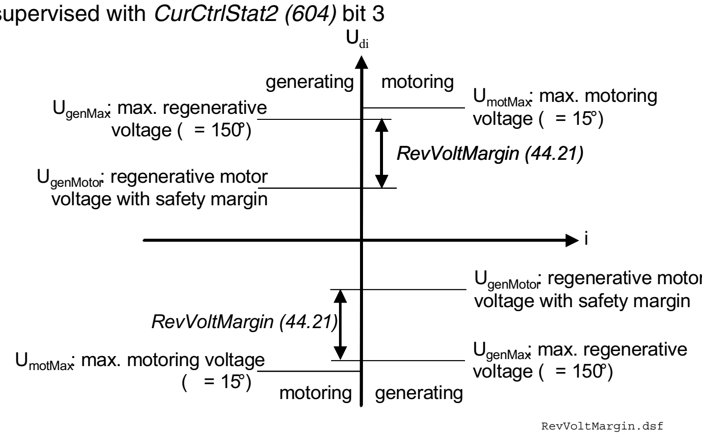
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
44.18	**FldBoostFact (field boost factor)** Field boost factor in percent of *M1NomFldCur (99.11).* The resulting field boost current must be lower than the nominal current of the used field exciter. If the field boost current is out of range **A132 ParConflict** [*AlarmWord2 (9.07)* bit 15] is generated. **Note:** If *FldBoostFact (44.18)* > 100 % and *M1UsedFexType (99.12)* = **OnBoard** to **DCF804-0060** or **FEX-4-Term5A** *S M1FldSacle (45.20)* has to be set accordingly. Example: *M1NomFldCur (99.11)* = 20 A and *FldBoostFact (44.18)* = 150 % then *S M1FldSacle (45.20)* = 30 A **Note:** If *FldBoostFact (44.18)* > 100 % and *M2UsedFexType (49.07)* = **OnBoard** to **DCF804-0060** or **FEX-4-Term5A** *S M2FldSacle (45.21)* has to be set accordingly. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	100	160	100	%	E
44.19	**FldBoostTime (field boost time)** Time the field boost should last. **Int. Scaling: 1 == 1 s Type: I Volatile: N**	0	600	0	s	E
44.20	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
44.21	**RevVoltMargin (reversal voltage margin)**	0	20	6	%	E

RevVoltMargin (44.21) - in percent of *NomMainsVolt (99.10)* - is a safety margin for the motor voltage during regenerative mode. Setting *RevVoltMargin (44.21)* to 0 provides no protection against commutation faults (shooting through).
The function of *RevVoltMargin (44.21)* is the following:
To prevent the drive from blowing fuses when going from motoring (using forward bridge) to generating (using reverse bridge) the armature voltage has to be lower than the corresponding mains voltage. This is automatically checked by the DCS800 and the reverse bridge is blocked as long as the armature voltage is too high. To lower the armature voltage two ways are possible:

- lowering the motor speed by idling or
- adapting the flux by lowering the field current - e.g. set *FldCtrlMode (44.01)* = **EMF**

Both options take time and thus delaying the current / torque reversal. For faster adapting of the motor voltage activate the field weakening function.
This can be supervised with *CurCtrlStat2 (604)* bit 3



For regenerative mode is valid:

$$U_{genMotor} = |U_{genMax}| - U_{Safety}$$

$$with \quad U_{genMax} = 1.35 * \cos\alpha_{\max} * U_{Mains\_act}$$

$$U_{genMax} = 1.35 * \cos(20.14) * U_{Mains\_act}$$

$$and \quad U_{Safety} = (44.21)$$

$$follows:$$

$$U_{genMotor} = |1.35 * \cos(20.14) * U_{Mains\_act}| - (44.21) * U_{Mains\_act}$$

Example:
With *ArmAlphaMax (20.14)* = 150°, *RevVoltMargin (44.21)* = 10 % and U_{Mains_act} = *NomMainsVolt (99.10)* follows:

$$U_{genMotor} = |1.35 * \cos 150° * U_{Mains\_act}| - 0.1 * U_{Mains\_act}$$

$$U_{genMotor} = |-1.16 * U_{Mains\_act}| - 0.1 * U_{Mains\_act}$$

$$follows:$$

$$U_{genMotor} = 1.06 * U_{Mains\_act}$$

Int. Scaling: 100 == 1 % Type: I Volatile: N

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
44.22	**VoltRefExt (external EMF voltage reference)** External EMF voltage reference in percent of *M1NomVolt (99.02).* **Note:** *VoltRefExt (44.22)* is only valid, if *EMF RefSel (44.23)* = **VoltRefExt**. **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-100	100	0		E
44.23	**EMF RefSel (EMF reference selector)** *EMF RefSel (44.23)* selector: 0 = **Internal** internally calculated EMF, default 1 = **Ext4422** *VoltRefExt (44.22)* external EMF voltage reference 2 = **AI1** analog input AI1 3 = **AI2** analog input AI2 4 = **AI3** analog input AI3 5 = **AI4** analog input AI4 6 = **AI5** analog input AI5 7 = **AI6** analog input AI6 **Int. Scaling: 1 == 1 Type: C Volatile: N**	Internal	AI6	Internal	-	E
44.24	**Unused**					
44.25	**VoltCorr (EMF voltage correction)** EMF voltage correction in percent of *M1NomVolt (99.02).* Added to *VoltRef1 (3.25).* **Int. Scaling: 100 == 1 % Type: SI Volatile: Y**	-100	100	0		E
44.26	**VoltRefSlope (EMF voltage reference slope)** EMF voltage reference slope in percent *M1NomVolt (99.02)* per 1 ms. The dv/dt limitation is located at the input of the EMF controller. **Int. Scaling: 100 == 1 %/ms Type: I Volatile: N**	0.01	100	30	%/ms	E
44.27	**FluxCorr (flux correction)** *FluxCorr (44.27)* in percent of nominal flux is added to the sum of the flux reference *FluxRefSum (3.28).* **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-100	100	0	%	E

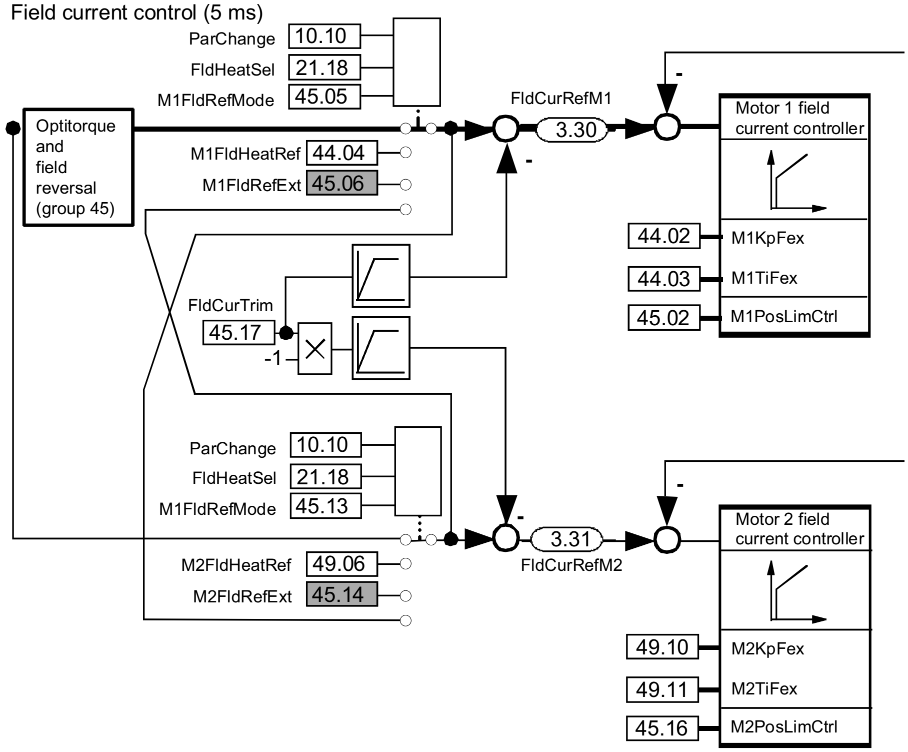
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
44.28	**MG ConfigWord (MG-set configuration word)** MG-set configuration word. For more information see *DCS800 MG-set motor control (3ADW000310)*.	-	-	-	-	E

Bit	Name	Value	Comment
B0	reserved	1	
		0	
B1	reserved	1	
		0	
B2	reserved	1	
		0	
B3	reserved	1	
		0	
B4	reserved	1	
		0	
B5	reserved	1	
		0	
B6	reserved	1	
		0	
B7	reserved	1	
		0	
B8	reserved	1	
		0	
B9	reserved	1	
		0	
B10	reserved	1	
		0	
B11	reserved	1	
		0	
B12	reserved	1	
		0	
B13	reserved	1	
		0	
B14	**SpeedController**	1	Release speed controller
		0	no action
B15	reserved	1	
		0	

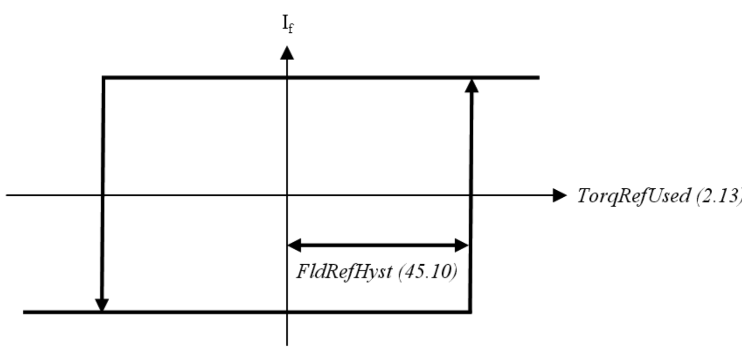
Int. Scaling: 1 == 1 Type: I Volatile: Y

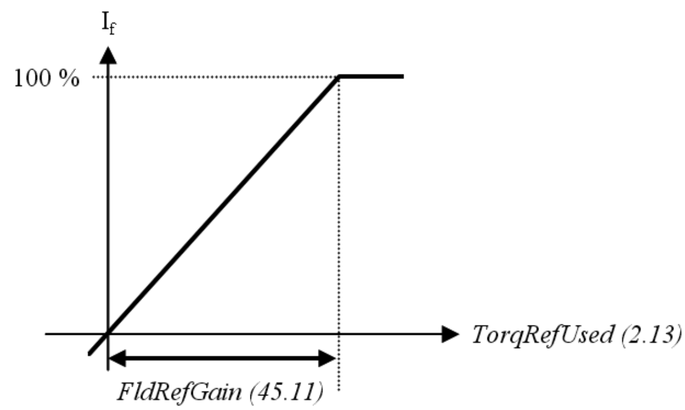
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 45	**Field converter settings**					
45.01	**M1FreewhlLev (motor 1 freewheeling level)** Motor 1 field exciter free wheeling level [only when *M1UsedFexType (99.12)* = **DCF804-0050** or **DCF804-0060**] in percent / ms of the actual field exciter supply voltage. If 2 successive AC-voltage measurements differ more than *M1FreewhlLev (45.01),* the free-wheeling function is activated. **Int. Scaling: 1 == 1 %/ms Type: I Volatile: N**	0	100	20	%/ms	E
45.02	**M1PosLimCtrl (motor 1 positive voltage limit for field exciter)** Positive voltage limit for motor 1 field exciter in percent of the maximum field exciter output voltage. Example: With a 3-phase supply voltage of 400 VAC the field current controller can generate a maximum output voltage of 521 VDC. In case the rated field supply voltage is 200 VDC, then it is possible to limit the controllers' output voltage to 46 %. That means the firing angle of the field current controller is limited in such a way that the average output voltage is limited to a maximum of 240VDC. **Note:** 4-Q field exciters which can reverse the field current will used *M1PosLimCtrl (45.02)* also as negative limit. **Int. Scaling: 100 = 1 % Type: I Volatile: N**	0	100	100	%	E
45.03	**Unused**					
45.04	**Unused**					

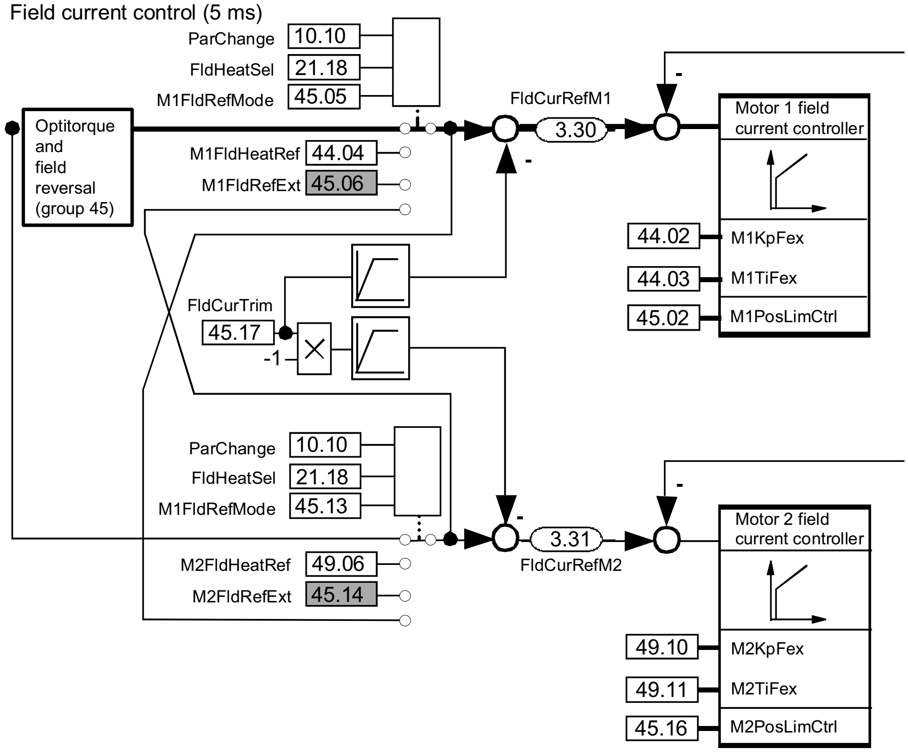
<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>45.05</td><td>M1FldRefMode (motor 1 field current reference mode)
M1FldRefMode (45.05) selector:
0 = Internal motor 1 field current reference according to shared motion MotSel (8.09) or field heating FldHeatSel (21.18), default
1 = M2FldCurRef field current reference is taken from motor 2
2 = M1FldRefExt M1FldRefExt (45.06) external field current reference

Field current control (5 ms)


Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>Internal</td><td>M1FldRefExt</td><td>Internal</td><td>-</td><td>E</td></tr>
<tr><td>45.06</td><td>M1FldRefExt (motor 1 external field current reference)
Motor 1 external field current reference input in percent of M1NomFldCur (99.11).
Note:
M1FldRefExt (45.06) is only valid, if M1FldRefMode (45.05) = M1FldRefExt.
Int. Scaling: 100 == 1 % Type: SI Volatile: N</td><td>-100</td><td>100</td><td>100</td><td>%</td><td>E</td></tr>
<tr><td>45.07</td><td>ForceFldDir (force field current direction)
Motor 1 field direction force command:
0 = NotUsed the field direction is controlled by FldCtrlMode (44.01) and TorqRefUsed (2.13), default
1 = Forward field direction is forced to forward direction
2 = Reverse field direction is forced to reverse direction
3 = ExtReverse In case an external contactor in the field current loop is used to change the field direction, ForceFldDir (45.07) has to be switched between Forward and ExtReverse. ExtReverse adapts the armature voltage and speed supervision. The external contactor interlocking and the control of ForceFldDir (45.07) have to be done by means of Adaptive Program, application program or overriding control.
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>NotUsed</td><td>ExtReverse</td><td>NotUsed</td><td>-</td><td>E</td></tr>
</table>

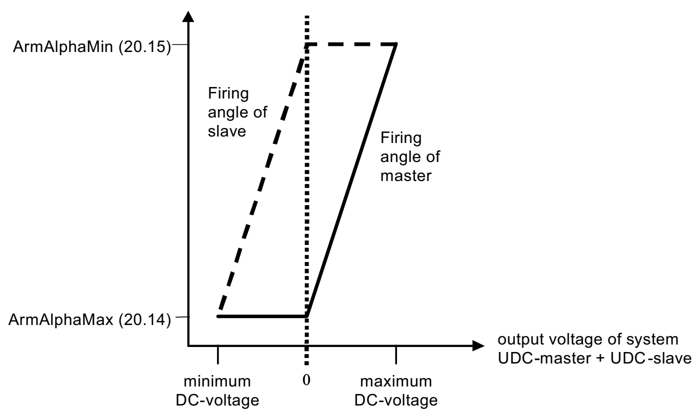
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
45.08	**FluxRevMonDly (flux reversal monitoring delay)** Maximum allowed time within *Mot1FldCurRel (1.29)* and the internal motor flux doesn't correspond to each other during field reversal. During this time **F522 SpeedFb** [*FaultWord2 (9.02)* bit 5] is disabled. **Note:** *FluxRevMonDly (45.08)* is only effective for *FldCtrlMode (44.01)* = **Fix/Rev**, **EMF/Rev**, **Fix/Rev/Opti** or **EMF/Rev/Opti**. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	20000	0	ms	E
45.09	**FldRevHyst (field current reversal hysteresis)** The sign of *Mot1FldCurRel (1.29)* is used to generate the field reversal acknowledge. To avoid signal noise problems a small hysteresis - in percent of *M1NomFldCur (99.11)* - is needed. **Note:** *FldRevHyst (45.09)* is only effective for *FldCtrlMode (44.01)* = **Fix/Rev**, **EMF/Rev**, **Fix/Rev/Opti** or **EMF/Rev/Opti**. **Int. Scaling: 100 = 1 % Type: I Volatile: N**	0	100	2	%	E
45.10	**FldRefHyst (field torque reference hysteresis)** To prevent the field reversal from continuous toggling due to a too small torque reference a *TorqRefUsed (2.13)* hysteresis - in percent of *MotNomTorque (4.23)* - is available. The hysteresis is symmetrical and is set by *FldRefHyst (45.10)*. The field reversal is controlled by the sign of *TorqRefUsed (2.13)*:  **Note:** *FldRefHyst (45.10)* is only effective for *FldCtrlMode (44.01)* = **Fix/Rev** or **EMF/Rev**. **Int. Scaling: 100 = 1 % Type: I Volatile: N**	0	100	2	%	E

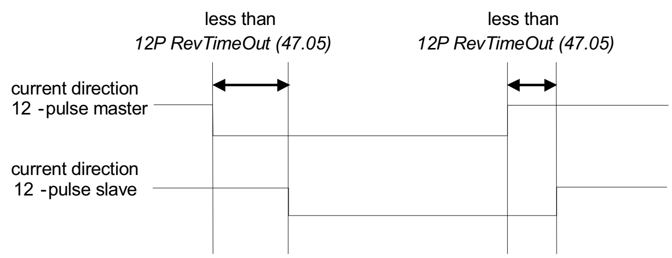
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
45.11	**FldRefGain (field current reference gain)** Optitorque calculates the field current reference depending on *TorqRefUsed (2.13)*. Thus, the field current is reduced to a smaller value, if *TorqRefUsed (2.13)* is accordingly low. This speeds up the field reversal, assuming *TorqRefUsed (2.13)* is low during field reversal. Optitorque is activated by means of *FldCtrlMode (44.01)* and like field reversal only available for motor 1 field exciter. The relation between *TorqRefUsed (2.13)* and *FldCurRefM1 (3.30)* is linear and without offset. It is defined by means of the *FldRefGain (45.11)*. The gain is related to *M1NomFldCur (99.11)* as well as to *MotNomTorque (4.23)*.  Example: With *FldRefGain (45.11)* = 20 %, 100 % field current is generated at *TorqRefUsed (2.13)* = 20 %. **Note:** *FldRefGain (45.11)* is only effective for *FldCtrlMode (44.01)* = **Fix/Opti**, **EMF/Opti**, **Fix/Rev/Opti** or **EMF/Rev/Opti**. **Int. Scaling: 100 = 1 % Type: I Volatile: N**	0	100	50	%	E
45.12	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
45.13	**M2FldRefMode (motor 2 field current reference mode)** *M2FldRefMode (45.13)* selector: 0 = **Internal** motor 2 field current reference according to shared motion *MotSel (8.09)* or field heating *FldHeatSel (21.18),* default 1 = **M1FldCurRef** field current reference is taken from motor 1 2 = **M2FldRefExt** *M2FldRefExt (45.14)* external field current reference Field current control (5 ms)  **Int. Scaling: 1 == 1 Type: C Volatile: N**	Internal	M2FldRefExt	Internal	-	E
45.14	**M2FldRefExt (motor 2 external field current reference)** Motor 2 external field current reference input in percent of *M2NomFldCur (49.05).* **Note:** *M2FldRefExt (45.14)* is only valid, if *M2FldRefMode (45.13)* = **M2FldRefExt**. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-100	100	100	%	E
45.15	**M2FreewhlLev (motor 2 freewheeling level)** Motor 2 field exciter free wheeling level [only when *M2UsedFexType (49.07)* = **DCF804-0050** or **DCF804-0060**] in percent / ms of the actual field exciter supply voltage. If 2 successive AC-voltage measurements differ more than *M2FreewhlLev (45.15),* the free-wheeling function is activated. **Int. Scaling: 1 == 1 %/ms Type: I Volatile: N**	0	100	20	%/ms	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
45.16	**M2PosLimCtrl (motor 2 positive voltage limit for field exciter)** Positive voltage limit for motor 2 field exciter in percent of the maximum field exciter output voltage. Example: With a 3-phase supply voltage of 400 VAC the field current controller can generate a maximum output voltage of 521 VDC. In case the rated field supply voltage is 200 VDC, then it is possible to limit the controllers' output voltage to 46 %. That means the firing angle of the field current controller is limited in such a way that the average output voltage is limited to a maximum of 240VDC. **Note:** 4-Q field exciters which can reverse the field current will used *M2PosLimCtrl (45.16)* also as negative limit. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	100	100	%	E
45.17	**FldCurTrim (field current trimming)** The field current of motor 1 and motor 2 can be corrected by means of *FldCurTrim (45.17)* in percent of *M1NomFldCur (99.11)* respectively *M2NomFldCur (49.05)*: – 0 % to 20 %: The value is subtracted from motor 1 field current reference. The result is visible in *FldCurRefM1 (3.30)*. – -20 % to 0 %: The absolute value is subtracted from motor 2 field current reference. The result is visible in *FldCurRefM2 (3.31)*. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-20	20	0	%	E
45.18	**FldMinTripDly (delay field current minimum trip)** *FldMinTripDly (45.18)* delays **F541 M1FexLowCur** [*FaultWord3 (9.03)* bit 8] respectively **F542 M2FexLowCur** [*FaultWord3 (9.03)* bit 9]. If the field current recovers before the delay is elapsed **F541** / **F542** will be disregarded: – *M1FldMinTrip (30.12)* – *M2FldMinTrip (49.08)* **Note:** *FldMinTripDly (45.18)* is blocked when *OperModeSel (43.01)* = **FieldConv**. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	50	10000	2000	ms	E
45.19	**Unused**					
45.20	**S M1FldScale (set: motor 1 field current scaling factor)** Motor 1 field exciter scaling factor. *S M1FldScale (45.20)* is write protected, unless *ServiceMode (99.06)* = **SetTypeCode**. To use *S M1FldScale (45.20)* following inequation has to be valid: *M1NomFldCur (99.11)* ≤ *S M1FldScale (45.20)* ≤ maximum field current of the used field exciter – For *S M1FldScale (45.20)* > maximum field current of the used field exciter **A132 ParConflict** [*AlarmWord2 (9.07)* bit 15] is generated. – For *M1NomFldCur (99.11)* > *S M1FldScale (45.20)* the scaling is automatically set by *M1NomFldCur (99.11)*. – The scaling factor is released when *M1NomFldCur (99.11)* < *S M1FldScale (45.20)* and *M1UsedFexType (99.12)* = **OnBoard** to **DCF804-0060** or **FEX-4-Term5A**. If the scaling is changed its new value is taken over immediately. **Int. Scaling: 100 == 1 A Type: I Volatile: N**	0	60	0	A	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
45.21	**S M2FldScale (set: motor 2 field current scaling factor)** Motor 2 field exciter scaling factor. *S M2FldScale (45.21)* is write protected, unless *ServiceMode (99.06)* = **SetTypeCode**. To use *S M2FldScale (45.21)* following inequation has to be valid: *M2NomFldCur (49.05)* ≤ *S M2FldScale (45.21)* ≤ maximum field current of the used field exciter – For *S M2FldScale (45.21)* > maximum field current of the used field exciter **A132 ParConflict** [*AlarmWord2 (9.07)* bit 15] is generated. – For *M2NomFldCur (49.05)* > *S M2FldScale (45.21)* the scaling is automatically set by *M2NomFldCur (49.05)*. – The scaling factor is released when *M2NomFldCur (49.05)* < *S M2FldScale (45.21)* and *M2UsedFexType (49.07)* = **OnBoard** to **DCF804-0060** or **FEX-4-Term5A**. If the scaling is changed its new value is taken over immediately. **Int. Scaling: 100 == 1 A Type: I Volatile: N**	0	60	0	A	E
45.22	**M1OperModeFex4 (motor 1 fex4 operation mode selector)** The FEX-425-Int, DCF803-0016 and DCF803-0035 can be connected to either a 3-phase supply or a single phase supply: 0 = **1-phase** single phase supply 1 = **3-phase** 3-phase supply, default **Int. Scaling: 1 == 1 Type: C Volatile: N**	1-phase	3-phase	3-phase	-	E
45.23	**M2OperModeFex4 (motor 2 fex4 operation mode selector)** The FEX-425-Int, DCF803-0016 and DCF803-0035 can be connected to either a 3-phase supply or a single phase supply: 0 = **1-phase** single phase supply 1 = **3-phase** 3-phase supply, default **Int. Scaling: 1 == 1 Type: C Volatile: N**	1-phase	3-phase	3-phase	-	E
45.24	**MultiFexCount (Multi fex count)** Number of connected field exciters. For more information see *DCS800 MultiFex motor control (3ADW000309)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	32	0	-	E
45.25	**MultiFexOff1 (Multi fex off 1)** For more information see *DCS800 MultiFex motor control (3ADW000309)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-	-	-	-	E
45.26	**MultiFexOff2 (Multi fex off 2)** For more information see *DCS800 MultiFex motor control (3ADW000309)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 47	**12-pulse operation**					
47.01	**12P Mode (12-pulse mode)** The setting of *OperModeSel (43.01)* determines the reaction of *12P Mode (47.01).* *OperModeSel (43.01)* = **12PParMaster** respectively **12PParSlave**: 0 = **Normal** 12-pulse parallel master and 12-pulse parallel slave use their own current controller independently, default 1 = **Difference** the 12-pulse parallel slave calculates the difference between the 12-pulse parallel master actual current and its own actual current and controls this difference to zero by means of its current controller, not implemented yet 2 = Sequential not used for 12-pulse parallel mode 3 = DiodeBridge not used for 12-pulse parallel mode *OperModeSel (43.01)* = **12PSerMaster** respectively **12PSerSlave**: 0 = **Normal** 12-pulse serial master and 12-pulse serial slave are controlled by the same firing angle, default 1 = Difference not used for 12-pulse serial mode 2 = **Sequential** Sequential control of the firing angles. Only one unit changes its firing angle, while the other unit's firing angle is fixed at the minimum- or maximum firing angle. See diagram below. 3 = **DiodeBridge** the 12-pulse serial slave converter is a diode bridge  *12P Mode (47.01)* must have the same setting for 12-pulse master and 12-pulse slave. In case of **DiodeBridge** the setting is only possible in the 12-pulse master. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Normal	DiodeBridge	Normal	-	E
47.02	**DiffCurLim (current difference level)** Permitted current difference between the converters in 12-pulse parallel configuration in percent of *M1NomCur (99.03).* The drive trips with **F534 12PCurDiff** [*FaultWord3 (9.03)* bit 1] if *DiffCurLim (47.02)* is still exceeded when *DiffCurDly (47.03)* is elapsed. *DiffCurLim (47.02)* is only active in the 12-pulse parallel master. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	1	50	10	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
47.03	**DiffCurDly (current difference delay)** *DiffCurDly (47.03)* delays **F534 12PCurDiff** [*FaultWord3 (9.03)* bit 1]. If the current difference becomes smaller than *DiffCurLim (47.02)* before the delay is elapsed **F534** will be disregarded: – *DiffCurLim (47.02)* *DiffCurDly (47.03)* is only active in the 12-pulse parallel master. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	10	64000	500	ms	E
47.04	**Unused**					
47.05	**12P RevTimeOut (12-pulse reversal timeout)** In 12-pulse mode the current direction of both - master and slave - bridges is monitored. The drive trips with **F533 12PRevTime** [*FaultWord3 (9.03)* bit 0] if the 2 converters have different bridges fired for more than *12P RevTimeOut (47.05)*. The reversal fault for 12-pulse is inactive, if *12P RevTimeOut (47.05)* is set to 999 ms or 1000 ms. *12P RevTimeOut (47.05)* is only active in the 12-pulse master.  **Note:** *12P RevTimeOut (47.05)* must be longer than *ZeroCurTimeOut (97.19)* and *ZeroCurTimeOut (97.19)* must be longer than *RevDly (43.14)*. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	1000	100	ms	E
Group 49	**Shared motion**					
49.01	**M2NomVolt (motor 2 nominal DC voltage)** Motor 2 nominal armature voltage (DC) from the motor rating plate. **Note:** In 12-pulse serial mode, this parameter has to be set to the value of the voltage the converter itself is providing. This is usually 50 % of the rated motor voltage, if one motor is connected. In case 2 motors in series are connected it is 100 % of one motor's rated voltage. **Note:** The hardware of the measuring circuit has to be adapted for motor voltages lower than 50 V. **Int. Scaling: 1 == 1 V Type: I Volatile: N**	5	2000	350	V	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.02	**M2NomCur (motor 2 nominal DC current)** Motor 2 nominal armature current (DC) from the motor rating plate. If several motors are connected to the drive, enter the total current of all motors. **Note:** In 12-pulse parallel mode, this parameter has to be set to the value of the current the converter itself is providing. This is usually 50 % of the rated motor current, if one motor is connected. In case 2 motors in parallel are connected it is 100 % of one motor's rated current. **Note:** In case the converter is used as a 3-phase field exciter use *M2NomCur (49.02)* to set the nominal field current. **Int. Scaling: 1 == 1 A Type: I Volatile: N**	0	30000	0	A	E
49.03	**M2BaseSpeed (motor 2 base speed)** Motor 2 base speed from the rating plate, usually the field weak point. *M2BaseSpeed (49.03)* is must be set in the range of: 0.2 to 1.6 times of *SpeedScaleAct (2.29)*. If the scaling is out of range **A124 SpeedScale** [*AlarmWord2 (9.07)* bit 7] is generated. **Int. Scaling: 10 == 1 rpm Type: I Volatile: N**	10	6500	1500	rpm	E
49.04	**M2ZeroSpeedLim (motor 2 zero speed limit)** When the **Run** command is removed [set *UsedMCW (7.04)* bit 3 to zero], the drive will stop as chosen by *StopMode (21.03)*. As soon as the actual speed reaches the limit set by *M2ZeroSpeedLim (49.04)* the motor will coast independent of the setting of *StopMode (21.03)*. Existing brakes are closed (applied). While the actual speed is in the limit **ZeroSpeed** [*AuxStatWord (8.02)* bit 11] is high. **Note:** In case *FlyStart (21.10)* = **StartFrom0** and if the restart command comes before zero speed is reached **A137 SpeedNotZero** [*AlarmWord3 (9.08)* bit 4] is generated. Internally limited from: $0rpm\ \ to\ \ (2.29)rpm$ **Int. Scaling: (2.29) Type: I Volatile: N**	0	1000	75	rpm	C
49.05	**M2NomFldCur (motor 2 nominal field current)** Motor 2 nominal field current from the motor rating plate. **Note:** In case the converter is used as a 3-phase field exciter use *M2NomCur (49.05)* to set the nominal field current. **Int. Scaling: 100 == 1 A Type: I Volatile: N**	0.3	655	0.3	A	E
49.06	**M2FldHeatRef (motor 2 field heating reference)** Field current reference - in percent of *M2NomFieldCur (49.05)* - for field heating and field economy. Field heating: Field heating is released according to *FldHeatSel (21.18)*. Field economy: Field economy is only available when 2 motors with 2 independent field exciters are connected to the drive. Field economy for motor 2 is released by means of *M2FldHeatRef (49.06)* < 100 % and activated, if: – **On** = 1 [*UsedMCW (7.04)* bit 0] for longer than 10 s, – the other motor is selected via *ParChange (10.10)*, – the other motor can be seen in *MotSel (8.09)* and – *M1FldRefMode (45.05)* = *M2FldRefMode (45.13)* = **Internal**. **Int. Scaling: 1 == 1 % Type: I Volatile: N**	0	100	100	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.07	**M2UsedFexType (motor 2 used field exciter type)** Motor 2 used field exciter type: 0 = **NotUsed** no or third party field exciter connected 1 = **OnBoard** integrated 1-Q field exciter (for sizes D1 - D4 only), default 2 = **FEX-425-Int** internal 1-Q 25 A field exciter (for size D5 only) used for field currents from 0.3 A to **25 A** (terminals X100.1 and X100.3) 3 = **DCF803-0035** external 1-Q 35 A field exciter used for field currents from 0.3 A to **35 A** (terminals X100.1 and X100.3) 4 = **DCF803-0050** external 1-Q 50 A field exciter (DCF803-0050 or DCF503B-0050) 5 = **DCF804-0050** external 4-Q 50 A field exciter (DCF804-0050 or DCF504B-0050) 6 = **DCF803-0060** external 1-Q 60 A field exciter; not implemented yet 7 = **DCF804-0060** external 4-Q 60 A field exciter; not implemented yet 8 = **DCS800-S01** external 2-Q 3-phase field exciter 9 = **DCS800-S02** external 4-Q 3-phase field exciter 10 = **DCF803-0016** external 1-Q 16 A field exciter used for field currents from 0.3 A to **16 A** (terminals X100.1 and X100.3) 11 = reserved to 14 = reserved 15 = **ExFex AITAC** third party field exciter, acknowledge via AITAC 16 = **ExFex AI1** third party field exciter, acknowledge via AI1 17 = **ExFex AI2** third party field exciter, acknowledge via AI2 18 = **ExFex AI3** third party field exciter, acknowledge via AI3 19 = **ExFex AI4** third party field exciter, acknowledge via AI4 20 = **FEX-4-Term5A** internal 2-Q 25 A field exciter (FEX-425-Int), external 2-Q 16 A field exciter (DCF803-0016) or external 2-Q 35 A field exciter (DCF803-0035) used for field currents from 0.3 A to **5 A** (terminals X100.2 and X100.3) 21 = reserved 22 = **Exc-Appl-1** see *DCS800 Series wound motor control (3ADW000311)* If the fex type is changed its new value is taken over after the next power-up. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	Exc-Appl-1	NotUsed	-	E
49.08	**M2FldMinTrip (motor 2 minimum field trip)** The drive trips with **F542 M2FexLowCur** [*FaultWord3 (9.03)* bit 9] if *M2FldMinTrip (49.08)* - in percent of *M2NomFldCur (49.05)* - is still undershot when *FldMinTripDly (45.18)* is elapsed. **Note:** *M2FldMinTrip (49.08)* is not valid during field heating and field economy. In this case the trip level is automatically set to 50 % of *M2FldHeatRef (49.06).* The drive trips with **F542 M2FexLowCur** [*FaultWord3 (9.03)* bit 9] if 50 % of *M2FldHeatRef (49.06)* is still undershot when *FldMinTripDly (45.18)* is elapsed. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	100	50	%	E
49.09	**M2FldOvrCurLev (motor 2 field overcurrent level)** The drive trips with **F518 M2FexOverCur** [*FaultWord2 (9.02)* bit 1] if *M2FldOvrCurLev (49.09)* - in percent of *M2NomFldCur (49.05)* - is exceeded. It is recommended to set *M2FldOvrCurtLev (49.09)* at least 25 % higher than *M2NomFldCur (49.05)*. The field overcurrent fault is inactive, if *M2FldOvrCurLev (49.09)* is set to 135 %. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	135	125	%	E
49.10	**M2KpFex (motor 2 p-part field current controller)** Proportional gain of the field current controller. Example: The controller generates 15 % of motor nominal field current [*M2NomFldCur (49.05)*] with *M2KpFex (49.10)* = 3, if the field current error is 5 % of *M2NomFldCur (49.05)*. **Int. Scaling: 100 == 1 Type: I Volatile: N**	0	325	0.2	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.11	**M2TiFex (motor 2 i-part field current controller)** Integral time of the field current controller. *M2TiFex (49.11)* defines the time within the integral part of the controller achieves the same value as the proportional part. Example: The controller generates 15 % of motor nominal field current [*M2NomFldCur (49.05*] with *M2KpFex (49.10)* = 3, if the field current error is 5 % of *M2NomFldCur (49.05)*. On that condition and with *M2TiFex (49.11)* = 200 ms follows: – the controller generates 30 % of motor nominal field current, if the current error is constant, after 200 ms are elapsed (15 % from proportional part and 15 % from integral part). Setting *M2TiFex (49.11)* to 0 ms disables the integral part of the field current controller and resets its integrator. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	200	ms	E
49.12	**M2CurLimBrdg1 (motor 2 current limit of bridge 1)** Current limit bridge 1 in percent of *M2NomCur (49.02)*. Setting *M2CurLimBrdg1 (49.12)* to 0 % disables bridge 1. **Note:** The used current limit depends also on the converter's actual limitation situation (e.g. torque limits, other current limits, field weakening). The limit with the smallest value is valid. **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	0	325	100	%	E
49.13	**M2CurLimBrdg2 (motor 2 current limit of bridge 2)** Current limit bridge 2 in percent of *M2NomCur (49.02)*. Setting *M2CurLimBrdg2 (49.13)* to 0 % disables bridge 2. **Note:** The used current limit depends also on the converter's actual limitation situation (e.g. torque limits, other current limits, field weakening). The limit with the smallest value is valid. **Note:** *M2CurLimBrdg2 (49.13)* is internally set to 0 % if *QuadrantType (4.15)* = 2-Q (2-Q drive). **Int. Scaling: 100 == 1 % Type: SI Volatile: N**	-325	0	-100	%	E
49.14	**M2KpArmCur (motor 2 p-part armature current controller)** Proportional gain of the current controller. Example: The controller generates 15 % of motor nominal current [*M2NomCur (49.02)*] with *M2KpArmCur (49.14)* = 3, if the current error is 5 % of *M2NomCur (49.02)*. **Int. Scaling: 100 == 1 Type: I Volatile: N**	0	100	0.1	-	E
49.15	**M2TiArmCur (motor 2 i-part armature current controller)** Integral time of the current controller. *M2TiArmCur (49.15)* defines the time within the integral part of the controller achieves the same value as the proportional part. Example: The controller generates 15 % of motor nominal current [*M2NomCur (49.02)*] with *M2KpArmCur (49.14)* = 3, if the current error is 5 % of *M2NomCur (49.02)*. On that condition and with *M2TiArmCur (49.15)* = 50 ms follows: – the controller generates 30 % of motor nominal current, if the current error is constant, after 50 ms are elapsed (15 % from proportional part and 15 % from integral part). Setting *M2TiArmCur (49.15)* to 0 ms disables the integral part of the current controller and resets its integrator. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	50	ms	E
49.16	**M2DiscontCurLim (motor 2 discontinuous current limit)** Threshold continuous / discontinuous current in percent of *M2NomCur (49.02)*. The actual continuous / discontinuous current state can be read from *CurCtrlStat1 (6.03)* bit 12. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	0	325	100	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.17	**M2ArmL (motor 2 armature inductance)** Inductance of the armature circuit in mH. Used for the EMF compensation: $EMF = U_A - R_A * I_A - L_A * \frac{dI_A}{dt}$ **Attention:** Do not change the default values of *M2ArmL (49.17)* and *M2ArmR (49.18)*! Changing them will falsify the results of the autotuning. **Int. Scaling: 100 == 1 mH Type: I Volatile: N**	0	640	0	mH	E
49.18	**M2ArmR (motor 2 armature resistance)** Resistance of the armature circuit in mΩ. Used for the EMF compensation: $EMF = U_A - R_A * I_A - L_A * \frac{dI_A}{dt}$ **Attention:** Do not change the default values of *M2ArmL (49.17)* and *M2ArmR (49.18)*! Changing them will falsify the results of the autotuning. **Int. Scaling: 1 == 1 mΩ Type: I Volatile: N**	0	65500	0	mΩ	E
49.19	**M2SpeedMin (motor 2 minimum speed)** Motor 2 negative speed reference limit in rpm for: – *SpeedRef2 (2.01)* – *SpeedRefUsed (2.17)* Internally limited from: $-(2.29) * \frac{32767}{20000} rpm \ \ to \ \ (2.29) * \frac{32767}{20000} rpm$ **Note:** *M2SpeedMin (49.19)* is must be set in the range of: 0.625 to 5 times of *M1BaseSpeed (99.04).* If the scaling is out of range **A124 SpeedScale** [*AlarmWord2 (9.07)* bit 7] is generated. **Note:** *M2SpeedMin (49.19)* is also applied to *SpeedRef4 (2.18*) to avoid exceeding the speed limits by means of *SpeedCorr (23.04).* To be able to overspeed the drive (e.g. for winder) it is possible to switch off the speed limit for *SpeedRef4 (2.18)* by means of *AuxCtrlWord (7.02)* bit 4. **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	-1500	rpm	E
49.20	**M2SpeedMax (motor 2 maximum speed)** Motor 2 positive speed reference limit in rpm for: – *SpeedRef2 (2.01)* – *SpeedRefUsed (2.17)* Internally limited from: $-(2.29) * \frac{32767}{20000} rpm \ \ to \ \ (2.29) * \frac{32767}{20000} rpm$ **Note:** *M2SpeedMax (49.20)* is must be set in the range of: 0.625 to 5 times of *M1BaseSpeed (99.04).* If the scaling is out of range **A124 SpeedScale** [*AlarmWord2 (9.07)* bit 7] is generated. **Note:** *M2SpeedMax (49.20)* is also applied to *SpeedRef4 (2.18*) to avoid exceeding the speed limits by means of *SpeedCorr (23.04).* To be able to overspeed the drive (e.g. for winder) it is possible to switch off the speed limit for *SpeedRef4 (2.18)* by means of *AuxCtrlWord (7.02)* bit 4. **Int. Scaling: (2.29) Type: SI Volatile: N**	-10000	10000	1500	rpm	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.21	**M2OvrSpeed (motor 2 overspeed)** The drive trips with **F532 MotOverSpeed** [*FaultWord2 (9.02)* bit 15] if *M2OvrSpeed (49.21)* is exceeded. It is recommended to set *M2OvrSpeed (49.21)* at least 20 % higher than the maximum motor speed. Internally limited from: $0rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ The overspeed fault for motor 2 is inactive, if *M2OvrSpeed (49.21)* is set to zero. **Int. Scaling: (2.29) Type: I Volatile: N**	0	10000	1800	rpm	E
49.22	**M2SpeedScale (motor 2 speed scaling)** Motor 2 speed scaling in rpm. *M2SpeedScale (49.22)* defines the speed - in rpm - that corresponds to 20.000 speed units. The speed scaling is released when *M2SpeedScale (49.22)* ≥ 10: – 20.000 speed units == *M2SpeedScale (49.22)*, in case *M2SpeedScale (49.22)* ≥ 10 – 20.000 speed units == maximum absolute value of *M2SpeedMin (49.19)* and *M2SpeedMax (49.20),* in case *M2SpeedScale (49.22)* < 10 or mathematically – If *(49.22)* ≥ 10 then 20.000 == *(49.22)* in rpm – If *(49.22)* < 10 then 20.000 == Max [|*(49.19)*|, |*(49.20)*|] in rpm The actual used speed scaling is visible in *SpeedScale Act (2.29).* **Note:** *M2SpeedScale (49.22)* has to be set in case the speed is read or written by means of an overriding control (e.g. fieldbus). **Note:** *M2SpeedScale (49.22)* is must be set in the range of: 0.625 to 5 times of *M2BaseSpeed (49.03).* If the scaling is out of range **A124 SpeedScale** [*AlarmWord2 (9.07)* bit 7] is generated. **Commissioning hint:** – set *M2SpeedScale (49.22)* to maximum speed – set *M2BaseSpeed (49.03)* to base speed – set *M2SpeedMax (49.20)* / *M2SpeedMin (49.19)* to ±maximum speed **Int. Scaling: 10 == 1 rpm Type: I Volatile: N**	0	6500	0	rpm	E
49.23	**M2EncMeasMode (motor 2 encoder 1 measuring mode)** *M2EncMeasMode (49.23)* selects the measurement mode for pulse encoder 1: 0 = **A+/B Dir** channel A: rising edges for speed; channel A not: not used; channel B: direction; channel B not: not used; speed evaluation factor = 1 1 = **A+-** channels A and A not: rising and falling edges for speed; channels B and B not: not used; speed evaluation factor = 2 2 = **A+-/B Dir** channels A and A not: rising and falling edges for speed; channel B: direction; channel B not: not used; speed evaluation factor = 2 3 = **A+-/B+-** channels A, A not and B, B not: rising and falling edges for speed and direction; speed evaluation factor = 4, default **Int. Scaling: 1 == 1 Type: C Volatile: N**	A+/B Dir	A+-/B+-	A+-/B+-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.24	**M2SpeedFbSel (motor 2 speed feedback selector)** Motor 2 speed feedback selection: 0 = **EMF** speed is calculated by means of the EMF feedback with flux compensation, default 1 = **Encoder** speed is measured by means of pulse encoder 1 connected to either SDCS-CON-4 or SDCS-IOB-3 2 = **Tacho** speed is measured by means of an analog tacho 3 = **External** *MotSpeed (1.04)* is updated by Adaptive Program, application program or overriding control. 4 = **Encoder2** speed is measured by means of pulse encoder 2 connected to a RTAC-xx, see *Encoder2Module (98.01)* 5 = **EMF Volt** speed is calculated by means of the EMF feedback without flux compensation **Note1:** It is not possible to go into field weakening range when *M1SpeeFbSel (50.03)* = **EMF**. **Note2:** When using EMF speed feedback together with a DC-breaker wrong voltage measurements can lead to **F532 MotOverSpeed** [*FaultWord2 (9.02)* bit 15]. In case of an open DC-breaker the voltage measurement might show high values caused by leakage currents through the snubber circuits of the thyristors, because there is no load on the DC side. To prevent these trips set *MainContAck (10.21)* = **DCcontact**. **Int. Scaling: 1 == 1 Type: C Volatile: N**	EMF	EMF Volt	EMF	-	E
49.25	**M2EncPulseNo (motor 2 encoder 1 pulse number)** Amount of pulses per revolution (ppr) for pulse encoder 1. **Int. Scaling: 1 == 1 ppr Type: I Volatile: N**	20	10000	1024	ppr	E
49.26	**M2TachoAdjust (motor 2 tacho adjust)** Fine tuning of analog tacho. The value equals the actual speed measured by means of a hand held tacho: – *M2TachoAdjust (49.26)* = speed actual$_{HandHeldTacho}$ Internally limited to: $\pm(2.29)*\frac{32767}{20000}rpm$ **Note:** Changes of *M2TachoAdjust (49.26)* are only valid during tacho fine tuning [*ServiceMode (99.06)* = **TachFineTune**]. During tacho fine tuning *M2SpeedFbSel (49.24)* is automatically forced to **EMF**. **Attention:** The value of *M2TachoAdjust (49.26)* has to be the speed measured by the hand held tacho and **not** the delta between speed reference and measured speed. **Int. Scaling: (2.29) Type: I Volatile: Y**	-10000	10000	0	rpm	E
49.27	**M2TachoVolt1000 (motor 2 tacho voltage at 1000 rpm)** *M2TachoVolt1000 (49.27)* is used to adjust the voltage the analog tacho is generating at a speed of 1000 rpm: – *M2TachoVolt1000 (49.27)* ≥ 1 V, the setting is used to calculate tacho gain – *M2TachoVolt1000 (49.27)* = 0 V, the tacho gain is measured by means of the speed feedback assistant – *M2TachoVolt1000 (49.27)* = -1 V, the tacho gain was successfully measured by means of the speed feedback assistant **Note:** Use *ServiceMode (99.06)* = **TachFineTune** **Int. Scaling: 10 == 1 V Type: I Volatile: N**	-1	270	0	V	E

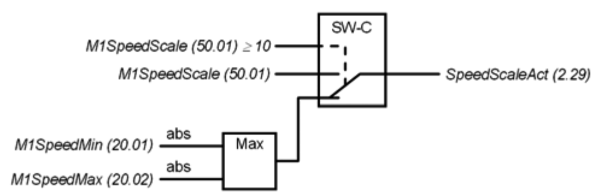
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.28	**M2BrakeCtrl (motor 2 brake control)** Releases the control of motor 2 brake: 0 = **NotUsed** brake logic is blocked, default 1 = **On** brake logic is released according to it's parameter settings 2 = **BrakeClose** test mode, the brake logic will work, but the brake is always closed (applied) 3 = **BrakeOpen** test mode, the brake logic will work, but the brake is always opened (lifted) **Attention**: A closed (applied) brake will open (lift) immediately! Do **not** use this mode with e.g. an unsaved crane drive! The brake open (lift) command **BrakeCmd** is readable in *AuxStatWord (8.02)* bit 8 and can be connected to the digital output controlling the brake. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	BrakeOpen	NotUsed	-	E
49.29	**M2BrakeAckSel (motor 2 brake acknowledge selector)** The drive sets either **A122 MechBrake** [*AlarmWord2 (9.07)* bit 5], **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] or **A116 BrakeLongFalling** [*AlarmWord1 (9.06)* bit 15] depending on *BrakeFaultFunc (42.06)* if a digital input is selected and the brake acknowledge fails: 0 = **NotUsed** brake acknowledge is blocked, default 1 = **DI1** 0 = brake is closed (applied), 1 = brake is open (lifted) 2 = **DI2** 0 = brake is closed (applied), 1 = brake is open (lifted) 3 = **DI3** 0 = brake is closed (applied), 1 = brake is open (lifted) 4 = **DI4** 0 = brake is closed (applied), 1 = brake is open (lifted) 5 = **DI5** 0 = brake is closed (applied), 1 = brake is open (lifted) 6 = **DI6** 0 = brake is closed (applied), 1 = brake is open (lifted) 7 = **DI7** 0 = brake is closed (applied), 1 = brake is open (lifted) 8 = **DI8** 0 = brake is closed (applied), 1 = brake is open (lifted) 9 = **DI9** 0 = brake is closed (applied), 1 = brake is open (lifted), only available with digital extension board 10 = **DI10** 0 = brake is closed (applied), 1 = brake is open (lifted), only available with digital extension board 11 = **DI11** 0 = brake is closed (applied), 1 = brake is open (lifted), only available with digital extension board 12 = **MCW Bit11** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 11* 13 = **MCW Bit12** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 12* 14 = **MCW Bit13** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 13* 15 = **MCW Bit14** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 14* 16 = **MCW Bit15** 0 = brake is closed (applied), 1 = brake is open (lifted), *MainCtrlWord (7.01) bit 15* 17 = **ACW Bit12** 0 = brake is closed (applied), 1 = brake is open (lifted), *AuxCtrlWord (7.02) bit 12* 18 = **ACW Bit13** 0 = brake is closed (applied), 1 = brake is open (lifted), *AuxCtrlWord (7.02) bit 13* 19 = **ACW Bit14** 0 = brake is closed (applied), 1 = brake is open (lifted), *AuxCtrlWord (7.02) bit 14* 20 = **ACW Bit15** 0 = brake is closed (applied), 1 = brake is open (lifted), *AuxCtrlWord (7.02) bit 15* **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.30	**M2BrakeRefDly (motor 2 brake reference delay)** Brake open (lift) delay. This function compensates for the mechanical open (lift) delay of the brake. During the start - **Run** [*MainCtrlWord (7.01)* bit 3] = 1 - of the drive the speed reference is clamped (ramp output is set to zero) and the speed controller output is set to start torque [see *M2StrtTorqRefSel (49.44)*] until *M2BrakeRefDly (49.30)* is elapsed. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	0.1	s	E
49.31	**M2ZeroSpeedDly (motor 2 zero speed delay)** This function compensates for the time the drive needs to decelerate from *M2ZeroSpeedLim (49.04)* to actual speed = 0. Until *M2ZeroSpeedDly (49.31)* is elapsed the brake is kept open (lifted). **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	0	s	E
49.32	**M2ModelTime (motor 2 model time constant)** Thermal time constant for motor 2 with fan/forced cooling. The time within the temperature rises to 63% of its nominal value. The motor thermal model is blocked, if *M2ModelTime (49.32)* is set to zero. The value of *Mot2TempCalc (1.21)* is saved at power down of the drives electronics. With the very first energizing of the drives electronics the motor's ambient temperature is set to 30°C. ⚠ **WARNING!** The model does not protect the motor if it is not properly cooled e.g. due to dust and dirt. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	6400	240	s	E
49.33	**M2AlarmLimLoad (motor 2 alarm limit load)** The drive sets **A110 M2OverLoad** [*AlarmWord1 (9.06)* bit 9] if *M2AlarmLimLoad (49.33)* - in percent of *M2NomCur (49.02)* - is exceeded. Output value for motor 2 thermal model is *Mot2TempCalc (1.21)*. **Int. Scaling: 10 == 1 % Type: I Volatile: N**	10	325	102	%	E
49.34	**M2FaultLimLoad (motor 2 fault limit load)** The drive trips with **F510 M2OverLoad** [*FaultWord1 (9.01)* bit 9] if *M2FaultLimLoad (49.34)* - in percent of *M2NomCur (49.02)* - is exceeded. Output value for motor 2 thermal model is *Mot2TempCalc (1.21)*. **Int. Scaling: 10 == 1 % Type: I Volatile: N**	10	325	106	%	E

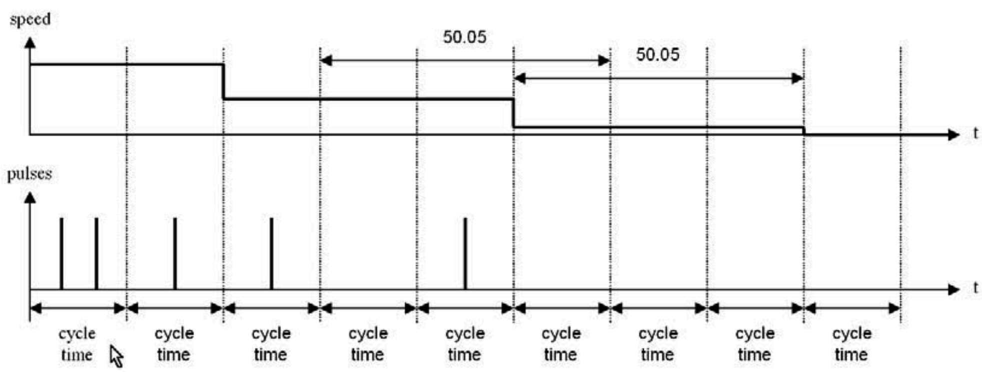
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.35	**M2TempSel (motor 2 temperature selector)** *M2TempSel (49.33)* selects motor 2 measured temperature input. The result can be seen in *Mot2TemopMeas (1.23)*. Connection possibilities for PT100: – max. 3 PT100 for motor 2 and max. 3 PT100 for motor 1 or – up to 6 PT100 for motor 2 only. Connection possibilities PTC: – max. 1 PTC for motor 2 and max. 1 PTC for motor 1 or – up to 2 PTC for motor 2 only: 0 = **NotUsed** motor 2 temperature measurement is blocked, default 1 = **1PT100 AI3** one PT100 connected to AI3 on SDCS-IOB-3 2 = **2PT100 AI3** two PT100 connected to AI3 on SDCS-IOB-3 3 = **3PT100 AI3** three PT100 connected to AI3 on SDCS-IOB-3 4 = **4PT100 AI3/2** four PT100, 3 connected to AI3 and 1 connected to AI2 on SDCS-IOB-3 5 = **5PT100 AI3/2** five PT100, 3 connected to AI3 and 2 connected to AI2 on SDCS-IOB-3 6 = **6PT100 AI3/2** six PT100, 3 connected to AI3 and 3 connected to AI2 on SDCS-IOB-3 7 = **1PT100 AI8** one PT100 connected to AI8 on RAIO2 8 = **2PT100 AI8** two PT100 connected to AI8 on RAIO2 9 = **3PT100 AI8** three PT100 connected to AI8 on RAIO2 10 = **4PT100 AI8/7** four PT100, 3 connected to AI8 and 1 connected to AI7 on RAIO2 11 = **5PT100 AI8/7** five PT100, 3 connected to AI8 and 2 connected to AI7 on RAIO2 12 = **6PT100 AI8/7** six PT100, 3 connected to AI8 and 3 connected to AI7 on RAIO2 13 = **1PTC AI3** one PTC connected to AI3 on SDCS-IOB-3 14 = **2PTC AI3/2** two PTC, 1 connected to AI3 and 1 connected to AI2 on SDCS-IOB-3 15 = **1PTC AI2/Con** one PTC connected to AI2 on SDCS-CON-4 For more information see section *Motor protection*. **Note:** AI7 and AI8 have to be activated by means of *AIO ExtModule (98.06)*. **Note:** In case only one PT100 is connected to an AI of the SDCS-IOB-3 the input range must be configured by jumpers to a gain of 10. Jumper settings for input range and constant current source see *DCS800 Hardware Manual*. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	1PTC AI2/Con	NotUsed	-	E
49.36	**M2AlarmLimTemp (motor 2 alarm limit temperature)** The drive sets **A108 M2OverTemp** [*AlarmWord1 (9.06)* bit 8] if *M2AlarmLimTemp (49.36)* is exceeded. Output value for motor 2 measured temperature is *Mot2TempMeas (1.23)*. **Note:** The unit depends on *M2TempSel (49.35)*. **Int. Scaling: 1 == 1 °C / 1 Ω / 1 Type: SI Volatile: N**	-10	4000	0	°C	E
49.37	**M2FaultLimTemp (motor 2 fault limit temperature)** The drive trips with **F509 M2OverTemp** [*FaultWord1 (9.01)* bit 8] if *M2FaultLimTemp (49.37)* is exceeded. Output value for motor 2 measured temperature is *Mot2TempMeas (1.23)*. **Note:** The unit depends on *M2TempSel (49.35)*. **Int. Scaling: 1 == 1 °C / 1 Ω / 1 Type: SI Volatile: N**	-10	4000	0	°C	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.38	**M2KlixonSel (motor 2 klixon selector)** The drive trips with **F509 M2OverTemp** [*FaultWord1 (9.01)* bit 8] if a digital input selected and the klixon is open: 0 = **NotUsed** no reaction, default 1 = **DI1** 0 = fault, 1 = no fault 2 = **DI2** 0 = fault, 1 = no fault 3 = **DI3** 0 = fault, 1 = no fault 4 = **DI4** 0 = fault, 1 = no fault 5 = **DI5** 0 = fault, 1 = no fault 6 = **DI6** 0 = fault, 1 = no fault 7 = **DI7** 0 = fault, 1 = no fault 8 = **DI8** 0 = fault, 1 = no fault 9 = **DI9** 0 = fault, 1 = no fault. Only available with digital extension board 10 = **DI10** 0 = fault, 1 = no fault. Only available with digital extension board 11 = **DI11** 0 = fault, 1 = no fault. Only available with digital extension board **Note:** It is possible to connect several klixons in series. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	DI11	NotUsed	-	E
49.39	**M2BrakeFltTime (motor 2 brake fault time)** Brake open (lift) acknowledge monitor. During this time the brake open (lift) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] and the brake acknowledge signal [*M2BrakeAckSel (49.29)*] can be different without causing **A122 MechBrake** [*AlarmWord2 (9.07)* bit 5] or **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] depending on *BrakeFaultFunc (42.06)*. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	1	s	E
49.40	**M2TorqProvTime (motor 2 torque proving time)** Brake torque proving acknowledge. The drive trips with **F556 TorqProv** [*FaultWord4 (9.04)* bit 7] if the **Run** [*MainCtrlWord (7.01)* bit 3] command is set and the acknowledge **TorqProvOK** [*AuxCtrlWord2 (7.03)* bit 11] is not set before *M2TorqProvTime (49.40)* is elapsed. The torque proving is inactive, if *M2TorqProvTime (49.40)* is set to 0. **Note:** The acknowledge signal **TorqProvOK** has to be provided by Adaptive Program, application program or overriding control and is set by means of a rising edge (0 $\rightarrow$ 1). The torque reference might be set by means of *BalRef (24.11)* or *TorqSel (26.01)* and **BalSpeedCtrl** [*AuxCtrlWord (7.02)* bit 8] or *TorqRefA (25.01)*. The reaction of the drive might be taken from *MotCur (1.06)*. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	100	0	s	E
49.41	**M2BrakeLiftDly (motor 2 brake lift delay)** Brake open (lift) delay. This function delays the brake open (lift) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] until *M2BrakeLiftDly (49.41)* is elapsed. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	0	s	E
49.42	**M2BrakeLongTime (motor 2 brake long time)** Brake close (apply) acknowledge monitor. During this time the brake close (apply) command **BrakeCmd** [*AuxStatWord (8.02)* bit 8] and the brake acknowledge signal [*M2BrakeAckSel (49.29)*] can be different without causing either **A122 MechBrake** [*AlarmWord2 (9.07)* bit 5], **F552 MechBrake** [*FaultWord4 (9.04)* bit 3] or **A116 BrakeLongFalling** [*AlarmWord1 (9.06)* bit 15] depending on *BrakeFaultFunc (42.06)*. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	4	s	E
49.43	**M2BrakeStopDly (motor 2 brake stop delay)** Brake close (apply) delay. This function starts after the brake acknowledge - if selected with *M2BrakeAckSel (49.29)* - is zero and compensates for the mechanical close (apply) delay of the brake. During the stop - **Run** [*MainCtrlWord (7.01)* bit 3] = 0 - of the drive the speed reference is clamped (ramp output is set to zero) and the speed controller stays active until *M2BrakeStopDly (49.43)* is elapsed. **Int. Scaling: 10 == 1 s Type: I Volatile: N**	0	60	1	s	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
49.44	**M2StrtTorqRefSel (motor 2 start torque reference selector)** Motor 2, start torque selector: 0 = **NotUsed** start torque function is blocked and the start torque reference is fixed zero, default 1 = **Memory** torque memory released, the minimum value equals the absolute value of *StrtTorqRef (42.08)* 2 = **StrtTorqRef** *StrtTorqRef (42.08)* 3 = **AI1** analog input AI1 4 = **AI2** analog input AI2 5 = **AI3** analog input AI3 6 = **AI4** analog input AI4 7 = **AI5** analog input AI5 8 = **AI6** analog input AI6 **Note:** Torque memory is the presetting of the torque when starting with e.g. suspended load. The preset torque equals the actual torque stored when the brake open (lift) command is removed. After energizing the drive the value of *StrtTorqRef (42.08)* is set as torque memory. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	AI6	NotUsed	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 50	**Speed measurement**					
50.01	**M1SpeedScale (motor 1 speed scaling)** Motor 1 speed scaling in rpm. *M1SpeedScale (50.01)* defines the speed - in rpm - that corresponds to 20.000 speed units. The speed scaling is released when *M1SpeedScale (50.01)* ≥ 10: – 20.000 speed units == *M1SpeedScale (50.01*), in case *M1SpeedScale (50.01)* ≥ 10 – 20.000 speed units == maximum absolute value of *M1SpeedMin (20.01)* and *M1SpeedMax (20.02),* in case *M1SpeedScale (50.01)* < 10 or mathematically – If *(50.01)* ≥ 10 then 20.000 == *(50.01)* in rpm – If *(50.01)* < 10 then 20.000 == Max [|*(20.01)*|, |*(20.02)*|] in rpm The actual used speed scaling is visible in *SpeedScale Act (2.29).*  **Note:** *M1SpeedScale (50.01)* has to be set in case the speed is read or written by means of an overriding control (e.g. fieldbus). **Note:** *M1SpeedScale (50.01)* is must be set in the range of: 0.625 to 5 times of *M1BaseSpeed (99.04)*, because the maximum amount of speed units is 32.000. If the scaling is out of range **A124 SpeedScale** [*AlarmWord2 (9.07)* bit 7] is generated. **Commissioning hint:** – set *M1SpeedScale (50.01)* to maximum speed – set *M1BaseSpeed (99.04)* to base speed – set *M1SpeedMax (20.02)* / *M1SpeedMin (20.01)* to ± maximum speed **Int. Scaling: 10 == 1 rpm Type: I Volatile: N**	0	6500	0	rpm	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
50.02	**M1EncMeasMode (motor 1 encoder 1 measuring mode)** *M1EncMeasMode (50.02)* selects the measurement mode for pulse encoder 1: 0 = **A+/B Dir** channel A: rising edges for speed; channel A not: not used; channel B: direction; channel B not: not used; speed evaluation factor = 1 1 = **A+-** channels A and A not: rising and falling edges for speed; channels B and B not: not used; speed evaluation factor = 2 2 = **A+-/B Dir** channels A and A not: rising and falling edges for speed; channel B: direction; channel B not: not used; speed evaluation factor = 2 3 = **A+-/B+-** channels A, A not and B, B not: rising and falling edges for speed and direction; speed evaluation factor = 4, default **Int. Scaling: 1 == 1 Type: C Volatile: N**	A+/B Dir	A+-/B+-	A+-/B+-	-	E
50.03	**M1SpeedFbSel (motor 1speed feedback selector)** Motor 1 speed feedback selection: 0 = **EMF** speed is calculated by means of the EMF feedback with flux compensation, default 1 = **Encoder** speed is measured by means of pulse encoder 1 connected to either SDCS-CON-4 or SDCS-IOB-3 2 = **Tacho** speed is measured by means of an analog tacho 3 = **External** *MotSpeed (1.04)* is updated by Adaptive Program, application program or overriding control. 4 = **Encoder2** speed is measured by means of pulse encoder 2 connected to a RTAC-xx, see *Encoder2Module (98.01)* 5 = **EMF Volt** speed is calculated by means of the EMF feedback without flux compensation **Note1:** It is not possible to go into field weakening range when *M1SpeeFbSel (50.03)* = **EMF**. **Note2:** When using EMF speed feedback together with a DC-breaker wrong voltage measurements can lead to **F532 MotOverSpeed** [*FaultWord2 (9.02)* bit 15]. In case of an open DC-breaker the voltage measurement might show high values caused by leakage currents through the snubber circuits of the thyristors, because there is no load on the DC side. To prevent these trips set *MainContAck (10.21)* = **DCcontact**. **Int. Scaling: 1 == 1 Type: C Volatile: N**	EMF	EMF Volt	EMF	-	C
50.04	**M1EncPulseNo (motor 1 encoder 1 pulse number)** Amount of pulses per revolution (ppr) for pulse encoder 1 **Int. Scaling: 1 == 1 ppr Type: I Volatile: N**	20	10000	1024	ppr	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
50.05	**MaxEncoderTime (maximum encoder time)** When an encoder is used as speed feedback device the actual speed is measured by counting the amount of pulses per cycle time. The cycle time for the measurement is synchronized with the mains (every 3.3 ms or 2.77 ms). In case very small speeds have to be measured - that means there is less than one pulse per cycle time - it is possible to increase the measuring time by means of *MaxEncoderTime (50.05)*. The speed is set to zero after *MaxEncoderTime (50.05)* is elapsed without a measured pulse.  **Note:** *MaxEncoderTime (50.05)* is valid for motor 1, motor 2, encoder 1 and encoder 2. **Note:** Formula to calculate the maximum speed using an encoder: $n_{\max}[rpm] = \frac{300\ kHz * 60\ s}{ppr}$ with: ppr = pulses per revolution - see *M1EncPulseNo (50.04)* 300 kHz are the maximum allowed input frequency **Note:** Formula to calculate the minimum speed resolution using an encoder: $n_{\min}[rpm] = \frac{60\ s}{k * ppr * t_{cycle}}$ with: k = speed evaluation factor - see *M1EncMeasMode (50.02)* ppr = pulses per revolution - see *M1EncPulseNo (50.04)* t_{cycle} = cycle time of the speed controller, either 3.3 ms or 2.77 ms **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	3	200	3	ms	E
50.06	**SpeedFiltTime (actual speed filter time)** Speed actual filter time for *MotSpeed (1.04)*. There are three different filters for actual speed and speed error (Δn). *SpeedFiltTime (50.06)* is filtering the actual speed and should be used for filter times smaller than 30 ms. *SpeedErrFilt (23.06)* and *SpeedErrFilt2 (23.11)* are filtering the speed error (Δn) and should be used for filter times greater than 30 ms. It is recommended to set *SpeedErrFilt (23.06)* = *SpeedErrFilt2 (23.11)*. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	5	ms	E

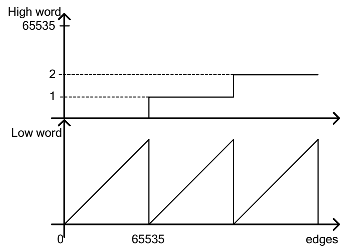
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
50.07	**PosCountMode (position counter mode)** The position counter is based on the pulse count of pulse encoder 1 and / or pulse encoder 2, with all pulse edges are counted. The 32-bit position value is divided into two 16-bit words for each pulse encoder: 0 = **PulseEdges** for the low words *PosCountLow (3.07), PosCount2Low (3.04)*, *PosCountInitLo (50.08)* and *PosCount2InitLo (50.21)* is valid: 1 == 1 pulse edge for the high words *PosCountHigh (3.08)*, *PosCount2High (3.05)*, *PosCountInitHi (50.09)* and *PosCount2InitHi (50.22)* is valid: 1 == 65536 pulse edges 1 = **Scaled** for the low words *PosCountLow (3.07), PosCount2Low (3.04)*, *PosCountInitLo (50.08)* and *PosCount2InitLo (50.21)* is valid: 0 == 0° and 65536 == 360° for the high words *PosCountHigh (3.08)*, *PosCount2High (3.05)*, *PosCountInitHi (50.09)* and *PosCount2InitHi (50.22)* is valid: 1 == 1 revolution, default 2 = **Rollover** for the low words *PosCountLow (3.07), PosCount2Low (3.04)*, *PosCountInitLo (50.08)* and *PosCount2InitLo (50.21)* is valid: 0 == 0° and 65536 == 360° for the high words *PosCountHigh (3.08)*, *PosCount2High (3.05)*, *PosCountInitHi (50.09)* and *PosCount2InitHi (50.22)* is valid: always 0	PulseEdges	Rollover	Scaled	-	E

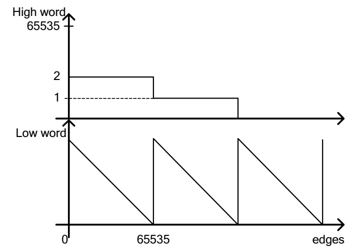
Index	Signal / Parameter name	min.	max.	def.	unit	E/C

Forward direction:

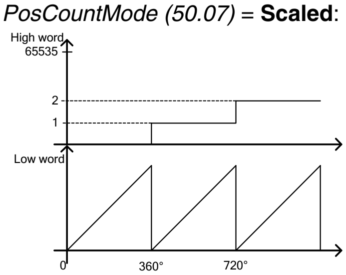
Reverse direction:

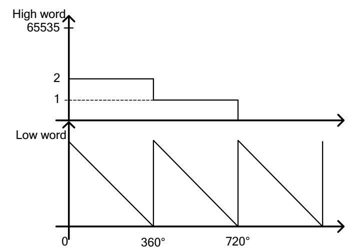
PosCountMode (50.07) = **PulseEdges**:



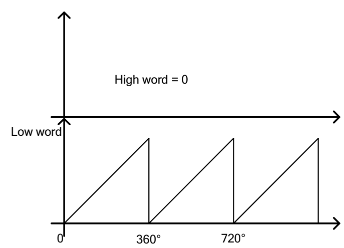


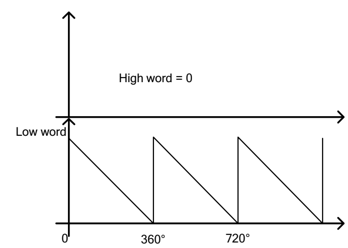
PosCountMode (50.07) = **Scaled**:





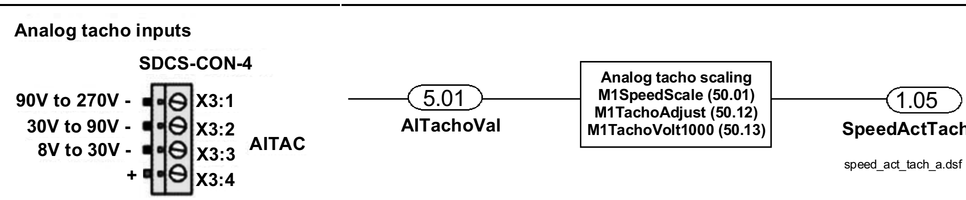
PosCountMode (50.07) = **Rollover**:





The position counter is controlled by *SyncCommand (10.04)*, *SyncCommand2 (10.05)* and *AuxCtrlWord (7.02)* bits 9 to 11.
The status can be seen from *AuxStatWord (8.02)* bit 5 **SyncRdy**.
The position control function has to be implemented by Adaptive Program, application program or overriding control.
Int. Scaling: 1 == 1 Type: C Volatile: N

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
50.08	**PosCountInitLo (Position counter encoder 1 low initial value)** Position counter initial low word for pulse encoder 1. Unit depends on setting of *PosCountMode (50.07):* – **PulseEdges** 1 == 1 pulse edge – **Scaled** 0 == 0° and 65536 == 360° – **Rollover** 0 == 0° and 65536 == 360° See also *SyncCommand (10.04)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	65536	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
50.09	**PosCountInitHi (Position counter encoder 1 high initial value)** Position counter initial high word for pulse encoder 1. Unit depends on setting of *PosCountMode (50.07)*: – **PulseEdges** 1 == 65536 pulse edges – **Scaled** 1 == 1 revolution – **Rollover** always 0 See also *SyncCommand (10.04)*. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
50.10	**SpeedLev (speed level)** When *MotSpeed (1.04)* reaches *SpeedLev (50.10)* the bit **AboveLimit** [*MainStatWord (8.01)* bit 10] is set. Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Note:** With *SpeedLev (50.10)* it is possible to automatically switch between the two p- and i-parts of the speed controller, see *Par2Select (24.29)* = **SpeedLevel** or **SpeedError**. **Int. Scaling: (2.29) Type: I Volatile: N**	0	10000	1500	rpm	E
50.11	**DynBrakeDly (dynamic braking delay)** In case of dynamic braking with EMF feedback [*M1SpeedFbSel (50.03)* = **EMF**] or a speed feedback fault there is no valid information about the motor speed and thus no zero speed information. To prevent an interlocking of the drive after dynamic braking the speed is assumed zero after *DynBrakeDly (50.11)* is elapsed: **-1 s** = the motor voltage is measured directly at the motor terminals and is thus valid during dynamic braking **0 s** = no zero speed signal for dynamic braking is generated **1 s to 3000 s** = zero speed signal for dynamic braking is generated after the programmed time is elapsed **Int. Scaling: 1 == 1 s Type: I Volatile: N**	-1	3000	0	s	E
						
50.12	**M1TachoAdjust (motor 1 tacho adjust)** Fine tuning of analog tacho. The value equals the actual speed measured by means of a hand held tacho: – *M1TachoAdjust (50.12)* = speed $actual_{HandHeldTacho}$ Internally limited to: $\pm(2.29)*\frac{32767}{20000}rpm$ **Note:** Changes of *M1TachoAdjust (50.12)* are only valid during tacho fine tuning [*ServiceMode (99.06)* = **TachFineTune**]. During tacho fine tuning *M1SpeedFbSel (50.03)* is automatically forced to **EMF**. **Attention:** The value of *M1TachoAdjust (50.12)* has to be the speed measured by the hand held tacho and **not** the delta between speed reference and measured speed. **Int. Scaling: (2.29) Type: I Volatile: Y**	-10000	10000	0	rpm	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
50.13	**M1TachoVolt1000 (motor 1 tacho voltage at 1000 rpm)** *M1TachoVolt1000 (50.13)* is used to adjust the voltage the analog tacho is generating at a speed of 1000 rpm: – *M1TachoVolt1000 (50.13)* ≥ 1 V, the setting is used to calculate the tacho gain – *M1TachoVolt1000 (50.13)* = 0 V, the tacho gain is measured by means of the speed feedback assistant – *M1TachoVolt1000 (50.13)* = -1 V, the tacho gain was successfully measured and set by means of the speed feedback assistant **Int. Scaling: 10 == 1 V Type: I Volatile: N**	0	270	60	V	C
50.14	**Unused**					
50.15	**PosSyncMode (position counter synchronization mode)** Position counter synchronization mode for pulse encoder 1 and / or pulse encoder 2 [depends on the setting of *SyncCommand (10.04)* and *SyncCommand2 (10.05)*]: 0 = **Single** the next synchronization of the pulse encoders must be prepared by resetting **SyncRdy** [*AuxStatWord (8.02)* bit 5] with **ResetSyncRdy** [*AuxCtrlWord (7.02)* bit 11], default 1 = **Cyclic** the synchronization of the pulse encoders happens on every occurrence of the synchronization event **Int. Scaling: 1 == 1 Type: C Volatile: N**	Single	Cyclic	Single	-	E
50.16	**Unused**					
50.17	**WinderScale (winder scaling)** Speed actual scaling. Before speed error (Δn) generation. **Int. Scaling: 100 == 1 Type: I Volatile: N**	-100	100	1	-	E
50.18	**Enc2MeasMode (encoder 2 measuring mode)** *Enc2MeasMode (50.18)* selects the measurement mode for pulse encoder 2: 0 = **A+/B Dir** channel A: rising edges for speed; channel A not: not used; channel B: direction; channel B not: not used; speed evaluation factor = 1 1 = **A+-** channels A and A not: rising and falling edges for speed; channels B and B not: not used; speed evaluation factor = 2 2 = **A+-/B Dir** channels A and A not: rising and falling edges for speed; channel B: direction; channel B not: not used; speed evaluation factor = 2 3 = **A+-/B+-** channels A, A not and B, B not: rising and falling edges for speed and direction; speed evaluation factor = 4, default **Int. Scaling: 1 == 1 Type: C Volatile: N**	A+/B Dir	A+-/B+-	A+-/B+-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
50.19	**Enc2PulseNo (encoder 2 pulse number)** Amount of pulses per revolution (ppr) for pulse encoder 2, if a pulse encoder extension module RTAC-xx is used. In case a resolver is connected via an extension module RRIA-xx *Enc2PulseNo (50.19)* defines the number of pole pairs. Following formula is valid: $Enc2PulseNo\,(50.19) = 1024 * number\ of\ \ pole\ pairs$ **Note:** The position counter 2 can be used with the resolver if following conditions are fulfilled: – number of pole pairs = 1 and thus *Enc2PulseNo (50.19)* = 1024, – *PosCountMode (50.07)* = **Rollover** and – the resolver's gear ratio is 1:1 (this can be adapted by means of the application program - see block PosSetGear) **Int. Scaling: 1 == 1 ppr Type: I Volatile: N**	20	10000	1024	ppr	C
50.20	**Unused**					
50.21	**PosCount2InitLo (Position counter encoder 2 low initial value)** Position counter initial low word for pulse encoder 2. Unit depends on setting of *PosCountMode (50.07):* – **PulseEdges** 1 == 1 pulse edge – **Scaled** 0 == 0° and 65536 == 360° – **Rollover** 0 == 0° and 65536 == 360° See also *SyncCommand2 (10.05)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	65536	0	-	E
50.22	**PosCount2InitHi (Position counter encoder 2 high initial value)** Position counter initial high word for pulse encoder 2. Unit depends on setting of *PosCountMode (50.07):* – **PulseEdges** 1 == 65536 pulse edges – **Scaled** 1 == 1 revolution – **Rollover** always 0 See also *SyncCommand2 (10.05)*. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
Group 51	**Fieldbus**					
	This parameter group defines the communication parameters for fieldbus adapters (F-type, R-type and N-type). The parameter names and the number of the used parameters depend on the selected fieldbus adapter (see fieldbus adapter manual). **Note:** If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter.					
51.01	**Fieldbus1 (fieldbus parameter 1)** Fieldbus parameter 1 **Int. Scaling: 1 == 1 Type: C Volatile: Y**	-	-	-	-	C
…	…					C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
51.15	**Fieldbus15 (fieldbus parameter 15)** Fieldbus parameter 15 **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	32767	0	-	C
51.16	**Fieldbus16 (fieldbus parameter 16)** Fieldbus parameter 16 **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	32767	0	-	C
…	…					C
51.27	**FBA PAR REFRESH (fieldbus parameter refreshing)** If a fieldbus parameter is changed its new value takes effect only upon setting *FBA PAR REFRESH (51.27)* = **RESET** or at the next power up of the fieldbus adapter. *FBA PAR REFRESH (51.27)* is automatically set back to **DONE** after the refreshing is finished. 0 = **DONE** default 1 = **RESET** refresh the parameters of the fieldbus adapter **Note:** This service is only available for R-type fieldbus adapters. **Int. Scaling: 1 == 1 Type: C Volatile: N**	DONE	RESET	DONE	-	C
…	…					C
51.36	**Fieldbus36 (fieldbus parameter 36)** Fieldbus parameter 36 **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	32767	0	-	C
Group 52	**Modbus**					
	This parameter group defines the communication parameters for the Modbus adapter RMBA-xx (see also Modbus adapter manual). **Note:** If a Modbus parameter is changed its new value takes effect only upon the next power up of the Modbus adapter.					
52.01	**StationNumber (station number)** Defines the address of the station. Two stations with the same station number are not allowed online. **Int. Scaling: 1 == 1 Type: I Volatile: N**	1	247	1	-	E
52.02	**BaudRate (baud rate)** Defines the transfer rate of the Modbus link: 0 = reserved 1 = **600** 600 Baud 2 = **1200** 1200 Baud 3 = **2400** 2400 Baud 4 = **4800** 4800 Baud 5 = **9600** 9600 Baud, default 6 = **19200** 19200 Baud **Int. Scaling: 1 == 1 Type: C Volatile: N**	600	19200	9600	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
52.03	**Parity (parity)** Defines the use of parity and stop bit(s). The same setting must be used in all online stations: 0 = reserved 1 = **None1Stopbit** no parity bit, one stop bit 2 = **None2Stopbit** no parity bit, two stop bits 3 = **Odd** odd parity indication bit, one stop bit 4 = **Even** even parity indication bit, one stop bit, default **Int. Scaling: 1 == 1 Type: C Volatile: N**	reserved	Even	Even	-	E
Group 60, ..., 69	**Application program parameters**					
	These parameter groups contain all parameters created by the application program.					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 70	**DDCS control**					
70.01	**Ch0 NodeAddr (channel 0 node address)** Channel 0 is used for communication with the overriding control. Node address channel 0: – if APC2 or NCSA-01 (AC31) is used *Ch0 NodeAddr (70.01)* = 1 – if AC70 or AC80 is used via the optical module bus (adapters TB810 or TB811) *Ch0 NodeAddr (70.01)* is calculated from the **POSTION** terminal of the **DRIENG** data base element as follows: 1. multiply the hundreds of the value **POSITION** by 16 2. add the tens and ones of the value **POSITION** to the result Example: **POSITION** I *Ch0 NodeAddr (70.01)* 101 I 16*1+01 = 17 712 I 16*7+12 = 124 – if AC 800M is used via the optical module bus *Ch0 NodeAddr (70.01)* is calculated from the position of the **DCS600 ENG** hardware module as follows: 1. multiply the hundreds of the value **POSITION** by 16 2. add the tens and ones of the value **POSITION** to the result Example: **POSITION** I *Ch0 NodeAddr (70.01)* 112 I 16*1+12 = 28 503 I 16*5+03 = 83	0	254	1	-	E

Controller	Node address DDCS	Node address DriveBus	Node address ModuleBus	*Ch0 DriveBus (71.01)*
APC / AC31	1	-	-	**No**
AC70	-	-	17-124	**No**
AC80 DriveBus	-	1-12	-	**Yes**
AC80 ModuleBus	-	-	17-124	**No**
FCI (CI810A)	-	-	17-124	**No**
CI858	-	1-12	-	**Yes**

Int. Scaling: 1 == 1 Type: I Volatile: N

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
70.02	**Ch0 LinkControl (channel 0 link control)** DDCS channel 0 light intensity control for transmission LEDs. When using the maximum allowed length of the fiber optic cable set the value to 15. **Int. Scaling: 1 == 1 Type: I Volatile: N**	1	15	10	-	E
70.03	**Ch0 BaudRate (channel 0 baud rate)** Channel 0 communication speed. *Ch0 BaudRate (70.03)* must be set to **4 Mbits/s** when ABB overriding control modules (e.g. FCI or AC 800M) are used. Otherwise the overriding control automatically sets the communication speed. 0 = **8 Mbits/s** 1 = **4 Mbits/s**, default 2 = **2 Mbits/s** 3 = **1 Mbits/s** **Int. Scaling: 1 == 1 Type: C Volatile: N**	8 Mbits/s	1 Mbits/s	4 Mbits/s	-	E

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>70.04</td><td>Ch0 TimeOut (channel 0 timeout)
Time delay before a communication loss with channel 0 is declared. Depending on the setting of Ch0 ComLossCtrl (70.05) either F543 COM8Com [FaultWord3 (9.03) bit 10] or A113 COM8Com [AlarmWord1 (9.06) bit 12] is set.
The communication fault and alarm are inactive, if Ch0 TimeOut (70.04) is set to 0 ms.
Note:
The supervision is activated after the reception of the first valid message.
Note:
The time out starts when the link doesn’t update any of the first 2 receive data sets addressed by Ch0 DsetBaseAddr (70.24).
Example:
When Ch0 DsetBaseAddr (70.24) = 10 the reception of data sets 10 and 12 is supervised.
Int. Scaling: 1 == 1 ms Type: I Volatile: N</td><td>0</td><td>64000</td><td>100</td><td>ms</td><td>E</td></tr>
<tr><td>70.05</td><td>Ch0 ComLossCtrl (channel 0 communication loss control)
Ch0 ComLossCtrl (70.05) determines the reaction to a communication loss of channel 0 control.
F543 COM8Com [FaultWord3 (9.03) bit 10] is set with:
0 = RampStop The input of the drives ramp is set to zero. Thus the drive stops according to DecTime1 (22.02) or DecTime2 (22.10). When reaching M1ZeroSpeedLim (20.03) the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped, default.
In case TorqSelMod (26.03) = Auto and communication loss is active the torque selector is bypassed and the drive is forced to speed control, default.
1 = TorqueLimit The output of the drives ramp is set to zero. Thus the drive stops at the active torque limit. When reaching M1ZeroSpeedLim (20.03) the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped.
In case TorqSelMod (26.03) = Auto and communication loss is active the torque selector is bypassed and the drive is forced to speed control, default.
2 = CoastStop The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped.
3 = DynBraking dynamic braking
A113 COM8Com [AlarmWord1 (9.06) bit 12] is set with:
4 = LastSpeed the drive continues to run at the last speed before the warning
5 = FixedSpeed1 the drive continuous to run with FixedSpeed1 (23.02)
Note:
The time out for Ch0 ComLossCtrl (70.05) is set by:
– Ch0 TimeOut (70.04)
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>RampStop</td><td>FixedSpeed1</td><td>RampStop</td><td>-</td><td>E</td></tr>
<tr><td>70.06</td><td>CH0 HW Config (channel 0 hardware configuration)
CH0 HW Config (70.06) is used to enable / disable the regeneration of the Channel 0 optotransmitters in DDCS mode [Ch0 DriveBus (71.01) = No]. Regeneration means that the drive echoes all messages back. DDCS mode is typically used with APC2, AC70, AC80 and module bus of AC 800M.
0 = Ring Regeneration is enabled. Used with ring-type bus topology. Typically when Channel 0 of all SDCS-COM-8 has been connected to a ring.
1 = Star Regeneration is disabled. Used with star-type topology. Typically with configurations using the NDBU-x5 branching units, default
Note:
This parameter has no effect in DriveBus mode [Ch0 DriveBus (71.01) = Yes].
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>Ring</td><td>Star</td><td>Star</td><td>-</td><td>E</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
70.07	**Ch1 LinkControl (channel 1 link control)** Channel 1 is used for communication with the AIMA-xx adapter. DDCS channel 1 light intensity control for transmission LEDs. When using the maximum allowed length of the fiber optic cable set the value to 15. **Int. Scaling: 1 == 1 Type: I Volatile: N**	1	15	10	-	E
70.08	**Ch2 NodeAddr (channel 2 node address)** Channel 2 is used for point to point communication connections between drives (e.g. master-follower communication). Node address channel 2: 1, …, 125 = Node addresses of slave drives, not valid if *Ch2 MaFoMode (70.09)* = **Master** **Int. Scaling: 1 == 1 Type: I Volatile: N**	1	125	1	-	E
70.09	**Ch2 MaFoMode (channel 2 master-follower mode)** Channel 2 can be used to send reference values (e.g. torque reference) from the master to one or several followers. Master-follower is an application in which machinery is run by several drives with all motor shafts coupled to each other by gears, chains, belts etc. 0 = reserved 1 = **NotUsed** channel 2 is not used for master-follower communication, default 2 = **Master** the drive is the master of the master-follower link and broadcasts via channel 2 the contents of data set 41 [defined by *Ch2 MasSig1 (70.10)* to *Ch2 MasSig3 (70.12)*] 3 = **Follower** the drive is a follower of the master-follower link and receives via channel 2 the contents of data set 41 [defined by *Ch2 FolSig1 (70.18)* to *Ch2 FolSig3 (70.20)*] **Note:** The follower's node address is defined by *Ch2 NodeAddr (70.08)*. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	Follower	NotUsed	-	E
70.10	**Ch2 MasSig1 (channel 2 master signal 1)** Master signal 1 broadcasts via channel 2 as 1st value of data set 41 to all followers. The format is **xxyy**, with: **xx** = group and **yy** = index. Default setting of 701 equals *MainCtrlWord (7.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	701	-	E
70.11	**Ch2 MasSig2 (channel 2 master signal 2)** Master signal 2 broadcasts via channel 2 as 2nd value of data set 41 to all followers. The format is **xxyy**, with: **xx** = group and **yy** = index. Default setting of 2301 equals *SpeedRef (23.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	2301	-	E
70.12	**Ch2 MasSig3 (channel 2 master signal 3)** Master signal 3 broadcasts via channel 2 as 3rd value of data set 41 to all followers. The format is **xxyy**, with: **xx** = group and **yy** = index. Default setting of 210 equals *TorqRef3 (2.10)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	210	-	E
70.13	**Ch2 LinkControl (channel 2 link control)** DDCS channel 2 light intensity control for transmission LEDs. When using the maximum allowed length of the fiber optic cable set the value to 15. **Int. Scaling: 1 == 1 Type: I Volatile: N**	1	15	10	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
70.14	**Ch2 TimeOut (channel 2 timeout)** Time delay before a communication loss with channel 2 is declared. Depending on the setting of *Ch2 ComLossCtrl (70.15)* either **F543 COM8Com** [*FaultWord3 (9.03)* bit 10] or **A113 COM8Com** [*AlarmWord1 (9.06)* bit 12] is set. The communication fault and alarm are inactive, if *Ch2 TimeOut (70.14)* is set to 0 ms. **Note:** The supervision is activated after the reception of the first valid message. **Note:** The time out starts when the link doesn't update the master-follower data set. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	100	ms	E
70.15	**Ch2 ComLossCtrl (channel 2 communication loss control)** *Ch2 ComLossCtrl (70.15)* determines the reaction to a communication loss of channel 2. **F543 COM8Com** [*FaultWord3 (9.03)* bit 10] is set with: 0 = **RampStop** The input of the drives ramp is set to zero. Thus the drive stops according to *DecTime1 (22.02)* or *DecTime2 (22.10)*. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped, default. In case *TorqSelMod (26.03)* = **Auto** and communication loss is active the torque selector is bypassed and the drive is forced to speed control, default. 1 = **TorqueLimit** The output of the drives ramp is set to zero. Thus the drive stops at the active torque limit. When reaching *M1ZeroSpeedLim (20.03)* the firing pulses are set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. In case *TorqSelMod (26.03)* = **Auto** and communication loss is active the torque selector is bypassed and the drive is forced to speed control, default. 2 = **CoastStop** The firing pulses are immediately set to 150 degrees to decrease the armature current. When the armature current is zero the firing pulses are blocked, the contactors are opened, field exciter and fans are stopped. 3 = **DynBraking** dynamic braking **A113 COM8Com** [*AlarmWord1 (9.06)* bit 12] is set with: 4 = **LastSpeed** the drive continues to run at the last speed before the warning 5 = **FixedSpeed1** the drive continuous to run with *FixedSpeed1 (23.02)* **Note:** The time out for *Ch2 ComLossCtrl (70.15)* is set by: – *Ch2 TimeOut (70.14)* **Int. Scaling: 1 == 1 Type: C Volatile: N**	RampStop	FixedSpeed1	RampStop	-	E
70.16	**Unused**					
70.17	**Unused**					
70.18	**Ch2 FolSig1 (channel 2 follower signal 1)** Follower signal 1 receives via channel 2 the 1st value of data set 41 from the master. The format is **xxyy**, with: **xx** = group and **yy** = index. Default setting of 701 equals *MainCtrlWord (7.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	701	-	E
70.19	**Ch2 FolSig2 (channel 2 follower signal 2)** Follower signal 2 receives via channel 2 the 2nd value of data set 41 from the master. The format is **xxyy**, with: **xx** = group and **yy** = index. Default setting of 2301 equals *SpeedRef (23.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	2301	-	E

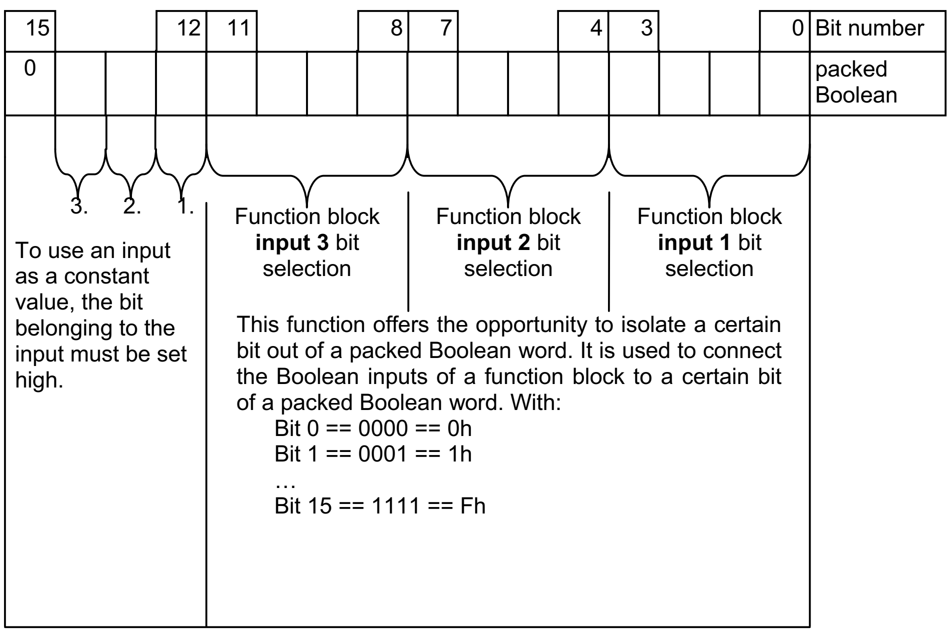
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
70.20	**Ch2 FolSig3 (channel 2 follower signal 3)** Follower signal 3 receives via channel 2 the 3rd value of data set 41 from the master. The format is **xxyy**, with: **xx** = group and **yy** = index. Default setting of 2501 equals *TorqRefA (25.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	2501	-	E
70.21	**Ch3 HW Config (channel 3 hardware configuration)** *CH3 HW Config (70.21)* is used to enable / disable the regeneration of the Channel 3 optotransmitters. Regeneration means that the drive echoes all messages back. 0 = **Ring** Regeneration is enabled. Used with ring-type bus topology. 1 = **Star** Regeneration is disabled. Used with star-type topology. Typically with configurations using the NDBU-x5 branching units, default **Note:** This parameter has no effect in DriveBus mode [*Ch0 DriveBus (71.01)* = **Yes**]. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Ring	Star	Star	-	E
70.22	**Ch3 NodeAddr (channel 3 node address)** Channel 3 is used for communication with start-up and maintenance tools (e.g. DriveWindow). If several drives are connected together via channel 3, each of them must be set to a unique node address. Node address channel 3: 0, …, 75 valid node address for SDCS-COM-8 76, …, 124 reserved node address for NDBU-x5 branching units 125, …, 254 valid node address for SDCS-COM-8 **Attention:** A new node address becomes only valid after the next SDCS-COM-8 power-up. **Int. Scaling: 1 == 1 Type: I Volatile: N**	1	254	1	-	E
70.23	**Ch3 LinkControl (channel 3 link control)** DDCS channel 3 light intensity control for transmission LEDs. When using the maximum allowed length of the fiber optic cable set the value to 15. **Int. Scaling: 1 == 1 Type: I Volatile: N**	1	15	15	-	E
70.24	**Ch0 DsetBaseAddr (channel 0 data set base address)** Data set number of the 1st data set used for the communication with the overriding control system (e.g. field bus adapters, ABB overriding control). The data set addressed by *Ch0 DsetBaseAddr (70.24)* is the 1st data set send from the overriding control to the drive, while the next - 2nd - data set is the first one send from the drive to the overriding control and so on. Up to 8 data sets for each direction are supported (addressing of the data sets see groups 90 to 93). Examples: – *Ch0 DsetBaseAddr(70.24)* = 1 data set range 1, …, 16 – *Ch0 DsetBaseAddr(70.24)* = 10 data set range 10, …, 25 **Note:** The data sets for the APC-mailbox function (32 and 33) as well as for the master-follower communication (41) are not programmable. **Int. Scaling: 1 == 1 Type: I Volatile: N**	1	16	10	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 71	**Drivebus**					
71.01	**Ch0 DriveBus (channel 0 drive bus)** Communication mode selection for channel 0. The DriveBus mode is used with the AC80 and AC 800M controllers. 0 = **No** DDCS mode (recommended when ModuleBus is used) 1 = **Yes** DriveBus mode, default **Note:** Before changing *Ch0 DriveBus (71.01)* the communication from the overriding control system has to be disabled e.g. by removing the fiber optic cables. **Note:** A new mode becomes only valid after the next SDCS-COM-8 power-up. **Int. Scaling: 1 == 1 Type: C Volatile: N**	No	Yes	Yes	-	E
Group 83	**Adaptive Program control**					
83.01	**AdapProgCmd (Adaptive Program command)** Selects the operation mode for the Adaptive Program: 0 = **Stop** stop, the Adaptive Program is not running and cannot be edited, default 1 = **Start** running, the Adaptive Program is running and cannot be edited 2 = **Edit** edit, the Adaptive Program is not running and can be edited 3 = **SingleCycle** The Adaptive Program runs only once. If a breakpoint is set with *BreakPoint (83.06)* the Adaptive Program will stop before the breakpoint. After the **SingleCycle** *AdapProgCmd (83.01)* is automatically set back to **Stop**. 4 = **SingleStep** Runs only one function block. *LocationCounter (84.03)* shows the function block number, which will be executed during the next **SingleStep**. After a **SingleStep** *AdapProgCmd (83.01)* is automatically set back to **Stop**. *LocationCounter (84.03)* shows the next function block to be executed. To reset *LocationCounter (84.03)* to the first function block set *AdapProgCmd (83.01)* to **Stop** again (even if it is already set to **Stop**). **A136 NoAPTaskTime** [*AlarmWord3 (9.08)* bit 3] is set when *TimeLevSel (83.04)* is not set to **5 ms, 20 ms, 100 ms** or **500 ms** but *AdapProgCmd (83.01)* is set to **Start**, **SingleCycle** or **SingleStep** **Note:** *AdapProgCmd (83.01)* = **Start**, **SingleCycle** or **SingleStep** is only valid, if *AdapPrgStat (84.01)* ≠ **Running**. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Stop	SingleStep	Stop	-	E

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>83.02</td><td>EditCmd (edit command)
Edit Adaptive Program. EditCmd (83.02) is automatically set back to Done after the chosen action is finished:
0 = Done no action or edit of Adaptive Program completed, default
1 = Push Shifts the function block in the spot defined by EditBlock (83.03) and all subsequent function blocks one spot forward. A new function block can be placed in the now empty spot by programming its parameter set as usual.
Example:
A new function block needs to be placed in between the function block number four (84.22) to (84.27) and five (84.28) to (84.33). In order to do this:
1. set AdapProgCmd (83.01) = Edit
2. set EditBlock (83.03) = 5 (selects function block 5 as the desired spot for the new function block)
3. set EditCmd (83.02) = Push (shifts function block 5 and all subsequent function blocks one spot forward)
4. Program empty spot 5 by means of (84.28) to (84.33)
2 = Delete Deletes the function block in the spot defined by EditBlock (83.03) and shifts all subsequent function blocks one spot backward. To delete all function blocks set EditBlock (83.03) = 17.
3 = Protect Turns all parameters of the Adaptive Program into protected mode (parameters cannot be read or written to). Before using the Protect command set the pass code by means of PassCode (83.05).
Attention: Do not forget the pass code!
4 = Unprotect Reset of protected mode. Before the Unprotect command can be used, PassCode (83.05) has to be set.
Attention: The proper pass code has to be used!
Int. Scaling: 1 == 1 Type: C Volatile: Y</td><td>Done</td><td>Unprotect</td><td>Done</td><td>-</td><td>E</td></tr>
<tr><td>83.03</td><td>EditBlock (edit block)
Defines the function block which is selected by EditCmd (83.02) = Push or Delete. After a Push or Delete EditBlock (83.03) is automatically set back to 1.
Note:
To delete all function blocks set EditBlock (83.03) = 17.
Int. Scaling: 1 == 1 Type: I Volatile: Y</td><td>0</td><td>17</td><td>0</td><td>-</td><td>E</td></tr>
<tr><td>83.04</td><td>TimeLevSel (time level select)
Selects the cycle time for the Adaptive Program. This setting is valid for all function blocks.
0 = Off no task selected
1 = 5ms Adaptive Program runs with 5 ms
2 = 20ms Adaptive Program runs with 20 ms
3 = 100ms Adaptive Program runs with 100 ms
4 = 500ms Adaptive Program runs with 500 ms
A136 NoAPTaskTime [AlarmWord3 (9.08) bit 3] is set when TimeLevSel (83.04) is not set to 5 ms, 20 ms, 100 ms or 500 ms but AdapProgCmd (83.01) is set to Start, SingleCycle or SingleStep.
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>Off</td><td>500ms</td><td>Off</td><td>-</td><td>E</td></tr>
<tr><td>83.05</td><td>PassCode (pass code)
The pass code is a number between 1 and 65535 to write protect Adaptive Programs by means of EditCmd (83.02). After using Protect or Unprotect PassCode (83.05) is automatically set back to zero.
Attention:
Do not forget the pass code!
Int. Scaling: 1 == 1 Type: I Volatile: Y</td><td>0</td><td>65535</td><td>0</td><td>-</td><td>E</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
83.06	**BreakPoint (break point)** Breakpoint for *AdapProgCmd (83.01)* = **SingleCycle**. The break point is not used, if *BreakPoint (83.06)* is set to zero. **Int. Scaling: 1 == 1 Type: I Volatile: Y**	0	16	0	-	E
Group 84	**Adaptive Program**					
84.01	**AdapPrgStat (Adaptive Program status word)** Adaptive Program status word: Bit Name Value Comment B0 **Bit 0** 1 Adaptive Program is running 0 Adaptive Program is stopped B1 **Bit 1** 1 Adaptive Program can be edited 0 Adaptive Program cannot be edited B2 **Bit 2** 1 Adaptive Program is being checked 0 no action B3 **Bit 3** 1 Adaptive Program is faulty 0 Adaptive Program is OK B4 **Bit 4** 1 Adaptive Program is protected 0 Adaptive Program is unprotected Faults in the Adaptive Program can be: – used function block with not at least input 1 connection – used pointer is not valid – invalid bit number for function block **Bset** – location of function block **PI-Bal** after **PI** function block **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E
84.02	**FaultedPar (faulted parameters)** The Adaptive Program will be checked before running. If there is a fault, *AdapPrgStat (84.01)* is set to “faulty” and *FaultedPar (84.02)* shows the faulty input. **Note:** In case of a problem check the value and the attribute of the faulty input. **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E
84.03	**LocationCounter (location counter)** Location counter for *AdapProgCmd (83.01)* = **SingleStep** shows the function block number, which will be executed next. **Int. Scaling: 1 == 1 Type: I Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
84.04	**Block1Type (function block 1 type)** Selects the type for function block 1 [Block Parameter Set 1 (BPS1)]. Detailed description of the type can be found in chapter '*Function blocks*': 0 = **NotUsed** function block is not used 1 = **ABS** absolute value 2 = **ADD** sum 3 = **AND** AND 4 = **Bitwise** bit compare 5 = **Bset** bit set 6 = **Compare** compare 7 = **Count** counter 8 = **D-Pot** ramp 9 = **Event** event 10 = **Filter** filter 11 = **Limit** limit 12 = **MaskSet** mask set 13 = **Max** maximum 14 = **Min** minimum 15 = **MulDiv** multiplication and division 16 = **OR** OR 17 = **ParRead** parameter read 18 = **ParWrite** parameter write 19 = **PI** PI-controller 20 = **PI-Bal** initialization for PI-controller 21 = **Ramp** ramp 22 = **SqWav** square wave 23 = **SR** SR flip-flop 24 = **Switch-B** switch Boolean 25 = **Switch-I** switch integer 26 = **TOFF** timer off 27 = **TON** timer on 28 = **Trigg** trigger 29 = **XOR** exclusive OR 30 = **Sqrt** square root 31 = **Jump** jump 32 = **TachoAdjust** adjust analog tacho 33 = **Position** position **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	Position	NotUsed	-	E
84.05	**Block1In1 (function block 1 input 1)** Selects the source for input 1 of function block 1 (BPS1). There are 2 types of inputs, signals/parameters and constants: – Signals/parameters are all signals and parameters available in the drive. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. Example: To connect negated *SpeedRef (23.01)* set *Block1In1 (84.05)* = -2301 and *Block1Attrib (84.08)* = 0h. To get only a certain bit e.g. **RdyRef** bit 3 of *MainStatWord (8.01)* set *Block1In1 (84.05)* = 801 and *Block1Attrib (84.08)* = 3h. – Constants are feed directly into the function block input and have to be declared by means of *Block1Attrib (84.08).* Example: To connect the constant value of 12345 set *Block1In1 (84.05)* = 12345 and *Block1Attrib (84.08)* = 1000h. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
84.06	**Block1In2 (function block 1 input 2)** Selects the source for input 2 of function block 1 (BPS1). Description see *Block1In1 (84.05)*, except: To get only a certain bit e.g. **RdyRef** bit 3 of *MainStatWord (8.01)* set *Block1In2 (84.06)* = 801 and *Block1Attrib (84.08)* = 30h. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
84.07	**Block1In3 (function block 1 input 3)** Selects the source for input 3 of function block 1 (BPS1). Description see *Block1In1 (84.05)*, except: To get only a certain bit e.g. **RdyRef** bit 3 of *MainStatWord (8.01)* set *Block1In3 (84.07)* = 801 and *Block1Attrib (84.08)* = 300h. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
84.08	**Block1Attrib (function block 1 attribute)** Defines the attributes of function block 1 for all three inputs [*Block1In1 (84.05)*, *Block1In2 (84.06)* and *Block1In3 (84.07)*] (BPS1). *Block1Attrib (84.08)* is divided into 4 parts: – Bit number 0 - 3 for input 1 to get a certain bit out of a packed Boolean word. – Bit number 4 - 7 for input 2 to get a certain bit out of a packed Boolean word. – Bit number 8 - 11 for input 3 to get a certain bit out of a packed Boolean word. – Bit number 12 - 14 for input 1 - 3 to feed a constant directly into the input  **Int. Scaling: 1 == 1 Type: h Volatile: N**	0h	FFFFh	0h	-	E
84.09	**Block1Output (function block 1 output)** Function block 1 output, can be used as an input for further function blocks. **Int. Scaling: 1 == 1 Type: SI Volatile: Y**	-	-	-	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
84.10 to 84.99	The description of the parameters for function blocks 2 to 16 is basically the same as for function block 1. For Your convenience the following table shows the parameter numbers of all function blocks1:					E

Function block	BlockxType	BlockxIn1 input 1	BlockxIn2 input 2	BlockxIn3 input 1	BlockxAttrib	BlockxOutput signal	BlockxOut pointer
1	84.04	84.05	84.06	84.07	84.08	84.09	86.01
2	84.10	84.11	84.12	84.13	84.14	84.15	86.02
3	84.16	84.17	84.18	84.19	84.20	84.21	86.03
4	84.22	84.23	84.24	84.25	84.26	84.27	86.04
5	84.28	84.29	84.30	84.31	84.32	84.33	86.05
6	84.34	84.35	84.36	84.37	84.38	84.39	86.06
7	84.40	84.41	84.42	84.43	84.44	84.45	86.07
8	84.46	84.47	84.48	84.49	84.50	84.51	86.08
9	84.52	84.53	84.54	84.55	84.56	84.57	86.09
10	84.58	84.59	84.60	84.61	84.62	84.63	86.10
11	84.64	84.65	84.66	84.67	84.68	84.69	86.11
12	84.70	84.71	84.72	84.73	84.74	84.75	86.12
13	84.76	84.77	84.78	84.79	84.80	84.81	86.13
14	84.82	84.83	84.84	84.85	84.86	84.87	86.14
15	84.88	84.89	84.90	84.91	84.92	84.93	86.15
16	84.94	84.95	84.96	84.97	84.98	84.99	86.16

Group 85

User constants

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
85.01	**Constant1 (constant 1)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
85.02	**Constant2 (constant 2)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
85.03	**Constant3 (constant 3)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
85.04	**Constant4 (constant 4)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
85.05	**Constant5 (constant 5)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
85.06	**Constant6 (constant 6)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
85.07	**Constant7 (constant 7)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
85.08	**Constant8 (constant 8)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
85.09	**Constant9 (constant 9)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
85.10	**Constant10 (constant 10)** Sets an integer constant for the Adaptive Program. **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
85.11	**String1 (string 1)** Sets a string for the Adaptive Program. With DriveWindow it is possible to fill in a string (e.g. name of an event) with a maximum of 12 characters. This string is shown in the DCS800 Control Panel and in DriveWindow. **Int. Scaling: 1 == 1 Type: SI/C Volatile: N**	'string'	'string'	' '	-	E
85.12	**String2 (string 2)** Sets a string for the Adaptive Program. With DriveWindow it is possible to fill in a string (e.g. name of an event) with a maximum of 12 characters. This string is shown in the DCS800 Control Panel and in DriveWindow. **Int. Scaling: 1 == 1 Type: SI/C Volatile: N**	'string'	'string'	' '	-	E
85.13	**String3 (string 3)** Sets a string for the Adaptive Program. With DriveWindow it is possible to fill in a string (e.g. name of an event) with a maximum of 12 characters. This string is shown in the DCS800 Control Panel and in DriveWindow. **Int. Scaling: 1 == 1 Type: SI/C Volatile: N**	'string'	'string'	' '	-	E
85.14	**String4 (string 4)** Sets a string for the Adaptive Program. With DriveWindow it is possible to fill in a string (e.g. name of an event) with a maximum of 12 characters. This string is shown in the DCS800 Control Panel and in DriveWindow. **Int. Scaling: 1 == 1 Type: SI/C Volatile: N**	'string'	'string'	' '	-	E
85.15	**String5 (string 5)** Sets a string for the Adaptive Program. With DriveWindow it is possible to fill in a string (e.g. name of an event) with a maximum of 12 characters. This string is shown in the DCS800 Control Panel and in DriveWindow. **Int. Scaling: 1 == 1 Type: SI/C Volatile: N**	'string'	'string'	' '	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 86	**Adaptive Program outputs**					
86.01	**Block1Out (block 1 output)** The value of function block 1 output [*Block1Output (84.09)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.02	**Block2Out (block 2 output)** The value of function block 2 output [*Block2Output (84.15)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.03	**Block3Out (block 3 output)** The value of function block 3 output [*Block3Output (84.21)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.04	**Block4Out (block 4 output)** The value of function block 4 output [*Block1Output (84.27)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.05	**Block5Out (block 5 output)** The value of function block 5 output [*Block1Output (84.33)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.06	**Block6Out (block 6 output)** The value of function block 6 output [*Block1Output (84.39)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.07	**Block7Out (block 7 output)** The value of function block 7 output [*Block1Output (84.45)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.08	**Block8Out (block 8 output)** The value of function block 8 output [*Block1Output (84.51)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.09	**Block9Out (block 9 output)** The value of function block 9 output [*Block1Output (84.57)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E

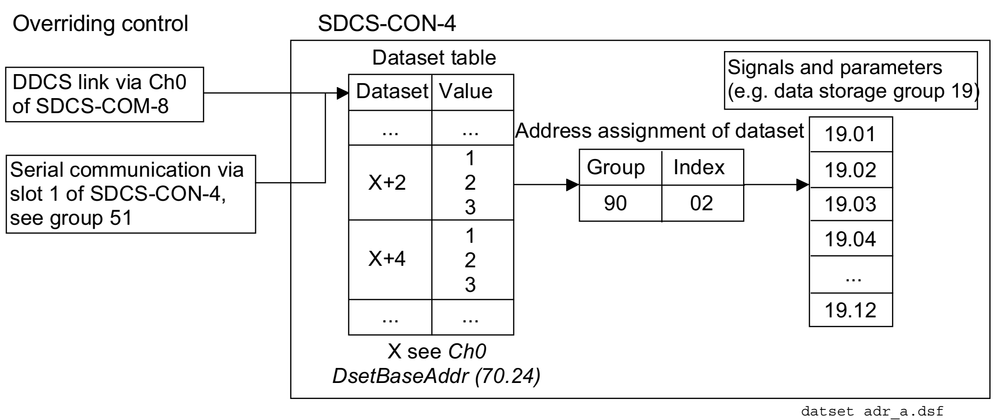
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
86.10	**Block10Out (block 10 output)** The value of function block 10 output [*Block1Output (84.63)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.11	**Block11Out (block 11 output)** The value of function block 11 output [*Block1Output (84.69)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.12	**Block12Out (block 12 output)** The value of function block 12 output [*Block1Output (84.75)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.13	**Block13Out (block 13 output)** The value of function block 13 output [*Block1Output (84.81)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.14	**Block14Out (block 14 output)** The value of function block 14 output [*Block1Output (84.87)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.15	**Block15Out (block 15 output)** The value of function block 15 output [*Block1Output (84.93)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
86.16	**Block16Out (block 16 output)** The value of function block 16 output [*Block16Output (84.99)*] is written to a sink (signal/parameter) by means of this index pointer [e.g. 2301 equals *SpeedRef (23.01)*]. The format is **-xxyy**, with: **-** = negate signal/parameter, **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-9999	9999	0	-	E
Group 88	**Internal**					
	This parameter group contains internal variables and should not be changed by the user					
88.01	**Reserved**					
…	…					
88.24	**Reserved**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
88.25	**M1TachMaxSpeed (motor 1 tacho maximum speed)** Internally used tacho maximum speed for motor 1. This value is depending on the analog tacho output voltage - e.g. 60 V at 1000 rpm - and the maximum speed of the drive system - which is the maximum of *SpeedScaleAct (2.29)*, *M1OvrSpeed (30.16)* and *M1BaseSpeed (99.04)*. This value should only be written to by: – tacho fine tuning via *ServiceMode (99.06)* = **TachFineTune**, – via *M1TachVolt1000 (50.13)*, – TachoAdjust block in Adaptive Program, – TachoAdjust block in application program and – parameter download Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Int. Scaling: (2.29) Type: SI Volatile: N**	0	10000	0	rpm	E
88.26	**M2TachMaxSpeed (motor 2 tacho maximum speed)** Internally used tacho maximum speed for motor 2. This value is depending on the analog tacho output voltage - e.g. 60 V at 1000 rpm - and the maximum speed of the drive system - which is the maximum of *SpeedScaleAct (2.29)*, *M2OvrSpeed (49.21)* and *M2BaseSpeed (49.03)*. This value should only be written to by: – tacho fine tuning via *ServiceMode (99.06)* = **TachFineTune**, – via *M2TachVolt1000 (49.27)*, – TachoAdjust block in Adaptive Program, – TachoAdjust block in application program and – parameter download Internally limited from: $-(2.29)*\frac{32767}{20000}rpm\ \ to\ \ (2.29)*\frac{32767}{20000}rpm$ **Int. Scaling: (2.29) Type: SI Volatile: N**	0	10000	0	rpm	E
88.27	**M1TachoTune (motor 1 tacho tuning factor)** Internally used tacho fine tuning factor for motor 1. This value should only be written to by: – tacho fine tuning via *ServiceMode (99.06)* = **TachFineTune**, – TachoAdjust block in Adaptive Program, – TachoAdjust block in application program and – parameter download **Int. Scaling: 1000 == 1 Type: I Volatile: N**	0.3	3	1	-	E
88.28	**M2TachoTune (motor 2 tacho tuning factor)** Internally used tacho fine tuning factor for motor 2. This value should only be written to by: – tacho fine tuning via *ServiceMode (99.06)* = **TachFineTune**, – TachoAdjust block in Adaptive Program, – TachoAdjust block in application program and – parameter download **Int. Scaling: 1000 == 1 Type: I Volatile: N**	0.3	3	1	-	E
88.29	**M1TachoGain (motor 1 tacho tuning gain)** Internally used tacho gain tuning for motor 1. This value should only be written to by: – tacho gain tuning via *ServiceMode (99.06)* = **SpdFbAssist**, – *M1TachoVolt1000 (50.13)* and – parameter download **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	15	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
88.30	**M2TachoGain (motor 2 tacho tuning gain)** Internally used tacho gain tuning for motor 2. This value should only be written to by: – tacho gain tuning via *ServiceMode (99.06)* = **SpdFbAssist**, – *M2TachoVolt1000 (49.27)* and – parameter download **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	15	15	-	E
88.31	**AnybusModType (last connected serial communication module)** Internally used memory for the last attached serial communication module. This value should only be written to by: – the DCS800 firmware and – parameter download **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	65535	0	-	E

Group 90 Receiving data sets addresses 1

Addresses for the received data transmitted from the overriding control to the drive.
The format is **xxyy**, with: **xx** = group and **yy** = index.
The data set base address is set in *Ch0 DsetBaseAddr (70.24)*.

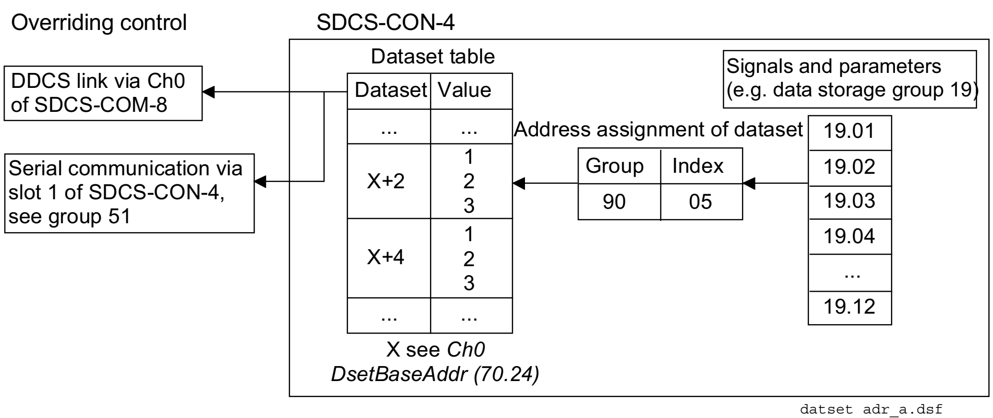


Index	Signal / Parameter name	min.	max.	def.	unit	E/C
90.01	**DsetXVal1 (data set X value 1)** Data set X value 1 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)*. Default setting of 701 equals *MainCtrlWord (7.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	701	-	E
90.02	**DsetXVal2 (data set X value 2)** Data set X value 2 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)*. Default setting of 2301 equals *SpeedRef (23.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	2301	-	E
90.03	**DsetXVal3 (data set X value 3)** Data set X value 3 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)*. Default setting of 2501 equals *TorqRefA (25.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	2501	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
90.04	**DsetXplus2Val1 (data set X+2 value 1)** Data set X+2 value 1 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 2. Default setting of 702 equals *AuxCtrlWord (7.02)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	702	-	E
90.05	**DsetXplus2Val2 (data set X+2 value 2)** Data set X+2 value 2 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 2. Default setting of 703 equals *AuxCtrlWord2 (7.03)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	703	-	E
90.06	**DsetXplus2Val3 (data set X+2 value 3)** Data set X+2 value 3 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 2. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.07	**DsetXplus4Val1 (data set X+4 value 1)** Data set X+4 value 1 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 4. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.08	**DsetXplus4Val2 (data set X+4 value 2)** Data set X+4 value 2 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 4. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.09	**DsetXplus4Val3 (data set X+4 value 3)** Data set X+4 value 3 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr(70.24)* + 4. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.10	**DsetXplus6Val1 (data set X+6 value 1)** Data set X+6 value 1 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 6. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.11	**DsetXplus6Val2 (data set X+6 value 2)** Data set X+6 value 2 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 6. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.12	**DsetXplus6Val3 (data set X+6 value 3)** Data set X+6 value 3 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 6. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.13	**DsetXplus8Val1 (data set X+8 value 1)** Data set X+8 value 1 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 8. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.14	**DsetXplus8Val2 (data set X+8 value 2)** Data set x+8 value 2 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 8. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.15	**DsetXplus8Val3 (data set X+8 value 3)** Data set X+8 value 3 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 8. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
90.16	**DsetXplus10Val1 (data set X+10 value 1)** Data set X+10 value 1 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 10. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.17	**DsetXplus10Val2 (data set X+10 value 2)** Data set X+10 value 2 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 10. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
90.18	**DsetXplus10Val3 (data set X+10 value 3)** Data set X+10 value 3 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 10. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
Group 91	**Receiving data sets addresses 2**					
91.01	**DsetXplus12Val1 (data set X+12 value 1)** Data set X+12 value 1 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 12. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
91.02	**DsetXplus12Val2 (data set X+12 value 2)** Data set X+12 value 2 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 12. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
91.03	**DsetXplus12Val3 (data set X+12 value 3)** Data set X+12 value 2 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 12. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
91.04	**DsetXplus14Val1 (data set X+14 value 1)** Data set X+14 value 1 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 14. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
91.05	**DsetXplus14Val2 (data set X+14 value 2)** Data set X+14 value 2 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 14. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
91.06	**DsetXplus14Val3 (data set X+14 value 3)** Data set X+14 value 3 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 14. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 92	**Transmit data sets addresses 1**					
	Addresses for the transmit data send from the drive to the overriding control. The format is **xxyy**, with: **xx** = group and **yy** = index. The data set base address is set in *Ch0 DsetBaseAddr (70.24)*.					
92.01	**DsetXplus1Val1 (data set X+1 value 1)** Data set X+1 value 1 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 1. Default setting of 801 equals *MainStatWord (8.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	801	-	E
92.02	**DsetXplus1Val2 (data set X+1 value 2)** Data set X+1 value 2 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 1. Default setting of 104 equals *MotSpeed (1.04)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	104	-	E
92.03	**DsetXplus1Val3 (data set X+1 value 3)** Data set X+1 value 3 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 1. Default setting of 209 equals *TorqRef2 (2.09)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	209	-	E
92.04	**DsetXplus3Val1 (data set X+3 value 1)** Data set X+3 value 1 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 3. Default setting of 802 equals *AuxStatWord (8.02)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	802	-	E
92.05	**DsetXplus3Val2 (data set X+3 value 2)** Data set X+3 value 2 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 3. Default setting of 101 equals *MotSpeedFilt (1.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	101	-	E
92.06	**DsetXplus3Val3 (data set X+3 value 3)** Data set X+3 value 3 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 3. Default setting of 108 equals *MotTorq (1.08)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	108	-	E



Index	Signal / Parameter name	min.	max.	def.	unit	E/C
92.07	**DsetXplus5Val1 (data set X+5 value 1)** Data set X+5 value 1 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 5. Default setting of 901 equals *FaultWord1 (9.01)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	901	-	E
92.08	**DsetXplus5Val2 (data set X+5 value 2)** Data set X+5 value 2 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 5. Default setting of 902 equals *FaultWord2 (9.02)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	902	-	E
92.09	**DsetXplus5Val3 (data set X+5 value 3)** Data set X+5 value 3 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 5. Default setting of 903 equals *FaultWord3 (9.03)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	903	-	E
92.10	**DsetXplus7Val1 (data set X+7 value 1)** Data set X+7 value 1 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 7. Default setting of 904 equals *FaultWord4 (9.04)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	904	-	E
92.11	**DsetXplus7Val2 (data set X+7 value 2)** Data set X+7 value 2 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 7. Default setting of 906 equals *AlarmWord1 (9.06)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	906	-	E
92.12	**DsetXplus7Val3 (data set X+7 value 3)** Data set X+7 value 3 (interval: 3 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 7. Default setting of 907 equals *AlarmWord2 (9.07)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	907	-	E
92.13	**DsetXplus9Val1 (data set X+9 value 1)** Data set X+9 value 1 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 9. Default setting of 908 equals *AlarmWord3 (9.08)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	908	-	E
92.14	**DsetXplus9Val2 (data set X+9 value 2)** Data set X+9 value 2 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 9. Default setting of 803 equals *LimWord (8.03)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	803	-	E
92.15	**DsetXplus9Val3 (data set X+9 value 3)** Data set X+9 value 3 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 9. Default setting of 805 equals *DI StatWord (8.05)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	805	-	E
92.16	**DsetXplus11Val1 (data set X+11 value 1)** Data set X+11 value 1 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 11. Default setting of 806 equals *DO StatWord (8.06)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	806	-	E
92.17	**DsetXplus11Val2 (data set x+11 value 2)** Data set X+11 value 2 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 11. Default setting of 124 equals *BridgeTemp (1.24)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	124	-	E
92.18	**DsetXplus11Val3 (data set X+11 value 3)** Data set X+11 value 3 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 11. Default setting of 112 equals *Mot1TempMeas (1.22)*. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	122	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 93	**Transmit data sets addresses 2**					
93.01	**DsetXplus13Val1 (data set X+13 value 1)** Data set X+13 value 1 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 13. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
93.02	**DsetXplus13Val2 (data set X+13 value 2)** Data set X+13 value 2 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 13. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
93.03	**DsetXplus13Val3 (data set X+13 value 3)** Data set X+13 value 3 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 13. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
93.04	**DsetXplus15Val1 (data set X+15 value 1)** Data set X+15 value 1 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 15. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
93.05	**DsetXplus15Val2 (data set X+15 value 2)** Data set X+15 value 2 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 15. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
93.06	**DsetXplus15Val3 (data set X+15 value 3)** Data set X+15 value 3 (interval: 30 ms). Data set address = *Ch0 DsetBaseAddr (70.24)* + 15. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C

Group 94

DCSLink control

This parameter group defines the communication parameters for the DCSLink board SDCS-DSL-4. For communication between the armature converter and the field exciters respectively 12-pulse communication only the basic communication parameters [(94.01) to (94.09)] have to be set.

For master-follower and drive-to-drive communication the basic communication parameters have to be set. The data transfer is done by means of the 4 available mailboxes [(94.12) to (94.35)].

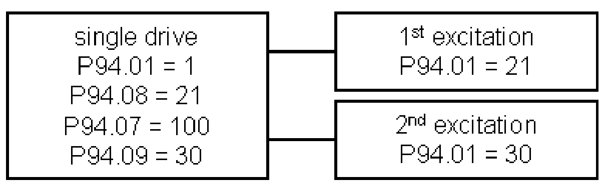
Parameter settings, default values:

single drive with excitation	*DCSLinkNodeID (94.01)* = 1 *M1FexNode (94.08)* = 21 *M2FexNode (94.09)* = 30	see example 1
12-pulse drive	*DCSLinkNodeID (94.01)* = 1 *12P SlaNode (94.04)* = 31 *M1FexNode (94.08)* = 21	see example 2

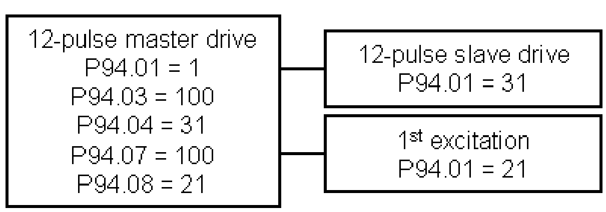
Example parameter settings for:

	Node number					
master-follower (94.01)	1	2	3	…	11	see example 3
field exciter (94.08)	21	22	23	…	31	see example 3
12-pulse slave (94.04) and (94.01)	31	32	-	-	-	see example 4
drive-to-drive (94.01)	1	2	3	-	-	see example 5

Example 1:
Single drive with one or two field exciters and communication supervision

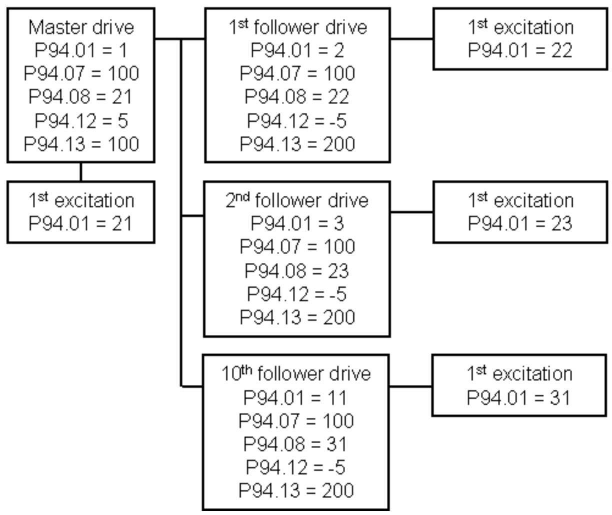


Example 2:
12-pulse configuration and communication supervision

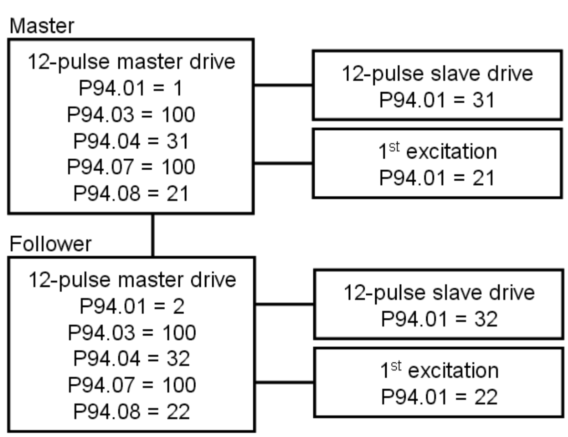


Index	Signal / Parameter name	min.	max.	def.	unit	E/C

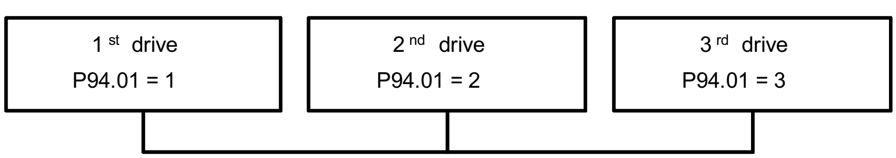
Example 3:
Master-follower configuration (broadcast) with one mailbox activated and communication supervision



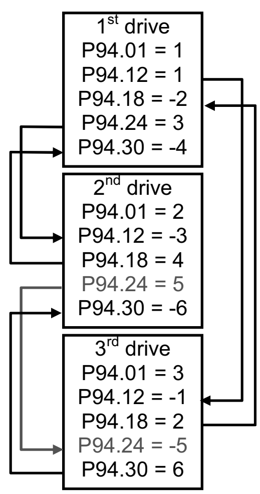
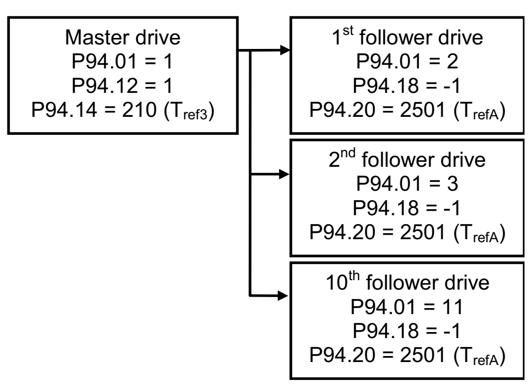
Example 4:
Two 12-pulse drives in master-follower configuration and communication supervision



Example 5:
Drive-to-drive configuration



Index	Signal / Parameter name	min.	max.	def.	unit	E/C
94.01	**DCSLinkNodeID (DCSLink node ID)** Defines the DCSLink node ID of the station. Two stations with the same node ID are not allowed. Maximum allowed station count is 50. See also examples 1 to 5 above. The DCSLink node ID is inactive, if *DCSLinkNodeID (94.01)* is set to 0. The drive trips with **F508 I/OBoardLoss** [*FaultWord1 (9.01)* bit 7], if the SDCS-DSL-4 board is chosen, but not connected or faulty. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	63	0	-	E
94.02	**BaudRate (baud rate)** Defines the transfer rate of the DCSLink. The transfer rate decreases with the total length of the DCSLink cable: 0 = **20 kBit/s** 20 kBit/s, total cable length max. 500 m 1 = **50 kBit/s** 50 kBit/s, total cable length max. 500 m 2 = **125 kBit/s** 125 kBit/s, total cable length max. 500 m 3 = **250 kBit/s** 250 kBit/s, total cable length max. 250 m 4 = **500 kBit/s** 500 kBit/s, total cable length max. 100 m, default 5 = **800 kBit/s** 800 kBit/s, total cable length max. 50 m 6 = **888 kBit/s** 888 kBit/s, total cable length max. 35 m 7 = **1 MBit/s** 1 MBit/s, total cable length approximately 25 m **Note:** Maximum total cable length should not exceed 100 m. Maximum amount of connected stations is 50 (e.g. 25 drives including one external field exciter each). **Int. Scaling: 1 == 1 Type: C Volatile: N**	20 kBit/s	1 MBit/s	500 kBit/s	-	E
94.03	**12P TimeOut (12-pulse timeout)** Time delay before a 12-pulse communication break is declared and **F535 12PulseCom** [*FaultWord3 (9.03)* bit 2] is set. *12P TimeOut (94.03)* is only active in the 12-pulse master. The communication fault is inactive, if *12P TimeOut (94.03)* is set to 0 ms. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	100	ms	E
94.04	**12P SlaNode (12-pulse slave node ID)** Defines the DCSLink node ID of the 12-pulse slave drive in the 12-pulse master drive. See also examples 2 and 4 above. The 12-pulse node ID is inactive, if *12P SlaNode (94.04)* is set to 0. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	63	31	-	E
94.05	**Unused**					
94.06	**Unused**					
94.07	**FexTimeOut (field exciter timeout)** Time delay before a field exciter communication break is declared. Depending on the fex with the communication break either **F516 M1FexCom** [*FaultWord1 (9.01)* bit 15] or **F519 M2FexCom** [*FaultWord2 (9.02)* bit 2] is set. *FexTimeOut (94.07)* is only active in the armature converter. The communication fault is inactive, if *FexTimeOut (94.07)* is set to 0 ms. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	100	ms	E
94.08	**M1FexNode (motor 1 field exciter node ID)** Defines the DCSLink node ID of motor 1 field exciter in the drive. See also examples 1 to 4 above. The field exciter node ID is inactive, if *M1FexNode (94.08)* is set to 0. **Note:** *M1FexNode (94.08)* is void, when *M1UsedFexType (99.12)* = **NotUsed** or **OnBoard**. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	32	21	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
94.09	**M2FexNode (motor 2 field exciter node ID)** Defines the DCSLink node ID of motor 2 field exciter in the drive. See also example 1 above. The field exciter node ID is inactive, if *M2FexNode (94.09)* is set to 0. **Note:** *M2FexNode (94.09)* is void, when *M2UsedFexType (49.07)* = **NotUsed** or **OnBoard**. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	32	30	-	E
94.10	**Unused**					
94.11	**Unused**					
	The drive-to-drive and master-follower communication utilizes 4 mailboxes to transfer data. Thus data transfer to any station in the system is possible. Each mailbox can transmit / receive up to 4 values. Positive mailbox node ID numbers only transmit data, negative only receive data. To get communication mailbox node ID pairs are needed.					
	Example 6: Drive-to-drive configuration, sending signals from drive 2 using *MailBox3 (94.24)* to drive 3 using *MailBox3 (94.24)* by means of 5 to transmit data and -5 to receive data. 					
	Example 7: Master-follower configuration; send *TorqRef3 (2.10)* from the master drive via *MailBox1 (94.12)* to *TorqRefA (25.01)* of the followers via *MailBox2 (94.18)*. 					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
94.12	**MailBox1 (mailbox 1 node ID)** Mailbox 1 can transmit / receive up to 4 values [*TrmtRecVal1.1 (94.13), TrmtRecVal1.2 (94.14), TrmtRecVal1.3 (94.15)* and *TrmtRecVal1.4 (94.16)*]. Positive mailbox node ID numbers transmit data, negative receive data. To get communication, mailbox node ID pairs are needed. See also examples 6 and 7 above. The mailbox is inactive, if *MailBox1 (94.12)* is set to 0. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-64	64	0	-	E
94.13	**MailBoxCycle1 (cycle time mailbox 1)** The function of *MailBoxCycle1 (94.13)* is depending on the setting of *MailBox1 (94.12)*. If *MailBox1 (94.12)* is positive: – data will be transmitted – *MailBoxCycle1 (94.13)* sets the transmitting and receiving intervals – if *MailBoxCycle1 (94.13)* is set to 3 ms the transmit and receiving intervals are synchronized with mains frequency, either 3.3 ms or 2.77 ms – values from 1 - 2 ms are too fast and will generate a fault – the communication is inactive, if *MailBoxCycle1 (94.13)* is set to 0 ms If *MailBox1 (94.12)* is negative: – data will be received – *MailBoxCycle1 (94.13)* sets the communication timeout. This is the time delay before a drive-to-drive or master-follower communication break is declared. Depending on the setting of *ComLossCtrl (30.28)* either **F544 P2PandMFCom** [*FaultWord3 (9.03)* bit 11] or **A112 P2PandMFCom** [*AlarmWord1 (9.06)* bit 11] is set. – the communication fault and alarm are inactive, if *MailBoxCycle1 (94.13)* is set to 0 ms **Attention:** The communication timeout has to be set at least twice as long as the corresponding mail box cycle time parameter. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	100	ms	E
94.14	**TrmtRecVal1.1 (mailbox 1 transmit / receive value 1)** Mailbox 1 transmit / receive value 1. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.15	**TrmtRecVal1.2 (mailbox 1 transmit / receive value 2)** Mailbox 1 transmit / receive value 2. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.16	**TrmtRecVal1.3 (mailbox 1 transmit / receive value 3)** Mailbox 1 transmit / receive value 3. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.17	**TrmtRecVal1.4 (mailbox 1 transmit / receive value 4)** Mailbox 1 transmit / receive value 4. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.18	**MailBox2 (mailbox 2 node ID)** Mailbox 2 can transmit / receive up to 4 values [*TrmtRecVal2.1 (94.20), TrmtRecVal2.2 (94.21), TrmtRecVal2.3 (94.22)* and *TrmtRecVal2.4 (94.23)*]. Positive mailbox node ID numbers transmit data, negative receive data. To get communication, mailbox node ID pairs are needed. See also examples 6 and 7 above. The mailbox is inactive, if *MailBox2 (94.18)* is set to 0. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-64	64	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
94.19	**MailBoxCycle2 (cycle time mailbox 2)** The function of *MailBoxCycle2 (94.19)* is depending on the setting of *MailBox2 (94.18)*. If *MailBox2 (94.18)* is positive: – data will be transmitted – *MailBoxCycle2 (94.19)* sets the transmitting and receiving intervals – if *MailBoxCycle2 (94.19)* is set to 3 ms the transmit and receiving intervals are synchronized with mains frequency, either 3.3 ms or 2.77 ms – values from 1 - 2 ms are too fast and will generate a fault – the communication is inactive, if *MailBoxCycle2 (94.19)* is set to 0 ms If *MailBox2 (94.18)* is negative: – data will be received – *MailBoxCycle2 (94.19)* sets the communication timeout. This is the time delay before a drive-to-drive or master-follower communication break is declared. Depending on the setting of *ComLossCtrl (30.28)* either **F544 P2PandMFCom** [*FaultWord3 (9.03)* bit 11] or **A112 P2PandMFCom** [*AlarmWord1 (9.06)* bit 11] is set. – the communication fault and alarm are inactive, if *MailBoxCycle2 (94.19)* is set to 0 ms **Attention:** The communication timeout has to be set at least twice as long as the corresponding mail box cycle time parameter. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	100	ms	E
94.20	**TrmtRecVal2.1 (mailbox 2 transmit / receive value 1)** Mailbox 2 transmit / receive value 1. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.21	**TrmtRecVal2.2 (mailbox 2 transmit / receive value 2)** Mailbox 2 transmit / receive value 2. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.22	**TrmtRecVal2.3 (mailbox 2 transmit / receive value 3)** Mailbox 2 transmit / receive value 3. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.23	**TrmtRecVal2.4 (mailbox 2 transmit / receive value 4)** Mailbox 2 transmit / receive value 4. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.24	**MailBox3 (mailbox 3 node ID)** Mailbox 3 can transmit / receive up to 4 values [*TrmtRecVal3.1 (94.26), TrmtRecVal3.2 (94.27), TrmtRecVal3.3 (94.28)* and *TrmtRecVal3.4 (94.29)*]. Positive mailbox node ID numbers transmit data, negative receive data. To get communication, mailbox node ID pairs are needed. See also examples 6 and 7 above. The mailbox is inactive, if *MailBox3 (94.24)* is set to 0. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-64	64	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
94.25	**MailBoxCycle3 (cycle time mailbox 3)** The function of *MailBoxCycle3 (94.25)* is depending on the setting of *MailBox3 (94.24)*. If *MailBox3 (94.24)* is positive: – data will be transmitted – *MailBoxCycle3 (94.25)* sets the transmitting and receiving intervals – values from 1 - 4 ms are too fast and will generate a fault – the communication is inactive, if *MailBoxCycle3 (94.25)* is set to 0 ms If *MailBox3 (94.24)* is negative: – data will be received – *MailBoxCycle3 (94.25)* sets the communication timeout. This is the time delay before a drive-to-drive or master-follower communication break is declared. Depending on the setting of *ComLossCtrl (30.28)* either **F544 P2PandMFCom** [*FaultWord3 (9.03)* bit 11] or **A112 P2PandMFCom** [*AlarmWord1 (9.06)* bit 11] is set. – the communication fault and alarm are inactive, if *MailBoxCycle3 (94.25)* is set to 0 ms **Attention:** The communication timeout has to be set at least twice as long as the corresponding mail box cycle time parameter. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	100	ms	E
94.26	**TrmtRecVal3.1 (mailbox 3 transmit / receive value 1)** Mailbox 3 transmit / receive value 1. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.27	**TrmtRecVal3.2 (mailbox 3 transmit / receive value 2)** Mailbox 3 transmit / receive value 2. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.28	**TrmtRecVal3.3 (mailbox 3 transmit / receive value 3)** Mailbox 3 transmit / receive value 3. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.29	**TrmtRecVal3.4 (mailbox 3 transmit / receive value 4)** Mailbox 3 transmit / receive value 4. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.30	**MailBox4 (mailbox 4 node ID)** Mailbox 4 can transmit / receive up to 4 values [*TrmtRecVal4.1 (94.32), TrmtRecVal4.2 (94.33), TrmtRecVal4.3 (94.34)* and *TrmtRecVal4.4 (94.35)*]. Positive mailbox node ID numbers transmit data, negative receive data. To get communication, mailbox node ID pairs are needed. See also examples 6 and 7 above. The mailbox is inactive, if *MailBox4 (94.30)* is set to 0. **Int. Scaling: 1 == 1 Type: I Volatile: N**	-64	64	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
94.31	**MailBoxCycle4 (cycle time mailbox 4)** The function of *MailBoxCycle4 (94.31)* is depending on the setting of *MailBox4 (94.30)*. If *MailBox4 (94.30)* is positive: – data will be transmitted – *MailBoxCycle4 (94.31)* sets the transmitting and receiving intervals – values from 1 - 4 ms are too fast and will generate a fault – the communication is inactive, if *MailBoxCycle4 (94.31)* is set to 0 ms If *MailBox4 (94.30)* is negative: – data will be receive – *MailBoxCycle4 (94.31)* sets the communication timeout. This is the time delay before a drive-to-drive or master-follower communication break is declared. Depending on the setting of *ComLossCtrl (30.28)* either **F544 P2PandMFCom** [*FaultWord3 (9.03)* bit 11] or **A112 P2PandMFCom** [*AlarmWord1 (9.06)* bit 11] is set. – the communication fault and alarm are inactive, if *MailBoxCycle4 (94.31)* is set to 0 ms **Attention:** The communication timeout has to be set at least twice as long as the corresponding mail box cycle time parameter. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	64000	100	ms	E
94.32	**TrmtRecVal4.1 (mailbox 4 transmit / receive value 1)** Mailbox 4 transmit / receive value 1. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.33	**TrmtRecVal4.2 (mailbox 4 transmit / receive value 2)** Mailbox 4 transmit / receive value 2. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.34	**TrmtRecVal4.3 (mailbox 4 transmit / receive value 3)** Mailbox 4 transmit / receive value 3. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E
94.35	**TrmtRecVal4.4 (mailbox 4 transmit / receive value 4)** Mailbox 4 transmit / receive value 4. The format is **xxyy**, with: **xx** = group and **yy** = index. **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	9999	0	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 97	**Measurement**					
97.01	**TypeCode (type code)** *TypeCode (97.01)* is preset in the factory and is write protected. It identifies the drives current-, voltage-, temperature measurement and its quadrant type. To un-protect the type code set *ServiceMode (99.06)* = **SetTypeCode**. The change of the type code is immediately taken over and *ServiceMode (99.06)* is automatically set back to **NormalMode**: 0 = **None** the type code is set by user, see *S ConvScaleCur (97.02), S ConvScaleVolt (97.03), S MaxBrdgTemp (97.04)* and *S BlockBridge2 (97.07)* for e.g. rebuild kits 1 = **S01-0020-04** type code, see table to 148 = **S02-5200-05** type code, see table	None	S01-5203-05	factory preset value	-	E

The drive's basic type code: **DCS800-AAX-YYYY-ZZB**			
Product family:	DCS800		
Type:	AA	= S0	Standard converter modules
		= R0	Rebuild system
		= E0	Panel solution
		= A0	Enclosed converter
Bridge type:	X	= 1	Single bridge (2-Q)
		= 2	2 anti parallel bridges (4-Q)
Module type:	YYYY	=	Rated DC current
Rated AC voltage:	ZZ	= 04	230 VAC - 400 VAC
		= 05	230 VAC - 525 VAC
		= 06	270 VAC - 600 VAC
		= 07	315 VAC - 690 VAC
		= 08	360 VAC - 800 VAC
		= 10	450 VAC - 990 VAC
		= 12	540 VAC - 1200 VAC
Power connection:	B	= -	Standard D1 - D6
		= L	Left side D7
		= R	Right side D7
		= a	Second thyristor type D5, D6

Attention:
When using D1, D2, D3 or D4 modules the current and voltage range of the type code setting is limited to max 1000 ADC and max 600 VAC.

Int. Scaling: 1 == 1 Type: C Volatile: Y

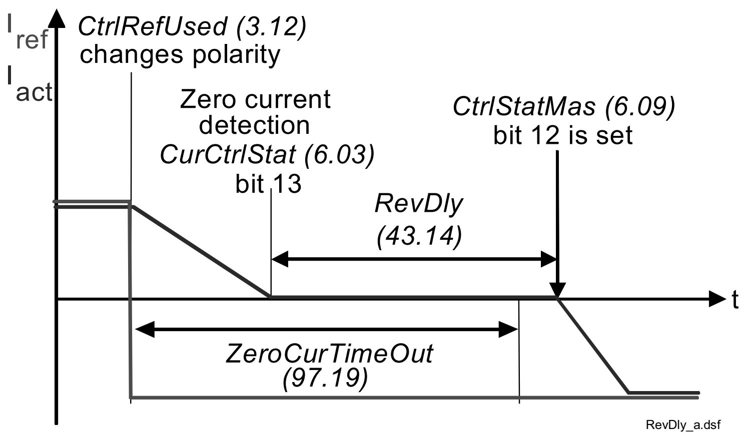
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	Type code table					

0	**None**	51	**S01-2600-10**	102	**S02-1000-04**
1	**S01-0020-04**	52	**S01-2600-12**	103	**S02-1000-05**
2	**S01-0020-05**	53	**S01-3000-04**	104	**S02-0900-06**
3	**S01-0045-04**	54	**S01-3000-05**	105	**S02-0900-07**
4	**S01-0045-05**	55	**S01-3000-06**	106	**S02-1200-04**
5	**S01-0065-04**	56	**S01-3000-07**	107	**S02-1200-05**
6	**S01-0065-05**	57	**S01-3000-08**	108	**S02-1500-04**
7	**S01-0090-04**	58	**S01-3300-04**	109	**S02-1500-05**
8	**S01-0090-05**	59	**S01-3300-05**	110	**S02-1500-06**
9	**S01-0125-04**	60	**S01-3300-06**	111	**S02-1500-07**
10	**S01-0125-05**	61	**S01-3300-07**	112	**S02-1900-08**
11	**S01-0180-04**	62	**S01-3300-08**	113	**S02-2000-04**
12	**S01-0180-05**	63	**S01-3300-12**	114	**S02-2000-05**
13	**S01-0230-04**	64	**S01-4000-04**	115	**S02-2050-05**
14	**S01-0230-05**	65	**S01-4000-05**	116	**S02-2050-06**
15	**S01-0315-04**	66	**S01-4000-06**	117	**S02-2050-07**
16	**S01-0315-05**	67	**S01-4000-07**	118	**S02-2500-04**
17	**S01-0290-06**	68	**S01-4000-08**	119	**S02-2500-05**
18	**S01-0405-04**	69	**S01-3300-10**	120	**S02-2050-10**
19	**S01-0405-05**	70	**S01-4000-10**	121	**S02-2600-10**
20	**S01-0470-04**	71	**S01-4800-06**	122	**S02-2600-12**
21	**S01-0470-05**	72	**S01-4800-07**	123	**S02-3000-04**
22	**S01-0590-06**	73	**S01-4800-08**	124	**S02-3000-05**
23	**S01-0610-04**	74	**S01-5200-04**	125	**S02-2500-06**
24	**S01-0610-05**	75	**S01-5200-05**	126	**S02-2500-07**
25	**S01-0740-04**	76	**S02-0025-04**	127	**S02-3000-06**
26	**S01-0740-05**	77	**S02-0025-05**	128	**S02-3000-07**
27	**S01-0900-04**	78	**S02-0050-04**	129	**S02-2500-08**
28	**S01-0900-05**	79	**S02-0050-05**	130	**S02-3000-08**
29	**S01-0900-06**	80	**S02-0075-04**	131	**S02-3300-04**
30	**S01-0900-07**	81	**S02-0075-05**	132	**S02-3300-05**
31	**S01-1200-04**	82	**S02-0100-04**	133	**S02-3300-06**
32	**S01-1200-05**	83	**S02-0100-05**	134	**S02-3300-07**
33	**S01-1500-04**	84	**S02-0140-04**	135	**S02-3300-08**
34	**S01-1500-05**	85	**S02-0140-05**	136	**S02-3300-12**
35	**S01-1500-06**	86	**S02-0200-04**	137	**S02-4000-04**
36	**S01-1500-07**	87	**S02-0200-05**	138	**S02-4000-05**
37	**S01-1900-08**	88	**S02-0260-04**	139	**S02-4000-06**
38	**S01-2000-04**	89	**S02-0260-05**	140	**S02-4000-07**
39	**S01-2000-05**	90	**S02-0350-04**	141	**S02-4000-08**
40	**S01-2000-06**	91	**S02-0350-05**	142	**S02-3300-10**
41	**S01-2000-07**	92	**S02-0320-06**	143	**S02-4000-10**
42	**S01-2050-05**	93	**S02-0450-04**	144	**S02-4800-06**
43	**S01-2050-06**	94	**S02-0450-05**	145	**S02-4800-07**
44	**S01-2050-07**	95	**S02-0520-04**	146	**S02-4800-08**
45	**S01-2500-04**	96	**S02-0520-05**	147	**S02-5200-04**
46	**S01-2500-05**	97	**S02-0650-06**	148	**S02-5200-05**
47	**S01-2500-06**	98	**S02-0680-04**	149	**S01-4000-12**
48	**S01-2500-07**	99	**S02-0680-05**	150	**S02-4000-12**
49	**S01-2500-08**	100	**S02-0820-04**		
50	**S01-2050-10**	101	**S02-0820-05**		

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
97.02	**S ConvScaleCur (set: converter DC current scaling)** Adjustment of DC current measuring channels (SDCS-PIN-4 or SDCS-PIN-51). *S ConvScaleCur (97.02)* is write protected, unless *ServiceMode (99.06)* = **SetTypeCode***:* **0 A** = take value from *TypeCode (97.01)*, default **1 A to 30000 A** = take value from *S ConvScaleCur (97.02)* This value overrides the type code and is immediately visible in *ConvNomCur (4.05)*. *ServiceMode (99.06)* has to be set back to **NormalMode** by the user. **Attention:** When using D1, D2, D3 or D4 modules the current and voltage range of the type code setting is limited to max 1000 ADC and max 600 VAC. **Int. Scaling: 1 == 1 A Type: I Volatile: N**	0	30000	0	A	E
97.03	**S ConvScaleVolt (set: converter AC voltage scaling)** Adjustment of AC voltage measuring channels (SDCS-PIN-4 or SDCS-PIN-51). *S ConvScaleVolt (97.03)* is write protected, unless *ServiceMode (99.06)* = **SetTypeCode***:* **0 V** = take value from *TypeCode (97.01)*, default **1 V to 2000 V** = take value from *S ConvScaleVolt (97.03)* This value overrides the type code and is immediately visible in *ConvNomVolt (4.04)*. *ServiceMode (99.06)* has to be set back to **NormalMode** by the user. **Attention:** When using D1, D2, D3 or D4 modules the current and voltage range of the type code setting is limited to max 1000 ADC and max 600 VAC. **Int. Scaling: 1 == 1 V Type: I Volatile: N**	0	2000	0	V	E
97.04	**S MaxBrdgTemp (set: maximum bridge temperature)** Adjustment of the converters heat sink temperature tripping level in degree centigrade: **0 °C** = take value from *TypeCode (97.01)*, default **1 °C to 149 °C** = take value from *S MaxBrdgTemp (97.04)* **150 °C** = the temperature supervision is inactive, if *S MaxBrdgTemp (97.04)* is set to 150 °C (e.g. for rebuild kits) This value overrides the type code and is immediately visible in *MaxBridgeTemp (4.17)*. **Note:** Maximum setting for converters size D6 and D7 is 55 °C, because the cooling air input temperature is measured. For more details see *DCS800 Hardware Manual*. **Int. Scaling: 1 == 1 °C Type: I Volatile: N**	0	150	0	°C	E
97.05	**ConvTempDly (converter temperature delay)** Instead of measuring the converter temperature it is possible to measure the converter fan current by means of the PW-1002/3 board. *ConvTempDly (97.05)* avoids false fault messages during the fan acceleration: **0 s** = Converter temperature measurement is released. The drive trips with **F504 ConvOverTemp** [*FaultWord1 (9.01)* bit 4] in case of excessive converter temperature, default **1 s to 300 s** = Converter fan current measurement is released when the drive is in **On** state [*UsedMCW (7.04)* bit 0 **On** = 1]. The drive trips with **F511 ConvFanCur** [*FaultWord1 (9.01)* bit 10] in case of missing or excessive converter fan current, after *ConvTempDly (97.05)* is elapsed. **Int. Scaling: 1 == 1 s Type: I Volatile: N**	0	300	0	s	E
97.06	**Unused**					
97.07	**S BlockBridge2 (set: block bridge 2)** Bridge 2 can be blocked: 0 = **Auto** operation mode is taken from *TypeCode (97.01),* default 1 = **BlockBridge2** block bridge 2 (== 2-Q operation), for e.g. 2-Q rebuild kits 2 = **RelBridge2** release bridge 2 (== 4-Q operation), for e.g. 4-Q rebuild kits This value overrides the type code and is immediately visible in *QuadrantType (4.15)*. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Auto	RelBridge2	Auto	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
97.08	**Unused**					
97.09	**MainsCompTime (mains compensation time)** Mains voltage compensation filter time constant. Is used for the mains voltage compensation at the current controller output. Setting MainsCompTime (97.09) to 1000 ms disables the mains voltage compensation. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	1000	10	ms	E
97.10	**Unused**					
97.11	**Unused**					
97.12	**CompUkPLL (phase locked loop to compensate for uk)** The measured phase angle of the firing unit's PLL can be corrected in order to compensate the error caused by the commutation related voltage drops. The compensation depends on the uk (short circuit voltage) of the mains. *CompUkPLL (97.12)* defines the mains short circuit voltage - in percent of *NomMainsVolt (99.10)* - which is caused by the converter's nominal current for the PLL correction: $$\mathrm{CompUkPLL} = \mathrm{uk} * \frac{S_c}{S_t} * 100\%$$ with: uk = related mains short circuit voltage, S_c = apparent power of converter and S_t = apparent power of transformer **Commissioning hint:** *CompUkPLL (97.12)* is used to compensate for the phase shift of the mains due to commutation notches, in case the mains are measured on the secondary side of the dedicated transformer. The whole situation leads to unstable armature current during high motor loads. Increase *CompUkPLL (97.12)* slowly (1 by 1) until the armature current becomes stable. **Int. Scaling: 10 == 1 % Type: I Volatile: N**	0	15	0	%	E
97.13	**DevLimPLL (phase locked loop deviation limit)** Maximum allowed deviation of the PLL controller. The current controller is blocked in case the limit is reached - see *CurCtrlStat2 (6.04)* bit 13: – for 50 Hz mains is valid: $360° == 20ms = \frac{1}{50Hz} == 20.000$ – for 60 Hz mains is valid: $360° == 16.67ms = \frac{1}{60Hz} == 16.667$ The PLL input can be seen in *PLLIn (3.20)*. The PLL output can be seen in *MainsFreqAct (1.38)*. **Int. Scaling: 100 == 1 ° Type: I Volatile: N**	5	20	10	°	E
97.14	**KpPLL (phase locked loop p-part)** Gain of firing unit's phase lock loop. **Int. Scaling: 100 == 1 Type: I Volatile: N**	0.25	8	2	-	E
97.15	**TfPLL (phase locked loop filter)** Filter of firing unit's phase lock loop. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	1000	0	ms	E
97.16	**AdjIDC (adjust DC current)** *AdjIDC (97.16)* is used to cover drives with different current measuring circuits for bridge 1 and bridge 2. It rescales the measured armature current if bridge2 is active. **Int. Scaling: 10 == 1 % Type: I Volatile: N**	12.5	800	100	%	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
97.17	**OffsetIDC (offset DC current measurement)** Offset value - in percent of *M1NomCur (99.03)* - added to the armature current measurement. *OffsetIDC (97.17)* adjusts *ConvCurAct (1.16)* and the real armature current. Setting *OffsetIDC (97.17)* to 0 disables the manual offset. **Commissioning hint:** In case a 2-Q converter module is used and the motor turns with speed reference equals zero increase *OffsetIDC (97.17)* until the motor is not turning anymore. **Int. Scaling: 100 == 1 % Type: I Volatile: N**	-5	5	0	%	E
97.18	**ZeroCurDetect (zero current detection)** Selects the zero current detection method. Use a binary signal, if the zero current detection is done by another converter: 0 = **Current** based on the converter's own zero current detection resistors, default 1 = **Voltage** based on the converter's own thyristor voltages, not valid when galvanic isolation is used 2 = **CurAndVolt** based on discontinuous current and thyristor voltages, not valid when galvanic isolation is used 3 = **DI1** 1 = zero current detected, 0 = current not zero 4 = **DI2** 1 = zero current detected, 0 = current not zero 5 = **DI3** 1 = zero current detected, 0 = current not zero 6 = **DI4** 1 = zero current detected, 0 = current not zero 7 = **DI5** 1 = zero current detected, 0 = current not zero 8 = **DI6** 1 = zero current detected, 0 = current not zero 9 = **DI7** 1 = zero current detected, 0 = current not zero 10 = **DI8** 1 = zero current detected, 0 = current not zero 11 = **DI9** 1 = zero current detected, 0 = current not zero, only available with digital extension board 12 = **DI10** 1 = zero current detected, 0 = current not zero, only available with digital extension board 13 = **DI11** 1 = zero current detected, 0 = current not zero, only available with digital extension board 14 = **MCW Bit11** 1 = zero current detected, 0 = current not zero, *MainCtrlWord (7.01)* bit 11 15 = **MCW Bit12** 1 = zero current detected, 0 = current not zero, *MainCtrlWord (7.01)* bit 12 16 = **MCW Bit13** 1 = zero current detected, 0 = current not zero, *MainCtrlWord (7.01)* bit 13 17 = **MCW Bit14** 1 = zero current detected, 0 = current not zero, *MainCtrlWord (7.01)* bit 14 18 = **MCW Bit15** 1 = zero current detected, 0 = current not zero, *MainCtrlWord (7.01)* bit 15 19 = **ACW Bit12** 1 = zero current detected, 0 = current not zero, *AuxCtrlWord (7.02)* bit 12 20 = **ACW Bit13** 1 = zero current detected, 0 = current not zero, *AuxCtrlWord (7.02)* bit 13 21 = **ACW Bit14** 1 = zero current detected, 0 = current not zero, *AuxCtrlWord (7.02)* bit 14 22 = **ACW Bit15** 1 = zero current detected, 0 = current not zero, *AuxCtrlWord (7.02)* bit 15 **Note:** If zero current is detected by means of the thyristor voltages either 10 % of *MainsVoltAct (1.11)* or 10 V is undershot. **Int. Scaling: 1 == 1 Type: C Volatile: N**	Current	ACW Bit15	Current	-	E

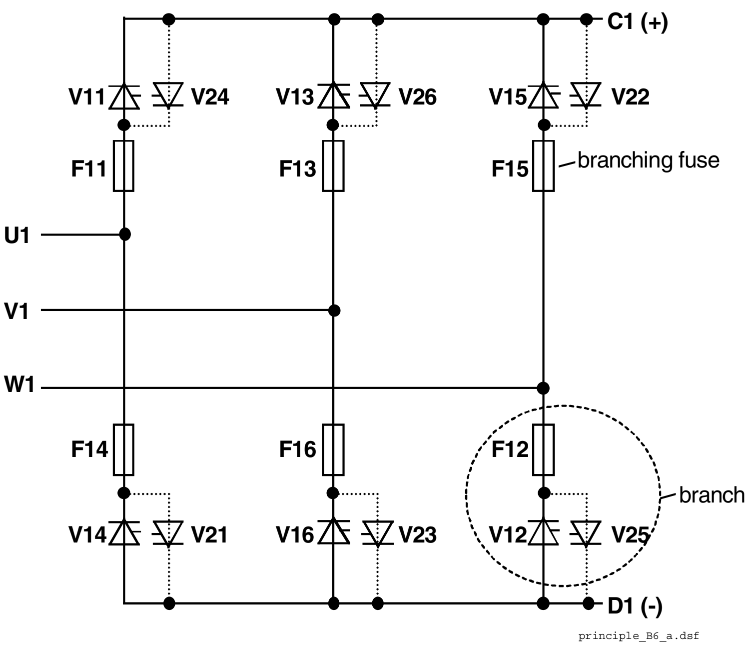
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
97.19	**ZeroCurTimeOut (zero current timeout)** After a command to change current direction - see *CurRefUsed (3.12)* - the opposite current has to be reached before *ZeroCurTimeOut (97.19)* has been elapsed otherwise the drive trips with **F557 ReversalTime** [*FaultWord4 (9.04)* bit 8].  The reversal delay starts when zero current has been detected - see *CurCtrlStat1 (6.03)* bit 13 - after a command to change current direction - see *CurRefUsed (3.12)* - has been given. The time needed to change the current direction can be longer when changing from motoring mode to regenerative mode at high motor voltages, because the motor voltage must be reduced before switching to regenerative mode - see also *RevVoltMargin (44.21)*. *ZeroCurTimeOut (97.19)* must have the same setting for 12-pulse master and 12-pulse slave with one exception only: If there is no current measurement in the 12-pulse serial slave, set *ZeroCurTimeOut (97.19)* in the 12-pulse serial slave to maximum (12000 ms). **Note:** *12P RevTimeOut (47.05)* must be longer than *ZeroCurTimeOut (97.19)* and *ZeroCurTimeOut (97.19)* must be longer than *RevDly (43.14)*. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	12000	20	ms	E
97.20	**TorqActFiltTime (actual torque filter time)** Torque actual filter time constant for *MotTorqFilt (1.07)*. Is used for the EMF controller and the EMF feed forward. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	1000	ms	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
97.21	**ResetAhCounter (reset ampere hour counter)** Binary signal to reset *AhCounter (1.39):* 0 = **NotUsed** default 1 = **DI1** **Reset** by rising edge (0 → 1) 2 = **DI2** **Reset** by rising edge (0 → 1) 3 = **DI3** **Reset** by rising edge (0 → 1) 4 = **DI4** **Reset** by rising edge (0 → 1) 5 = **DI5** **Reset** by rising edge (0 → 1) 6 = **DI6** **Reset** by rising edge (0 → 1) 7 = **DI7** **Reset** by rising edge (0 → 1) 8 = **DI8** **Reset** by rising edge (0 → 1) 9 = **DI9** **Reset** by rising edge (0 → 1), only available with digital extension board 10 = **DI10** **Reset** by rising edge (0 → 1), only available with digital extension board 11 = **DI11** **Reset** by rising edge (0 → 1), only available with digital extension board 12 = **MCW Bit11** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 11 13 = **MCW Bit12** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 12 14 = **MCW Bit13** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 13 15 = **MCW Bit14** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 14 16 = **MCW Bit15** **Reset** by rising edge (0 → 1), *MainCtrlWord (7.01)* bit 15 17 = **ACW Bit12** **Reset** by rising edge (0 → 1), *AuxCtrlWord (7.02)* bit 12 18 = **ACW Bit13** **Reset** by rising edge (0 → 1), *AuxCtrlWord (7.02)* bit 13 19 = **ACW Bit14** **Reset** by rising edge (0 → 1), *AuxCtrlWord (7.02)* bit 14 20 = **ACW Bit15** **Reset** by rising edge (0 → 1), *AuxCtrlWord (7.02)* bit 15 **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	ACW Bit15	NotUsed	-	E
97.22	**Unused**					
97.23	**AdjUDC (adjust DC voltage)** *AdjUDC (97.23)* is used to cover drives with different voltage measuring circuits for armature and mains voltage. It rescales the armature voltage measurement. **Int. Scaling: 10 == 1 % Type: I Volatile: N**	12.5	800	100	%	E
97.24	**OffsetUDC (offset DC voltage measurement)** Offset value - in percent of *M1NomVolt (99.02)* - added to the armature voltage measurement. *OffsetUDC (97.24)* adjusts *ArmVoltAct (1.14)* and the real armature voltage. Setting *OffsetUDC (97.24)* to 5.1 % disables the manual offset. If a DC-breaker is used set *OffsetUDC (97.24)* = 0 **Int. Scaling: 100 == 1 % Type: I Volatile: N**	-5.0	5.1	5.1	%	E
97.25	**EMF ActFiltTime (actual EMF filter time)** EMF actual filter time constant for *EMF VoltActRel (1.17).* Is used for the EMF controller and the EMF feed forward. **Int. Scaling: 1 == 1 ms Type: I Volatile: N**	0	10000	10	ms	E
97.26	**HW FiltUDC (hardware filter DC voltage measurement)** Hardware filter for the UDC measuring circuit: 0 = **FilterOff** the filter time is set to 200 µs 1 = **FilterOn** the filter time is set to 10 ms, default **Int. Scaling: 1 == 1 Type: C Volatile: N**	FilterOff	FilterOn	FilterOn	-	E
97.27	**Measurement (measurement)** reserved **Int. Scaling: 1 == 1 Type: I Volatile: N**	0	1000	0	-	E

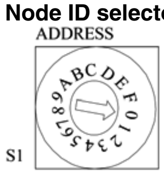
<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td>97.28</td><td>

TestFire (type of thyristor diagnosis)
The thyristor diagnosis is started by setting *ServiceMode (99.06)* = **ThyDiagnosis**. *TestFire (97.28)* defines which type of thyristor diagnosis should be used:

0 = **Off** all thyristors are tested, the result is shown in *Diagnosis (9.11)*, default
1 = **V11** firing pulses for thyristor V11 are released
2 = **V12** firing pulses for thyristor V12 are released
3 = **V13** firing pulses for thyristor V13 are released
4 = **V14** firing pulses for thyristor V14 are released
5 = **V15** firing pulses for thyristor V15 are released
6 = **V16** firing pulses for thyristor V16 are released
7 = **V21** firing pulses for thyristor V21 are released
8 = **V22** firing pulses for thyristor V22 are released
9 = **V23** firing pulses for thyristor V23 are released
10 = **V24** firing pulses for thyristor V24 are released
11 = **V25** firing pulses for thyristor V25 are released
12 = **V26** firing pulses for thyristor V26 are released



Int. Scaling: 1 == 1 **Type: C** **Volatile: N**
</td><td>Off</td><td>V26</td><td>Off</td><td>-</td><td>E</td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 98	**Option modules**					
98.01	**Encoder2Module (encoder 2 extension module)** This parameter is used to activate an extension module for either a second encoder (RTAC-xx) or a resolver (RRIA-xx). RTAC-xx / RRIA-xx extension module interface selection. *Encoder2Module (98.01)* releases pulse encoder 2 or a resolver. The modules can be connected in option slot 1, 2, 3 or alternatively onto the external I/O module adapter (AIMA) connected via SDCS-COM-8. The node ID 0 (see Node ID selector S1) is only required for connection via AIMA: 0 = **NotUsed** no RTAC-xx / RRIA-xx is used, default 1 = **Slot1** RTAC-xx / RRIA-xx is connected in option slot 1 2 = **Slot2** RTAC-xx / RRIA-xx is connected in option slot 2 3 = **Slot3** RTAC-xx / RRIA-xx is connected in option slot 3 4 = **AIMA** RTAC-xx / RRIA-xx is connected onto the external I/O module adapter (AIMA), node ID = 0 The drive trips with **F508 I/OBoardLoss** [*FaultWord1 (9.01)* bit 7], if the RTAC-xx / RRIA-xx extension module is chosen, but not connected or faulty. **Attention:** To ensure proper connection and communication of the RTAC-xx / RRIA-xx board with the SDCS-CON-4 use the screws included in the scope of delivery. **Switches on RTAC-xx** or **RRIA-xx:** **Node ID selector (S1)** is only valid when plugged in an AIMA board  **Int. Scaling:** **1 == 1** **Type:** **C** **Volatile: N**	NotUsed	AIMA	NotUsed	-	E

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
98.02	**CommModule (communication modules)** For the communication modules following selections are available:	NotUsed	FldBusModbus	NotUsed	-	E

	Fieldbus (R-type)	DDCS (e.g. AC 800M)	DDCS (N-type fieldbus)	Modbus (RMBA-xx)
0	-	-	-	-
1	X	-	-	-
2	-	X	-	-
3	-	-	X	-
4	-	-	-	X
5	X (read only)	X	-	-
6	-	X	-	X (read only)
7	-	-	X	X (read only)
8	X	-	-	X /read only)

0 = **NotUsed** no communication used, default
1 = **Fieldbus** The drive communicates with the overriding control via an R-type fieldbus adapter connected in option slot 1. The data set base address has to be set to 1, set *Ch0 DsetBaseAddr (70.24)* = 1. This choice is not valid for the Modbus.
2 = **COM-8/AC800x** The drive communicates with the ABB overriding control via SDCS-COM-8 connected in option slot 3. The data set base address is selected by means of *Ch0 DsetBaseAddr (70.24).*
3 = **COM-8/Nxxx** The drive communicates with the overriding control via SDCS-COM-8 connected in option slot 3 and an N-type fieldbus adapter. The data set base address has to be set to 1, set *Ch0 DsetBaseAddr (70.24)* = 1.
4 = **Modbus** The drive communicates with the overriding control via the Modbus (RMBA-xx) connected in option slot 1, for that set *ModBusModule2 (98.08)* = **Slot1**. The data set base address has to be set to 1, set *Ch0 DsetBaseAddr (70.24)* = 1.
5 = **AC800xFldbus** The drive communicates with the ABB overriding control via SDCS-COM-8 connected in option slot 3. The data set base address is selected by means of *Ch0 DsetBaseAddr (70.24).*
An additional R-type fieldbus adapter connected in option slot 1 is used for monitoring purposes only. This choice is not valid for the Modbus.
6 = **AC800xModbus** The drive communicates with the ABB overriding control via SDCS-COM-8 connected in option slot 3. The data set base address is selected by means of *Ch0 DsetBaseAddr (70.24).*
An additional Modbus (RMBA-xx) connected in option slot 1 or 2 [see *ModBusModule2 (98.08)*] is used for monitoring purposes only.
7 = **NxxxModbus** The drive communicates with the overriding control via SDCS-COM-8 connected in option slot 3 and an N-type fieldbus adapter. The data set base address is selected by means of *Ch0 DsetBaseAddr (70.24).*
An additional Modbus (RMBA-xx) connected in option slot 1 or 2 [see *ModBusModule2 (98.08)*] is used for monitoring purposes only.
8 = **FldBusModbus** The drive communicates with the overriding control via an R-type fieldbus adapter connected in option slot 1. The data set base address has to be set to 1, set *Ch0 DsetBaseAddr (70.24)* = 1. This choice is not valid for the Modbus.
An additional Modbus (RMBA-xx) connected in option slot 2 or 3 [see *ModBusModule2 (98.08)*] is used for monitoring purposes only.

The drive trips with **F508 I/OBoardLoss** [*FaultWord1 (9.01)* bit 7], if the communication module configuration is not met.
Attention:
To ensure proper connection and communication of the communication modules with the SDCS-CON-4 use the screws included in the scope of delivery.
Int. Scaling: 1 == 1 Type: C Volatile: N

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
98.03	**DIO ExtModule1 (digital extension module 1)**	NotUsed	AIMA	NotUsed	-	E

First RDIO-xx extension module interface selection. *DIO ExtModule1 (98.03)* releases DI9, DI10, DI11, DO9 and DO10.
The module can be connected in option slot 1, 2, 3 or alternatively onto the external I/O module adapter (AIMA) connected via SDCS-COM-8. The node ID 2 (see Node ID selector S1) is only required for connection via AIMA:

- 0 = **NotUsed** no first RDIO-xx is used, default
- 1 = **Slot1** first RDIO-xx is connected in option slot 1
- 2 = **Slot2** first RDIO-xx is connected in option slot 2
- 3 = **Slot3** first RDIO-xx is connected in option slot 3
- 4 = **AIMA** first RDIO-xx is connected onto the external I/O module adapter (AIMA), node ID = 2

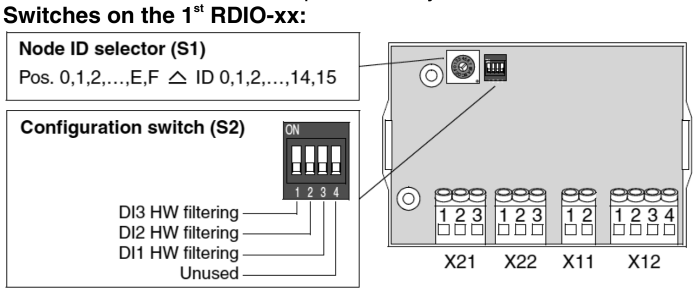
The drive trips with **F508 I/OBoardLoss** [*FaultWord1 (9.01)* bit 7], if the RDIO-xx extension module is chosen, but not connected or faulty.

Note:
For faster input signal detection disable the hardware filters of the RDIO-xx by means of dip switch S2. Always have the hardware filter enabled when an AC signal is connected.

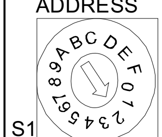
Note:
The digital outputs are available via *DO CtrlWord (7.05)*.

Attention:
To ensure proper connection and communication of the RDIO-xx board with the SDCS-CON-4 use the screws included in the scope of delivery.

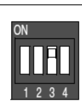
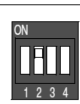
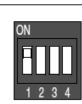
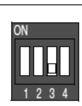
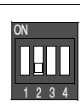
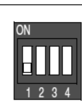
Switches on the 1st RDIO-xx:



Node ID selector (S1) is only valid when plugged in an AIMA board



Configuration switch (S2)
For faster detection the hardware filter of the digital input in question can be disabled. Disabling the hardware filtering will however reduce the noise immunity of the input.

Hardware Filtering	DIP switch settings		
	Digital input DI1	Digital input DI2	Digital input DI3
Enabled (Default)			
Disabled			

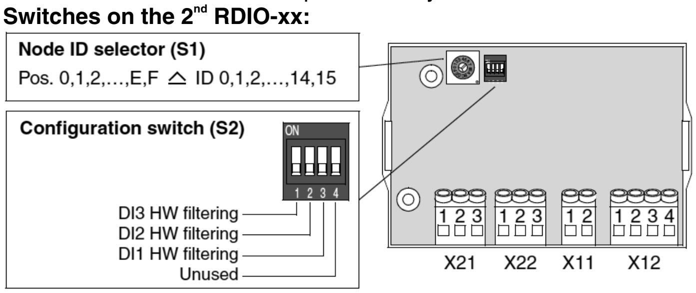
Int. Scaling: 1 == 1 Type: C Volatile: N

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
98.04	**DIO ExtModule2 (digital extension module 2)**	NotUsed	AIMA	NotUsed	-	E

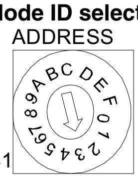
Second RDIO-xx extension module interface selection. *DIO ExtModule2 (98.04)* releases DI12, DI13, DI14, DO11 and DO12.
The module can be connected in option slot 1, 2, 3 or alternatively onto the external I/O module adapter (AIMA) connected via SDCS-COM-8. The node ID 3 (see Node ID selector S1) is only required for connection via AIMA:

- 0 = **NotUsed** no second RDIO-xx is used, default
- 1 = **Slot1** second RDIO-xx is connected in option slot 1
- 2 = **Slot2** second RDIO-xx is connected in option slot 2
- 3 = **Slot3** second RDIO-xx is connected in option slot 3
- 4 = **AIMA** second RDIO-xx is connected onto the external I/O module adapter (AIMA), node ID = 3

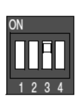
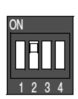
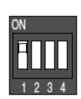
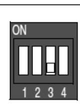
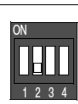
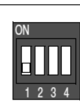
The drive trips with **F508 I/OBoardLoss** [*FaultWord1 (9.01)* bit 7], if the RDIO-xx extension module is chosen, but not connected or faulty.
Note:
For faster input signal detection disable the hardware filters of the RDIO-xx by means of dip switch S2. Always have the hardware filter enabled when an AC signal is connected.
Note:
The digital inputs are available via *DI StatWord (8.05)*
The digital outputs are available via *DO CtrlWord (7.05).*
Attention:
To ensure proper connection and communication of the RDIO-xx board with the SDCS-CON-4 use the screws included in the scope of delivery.
Switches on the 2nd RDIO-xx:



Node ID selector (S1) is only valid when plugged in an AIMA board



Configuration switch (S2)
For faster detection the hardware filter of the digital input in question can be disabled. Disabling the hardware filtering will however reduce the noise immunity of the input.

Hardware Filtering	DIP switch settings		
	Digital input DI1	Digital input DI2	Digital input DI3
Enabled (Default)			
Disabled			

Int. Scaling: 1 == 1 **Type: C** **Volatile: N**

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
98.05	**Unused**					
98.06	**AIO ExtModule (analog extension module)**	NotUsed	AIMA	NotUsed	-	E

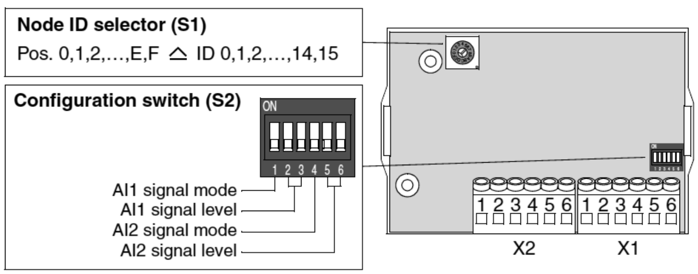
First RAIO-xx extension module interface selection. *AIO ExtModule (98.06)* releases AI5, AI6, AO3 and AO4.
The module can be connected in option slot 1, 2, 3 or alternatively onto the external I/O module adapter (AIMA) connected via SDCS-COM-8. The node ID 5 (see Node ID selector S1) is only required for connection via AIMA:

0 = **NotUsed** no first RAIO-xx is used, default
1 = **Slot1** first RAIO-xx is connected in option slot 1
2 = **Slot2** first RAIO-xx is connected in option slot 2
3 = **Slot3** first RAIO-xx is connected in option slot 3
4 = **AIMA** first RAIO-xx is connected onto the external I/O module adapter (AIMA), node ID = 5

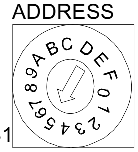
The drive trips with **F508 I/OBoardLoss** [*FaultWord1 (9.01)* bit 7], if the RAIO-xx extension module is chosen, but not connected or faulty.

Attention:
To ensure proper connection and communication of the RAIO-xx board with the SDCS-CON-4 use the screws included in the scope of delivery.

Switches on the 1st RAIO-xx:



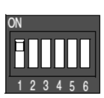
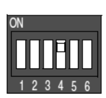
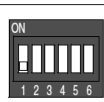
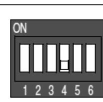
Node ID selector (S1) is only valid when plugged in an AIMA board
ADDRESS

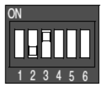
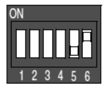
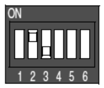
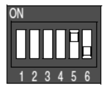
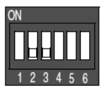
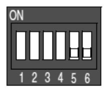


S1

Configuration switch (S2)
The operation of the analog inputs can be selected using the configuration DIP switch (S2) on the circuit board of the module. The drive parameters must be set accordingly.

Input mode selection:
In bipolar mode, the analog inputs can handle positive and negative signals. The resolution of the A/D conversion is 11 data bits (+ 1 sign bit). In unipolar mode (default), the analog inputs can handle positive signals only. The resolution of the A/D conversion is 12 data bits.

DIP switch setting		Input signal type
Analogue input AI1	Analogue input AI2	
		±0(4)…20 mA ±0(2)…10 V ±0…2 V
		0(4)…20 mA 0(2)…10 V 0…2 V (Default)

<table>
<tr><th>Index</th><th>Signal / Parameter name</th><th>min.</th><th>max.</th><th>def.</th><th>unit</th><th>E/C</th></tr>
<tr><td></td><td>Input signal type selection:
Each input can be used with a current or voltage signal.
<table>
<tr><th>Input signal type</th><th>DIP switch settings
Analogue input 1</th><th>DIP switch settings
Analogue input 2</th></tr>
<tr><td>Current signal
±0(4)…20 mA
(Default)</td><td></td><td></td></tr>
<tr><td>Voltage signal
±0(2)…10 V</td><td></td><td></td></tr>
<tr><td>Voltage signal
±0…2 V</td><td></td><td></td></tr>
</table>
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>98.07</td><td>Unused</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>98.08</td><td>ModBusModule2 (Modbus module 2)
The Modbus module (RMBA-xx) can be connected in option slot 1, 2 or 3 [see also CommModule (98.02)]:
0 = NotUsed no RMBA-xx is used, default
1 = Slot1 RMBA-xx is connected in option slot 1
2 = Slot2 RMBA-xx is connected in option slot 2
3 = Slot3 RMBA-xx is connected in option slot 3
4 = DSL reserved
Int. Scaling: 1 == 1 Type: C Volatile: N</td><td>NotUsed</td><td>Slot3</td><td>NotUsed</td><td>-</td><td>E</td></tr>
<tr><td>98.09</td><td>Unused</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>98.10</td><td>Unused</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>98.11</td><td>Unused</td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
98.12	**AIO MotTempMeas (analog extension module for motor temperature measurement)**	NotUsed	AIMA	NotUsed	-	E

Second RAIO-xx extension module interface selection. *AIO MotTempMeas (98.12)* releases AI7, AI8, AO5 and AO6. The analog in- and outputs are only used for motor temperature measurement [see *M1TempSel (31.05)* and *M2TempSel (49.33)*].

The module can be connected in option slot 1, 2, 3 or alternatively onto the external I/O module adapter (AIMA) connected via SDCS-COM-8. The node ID 9 (see Node ID selector S1) is only required for connection via AIMA:

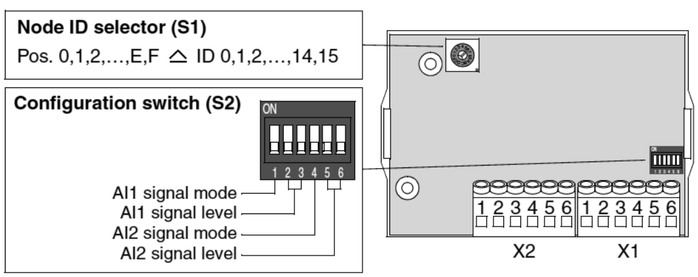
- 0 = **NotUsed** no second RAIO-xx is used, default
- 1 = **Slot1** second RAIO-xx is connected in option slot 1
- 2 = **Slot2** second RAIO-xx is connected in option slot 2
- 3 = **Slot3** second RAIO-xx is connected in option slot 3
- 4 = **AIMA** second RAIO-xx is connected onto the external I/O module adapter (AIMA), node ID = 9

The drive trips with **F508 I/OBoardLoss** [*FaultWord1 (9.01)* bit 7], if the RAIO extension module is chosen, but not connected or faulty.

Attention:

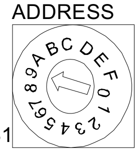
To ensure proper connection and communication of the RAIO-xx board with the SDCS-CON-4 use the screws included in the scope of delivery.

Switches on the 2nd RAIO-xx:



Node ID selector (S1) is only valid when plugged in an AIMA board

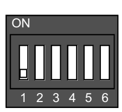
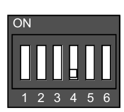
ADDRESS



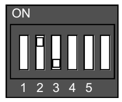
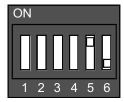
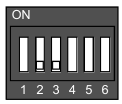
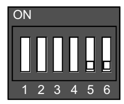
S1

Configuration switch (S2)

For temperature measurement set the operating mode to unipolar and

DIP switch setting (unipolar)		Input signal type
Analog input AI1	Analog input AI2	
		0(4) ... 20 mA 0(2) ... 10 V 0 ... 2 V (Default)

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	set the number of connected PT100 per channel.					

Input signal type	DIP switch settings	
	Analog input AI1	
2 or 3 PT100 set the voltage signal to 0 … 10 V		
1 PT100 set the voltage signal to 0 … 2 V		

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
	Int. Scaling: 1 == 1 Type: C Volatile: N					
98.13	**Unused**					
98.14	**Unused**					
98.15	**IO BoardConfig (I/O board configuration)** *IO BoardConfig (98.15)* selects the optional interface boards (SDCS-IOB-2 and / or SDCS-IOB-3) for the standard I/O of the SDCS-CON-4: 0 = **NotUsed** no optional interface boards connected, default 1 = **SDCS-IOB-2** only SDCS-IOB-2 connected 2 = **SDCS-IOB-3** only SDCS-IOB-3 connected 3 = **IOB-2+IOB-3** SDCS-IOB-2 and SDCS-IOB-3 connected The drive trips with **F508 I/OBoardLoss** [*FaultWord1 (9.01)* bit 7], if the IO board configuration is not met [e.g. one or two boards are physically connected, but not selected by *IO BoardConfig (98.15)*]. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	IOB-2+IOB-3	NotUsed	-	E
98.16	**Unused**					

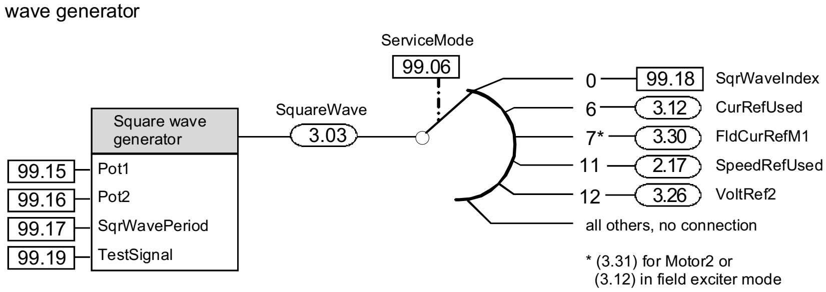
Index	Signal / Parameter name	min.	max.	def.	unit	E/C
Group 99	**Start-up data**					
99.01	**Language (language)** Select language: 0 = **English** default 1 = **English AM** not implemented yet 2 = **Deutsch** 3 = **Italiano** 4 = **Español** 5 = **Português** not implemented yet 6 = **Nederlands** not implemented yet 7 = **Français** 8 = **Dansk** not implemented yet 9 = **Suomi** not implemented yet 10 = **Svenska** not implemented yet 11 = **Po-Russki** not implemented yet 12 = **Polski** 13 = **Turkish** not implemented yet 14 = **Cesky** not implemented yet **Int. Scaling: 1 == 1 Type: C Volatile: N**	English	Cesky	English	-	C
99.02	**M1NomVolt (motor 1 nominal DC voltage)** Motor 1 nominal armature voltage (DC) from the motor rating plate. **Note:** In 12-pulse serial mode, this parameter has to be set to the value of the voltage the converter itself is providing. This is usually 50 % of the rated motor voltage, if one motor is connected. In case 2 motors in series are connected it is 100 % of one motor's rated voltage. **Int. Scaling: 1 == 1 V Type: I Volatile: N**	5	2000	350	V	C
99.03	**M1NomCur (motor 1 nominal DC current)** Motor 1 nominal armature current (DC) from the motor rating plate. If several motors are connected to the drive, enter the total current of all motors. **Note:** In 12-pulse parallel mode, this parameter has to be set to the value of the current the converter itself is providing. This is usually 50 % of the rated motor current, if one motor is connected. In case 2 motors in parallel are connected it is 100 % of one motor's rated current. **Note:** In case the converter is used as a 3-phase field exciter use *M1NomCur (99.03)* to set the nominal field current. **Int. Scaling: 1 == 1 A Type: I Volatile: N**	0	30000	0	A	C
99.04	**M1BaseSpeed (motor 1 base speed)** Motor 1 base speed from the rating plate, usually the field weak point. *M1BaseSpeed (99.04)* is must be set in the range of: 0.2 to 1.6 times of *SpeedScaleAct (2.29).* If the scaling is out of range **A124 SpeedScale** [*AlarmWord2 (9.07)* bit 7] is generated. **Int. Scaling: 10 == 1 rpm Type: I Volatile: N**	10	6500	1500	rpm	C
99.05	**Unused**					

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
99.06	**ServiceMode (service mode)** *ServiceMode (99.06)* contains several test modes, auto- and manual tuning procedures. The drive mode is automatically set to **NormalMode** after an autotuning procedure or after the thyristor diagnosis is finished or failed. In case errors occur during the selected procedure **A121 AutotuneFail** [*AlarmWord2 (9.07)* bit 4] is generated. The reason of the error can be seen in *Diagnosis (9.11)*. **SetTypeCode** is automatically set to **NormalMode** after the next power up. 0 = **NormalMode** normal operating mode depending on *OperModeSel (43.01)*, default 1 = **ArmCurAuto** autotuning armature current controller 2 = **FieldCurAuto** autotuning field current controller 3 = **EMF FluxAuto** autotuning EMF controller and flux linearization 4 = **SpdCtrlAuto** autotuning speed controller 5 = **SpdFbAssist** test speed feedback, see *M1EncMeasMode (50.02), M1SpeedFbSel (50.03), M1EncPulseNo (50.04)* and *M1TachoVolt1000 (50.13)* 6 = **ArmCurMan** manual tuning of armature current controller 7 = **FieldCurMan** manual tuning of field current controller 8 = **ThyDiagnosis** the thyristor diagnosis mode is set with *TestFire (97.28),* the result is shown in *Diagnosis (9.11)* 9 = **FldRevAssist** test field reversal 10 = **SetTypeCode** set type code, releases following parameters: *TypeCode (97.01)* *S ConvScaleCur (97.02)* *S ConvScaleVolt (97.03)* *S M1FldScale (45.20)* *S M2FldScale (45.21)* 11 = **SpdCtrlMan** manual tuning of speed controller 12 = **EMF Man** manual tuning of EMF controller 13 = **Simulation** reserved 14 = **TachFineTune** tacho fine tuning, see *M1TachoAdjust (50.12)* 15 = **LD FB Config** reserved for future use (load fieldbus configuration file) 16 = **DeleteAppl** releases *ParApplSave (16.06)* = **DeleteAppl** 17 = **FindDiscCur** find discontinuous current limit **Note:** The reference chain is blocked while ServiceMode (99.06) ≠ **NormalMode**. **Note:** Depending on *MotSel (8.09)* the field current of motor 1 or motor 2 is tuned. **Note:** A standard DCS800 converter used as field exciter cannot be tuned by means of its armature converter. Tune it by setting *ServiceMode (99.06)* = **FieldCurAuto** in the field exciter itself. **Int. Scaling: 1 == 1 Type: C Volatile: Y**	NormalMode	FindDiscCur	NormalMode	-	C
99.07	**ApplRestore (application restore)** Setting *ApplRestore (99.07)* = **Yes** starts the loading / storing of the macro (preset parameter set) selected by means of *ApplMacro (99.08). ApplRestore (99.07)* is automatically set back to **Done** after the chosen action is finished: 0 = **Done** no action or macro change completed, default 1 = **Yes** macro selected with *ApplMacro (99.08)* will be loaded into the drive **Note:** Macro changes are only accepted in **Off** state [*MainStatWord (8.01)* bit 1 = 0]. **Note:** It takes about 2 s, until the new parameter values are active. **Int. Scaling: 1 == 1 Type: C Volatile: Y**	Done	Yes	Done	-	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
99.08	**ApplMacro (application macro)** *ApplMacro (99.08)* selects the macro (preset parameter sets) to be loaded / stored into the RAM and flash. In addition to the preset macros, two user-defined macros (**User1** and **User2**) are available. The operation selected by *ApplMacro (99.08)* is **started** immediately by setting *ApplRestore (99.07)* = **Yes**. *ApplMacro (99.08)* is automatically set back to **NotUsed** after the chosen action is finished. The selected macro is shown in *MacroSel (8.10)*: 0 = **NotUsed** default 1 = **Factory** load macro factory (default parameter set) into RAM and flash - **User1** and **User2** will not be influenced 2 = **User1Load** load macro **User1** into RAM and flash 3 = **User1Save** save actual parameter set form RAM into macro **User1** 4 = **User2Load** load macro **User2** into RAM and flash 5 = **User2Save** save actual parameter set form RAM into macro **User2** 6 = **Standard** load macro standard into RAM and flash 7 = **Man/Const** load macro manual / constant speed into RAM and flash 8 = **Hand/Auto** load macro hand (manual) / automatic into RAM and flash 9 = **Hand/MotPot** load macro hand (manual) / motor potentiometer into RAM and flash 10 = reserved reserved 11 = **MotPot** load macro motor potentiometer into RAM and flash 12 = **TorqCtrl** load macro torque control into RAM and flash 13 = **TorqLimit** load macro torque limit into RAM and flash 14 = **DemoStandard** load macro demo standard into RAM and flash 15 = **2WreDCcontUS** load macro 2 wire with US style DC-breaker into RAM and flash 16 = **3WreDCcontUS** load macro 3 wire with US style DC-breaker into RAM and flash 17 = **3WreStandard** load macro 3 wire standard into RAM and flash **Note:** When loading a macro, group 99 is set / reset as well. **Note:** If **User1** is active AuxStatWord (8.02) bit 3 is set. If **User2** is active AuxStatWord (8.02) bit 4 is set. **Note:** It is possible to change all preset parameters of a loaded macro. On a macro change or an application restore command of the actual macro the macro depending parameters are restored to the macro's default values. **Note:** In case macro **User1** or **User2** is loaded by means of *ParChange (10.10)* it is not saved into the flash and thus not valid after the next power on. **Note:** The DriveWindow backup function only saves the active macro. Thus both macros **User1** and **User2** must be backed-up separately. **Int. Scaling: 1 == 1 Type: C Volatile: Y**	NotUsed	3WreStandard	NotUsed	-	C
99.09	**DeviceName (device name)** The user can set a drive number by means of the DCS800 Control Panel or DriveWindow Light. With DriveWindow it is possible to fill in a string (name) with a maximum of 12 characters. This name will override the numbers and is shown as well in the DCS800 Control Panel and in DriveWindow. **Int. Scaling: 1 == 1 Type: I/C Volatile: N**	0	65535	0	-	E
99.10	**NomMainsVolt (nominal AC mains voltage)** Nominal mains voltage (AC) of the supply. The default and maximum values are preset automatically according to *TypeCode (97.01)* respectively *S ConvScaleVolt (97.03)*. Absolute max. is 1200 V **Int. Scaling: 1 == 1 V Type: I Volatile: N**	0	(97.01) / (97.03)	(97.01) / (97.03)	V	C

Index	Signal / Parameter name	min.	max.	def.	unit	E/C
99.11	**M1NomFldCur (motor 1 nominal field current)** Motor 1 nominal field current from the motor rating plate. **Note:** In case the converter is used as a 3-phase field exciter use *M1NomCur (99.03)* to set the nominal field current. **Int. Scaling: 100 == 1 A Type: I Volatile: N**	0.3	655	0.3	A	C
99.12	**M1UsedFexType (motor 1 used field exciter type)** Motor 1 used field exciter type: 0 = **NotUsed** no or third party field exciter connected 1 = **OnBoard** integrated 1-Q field exciter (for sizes D1 - D4 only), default 2 = **FEX-425-Int** internal 1-Q 25 A field exciter (for size D5 only) used for field currents from 0.3 A to **25 A** (terminals X100.1 and X100.3) 3 = **DCF803-0035** external 1-Q 35 A field exciter used for field currents from 0.3 A to **35 A** (terminals X100.1 and X100.3) 4 = **DCF803-0050** external 1-Q 50 A field exciter (DCF803-0050 or DCF503B-0050) 5 = **DCF804-0050** external 4-Q 50 A field exciter (DCF804-0050 or DCF504B-0050) 6 = **DCF803-0060** external 1-Q 60 A field exciter; not implemented yet 7 = **DCF804-0060** external 4-Q 60 A field exciter; not implemented yet 8 = **DCS800-S01** external 2-Q 3-phase field exciter 9 = **DCS800-S02** external 4-Q 3-phase field exciter 10 = **DCF803-0016** external 1-Q 16 A field exciter used for field currents from 0.3 A to **16 A** (terminals X100.1 and X100.3) 11 = reserved to 14 = reserved 15 = **ExFex AITAC** third party field exciter, acknowledge via AITAC 16 = **ExFex AI1** third party field exciter, acknowledge via AI1 17 = **ExFex AI2** third party field exciter, acknowledge via AI2 18 = **ExFex AI3** third party field exciter, acknowledge via AI3 19 = **ExFex AI4** third party field exciter, acknowledge via AI4 20 = **FEX-4-Term5A** internal 2-Q 25 A field exciter (FEX-425-Int), external 2-Q 16 A field exciter (DCF803-0016) or external 2-Q 35 A field exciter (DCF803-0035) used for field currents from 0.3 A to **5 A** (terminals X100.2 and X100.3) 21 = **VariFexType** see *DCS800 MultiFex motor control (3ADW000309)* 22 = **Exc-Appl-1** see *DCS800 Series wound motor control (3ADW000311)* If the fex type is changed its new value is taken over after the next power-up. **Int. Scaling: 1 == 1 Type: C Volatile: N**	NotUsed	Exc-Appl-1	OnBoard	-	C
99.13	**Unused**					
99.14	**Unused**					

Square wave generator



Index	Signal / Parameter name	min.	max.	def.	unit	E/C
99.15	**Pot1 (potentiometer 1)** Constant test reference 1 for the manual tuning functions - see *ApplMacro (99.08)* - and the square wave generator. **Note:** The value is depending on the chosen destination of the square wave [e.g. *SqrWaveIndex (99.18)* = 2301 relates to *SpeedScaleAct (2.29)]*: – 100 % voltage == 10,000 – 100 % current == 10,000 – 100 % torque == 10,000 – 100 % speed == *SpeedScaleAct (2.29)* == 20,000 **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
99.16	**Pot2 (potentiometer 2)** Constant test reference 2 for the manual tuning functions - see *ApplMacro (99.08)* - and the square wave generator. **Note:** The value is depending on the chosen destination of the square wave [e.g. *SqrWaveIndex (99.18)* = 2301 relates to *SpeedScaleAct (2.29)]*: – 100 % voltage == 10,000 – 100 % current == 10,000 – 100 % torque == 10,000 – 100 % speed == *SpeedScaleAct (2.29)* == 20,000 **Int. Scaling: 1 == 1 Type: SI Volatile: N**	-32768	32767	0	-	E
99.17	**SqrWavePeriod (square wave period)** The time period for the manual tuning functions - see *ApplMacro (99.08)* - and the square wave generator. **Int. Scaling: 100 == 1 s Type: I Volatile: N**	0.01	655	10	s	E
99.18	**SqrWaveIndex (square wave index)** Index pointer to the source (signal/parameter) for the square wave generator. E.g. signal [e.g. 2301 equals *SpeedRef (23.01)*]. **Note:** *SqrWaveIndex (99.18)* must **not** be used for the manual tuning functions - see *ApplMacro (99.08)*. **Note:** After a power-up *SqrWaveIndex (99.18)* is set back to 0 and thus disables the square wave generator. **Int. Scaling: 1 == 1 Type: I Volatile: Y**	0	9999	0	-	E
99.19	**TestSignal (square wave signal form)** Signal forms for the manual tuning functions - see *ApplMacro (99.08)* - and the square wave generator: 0 = **SquareWave** a square wave is used, default 1 = **Triangle** a triangle wave is used 2 = **SineWave** a sine wave is used 3 = **Pot1** a constant value set with *Pot1 (99.15)* is used **Int. Scaling: 1 == 1 Type: C Volatile: Y**	SquareWave	Pot1	SquareWave	-	E

DCS800 Control Panel operation

Chapter overview

This chapter describes the handling of the DCS800 Control Panel.

Start-up

The commissioning configures the drive and sets parameters that define how the drive operates and communicates. Depending on the control and communication requirements, the commissioning requires any or all of the following:

- The Start-up Assistant (via DCS800 Control Panel or DriveWindow Light) steps you through the default configuration. The DCS800 Control Panel Start-up Assistant runs automatically at the first power up, or can be accessed at any time using the main menu.
- Application macros can be selected to define common, system configurations.
- Additional adjustments can be made using the DCS800 Control Panel to manually select and set individual parameters. See *chapter Signal and parameter list*.

DCS800 Control Panel

Use the DCS800 Control Panel to control the drive, to read status data, to adjust parameters and to use the pre-programmed assistants.

Features:

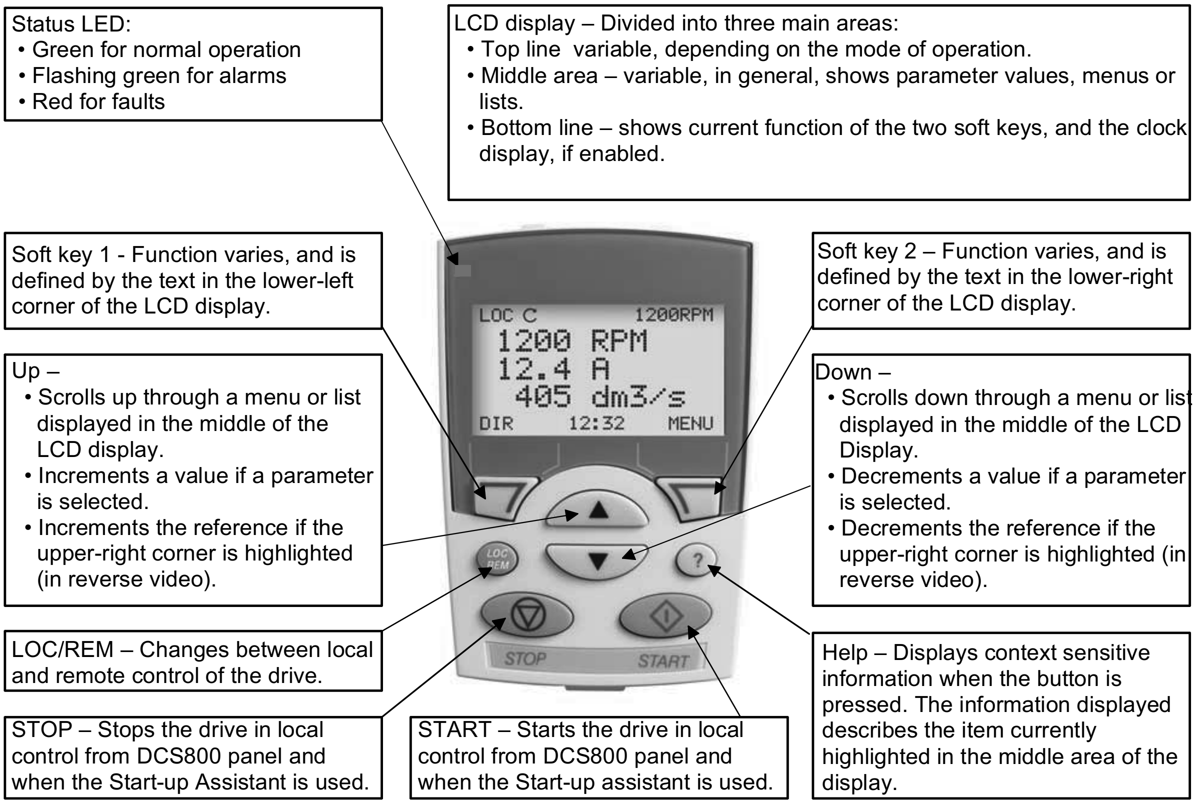
The DCS800 Control Panel features:

- Alphanumeric LCD display
- Language selection for the display by means of *Language (99.01)*
- Panel can be connected or detached at any time
- Start-up Assistant for ease drive commissioning
- Copy function, parameters can be copied into the DCS800 Control Panel memory to be downloaded to other drives or as backup
- Context sensitive help

Fault- and alarm messages including fault history

Display overview

The following table summarizes the button functions and displays of the DCS800 Control Panel.



DCS800 FW pan sum.dsf

General display features

Soft key functions:

The soft key functions are defined by the text displayed just above each key.

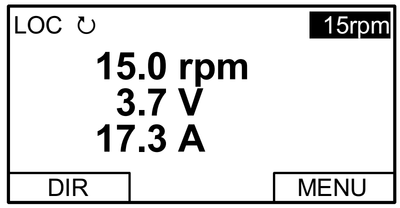
Display contrast:

To adjust display contrast, simultaneously press the MENU key and UP or DOWN, as appropriate.

Output mode

Use the output mode to read information on the drive's status and to operate the drive. To reach the output mode, press EXIT until the LCD display shows status information as described below.

Status information:



Top: The top line of the LCD display shows the basic status information of the drive:

- LOC indicates that the drive control is local from the DCS800 Control Panel.
- REM indicates that the drive control is remote, via local I/O or overriding control.
- indicates the drive and motor rotation status as follows:

DCS800 Control Panel display	Significance
Rotating arrow (clockwise or counter clockwise)	– Drive is running and at setpoint – Shaft direction is forward or reverse
Rotating dotted blinking arrow	Drive is running but not at setpoint
Stationary dotted arrow	Start command is present, but motor is not running. E.g. start enable is missing

- Upper right position shows the active reference, when in local from DCS800 Control Panel.

Middle: Using parameter Group 34, the middle of the LCD display can be configured to display up to three parameter values:

- By default, the display shows three signals.
- Use *DispParam1Sel (34.01)*, *DispParam2Sel (34.08)* and *DispParam3Sel (34.15)* to select signals or parameters to display. Entering value 0 results in no value displayed. For example, if 34.01 = 0 and 34.15 = 0, then only the signal or parameter specified by 34.08 appears on the DCS800 Control Panel display.

Bottom: The bottom of the LCD display shows:

- Lower corners show the functions currently assigned to the two soft keys.
- Lower middle displays the current time (if configured to do so).

Operating the Drive:
LOC/REM: Each time the drive is powered up, it is in remote control (REM) and is controlled as specified in *CommandSel (10.01)*.
To switch to local control (LOC) and control the drive using the DCS800 Control Panel, press the LOC REM button.

– When switching from local control (LOC) to remote control (REM) the drive's status (e.g. **On**, **Run**) and the speed reference of the remote control are taken.

To switch back to remote control (REM) press the LOC REM button.

Start/Stop: To start and stop the drive press the START and STOP buttons.

Shaft direction: To change the shaft direction press DIR.

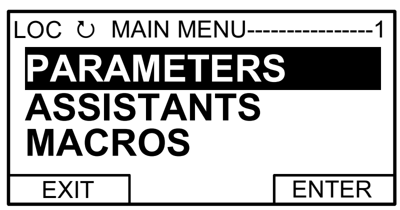
Speed reference: To modify the speed reference (only possible if the display in the upper right corner is highlighted) press the UP or DOWN button (the reference changes immediately).
The speed reference can be modified via the DCS800 Control Panel when in local control (LOC).

Note:
The START / STOP buttons, shaft direction (DIR) and reference functions are only valid in local control (LOC).

Other modes

Below the output mode, the DCS800 Control Panel has:

- Other operating modes are available through the MAIN MENU.
- A fault mode that is triggered by faults. The fault mode includes a diagnostic assistant mode.
- An alarm mode that is triggered by drive alarms.



Access to the MAIN MENU and other modes:
To reach the MAIN MENU:

1. Press EXIT, as necessary, to step back through the menus or lists associated with a particular mode. Continue until you are back to the output mode.
2. Press MENU from the output mode. At this point, the middle of the display is a listing of the other modes, and the top-right text says “MAIN MENU”.
3. Press UP/DOWN to scroll to the desired mode.
4. Press ENTER to enter the mode that is highlighted.

Following modes are available in the MAIN MENU:

1. Parameters mode
2. Start-up assistants mode
3. Macros mode (currently not used)
4. Changed parameters mode

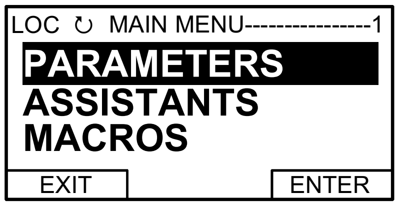
5. Fault logger mode
6. Clock set mode
7. Parameter backup mode
8. I/O settings mode (currently not used)

The following sections describe each of the other modes.

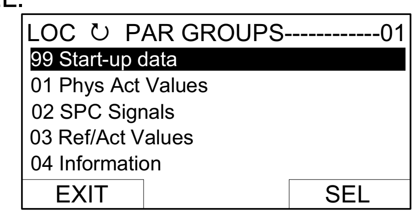
Parameters mode:

Use the parameters mode to view and edit parameter values:

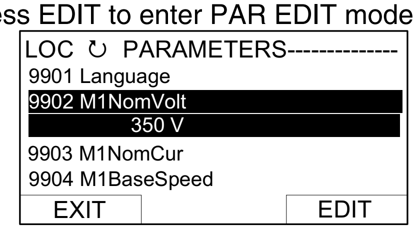
1. Press UP/DOWN to highlight PARAMETERS in the MAIN MENU, then press ENTER.



2. Press UP/DOWN to highlight the appropriate parameter group, then press SEL.

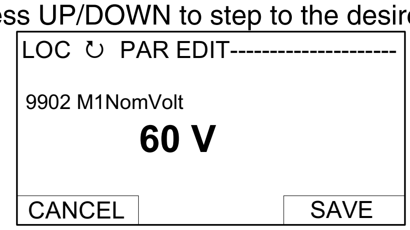


3. Press UP/DOWN to highlight the appropriate parameter in a group, then press EDIT to enter PAR EDIT mode.



Note:
The current parameter value appears below the highlighted parameter.

4. Press UP/DOWN to step to the desired parameter value.



Note:
To get the parameter default value press UP/DOWN simultaneously.

5. Press SAVE to store the modified value and leave the PAR EDIT mode or press CANCEL to leave the PAR EDIT mode without modifications.
6. Press EXIT to return to the listing of parameter groups, and again to step back to the MAIN MENU.

Start-up assistants mode:

Use the start-up assistants mode for basic commissioning of the drive.
When the drive is powered up the first time, the start-up assistants guides you through the setup of the basic parameters.
There are seven start-up assistants available. They can be activated one after the other, as the ASSISTANTS menu suggests, or independently. The use of the assistants is not required. It is also possible to use the parameter mode instead.
The assistant list in the following table is typical:

1. Name plate data	– Enter the motor data, the mains (supply) data, the most important protections and follow the instructions of the assistant. – After filling out the parameters of this assistant it is - in most cases - possible to turn the motor for the first time.
2. Macro assistant	– Selects an application macro.
3. Autotuning field current controller	– Enter the field circuit data and follow the instructions of the assistant. – During the autotuning the main respectively field contactor will be closed, the field circuit is measured by means of increasing the field current to nominal field current and the field current control parameters are set. The armature current is not released while the autotuning is active and thus the motor should not turn. – When the autotuning is finished successfully the parameters changed by the assistant are shown for confirmation. If the assistant fails it is possible to enter the fault mode for more help.
4. Autotuning armature current controller	– Enter the motor nominal current, the basic current limitations and follow the instructions of the assistant. – During the autotuning the main contactor will be closed, the armature circuit is measured by means of armature current bursts and the armature current control parameters are set. The field current is not released while the autotuning is active and thus the motor should not turn, but due to remanence in the field circuit about 40% of all motors will turn (create torque). These motors have to be locked. – When the autotuning is finished successfully the parameters changed by the assistant are shown for confirmation. If the assistant fails it is possible to enter the fault mode for more help.
5. Speed feedback assistant	– Enter the EMF speed feedback parameters, - if applicable - the parameters for the pulse encoder respectively the analog tacho and follow the instructions of the assistant. – The speed feedback assistant detects the kind of speed feedback the drive is using and provides help to set up pulse encoders or analog tachometers. – During the autotuning the main contactor and the field contactor - if existing - will be closed and the motor will run up to base speed [*M1BaseSpeed (99.04)*]. During the whole procedure the drive will be in EMF speed control despite the setting of *M1SpeedFbSel (50.03)*. – When the assistant is finished successfully the speed feedback is set. If the assistant fails it is possible to enter the fault mode for more help.
6. Autotuning speed controller	– Enter the motor base speed, the basic speed limitations, the speed filter time and follow the instructions of the assistant. – During the autotuning the main contactor and the field contactor - if existing - will be closed, the ramp is bypassed and torque respectively current limits are valid. The speed controller is tuned by means of speed bursts up to base speed [*M1BaseSpeed (99.04)*] and the speed controller parameters are set. **Attention:** During the autotuning the torque limits will be reached. – When the autotuning is finished successfully the parameters changed by the assistant are shown for confirmation. If the assistant fails it is possible to enter the fault mode for more help.

	Attention: This assistant is using the setting of *M1SpeedFbSel (50.03)*. If using setting **Encoder**, **Encoder2** or **Tacho** make sure the speed feedback is working properly!
7. Field weakening assistant (only used when maximum speed is higher than base speed)	– Enter the motor data, the field circuit data and follow the instructions of the assistant. – During the autotuning the main contactor and the field contactor - if existing - will be closed and the motor will run up to base speed [*M1BaseSpeed (99.04)*]. The EMF controller data are calculated, the flux linearization is tuned by means of a constant speed while decreasing the field current and the EMF controller respectively flux linearization parameters are set. – When the autotuning is finished successfully the parameters changed by the assistant are shown for confirmation. If the assistant fails it is possible to enter the fault mode for more help.

1. Press UP/DOWN to highlight ASSISTANTS in the MAIN MENU, then press ENTER.
2. Press UP/DOWN to highlight the appropriate start-up assistant, then press SEL to enter PAR EDIT mode.
3. Make entries or selections as appropriate.
4. Press SAVE to save settings. Each individual parameter setting is valid immediately after pressing SAVE.
5. Press EXIT to step back to the MAIN MENU.

Macros mode:
Currently not used!

Changed parameters mode:
Use the changed parameters mode to view and edit a listing of all parameter that have been changed from their default values:

1. Press UP/DOWN to highlight CHANGED PAR in the MAIN MENU, then press ENTER.
2. Press UP/DOWN to highlight a changed parameter, then press EDIT to enter PAR EDIT mode.

Note:
The current parameter value appears below the highlighted parameter.

3. Press UP/DOWN to step to the desired parameter value.

Note:
To get the parameter default value press UP/DOWN simultaneously.

4. Press SAVE to store the modified value and leave the PAR EDIT mode or press CANCEL to leave the PAR EDIT mode without modifications.

Note:
If the new value is the default value, the parameter will no longer appear in the changed parameter list.

5. Press EXIT to step back to the MAIN MENU.

Fault logger mode:
Use the fault logger mode to see the drives fault, alarm and event history, the fault state details and help for the faults:

1. Press UP/DOWN to highlight FAULT LOGGER in the MAIN MENU, then press ENTER to see the latest faults (up to 20 faults, alarms and events are logged).
2. Press DETAIL to see details for the selected fault. Details are available for the three latest faults, independent of the location in the fault logger.
3. Press DIAG to get additional help (only for faults).
4. Press EXIT to step back to the MAIN MENU.

Clock set mode:
Use the Clock set mode to:
- Enable or disable the clock function.
- Select the display format.
- Set date and time.

1. Press UP/DOWN to highlight CLOCK SET in the MAIN MENU, then press ENTER.
2. Press UP/DOWN to highlight the desired option, then press SEL.
3. Choose the desired setting, then press SEL or OK to store the setting or press CANCEL to leave without modifications.
4. Press EXIT to step back to the MAIN MENU.

Note:
To get the clock visible on the LCD display at least one change has to be done in the clock set mode and the DCS800 Control Panel has to be de-energized and energized again.

Parameter backup mode:
The DCS800 Control Panel can store a full set of drive parameters.
- AP will be uploaded and downloaded.
- The type code of the drive is write protected and has to be set manually by means of *ServiceMode (99.06)* = **SetTypeCode** and *TypeCode (97.01)*.

The parameter backup mode has following functions:

UPLOAD TO PANEL: Copies all parameters from the drive into the DCS800 Control Panel. This includes both user sets (**User1** and **User2**) - if defined - and internal parameters such as those created by tacho fine tuning. The DCS800 Control Panel memory is non-volatile and does not depend on its battery. Can only be done in drive state **Off** and **local** from DCS800 Control Panel.

DOWNLOAD FULL SET: Restores the full parameter set from the DCS800 Control Panel into the drive. Use this option to restore a drive, or to configure identical drives. Can only be done in drive state **Off** and **local** from DCS800 Control Panel.

Note:
This download does not include the user sets.

DOWNLOAD APPLICATION: Currently not used!

The general procedure for parameter backup operations is:
1. Press UP/DOWN to highlight PAR BACKUP in the MAIN MENU, then press ENTER.

2. Press UP/DOWN to highlight the desired option, then press SEL.
3. Wait until the service is finished, then press OK.
4. Press EXIT to step back to the MAIN MENU.

I/O settings mode:
Currently not used!

Maintenance

Cleaning:
Use a soft damp cloth to clean the DCS800 Control Panel. Avoid harsh cleaners which could scratch the display window.

Battery:
A battery is used in the DCS800 Control Panel to keep the clock function available and enabled. The battery keeps the clock operating during power interruptions. The expected life for the battery is greater than ten years. To remove the battery, use a coin to rotate the battery holder on the back of the control panel. The type of the battery is CR2032.

Note:
The battery is **not** required for any DCS800 Control Panel or drive functions, except for the clock.

Fault tracing

Chapter overview

This chapter describes the protections and fault tracing of the drive.

General

Fault modes

Depending on the trip level of the fault the drive reacts differently. The drive's reaction to a fault with trip level 1 and 2 is fixed. See also paragraph <u>*Fault signals*</u> of this manual. The reaction to a fault of level 3 and 4 can be chosen by means of *SpeedFbFltMode (30.36)* respectively *FaultStopMode (30.30)*.

Converter protection

Auxiliary undervoltage

If the auxiliary supply voltage fails while the drive is in **RdyRun** state (MSW bit 1), fault **F501 AuxUnderVolt** is generated.

Auxiliary supply voltage	Trip level
230 VAC	< 185 VAC
115 VAC	< 96 VAC

Armature overcurrent

The nominal value of the armature current is set with *M1NomCur (99.02)*.

The overcurrent level is set by means of *ArmOvrCurLev (30.09)*.
Additionally the actual current is monitored against the overcurrent level of the converter module. The converter's actual overcurrent level can be read from *ConvOvrCur (4.16)*.

Exceeding one of the two levels causes **F502 ArmOverCur**.

Converter overtemperature

The maximum temperature of the bridge can be read from *MaxBridgeTemp (4.17)* and is automatically set by *TypeCode (97.01)* or manually set by *S MaxBrdgTemp (97.04)*.
Note:
When setting the air entry temperature for D6 and D7 modules manually use *MaxBrdgTemp (97.04)* = 50 °C as absolute maximum.

Exceeding this level causes **F504 ConvOverTemp**. The threshold for **A104 ConvOverTemp** is 5 °C below the tripping level. The measured temperature can be read from *BridgeTemp (1.24)*.

If the measured temperature drops below minus 10 °C, **F504 ConvOverTemp** is generated.

Auto-reclosing (mains undervoltage)

Auto-reclosing allows continuing drive operation immediately after a short mains undervoltage without any additional functions in the overriding control system.

In order to keep the overriding control system and the drive control electronics running through short mains undervoltage, an UPS is needed for the 115/230 VAC auxiliary voltages. Without the UPS all DI like e.g. E-stop, start inhibition, acknowledge signals etc. would have false states and trip the drive although the system itself could stay alive. Also the control circuits of the main contactor must be supplied during the mains undervoltage.

Auto-reclosing defines whether the drive trips immediately with **F512 MainsLowVolt** or if the drive will continue running after the mains voltage returns. To activate the auto-reclosing set *PwrLossTrip (30.21)* = **Delayed**.

Short mains undervoltage

The supervision of mains undervoltage has two levels:

1. *UNetMin1 (30.22)* alarm, protection and trip level
2. *UNetMin2 (30.23)* trip level

If the mains voltage falls below *UNetMin1 (30.22)* but stays above *UNetMin2 (30.23)*, the following actions take place:

1. the firing angle is set to *ArmAlphaMax (20.14)*,
2. single firing pulses are applied in order to extinguish the current as fast as possible,
3. the controllers are frozen,
4. the speed ramp output is updated from the measured speed and
5. **A111 MainsLowVolt** is set as long as the mains voltage recovers before *PowrDownTime (30.24)* is elapsed, otherwise **F512 MainsLowVolt** is generated.

If the mains voltage returns before *PowrDownTime (30.24)* is elapsed and the overriding control keeps the commands **On** (MCW bit 0) and **Run** (MCW bit 3) = 1, the drive will start again after 2 seconds. Otherwise the drive trips with **F512 MainsLowVolt**.

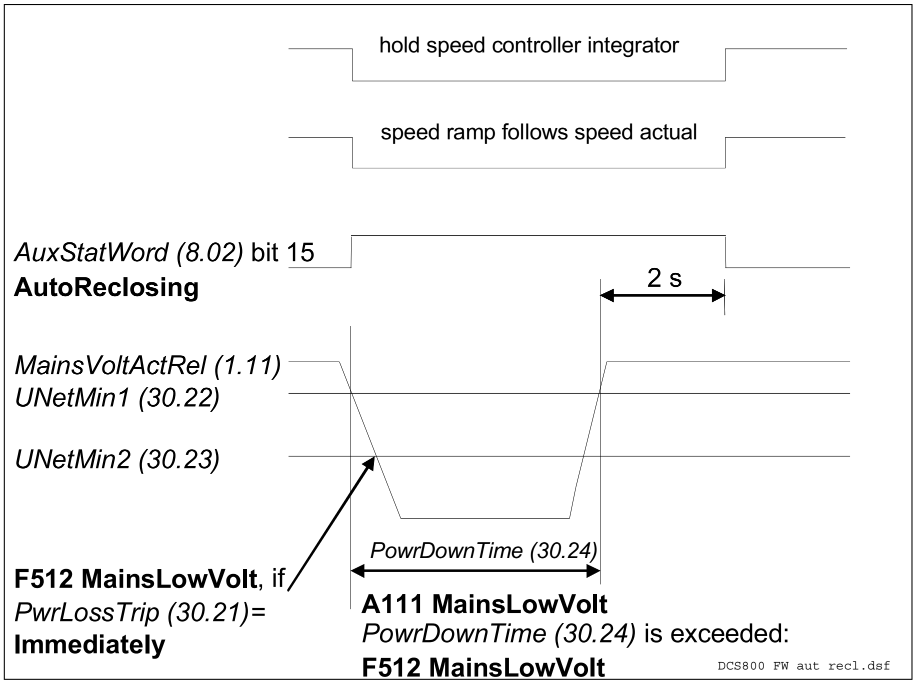
When the mains voltage drops below *UNetMin2 (30.23)*, the action is selected by means of *PwrLossTrip (30.21)*:

1. the drive is immediately tripped with **F512 MainsLowVolt** or
2. the drive starts up automatically, see description for *UNetMin1 (30.22)*. Below *UNetMin2 (30.23)* the field acknowledge signals are ignored and blocked

Note:
UNetMin2 (30.23) isn't monitored, unless the mains voltage drops below *UNetMin1 (30.22)*. Thus, for proper operation, *UNetMin1 (30.22)* must be larger than *UNetMin2 (30.23)*.

Note:
If no UPS is available, set *PwrLossTrip (30.21)* to **Immediately**. Thus the drive will trip with **F512 MainsLowVolt** avoiding secondary phenomena due to missing power for AI’s and DI’s.

Drive behavior during auto-reclosing



Auto-reclosing

Mains synchronism

As soon as the main contactor is closed and the firing unit is synchronized with the incoming voltage, supervising of the synchronization is activated. If the synchronization fails, **F514 MainsNotSync** will be generated.

The synchronization of the firing unit takes typically 300 ms before the current controller is ready.

Mains overvoltage

The overvoltage level is fixed to 1.3 * *NomMainsVolt (99.10)*. Exceeding this level for more than 10 s and RdyRun = 1 causes **F513 MainsOvrVolt**.

Communication loss

The communication to several devices is supervised. The reaction to a communication loss can be chosen by means of *LocalLossCtrl (30.27)* or *ComLossCtrl (30.28)*.

The time out is set by the parameters listed in the table as well as all dependent fault- and alarm messages.

Overview local and communication loss:				
Device	**Loss control**	**Time out**	**Related fault**	**Related alarm**
DCS800 Control Panel	*LocalLossCtrl (30.27)*	fixed to 5s	**F546 LocalCmdLoss**	**A130 LocalCmdLoss**
DW				
DWL				
R-type fieldbus	*ComLossCtrl (30.28)*	*FB TimeOut (30.35)*	**F528 FieldBusCom**	**A128 FieldBusCom**
DCSLink		*MailBoxCycle1 (94.13), MailBoxCycle2 (94.19), MailBoxCycle3 (94.25), MailBoxCycle4 (94.31)*	**F544 P2PandMFCom**	**A112 P2PandMFCom**
	-	*12P TimeOut (94.03)*	**F535 12PulseCom**	-
	-	*FexTimeOut (94.07)*	**F516 M1FexCom** **F519 M2FexCom**	-
SDCS-COM-8	*Ch0 ComLossCtrl (70.05)*	*Ch0 TimeOut (70.04)*	**F543 COM8Com**	**A113 COM8Com**
	Ch2 ComLossCtrl (70.15)	*Ch2 TimeOut (70.14)*		

Overview local and communication loss

Fan, field and mains contactor acknowledge

When the drive is switched **On** (MCW bit 0), the firmware closes the fan contactor and waits for acknowledge. After it is received, the field contactor is closed respectively the field converter is started and the firmware waits for the field acknowledge. Finally the main contactor is closed and its acknowledge is waited for.

If the acknowledges are not received during 10 seconds after the **On** command (MCW bit 0) is given, the corresponding fault is generated. These are:

1. **F521 FieldAck**, see *Mot1FexStatus (6.12)*
2. **F523 ExtFanAck**, see *MotFanAck (10.06)*
3. **F524 MainContAck**, see *MainContAck (10.21)*
4. **F527 ConvFanAck**, see *ConvFanAck (10.20)*

Note:
F521 FieldAck is the sum fault for all field related faults like:

1. **F515 M1FexOverCur**, see *M1FldOvrCurLev (30.13)*
2. **F516 M1FexCom**, see *FexTimeOut (94.07)*
3. **F529 M1FexNotOK**, fault during self-diagnosis
4. **F537 M1FexRdyLost**, AC voltage is missing or not in synchronism
5. **F541 M1FexLowCur**, see *M1FldMinTrip (30.12)*

External fault

The user has the possibility to connect external faults to the drive. The source can be connected to DI's, *MainCtrlWord (7.01)* or *AuxCtrlWord (7.02)* and is selectable by *ExtFaultSel (30.31)*. External faults generate **F526 ExternalDI**.

ExtFaultOnSel (30.33) selects the reaction:

1. external fault is always valid independent from drive state
2. external fault is only valid when drive state is **RdyRun** (MSW bit 1) for at least 6 s

Note:
In case inverted fault inputs are needed, it is possible to invert the DI's.

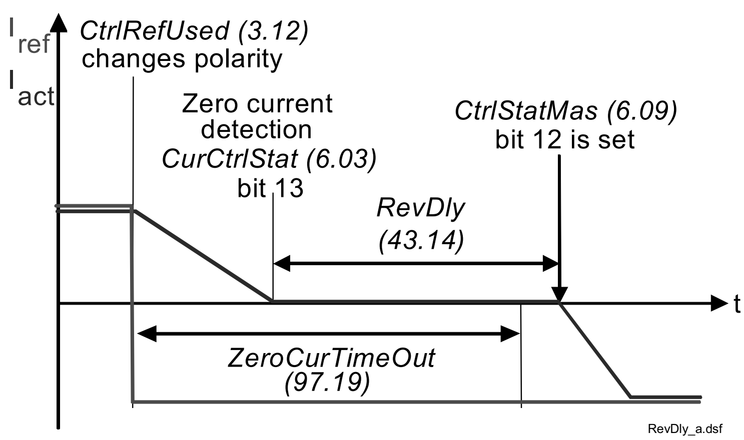
Bridge reversal

With a 6-pulse converter, the bridge reversal is initiated by changing the polarity of the current reference - see *CurRefUsed (3.12)*. Upon zero current detection - see *CurCtrlStat1 (6.03)* bit 13 - the bridge reversal is started. Depending on the moment, the new bridge may be "fired" either during the same or during the next current cycle.

The switchover can be delayed by *RevDly (43.14)*. The delay starts after zero current has been detected - see *CurCtrlStat1 (6.03)* bit 13. Thus *RevDly (43.14)* is the length of the forced current gap during a bridge changeover. After the reversal delay is elapsed the system changes to the selected bridge without any further consideration.

This feature may prove useful when operating with large inductances. Also the time needed to change the current direction can be longer when changing from motoring mode to regenerative mode at high motor voltages, because the motor voltage must be reduced before switching to regenerative mode - see also *RevVoltMargin (44.21)*.

After a command to change current direction - see *CurRefUsed (3.12)* - the opposite current has to be reached before *ZeroCurTimeOut (97.19)* has been elapsed otherwise the drive trips with **F557 ReversalTime** [*FaultWord4 (9.04)* bit 8].
Example:
Drive is tripping with **F557 ReversalTime** [*FaultWord4 (9.04)* bit 8]:



Bridge reversal

Analog input monitor

In case the analog input is set to 2 V to 10 V respectively 4 mA to 20 mA it is possible to check for wire breakage by means of *AI Mon4mA (30.29)*.

In case the threshold is undershoot one of the following actions will take place:

1. the drive stops according to *FaultStopMode (30.30)* and trips with **F551 AIRange**
2. the drive continues to run at the last speed and sets **A127 AIRange**

3. the drive continues to run with *FixedSpeed1 (23.02)* and sets **A127 AIRange**

Motor protection

Armature overvoltage

The nominal value of the armature voltage is set with *M1NomVolt (99.02)*.

The overvoltage level is set by means of *ArmOvrVoltLev (30.08)*. Exceeding this level causes **F503 ArmOverVolt**.

Residual current detection

The residual current detection (earth fault) is based on:
- a sum current transformer at the AC-side of the converter or
- an external device (e.g. Bender relays).

If a current transformer (ratio is 400 : 1) is used its secondary winding is connected to AI4 (X3:11 and X3:12) on the SDCS-IOB-3 board. The sum current of all three phases has to be zero, otherwise a residual current is detected and **F505 ResCurDetect** is set.

ResCurDetectSel (30.05) activates the residual current detection and selects the choice of connected hardware (transformer or external device).

The residual current detection tripping level, in amperes at the primary side of the current transformer, is set with *ResCurDetectLim (30.06)*, if a sum current transformer is used. In case an external device is used *ResCurDetectLim (30.06)* is deactivated.

ResCurDetectDel (30.07) delays **F505 ResCurDetect**.

Measured motor temperature

General

The temperatures of motor 1 and motor 2 (parameter for motor 2 see group 49) can be measured at the same time. Alarm and tripping levels are selected by means of *M1AlarmLimTemp (31.06)* and *M1FaultLimTemp (31.07)*. If the levels are exceeded **A106 M1OverTemp** respectively **F506 M1OverTemp** is set. The motor fan will continue to work until the motor is cooled down to alarm limit.

The measurement is configured by means of *M1TempSel (31.05)* and the measured temperature is shown in *Mot1TempMeas (1.22)*. The unit of the measurement depends on the selected measurement mode. For PT100 the unit is degree Celsius and for PTC the unit is Ω.

The motor temperature measurement uses either AI2 and AI3 of the SDCS-IOB-3 or AI7 and AI8 of the RAIO. Additionally the SDCS-IOB-3 features a selectable constant current source for PT100 (5 mA) or PTC (1.5 mA).

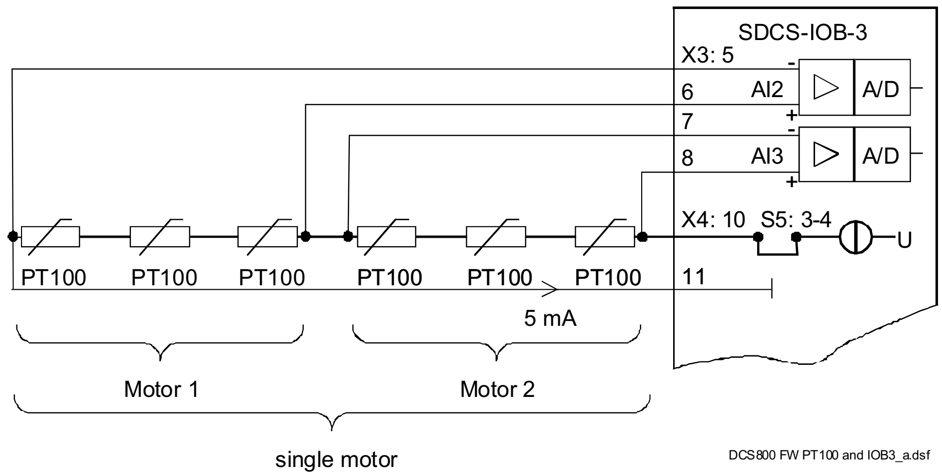
Measurement selection

Connection possibilities for PT100:

- max. 3 PT100 for motor 1 and max. 3 PT100 for motor 2 or
- up to 6 PT100 for a single motor.

SDCS-IOB-3:

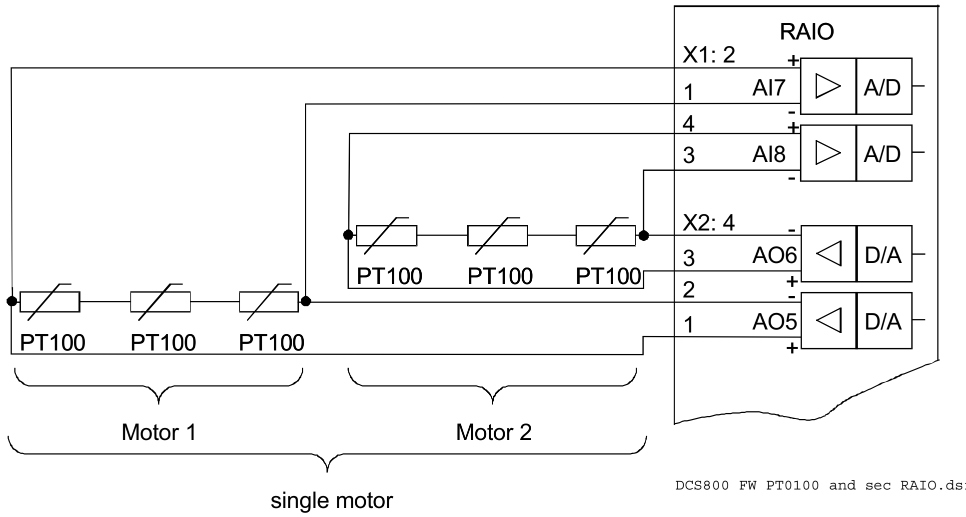
AI2 (motor 1) and AI3 (motor 2) are used for the temperature measurement with PT100. In case only one PT100 is connected to an AI the input range must be configured by jumpers to a gain of 10. Jumper settings for input range and constant current source see *DCS800 Hardware Manual*. All parameters for AI2 and AI3 in group 15 have to set to default.



PT100 and SDCS-IOB-3

For more information see *section* Analog Inputs.

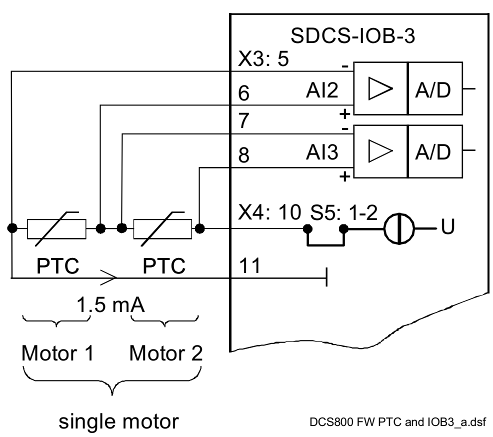
RAIO for motor temperature measurement:
AI7 (motor 1) and AI8 (motor 2) are used for the temperature measurement with PT100. AO5 and AO6 are used as current source. AI7 / AO5 and AI8 / AO6 have to be activated by means of *AIO MotTempMeas (98.12)*.



PT100 and second RAIO

SDCS-IOB-3:
Connection possibilities for PTC:
- max. 1 PTC for motor 1 and max. 1 PTC for motor 2 or
- up to 2 PTC for a single motor.

AI2 (motor 1) and AI3 (motor 2) are used for the temperature measurement with PTC. Jumper settings see *DCS800 Hardware Manual*. All parameters for AI2 and AI3 in group 15 have to set to default.

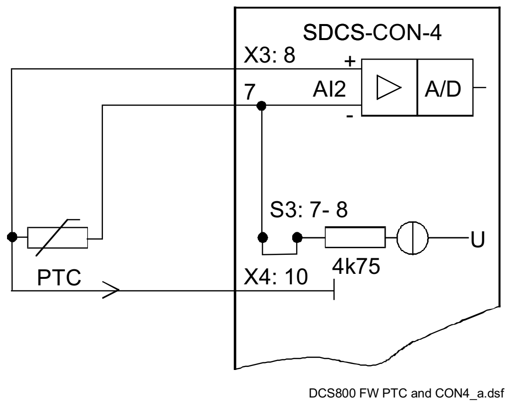


PTC and SDCS-IOB-3

SDCS-CON-4:
Connection possibilities for PTC:

- max. 1 PTC for motor 1 or max. 1 PTC for motor 2.

Only AI2 can be used for the temperature measurement with PTC. Jumper settings see *DCS800 Hardware Manual*. All parameters for AI2 in group 15 have to set to default.



PTC and SDCS-CON-4

Klixon

The temperature of motor 1 and motor 2 can be supervised by means of klixons. The klixon is a thermal switch, opening its contact at a defined temperature. This can be used for supervision of the temperature by means of connecting the switch to a digital input of the drive. The digital input for the klixon(s) is selected with *M1KlixonSel (31.08)*. The drive trips with **F506 M1OverTemp** when the klixon opens. The motor fan will continue to work until the klixon is closed again.

Note:
It is possible to connect several klixons in series.

Motor thermal model

General

The drive includes two thermal models one for motor 1 and one for motor 2. The models can be used at the same time. Two models are needed in case one converter is shared by two motors (e.g. shared motion). During normal operation only one thermal model is needed.

It is recommended to use the thermal model of the motor if a direct motor temperature measurement isn't available and the current limits of the drive are set higher than the motor nominal current.

The thermal model is based on the actual motor current related to motor nominal current and rated ambient temperature. Thus the thermal model does not directly calculate the temperature of the motor, but it calculates the ***temperature rise*** of the motor. This is based on the fact that the motor will reach its end temperature after the specified time when starting to run the cold motor (40°C) with nominal current. This time is about four times the motor thermal time constant.

The temperature rise of the motor behaves like the time constant which is proportional with the motor current to the power of two:

$$\Phi = \frac{I_{act}^2}{I_{Motn}^2} * \left(1 - e^{-\frac{t}{\tau}}\right) \quad (1)$$

When the motor is cooling down, following temperature model is valid:

$$\Phi = \frac{I_{act}^2}{I_{Motn}^2} * e^{-\frac{t}{\tau}} \quad (2)$$

with: Φ_{alarm} = temperature rise == [*M1AlarmLimLoad (31.03)*]^2
Φ_{trip} = temperature rise == [*M1FaultLimLoad (31.04)*]^2
Φ = temperature rise == *Mot1TempCalc (1.20)*
I_{act} = actual motor current (overload e.g. 170%)
I_{MotN} = nominal motor current (100%)
t = length of overload (e.g. 60 s)
τ = temperature time constant (in seconds) == *M1ModelTime (31.01)*

As from the formulas (1) and (2) can be seen, the temperature model uses the same time constant when the motor is heating or cooling down.

Alarm and tripping levels

Alarm and tripping levels are selected by means of *M1AlarmLimLoad (31.03)* and *M1FaultLimLoad (31.04)*. If the levels are exceeded **A107 M1OverLoad** respectively **F507 M1OverLoad** is set. The motor fan will continue to work until the motor is cooled down under the alarm limit.

The default values are selected in order to achieve quite high overload ability. Recommended value for alarming is 102 % and for tripping 106 % of nominal motor current. Thus the temperature rise is:

- Φ_{alarm} == [*M1AlarmLimLoad (31.03)*]^2 = (102%)^2 = 1.02^2 = 1.04 and
- Φ_{trip} == [*M1FaultLimLoad (31.04)*]^2= (106%)^2 = 1.06^2 = 1.12.

The temperature rise output of the model is shown in *Mot1TempCalc (1.20)*.

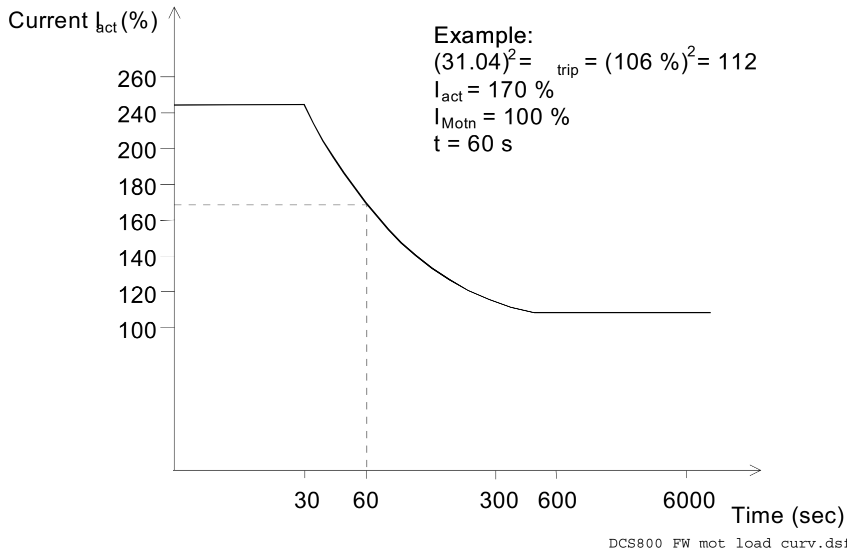
Thermal model selection

The activation of the thermal models is made by setting *M1ModelTime (31.01)* greater than zero.

Thermal time constant

The time constant for the thermal model is set by means of *M1ModelTime (31.01)*. If the thermal time constant of a motor is given by the manufacturer just write it into *M1ModelTime (31.01)*.
In many cases the motor manufacturer provides a curve that defines how long the motor can be overloaded by a certain overload factor. In this case the proper thermal time constant must be calculated.

Example:
The drive is desired to trip if the motor current exceeds 170 % of motor nominal current for more than 60 seconds.
Selected tripping base level is 106 % of nominal motor current, thus *M1FaultLimLoad (31.04)* = 106 %.



Motor load curve

Note:
This is an example and does not necessarily correspond to any motor!

Using formula (1) we can calculate the correct value for τ, when starting with a cold motor.
With:

$$(31.04)^2 = \Phi_{trip} = \frac{I_{act}^2}{I_{Motn}^2} * \left(1 - e^{-\frac{t}{\tau}}\right)$$

Follows:

$$\tau = -\frac{t}{\ln\left(1-(31.04)^2 * \frac{I_{Motn}{}^2}{I_{act}{}^2}\right)} = -\frac{60s}{\ln\left(1-1.06^2 * \frac{1.0^2}{1.7^2}\right)} = 122s$$

Set *M1ModelTime (31.01)* = 122 s.

Field overcurrent

The nominal value of the field current is set with *M1NomFldCur (99.11)*.

The overcurrent level is set by means of *M1FldOvrCurLev (30.13)*. Exceeding this level causes **F515 M1FexOverCur**.

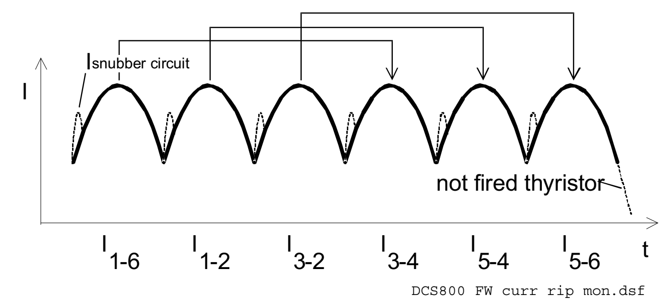
Armature current ripple

The current control is equipped with a current ripple monitor. This function can detect:

1. a broken fuse or thyristor
2. too high gain (e.g. wrong tuning) of the current controller
3. a broken current transformer (T51, T52)

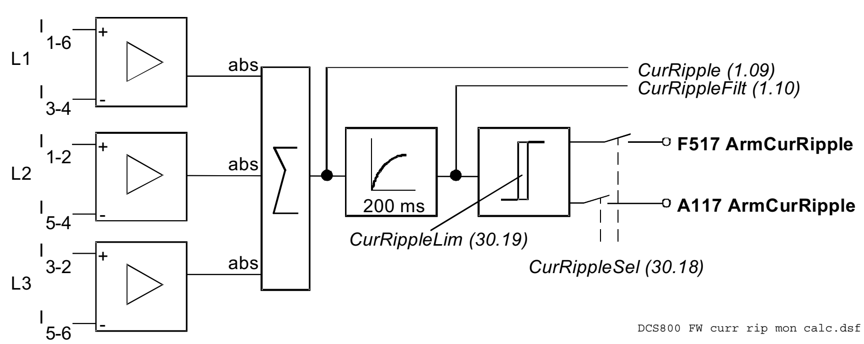
The current ripple monitor level is set by means of *CurRippleLim (30.19)*. Exceeding this level causes either **F517 ArmCurRipple** or **A117 ArmCurRipple** depending on *CurRippleSel (30.18)*.

Current ripple monitor method is based on comparing positive and negative currents of each phase. The calculation is done per thyristor pair:



Current ripple monitor method

CurRipple (1.09) is calculated as abs($I_{1\text{-}6}$-$I_{3\text{-}4}$) + abs($I_{1\text{-}2}$-$I_{5\text{-}4}$) + abs($I_{3\text{-}2}$-$I_{5\text{-}6}$). By low-pass filtering with 200 ms *CurRippleFilt (1.10)* is generated and compared against *CurRippleLim (30.19)*.



Current ripple monitor calculation

Note:

The load influences the error signal *CurRippleFilt (1.10)*.

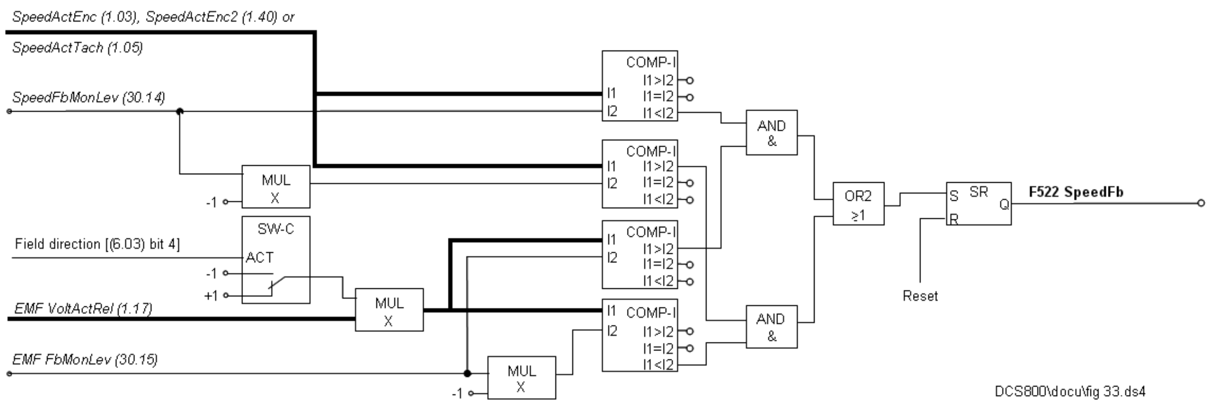
Current near discontinuous level will create values of about 300 % *

ConvCurActRel (1.15) if a thyristor is not fired.
High inductive loads will create values of about 90% * *ConvCurActRel (1.15)* if a thyristor is not fired.

Commissioning hint:
It is not possible to pre-calculate clear levels.
The current control reacts to unstable current feedback.
The load is continuously driving the current if a thyristor is not fired.

Speed feedback monitor

The speed feedback monitor supervises an attached analog tacho or encoder for proper function by means of measured speed and measured EMF. Above a certain EMF the measured speed feedback must be above a certain threshold. The sign of the speed measurement must be correct as well:



Speed measurement supervision

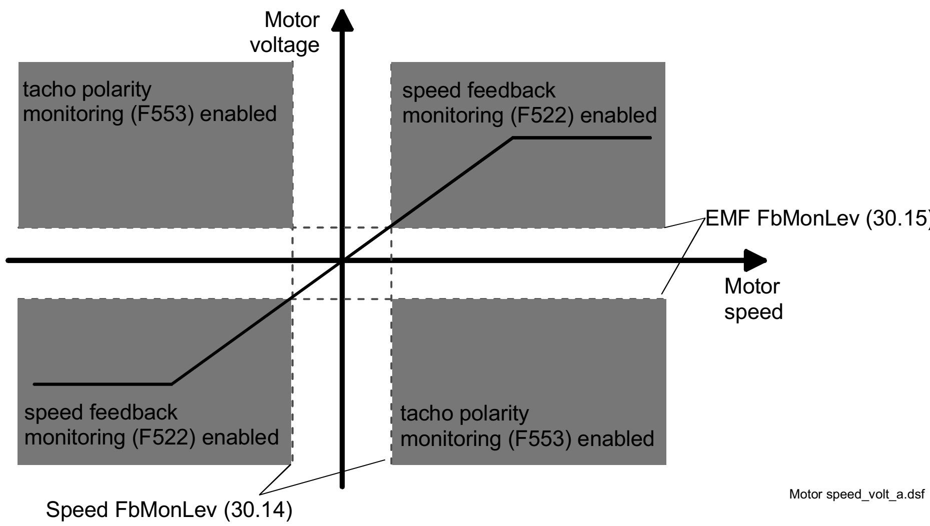
The drive reacts according to *SpeedFbFltSel (30.17)* when:

1. the measured EMF is greater than *EMF FbMonLev (30.15)* and
2. the measured speed feedback *SpeedActEnc (1.03), SpeedActTach (1.05)* or *SpeedActEnc2 (1.42)* is lower than *SpeedFbMonLev (30.14)*.

Example:

– *SpeedFbMonLev (30.14)* = 15 rpm
– *EMF FbMonLev (30.15)* = 50 V

The drive trips when the EMF is greater than 50 V while the speed feedback is ≤ 15 rpm.



Speed feedback monitor

SpeedFbFltSel (30.17) selects the reaction to a speed feedback problem:

1. the drive is immediately tripped with **F522 SpeedFb**
2. the speed feedback is switched to EMF and the drive is stopped according to *E StopRamp (22.11),* then **F522 SpeedFb** is set
3. the speed feedback is switched to EMF and **A125 SpeedFb** is set
4. This selection is only valid if 2 pulse encoders are connected. Depending on the setting of *M1SpeeFbSel (50.03)* the speed feedback is switched from pulse encoder 1 to pulse encoder 2 or vice versa in case of a problem and **A125 SpeedFb** [*AlarmWord2 (9.07)* bit 8] is set.

In case the field is weakened the drive is immediately tripped with **F522 SpeedFb**, except two pulse encoders are in use.

Stall protection

The stall protection trips the converter with **F531 MotorStalled** when the motor is in apparent danger of overheating. The rotor is either mechanically stalled or the load is continuously too high. It is possible to adjust the supervision (time, speed and torque).

The stall protection trips the drive if:

1. the actual speed is below *StallSpeed (30.02)* and
2. the actual torque *- in percent of MotNomTorque (4.23)* - exceeds *StallTorq (30.03)*
3. for a time longer than programmed in *StallTime (30.01).*

Overspeed protection

The motor is protected against overspeed e.g. in a case when the drive is in torque control mode and the load drops unexpected.
The overspeed level is set by means of *M1OvrSpeed (30.16)*. Exceeding this level causes **F532 MotOverSpeed**.

Current rise

The protection against fast current rise during generating is configured by means of *ArmCurRiseMax (30.10)*.
Exceeding this level causes **F539 FastCurRise**. If present the DC-breaker is tripped and the main contactor is opened.

Field undercurrent

The nominal value of the field current is set with *M1NomFldCur (99.11)*.
The minimum field current level is set by means of *M1FldMinTrip (30.12)*. Undershooting this level causes **F541 M1FexLowCur**.
FldMinTripDly (45.18) delays **F541 M1FexLowCur**.

Tacho / pulse encoder polarity

The polarity of the analog tacho or pulse encoder [depending on *M1SpeedFbSell (50.03)*] is checked against the EMF. If the polarity is wrong **F553 TachPolarity** is generated.

Tacho range

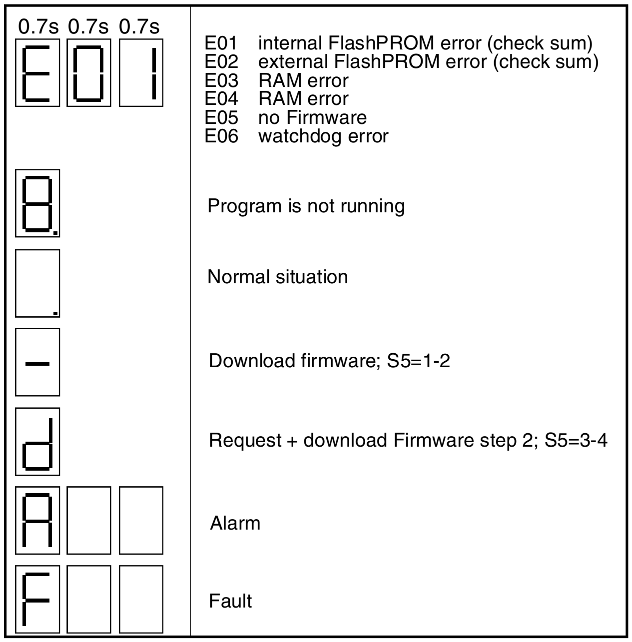
If an overflow of the AITacho input is imminent **F554 TachoRange** is generated. Check for the right connections (X3:1 to X3:4) on the SDCS-CON-4.

Status messages

Display of status, fault and alarm signals

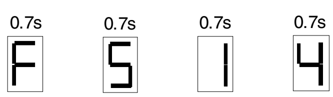
Categories of signals and display options

A seven segment display (H2500) is located on the control board SDCS-CON-4 and it shows the state of drive:



7seg_DCS8_a.dsf

The seven-segment display shows the messages in code. The letters and numbers of multi-character codes are displayed one after the other for 0.7 seconds at a time. Plain text messages are available on the DCS800 Control Panel and in the fault logger of DriveWindow and DriveWindow Light.



F514 = mains not in synchronism

For evaluation via digital outputs or communication to the overriding control 16 bit words are available, containing all fault and alarm signals as binary code:

- *FaultWord1 (9.01),*
- *FaultWord2 (9.02),*
- *FaultWord3 (9.03),*
- *FaultWord4 (9.04),*
- *UserFaultWord (9.05),*
- *AlarmWord1 (9.06),*
- *AlarmWord2 (9.07),*
- *AlarmWord3 (9.08) and*

UserAlarmWord (9.09)

General messages

SDCS-CON-4

General messages will only be indicated on the seven-segment display of the SDCS-CON-4.

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition	Remark
8	not available	firmware is not running	1
.	not available	firmware is running, no faults, no alarms	-
-	not available	indication while loading firmware into SDCS-CON-4	-
d	not available	indication while loading DCS800 Control Panel texts into SDCS-CON-4	-
u	not available	DCS800 Control Panel text now formatting in the flash - don't switch off	-

Power-up errors (E)

SDCS-CON-4

Power-up errors will only be indicated on the seven segment display of the SDCS-CON-4. With a power-up error active it is not possible to start the drive.

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition	Remark
E01	not available	Checksum fault firmware flash	1,2
E02	not available	SDCS-CON-4 ROM memory test error	1,2
E03	not available	SDCS-CON-4 RAM memory test error (even addresses)	1,2
E04	not available	SDCS-CON-4 RAM memory test error (odd addresses)	1,2
E05	not available	SDCS-CON-4 hardware is not compatible, unknown board	1,2
E06	not available	SDCS-CON-4 watchdog timeout occurred	1,2

1. Units should be de-energized and energized. If the fault occurs again check the SDCS-CON-4, SDCS-PIN-4 respectively SDCS-POW-4 boards and change them if necessary.
2. Power-up errors are only enabled immediately after power on. If a power-up error is indicated during normal operation the reason is usually caused by EMC. In this case please check for proper grounding of cables, converter and cabinet.

Fault signals (F)

To avoid dangerous situations, damage of the motor, the drive or any other material some physical values must not exceed certain limits. Therefore limit values can be specified for these values by parameter setting which cause an alarm or a fault when the value exceeds the limits (e.g. max. armature voltage, max. converter temperature). Faults can also be caused by situations which inhibit the drive from normal operation (e.g. blown fuse).

A fault is a condition which requires an immediate stop of the drive in order to avoid danger or damage. The drive is stopped automatically and cannot be restarted before removing its cause.

All fault signals, with the exception of:
- **F501 AuxUnderVolt,**
- **F525 TypeCode,**
- **F547 HwFailure and**
- **F548 FwFailure**

are resetable in case the fault is eliminated.
To reset a fault following steps are required:
- remove the **Run** and **On** commands [*UsedMCW (7.04)* bit 3 and 0]
- eliminate the faults
- acknowledge the fault with **Reset** [*UsedMCW (7.04)* bit 7] via digital input, overriding control system or in **Local** mode with DCS800 Control Panel, DriveWindow or DriveWindow Light
- depending on the systems condition, generate **Run** and **On** commands [*UsedMCW (7.04)* bit 3 and 0] again

The fault signals will switch the drive off completely or partly depending on its trip level.
Trip level 1:
- main contactor is switched off immediately
- field contactor is switched off immediately
- fan contactor is switched off immediately

Trip level 2:
- main contactor is switched off immediately
- field contactor is switched off immediately
- fan contactor stays on as long as the fault is pending or as long as *FanDly (21.14)* is running

Trip level 3:
The drive is stopping via *SpeedFbFltMode (30.36)* , thus the
- main contactor is switched off immediately
- field contactor is switched off immediately in case of *SpeedFbFltMode (30.36)* = **CoastStop**, but it stays on in case of field heating or *SpeedFbFltMode (30.36)* = **DynBraking**
- fan contactor stays on

At standstill the
- main contactor cannot be switched on again
- field contactor stays on in case of field heating
- fan contactor stays on as long as *FanDly (21.14)* is running

Trip level 4:
As long as the drive is stopping via *FaultStopMode (30.30)* , the

- main contactor is switched off immediately in case of *FaultStopMode (30.30)* = **CoastStop** or **DynBraking**, but it stays on in case of *FaultStopMode (30.30)* = **RampStop** or **TorqueLimit**
- field contactor is switched off immediately in case of *FaultStopMode (30.30)* = **CoastStop**, but it stays on in case of field heating or *FaultStopMode (30.30)* = **RampStop**, **TorqueLimit** or **DynBraking**
- fan contactor is switched off immediately in case of *FaultStopMode (30.30)* = **CoastStop**, but stays on in case of *FaultStopMode (30.30)* = **RampStop**, **TorqueLimit** or **DynBraking**

At standstill the

- main contactor is switched off immediately
- field contactor stays on in case of field heating
- fan contactor stays on as long as *FanDly (21.14)* is running

Trip level 5
As long as the drive is stopping via any communication loss control [*LocalLossCtrl (30.27), ComLossCtrl (30.28), Ch0ComLossCtrl (70.05)* or *Ch2ComLossCtrl (70.15)*], the

- main contactor is switched off immediately or stays on depending on the selected communication loss control
- field contactor is switched off immediately or stays on depending on the selected communication loss control, but it stays on in case of field heating
- fan contactor is switched off immediately or stays on depending on the selected communication loss control

At standstill

- main contactor is switched off immediately
- field contactor stays on in case of field heating
- fan contactor stays on as long as *FanDly (21.14)* is running

In case a fault occurs, it stays active until the cause is eliminated and a **Reset** [*UsedMCW (7.04)* bit 7] is given.

Fault name	Fault number	Fault name	Fault number
12PulseCom	F535	M1FexNotOK	F529
12PCurDiff	F534	M1FexOverCur	F515
12PRevTime	F533	M1FexRdyLost	F537
12PSlaveFail	F536	M1OverLoad	F507
		M1OverTemp	F506
AIRange	F551	M2FexCom	F519
ApplLoadFail	F545	M2FexLowCur	F542
ArmCurRipple	F517	M2FexNotOK	F530
ArmOverCur	F502	M2FexOverCur	F518
ArmOverVolt	F503	M2FexRdyLost	F538
AuxUnderVolt	F501	M2OverLoad	F510
		M2OverTemp	F509
COM8Com	F543	MainContAck	F524
COM8Faulty	F540	MainsLowVolt	F512
ConvFanAck	F527	MainsNotSync	F514
ConvFanCur	F511	MainsOvrVolt	F513
ConvOverTemp	F504	MechBrake	F552
		MotorStalled	F531
ExternalDI	F526	MotOverSpeed	F532
ExtFanAck	F523		
		P2PandMFCom	F544
FastCurRise	F539	ParComp	F549
FieldAck	F521	ParMemRead	F550
FieldBusCom	F528		
FwFailure	F548	ResCurDetect	F505
		ReversalTime	F557
HwFailure	F547		
		SpeedFb	F522
I/OBoardLoss	F508		
		TachPolarity	F553
LocalCmdLoss	F546	TachoRange	F554
		TorqProving	F556
M1FexCom	F516	TypeCode	F525
M1FexLowCur	F541		

For additional fault messages see *SysFaultWord (9.10)*.

<table>
<tr><th>7-segment display</th><th>Text on DCS800 Control Panel, DriveWindow and DriveWindow Light</th><th>Definition / Action</th><th>Fault-word</th><th>Fault is active when</th><th>Triplevel</th></tr>
<tr><td>F501</td><td>501 AuxUnderVolt</td><td>Auxiliary undervoltage:
The auxiliary voltage is too low while the drive is in operation. If resetting fails, check:
– internal auxiliary voltages (SDCS-CON-4)
– and change SDCS-CON-4 and / or SDCS-PIN-4 respectively SDCS-POW-4 board
<table><tr><td>Auxiliary supply voltage</td><td>Trip level</td></tr><tr><td>230 VAC</td><td>< 185 VAC</td></tr><tr><td>115 VAC</td><td>< 96 VAC</td></tr></table></td><td>9.01, bit 0</td><td>RdyRun = 1</td><td>1</td></tr>
<tr><td>F502</td><td>502 ArmOverCur</td><td>Armature overcurrent:
Check:
– ArmOvrCurLev (30.09)
– parameter settings of group 43 (current control: armature current controller tuning)
– current and torque limitation in group 20
– all connections in the armature circuit, especially the incoming voltage for synchronizing. If the synchronizing voltage is not taken from the mains (e.g. via synchronizing transformer or 230 V / 115 V network) check that there is no phase shift between the same phases (use an oscilloscope).
– for faulty thyristors
– armature cabling
– in case of a rebuild kit proper connection of firing pulses and CT’s
– if TypeCode (97.01) = None and S ConvScaleCur (97.02) is set properly</td><td>9.01, bit 1</td><td>always</td><td>3</td></tr>
<tr><td>F503</td><td>503 ArmOverVolt</td><td>Armature overvoltage (DC):
Check:
– if setting of ArmOvrVoltLev (30.08) is suitable for the system
– parameter settings of group 44 (field excitation: field current controller tuning, EMF controller tuning, flux linearization)
– too high field current (e.g. problems with field weakening)
– if the motor was accelerated by the load,
– overspeed
– does the speed scaling fit, see SpeedScaleAct (2.29)
– proper armature voltage feedback
– connector X12 and X13 on SDCS-CON-4
– connector X12 and X13 on SDCS-PIN-4/51
– cutting of resistors for voltage coding on SDCS-PIN-51</td><td>9.01, bit 2</td><td>always</td><td>1</td></tr>
</table>

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F504	504 ConvOverTemp	**Converter overtemperature:** Wait until the converter is cooled down. Shutdown temperature see *MaxBridgeTemp (4.17)*. Check: – converter door open – converter fan supply voltage – converter fan direction of rotation – converter fan components – converter cooling air inlet (e.g. filter) – converter cooling air outlet – ambient temperature – inadmissible load cycle – connector X12 on SDCS-CON-4 – connector X12 and X22 on SDCS-PIN-4/51 – if *TypeCode (97.01)* = **None** and *S MaxBridgeTemp (97.04)* is set properly	9.01, bit 3	always	2
F505	505 ResCurDetect	**Residual current detection (sum of I_{L1}, I_{L2}, I_{L3} ≠ zero):** Check: – ResCurDetectSel (30.05), ResCurDetectLim (30.06), ResCurDetectDel (30.07) – sum current transformer, if necessary change transformer or SDCS-IOB-3 – disconnect the mains, verify safe isolation from supply in armature and field circuits and make insulation tests for the complete installation	9.01, bit 4	always	1
F506	506 M1OverTemp	**Motor 1 measured overtemperature:** Wait until the motor is cooled down. The motor fan will continue to work until the motor is cooled down under the alarm level. It is not possible to reset the fault as long as the motor remains too hot. Check: – *M1FaultLimTemp (31.07)*, *M1KlixonSel (31.08)* – *M1AlarmLimTemp (31.08)* – motor temperature – motor fan supply voltage – motor fan direction of rotation – motor fan components – motor cooling air inlet (e.g. filter) – motor cooling air outlet – motor temperature sensors and cabling – ambient temperature – inadmissible load cycle – inputs for temperature sensors on SDCS-CON-4 and SDCS-IOB-3	9.01, bit 5	always	2

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F507	507 M1OverLoad	**Motor 1 calculated overload**: Wait until the motor is cooled down. The motor fan will continue to work until the motor is calculated down under the alarm level. It is not possible to reset the fault as long as the motor remains too hot. Check: – *M1FaultLimLoad (31.04)* – *M1AlarmLimLoad (31.03)*	9.01, bit 6	always	2
F508	508 I/OBoardLoss	**I/O board not found or faulty:** Check: – *Diagnosis (9.11)* – *Ext IO Status (4.20)* – flat cable connections between SDCS-CON-4 and SDCS-IOB-2/3 – SDCS-COM-8 – *DCSLinkNodeID (94.01), Encoder2Module (98.01), CommModule (98.02), DIO ExtModule1 (98.03), DIO ExtModule2 (98.04), AIO ExtModule (98.06), AIO MotTempMeas (98.12), IO BoardConfig (98.15)*	9.01, bit 7	always	1
F509	509 M2OverTemp	**Motor 2 measured overtemperature:** Wait until the motor is cooled down. The motor fan will continue to work until the motor is cooled down under the alarm level. It is not possible to reset the fault as long as the motor remains too hot. Check: – *M2FaultLimTemp (49.37), M2KlixonSel (49.38)* – *M2AlarmLimTemp (49.36)* – motor temperature (let motor cool down and restart) – motor fan supply voltage – motor fan direction of rotation – motor fan components – motor cooling air inlet (e.g. filter) – motor cooling air outlet – motor temperature sensors and cabling – ambient temperature – inadmissible load cycle – inputs for temperature sensors on SDCS-CON-4 and SDCS-IOB-3	9.01, bit 8	always	2

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F510	510 M2OverLoad	**Motor 2 calculated overload**: Wait until the motor is cooled down. The motor fan will continue to work until the motor is cooled down under the alarm level. It is not possible to reset the fault as long as the motor remains too hot. Check: – *M2FaultLimLoad (49.34)* – *M2AlarmLimLoad (49.33)*	9.01, bit 9	always	2
F511	511 ConvFanCur	**Converter fan current:** only with *ConvTempDly (97.05)* ≠ 0 and a PW-10002/3 board connected to SDCS-PIN-4/51. Check: – converter fan supply voltage – converter fan direction of rotation – converter fan components – converter cooling air inlet (e.g. filter) – converter cooling air outlet – connector X12 on SDCS-CON-4 – connector X12 and X22 on SDCS-PIN-4/51	9.01, bit 10	RdyRun = 1	4
F512	512 MainsLowVolt	**Mains low (under-) voltage (AC):** Check: – *PwrLossTrip (30.21), UNetMin1 (30.22), UNetMin2 (30.23), PowrDownTime (30.24)* – if all 3 phases are present: o D1 to D4: measure also the fuses F100 to F102 on the SDCS-PIN-4 (see *Appendix B*) o D5 to D7: check also the connections U1, V1 and W1 on the SDCS-PIN-51 – if the mains voltage is within the set tolerance – if the main contactor closes and opens – if the mains voltage scaling is correct *[NomMainsVolt (99.10)]* – connector X12 and X13 on SDCS-CON-4 – connector X12 and X13 on SDCS-PIN-4/51 – cutting of resistors for voltage coding on SDCS-PIN-51 – D1 to D4: check if the field circuit has no short circuit or ground fault	9.01, bit 11	RdyRun = 1	3

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F513	513 MainsOvrVolt	**Mains overvoltage (AC):** Actual mains voltage is > 1.3 * *NomMainsVolt (99.10)* for more than 10 s and RdyRun = 1. Check: – if the mains voltage is within the set tolerance – if the mains voltage scaling is correct *[NomMainsVolt (99.10)]* – connector X12 and X13 on SDCS-CON-4 – connector X12 and X13 on SDCS-PIN-4/51 – cutting of resistors for voltage coding on SDCS-PIN-51	9.01, bit 12	RdyRun = 1	1
F514	514 MainsNotSync	**Mains not in synchronism (AC):** The synchronization with the mains frequency has been lost. Check: – mains supply – fuses etc. – mains frequency (50 Hz ±5 Hz; 60 Hz ±5 Hz) and stability (df/dt = 17 %/s) see *PLLIn (3.20)* at 50 Hz one period == 360° == 20 ms = 20,000 and at 60 Hz one period == 360° == 16.7 ms = 16,6667	9.01, bit 13	RdyRun = 1	3
F515	515 M1FexOverCur	**Motor 1 field exciter overcurrent:** Check: – in case this fault happens during field exciter autotuning deactivate the supervision by setting *M1FldOvrCurLev (30.13)* = 135 – *M1FldOvrCurLev (30.13)* – parameter settings of group 44 (field excitation: field current controller tuning) – connections of field exciter – insulation of cables and field winding – resistance of field winding – fault message at field exciter (7-segment display or flashing LED's)	9.01, bit 14	RdyRun = 1	1

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F516	516 M1FexCom	**Motor 1 field exciter communication loss:** Check: – *M1UsedFexType (99.12)* – *FexTimeOut (94.07)* – flat cable connections between SDCS-CON-4 and SDCS-PIN-4 – auxiliary voltage for integrated and external field exciter – DCSLink cable connections – DCSLink termination set dip switch S1100:1 = ON (DCF803-0016, DCF803-0035 and FEX-425-Int) – DCSLink node ID settings *[DCSLinkNodeID (94.01),* M1FexNode *(94.08)* respectively switches S800 and S801 on DCF803-0016, DCF803-0035 and FEX-425-Int] – fault message at field exciter (7-segment display or flashing LED's)	9.01, bit 15	RdyRun = 1	1
F517	517 ArmCurRipple	**Armature current ripple:** One or several thyristors may carry no current. Check: – *CurRippleSel (30.18), CurRippleLim (30.19)* – for too high gain of current controller *[M1KpArmCur (43.06)]* – current feedback with oscilloscope (6 pulses within one cycle visible?) – branch fuses – thyristor gate-cathode resistance – thyristor gate connection – current transformers (T51, T52)	9.02, bit 0	RdyRef = 1	3
F518	518 M2FexOverCur	**Motor 2 field exciter overcurrent:** Check: – *M2FldOvrCurLev (49.09)* – parameter settings of group 49 (field excitation: field current controller tuning) – connections of field exciter – insulation of cables and field winding – resistance of field winding – fault message at field exciter (7-segment display or flashing LED's)	9.02, bit 1	RdyRun = 1	1

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F519	519 M2FexCom	**Motor 2 field exciter communication loss:** Check: – *M2UsedFexType (49.07)* – *FexTimeOut (94.07)* – flat cable connections between SDCS-CON-4 and SDCS-PIN-4 – auxiliary voltage for integrated and external field exciter – DCSLink cable connections – DCSLink termination set dip switch S1100:1 = ON (DCF803-0016, DCF803-0035 and FEX-425-Int) – DCSLink node ID settings *[DCSLinkNodeID (94.01) , M2FexNode (94.09)* respectively switches S800 and S801 on DCF803-0016, DCF803-0035 and FEX-425-Int] – fault message at field exciter (7-segment display or flashing LED's)	9.02, bit 2	RdyRun = 1	1
F521	521 FieldAck	**Selected motor, field acknowledge missing:** Check: – *M1UsedFexType (99.12)*, if selection matches the field exciter type, *Mot1FexStatus (6.12), Mot2FexStatus (6.13)* – fault message at field exciter (7-segment display or flashing LED's) – **F521 FieldAck** is the sum fault for all field related faults like: 1. **F515 M1FexOverCur** 2. **F516 M1FexCom** 3. **F529 M1FexNotOK** 4. **F537 M1FexRdyLost** 5. **F541 M1FexLowCur**	9.02, bit 4	RdyRun = 1	1
F522	522 SpeedFb	**Selected motor, speed feedback:** The comparison of the speed feedback from pulse encoder or analog tacho has failed. Check: – *M1SpeedFbSel (50.03), SpeedFbFltMode (30.36), SpeedFbFltSel (30.17), EMF FbMonLev (30.15), SpeedFbMonLev (30.14)* – pulse encoder: encoder itself, alignment, cabling, coupling, power supply (feedback might be too low), mechanical disturbances, jumper S4 on SDCS-CON-4 – analog tacho: tacho itself, tacho polarity and voltage, alignment, cabling, coupling, mechanical disturbances, jumper S1 on SDCS-CON-4 – EMF: connection converter - armature circuit closed – SDCS-CON-4, SDCS-IOB-3, SDCS-POW-4	9.02, bit 5	always	3

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F523	523 ExtFanAck	**External fan acknowledge missing:** Check: – *MotFanAck (10.06)* – external fan contactor – external fan circuit – external fan supply voltage – used digital inputs and outputs (group 14)	9.02, bit 6	RdyRun = 1	4
F524	524 MainContAck	**Main contactor acknowledge missing:** Check: – *MainContAck (10.21)* – switch on - off sequence – auxiliary contactor (relay) switching the main contactor after On/Off command – safety relays – used digital inputs and outputs (group 14)	9.02, bit 7	RdyRun = 1	3
F525	525 TypeCode	**Type code mismatch:** When using D1, D2, D3 or D4 modules the current and voltage range of the type code setting is limited to max 1000 ADC and max 600 VAC. Check: – *TypeCode (97.01)*, S *ConvScaleCur (97.02)*, *S ConvScaleVolt (97.03)*	9.02, bit 8	always	1
F526	526 ExternalDI	**External fault via binary input:** There is no problem with the drive itself! Check: – *ExtFaultSel (30.31)*, *ExtFaultOnSel (30.33)*	9.02, bit 9	Always or RdyRun = 1	1
F527	527 ConvFanAck	**Converter fan acknowledge missing**: Check: – *ConvFanAck (10.20)* – *FanDly (21.14)* – converter fan contactor – converter fan circuit – converter fan klixon – converter fan components – converter fan supply voltage – converter fan direction of rotation – converter door open – converter cooling air inlet (e.g. filter) – converter cooling air outlet – D6 an D7 pressure switch (setting should be 2 mbar) – used digital inputs and outputs (group 14)	9.02, bit 10	RdyRun = 1	4

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F528	528 FieldBusCom	**Fieldbus communication loss**: **F528 FieldBusCom** is only activated after the first data set from the overriding control is received by the drive. Before the first data set is received only **A128 FieldBusCom** is active. The reason is to suppress unnecessary faults (the start up of the overriding control is usually slower than the one of the drive). Check: – *CommandSel (10.01), ComLossCtrl (30.28), FB TimeOut (30.35), CommModule (98.02)* – parameter settings of group 51 (fieldbus) – fieldbus cable – fieldbus termination – fieldbus adapter	9.02, bit 11	always if *FB TimeOut (30.35)* $\neq$ 0	5
F529	529 M1FexNotOK	**Motor 1 field exciter not okay:** A fault was found during self-diagnosis of field exciter or power failure in field exciter 1. Check: – field exciter operation and change the field exciter, if necessary – fault message at field exciter (7-segment display or flashing LED's)	9.02, bit 12	always	1
F530	530 M2FexNotOK	**Motor 2 field exciter not okay:** A fault was found during self-diagnosis of field exciter or power failure in field exciter 2. Check: – field exciter operation and change the field exciter, if necessary – fault message at field exciter (7-segment display or flashing LED's)	9.02, bit 13	always	1
F531	531 MotorStalled	**Selected motor, motor stalled:** The motor torque exceeded *StallTorq (30.03)* for a time longer than *StallTime (30.01)* while the speed feedback was below *StallSpeed (30.02)*. Check: – motor stalled (mechanical couplings of the motor) – proper conditions of load – correct field current – parameter settings of group 20 (limits: current and torque limits)	9.02, bit 14	RdyRef = 1	3

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F532	532 MotOverSpeed	**Selected motor, motor overspeed:** Check: – *M1OvrSpeed (30.16)* – parameter settings of group 24 (speed control: speed controller) – scaling of speed controller loop *[SpeedScaleAct (2.29)]* – drive speed *[MotSpeed (1.04)]* vs. measured motor speed (hand held tacho) – field current too low – speed feedback (encoder, tacho) – connection of speed feedback – if the motor was accelerated by the load – in case of EMF speed feedback if the DC-voltage measurement (C1, D1) might be swapped or if the armature circuit is open (e.g. DC-fuses, DC-breaker)	9.02, bit 15	always	3
F533	533 12PRevTime	**12-pulse reversal timeout:** Current direction not changed before *12P RevTimeOut (47.05)* is elapsed. Check: – for high inductive motor – too high motor voltage compared to mains voltage	9.03, bit 0	RdyRef = 1	3
F534	534 12PCurDiff	**12-pulse current difference (only for 12-pulse parallel operation):** Check: – *DiffCurLim (47.02), DiffCurDly (47.03)* – parameter settings of group 43 (current control: armature current controller)	9.03, bit 1	always	3
F535	535 12PulseCom	**12-pulse communication:** Check: – *12P TimeOut (94.03)* – DCSLink cable connections – DCSLink termination – DCSLink node ID settings *[DCSLinkNodeID (94.01) , 12P SlaNode (94.04)]*	9.03, bit 2	RdyOn = 1	3
F536	536 12PSlaveFail	**12-pulse slave failure:** 12-pulse master is tripped by a fault of the 12-pulse slave. Check: – Fault logger of 12-pulse slave	9.03, bit 3	RdyOn = 1	4

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F537	537 M1FexRdyLost	**Motor 1 field exciter ready lost:** Field exciter lost ready-for-operation message while working. AC-voltage missing or not in synchronism. Check: – if all phases are present – if the mains voltage is within the set tolerance – fault message at field exciter (7-segment display or flashing LED's)	9.03, bit 4	RdyRun = 1	1
F538	538 M2FexRdyLost	**Motor 2 field exciter ready lost:** Field exciter lost ready-for-operation message while working. AC-voltage missing or not in synchronism. Check: – if all phases are present – if the mains voltage is within the set tolerance – fault message at field exciter (7-segment display or flashing LED's)	9.03, bit 5	RdyRun = 1	1
F539	539 FastCurRise	**Fast current rise:** Actual current di/dt too fast. Check: – *ArmCurRiseMax (30.10)*	9.03, bit 6	RdyRef = 1 and generating	1
F540	540 COM8Faulty	**SDCS-COM-8 faulty:** Check: – Change SDCS-COM-8 and / or SDCS-CON-4	9.03, bit 7	RdyOn = 1	1
F541	541 M1FexLowCur	**Motor 1 field exciter low (under-) current:** Check: – *M1FldMinTrip (30.12) , FldMinTripDly (45.18)* – parameter settings of group 44 (field excitation: field current controller tuning, EMF controller tuning, flux linearization) – motor name plate for minimum current at maximum field weakening (maximum speed) – field circuit fuses – field contactor is not closed – if the field current oscillates – if the motor is not compensated and has a high armature reaction – fault message at field exciter (7-segment display or flashing LED's)	9.03, bit 8	always	1

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F542	542 M2FexLowCur	**Motor 2 field exciter low (under-) current:** Check: – *M2FldMinTrip (49.08), FldMinTripDly (45.18)* – parameter settings of group 44 (field excitation: field current controller tuning, EMF controller tuning, flux linearization) – motor name plate for minimum current at maximum field weakening (maximum speed) – field circuit fuses – field contactor is not closed – if the field current oscillates – if the motor is not compensated and has a high armature reaction – fault message at field exciter (7-segment display or flashing LED's)	9.03, bit 9	always	1
F543	543 COM8Com	**SDCS-COM-8 communication loss (overriding control and master-follower):** Check: – *CommandSel (10.01), Ch0 ComLossCtrl (70.05), Ch0 TimeOut (70.04), Ch2 ComLossCtrl (70.15), Ch2 TimeOut (70.14*), *Ch0 DriveBus (71.01)* – fiber optic cables to overriding control (channel 0) – overriding control adapters – fiber optic cables between master and followers (channel 2)	9.03, bit 10	RdyOn = 1	5
F544	544 P2PandMFCom	**Peer to peer and master-follower communication loss:** Check: – *ComLossCtrl (30.28), MailBox1 (94.12), MailBox2 (94.18), MailBox3 (94.24), MailBox4 (94.30), MailBoxCycle1 (94.13), MailBoxCycle2 (94.19), MailBoxCycle3 (94.25), MailBoxCycle4 (94.31)* – DCSLink cable connections – DCSLink termination – DCSLink node ID settings *[DCSLinkNodeID (94.01)]*	9.03, bit 11	always	5
F545	545 ApplLoadFail	**Application load failure:** Check: – *Diagnosis (9.11)*	9.03, bit 12	always	1

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F546	546 LocalCmdLoss	**Local command loss:** Communication fault with DCS800 Control Panel, DriveWindow or DriveWindow Light during local mode. Check: – *LocalLossCtrl (30.27)* – if control DCS800 Control Panel is disconnected – connection adapter – cables	9.03, bit 13	local	5
F547	547 HwFailure	**Hardware failure:** For more details check *Diagnosis (9.11)*.	9.03, bit 14	always	1
F548	548 FwFailure	**Firmware failure:** For more details check *Diagnosis (9.11)*. Can happen after firmware download using an USB to COMx converter.	9.03, bit 15	always	1
F549	549 ParComp	**Parameter compatibility:** When downloading parameter sets or during power-up the firmware attempts to write their values. If the setting is not possible or not compatible the parameter is set to default. The parameters causing the fault can be identified in *Diagnosis (9.11)*. Check: – parameter setting	9.04, bit 0	always	1
F550	550 ParMemRead	**Parameter or Memory Card read:** Reading the actual parameter set or a user parameter set from either flash or Memory Card failed (checksum fault) Check: – one or both parameter sets (**User1** and / or **User2**) have not been saved properly - see *ApplMacro (99.08)* – Memory Card and – SDCS-CON-4	9.04, bit 1	always	1
F551	551 AIRange	**Analog input range:** Undershoot of one of the analog input values under 4mA / 2V. Check: – *AI Mon4mA (30.29)* – used analog inputs connections and cables – polarity of connection	9.04, bit 2	always	4
F552	552 MechBrake	**Selected motor, mechanical brake:** The acknowledge signal for brake opened (lifted) or brake closed (applied) is missing. Check: – *M1BrakeAckSel (42.02), M1BrakeFltTime (42.05), BrakeFaultFunc (42.06), M1BrakeLongTime (42.12)* – brake – brake cabling – used digital inputs and outputs (group 14)	9.04, bit 3	always	3

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F553	553 TachPolarity	**Selected motor, tacho polarity:** The polarity of the analog tacho respectively pulse encoder [depending on *M1SpeedFbSell (50.03)*] is checked against the EMF. Check: – *EMF FbMonLev (30.15)*, *SpeedFbMonLev (30.14)* – polarity of tacho cable – polarity of pulse encoder cable (e.g. swap channels A and A not) – polarity of armature and field cables – direction of motor rotation	9.04, bit 4	always	3
F554	554 TachoRange	**Selected motor, tacho range:** Overflow of AITacho input Check: – for the right connections (X3:1 to X3:4) on the SDCS-CON-4	9.04, bit 5	always	3
F556	556 TorqProving	**Selected motor, torque proving:** The acknowledge signal for torque proving is missing. Check: – *M1TorqProvTime (42.10)* – the Adaptive Program, application program or overriding control providing the acknowledge signal **TorqProvOK** [*AuxCtrlWord2 (7.03)* bit 11]	9.04, bit 7	while *M1TorqProvTime (42.10)* is active	3
F557	557 ReversalTime	**Reversal time:** Current direction not changed before *ZeroCurTimeOut (97.19)* is elapsed. Check: – for high inductive motor – too high motor voltage compared to mains voltage – lower *RevDly (43.14)* if possible and – increase *ZeroCurTimeOut (97.19)*	9.04, bit 8	RdyRef = 1	3
F601	601 APFault1	**User defined fault by Adaptive Program**	9.04, bit 11	always	1
F602	602 APFault2	**User defined fault by Adaptive Program**	9.04, bit 12	always	1
F603	603 APFault3	**User defined fault by Adaptive Program**	9.04, bit 13	always	1
F604	604 APFault4	**User defined fault by Adaptive Program**	9.04, bit 14	always	1
F605	605 APFault5	**User defined fault by Adaptive Program**	9.04, bit 15	always	1
F610	610 UserFault1	**User defined fault by application program**	9.05, bit 0	always	*
F611	611 UserFault2	**User defined fault by application program**	9.05, bit 1	always	*
F612	612 UserFault3	**User defined fault by application program**	9.05, bit 2	always	*

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Fault-word	Fault is active when	Triplevel
F613	613 UserFault4	**User defined fault by application program**	9.05, bit 3	always	*
F614	614 UserFault5	**User defined fault by application program**	9.05, bit 4	always	*
F615	615 UserFault6	**User defined fault by application program**	9.05, bit 5	always	*
F616	616 UserFault7	**User defined fault by application program**	9.05, bit 6	always	*
F617	617 UserFault8	**User defined fault by application program**	9.05, bit 7	always	*
F618	618 UserFault9	**User defined fault by application program**	9.05, bit 8	always	*
F619	619 UserFault10	**User defined fault by application program**	9.05, bit 9	always	*
F620	620 UserFault11	**User defined fault by application program**	9.05, bit 10	always	*
F621	621 UserFault12	**User defined fault by application program**	9.05, bit 11	always	*
F622	622 UserFault13	**User defined fault by application program**	9.05, bit 12	always	*
F623	623 UserFault14	**User defined fault by application program**	9.05, bit 13	always	*
F624	624 UserFault15	**User defined fault by application program**	9.05, bit 14	always	*
F625	625 UserFault16	**User defined fault by application program**	9.05, bit 15	always	*

* Triplevel is set in the application program

SDCS-COM-8 messages

Details of the SDCS-COM-8 messages are available in *SysFaultWord (9.10)*.

7-segment display	Text on DriveWindow	Definition / Action	Fault-word	Fault is active when	Triplevel
-	OS_xx	**Operating system message xx:** An OS_xx message is an empty and thus not used message of the SDCS-COM-8 operating system. If an OS_xx message is indicated during normal operation the reason is usually caused by EMC. In this case please check for proper version of the SDCS-COM-8 (revision I and higher), grounding of cables, converter and cabinet.	-	-	-

Alarm signals (A)

An alarm is a message, that a condition occurred, which may lead to a dangerous situation. It is displayed and written into the fault logger. However, the cause for the alarm can inhibit the drive from continuing with normal operation. If the cause of the alarm disappears the alarm will be automatically reset. The fault logger shows the appearing alarm (A**1**xx) with a plus sign and the disappearing alarm (A**2**xx) with a minus sign. An appearing user defined alarm is indicated as A**3**xx. A disappearing user defined alarm is indicated as A**4**xx.

The alarm handling must provides 4 alarm levels.

Alarm level 1:

- the drive keeps on running and the alarm is indicated
- after the drive is stopped, the main contactor cannot be switched on again (no re-start possible)

Alarm level 2:

- the drive keeps on running and the alarm is indicated
- fan contactor stays on as long as the alarm is pending
- if the alarm disappears *FanDly (21.14)* will start

Alarm level 3:

- **AutoReclosing** (auto re-start) is [*AuxStatWord (8.02)* bit 15] active
- **RdyRun** [*MainStatWord (8.01)* bit 1] is disabled, but the drive is automatically restarted when the alarm condition vanishes
- α is set to 150°
- single firing pulses

Alarm level 4:

- the drive keeps on running and the alarm is indicated

In case an alarm occurs, it stays active until the cause is eliminated. Then the alarm will automatically disappear, thus a **Reset** [*UsedMCW (7.04)* bit 7] is not needed and will have no effect.

Alarm name	Alarm number		Alarm name	Alarm number	
	appearing	disappearing		appearing	disappearing
AIRange	A127	A227	M2OverTemp	A109	A209
ApplDiff	A119	A219	MainsLowVolt	A111	A211
ArmCurDev	A114	A214	MechBrake	A122	A222
ArmCurRipple	A117	A217	MemCardFail	A143	A243
AutotuneFail	A121	A221	MemCardMiss	A142	A242
BrakeLongFalling	A116	A216	NoAPTaskTime	A136	A236
COM8Com	A113	A213	Off2FieldBus	A138	A238
COM8FwVer	A141	A241	Off2ViaDI	A101	A201
ConvOverTemp	A104	A204	Off3FieldBus	A139	A239
			Off3ViaDI	A102	A202
DC BreakAck	A103	A203	OverVoltProt	A120	A220
DynBrakeAck	A105	A205			
			P2PandMFCom	A112	A212
ExternalDI	A126	A226	ParAdded	A131	A231
			ParComp	A134	A234
FaultSuppres	A123	A223	ParConflict	A132	A232
FieldBusCom	A128	A228	ParRestored	A129	A229
FoundNewAppl	A118	A218	ParUpDwnLoad	A135	A235
IllgFieldBus	A140	A240	RetainInv	A133	A233
LocalCmdLoss	A130	A230	SpeedFb	A125	A225
			SpeedNotZero	A137	A237
M1OverLoad	A107	A207	SpeedScale	A124	A224
M1OverTemp	A106	A206			
M2OverLoad	A110	A210	TachoRange	A115	A215

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A101	101 Off2ViaDI	**Off2 (Emergency Off / Coast stop) pending via digital input - start inhibition:** There is no problem with the drive itself! Check: – *Off2 (10.08)*, if necessary invert the signal (group 10)	9.06, bit 0	RdyRun = 1	1
A102	102 Off3ViaDI	**Off3 (E-stop) pending via digital input:** There is no problem with the drive itself! Check: – *E Stop (10.09)*, if necessary invert the signal (group 10)	9.06, bit 1	RdyRun = 1	1
A103	103 DC BreakAck	**Selected motor, DC-Breaker acknowledge missing:** α is set to 150° and single firing pulses are given, thus the drive cannot be started or re-started while the DC-breaker acknowledge is missing. Check: – *DC BreakAck (10.23)*, if necessary invert the signal (group 10)	9.06, bit 2	RdyRun = 1	3
A104	104 ConvOverTemp	**Converter overtemperature:** Wait until the converter is cooled down. Shutdown temperature see *MaxBridgeTemp (4.17)*. The converter overtemperature alarm will already appear at approximately 5°C below the shutdown temperature. Check: – *ConvFanAck (10.20)* – *FanDly (21.14)* – converter door open – converter fan supply voltage – converter fan direction of rotation – converter fan components – converter cooling air inlet (e.g. filter) – converter cooling air outlet – ambient temperature – inadmissible load cycle – connector X12 on SDCS-CON-4 – connector X12 and X22 on SDCS-PIN-4/51	9.06, bit 3	always	2
A105	105 DynBrakeAck	**Selected motor, dynamic braking is still pending:** α is set to 150° and single firing pulses are given, thus the drive cannot be started or re-started while dynamic braking is active, except if *FlyStart (21.10)* = **FlyStartDyn**. Check: – *DynBrakeAck (10.22)* – *FlyStart (21.10)*	9.06, bit 4	RdyRun = 1	3

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A106	106 M1OverTemp	**Motor 1 measured overtemperature:** Check: – *M1AlarmLimTemp (31.06)* – motor temperature – motor fan supply voltage – motor fan direction of rotation – motor fan components – motor cooling air inlet (e.g. filter) – motor cooling air outlet – motor temperature sensors and cabling – ambient temperature – inadmissible load cycle – inputs for temperature sensors on SDCS-CON-4 and SDCS-IOB-3	9.06, bit 5	always	2
A107	107 M1OverLoad	**Motor 1 calculated overload**: Check: – *M1AlarmLimLoad (31.03)*	9.06, bit 6	always	2
A109	109 M2OverTemp	**Motor 2 measured overtemperature:** Check: – *M2AlarmLimTemp (49.36)* – motor temperature – motor fan supply voltage – motor fan direction of rotation – motor fan components – motor cooling air inlet (e.g. filter) – motor cooling air outlet – motor temperature sensors and cabling – ambient temperature – inadmissible load cycle – inputs for temperature sensors on SDCS-CON-4 and SDCS-IOB-3	9.06, bit 8	always	2
A110	110 M2OverLoad	**Motor 2 calculated overload**: Check: – *M2AlarmLimLoad (49.33)*	9.06, bit 9	always	2
A111	111 MainsLowVolt	**Mains low (under-) voltage (AC):** α is set to 150°; single firing pulses Check: – *PwrLossTrip (30.21), UNetMin1 (30.22), UNetMin2 (30.23)*, – If all 3 phases are present – if the mains voltage is within the set tolerance – if the main contactor closes and opens – if the mains voltage scaling is correct *[NomMainsVolt (99.10)]* – connector X12 and X13 on SDCS-CON-4 – connector X12 and X13 on SDCS-PIN-4/51 – cutting of resistors for voltage coding on SDCS-PIN-51	9.06, bit 10	RdyRun = 1	3

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A112	112 P2PandMFCom	**Peer to peer and master-follower communication loss:** Check: – *ComLossCtrl (30.28), MailBox1 (94.12), MailBox2 (94.18), MailBox3 (94.24), MailBox4 (94.30), MailBoxCycle1 (94.13), MailBoxCycle2 (94.19), MailBoxCycle3 (94.25), MailBoxCycle4 (94.31)* – DCSLink cable connections – DCSLink termination – DCSLink node ID settings *[DCSLinkNodeID (94.01)]*	9.06, bit 11	always	4
A113	113 COM8Com	**SDCS-COM-8 communication loss (overriding control and master-follower):** Check: – *CommandSel (10.01), Ch0 ComLossCtrl (70.05), Ch0 TimeOut (70.04), Ch2 ComLossCtrl (70.15), Ch2 TimeOut (70.14)*, *Ch0 DriveBus (71.01)* – fiber optic cables to overriding control (channel 0) – overriding control adapters – fiber optic cables between master and followers (channel 2)	9.06, bit 12	always	4
A114	114 ArmCurDev	**Armature Current Deviation:** Is shown, if the current reference [*CurRefUsed (3.12)*] differs from current actual [*MotCur (1.06)*] for longer than 5 sec by more than 20% of nominal motor current. In other words if the current controller cannot match the given reference, the alarm signal is created. Normally the reason is a too small incoming voltage compared to the motor EMF. For non motoric applications it is possible to block the alarm using *AuxCtrlWord2 (7.03)* bit 6. Check: – DC fuses blown – ratio between mains voltage and armature voltage (either the mains voltage is too low or the motor's armature voltage is too high) – *ArmAlphaMin (20.15)* is set too high	9.06, bit 13	RdyRef = 1	4

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A115	115 TachoRange	**Selected motor, tacho range:** If **A115 TachoRange** comes up for longer than 10 seconds there is an overflow of the AITacho input. Check: – for the right connections (X3:1 to X3:4) on the SDCS-CON-4 If **A115 TachoRange** comes up for 10 seconds and vanishes again *M1OvrSpeed (30.16)* or *M2OvrSpeed (49.21)* has been changed. In this case a new tacho fine tuning has to be done [*ServiceMode (99.06)* = **TachFineTune**].	9.06, bit 14	always	4
A116	116 BrakeLongFalling	**Selected motor, mechanical brake:** The acknowledge signal for brake closed (applied) is missing. Check: – *M1BrakeAckSel (42.02), BrakeFaultFunc (42.06), M1BrakeLongTime (42.12)* – brake – brake cabling – used digital inputs and outputs (group 14)	9.06, bit 15	always	4
A117	117 ArmCurRipple	**Armature current ripple:** One or several thyristors may carry no current. Check: – *CurRippleSel (30.18), CurRippleLim (30.19)* – for too high gain of current controller *[M1KpArmCur (43.06)]* – current feedback with oscilloscope (6 pulses within one cycle visible?) – branch fuses – thyristor gate-cathode resistance – thyristor gate connection – current transformers (T51, T52)	9.07, bit 0	RdyRef = 1	4
A118	118 FoundNewAppl	**Found new application on Memory Card**: Activate application on Memory Card by means of *ParApplSave (16.06)* = **EableAppl**	9.07, bit 1	directly after energizing of auxiliary supply	1
A119	119 ApplDiff	**Application on drive and Memory Card are different**: Activate application on Memory Card by means of *ParApplSave (16.06)* = **EableAppl**	9.07, bit 2	directly after energizing of auxiliary supply	1
A120	120 OverVoltProt	**Overvoltage protection active:** Overvoltage protection DCF806 is active and converter is blocked. α is set to 150°; single firing pulses Check: – *OvrVoltProt (10.13)* if necessary invert the signal (group 10) – field converter cables and connections	9.07, bit 3	always	3

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A121	121 AutotuneFail	**Autotuning failed:** For more details check *Diagnosis (9.11)* To clear the alarm set *ServiceMode (99.06)* = **NormalMode**	9.07, bit 4	always	4
A122	122 MechBrake	**Selected motor, mechanical brake:** Acknowledge brake applied (closed) is missing or torque actual does not reach *StrtTorqRef (42.08)*, during torque proving. Check: – *BrakeFaultFunc (42.06), M1StrtTorqRefSel (42.07)*, *M2StrtTorqRefSel (49.44)* – brake – brake cabling – used digital inputs and outputs (group 14)	9.07, bit 5	always	4
A123	123 FaultSuppres	**Fault suppressed:** At least one fault message is currently active and suppressed. –	9.07, bit 6	always	4
A124	124 SpeedScale	**Speed scaling out of range:** The parameters causing the alarm can be identified in *Diagnosis (9.11)*. α is set to 150°; single firing pulses Check: – *M1SpeedMin (20.01), M1SpeedMax (20.02), M2BaseSpeed (49.03), M2SpeedMin (49.19), M2SpeedMax (49.20), M2SpeedScale (49.22), M1SpeedScale (50.01), M1BaseSpeed (99.04)*	9.07, bit 7	always	3
A125	125 SpeedFb	**Selected motor, speed feedback:** The comparison of the speed feedback from pulse encoder or analog tacho has failed. Check: – *M1SpeedFbSel (50.03), SpeedFbFltMode (30.36), SpeedFbFltSel (30.17)*, *EMF FbMonLev (30.15)*, *SpeedFbMonLev (30.14)* – pulse encoder: encoder itself, alignment, cabling, coupling, power supply (feedback might be too low), mechanical disturbances, jumper S4 on SDCS-CON-4 – analog tacho: tacho itself, tacho polarity and voltage, alignment, cabling, coupling, mechanical disturbances, jumper S1 on SDCS-CON-4 – EMF: connection converter - armature circuit closed – SDCS-CON-4, SDCS-IOB-3, SDCS-POW-4	9.07, bit 8	always	4

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A126	126 ExternalDI	**External alarm via binary input:** There is no problem with the drive itself! Check: – *ExtAlarmSel (30.32)*, alarm = 0, ExtAlarmOnSel *(30.34)*	9.07, bit 9	always	4
A127	127 AIRange	**Analog input range:** Undershoot of one of the analog input values under 4mA / 2V. Check: – *AI Mon4mA (30.29)* – used analog inputs connections and cables – polarity of connection	9.07, bit 10	always	4
A128	128 FieldBusCom	**Fieldbus communication loss**: **F528 FieldBusCom** is only activated after the first data set from the overriding control is received by the drive. Before the first data set is received only **A128 FieldBusCom** is active. The reason is to suppress unnecessary faults (the start up of the overriding control is usually slower than the one of the drive). Check: – *ComLossCtrl (30.28), FB TimeOut (30.35), CommModule (98.02)* – parameter settings of group 51 (fieldbus) – fieldbus cable – fieldbus termination – fieldbus adapter	9.07, bit 11	always if *FB TimeOut (30.35)* ≠ 0	4
A129	129 ParRestored	**Parameter restored:** The parameters found in the flash were invalid at power-up (checksum fault). All parameters were restored from the parameter backup.	9.07, bit 12	always	4
A130	130 LocalCmdLoss	**Local command loss:** Connection fault with DCS800 Control Panel, DriveWindow or DriveWindow Light. Check: – *LocalLossCtrl (30.27)* – if control DCS800 Control Panel is disconnected – connection adapter – cables	9.07, bit 13	local	4
A131	131 ParAdded	**Parameter added:** A new firmware with a different amount of parameters was downloaded. The new parameters are set to their default values. The parameters causing the alarm can be identified in *Diagnosis (9.11)*. Check: – new parameters and set them to the desired values	9.07, bit 14	after download of firmware for max. 10 s	4

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A132	132 ParConflict	**Parameter setting conflict:** Is triggered by parameter settings conflicting with other parameters. The parameters causing the alarm can be identified in *Diagnosis (9.11)*.	9.07, bit 15	always	4
A133	133 RetainInv	**Retain data invalid:** Set when the retain data in the flash are invalid during power-up. In this case the backup data are used. **Note:** The backup of the lost retain data reflects the status at the previous power-up. Examples for retain data are: – fault logger data, – *Data1 (19.01)* to *Data4 (19.04)*, – I/O options (see group 98) and – parameters defined by means of DCS800 ControlBuilder (CoDeSys) with the box RETAIN ticked The situation of invalid retain data occurs, if the auxiliary voltage of the DCS800 is switched off about 2 seconds after power-up (while the retain data sector is being rearranged). Check: – if the flash of the SDCS-CON-4 is defective and – if the auxiliary power supply has a problem	9.08, bit 0	directly after energizing of electronics for max. 10 s	4
A134	134 ParComp	**Parameter compatibility:** When downloading parameter sets or during power-up the firmware attempts to write the parameters. If the setting is not possible or not compatible the parameter is set to default. The parameters causing the alarm can be identified in *Diagnosis (9.11)*. Check: – parameter setting	9.08, bit 1	after download of a parameter set for max. 10 s	4
A135	135 ParUpDwnLoad	**Parameter up- or download failed:** The checksum verification failed during up- or download of parameters. Please try again. Two or more parameter set actions were requested at the same time. Please try again.	9.08, bit 2	after up- or download of parameters for max. 10 s	4
A136	136 NoAPTaskTime	**Adaptive Program task time not set:** The task time for the Adaptive Program is not set, while the Adaptive Program is started. Check: – that *TimeLevSel (83.04)* is set to **5 ms, 20 ms, 100 ms** or **500 ms** when *AdapProgCmd (83.01)* is set to **Start**, **SingleCycle** or **SingleStep**	9.08, bit 3	always	4

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A137	137 SpeedNotZero	**Speed not zero:** Re-start of drive is not possible. Speed zero [see *M1ZeroSpeedLim (20.03)* or *M2ZeroSpeedLim (49.04)*] has not been reached. In case of an alarm set **On** = **Run** = 0 and check if the actual speed is within the zero speed limit. This alarm is valid for: – normal stop, **Off1N** [UsedMCW (7.04) bit 0] in case *FlyStart (21.10)* = **StartFrom0**, – Coast Stop, **Off2N** [UsedMCW (7.04) bit 1], – E-stop, **Off3N** [UsedMCW (7.04) bit 2] and – if the drive is de-energized and then re-energized. Check: – *M1ZeroSpeedLim (20.03)* – *FlyStart (21.10)* – *M1SpeedFbSel (50.03)* – *M2SpeedFbSel (49.24)* – *M2ZeroSpeedLim (49.04)* – for proper function of the used speed feedback devices (analog tacho / encoder)	9.08, bit 4	Not active if RdyRef = 1	1
A138	138 Off2FieldBus	**Off2 (Emergency Off / Coast Stop) pending via** *MainCtrlWord (7.01)* **/ fieldbus - start inhibition:** There is no problem with the drive itself! Check: – *MainCtrlWord (7.01)* bit1 **Off2N**	9.08, bit 5	RdyRun = 1	1
A139	139 Off3FieldBus	**Off3 (E-stop) pending via** *MainCtrlWord (7.01)* **/ fieldbus:** There is no problem with the drive itself! Check: – *MainCtrlWord (7.01)* bit2 **Off3N**	9.08, bit 6	RdyRun = 1	1
A140	140 IllgFieldBus	**Illegal fieldbus settings:** The fieldbus parameters in group 51 (fieldbus) are not set according to the fieldbus adapter or the device has not been selected. Check: – group 51 (fieldbus) – configuration of fieldbus adapter	9.08, bit 7	always	4
A141	141 COM8FwVer	**SDCS-COM-8 firmware version conflict:** Invalid combination of SDCS-CON-4 firmware and SDCS-COM-8 firmware. Check: – for valid combination of SDCS-CON-4 [*FirmwareVer (4.01)*] and SDCS-COM-8 [*Com8SwVersion (4.11)*] firmware version according to the release notes	9.08, bit 8	always	4

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A142	142 MemCardMiss	**Memory Card missing:** There is an application loaded in the drive. The Memory Card belonging to the application is not found. Check: – if the Memory Card is properly plugged into the SDCS-CON-4 (X20) – de-energize the electronics, insert the proper Memory Card and reenergize – *ParApplSave (16.06)* – in case there is no Memory Card used set *ParApplSave (16.06)* = **DisableAppl**	9.08, bit 9	directly after energizing of electronics	1
A143	143 MemCardFail	**Memory Card failure:** Checksum failure or wrong Memory Card Check: – Memory Card – if proper ABB Memory Card is used – *ParApplSave (16.06)*	9.08, bit 10	directly after energizing of electronics	1
A2xx	2xx <alarm name>	**Disappearing system alarm**	-	-	
A301	301 APAlarm1	**User defined alarm by Adaptive Program**	9.08, bit 11	always	4
A302	302 APAlarm2	**User defined alarm by Adaptive Program**	9.08, bit 12	always	4
A303	303 APAlarm3	**User defined alarm by Adaptive Program**	9.08, bit 13	always	4
A304	304 APAlarm4	**User defined alarm by Adaptive Program**	9.08, bit 14	always	4
A305	305 APAlarm5	**User defined alarm by Adaptive Program**	9.08, bit 15	always	4
A310	310 UserAlarm1	**User defined fault by application program**	9.09, bit 0	always	*
A311	311 UserAlarm1	**User defined fault by application program**	9.09, bit 1	always	*
A312	312 UserAlarm2	**User defined fault by application program**	9.09, bit 2	always	*
A313	313 UserAlarm3	**User defined fault by application program**	9.09, bit 3	always	*
A314	314 UserAlarm4	**User defined fault by application program**	9.09, bit 4	always	*
A315	315 UserAlarm5	**User defined fault by application program**	9.09, bit 5	always	*
A316	316 UserAlarm6	**User defined fault by application program**	9.09, bit 6	always	*
A317	317 UserAlarm7	**User defined fault by application program**	9.09, bit 7	always	*
A318	318 UserAlarm8	**User defined fault by application program**	9.09, bit 8	always	*
A319	319 UserAlarm9	**User defined fault by application program**	9.09, bit 9	always	*
A320	320 UserAlarm10	**User defined fault by application program**	9.09, bit 10	always	*

7-segment display	Text on DCS800 Control Panel, DriveWindow and DriveWindow Light	Definition / Action	Alarm-word	Alarm is active when	Alarmlevel
A321	321 UserAlarm11	**User defined fault by application program**	9.09, bit 11	always	*
A322	322 UserAlarm12	**User defined fault by application program**	9.09, bit 12	always	*
A323	323 UserAlarm13	**User defined fault by application program**	9.09, bit 13	always	*
A324	324 UserAlarm14	**User defined fault by application program**	9.09, bit 14	always	*
A325	325 UserAlarm16	**User defined fault by application program**	9.09, bit 15	always	*
A4xx	4xx UserAlarmxx	**Disappearing user alarm**	-	-	-

* Alarmlevel is set in the application program

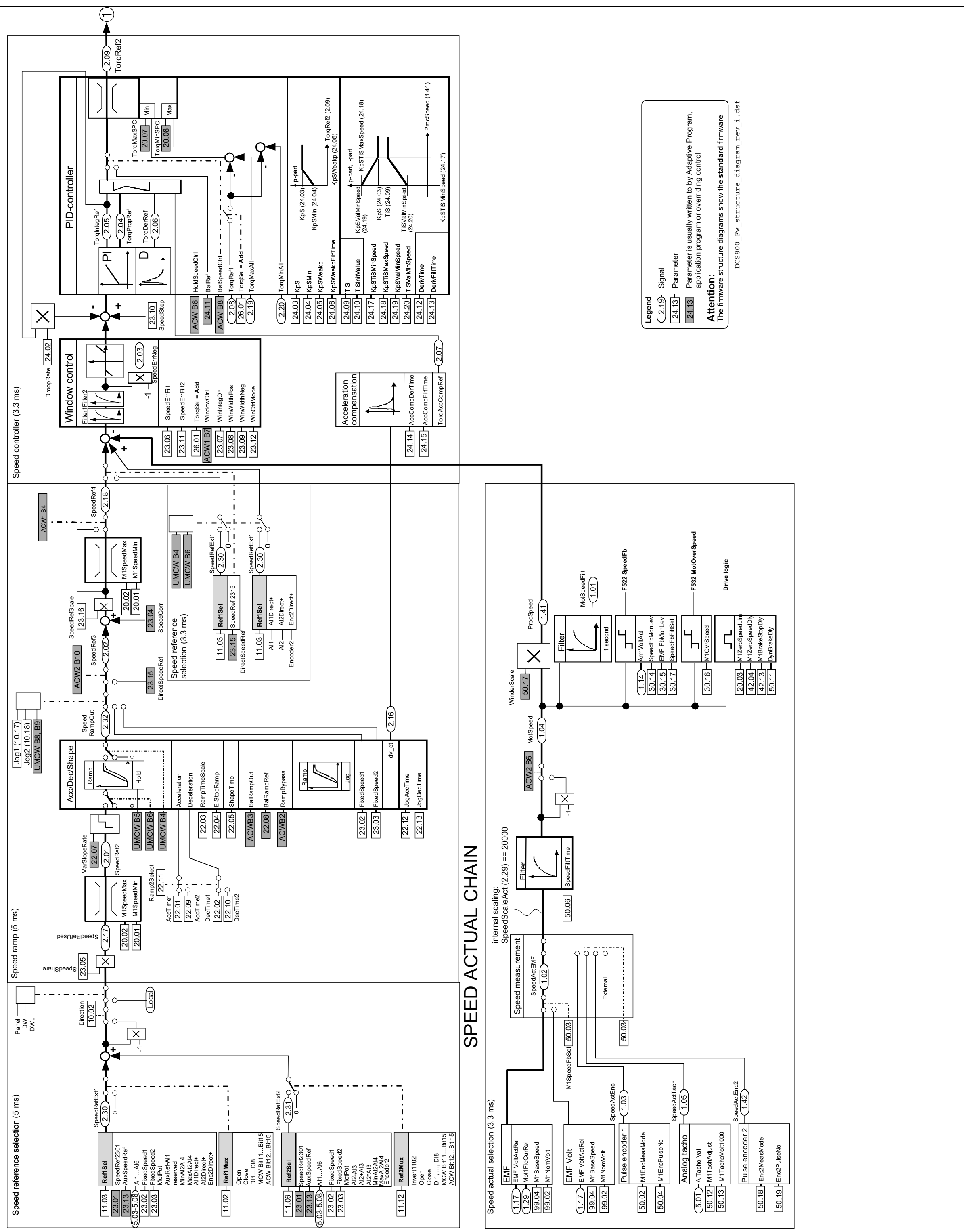
Notices

A notice is a message to inform the user about a specific occurrence which happened to the drive.

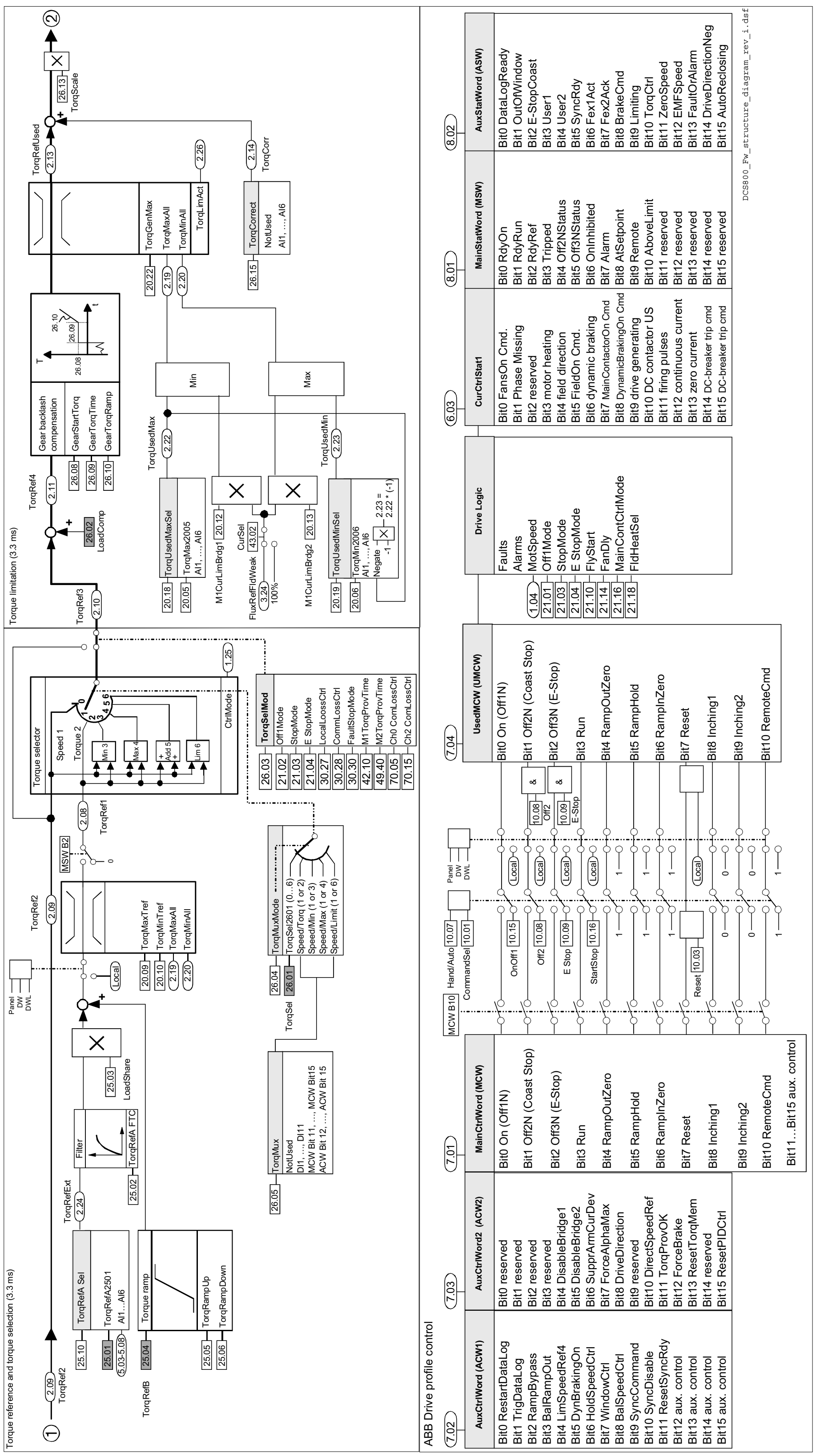
Text on DCS800 Control Panel	Definition / Action
718 PowerUp	**Energize electronics:** The auxiliary voltage for the drives electronics is switched on
719 FaultReset	**Reset:** Reset of all faults which can be acknowledged
801 APNotice1	**User defined notice by Adaptive Program**
802 APNotice2	**User defined notice by Adaptive Program**
803 APNotice3	**User defined notice by Adaptive Program**
804 APNotice4	**User defined notice by Adaptive Program**
805 APNotice5	**User defined notice by Adaptive Program**
AccessDenied	**Access to Memory Card:** Access to Memory Card is denied, due to another access
ParNoCyc	**Cyclic parameters:** A non cyclical parameter is written to (e.g. the overriding control writes cyclical on a non cyclical parameter). The parameters causing the notice can be identified in *Diagnosis (9.11)*.
PrgInvMode	**Adaptive Program not in Edit mode:** Push or Delete action while the Adaptive Program is not in **Edit** mode Check: – *EditCmd (83.02)* – *AdapProgCmd (83.01)*
PrgFault	**Adaptive Program faulty:** Adaptive Program faulty Check: – *FaultedPar (84.02)*
PrgProtected	**Adaptive Program protected:** Adaptive Program is protected by password and cannot be edited Check: – *PassCode (83.05)*
PrgPassword	**Adaptive Program wrong password:** Wrong password is used to unlock the Adaptive Program Check: – *PassCode (83.05)*
FB found	**R-type fieldbus adapter found:** R-type fieldbus adapter found
Modbus found	**R-type Modbus adapter found:** R-type Modbus adapter found
COM8 found	**SDCS-COM-8 found:** Communication board SDCS-COM-8 found
AIO found	**Analog extension module found:** Analog extension module connected to SDCS-CON-4 or SDCS-COM-8 found
DIO found	**Digital extension module found:** Digital extension module connected to SDCS-CON-4 or SDCS-COM-8 found
Enc found	**Encoder module found:** Encoder module (RTAC-01 or RTAC-03) connected to SDCS-CON-4 or SDCS-COM-8 found
Resolv found	**Resolver module found:** Resolver module (RRIA-01) connected to SDCS-CON-4 or SDCS-COM-8 found

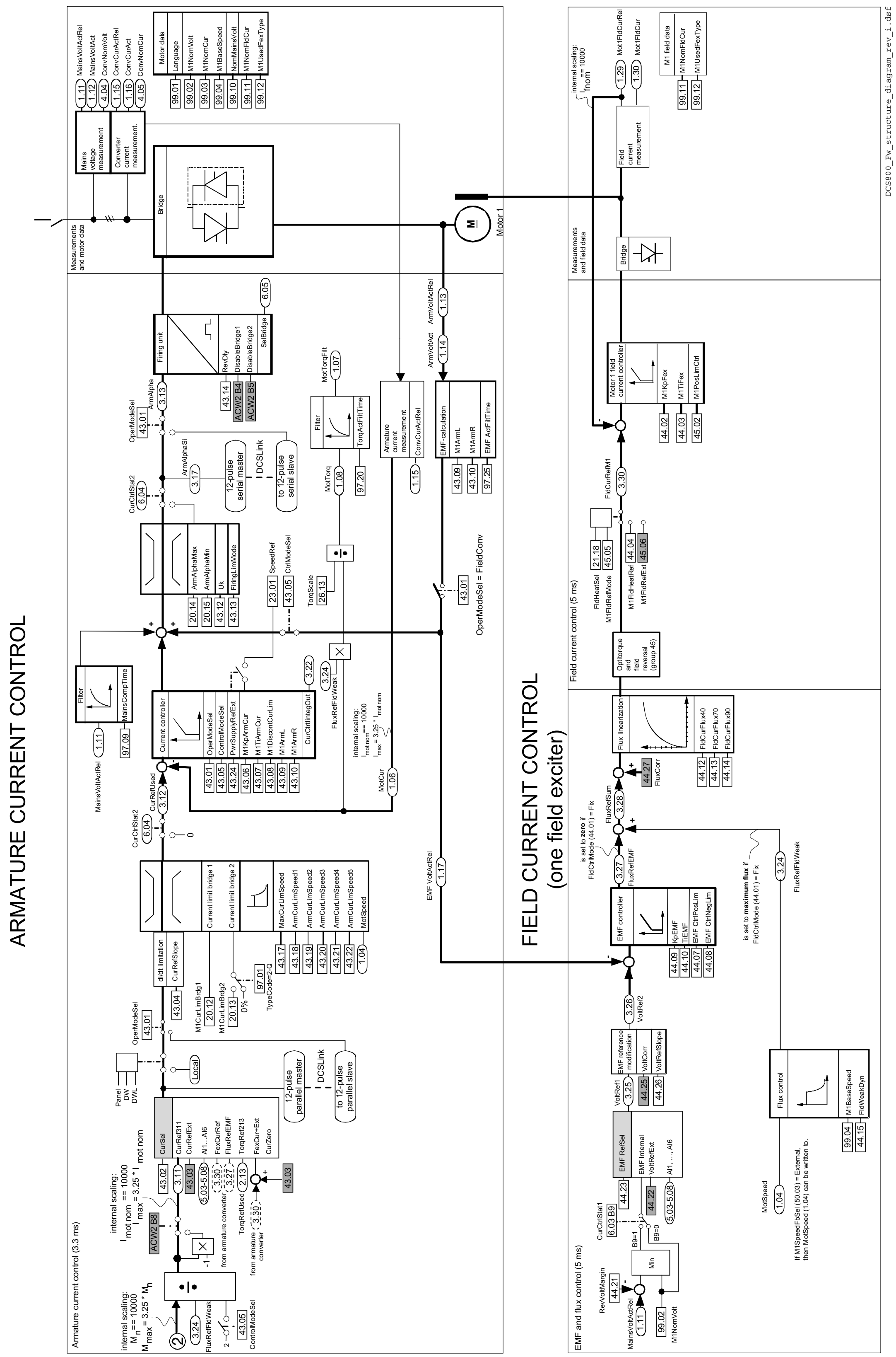
Text on DCS800 Control Panel	Definition / Action
DSL found	**SDCS-DSL-4 found:** DCSLink board found
Drive not responding	**Drive not responding:** The communication between drive and DCS800 Control Panel was not established or was interrupted. Check: – Change the DCS800 Control Panel – Change the cable / connector which is used to connect the DCS800 Control Panel to the SDCS-CON-4 – Change the SDCS-CON-4 – Change the SDCS-PIN-4

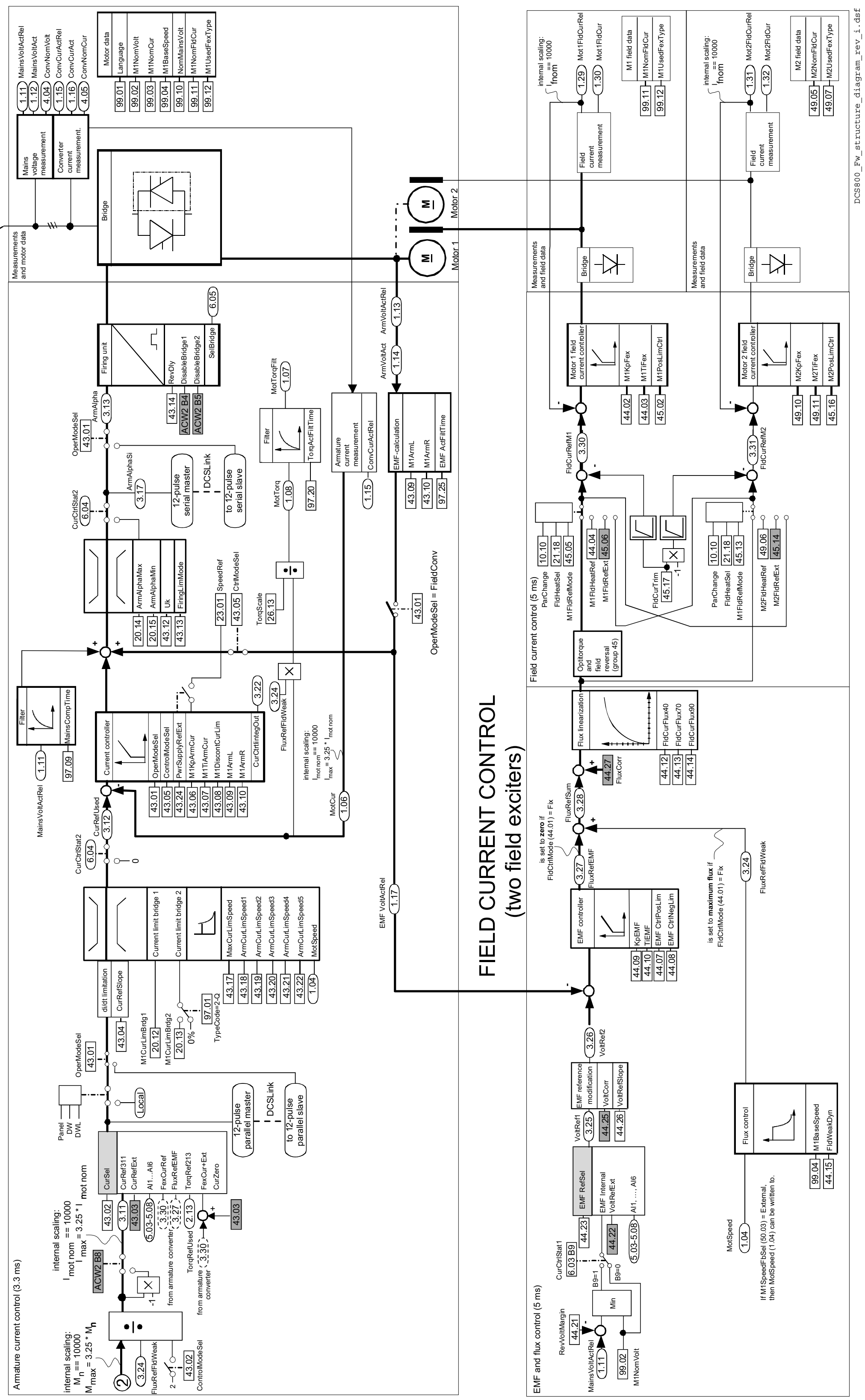
Appendix A: Firmware structure diagrams



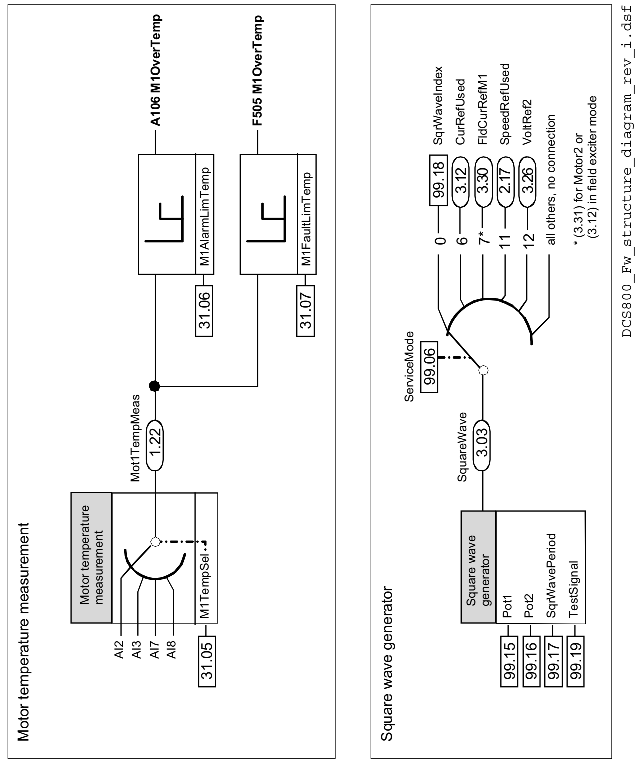
TORQUE CONTROL CHAIN




ARMATURE CURRENT CONTROL
Armature current control (3.3 ms)
Measurements and motor data
Mains voltage measurement
Converter current measurement
Motor data
Bridge
Motor 1
Firing unit
Current controller
Current limit bridge 1
Current limit bridge 2
di/dt limitation
CurSel
Filter
Armature current measurement
EMF-calculation
FIELD CURRENT CONTROL
(one field exciter)
EMF and flux control (5 ms)
Field current control (5 ms)
Measurements and field data
EMF reference modification
EMF RefSel
EMF controller
Flux linearization
Flux control
Motor 1 field current controller
Field current measurement
M1 field data
Optitorque and field reversal (group 45)
OperModeSel = FieldConv
DCS800_Fw_structure_diagram_rev_i.dsf

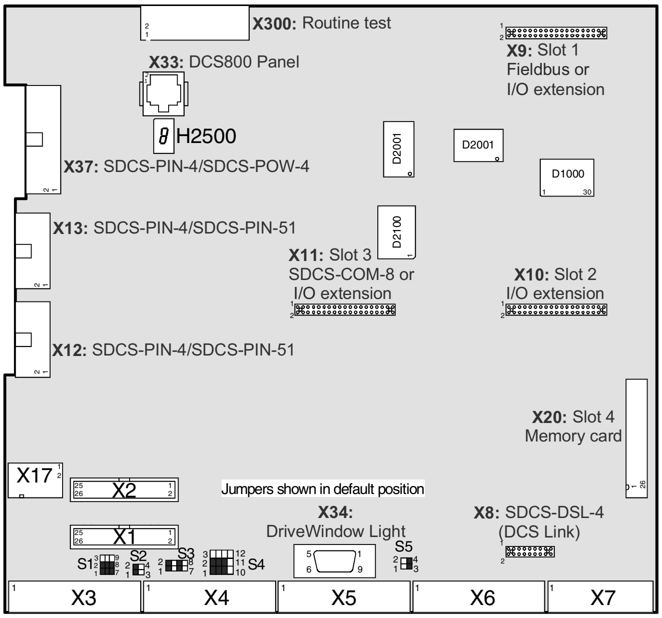

ARMATURE CURRENT CONTROL
FIELD CURRENT CONTROL
(two field exciters)
Armature current control (3.3 ms)
Measurements and motor data
EMF and flux control (5 ms)
Field current control (5 ms)
Motor 1
Motor 2
DCS800_Fw_structure_diagram_rev_i.dsf

ADDITIONAL FUNCTIONS

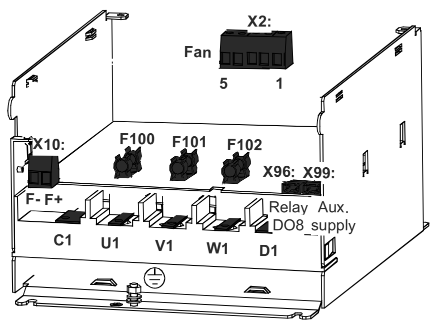


Appendix B: SDCS-CON-4 Terminal Allocation

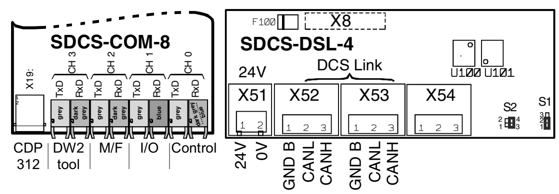
SDCS-CON-4 Connector allocation



DCS800 module
Terminal allocation



DCS800 Accessories



SDCS-CON-4: Terminal allocation

Terminal	X3 Tacho and AI	X4 AI and AO	X5 Encoder	X6 DI	X7 DO
1	90...270V	AI4-	+A	DI1	DO1
2	30...90V	AI4+	-A	DI2	DO2
3	8...30V	0V	+B	DI3	DO3
4	0V	+10V	-B	DI4	DO4
5	AI1-	-10V	+Z	DI5	DO5
6	AI1+	0V	-Z	DI6	DO6
7	AI2-	AO1	GND	DI7	DO7
8	AI2+	AO2	Sense GND	DI8	0V
9	AI3-	AO3 (Iact)	Sense Us	+24	
10	AI3+	0V	Us	0V	

DCS800 terminal alloc_a.dsf

Appendix C: Index of signals and parameters

Index of signals and parameters (alphabetic order)

DCS800 family

DCS800-S modules
The versatile drive for any application

20 … 5,200 A_{DC}
0 … 1,160 V_{DC}
230 … 1,000 V_{AC}
IP00

- Compact
- Highest power ability
- Simple operation
- Comfortable assistants, e.g. for commissioning or fault tracing
- Scalable to all applications
- Free programmable by means of integrated IEC61131-PLC

DCS800-A enclosed converters
Complete drive solutions

20 … 20,000 A_{DC}
0 … 1,500 V_{DC}
230 … 1,200 V_{AC}
IP21 – IP54

- Individually adaptable to customer requirements
- User-defined accessories like external PLC or automation systems can be included
- High power solutions in 6- and 12-pulse up to 20,000 A, 1,500 V
- In accordance to usual standards
- Individually factory load tested
- Detailed documentation

DCS800-E series
Pre-assembled drive-kits

20 … 2,000 A_{DC}
0 … 700 V_{DC}
230 … 600 V_{AC}
IP00

- DCS800 module with all necessary accessories mounted and fully cabled on a panel
- Very fast installation and commissioning
- Squeezes shut-down-times in revamp projects to a minimum
- Fits into Rittal cabinets
- Compact version up to 450 A and Vario version up to 2,000 A

DCS800-R Rebuild Kit
Digital control-kit for existing powerstacks

20 … 20,000 A_{DC}
0 … 1,160 V_{DC}
230 … 1,200 V_{AC}
IP00

- Proven long life components are re-used, such as power stacks, (main) contactors, cabinets and cabling / busbars, cooling systems
- Use of up-to-date communication facilities
- Increase of production and quality
- Very cost-effective solution
- Open Rebuild Kits for nearly all existing DC drives
- tailor-made solutions for…
  - BBC PxD
  - BBC SZxD
  - ASEA TYRAK
  - other manufacturers

ABB Automation Products
Wallstadter-Straße 59
68526 Ladenburg • Germany
Tel: +49 (0) 6203-71-0
Fax: +49 (0) 6203-71-76 09
www.abb.com/motors&drives

Ident. No.: 3ADW000193 R0701 Rev G
01_2011

193R0701A1030000