Gold Line

Command Reference for Gold Line Drives

February 2013 (Ver. 1.406)

Elmo Motion Control

www.elmomc.com

Notice

This guide is delivered subject to the following conditions and restrictions:

- This guide contains proprietary information belonging to Elmo Motion Control Ltd. Such information is supplied solely for the purpose of assisting users of the Gold Line technology.
- The text and graphics included in this manual are for the purpose of illustration and reference only. The specifications on which they are based are subject to change without notice.
- Information in this document is subject to change without notice.

Elmo Motion Control and the Elmo Motion Control logo are registered trademarks of Elmo Motion Control Ltd.

EtherCAT Conformance Tested. EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

Revision History

Version	Date	Details
Ver. 1.0		Initial version
Ver. 1.1		Following commands added: **AD**, **AR**, **EA**, **GO**, **GP**, **GS**, **GV**, **GW**, **PC**, **SE**, **SO**
Ver. 1.2	Mar 2012	Following commands updated or added: **AA, AC, AG, AS, BP, CA, CD, CL, CP, DC, DF, DL, EA, EE[], GI[], GO, IB, IF, IP, JV, KR, LD, MR, MS, OB[N], OC[N], OL, PB, RP, RR, RS, SV, TW, VH_VL, VP, WS, XQ**
Ver. 1.3	May 2012	Following commands added: **JP**, **NF**, **PO**, **PU** Following commands updated: **AA**, **BG**, **CD**, **DC**, **FC**, **FF**, **FX**, **GI**, **IL**, **IP**, **JV**, **KI**, **KP**, **MF**, **MO/SO**, **OV**, **PA**, **PL**, **PR**, **PX**, **SC**, **SD**, **SP**, **VE**, **VH[]/VL[]**, **VX**, **XA**, **XM**
Ver. 1.401	Oct 2012	Following commands added: **RM**, **VU, US, HT** Following commands updated: **BS**, **EE**, **FC**, **KV**, **PU**, **PX**, **VX**, **XM[]** Following commands updated: **AA[], EE[], GO, HF/HM, IB[], IL[],IP, MF, OC, WS, CA, EC, EE, FF, XA, XP**
Ver.1.402	Oct 2012	Correction to **GO**
Ver. 1.403	Nov 2012	Corrections to **EE**, **CA**, **EA**
Ver.1.404	Dec 2012	Addition of ECAM commands **DV[]**, **EI**, **EM**, **ET**, **IL**, **PY**, **RM**, **YM** Addition to command **OV** Corrections to **BS**, **EE**, **EM**, **IL**, **KV**, **MF**, **RP**, **XM**, **YM**
Ver. 1.405	Jan 2013	Corrections to **EE**, **GS**, **KV**, **NF**, **NT**, **TW**
Ver. 1.406	Feb 2013	Correction to **NF**

Elmo Worldwide

Head Office

Elmo Motion Control Ltd.

60 Amal St., P.O. Box 3078, Petach Tikva 49516
Israel

Tel: +972 (3) 929-2300 • Fax: +972 (3) 929-2322 • info-il@elmomc.com

North America

Elmo Motion Control Inc.

42 Technology Way, Nashua, NH 03060
USA

Tel: +1 (603) 821-9979 • Fax: +1 (603) 821-9943 • info-us@elmomc.com

Europe

Elmo Motion Control GmbH

Hermann-Schwer-Strasse 3, 78048 VS-Villingen
Germany

Tel: +49 (0) 7721-944 7120 • Fax: +49 (0) 7721-944 7130 • info-de@elmomc.com

China

Elmo Motion Control Technology (Shanghai) Co. Ltd.

Room 1414, Huawen Plaza, No. 999 Zhongshan West Road, Shanghai (200051)
China

Tel: +86-21-32516651 • Fax: +86-21-32516652 • info-asia@elmomc.com

Asia Pacific

Elmo Motion Control

#807, Kofomo Tower, 16-3, Sunae-dong, Bundang-gu, Seongnam-si, Gyeonggi-do,
South Korea

Tel: +82-31-698-2010 • Fax: +82-31-698-2013 • info-asia@elmomc.com

Table of Contents

Description of Attributes

For indexed commands:

Attribute	Description
Type	The data type of the variable (integer, float, bit field etc.) and the access mode (read-only, read/write or command).
Source	The entry source: RS232, USB, TCP, EoE, CoE or CANopen *Mapping* means that the object can be mapped to a process data object (PDO). *User Program* means that the variable can be manipulated from an Elmo user program. *FoE* in some commands means that File Over EtherCAT can be used.
Restrictions	Commands might be limited to specific conditions. These limits should be described in this attribute. The reason for a restriction may be a safety consideration, consistency with other commands or relevance in a specific context or product model. For example: *Not User Program* means that the command cannot be used in a User Program. *Motor Must Be On* indicates that the command can only be executed if the servo is enabled.
Range	Indicates the maximum and minimum permissible values for the specific commands. In some cases the command alters CANopen (CoE) objects which are user units, and conflicts may occur when the resulting value is out of range. This may happen when the user sets a factor that multiplies the value to an "out of range" value.
Index range	For indexed commands that have several inputs, there are two cases: 1) Inputs with the same meaning and weight. For these commands all entries have the same meaning and are described as scalar commands (e.g., **ZX[*N*]**, **ET[*N*]**). 2) Inputs with different meanings. Inputs may have different meanings for different uses. In this manner even the context might differ. The manual describes each input as a specific command (e.g., **CA[*N*]**, **SC[*N*]**). In this case an entry description that details every input is added (see below). In some cases, when the specific feature which is normally used by the array is not defined, the array memory space may be used as a buffer for other needs. These cases are described in the relevant commands (see, for example, **HM[*N*]** (captured mode))

Default	The default value after the **RS** command.
Unit modes	**UM**. The relevant controller which can be used for the specific command/parameter.
Non- Volatile	*Yes* means that the information is saved to flash memory after the **SV** command (or CANopen object 0x1010). *No* means that the information is not saved.

Remarks

An indexed command which has a different meaning for every input will include the following table with a description of each entry:

Index	Description	Type	Values	Restrictions
Number of input	Command description	Similar to the **Type** attribute	Range or any other value description, as is applicable to the command	Similar to the **Restriction** attribute

Constants

MAX_POSITION_RANGE	2,147,483,647
MIN_POSITION_RANGE	-2,147,483,648
MAX_VELOCITY	2,000,000,000
MAX_ACCELERATION	2,000,000,000
MAX_CURRENT	Value of **MC**

Terms

AA[*N*] – Communication Server Commands

The **AA[]** command addresses the communication server (ATMEL).

CANopen/CoE

Attributes

Attribute	Description
Type	General use, integer
Source	All, except RS232 and the user program from page 4
Restrictions	According to the entry description
Range	According to the entry description
Index range	1 to 99
Unit modes	All
Non-volatile	According to entry description
Attribute	None

Remarks

Some of the **AA[*N*]** entries (indices) are for internal use. These entries are basically protected, and modifying these entries may harm the functionality of the drive.

The **AA[]** command is handled in the communication server and should not be addressed to the digital signal processor (DSP), which is the main processor. Thus, limited interpretation is available. For example, unlike other commands, the **AA[]** command is case-sensitive and responds only to its upper-case name. If the command generates an error, the reply is 19? The number 19 is the error code for a command syntax error.

The **AA[]** command is mainly for internal use and includes very limited interpretation abilities.

When an error is returned (19?;) the EC should not be influenced.

Indices

The following table describes the **AA[*N*]** entries. Index values which are not indicated are for internal use only.

<table>
<tr><th>Index</th><th>Description</th><th>Type</th><th colspan="2">Possible Values</th><th>Restrictions</th></tr>
<tr><td rowspan="5">1</td><td rowspan="5">Drive FW status</td><td rowspan="5">Bit field</td><td>Bit</td><td>Description</td><td rowspan="5">Read-only</td></tr>
<tr><td>0</td><td>ATMEL in boot</td></tr>
<tr><td>1</td><td>ATMEL in FW</td></tr>
<tr><td>2</td><td>DSP in boot</td></tr>
<tr><td>3</td><td>DSP in FW</td></tr>
<tr><td>2</td><td>MAC address</td><td>String</td><td colspan="2">Depends on the drive HW</td><td>Read-only</td></tr>
<tr><td rowspan="2">4</td><td rowspan="2">Ethernet/EtherCAT selection. Set by the factory.</td><td rowspan="2">String</td><td>0</td><td>Drive supports Ethernet only</td><td rowspan="2">Read-only</td></tr>
<tr><td>1</td><td>Drive supports EtherCAT and Ethernet</td></tr>
<tr><td>5</td><td>ATMEL version string</td><td>String</td><td colspan="2">(for example)
In the boot state:
ATMEL Boot 1.1.1.0 ([compilation date])
In the FW state:
ComServer 1.1.1.0 ([compilation date])</td><td>Read-only</td></tr>
<tr><td>6</td><td>Drive serial number</td><td>String</td><td colspan="2">Same value as the SN[4] command</td><td>Read-only</td></tr>
</table>

8	ATMEL general status bits	Bit field	Bit	Value	Description
			0	0	EtherCAT HW init OK
				1	EtherCAT HW failed to initiate.
			1	0	EtherCAT SW init OK
				1	EtherCAT SW failed to initiate.
			2	0	ATMEL flash memory size is 512.
				1	ATMEL flash memory size is 256.
			3	0	Communication between the DSP & ATMEL is synchronized.
				1	Communication between the DSP & ATMEL is not synchronized. Messages are not transferred between the CPUs.
			4	0	Drive is configured for Ethernet communication.
				1	Drive is configured to EtherCAT communication.
			5	0	EtherCAT chip is ready for use.
				1	EtherCAT chip failed to initiate.
			6 to 31		Reserved
10	IP address set	String	IP address set by the host. The IP address has the following format: xx.xx.xx.xx The actually used IP address can be retrieved by calling the command with Index 20. If DHCP or EoE is used, the actual IP address might differ from this address.		Read/Write Saved to flash memory

<table>
<tr><td>11</td><td>Subnet mask set</td><td>String</td><td colspan="2">Net mask set by the host.
The Net mask has the following format: xx.xx.xx.xx
The actually used Net mask address can be retrieved by calling the command with Index 21.</td><td>Read/Write
Saved to flash memory</td></tr>
<tr><td>12</td><td>Gateway set
Gateway set by the host</td><td>String</td><td colspan="2">The gateway has the following format: xx.xx.xx.xx</td><td>Read/Write
Saved to flash memory
The actual gateway can be retrieved by calling the command with Index 22.</td></tr>
<tr><td rowspan="2">13</td><td rowspan="2">EtherCAT/Ethernet switching</td><td rowspan="2">String</td><td>42330</td><td>Ethernet</td><td rowspan="2">Read/Write
Saved to flash memory
Note: The change takes effect after power-up.
All values, except 1 and 42330, are reserved, but will be treated as 1.</td></tr>
<tr><td>1</td><td>EtherCAT (only if AA[4] supports EtherCAT)</td></tr>
<tr><td rowspan="3">14</td><td rowspan="3">DHCP enable/disable
If Ethernet is selected, this command enables or disables the use of DHCP to configure the Ethernet parameters.</td><td rowspan="3">String</td><td>0</td><td>DHCP disabled (default)</td><td rowspan="3">Read/Write
Saved to flash memory</td></tr>
<tr><td>1</td><td>DHCP enabled</td></tr>
<tr><td colspan="2">Note: The change takes effect after power-up.</td></tr>
</table>

15	ATMEL boot version	String	Format: AtmelBoot 1.1.1.0 ([compilation date][CPU flash memory size])	Read-only
20	Actual IP address	String	Format: xx.xx.xx.xx	Read-only
21	Actual Net mask	String	Format: xx.xx.xx.xx	Read-only
22	Actual gateway	String	Format: xx.xx.xx.xx	Read-only
23	Actual MAC address	String	Format: XX:XX:XX:XX:XX:XX	Read-only Note: In some cases in addition to ':', '.' and '-' are used.
25	ATMEL mini-boot version	String	Format: MiniBoot 1.0.0.1: ([compilation date])	Read-only
27	EtherCAT Logical ID switches	String	Hexadecimal format: 0xYZ Y – MSB Switch 4-bit hexadecimal value. Z – LSB Switch 4-bit hexadecimal value. **Note:** Evaluated at power-up only.	Read-only
29	Burnt in PAL description	String		Read-only
30	User defined EtherCAT Switch-ID	String	Decimal format: 0 - 65535	Read/Write
31	Product information (same as **WS[30]**)	String	Hexadecimal format: To write lead with password: "Elmo". Example: Elmo0x1400A.	Read/Write Saved on flash memory
			Bit / **Description** 0 to 7 / Product Name 8 to 11 / Reserved for product name	

			12 to 13	Always 0	
			14	Project (always 1 for NG (Gold Line))	
			15	Always 0	
			16	CAN integrated	
			17	0: EtherCAT 1: TCP\IP	
			18 to 20	Feedback type: 0: E type (Encoder + Encoder, Analog sensor) 1: A type (Absolute + Encoder, Analog sensor) 2: R type (Encoder, Analog sensor + Resolver) 3: M type (Absolute + Resolver)	
			21	Define R type drive: Current saturation stays on PL	
			22	EtherCAT ID switches	
			23	Reserved	
			24	Ethernet hardware: 0: Absent 1: Present	
			25 to 31	Reserved	

Indices 2, 3 and 7 are for internal use.

All indices that are not listed include the following logic:

- A Read operation will return 0.
- A Write operation will return an error, i.e., 19?; - "Command syntax error"

AC – Set Acceleration

AC specifies the maximum allowed profiler acceleration.

CANopen/CoE

0x6083

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	Effective on the next call to **BG**
Range	10 to 2e9
Default	1,000,000,000
Unit modes	**UM** = 5, **UM** = 2
Non-volatile	Yes

Remarks

The command defines the maximum allowed acceleration during the operation of point-to-point (**PA**, **PR**) and jog (**JV**) profilers.

The **AC** command does not affect the present motion. It takes effect only on the next call to Begin Motion (**BG**).

The **AC** value is fed to object 0x6083 during power-up and when the Begin Motion (**BG**) command is used .

The **AC** command does not affect time-dependent motion, such as Interpolated Position or Cyclic Synchronous Mode (See DS-402 manual).

The acceleration and deceleration of the drive are subject to the limits of the **SD** value. If the **AC** value is higher than the **SD** value, the **SD** value is used, and the **AC** value is ignored.

The **AC** value can be given in user-defined units specified by the **FC** command.

References

DC, **SD**, **BG**, **PA**, **PR**, **JV**, **FC**

AD[] – Set Analog Dead Band

AD[] specifies the dead band set for analog input 1.

CANopen/CoE

N/A

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Range	±10V
Index range	1 , 2
Default	**AD[1]** = 0 **AD[2]** = 0
Unit modes	All
Non-volatile	Yes

Remarks

The **AD[]** command sets the dead band of the analog input in the range from **AD[1]** to **AD[2]**.

While the analog input is in the dead band, the reading is ignored.

The sensor reading output slope will start from 0 at the dead band. For example, if the input is greater than **AD[2]**, then, `Reading=(input-AD[2])*AG[1]`.

AD[] does not affect analog input 2.

Indices

The following table describes the available options for **AD[]**.

Index	Description	Type	Units	Restrictions
1	Negative dead band	Float	Volts	
2	Positive dead band	Float	Volts	

References

AS[], AR[], AG[]

AG[] – Set Analog Gain

AG[] specifies the analog gain set for analog input 1.

CANopen/CoE

N/A

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Range	The total command (`AN[1]*AG[1]`) cannot exceed the applicable reference limit: Current (**AR[1]** = 1): ± **MC** Velocity (**AR[1]** = 2): ±**VH[2]** Position (**AR[1]** = 3): **VL[3]** to **VH[3]** Please refer to the applicable commands for more details.
Index range	[1]
Default	**AG[1]** = 0.1000000
Unit modes	All
Non-volatile	Yes

Remarks

The **AG[]** command sets the conversion gain for converting the input voltage to the specific units that are used (according to **AR[]**).

The conversion gain can be set to a current, velocity or position reference.

For example, if analog input 1 is used for velocity (**AR[1]** = 2), the value set by **AG[1]** is the unit conversion factor for converting from voltage to velocity in counts/sec. In this example if the value set by **AG[1]** is 1,000, this means that every 1 volt in the input is applied as a velocity command of 1,000 counts/sec.

The analog input is read by the **AN[]** command every real time as a general-purpose input. In order to map the analog reading as a reference signal, a socket must be used. Please refer to **CA[41]**…**CA[44]** for the functional mapping of the sockets. The analog input identification number is 16 (see example below).

After the socket is selected, **AR[]** is used to determine the usage of the analog input. Please refer to **AR[]** for more details.

For example, to set analog input 1 as a current reference, the following should be done:

1. Select a free socket (e.g., socket 4).
2. Disable the motor.

   ```
   MO=0
   ```
3. Set the drive to current loop.

   ```
   UM=1
   ```
4. Map socket 4 to an analog input reference.

   ```
   CA[44]=16
   ```
5. Specify that the units of analog input 1 are for a current (torque) reference.

   ```
   AR[1]=1
   ```
6. Specify that every 1 volt reading of the analog input will produce a 0.1 ampere reference command.

   ```
   AG[1]=0.1
   ```

The direction of the reference can be modified by changing the sign of **AG[]** from minus (-) to plus (+) and vice versa.

In order to use analog input 1 as a reference, the selected socket should be set to 16.

When used as a reference, the analog entry can be filtered using **KV[71]** to **KV[75]**.

Indices

The following table describes the operation options for **AG[]**:

Index	Description	Type	Units	Restrictions
1	Gain of analog input 1	Float	Defined units/volt	

References

AD[], AR[], AS[], CA[], KV[]

AN[] – Get Analog Input

AN[] reads values from the drive's analog inputs.

CANopen/CoE

Text

Attributes

Attribute	Description
Type	Float, Read-only
Source	All
Restrictions	None
Range	See the table below.
Index range	1 to 6
Default	N/A
Unit modes	All
Non-volatile	No

Remarks

The **AN[]** command reads analog values from the drive's analog-to-digital (A2D) converter. The values are converted to user units according to the table below.

Indices

The following table describes the **AN[]** entries.

Index	Description	Units	Values	Restrictions
1	Reads analog input 1. When **RM** = 1, the value is used as an auxiliary reference for a motion command (refer to **AG[]**).	Volts	+/- 10	
2	Reserved			
3	Reads the instantaneous current feedback from motor phase A	Amperes	N/A	The range of values depends on the **MC** of the drive.

Index	Description	Units	Values	Restrictions
4	Reads the instantaneous current feedback from motor phase B	Amperes	N/A	The range of values depends on **MC** of the drive
5	Reads the instantaneous current feedback from motor phase C	Amperes	N/A	The range of values depends on the **MC** of the drive
6	Reads the bus voltage	Volts	N/A	The range of values depends on the actual power voltage supplied to the drive

References

AG[], **AS[]**, **BV**, **MC**

AR[] – Set Analog Units

AR[] specifies the function of the analog input when used as a reference command.

CANopen/CoE

N/A

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Range	1 to 3
Index range	[1]
Default	**AR[1]** = 3
Unit modes	All
Non-volatile	Yes

Remarks

AR[*N*] determines the mapping of the analog input N to either Current, Velocity or Position reference. This is used when the analog input serves as an auxiliary command or feedback for one of the mentioned references.

After the socket number (refer to the **CA[41]** to **CA[44]** commands) is determined, **AR[]** is used to select the reference mode. Please refer to the **AG[]** command for an example.

The analog input can be served as a feedback for Tachometer and Potentiometer readings. In this case the selected socket should be defined by **CA[45]** or **CA[46]**, respectively.

Indices

The units which are used by the reference generator are according to the value of **AR[]** as follows:

Index	Description	Type	Units		Restrictions
1	Units of analog input #1	Integer	1	Ampere (for current)	
			2	Counts/sec (for velocity)	
			3	Counts (for position)	
2	Reserved				

References

AD[], **AG[]**, **AS[]**,**CA[]**

AS[*N*] – Analog Input Offset

AS[] compensates the offset of analog input *N*.

CANopen/CoE

Attributes

Attribute	Description
Type	Float
Source	All
Restrictions	None
Range	According to the table
Index range	[1]
Default	0

Remarks

The analog input of the drive is offset during production of the drive. However, in some cases the reference input is not adjusted to the input stage of the host.

The **AS[]** command should be tuned so that when a zero voltage is applied to analog input *N*, a measurement will also output a zero voltage.

A typical method for setting the offset value is to set **AS[1]** to zero, set the input to 0 V (i.e., short to ground), get the value of **AN[1]** several tens of times and average the readings.

Analog input 1 can also be recorded using the EAS recorder. The signal is "Analog Input 1".

Note that both the recorder signal and the **AN[1]** value includes the values of **AS[1]** and **AD[]**.

The Gold drives include a single analog reference.

Indices

The following table describes the analog input 1 entry:

Index	Description	Type	Values	Restrictions
1	Offset value of input 1	Float	+/- 10.0 volts	
2	Reserved			

References

AD[], **AG[]**, **AN[]**, **AR[]**

BG – Begin Motion

BG starts the next profiled motion.

CANopen/CoE

Attributes

Attribute	Description
Type	Command
Source	All
Restrictions	N/A
Range	N/A
Default	N/A
Unit modes	**UM** = 5, **UM** = 3, **UM** = 2
Non-volatile	N/A

Remarks

The **BG** command is used to activate the next profiled motion.

On the **BG** command, the relevant motion target data (set point) is sent to the profiler, which then calculates the command for the control loop.

BG affects the present motion mode by modifying the profiler and/or controller which are used by the mode.

The "Motion Mode" is determined by the "Elmo Motion Command," which should be the last effective command for the presently required motion (see the table below).

The Actual Motion Mode is presented by CANopen object 0x6061 (it can also be retrieved using **OV[2]**).

The effect of **BG** on the motion mode is considered according to the next table.

Motion mode	UM value	Elmo motion command	DS-402 Motion (0x6061)	Relevant parameters considered
PTP	**UM** = 5, **UM** = 3	**PA**, **PR**	Profile Position, 1	**AC**, **DC**, **SP**, **SF**, **FS**
JV	**UM** = 2, **UM** = 5	**JV**	Profile Velocity, 3	**AC**, **DC**, **SF**
JP	**UM** = 5	**JP**	Profile Position, 1	**AC**, **DC**, **SF**

Note: UM is the minimum unit mode for the relevant motion. Refer to the **UM** command.

BG is also used to convert between DS-402 objects and Elmo's commands. For details, see the following table.

Elmo Command	Converts to CANopen Object	Action Performed	Note
AC	0x6083	Saturation to maximum acceleration	
DC	0x6084	Saturation to maximum acceleration	
JV	0x60FF	• Motion mode to: Profile Velocity • Saturation to maximum speed	
JP	0x6081 0x6082	• Motion mode to: Profile Position • Saturation to maximum speed	
-	0x607E	Set to 0 indicating: do not convert polarity	
SP	0x6081	Saturation to maximum speed	
FS	0x6082	Final speed at target position	
PA/PR	0x607A	Motion mode to: Profile Position	

Note: The command also affects the DS-402 control word (object 0x6040). The value of this object is determined according to the actual motion mode.

The **BG** command removes any pending Halt from DS-402.

BG should have no effect in torque modes or in a time-dependent mode, such as Synchronous Cyclic Position or Interpolated Position.

If the recorder is triggered by a Begin Motion command, **BG** will start the recording.

If the User Program includes an Auto_BG routine, the routine should be called automatically in the next cycle after **BG**.

References

PA, **JV**, **JP**, **SP**, **SF**, **FS**, **PO**

BH – Get Recorder Signal

BH uploads the values recorded by the recorder to a host in hexadecimal format.

CANopen / CoE

Attributes

Attribute	Description
Type	Command, read-only
Source	All, except FoE
Restrictions	• Valid recorded data is ready (i.e., RR == 0). • No other uploading sequence is performed (UL).
Range	1 to 2^{16}, bit field format
Default	No
Unit modes	All
Non-volatile	No

Remarks

BH is a bit-field command, where every bit points to a specific recorded signal (e.g., "Position") that can be uploaded. The signals are requested by using the **RC** command in conjunction with the **RV[]** command. **BH** can only be used after the recorder was successfully finished and the **RR** command, which launches and retrieves the status of the recorder, is equals to 0.

Please refer to **RR** command for further information about the recorder procedure.

The **BH** command is designed to optimize data transfer from the drive to the host while allowing fetching of the data in a simple Hex-Binary text format. The format transfers each number in two printable characters. For example, the value 10 (0x0A) is transmitted in two chars: 0x30 and 0x41.

The uploaded stream consists of two main parts:

- A data header
- A data value

The data header contains the following information about the recorded data.

Byte	Description
0 to 1	Variable type (0, integer; 1, float)
2 to 3	Variable size: (2, short; 4, long; 8, double)
4 to 7	Transmitted data length

Byte	Description
8 to 11	Data sampling time in TS multiplier (1, every TS; 2, every TSx2). The sampling intervals are equal to **RG** * sampling multiplier.
12 to 19	Floating factor to convert the data from internal units to physical units (not user units). Factors are used to convert the following from internal units: Current: (e.g., "Active Current") to amperes Velocity: (e.g., "Velocity Command") to counts/sec Voltage: (e.g., "Bus Voltage") to volts The user must multiply the data by the factor to obtain the actual value.
20 to transmit data length	Data of the specific uploaded signal

During the uploading of data, the drive can receive and execute other commands. However, the drive will not be able to reply the commands unless it is used in a communication channel that is different from the channel used by the **BH** request. For example, if the recorded data is fetched from the RS232 communication channel, the USB communication channel can still be used to execute motion commands while the RS232 channel is uploading the data.

Recorded data can only be fetched one at a time. This means that if **BH** is already in process, other **BH** commands cannot be executed by any other communication line.

During the uploading the following commands cannot be executed:

- A **PP[1]** command for engaging new communication parameters
- An **HM[]/HF[]** command when the recorder buffer is used as a position capture buffer
- **FT[]**,**RC**, **RG**, **RL**,**RP[]**,**RR** and **RV[]** commands are used during the recorder setting.

The recorder allows the uploading of global variables from the User Program as well. See the **RR** command and the User Program manual for further information.

References

FT[], **RC**, **RG**, **RL**, **RP[]**, **RR**, **RV[]**

BP[] – Brake Parameters

BP[] specifies the time parameters for the logical brake.

CANopen/CoE

N/A

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	One of the digital outputs must be defined as a brake using **OL[*N*]**.
Range	According to the table below
Index range	1, 2
Default	0
Unit modes	All
Non-volatile	Yes

Remarks

The drive allows the application to use a brake to hold the motor while the servo is off (SO == 0). **BP[]** is used to define the times that are needed to engage and disengage the brake.

The brake will be activated only if one of the digital outputs is defined as a brake by the **OL[*N*]** command.

OL[] also defines the logic level by which the output is activated. Normally, the hardware connection to the brake is such that when the drive is powered off the brake is engaged (current flows though the brake windings).

Any digital output can be used as a brake logic output. Output 1 (**OL[1]**) also supplies current for the brake. Please refer to the specific drive's Installation Guide for more information about the current source for this purpose.

Disabling the servo

When the servo is disabled by setting **MO** = 0, the brake is engaged, and the corresponding indication (**SO** = 0) is received only after the time set by **BP[1]**. Please refer to the **SO** command.

When a DS-402 state machine is used, Switch On, Ready to Switch On or Switch On Disable should be indicated by the status word only after the time needed to engage the brake has elapsed.

Enabling the servo

When the motor is enabled by setting **MO** = 1, the **SO** (Servo On) indication should be set to 1 only after **BP[2]** msec under the assumption that the brake was released. Please refer to **SO** command.

When a DS-402 state machine is used to enable the motor, the Status Word object (object 0x6041) should indicate Operation Enabled only after the brake is released. The host should consider this when the time-out is calculated.

Fault reaction

The brake output is activated immediately when a motor fault occurs (**MF** > 0). Both the Motor On and Servo On indications (**MO** and **SO**) are set to 0. In the case of an amplifier fault (i.e., Overvoltage, Overtemperature, Short Protection and Safety Active) there may be no drive controlling the servo before the brake is fully engaged.

Notes

- In cases in which the drive is in Stepper Mode (**UM** = 3) and **SC[8]** is used for automatic setting of the torque, the torque will be applied regardless of the state of the brake.
- The effect of **BP[2]** is considered only on the next motor on.
- The effect of **BP[1]** is considered only on the next motor off.
- The resolution for the brake output response is 250 µs.

Indices

Outputs are logically set/reset according to the following table:

Index	Description	Values	Restrictions
1	The delay time that is needed to engage the brake before the motor is actually stopped. The delay time set by **BP[1]** is the time between the request to disable the motor (a change from **MO** = 1 to **MO** = 0) and the actual time when the servo is off.	0 to 1000 milliseconds	
2	The delay time that is needed to release the brake before the motor is actually started. The delay time set by **BP[2]** is the time between the request to enable the motor (a change from **MO** = 0 to **MO** = 1) and the actual time when the servo is on and the profiler can actually be used. During this time profiler references (software set points & auxiliary) are ignored. If auto-phasing commutation is required, it should be activated after the time set by **BP[2]**.	0 to 1000 milliseconds	

References

OL[], **OP**

BS[] – Bring Recorded Sample

BS[*N*] obtains the (*N* + 1)th sample from a pre-selected recorded data vector.

CANopen/CoE

Attributes

Attribute	Description
Type	32-Bit (short, long or float), Read-only
Source	USB, TCP, EoE, RS232
Restrictions	Recorder ready (RR == 0); Valid vector selected (**RP[11]**)
Range	NA
Index range	1 to 16384
Default	NA
Unit modes	All
Non-volatile	No

Remarks

The **BS[]** command provides an interface which is used by the User Program to read variables previously recorded by the Recorder. **BS[]** returns a specified sample from a pre-selected recorded vector. The selected vector, from which the recorded signal samples are retrieved, is specified by **RP[11]** command.

The **BS[]** command always returns a long variable. The variable can be an integer or float type. In the case of a short variable, the upper bits are padded with zeros.

An index range that starts from 0 allows simpler modulo operations.

N can have a maximum value of 16384 (the longest possible vector). However, *N* also depends on the number of recorded points. Refer to the **RL** command for more details.

Indices

Outputs are logically set/reset according to the following table:

Index	Description	Type	Values	Restrictions
0	First sample	Long		
1	Second sample	Long		
…	…	…		
N	(*N* + 1)th sample	Long		

References

RR, **RL**, **RP[11]**

BT – Begin Motion On Time (Reserved)

BT

CANopen/CoE

Attributes

Attribute	Description
Type	
Source	
Restrictions	
Range	
Default	
Unit modes	
Non-volatile	

Remarks

References

BV – Bus Voltage

BV gets the maximum drive bus voltage in volts.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read-only
Source	All
Restrictions	None
Range	BV > 0 (The bus voltage depends on the drive and has no range.)
Default	200 V
Unit modes	All
Non-volatile	Constant

Remarks

The configured bus voltage is burned into each drive. The **BV** value is burned-in in Elmo during manufacturing and provides the voltage which is stated on the label plate.

Note that the factory undervoltage threshold (**WI[37]**)and the factory overvoltage threshold (**WI[35]**) are indicated in their specific parameters.

The actual voltage can be read by calling **AN[6]**.

The voltage thresholds can be modified by calling **XP[1]** for the overvoltage and **XP[13]** for the undervoltage

The actual values of the voltage thresholds can be read by calling **WI[36]** for the overvoltage and **WI[38]** for the undervoltage.

References

AN[], **WI[35]**, **WI[36]**, **WI[37]**, **WI[38]**, **XP[1]**, **XP[13]**

CA[*N*] – Commutation Array (Not Final)

CA[] sets the commutation and sensor feedback configuration.

CANopen/CoE

N/A

Attributes

Attribute	Description
Type	Integer
Source	All
Restrictions	According to array index
Range	According to array index
Index range	According to array index
Default	**CA[4]** = 1 **CA[5]** = 2 **CA[6]** = 3 **CA[13]** = 32765 **CA[17]** = 1 **CA[18]** = 4000 **CA[19]** = 2 **CA[22]** = -1 **CA[31]** = 10 **CA[32]** = 1 **CA[34]** = 1 **CA[35]** = 17 **CA[36]** = 25000000 **CA[41]** = 2 **CA[45]** = 1 **CA[46]** = 1 **CA[47]** = 1 **CA[48]** = 1863226 **CA[49]** = 18633

	CA[50] = 10000000 **CA[51]** = 10000000 **CA[65]** = 1 **CA[79]** = 1 **CA[84]** = 2
Unit modes	All
Non-volatile	Yes

Remarks

The **CA[]** command sets the socket functionalities, as well as the sensor and commutation parameters. Depending on the command index, the motor should be turned off and/or the commutation should be reset and recalculated on the next motor on.

Indices

The following table details the **CA[]** entries.

Note: The default is 0 unless otherwise stated.

Index	Description	Default	Values	Restrictions/Notes
1 to 3	Polarity of Hall sensors A, B and C, respectively 0 – Reverse the Hall sensor polarity 1 – Do not reverse the Hall sensor polarity		0, 1	The motor must be off to change the setting. Changing the setting resets commutation.
4 to 6	Correlate between Hall sensors and motor phases A, B and C, respectively	**CA[4]** = 1 **CA[5]** = 2 **CA[6]** = 3	1 to 3 (for Hall sensors A, B and C)	Each Hall sensor must be assigned to a different motor phase. The motor must be off to change command. Changing command resets commutation.
7	Phase offset relative to the absolute commutation sensor (Hall, Absolute Serial, Analog Hall and Resolver) in stepper units		-512 to 512	The motor must be off to change the setting. Changing the setting resets commutation.

8	Ignore the encoder error. Only valid for General Biss, Panasonic, Kawasaki, Yaskawa, Sanyo, EnDat, Tamagawa and SSI encoders.		0, 1	The motor must be off to change the setting. Changing the setting resets commutation.
9	Phase shift compensation in a Sin/Cos sensor. Units are the factor of the sine coefficient/2^{15}.			The motor must be off to change the setting. Changing the setting resets commutation.
10	Reserved			
11	Analog encoder sine signal offset in ADC units		0 to 4500	The motor must be off to change the setting. Changing the setting resets commutation.
12	Analog encoder cosine signal offset in ADC units		0 to 4500	The motor must be off to change the setting. Changing the setting resets commutation.
13	Analog encoder sine gain to fit the cosine signal. Units are $1/(2^{15})$	32765	10000 to 62000	Gain 1 is 32765. The motor must be off to change the setting. Changing the setting resets commutation.
14	Command Reference Multiplier (Position and Velocity) Multiplies the assigned socket command input. Socket reference function is according to **CA[68]** or **CA[69]**.		0 to 2147483647	
15	Command Reference Divider (Position and Velocity) Divides the assigned socket command input by $2^{CA[15]}$. Socket reference function is according to **CA[68]** or **CA[69]**.		0 to 30	

16	Force commutation search for every motor on 0 – Use commutation when known 1 – Force commutation search for every motor on		0 or 1	
17	Commutation method 1 – By Hall sensor 2 – By stepper, motor will move to a certain stepper position 3 – By binary search, minimal movement when commutation is not known 4 – By analog Hall sensor 5 – By serial absolute encoder 6 – By virtual absolute Gurley 7 – Slave including commutation by PAL	1	1 to 6	The motor must be off to change the setting. Changing the setting resets commutation.
18	Feedback counts per electrical cycles. The value is used to determine the number of counts per electrical cycles (**CA[19]**) of the commutation socket. **In rotary motion**: counts per revolution **In linear motion**: counts per single electrical cycle **In analog feedbacks**: the value obtained after multiplication by the factor read from **CA[31]**	4000	6 to 2147483647	The motor must be off to change the setting. Changing the setting resets commutation.

19	Number of motor pole pairs **In rotary**: pole pairs per revolution **In linear**: usually equal to one		Positive values	The motor must be off to change the setting. Changing the setting resets commutation.
20	Use Digital Halls 0 Do not read Hall sensors 1 Read Digital Hall sensors 2 Read Yaskawa Hall		0, 1, 2	The motor must be off to change the setting. Changing the setting resets commutation.
21	Gurley encoder resolution		10 to 12	The motor must be off to change the setting. Changing the setting resets commutation.
22	Number of the error bit in the SSI protocol bits, or -1 if no error bit exists.	-1	-1 to 29	The motor must be off to change the setting. Changing the setting resets commutation.
23	SSI error bit logic 0 High indicates an error 1 Low indicates an error		0, 1	The motor must be off to change the setting. Changing the setting resets commutation.
24	Hiperface's analog and absolute signals direction are: 0 Same direction 1 Opposite directions		0, 1	
25	Reverse the direction of stepper angle, equivalent to switching the motor cables between M2 and M3 0 Do not reverse 1 Reverse the direction		0, 1	The motor must be off to change the setting. Changing the setting resets commutation.
26	Socket number used for adding a current reference command		1 to 4	**Notes:** • Stop manager functions do not apply for this entry. • In situations where

				the brake is used (**BP[]**) the reference command might be active regardless of the brake state.
27	Socket number used for adding a Yaw current reference command		1 to 4	**Notes:** • Stop manager functions do not apply for this entry. • In situations where the brake is used (**BP[]**) the reference command might be active regardless of the brake state. • Only with Gantry master (sensor ID=20)
28	Motor type: 0 Rotary brushless 1 Rotary DC brush 2 Linear brushless 3 Linear voice coil 4 Rotary two-phase 5 Reserved 6 Linear two-phase		0 to 6	If **CA[28]** is set to 1 or 3, **UM** (unit mode) should not be 3 (stepper). The motor must be off to change the setting. Changing the setting resets commutation.
29–30	Reserved			
31	Analog signal multiplication factor. Each Sin/Cos cycle is equal to $2^{CA[31]}$ counts	10	2 to 16	The motor must be off to change the setting. Changing the setting resets commutation.
32	Number of resolver pole pairs per revolution	1		The motor must be off to change the setting. Changing the setting resets commutation.
33	Resolver excitation signal offset. The offset units are equal to the excitation signal clock units [150 MHz].		Positive value	The motor must be off to change the setting. Changing the setting resets commutation.

34	Resolver *N* cycle interpolation. The value determines the resolver position interpolation and the frequency of the excitation signal $f_E = f_{TS}/(2*N)$, or $f_E = f_{TS}$ for $N = 0$	1	0,1,2,4	The motor must be off to change the setting Changing the setting resets commutation
35	PAL glitch filter for absolute sensor reading. Glitch filter value is (CA[35]+1)*13.333 nanosecond	17	2 to 255	Need to reset the sensor by **CA[41** to **44]**
36	Clock frequency to PAL (Hz). PAL divide this value by 10	25000000	6250000, 12500000, 25000000	Need to reset the sensor by **CA[41** to **44]** Each sensor has its own range of frequency
37–40	Reserved			

41–44	ID of sensor connected to socket **CA[41]**: socket number 1 **CA[42]**: socket number 2 **CA[43]**: socket number 3 **CA[44]**: socket number 4		**CA[41]** = 2	1 to 26	Sensor ID is unique. The same sensor ID cannot be set for two sockets. If the socket is used for commutation, the motor must be off to change the setting, and changing the setting resets commutation.
	1	Quad Encoder Port B			
	2	Quad Encoder Port A			
	3	Analog Sin/Cos			
	4	Digital Hall only			
	5	Serial Absolute Biss			
	6	Serial Absolute Panasonic			
	7	Serial Absolute Mitutoyo			
	8	Virtual two sine signal			
	9	Serial Absolute EnDat			
	10	Serial Absolute Tamagawa			
	11	Pulse & Direction port B			
	12	Pulse & Direction port A			
	13	Quad encoder port B used for emulated feedback			
	14	Quad Port A used for emulated feedback			
	15	Copy profiler to socket			
	16	Analog Input #1			
	17	Virtual Absolute Gurley			
	18	Absolute SSI			
	19	Serial (absolute or incremental) Yaskawa			
	20	Gantry Master			
	21	Serial Exclusive			
	22	Resolver			
	23	Serial absolute Kawasaki			
	24	Serial Absolute General			

	Biss			
	25 Serial Absolute Sanyo			
	26 Simple profiler			
	27 Copy gantry differential position			
	28 Serial Hiperface			
45	Socket number used for position loop	1	1 to 4	The motor must be off to change the setting.
46	Socket number used for velocity loop	1	1 to 4	
47	Socket number used for commutation	1	1 to 4	The motor must be off to change the setting. Changing the setting resets commutation.
48	Maximum allowed amplitude for analog Sin/Cos encoder, in $(\text{ADC-offset})^2$	1863226	1863225 to 15669722	$(1\text{ V})^2$ to $(2.9\text{ V})^2$ The motor must be off to change the setting. Changing the setting resets commutation.
49	Minimum allowed amplitude for analog Sin/Cos encoder, in ADC^2	18633	18632 to 1863225	$(0.1\text{ V})^2$ to $(1\text{ V})^2$ The motor must be off to change the setting. Changing the setting resets commutation.
50	Glitch filter of digital input in port B, in counts/sec. The hardware calculates pulse width per each input (A or B) equal to 2/CA[50] for the same value of input.	10000000	120000 to 75000000	
51	Glitch filter of digital input in port A, in counts/sec. The hardware calculates pulse width per each input (A or B) equal to 2/**CA[50]** for the same value of input.	10000000	120000 to 75000000	

52	Planar: check at motor off that Sin/Cos amplitude is above this value		non-negative	Set WS[16] and object 0x2085 bit 1 0 – below 1 – above or equal
53	Reserved			
54–57	Invert direction of sensor connected to socket **CA[54]**: socket number 1 **CA[55]**: socket number 2 **CA[56]**: socket number 3 **CA[57]**: socket number 4 0 – Do not invert 1 – Invert the direction		0, 1	If the socket is used for commutation, the motor must be off to change the setting, and changing the setting resets commutation.
58	Number of high-resolution bits to mask from a serial encoder. This is used when the total bits are more than what the drive can read. For a linear encoder or a single-turn rotary encoder the number of bits that the user sees is: **CA[59]** – **CA[61]** – **CA[58]** For a rotary encoder with multi-turn the total bits that the user sees is: **CA[59]** + **CA[62]** – **CA[61]** – **CA[58]**		0 to 8	The motor must be off to change the setting. Changing the setting resets commutation.
59	Resolution of serial encoder. For a rotary motor this value defines the single-turn resolution. For linear motor this value defines the total bits arriving in the protocol.		1 to 32	The motor must be off to change the setting. Changing the setting resets commutation.

60	**Sensor configuration** For Biss encoder: 0 – Do not use temperature readings. 1 – Use temperature readings. For Yaskawa encoder: 0 – Incremental encoder 4 – Absolute encoder		0, 12	The motor must be off to change the setting. Changing the setting resets commutation.
61	Reducing resolution from a serial absolute sensor, reduces the resolution by masking low-resolution bits. For a linear encoder or a single-turn rotary encoder the number of bits that the user sees is: **CA[59] – CA[61] – CA[58]** For a multi-turn rotary encoder the total bits that the user sees is: **CA[59] + CA[62] – CA[61] – CA[58]**		0 to 12	The motor must be off to change the setting. Changing the setting resets commutation.
62	Multi-turn bits in a serial absolute encoder		0 to 16	The motor must be off to change the setting. Changing the setting resets commutation.
63	Adjust the synchronization mask This command moves (back or forward) the synchronization mask for communication with the serial absolute encoder by the time specified in units of 6.667 nanoseconds (clock running at 150 MHz).		-1000 to 1000	
64	Reserved			

65	Socket number used for position gain schedule	1	1 to 4	
66	Number of bits that are transmitted in the SSI protocol		0 to 64	The motor must be off to change the setting. Changing the setting resets commutation.
67	Position of the LSB within the SSI protocol bits		0 to 56	The motor must be off to change the setting. Changing the setting resets commutation.
68	Socket number used for adding a position reference command		0 to 4	**Notes:** • Stop manager functions do not apply for this entry. • In situations where the brake is used (**BP[]**) command reference might be active regardless of the brake state. The motor must be off to change command.
69	Socket number used for adding a speed reference command		0 to 4	**Notes:** • Stop manager functions do not apply for this entry. • In situations where the brake is used (**BP[]**) the reference command might be active regardless of the brake state.

70	Socket number used for adding a current reference command		0 to 4	**Notes:** • Stop manager functions do not apply for this entry. • In situations where the brake is used (**BP[]**) the reference command might be active regardless of the brake state.
71–74	Number of points for FIR filter in socket **CA[71]**: socket number 1 **CA[72]**: socket number 2 **CA[73]**: socket number 3 **CA[74]**: socket number 4		0 to 8	If the socket is used for commutation, the motor must be off to change the setting, and changing the setting resets commutation.
75–78	Filter type in socket **CA[75]**: socket number 1 **CA[76]**: socket number 2 **CA[77]**: socket number 3 **CA[78]**: socket number 4 0 – No filter 1 – FIR filter		0, 1	If the socket is used for commutation, the motor must be off to change the setting, and changing the setting resets commutation.
79	Socket number used for additional sensor converted to DS402 master	1	1 to 4	
80	Socket number used as slave in Gantry master control		1 to 4	
81	Socket number used as master in Gantry master control		1 to 4	
82	Drive designation in Gantry mode		1 to 4	
83	Communication through Port A or B between Gantry drives 0 – Port A 1 – Port B		0, 1	

84	Communication frequency between Gantry drives 0 – 2.5 MHz 1 – 3.75 MHz 2 – 5 MHz	2	0, 1, 2	
85	Gantry motor on sequence 0 – No sequence at motor on 1 – Reset position at motor on		0, 1	
86	SSI start delay. Units are in bits		0 to 15	The motor must be off to change the setting. Changing the setting resets commutation.
87	Touch Probe socket selection		0 to 4	Socket change is not allowed while capture is enabled
88	Reserved			
89	Planar: speed for alignment search in stepper units for Y1 axis			
90	Planar: max stepper angle for alignment search for Y1 axis			
91-94	Offset of position in socket **CA[91]**: socket number 1 **CA[92]**: socket number 2 **CA[93]**: socket number 3 **CA[94]**: socket number 4			

Grouped by sensor type

Sensor ID	Sensor	Entries
1	Quad port B	**CA[50]**
2	Quad port A	**CA[51]**
3	Sin/Cos	**CA[9]**, **CA[11]**, **CA[12]**, **CA[13]**, **CA[31]**, **CA[48]**, **CA[49]**, **CA[50]**
4	Hall	**CA[1** to **7]**, **CA[20]**

5	Biss	**CA[35]**, **CA[36]**, **CA[58** to **62]**
6	Panasonic	**CA[35]**, **CA[36]**, **CA[58]**, **CA[59]**, **CA[61]**, **CA[62]**
7	Mitutoyo	**CA[35]**, **CA[36]**, **CA[58]**, **CA[59]**, **CA[61]**, **CA[62]**
8	Virtual two sine signals	**SE[1** to **7]**
9	EnDat	**CA[35]**, **CA[36]**, **CA[58]**, **CA[59]**, **CA[61]**, **CA[62]**
11	Pulse & Direction port B	**CA[50]**
12	Pulse & Direction port A	**CA[51]**
13	Emulation port B	**CA[50]**
14	Emulation port A	**CA[51]**
15	Copy profiler to socket	
16	Analog Input #1	**AD[1,2]**, **AG[1]**, **AR[1]**, **AS[1]**
17	Gurley	**CA[9]**, **CA[11]**, **CA[12]**, **CA[13]**, **CA[21]**, **CA[31]**, **CA[48]**, **CA[49]**
18	SSI	**CA[22]**, **CA[23]**, **CA[35]**, **CA[36]**, **CA[59]**, **CA[61]**, **CA[62]**, **CA[66]**, **CA[67]** , **CA[86]**
19	Yaskawa	**CA[20], CA[35], CA[36], CA[59], CA[58], CA[60], CA[61], CA[62]**
20	Gantry master	**CA[80], CA[81], KP[5],KP[4],KI[4],US[4],KP[6],ER[5],KV[81** to **90]**
21	Serial Exclusive	**CA[82], CA[83], CA[84]**
22	Resolver	**CA[9]**, **CA[11]**, **CA[12]**, **CA[13]**, **CA[31]**, **CA[32], CA[33]**, **CA[34]**, **CA[48]**, **CA[49]**, **CA[50]**
23	Kawasaki	**CA[35], CA[58]**, **CA[59]**, **CA[61]**, **CA[62]**
24	General Biss	**CA[22]**, **CA[23]**, **CA[35]**, **CA[36]**, **CA[59]**, **CA[61]**, **CA[62]**, **CA[66]**, **CA[67]**
25	Sanyo	**CA[35], CA[36], CA[58], CA[59], CA[61]**, **CA[62]**
26	Simple profiler	
27	Copy gantry differential position	
28	Serial Hiperface	**CA[9]**, **CA[11]**, **CA[12]**, **CA[13]**, **CA[24]**, **CA[31]**, **CA[48]**, **CA[49]**, **CA[50]**, **CA[58]**, **CA[59]**, **CA[62]**

Grouped by commutation method

Method ID	Method	Entries
1	Hall	**CA[1** to **7]**, **CA[20]**
2	Stepper	**SC[1** to **5]**
3	Binary search	**SC[1]**, **SC[2]**, **SC[3]**, **SC[6]**, **SC[7]**
4	Analog Hall	**CA[7]**
5	Serial absolute	**CA[7]**
6	Virtual absolute Gurley	**CA[7]**
7	Absolute commutation by PAL	

References

MO, **SC[]**, **SE[]**, **UM**

CC – Compilation Checksum

CC signs the user program and allows it to be executed after successfully completing the downloading procedure.

CANopen/CoE

Attributes

Attribute	Description
Type	Long, Read/Write
Source	All, except the user program
Restrictions	• The motor must be off. • The user program must not be running.
Range	None
Default	–1
Unit modes	All
Non-volatile	None

Remarks

In order to be able to run the user program after the downloading procedure, the host sets the **CC** command with the user program checksum. The checksum is calculated by the drive during the downloading procedure and is saved in the non-volatile memory of the drive. When the host calls the **CC** command, this checksum is compared with the given signature, and if the value matches, the program is loaded into the RAM and is ready to be executed by using the **XQ** command.

When a drive is powered up, **CC** is called automatically to allow updating of the user program without requiring the host to perform any action.

Depending on the size of the user program, the **CC** command typically launches a long process, during which background procedures, such as interpreters, should not be run. The existence of the user program will also increase the drive boot-up time after power-up or after drive reset.

The checksum calculated is the 2's complement of the 16-bit summation of the user program data (code + symbols table).

If a user program exists in the non-volatile memory of the drive (**PS** = –1), **CC** is executed automatically during the drive boot-up (after power-up or drive reset), if **CC** fails, the **PS** command returns –2.

References

XQ, **LP[*N*]**, **CP**, **HP**, **KL**, **PS**

CD – CPU Dump

CD reads the CPU database status.

CANopen/CoE

Attributes

Attribute	Description
Type	String, Read-only
Source	All, except the user program
Restrictions	None
Range	None
Default	Database OK
Unit modes	All
Non-volatile	No

Remarks

In some cases, when database processes return errors, the error should be reported in the **CD** command.

Databases are tested during some processes, such as drive boot-up (after power-up or drive reset), during motor enable (**MO** = 1 or "enable operation" of CANopen DS-402 state) or during the loading of parameters from non-volatile flash memory (after the **LD** command). If a parameter fails to be processed or encounters a conflict with another parameter, the event should be reported in the **CD** command.

Return Values

CD returns Database OK when all is OK.

CD returns Check Sum Error in cases in which there is a mismatch between the parameter structures in the non-volatile flash memory and in the RAM. This is typical for new drives (after production) and in a case of incompatibility in database size due to a new release.

CD returns Autoexec Fails in the case in which an autoexec routine exists in the user program and for some reason the routine could not executed.

During a motor enable procedure an error might occur. In some cases the error will be “Motor Could Not start – reason in CD” (Error Code 48). In such cases **CD** returns the value which is described in MF. For example, if **CD** returns **MF** = 20480, it means that an overvoltage appeared during the motor enable sequence (before the motor was actually enabled). For the whole list, refer to the **MF** command.

References

LD, **MF**, **SV**

CL – Current Limit Parameters

Set and obtain thecurrent continuous limitations and motor stuck conditions.

CANopen/CoE

Attributes

Attribute	Description
Type	Float
Source	All
Restrictions	None
Range	0 < CL[1] < MC (maximum current) 0 < CL[2] <= 100 0 < CL[3] <= 32000 0 < CL[4] <= 5000
Index range	1 to 4
Default	**CL[4]**: 3000
Unit modes	All
Non-volatile	Yes

Remarks

CL[1] defines the maximum continuous motor phase current allowed, in amperes. This parameter is used to protect the motor from overcurrent, and the load from excessive torques. The motor current (torque) command is normally limited to its peak limit, as defined by **PL[1]**. After a short period of torque demands higher than **CL[1]** (as defined by the **PL[2]** parameter and equations in the Gold Drive Administrative Manual), the torque command limit is decreased to **CL[1]**. The torque command remains limited to **CL[1]** until the average torque demand falls below 90% of **CL[1]** for a few seconds. **CL[1]** has no effect if CL[1] > PL[1].

CL[2], **CL[3]** and **CL[4]** define how the motor stuck protection is handled. A stuck motor is a motor that does not respond to the applied current command due to failure of the motor, the drive system or the motion sensor.

CL[2] defines the tested torque level as a percentage of the continuous current limit **CL[1]**.

CL[3] specifies the absolute threshold main sensor speed below which the motor is considered not moving.

CL[4] defines the present threshold time for the conditions declared by **CL[2]** and **CL[3]**. If the torque level is above the **CL[2]** limit, and the main sensor speed is below **CL[3]** and, in addition, this occurred continuously for more than **CL[4]** seconds, then the motor is stuck.

If the motor is stuck, the motion fault MF = 2,097,152 (0x200000) is set, and the motor is aborted.

If **CL[2]** < 2, the motor stuck protection is not applied.

For other values of **CL[2]**, the motor is disabled and **MF** is set to 0x200000 if the motor current command level exceeds a selected level for more than 3 seconds, without a change of significant motor speed (result), as defined by **CL[3]**.

Notes

The motor stuck protection is always applied to the velocity sensor converted to units of position sensor velocity. In dual-loop applications this protection does not pertain to failures in the auxiliary sensor.

The time constant of 3 seconds is used because almost every motion system applies high torques for short acceleration periods while the speed is slow.

The minimum current limit is MC/128. If CL[1] < MC/128, the **CL[1]** value will be accepted, but the actual current value will be limited to MC/128.

Index	Description	Type	Units	Restrictions
1	Continuous current limit	Float	Ampere	
2	Motor stuck current level	Float	Percent (of **CL[1]**)	
3	Motor stuck speed level	Float	Counts per second	
4	Motor stuck time-out	Float	Seconds	

References

PL[*N*], **MC**, **TC**, **MF**

CP – Clear Program

CP erases the user program.

CANopen/CoE

Attributes

Attribute	Description
Type	Command, Write
Source	All, except the user program
Restrictions	• The motor must be off. • The user program must be stopped (**KL** or **HP**). • Wizard mode must not be active.
Range	None
Default	None
Unit modes	All
Non-volatile	No

Remarks

The **CP** command completely clears the user program from the non-volatile flash memory of the drive.

After the **CP** command, the program status, **PS**, should be -2.

The **CP** command might take approximately 1 second. During this time, the background is idle.

References

CC, **KL**, **XQ**

CS – Force Commutation Angle

CS forces the commutation angle to specific angle without feedback checking.

Note: The **CS** command should be handled with extreme care as it modifies the commutation angle.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer
Source	All
Restrictions	Commutation may be wrong
Range	0 to **CA[18]**
Index range	
Default	
Unit modes	3
Non-volatile	No
Activation	Immediate

Remarks

The **CS** command forces the commutation angle by bypassing the commutation procedure. Should be used in stepper mode (**UM** = 3) at a specific angle and sets the commutation angle in units of counts/revolution.

Example of use:

```
MO=0

UM=3

MO=1

TC=1

PA=384; BG; // 384 is 270 degrees for stepper mode. The
commutation angle is 90 degrees from that point which means 0.

//Wait for few seconds for motor to stabilized.

CS=0 ;

MO=0
```

Handle the command with care. Incorrect commutation may cause severe damage.

References

UM, **MO**, **PA**

CU – CPU Usage

CU calculates the present CPU usage.

CANopen/CoE

Attributes

Attribute	Description
Type	Unsigned Integer , Read
Source	All, except CoE
Restrictions	No
Range	0 to 100
Default	None
Unit modes	All
Non-volatile	No

Remarks

CPU usage indicates how much of a workload is being handled by the CPU.

A load of 100% means that the processor is fully utilized and that background tasks, such as the user program and communications, will not receive any CPU time.

A load of 50% means that the CPU is available for background tasks half of the time.

The value of **CU** is between 0 and 100%.

This command pauses the background loop for 2 msec. This affects the execution time of the user program, connected communication channels and other background tasks.

References

CW – Control Word

CW specifies the control word imitating the DS-402 control word value.

CANopen/CoE

Attributes

Attribute	Description
Type	Bit Field, Read/Write
Source	USB, TCP
Restrictions	None
Range	N/A
Default	0
Unit modes	All
Non-volatile	No

Remarks

The control word is used in conjunction with the status word in the DS-402 CANopen standard for drives and motion. Typically, the control word is received in the CANopen or EtherCAT communication channel with object 0x6040. The user can imitate the object behavior or check the present results from a host which uses CANopen or EtherCAT.

For more details about the **CW** bit field, refer to the CANopen DS-402 manual.

References

DC – Set Deceleration

DC specifies the maximum allowed profiler deceleration.

CANopen/CoE

0x6084

Attributes

Attribute	Description
Type	Integer 32, Read/Write
Source	All
Restrictions	Effective on the next **BG**
Range	10 to 2e9
Default	1,000,000,000
Unit modes	**UM** = 5, **UM** = 2
Non-volatile	Yes

Remarks

The **DC** command defines the maximum allowed deceleration during the operation of point-to-point (**PA**, **PR**) and jog (**JV**, **JP**) profilers.

The **DC** command does not affect the present motion. It takes effect only on the next Begin Motion (**BG**).

The **DC** value is fed to object 0x6084 during power-up and when the Begin Motion (**BG**) command is used.

The **DC** command does not affect time-dependent motion, such as Interpolated Position or Cyclic Synchronous Mode (see DS-402 manual).

The acceleration and deceleration of the drive are subject to the limits of the **SD** value. If the **DC** value is higher than the **SD** value, the **SD** value is used and the **DC** value is ineffective.

The **DC** value can be given in user-defined units specified by the **FC** command.

References

AC, **SD**, **BG**, **PA**, **PR**, **JV**, **FC**

DD – CAN controller status (Not implemented)

DD returns the status of the CAN controller as a string. The return value is in hexadecimal format without the 0x prefix.

CANopen/CoE

Attributes

Attribute	Description
Type	Read
Source	All
Restrictions	None
Unit modes	All
Non-volatile	No

Remarks

Use the **DD** command in these cases:

- You suspect that the CAN controller is in Bus Off (no communication) mode.
- You suspect that there are many error frames on the CAN bus.
- You want to monitor the CAN controller error activities.

DD value reports:

- CAN receiver flag, indicating the following states:
  - Overrun
  - Bus off
  - Transmitter error
  - Receiver error
  - Transmitter warning
  - Receiver warning
- CAN receive error counter, which reflects the status of the MSCAN receive error counter
- CAN transmit error counter, which reflects the status of the MSCAN receive error counter
- Network status, which may have one of the following values:
  - 1 – Disconnected
  - 2 – Connected

  - 3 – Preparing
  - 4 – Stopped
  - 5 – Operational
  - 127 – Pre-operational
- All data is received from the hardware.

References

DF – Download Firmware

DF initiates the Download Firmware procedure.

Note: The **DF** command works with binary content, and the data is beyond the scope of this document.

CANopen/CoE

Download firmware is supported by using the FoE protocol. Please refer to EtherCAT manual.

Attributes

Attribute	Description
Type	Command, Write
Source	USB, FoE (RS232 & CANopen TBD)
Restrictions	• The motor must be off. • The user program must not be running. • Wizard mode must not be active.
Range	No
Default	No
Unit modes	All
Non-volatile	No

Remarks

The firmware downloading procedure involves retrieving data from the host in a binary format. The drive interpreter typically works in a string format. The **DF** command informs the drive to switch into binary mode, where the interpretation of the data differs from the regular ASCII interpreter.

Note: During execution of the **DF** command, the drive does not interpret the incoming string, and it assumes that everything is binary data.

Since the downloading is performed in the boot sector, no command can be received or processed, and all interrupts are disabled.

In case of an error and a loss of communication, the drive waits for 3 seconds before it automatically switches to the ASCII interpreter.

The **DF** command is used by the EAS. Its use beside that is not recommended.

Setting **DF** causes the shutdown of all interrupts and a reboot, after which no communication or any other sequence is allowed.

All data stored in temporary variables in the RAM are lost.

References

DL

DL – (Binary) Download

DL initiates binary download.

Note: The DL command works with binary content, and the data is beyond the scope of this document.

CANopen/CoE

Attributes

Attribute	Description
Type	Command, Write
Source	USB, FoE, TCP (RS-232 & CANopen TBD)
Restrictions	• The motor must be off. • The user program must not be running. • Wizard mode must not be active.
Range	No
Default	No
Unit modes	All
Non-volatile	No

Remarks

DL is used to download data to the drive. It uses a binary format to reduce communication load and time.

The **DL** command consists of three parts:

- Initiation packet
- Body packet
- Termination packet

Please refer to the Software Manual for more details.

The **DL** command is a complicated procedure, which is normally handled by the upper host. Elmo EAS uses this command to download firmware, database image files and user programs.

The **DL** command starts binary interpretation, during which there will be no reply over any command.

In case of an error in the procedure, the drive will automatically to the ASCII interpreter after 3 seconds of no communication.

The downloaded data is loaded to the flash memory. For this reason, the procedure might require a long time for the burning and validation of the data.

References

DF

DV[] – Desired Value

DV[] returns the desired value to the controller. The desired value is actually the command for the present control cycle derived from the profiler. The value in the case of velocity and position is in internal units, namely, counts/sec and counts.

CANopen/CoE

Attributes

Attribute	Description
Type	See the table below.
Source	All
Restrictions	None
Range	See the table below.
Index range	1 to 8
Default	0
Unit modes	All with respect to the relevant control loop
Non-volatile	No

Remarks

When the motor is disabled, the controller is not active and the desired value is 0.

Indices

The following table describes the **DV[]** entries.

Index	Description	Type	Values	Notes
Input number	Command description	Similar to the Type attribute	Range or any other value description as applicable to the command	Similar to the Restriction attribute
1	Current command	Float	-MC to +MC [amperes]	Actual current reference to the current controller, including all

2	Velocity command	Integer	-2e9 to +2e9 [counts/sec]	Actual speed reference to the speed controller, including the output of the position controller
3	Position command	Integer	-2e9 to +2e9 [counts]	Actual position reference to the position controller
4	Velocity command only from the socket reference	Integer	-2e9 to +2e9 [counts]	The socket reference portion of the speed command
5	Software velocity command	Integer	-2e9 to +2e9 [counts/sec]	The software portion of the speed command to the controller
6	Position command only from the socket reference	Integer	-2e9 to +2e9 [counts]	The socket reference portion of the position command
7	Software position command	Integer	-2e9 to +2e9 [counts]	The software portion of the position command to the controller
8	Stepper angle reference	Integer	0 to 511	The calculated angle to the current loop when stepper mode is used
12	ECAM table input	Integer	-2^{31} to $(2^{31}-1)$ [counts]	The input to the ECAM table (after ratio)

References

EA[*N*] – Feedback Emulation Parameters

EA[*N*] enables the configuration and activation of feedback emulation.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Unsigned Short
Source	USB, RS232, TCP, EoE
Restrictions	According to the array index
Range	According to the array index
Index range	1 to 8
Default	0
Unit modes	All
Non-volatile	Yes
Attribute	None

Remarks

The emulation function emulates any feedback/encoder (socket) readings to one of the following waveform formats on Port-C A/B/I outputs:

1. Quadrature A/B wave format
2. Pulse/Direction wave format
3. Up/Down wave format
4. Halls signals

Emulation is supported by specific drive hardware only (GCON-based).

If sockets which are used by the emulation are changed during the emulation operation, the emulation is terminated. The value of **EA[1]** is set to 0 (disabled).

The emulation configuration must include a quadrature socket that is configured as the emulation feedback. The ID of this socket must be 13 or 14.

If the follower error between the emulated socket and the emulation position is greater than ±1,000,000,000, the emulation will automatically stop, and **EA[1]** will report -1. After this error the user must disable the emulation, i.e., must set **EA[1]** = 0, before re-enabling it.

The emulation function is supported from PAL version 8 and above. Refer to the **VP** command.

Halls emulation output is supported from PAL version 12 and above. Refer to the **VP** command.

SCORE feedback emulation is not supported.

Indices

The following table describes the **EA[*N*]** entries.

<table>
<tr><th>Index</th><th colspan="2">Description</th><th>Type</th><th>Default</th><th>Restrictions</th></tr>
<tr><td rowspan="7">1</td><td>Value</td><td>Emulation Output</td><td rowspan="7">Integer</td><td rowspan="7">0</td><td rowspan="7"></td></tr>
<tr><td>-1</td><td>Error. Read-only
Cannot follow feedback rate</td></tr>
<tr><td>0</td><td>Emulation disable</td></tr>
<tr><td>1</td><td>Quadrature wave signals</td></tr>
<tr><td>2</td><td>Pulse/Direction wave signals</td></tr>
<tr><td>3</td><td>Up/Down wave signals</td></tr>
<tr><td>4</td><td>Hall signals</td></tr>
<tr><td>2</td><td colspan="2">Emulation pulse width N in up/down or pulse/direction waves
N = 2 to 75 pulse width is N * 13.3 [nsec]
N = 76 to 202 pulse width is (N – 75 + 1) * 1.04 [µsec]</td><td>Unsigned Short</td><td>2</td><td>2 to 202</td></tr>
<tr><td rowspan="3">3</td><td colspan="2">Emulation direction</td><td rowspan="3">Integer</td><td rowspan="3">0</td><td rowspan="3"></td></tr>
<tr><td>0</td><td>Direction similar to the emulated encoder</td></tr>
<tr><td>1</td><td>Direction of the emulated output is inversed</td></tr>
<tr><td>4</td><td colspan="2">Emulated socket number 1 to 4</td><td>Integer</td><td>1</td><td>If EA[8]!=0, then value must differ from EA[5].</td></tr>
<tr><td>5</td><td colspan="2">This value is updated during the initialization of the socket with ID 13 or 14. These socket ID values indicate emulation quadrature feedback sockets.
Applicable only to EA[8]!=0</td><td>Integer</td><td>0</td><td>Read-only</td></tr>
<tr><td>6</td><td colspan="2">Emulation multiplier
N = 1 to 2147483647 defines the emulation multiplier. The number of emulated encoder pulses and, as a result, the velocity value will be multiplied by N.</td><td>Unsigned Long</td><td>1</td><td></td></tr>
</table>

<table>
<tr><td>7</td><td colspan="2">Emulation scale factor
N = 0 to 30 defines the emulation scale factor 2^N. The number of emulated encoder pulses and velocity value will be divided by 2^N.</td><td>Unsigned Short</td><td>0</td><td></td></tr>
<tr><td rowspan="3">8</td><td>Value</td><td>Description</td><td rowspan="3">Unsigned Short</td><td rowspan="3">0</td><td rowspan="3"></td></tr>
<tr><td>0</td><td>Use internal emulation feedback</td></tr>
<tr><td>1</td><td>Use AqB socket for emulation feedback</td></tr>
</table>

References

CA[], **GO[]**, **OL[]**, **VP**

EC – Error Code

EC specifies the interpreter and communication error code.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All, except the user program
Restrictions	None
Range	0 to 255
Default	0
Unit modes	All
Non-volatile	None

Remarks

The **EC** command reports the error code of the last accepted command that returned an error.

When the processing of a command fails, the error code is returned immediately with a question mark in the response to that command.

For example,

WW=10 ;

The command does not exist and will generate the error code **3?;**

Here 3 is the number of the error code. The number is not a printable ASCII table.

The question mark (?) means that an error occurred in the last command.

The semicolon (;) is a terminator and means that the interpretation was completed.

The **EC** command returns a printable (ASCII) value of the error code.

EC always keeps the last error code, which will be overwritten when the next error occurs. **EC** is not updated in cases in which the command is completed with no error.

EC = 0 clears the **EC**.

Error Codes

The following table details the command. In some examples reserved commands are used.

Error Code	Error String	Description / Example / Remedy
1	Do not Update	**Reserved**

2
BAD_COMMAND
	Bad Command	Non-existent command
		This error occurs when an entry that refers to a non-existent command is used.
		Examples:
		`PG`
		The command **PG** does not exist.
		`GM[2]`
		The command **GM** does not exist.
		`5+14+GT`
		The command **GT** does not exist.

3 BAD_INDEX	Bad Index	**IL[7]** generates an error, because the index range is from 1 to 6. Adhere to the index range for the command used.
5 BEYOND_ALPHA_BET	Has No Interpreter Meaning	**Reserved**
6 PROGRAM_NOT_RUNNING	Program is not running	The command requires a running program. **Reserved**
7 BAD_MODE_INIT_DATA	Mode cannot be started - bad initialization data	This error is returned when preset values of a function introduce a conflict. For example, there may be a conflict between the first index in the PVT table (**PV**) and the available write pointer (**MP[6]**) when PVT motion begins.
8 MOTION_TERMINATED	Motion terminated, probably data underflow	Motion terminated. Data underflow probably occurred. **Reserved**

9 CAN_MESSAGE_LOST	CAN message was lost	CAN message was lost. **Reserved**
10 NOT_CAN_PDO	Cannot be used by PDO	CAN cannot be used by PDO.
11 FLASH_WRITE_FAILED	Cannot write to flash memory	Most probably a hardware problem **Reserved**
12 NOT_AVAILABLE_FOR_MODE	Command not available in this unit mode	**Reserved**
13 UART_IS_BUSY	Cannot reset communication - UART is busy	Cannot reset RS232 communication since the lines are busy. For example: while uploading Recorder data, (while BH command is in process).
14 CANT_DO_NMT	Cannot perform CAN NMT service	Cannot perform CAN NMT service. **Reserved**
15 LIFE_TIME_ERROR	CAN Life time exceeded - motor shut down	CAN life time exceeded – motor shut down. **Reserved**
16 ARRAY_IS_EXCEPTED	The command attribute is array '[ ]' is expected	The command attribute is array. '[ ]' is expected. **Reserved**

17
WRONG_UPLOAD_SETTING
	Format of UL command is not valid - check the command definition	The format of the **UL** command is not valid. Check the command definition.
		Example:
		`UL=0x10480000`
		The MSB digit is 1 instead of 0 (it must always be 0).
		This command should be
		`UL=0x00480000`
		Refer to the **UL** command.

18 EMPTY_ASSIGN	Empty Assign	The interpreter expects a numerical value to appear after the equals sign (=). **Reserved**

19
 	
BAD_COM_FORMAT
	Command syntax error	This error indicates that an incorrect syntax was used or that an incorrect use of legal command was performed.
		
		For example:
		
		Calling a non-parameter command:
		
		`SR=100 ;`
		
		Syntax error:
		
		`200 = AC ; // expression on the left-hand side`
		
		Calling a non-indexed command:
		
		`SD[5]; // SD is a scalar command`
		
		`CA=3 ; // CA[] is a vector command`

21 OUT_OF_RANGE	Operand Out of Range	FS=2,000,000,100 returns this error because the required speed is beyond the limits of the drive.
22 ZERO_DIVISION	Zero Division	Division by zero was attempted. **Reserved**

23
 	
NOT_ASSIGNED_CMD
	Command cannot be assigned	The command cannot be assigned.
		
		For example, assigning a digital input as Safety is illegal:
		
		`IL[1..6]=24`
		
		or
		
		`IL[1..6]=25`

24 BAD_OPERATOR	Bad Operation	**Reserved**
25 COM_NOT_VALID_WHILE_MOVING	Command Not Valid While Moving	**Reserved**
26 PROFILER_NOT_SUPPORTED_IN_UM	Profiler mode not supported in this unit mode (**UM**)	This error indicates that the requested profile cannot be performed with the present control loop. For example, in Torque Control (**UM** = 1) the user requests the position relative command (**RM** = x).

28
OUT_OF_LIMIT
	Out Of Limit	The value entered in a parameter is outside of the range declared by another related parameter.
		Example:
		`XM[1] = -20000`
		`XM[2] = 20000`
		`MO = 1`
		`PA = 30000`
		The value of **PA** is outside of the range for **XM[1]**/**XM[2]**.

29 SET_OBJECT_RET_ERR	CAN set object return an abort when called from interpreter	The CAN set object returned an abort when it was called from the interpreter. **Reserved**
30 NO_PROGRAM_TO_CONTINUE	No program to continue	There is no program to continue. **Reserved**
31 GET_OBJECT_RET_ERR	CAN get object return an abort when called from interpreter	The CAN get object returned an abort when it was called from the interpreter. **Reserved**
32 UART_ERROR	Communication overrun, parity, noise, or framing error	**Reserved**
33 BAD_SENSOR_SETTING	Bad sensor setting	A bad sensor setting was made during Set Motor Enable (**MO** = 1). **CA[18]** and **CA[19]** are zero when the mode is other than stepping mode (**UM** = 3).
34 CMD_CONFLICT	There is a conflict with another command	There is a conflict with another command. Cases: Set the **OB** command.

36 BAD_COMMUTATION_SETTING	Commutation method (**CA[17]**) or commutation table does not fit to sensor	This error occurs while attempting **MO** = 1 when the commutation method or the commutation table does not fit the sensor. Example: `CA[20] = 0`, i.e., no Halls are present. `CA[17] = 1`, i.e., commutation by Halls.
37 HALL_LOCATION_CRASH	Two Or More Hall sensors are defined to the same place	**Reserved**
39 BEGIN_IN_PAST	Motion start specified for the past	Motion start was specified for a time in the past. **Reserved**
41 MISMATCH_PRODUCT	Command is not supported by this product	**PP[1]** gets a wrong protocol value. The current value supported is 1 (RS232).
42 NO_SUCH_LABEL	No Such Label	Example: **XQ##FOO** will return this error if neither a label ##FOO nor a function with the name FOO exists in the user program.
43 MUST_RESET_FAULT	CAN state machine in fault(object 0x6041 in DS-402)	Reset the fault by sending the control word through CAN communication (the value of CAN object 0x6040 must be set to 0x80). Refer to the description of objects 0x6040 and 0x6041 in the Elmo CANopen Implementation Manual. **Reserved**
45 RETURN_ERROR_FROM_SUB	Return Error From Subroutine	Return error from a subroutine **Reserved**
46 MULTICAP_WITH_STOP	May Not Use Multi-capture Homing Mode With Stop Event	Multi-capture homing mode cannot be used when there is a stop event. **Reserved**

47 PROGRAM_NOT_COMPILED	Program does not exist or not Compiled	An attempt to run (**XQ** command) a program when the program has not been compiled (**CC** command) will generate this error.
48 MOTOR_ON_FAILED	Motor cold not start - fault reason in CD	The motor could not start. The reason for the fault can be retrieved from **CD**. For example: Trying to set **MO** = 1 while: there is a motor bus undervoltage, there is a safety input fault, there is a motor short status, there is an overtemperature situation etc.
50 STACK_OVERFLOW	Stack Overflow	**Reserved**
51 INHIBIT_ABORT_IS_ACTIVE	Inhibit OR Abort inputs are active, Cannot start motor	Inhibit or abort inputs are active. The motor cannot be started.
52 PVT_QUEUE_FULL	PVT Queue Full	The PVT queue is full. **Reserved**
53 ONLY_FOR_CURRENT	Only For Current	**Reserved**
54 BAD_DATABASE	Bad Data Base	An attempt to save post processing parameters failed because of an incorrect parameter value. Or User program download failed (wrong symbol table). Try to re-compile the program and download.
55 BAD_CONTEXT	Bad Context	This error is caused by privileged commands used in auto-setup sessions. **Reserved**

56 MISMATCH_GRADE	The product grade does not support this command	User may have attempted to set or activate features that are available only for the Advanced Gold models. **Reserved**
57 MOTOR_MUST_BE_OFF	Motor Must be Off	This error is caused by trying to set parameters which are not allowed to be set under Motor-On, that is, while the motor is on. Example: Set **TS** (**TS** = *n*) while **MO** = 1.
58 MOTOR_MUST_BE_ON	Motor Must be On	**PA** = 1000 generates an error if **MO** = 0. The absolute position reference is automatically set to the present position when **MO** = 1; therefore, when **MO** = 0, setting **PA** is pointless.
60 BAD_UNIT_MODE	Bad Unit Mode	The reference command is not suitable for the unit mode. For example: If **UM** = 2 (velocity loop) **PA** = 1000 (point-2-point motion) generates this error.
61 DATABASE_RESET	Data Base Reset	This error may occur after upgrading the drive version if the newer version uses a different database structure. Try to set the correct user parameters. Then save (**SV** command). The **SV** command will check the new database, save it or return errors.
64 ACTIVE_TABLE_LOCATION	Cannot set the index of an active table	**Reserved**
65 DISABLED_BY_SW	Disabled By SW	**Reserved**

66 AMP_NOT_READY	Amplifier Not Ready	This error may appear after **MO** = 1 is set immediately after a change to **MO** = 0 (in less than 10 ms). This sequence will show an error by calling prgerr(0). This will return the error.
67 RECORDER_IS_BUSY	Recorder Is Busy	This error appears when an attempt is made to change the recorder configuration or to set a new recording request while the recorder is busy. Let the recorder complete its job.
68 NON_EXISTING_PROFILER_MODE	Required profiler mode is not supported	
69 REC_SETTING_ERROR	Recorder Usage Error	Example: RC=2 (record second vector only); RR=2 and later… BH=1 (bring first vector only) is an error, because an attempt is made to bring a vector that was not recorded.
70 REC_INVALIDATE	Recorder data Invalid	Recorder settings (such as RC=n) have been changed since the last records were made or the recorder has not been operated at all since power-up.
71 HOMING_IS_BUSY	Homing is busy	This error is caused by incorrect activation of homing. It also can be that a shared buffer is used, preventing the recorder to be activated. For example: Trying to use data recording while homing is using the recorder buffer (**HM[11]** = 5 or **HF[11]** = 5) Or Setting **HM[1]** while the DS-402 homing mode is activated or the DS-402 touch probe is on.

72 MUST_BE_EVEN	Modulo range must be even	**Reserved**
73 PLEASE_SET_POSITION	Please Set Position	**Reserved**

74
DS-402_PROF_PROBLEM
	Bad profile database, see 0x2081 for object number (**EE[2]**)	During the initiation of the profiler, the database is tested for conflicts in the parameters. If one of the parameters conflicts with another, this error will be presented.
		The problematic parameter (object) can be retrieved by **EE[2]** or CANopen object 0x2081.
		For example:
		`MO=0`
		`UM=5`
		`XM[1]= -1000000`
		`XM[2]= 1000000`
		`VL[3]= -1500000`
		`VH[3]= 900000`
		`MO=1`
		EC returns this error.
		EE[2] returns 24701 (0x607D).

77 BUFFER_TOO_LARGE	Buffer Too Large	**Reserved**
78 OUT_OF_PROG_RANGE	Out of Program Range	Error: A virtual machine had to jump into an out-of-code-segment area. The amount of memory allocated for the user program is stated in the drive's User Manual.
80 ECAM_PARS_INCONSISTENT	ECAM data inconsistent	The ECAM data is inconsistent. **Reserved**
81 DL_PROCESS_FAIL	Download failed see specific error in **EE[3]**	Download failed. See the specific error in **EE[3]**.

82 PROGRAM_IS_RUNNING	Program Is Running	Wait until the program finishes, or use the **HP** command or **KL** command to stop the program.
83 CMD_NOT_FOR_PROGRAM	Command is not permitted in a program.	A command which was used inside a user program is not allowed to be used within a program. For example: The next expression **XQ##START** inside a user program is an error because this command (**XQ**) is a NON-PROGRAM command.
84 NOT_IN_PTP_MODE	The System Is Not In Point To Point Mode	A **PR** (position relative) value cannot be set in a non-PTP mode, because it has no reference position from which to start. **Reserved**
86 PVT_QUEUE_LOW	PVT table is soon going to underflow	The PVT table is soon going to underflow. **Reserved**
88 ECAM_LAST_OUT_OF_RANGE	ECAM last interval is larger than allowed	The ECAM last interval is larger than allowed. **Reserved**
90 CAN_SM_NOT_READY	CAN state machine not ready (object 0x6041 in DS-402)	**Reserved**
91 BAD_HEAD_POINTER	Bad PVT Head Pointer	Bad PVT Head Pointer **Reserved**
92 PDO_NOT_CONFIGURED	PDO not configured	PDO is not configured. **Reserved**
93 WRONG_INIT	There is a wrong initiation value for this command	Reset queue length before updating queries. **Reserved**
95 POSERR_OUT_OF_MODULO	ER[3] Too large for modulo setting applied	**Reserved**
96 PROG_TIME_OUT	User program time out	**Reserved**

97 COM_RX_OFLOW	RS232 receive buffer overflow	RS232 received a buffer overflow. Characters arrived through RS-232 at too high a rate, causing the internal storage to exceed its capacity. No more space is left to store new characters.
98 CANT_GET_ADC_OFFSETS	Cannot measure current offsets	Current offsets cannot be measured. **Reserved**
99 BAD_AUX_SENSOR_SETTING	Bad auxiliary sensor configuration	The auxiliary feedback entry does not configure as output during the activation of Output Compare. **Reserved**
100 CAN_NOT_MODIFY_PWM	The requested PWM value is not supported	The PWM frequency that was requested cannot be used with the drive. This error means that **XP[2]** cannot be set because **MC** (the maximum current) and **BV** (the maximum bus voltage) parameters were not identified (a default was set). Contact Technical Support.
101 BAD_SERIAL_PROTOCOL_FORMAT	Absolute encoder setting problem	When encoder temperature (**TI[3]**) is read, this error will be generated if the encoder is not configured for one of the supported modes (26 bits or 32 bits). See encoder configurations: `CA[59] = 26` or 32 (bits) `CA[41] = 5` (BIS mode) `CA[60] = 1` (temperature readable)

105
SPEED_PI_OF_RANGE
	Speed loop KP out of range	The speed loop **KP[2]** or **KI[2]** is out-of-range, i.e., has a negative value.
		Example:
		`KP[2]=-2.5`
		Or
		`KI[2]=-44.09`

106 POS_KP_OUT_OF_RANGE	Position loop KP out of range	**Reserved**

110 NUMBER_TOO_LONG	Too long number	The number is too long. The number of digits before or after the decimal point exceeds the limit of 20 digits.
111 KV_INVALID_VECTOR	KV vector is invalid	**Reserved**
112 KV_INVALID_SCHEDULING	KV defines scheduled block but scheduling is off	**Reserved**
113 EXP_TASK_QUEUE_FULL	Exp task queue is full	The Exp task queue is full. **Reserved**
114 EXP_TASK_QUEUE_EMPTY	Exp task queue is empty	The Exp task queue is empty. **Reserved**
115 EXP_OUT_QUEUE_FULL	Exp output queue is full	The Exp output queue is full. **Reserved**
116 EXP_OUT_QUEUE_EMPTY	Exp output queue is empty	The Exp output queue is empty. **Reserved**
117 KV_INVALID_SENSOR_FILTER	Bad KV setting for sensor filter	See the **KV** command section of this manual. **Reserved**
118 BAD_KV_VECTOR	Bad KV setting	This can happen when the **KV** parameters are not set according to the correct feedback with either a length or value restriction. **Reserved**
119 RES_FILT_OUT_OF_RANGE	Analog Sensor filter out of range	This error is generated when the filter **KV**, set for analog feedback, is beyond its legal range. **Reserved**
120 RES_FILT_BAD_BLK_NUM	Analog Sensor filter may contain 0 or 2 blocks	This error is generated when the analog sensor filter contains an incorrect number of blocks. **Reserved**

121 RES_RESOLVER_NOT_READY	Please wait until Analog Sensor initialized	This error is generated when the initiation procedure of the analog sensor is not completed and an attempt is made to enable the motor. **Reserved**
122 MODE_NOT_SUPPORTED_OR_DISABLED	Motion mode is not supported or with initialization conflict	The motion mode is not supported or is in a conflict during the initiation. This can be caused, for example, upon switching from DS-402 motion modes into Elmo's motion modes without adjusting the motion mode parameters.
123 PROFILER_QUEUE_FULL	Profiler queue is full	This error indicates that the point-to-point position buffers are full. The user is required to wait until the first position reaches "target reached" or flush the buffer according to the DS-402 Profile Position mode.
125 PERSONALITY_NOT_LOADED	Personality not loaded	There is a problem in the non-volatile flash memory (firmware image). Try to reload the firmware.
126 FAILED_USER_PROG	User Program failed - variable out of program size	A variable is outside of the program area. Try to reload the program. Try to reduce the number of variables.
127 INCONSIST_MODULO	Modulo range must be positive	The modulo range must be positive. **XM[2]** is less than or equal to **XM[1]**, or **YM[2]** is less than or equal to **YM[1]**. **Reserved**

128 BAD_VAR_INDEX	Bad variable index in database	This error occurs when an attempt is made to map user program (global) variables to the recorder. It is performed using the **DB##RV[*N*]** command. *N* must be in the range from 0 to 7. Using a value of *N* outside of this range (*N* < 0, *N* > 7) will cause this error. In case the recorder variable is an array, for example, VarArr[M]: `DB##RV[N]=VarArr[M]` If *M* is greater than the maximum array index, this error occurs.
129 VAR_NOT_ARRAY	Variable is not an array	When a variable is mapped into the recorder variables list (the **DB##RV[*N*]** command), this error occurs if an attempt is made to map a scalar variable as an array. Example: `DB##RV[1]=MyVar[3]` MyVar is not an array.
130 BAD_VAR_NAME	Variable name does not exist	When a variable is mapped into the recorder variables list (the **DB##RV[*N*]** command), this error occurs if the name of the variable does not exist (the wrong name is given). Example: `DB##RV[1]= MyVar` MyVar is an incorrect name.
131 LOCAL_USER_VAR	Cannot record local variable	This error occurs when an attempt is made to map a user local variable. Only global variables are allowed. Example: `DB##RV[1]=LocVar` where LocVar is a local variable.

132
VAR_IS_ARRAY
	Variable is an array	This error occurs when a user variable is mapped to the recorder and the variable is an array, but the brackets and index are missing.
		Example:
		`DB##RV[1]=VarName`
		VarName is an array.
		It should be as below:
		`DB##RV[1]=VarName[2]`

133
MISMATCH_FUNC_ARGS
	Number of function input arguments is not as expected	This error occurs when a user function defined with arguments is called through the **XQ** command with no input arguments or with a number of arguments that does not conform to the function definition.
		Example:
		The program defines a function:
		`function main(int a)`
		Calling this function as
		`XQ##main`
		will return this error.

134 LOCAL_USER_FUNC	Cannot run local label/function with the **XQ** command	A local label cannot be run with the **XQ** command. For example: `XQ##START` When **START** is defined in the user program inside a user **function**, it is considered to be a local label, and therefore it is illegal to use it with the **XQ** command. **Reserved**
135 FREQ_IDENTIFICATION_FAIL	Frequency identification failed	Frequency identification failed.

136 NOT_A_NUMBER	Not a number	Not a number. Float overflow was detected (number >= $1e^{37}$). Example: `UF[1]=5e39`
137 PROG_ALREADY_COMPILED	Program already compiled	**Reserved**
139 TOO_MANY_BREAK_PTS	The number of break points exceeds maximal number	The number of breakpoints exceeds the maximum number. The maximum number of breakpoints in the Gold line is 6. The command to add breakpoint is: `DB##BP=ProgramLine`
140 NOT_RELEVANT_BREAK_PNT	An attempt to set/clear break point at the not relevant line	An attempt was made to set/clear a breakpoint in a non-relevant line. For every line of the text program, there is a corresponding line of compiled code. This error appears during an attempt to set a breakpoint at a non-corresponding line of compiled code. Example: `DB##BP=653` If the actual Drive breakpoint was set at a different code line, this error appears.
141 SECTION_NOT_ERASE	Boot Identity parameters section is not clear	The boot identity parameters section is not clear. An internal error occurred during download of boot identity parameters. **Reserved**
142 MISMATCH_DATA_CHECKSUM	Checksum of data is not correct	Checksum of data is not correct. An internal error occurred during download of boot identity parameters. **Reserved**
143 MISSING_DI_PARAMETERS	Missing boot identity parameters	Boot identity parameters are missing. **Reserved**

144 NUM_STACK_UNDERFLOW	Numeric Stack underflow	Numeric stack underflow. An attempt was made to retrieve an entry from an empty stack.
145 NUM_STACK_OFLOW	Numeric stack overflow	Numeric stack overflow. An attempt was made to push a value to the numeric stack when it is full. The user program contains very complex code requiring more stack space than is available. It may also be that there are too many called subroutines. An expression in the command line of the interpreter is too complex; it calls too many functions, causing the numeric stack to overflow.
146 EXP_STACK_OFLOW	Expression stack overflow	An attempt was made to push a value to the expression stack when it is full. **Reserved**
147 EXEC_COMMAND	Executable command within math expression	An executable command was inserted within a mathematical expression. An attempt was made to assign an executable command. **Reserved**
148 EMPTY_EXPRESSION	Nothing in the expression	There is nothing in the expression. An attempt was made to evaluate an empty expression. Example: `AC=;` This is wrong, because the assigned value is missing.
149 UNEXPECTED_TERMINATOR	Unexpected sentence termination	Unexpected sentence termination An expression terminator appears in the middle of the expression. **Reserved**

150 ENDLESS_SENTENCE	Sentence terminator not found	Sentence terminator not found The expression is too long to be evaluated (it exceeds the maximum length). **Reserved**
151 PARANTHESES_MISMATCH	Parentheses mismatch	Parentheses mismatch There is a mismatch between opening and closing parentheses. This error pertains to both parentheses and brackets. Example: `sin(2;` This is wrong because a closing parenthesis is absent.
152 BAD_OPERAND_TYPE	Bad operand type	Bad operand type There is a mismatch between the actual value type and the expected value type. An internal compiler error occurred due to a mismatch between an operand type and its addressing mode. Contact Technical Support.

153
 	
NUM_OVERFLOW
	Overflow in a numeric operator	Overflow in a numeric operator:
		
		Example:
		
		`UI[1]= 200000000`
		
		`UI[1]=UI*2000 //It generates this error.`
		
		Division by zero:
		
		`UI[1]/0`
		
		Remainder calculation by zero.
		
		`UI[1]%0`

154 OUT_OF_DATA_SEG	Address is out of data memory segment	Address is out of data memory segment. The address of a variable in the data segment exceeds the data segment size. This internal compiler error is caused by corrupted compiled code. In such a case, email Technical Support for assistance. Attach the Composer date and version (in the Help menu) and the program you attempted to compile.
155 BEYOND_STACK_RANGE	Beyond stack range	Compiled code contains a pointer to a stack entry that exceeds the actual stack range (STACK_IMMEDIATELY addressing method). This internal compiler error is caused by corrupted compiled code. In such a case, email Technical Support for assistance. Attach the Composer date and version (in the Help menu) and the program you tried to compile.
156 BAD_OPCODE	Bad op-code	Compiled code contains mismatched addressing mode. This internal compiler error is caused by corrupted compiled code. In this case, email Technical Support for assistance. Attach Composer date and version (in Help menu) and the program you tried to compile.
157 NO_AVAILABLE_PROG_STACK	No Available program stack	An attempt was made to run too many user programs simultaneously. **Reserved**
158 OUT_OF_FLASH_RANGE	Out of flash memory range	Failure in download program procedure For example: The size of the program function table is greater than the maximum allowed size. Try to reduce number of functions in the program.

159 FLASH_VERIFY_ERROR	Flash memory verification error	Failure in download process: checksum does not match. Possible hardware problem. Contact Technical Support.
160 ABORTED_BY_OTHER_THREAD	Program aborted by another thread	**Reserved**
161 PROGRAM_NOT_HALTED	Program is not halted	An attempt was made to execute a command that requires the user program to be halted. For example: Activation of the **XC** command while the virtual machine is not in the halted state.
162 BAD_NUMBER	Badly formatted number	A floating point number exceeds the valid range supported by the Gold line. **Reserved**
163 OUT_OF_PROGDATA_MEM	Not enough space in program data segment	There is not enough space in the program data segment. Try to reduce variable usage in the user program.
164 EC_COMMAND	EC command (not an error)	**Reserved**
165 FLASH_READ_FAILED	An attempt to access flash memory while busy	An attempt was made to access serial flash memory while busy. Failure occurred on reading serial flash memory. This might be due to a hardware problem. Contact Technical Support.
166 OUT_OF_MODULO	Out Of Modulo Range	This error occurs when the main encoder (the **PX** command) is set to an out-of-range value: **XM[1]** is the lower limit **XM[2]** is the upper limit **XM[1]** < **PX** < **XM[2]** (See CANOpen objects 0x607B.2 and 0x607B.1).

167 INFINITE_LOOP	Infinite loop in for loop - zero step	**Reserved**
168 SPEED_2_LARGE_2_START	Speed too large to start motor	**MO** = 1 or the motor was started with the Enable switch while the motor was rotating too rapidly. **Reserved**
169 CPU_PERIPHERAL_IS_BUSY	Time out using peripheral.(overflow or busy)	This error occurs in cases in which CPU peripherals are used from two resources. For example: The **UL** or **PK** uploading command failed because the transmitter buffer is full. The OP command is not accepted because requests to update the output port arrived from two channels (CANopen & USB) simultaneously.
170 SFLASH_ERASE_SECT_FAILED	Cannot erase sector in flash memory	The serial flash memory cannot be erased. This problem may occur during downloading of the FW or parameters. Contact Technical Support.
171 SFLASH_READ_FAILED	Cannot read from flash memory	The serial flash memory (SFlash) cannot be read because an attempt was made to read an illegal area in the SFlash or an internal board communication failure occurred. Contact Technical Support.
172 SFLASH_WRITE_FAILED	Cannot write to flash memory	Writing to the serial flash memory (SFlash) failed because an attempt was made to write to an illegal area in the SFlash or an internal board communication or SFlash failure occurred. Contact Technical Support.
173 PROGRAM_TOO_LARGE	Executable area of program is too large	This error is detected during the **CC** command. The user program (the entire code image, excluding text) is too large. Try to reduce the size of the user program code.

174 NO_PROGRAM_LOADED	Program has not been loaded	This error is detected during the **CC** command. The user program image is illegal or does not exist. Try to download program again.
175 PROGRAM_CHK_NOT_ERASED	Cannot write program checksum - clear program (CP)	An incorrect checksum value was set in the CC command after the user program was downloaded. The checksum is calculated by the drive, and a failure occurred during the downloading session. Try to query the **CC** value and set it to the value calculated by the drive.
176 CODE_VAR_TOO_LARGE	User code, variables and functions are too large	The user program non-text area (code, variables and function table) is oversized. Try to reduce program code.
181 PROG_WRITE_FLASH_FAILED	Writing to Flash program area, failed	Writing to the serial flash program area during the Set CC command failed. Contact technical support.
182 PAL_BURN_IN_PROCESS	PAL Burn Is In Process or no PAL is burned	PAL burn is in process during: An attempt to set motor on (**MO** = 1). This action failed. An attempt to burn PAL. The Read **PB** command. If **PB** > 0, try again your action. If **PB** == 0, wait. If **PB** < 0, try to burn PAL again or contact technical support.
183 PAL_COMMAND_DISABLED	PAL Burn (PB Command) Is Disabled	PAL burning can be enabled or disabled during the DI downloading procedure. If PAL burn is disabled, an attempt to burn PAL will generate this error.

184 CAPTURE_ALREADY_USED	Capture option already used by other operation	Only one of the following Capture options can be used at the same time: ELMO Heritage Home Touch Probe (DS-402) DS-402 Home They all use the same capture module. An attempt to use another option without closing the active one will generate this error.
185	This element may be modified only when interpolation is not active	The following elements may be modified only when interpolation is not active: **OF[22]** (CANopen object 60C0) – Interpolation sub mode select. **OV[23** to **24]** (CANopen object 60C2) – Interpolated Data Period. **OV[25** to **27]** (CANopen object 60C4) – Interpolated Buffer Org.
186	Interpolation queue is full	The interpolation queue is full. This might appear while setting object 60C1 – Interpolated data record.
187 BAD_INTERPOLATION_SUBMODE	Incorrect Interpolation sub-mode	The interpolation sub-mode is not supported.
188 GANTRY_SLAVE_DISABLE	Gantry slave is disabled	Gantry master cannot enabled if gantry slave is not enabled at current mode

References

EE[], **MF**, **SR**

EE[*N*] – Extended Error

EE[*N*] reports detailed error codes according to a specific feature described below.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer
Source	RS232, USB, TCP, EoE
Restrictions	Read-only
Range	None
Index range	1 to 6
Default	0
Unit modes	All
Non-volatile	No
Attribute	None

Remarks

Extended error information is used by the following features:

- Feedback Error
- Profiler initialization
- ECAM initialization
- Download procedure
- SDO communication error

In the above cases, if an error occurs, the **EC** or **MF** value, or abort SDO, will indicate an error, and more specific and detailed error information will be available in the relevant **EE[]** command.

An extended error code is valid only after the following scenarios:

1. *Feedback error.* In some encoder types, a feedback error can be detected and reported to the host. The following feedbacks are inspected for a feedback error:
    a. Absolute serial sensor (BiSS, EnDAT, etc.). Refer to the **EE[1]** table data.

b. Virtual absolute sensor (Gurley). Refer to the **EE[1]** table data. The Gurley error is identified by the sensor fault **MF=1**, the reason for error can read by **EE[1]** command:

MSB						LSB
10 bit (unimportant)	Amp. Error	Data not valid	Rst. Out	Quad Error	Seq. Error	0

There are actually two types of possible error:

EE[1]=20 (hex)
Amplitude is out of range. Range is define in **CA[48]** (maximum amplitude squared) and **CA[49]** (minimum amplitude squared). Both parameters are in ADC units.

EE[1]=1x (hex)
Data not valid, where x are the four LSB bits above updated in the Gurley read out algorithm

c. Analog sensor (sine/cosine sensor). Refer to the **EE[1]** table data.

EE[1]=20 (hex)
Amplitude is out of range. Range is define in **CA[48]** (maximum amplitude squared) and **CA[49]** (minimum amplitude squared). Both parameters are in ADC units.

If any of these feedbacks fail, the motor is disabled automatically, and **MF** reports an error (**MF** = 1). In cases where there is more details regarding the reason for the failure, the **EE[1]** command supports these details .
Note that the errors depend on the specific details determined by the encoder manufacturer.

2. *Profiler Initialization.* The **EC** command value is 74, which indicates an error during profiler initialization. Check **EE[2]** for the detailed error.

3. *Download FW procedure fails.* The **EC** command value is 81, which indicates an error during the download procedure. Check **EE[3]** for the detailed error.

4. *SDO communication.* If SDO returns an abort message, which indicates an error while processing the SDO command, **EE[4]** gives more details about the error.

Indices

The following table describes the **EE[*N*]** entries.

Index	Description	Type	Values
1	Feedback error, relevant according to the above description	Integer	See the table for **EE[1]** below.
2	Profiler initialization error, relevant in the case of	Integer	Returns the number of the object that caused the error. See the table for **EE[2]**

	EC=74		below.
3	Download procedure error, relevant in the case of **EC**=81	Integer	Returns a value that indicates the cause of the error. See the table for **EE[3]** below.
4	When SDO (used in CANopen) returns an abort message, **EE[4]** provides further details about the error	Integer	Returns ELMO error value according to the **EC** command error list. If the value is -1, no further information will be provided as there was no information available.
5	Faults which are detected by the Motor On procedure (**MO**=1) are detailed in **EE[5]**.	Integer	See the table **EE[5]** fault below.
6	ECAM initialization error, relevant in the case of **EC**=27	Integer	Returns a value that indicates the cause of the error. See the table for **EE[6]** below.

Errors

The following table lists the errors reported by **EE[1]**.

Value	Description	
Bits 0 to 7	Bits 0 to 7: Capture of the status field (the errors are encoder manufacturer-dependent) in the case of a serial/absolute encoder error. See Table 2.	
Bit 8	CRC error	
	0	No error
	1	CRC error
Bit 9	Encoder ID OK (Panasonic and Tamagawa only)	
	0	ID is OK
	1	ID of the sensor does not fit the sensor
Bit 10	Flag that indicates if no data arrived from the encoder	
	0	OK
	1	No data

Bit 11	Sync or CRC error between Gantry drives (excluding the serial encoder)	
	0	OK
	1	Communication error
Bits 16 to 22	EnDat error message (See Table 2)	

Table 1: EE[1] - Serial Encoder Errors

Encoder	Description		
Biss	Bit 0: Warning	If it equals 0, the encoder scale should be cleaned.	
	Bit 1: Error	A value equal to 0 indicates that the absolute position data may not be valid or that the temperature is above the maximum operating temperature of the encoder.	
General Biss	Bit 0: Error	A value equal to 0 indicates that the absolute position data may not be valid or that the temperature is above the maximum operating temperature of the encoder.	
EnDat	Bit 0: Error	If it equals 1, the internal data check failed.	
	Bits 16 to 22: Error Message A bit equal to 1 signals an error message.	16	Light source
		17	Signal amplitude
		18	Position value
		19	Overvoltage
		20	Undervoltage
		21	Overcurrent
		22	Battery
Panasonic	**Single- turn**		
	Bits 0 to 7 **Note:** For more information refer to the manufacturer's	0	0
		1	System down
		2	0
		3	0
		4	Counter overflow

Encoder	Description		
	specification.	5	Count error
		6	Full Abs status
		7	Overspeed
	Multi-turn		
	Bits 2 and 3	2	One of the following: Multi-turn error, Battery alarm, Full absolute status, counter error, counter overflow, overspeed
		3	System down
Tamagawa	**Single-turn**		
	Bit 5	5	Count error
	Multi-turn		
	Bits 0 to 7	0	Battery alarm
		1	System down
		2	Multiple revolution error
		3	0
		4	Counter overflow
		5	Count error
		6	Full Abs status
		7	Overspeed
SSI	Bit 0	0	Error bit
Mitutoyo	Bits 0 to 7	0	0
		1	0
		2	0
		3	1
		4	0
		5	Communication error
		6	Encoder alarm
		7	System error
Kawasaki	Bits 0 to 2	0	Busy
		1	Absolute track error

Encoder	Description		
		2	Interpolation error
Yaskawa	Bits 0 to 1	0	When bit 0 is equal to 0 and bit 1 is equal to 1. The command was wrong
		1	
	Bit 6	6	Alarm state
Sanyo	Bits 0 to 3: Encoder Status	0	STErr + PSErr + BA
		1	OvSpd + MemErr + OvTemp
		2	BW
		3	Busy + MemBusy
	Bits 16 to 25: Specific alarm status codes	16	Over Temperature
		17	Memory Busy
		18	Busy
		19	PS Error (Multi turn)
		20	ST Error
		21	MemErr
		22	Overspeed
		23	0
		24	Battery Alarm
		25	Battery Warning
Gurley	Bit 1: Sequence Error	1	Refer to the Gurley distributor of the encoder for details
	Bit 2: Quadrature Error	2	Refer to the Gurley distributor of the encoder for details
	Bit 3: Reset Out	3	Refer to the Gurley distributor of the encoder for details
	Bit 4: Data Not Valid	4	For details see bits 1 to 3
	Bit 5: Wrong Amplitude	5	Analog signal amplitude is out of range (see **CA[48]** and **CA[49]**)
Sin/Cos, Resolver, or Hiperface	Bit 5	5	Wrong Amplitude

Table 2: Status Bits of Different Encoders (Bits 0 to 7)

The following table lists errors reported by **EE[2]**.

Value (object)	Description
0x6091	Position ratio is out of range.
0x607C	Homing offset is out of position limits.
0x607B	Minimum position range limit is greater than the maximum position range limit.
0x607D	Software position limits have one of the following errors: • Upper and lower software position limits are equal. • Position software limits and position range limits are ambiguous. Range boundaries must not overlap.

Table 3: EE[2] - Profiler Initialization Errors

The following table lists errors reported by **EE[3]**.

Value	Description
1	Header packet is missing DL
2	Header packet termination code is not 0x1234
3	Received header packet number does not match the expected packet number
4	Header packet size is incorrect
5	Header packet checksum error
6	Header packet type is not legal
7	Body packet termination code is not 0x1234
8	Received body packet number does not match the expected packet number
9	Body packet size is incorrect
10	Body packet checksum error
11	The last packet was identified, but the packet number is incorrect
12	Downloaded file checksum failed
13	After download database post process fail
14	Time-out while waiting for a message (1.5 seconds)
15	Downloaded data is larger than the user parameters reported in DL type 19
16	Error while writing downloaded data to FLASH memory
17	General download aborting error, with no precise error information

Table 4: EE[3] - Download Procedure Errors

The following table lists the faults which occurs during the motor on procedure and reported by **EE[5]**.

Value (Object)	Description
1	Main Feedback error. Check feedback connection or settings
16	External Inhibit is triggered. Check digital input which is define in "Inhibit" state
12288 (0x3000)	Drive is in under voltage state. Check power supply .Voltage should be at least **WI[38]** volts.
20480 (0x5000)	Drive is in over voltage state. Check power supply. Voltage should not exceed **WI[36]** volts.
28672 (0x7000)	Safety switch is in safety state. Must set the safety switch to active.
45056 (0xB000)	Drive power stage is in "short protection" state.
53248 (0xD000)	Drive power state is over temperature. Check **TI[1]** for temperature read out in Centigrade.
61440 (0xF000)	Additional External Inhibit is triggered. Check digital input which is define in "Additional Inhibit" state

Table 5: EE[5] – Motor On Fault Errors

The following table lists the errors reported by **EE[6]**.

Value (Object)	Description
1	Ratio denominator (**EM[10]**) is out of range
2	First or last table index (**EM[5]** or **EM[2]**) is out of range
3	Table gap parameter (**EM[4]** or **EM[7]**) is out of range
4	Velocity or acceleration FIR filter length (**EM[12]** or **EM[13]**) is out of range
5	Entire master+slave table length is not even
6	Master table does not start from 0.
7	Master table is not a monotonic rising

Table 6: EE[6] - ECAM Initialization Errors

References

EC, **MF**, **SR**

EI – Initialize External Reference Generator

EI configures the external reference generator (ECAM/Follower) according to the latest **EM[N]** settings.

CANopen/CoE

Attributes

Attribute	Description
Type	N/A
Source	All
Restrictions	N/A
Range	N/A
Default	N/A
Unit modes	UM=2, UM=5
Non-volatile	No

Remarks

EM[N] settings, except for **EM[9],** go into effect only after applying **EI** command.

References

EM[N]

EM[*N*] – ECAM / Follower Parameters

EM[*N*] determines the behavior of the ECAM (Electronic CAM) motions.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer
Source	All
Restrictions	According to array index
Range	According to array index
Index range	1 to 13
Default	According to array index
Unit modes	**UM=5** (Position - ECAM and Follower) **UM=2** (Velocity – Follower)
Non-volatile	Yes
Attribute	None

Indices

The following table details the **EM[]** entries.

Index	Description		Type	Default	Restrictions
1	**ECAM mode**				
	Value	**Mode**	Integer	0	RM=0
	0	Follower			
	1	Non-periodical (linear) ECAM			
	2	Periodical (cyclic) ECAM			
2	Last valid index of ECAM table.		Integer	10	2 to 2048
3	**Starting position** value of the input to the ECAM function, where the ECAM function output is **ET[EM[5]]** defined algebraically as **ET** x **EM[5]**.		Integer	0	-2^{31} to $(2^{31} - 1)$

<table>
<tr><td></td><td>If EM[3] is out of range for PY, the modulo PY is employed.</td><td></td><td></td><td></td></tr>
<tr><td>4</td><td>Master reference (ΔPY) distance (gap) between consecutive points in the ECAM table ET[N].
Effective when the constant master gap table is selected (EM[11]:bit2 = 0).</td><td>Integer</td><td>1</td><td>1 to $(2^{31} - 1)$</td></tr>
<tr><td>5</td><td>First valid index of the ECAM table.</td><td>Integer</td><td>1</td><td>1 to 2048</td></tr>
<tr><td>6</td><td>Reserved</td><td>Integer</td><td>0</td><td></td></tr>
<tr><td>7</td><td>Last master segment shortening.
Last master distance (gap) ΔPY=EM[4]-EM[7].
Effective when constant master gap table is selected (EM[11]:bit2 = 0)</td><td>Integer</td><td>0</td><td>0 to $(2^{31} - 1)$
EM[7]<EM[4]</td></tr>
<tr><td>8</td><td>Reports the present index in the ECAM table.
When the ECAM motion is not active, EM[8] reports 0.</td><td>Integer</td><td>0</td><td>1 to 2048
Read-only</td></tr>
<tr><td>9</td><td>Ratio numerator.
Set EM[9] to synchronously activate the new ECAM/Follower ratio = EM[9]/EM[10].</td><td>Integer</td><td>1</td><td>-2^{31} to $(2^{31} - 1)$</td></tr>
<tr><td>10</td><td>Ratio denominator.
The EM[10] setting becomes effect only after setting EM[9].</td><td>Integer</td><td>1</td><td>1 to $(2^{31} - 1)$</td></tr>
<tr><td>11</td><td>ECAM options</td><td>Integer</td><td>0</td><td></td></tr>
<tr><td></td><td>
<table>
<tr><th>Bit</th><th colspan="2">Description</th></tr>
<tr><td>0</td><td colspan="2">Reserved</td></tr>
<tr><td>1</td><td colspan="2">Reserved</td></tr>
<tr><td>2</td><td colspan="2">ECAM table type</td></tr>
<tr><td></td><td>0</td><td>One table, constant master gap: ET[n],n=1…2048</td></tr>
<tr><td></td><td>1</td><td>Two separate master and slave tables, with non-constant gap:
• master: ET[n]>=0, n=1…N, ET[EM[5]]=0;
• slave: ET[m], m=N+1…2*N;</td></tr>
</table>
</td><td></td><td></td><td></td></tr>
</table>

	3	ECAM input source				
		0	Additional encoder (defined in socket architecture)			
		1	Main Profiler or CAN Encoder			
	4	Interpolation type				
		0	Quadratic interpolation			
		1	Linear interpolation			
	5	Table index search mode				
		0	Incremental			
		1	Direct			
12	ECAM velocity FIR filter length			Integer	1	1 to 8
13	ECAM acceleration FIR filter length			Integer	1	1 to 8

Notes

- Modifying of ECAM input source (**EM[11]:**bit3) is only possible, if motor is off (**MO**=0)
- Parameters **EM[1]**-**EM[5]**, **EM[7]**, **EM[10]**-**EM[11]** are synchronously activated when **EI** command is applied. **EM[9]**, **EM[10]** are also activated when **EM[9]** is set.
- Parameters **EM[12]**, **EM[13]** are activated at **RM**=1 command
- If non-constant master gap table is selected (EM[11]:bit2=1):
  - Master table must start from **0** and be monotonic rising
  - Shared length of master and slave tables is (**EM[2]**-**EM[5]**+1), which must be an even number
  - **EM[8]** reports the slave index. Master index can be calculated by subtracting constant offset ((**EM[2]**-**EM[5]**+1)/2) from the slave index
- Use **EM[7]** to build a constant master gap table with length which is not an integer multiple of **EM[4]**.
- When the motor is enabled (**MO**=1) and ECAM table is running (**EM[1]**=1 or **EM[1]**=2), ECAM table entries **ET[N]** can be changed "on the fly", with the exception of the **ET[EM[8]-2]**…**ET[EM[8]+2]** entries
- **Incremental calculation of the ECAM table index (EM[11]:bit5=0)**
  In this case, the index is directly calculated only once, at the ECAM engage and then incremented or decremented when the master crosses the border between two adjacent ECAM table segments. When using this mode, the difference between two consecutive master samples must not exceed the master table segment length (table gap).

- **Direct search/calculation of the ECAM table index (EM[11]:bit5=1)**
  In this case, the index is calculated every ECAM execution period; this consumes extra DSP time.
- The input to acceleration FIR filter is the output of velocity FIR filter.

References

ET[],**RM**,**EI**

EO – Echo Off

EO specifies the communication echo mode.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	USB, RS232
Restrictions	None
Range	0, 1
Default	0
Unit modes	All
Non-volatile	Yes

Remarks

When serial communication is used, the command is prompted back to the host. **EO** can turn this off. Depending on the communication protocol (RS232 or USB), the echoing is performed on character level or on the command level.

EO = 1 Enable echo.
EO = 0 Disable echo.

EO can be set from other communication lines, but it affects only USB and RS232 communication.

References

ER[] – Maximum Tracking Error

ER[] specifies the maximum follower tracking error of the relevant control loop.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	Refer to the note below.
Range	See the table below.
Index range	2 to 3
Default	**ER[2]** = 100000000 **ER[3]** = 1000000000
Unit modes	**ER[2]** for the velocity loop under **UM** = 2 and **UM** = 5 **ER[3]** for the position loop under **UM** = 5
Non-volatile	Yes
Activation	Immediate

Remarks

The tracking error is the difference between the command (desired value) and its feedback. Tracking errors include velocity and position errors. If the error exceeds this value, the motor is automatically disabled. **MF** indicates the reason for the failure.

Indices

The following table describes the **ER[]** entries.

Index	Description	Type	Values	Default
0	Reserved			
1	Reserved			
2	The maximum allowed velocity error Its value is abs(DV[2]-VX)) in counts/second		0 to 2000000000 [counts/sec]	100000000 [counts/sec]

3	The maximum allowed position error in counts Its value is abs(DV[3]-PX) in counts		0 to 1000000000 [counts]	1000000000 [counts]

References

MF, **MO**, **SR**, **DV[]**, **VX**, **PX**

ET[*N*] – ECAM table

ET[*N*] specifies the integer array which can be used as general-purpose non-volatile memory. Currently it is used by the following drive algorithms: ECAM, error mapping, and output compare.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer
Source	All
Restriction	None
Range	-2^{31} to $(2^{31} - 1)$
Default	None
Index range	1 to 2048
Unit modes	All
Non-volatile	Yes

Remarks

- When the motor is enabled (**MO**=1) and ECAM table is running (**EM[1]=1 or EM[1]=2**), the ECAM table entries can be changed "on the fly", except for **ET[EM[8]-2]…ET[EM[8]+2]** entries

References

EM[*N*],**PC[N]**

FC[] – Scaling Factors

FC[] defines user-defined units and specifies the values of the sensor resolution and transmission ratio for position, velocity and acceleration.

CANopen/CoE

0x608F, 0x6090 (reserved), 0x6091, 0x6092, 0x6096, 0x6097

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	• FC[1]*FC[6]*FC[8]< 2^{63} • FC[2]*FC[5]*FC[7]< 2^{63}
Range	1 to (2^{31} – 1)
Index range	1 to 8
Default	1 for all settings
Unit modes	All
Non-volatile	Yes

Remarks

The gear ratio together with the feed constant determines the relation between the position in user units and the actual movement in counts.

$$\frac{\text{UserUnits}}{\text{EncoderCounts}} = \frac{\text{FeedConstant}}{\text{GearRatio} * \text{PositionEncoderResolution}}$$

Here the *feed constant* is the positional movement for any motor movement and is calculated using the formula

$$\text{FeedConstant} = \frac{\text{Feed}}{\text{DrivingShaftRevolutions}} = \frac{\text{FC[7]}}{\text{FC[8]}} = \frac{\text{0x6092.1}}{\text{0x6092.2}}$$

The *gear ratio* is the ratio that defines what a gear adds to the movement and is calculated using the formula

$$\text{GearRatio} = \frac{\text{MotorShaftRevolutions}}{\text{DrivingShaftRevolutions}} = \frac{\text{FC[5]}}{\text{FC[6]}} = \frac{\text{0x6091.1}}{\text{0x6091.2}}$$

The *position encoder resolution* is the ratio between the motor shaft and the encoder counts:

$$\text{PositionEncoderResolution} = \frac{\text{EncoderCounts}}{\text{MotorShaftRevolution}} = \frac{\text{FC[1]}}{\text{FC[2]}} = \frac{\text{0x608F.1}}{\text{0x608F.2}}$$

The drive uses these variables to convert the position in user units into internal units (counts) for all position references (e.g., 0x607A) and for position feedback (e.g., 0x6063).

The *velocity factor* can be used to match the velocity units to the user-defined velocity units. The *user-defined velocity unit* is the *user-defined position unit* / sec:

$$\text{VelocityValue} = \frac{\text{PositionValue}}{\text{sec}} * \text{VelocityFactor}$$

$$\text{VelocityFactor} = \frac{\text{FC[9]}}{\text{FC[10]}} = \frac{\text{0x6096.1}}{\text{0x6096.2}}$$

The *acceleration factor* can be used to match the acceleration units to the user-defined acceleration units. The *user-defined acceleration unit* is the *user-defined velocity unit* / sec:

$$\text{AccelerationValue} = \frac{\text{VelocityValue}}{\text{sec}} * \text{AccelerationFactor}$$

$$\text{AccelerationFactor} = \frac{\text{FC[11]}}{\text{FC[12]}} = \frac{\text{0x6097.1}}{\text{0x6097.2}}$$

The velocity and acceleration factors must be used when the required acceleration or velocity (**AC, DC, SP** etc.) exceeds the maximum value (>2e9), as in the following examples:

1. The required velocity is 5e9, and the acceleration is 10e9: **FC[9]**=1;**FC[10]**=10; **FC[11]**=1;**FC[12]**=1; **SP**=5e8; **AC**=**DC**=1e9;

2. The required velocity is 1e9, and the acceleration is 10e9: **FC[9]**=1;**FC[10]**=1; **FC[11]**=1;**FC[12]**=10; **SP**=1e9; **AC**=**DC**=1e9.

Indices

The following table describes the **FC[]** entries.

Index	Description	Type	Values	Note
0	Reserved			
1	Position encoder resolution numerator	Integer	1 to (2^{31} – 1)	0x608F.1
2	Position encoder resolution denominator	Integer	1 to (2^{31} – 1)	0x608F.2
3	Velocity encoder resolution	Integer	1 to (2^{31} – 1)	0x6090.1

Index	Description	Type	Values	Note
	numerator			(reserved)
4	Velocity encoder resolution denominator	Integer	1 to $(2^{31} - 1)$	0x6090.2 (reserved)
5	Gear ratio numerator	Integer	1 to $(2^{31} - 1)$	0x6091.1
6	Gear ratio denominator	Integer	1 to $(2^{31} - 1)$	0x6091.2
7	Feed constant numerator	Integer	1 to $(2^{31} - 1)$	0x6092.1
8	Feed constant denominator	Integer	1 to $(2^{31} - 1)$	0x6092.2
9	Velocity factor numerator	Integer	1 to $(2^{31} - 1)$	0x6096.1
10	Velocity factor denominator	Integer	1 to $(2^{31} - 1)$	0x6096.2
11	Acceleration factor numerator	Integer	1 to $(2^{31} - 1)$	0x6097.1
12	Acceleration factor denominator	Integer	1 to $(2^{31} - 1)$	0x6097.2

References

AC, **DC**, **SP**, **VH[2]**, **FS**, **SD**

FF[] – Feed Forward

FF[] specifies the feed forward configuration and is used to improve control performance.

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Restrictions	None
Range	See the table below.
Index Range	1 to 5
Default	See the table below.
Unit modes	See the table below.
Non-volatile	Yes

Remarks

Feed forward is available for velocity and for current.

To find the value of **FF[1]**, you can reset this value and record a motion. **FF[1]** is equal to the current command (during acceleration) divided by the profile acceleration. It is better to take half of this value.

The actual value of the velocity feed forward is the derivative of the position command multiplied by **FF[2]**.

FF[3] is the ratio between resolution of velocity sensor and the resolution of position sensor. When only one sensor is used, this value should be equal to 1.

FF[4] is used for phase advance in brushless motors. It extrapolates the phase according to the velocity.

FF[5] is used for additional current proportional to velocity in open loop two phase motor.

Indices

The following table details the **FF[]** entries:

Index	Description	Default	Values	Restrictions
0	Reserved			
1	Specifies how much of the second derivative of the position reference (or the first derivative of the velocity reference) is fed as a reference to the current controller. For open loop stepper specifies the factor of additional gain proportional to acceleration. In amperes per acceleration (counts per second^2).	0	0 to 2000	In position and velocity modes (See **OV[2]**)
2	Specifies the factor of velocity feed forward added to the position controller output in the velocity command.	1	0 to 1	In position modes (See **OV[2]**)
3	Specifies the ratio between the velocity sensor resolution and the position sensor resolution. The socket number for the velocity sensor is the value of **CA[46]**. The socket number for the position sensor is the value of **CA[45]**.	1	>=0	In position modes (See **OV[2]**)
4	Specifies the phase advance in brushless motors. When this value is set to 1, the phase advance is an extrapolation to half TS (where the voltage command is outputted) according to the velocity.	0	-30 to 30	When FF[4]<0, phase advance occurs only on current reading. When FF[4]>0, phase advanced occurs on current reading and also on voltage output.
5	For open loop stepper specifies additional current proportional to speed.	0	0 to 200	In stepper open loop mode

Examples

Example 1

Suppose that there is a gear motor with a reduction ratio of 5 drives per load. The motor has an encoder with 1000 lines. The motor speed is used for the inner feedback loop. The load position measured by an encoder with 2000 lines is used as feedback for the outer loop. To prevent a steady-state error at constant speed, set the following: **FF[2]** = 1;

$$\mathrm{FF}[3] = \frac{1000 * 5}{2000} = 2.5 .$$

Example 2

Suppose that you want to add feed forward of acceleration into the current controller. You know that one ampere will cause an acceleration of 1,000,000 counts/sec^2 . Then you need to set the following: $\mathrm{FF}[1] = \frac{1}{1{,}000{,}000} = 1.0e-6$

References

UM, **CA[]**

FP[] – Feedback Position

FP*[N]* specifies the position of the feedback which is associated with socket *N*.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	The motor must be off.
Range	None
Index Range	1 to 4
Unit modes	All
Non-volatile	No

Remarks

Each feedback can be mapped to a socket which can then be referenced using the socket index. **FP[*N*]** is the position of the feedback which is mapped to socket *N*.

The socket always returns a value in physical units (not user units).

On power-up the socket is reset to its feedback value. In case of absolute feedback the feedback is read, and the socket gets the relevant value. In all other cases the position is set to 0.

PX returns the value of the position socket as defined in **CA[45]**.

The main position feedback must be set when the motor is disabled (by the **PX** or **FP** command). In this case it is recommended to use the **PX** command and not the **FP[*N*]** command, because the **PX** command also synchronizes the profiler and the position actual value (object 0x6064).

When working with a gantry, homing should set all three sockets: the slave and master by the **FP[*N*]** command and the main feedback for position (average of the two) by the **PX** command.

Indices

The following table describes the **FP[*N*]** entries.

Index	Description	Notes
1	Position of the sensor in socket number 1	Counts
2	Position of the sensor in socket number 2	Counts
3	Position of the sensor in socket number 3	Counts
4	Position of the sensor in socket number 4	Counts

References

CA[], **PX**, **FV[]**

FS – PTP Final Speed

FS specifies the configured velocity, which the drive will have on reaching the target position.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	Effective on the next call to **BG**
Range	0 to 2e9
Default	0
Unit modes	**UM** = 5 & Profile Position mode (0x6041 = 1)
Non-volatile	Yes

Remarks

The **FS** command defines the final speed (or end speed), at which the drive will continue to jog after reaching the position target.

FS is part of the profiler command and is calculated on the **BG** command or, when DS 402 is used, on the rising edge of bit 4 in the control word when the motion mode is the Profile Position mode.

Note: In cases in which **FS** is not 0, after reaching the position, the drive will jog at the **FS** value. Setting **FS** to 0 and performing another **BG** will cause the profiler to reevaluate the profile and cause a movement to the last **PA** command.

FS is important for bland movement as defined in PLCopen.

Setting **FS** to 0 and then calling **BG** do not guarantee that the motion will stop. On the contrary, if the target position was not set correctly, the motor will spin backwards.

FS overrides object 0x6062 on **BG**.

The **FS** value can be given in user-defined velocity units.

The **FS** value can also be given in user-defined units specified by the **FC** command.

References

PA, **BG**

FT[] – Float Trigger

FT[] specifies the floating point trigger for the recorder.

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Restrictions	Recorder inactive (**RR** = 0 or **RR** = -1)
Range	As in the table below
Index range	1, 2
Default	0
Unit modes	All
Non-volatile	No

Remarks

The **FT[]** command allows capturing of floating point triggers for the drive recorder.

The value is typically used by the EAS recorder.

Indices

The following table describes the **FT[]** entries.

Index	Description	Type	Values	Restrictions
0	Reserved			
1	Set for positive slope	Float	Float range	
2	Set for window	Float	Float Range	

References

RP[5], **RP[6]**

FV[] – Feedback Velocity

FV[*N*] reads the velocity of the feedback which is associated with socket *N*.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read-only
Source	All
Restrictions	For non-active sockets return zero
Range	Integer
Index Range	1 to 4
Unit modes	All
Non-volatile	No

Remarks

Each feedback can be mapped to a socket which can then be referenced using the socket index. **FV[*N*]** is the velocity of the feedback which is mapped to socket *N*.

The socket always returns a value in physical units (not user units).

VX is the same as **FV[*N*]**, where *N* is the socket of the velocity feedback or position feedback, depending on the DS-402 polarity object (object 0x607E or **OV[14]**).

Indices

The following table describes the **FV[*N*]** entries.

Index	Description	Notes
1	Velocity of socket number 1	Counts/sec
2	Velocity of socket number 2	Counts/sec
3	Velocity of socket number 3	Counts/sec
4	Velocity of socket number 4	Counts/sec

References

CA[], **VX**, **FP[]**

GI[] – Capture Input MUX Selection

GI[] routes digital inputs into the Strobe and Index inputs of quadrature module 0 or 1 of the drive.

CANopen/CoE

Attributes

Attribute	Description
Type	Unsigned integer, Read/Write
Source	RS232, USB, TCP, EoE
Restrictions	Socket number (bits 8:15) 1 to 4
Range	See the restrictions entry below.
Index range	1 to 4
Default	Refer to the **GI[*N*]** command format table below.
Unit modes	All
Non-volatile	Yes
Attribute	None

Remarks

The Gold drive includes two quadrature encoder modules Quad 0 (Port B) and Quad 1(Port A). These modules are used, among other things, to capture the position of an input signal, and are mainly used to count the position and calculate the speed of AQB sensors. The quadrature modules include 2 hardware signals: Index and Strobe. These signals are basically used for homing and position capturing abilities.

The **GI[*N*]** command enables routing of general purpose digital inputs and sensors (motor index signal) into the Index and Strobe inputs of the drive's quad modules. This allows any position capturing from various types of inputs.

The Strobe of Quad module 0 or/and Quad module 1 can be connected to one of the following for position capturing and homing purposes:

- First motor Index output
- Second motor Index output
- Gold drive general purpose digital Inputs 1 to 6

Note:

Different drives may have different number of available digital inputs. Consult the specific drive's User Guide for details. The drive should not prevent any setting of a non-existing input.

The command includes the input to be routed and the Quad module to which it is routed.

GI values are saved to non-volatile memory. At power-up the routing of the signals is evaluated according to **GI[]** value.

This command must be configured before using DS-402 homing, DS-402 touch probe, and in some cases of Elmo's legacy homing (**HM**,**HF**) sequences.

Bits 8 to 15, which were previously used to select the socket number, are ignored for compatibility reasons.

Indices

The following table details the **GI[]** options (for the actual values, see the second table below):

<table>
<tr><th>Index</th><th>Description</th><th>Default</th><th>Value/Options</th></tr>
<tr><td>0</td><td>Reserved</td><td>None</td><td>None</td></tr>
<tr><td>1</td><td>Quad Module 0 (Port B)
Index routing</td><td>0:
Port B index is used for capture</td><td rowspan="4">For GCON based drives (GWHI, GTRO, GDRU):
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>Quad module 0. Index is routed for the Index entry (GI[1] and GI[3]) or Strobe entry (GI[2] and GI[4])</td></tr>
<tr><td>1</td><td>Quad module 1. Index is routed for the Index entry (GI[1] and GI[3]) or Strobe entry (GI[2] and GI[4])</td></tr>
<tr><td>2 - 7</td><td>Digital inputs 1 to 6 are routed for the Index entry (GI[1] and GI[3]) or Strobe entry (GI[2] and GI[4])</td></tr>
</table>
For *SCORE based drives (GGUI):
GI[1] and GI[3] cannot be modified. Any value will result in 0 without error.</td></tr>
<tr><td>2</td><td>Quad Module 0.
Strobe signal routing</td><td>7:
Input 6 selected as strobe</td></tr>
<tr><td>3</td><td>Quad Module 1 (Port A).
Index routing</td><td>1:
Port A index is used for capture</td></tr>
<tr><td>4</td><td>Quad Module 1
Configured for:
Strobe signal input</td><td>6 :
Input 5 selected as strobe</td></tr>
</table>

*** Note for SCORE:**

Only a strobe entry in GI[2] & GI[4] can be modified. The default of GI[2] & GI[4] is 0 where that Strobe is used as capture for the homing mode. This allows normal operation without modifying the GI[] value.

GI[*N*] command format:

Bits	Value Description
0 to 7	Mux input: 0 – Quad-0 (Port B) + index 1 – Quad-1 (Port A)+ index 2 – Input-1 3 – Input-2 4 – Input-3 5 – Input-4 6 – Input-5 7 – Input-6 8 – Input-7 – Not supported in this version 9 – Input-8 – Not supported in this version 10 – Input-9 – Not supported in this version 11 – Input-10 – Not supported in this version 12 – Input-11 – Not supported in this version 13 – Input-12 – Not supported in this version 14 – Input-13 – Not supported in this version 15 – Input-14 – Not supported in this version 16 – Input-15 – Not supported in this version 17 – Input-16 – Not supported in this version
8 to 15	Ignored

References

HM[], **HF[]**, DS-402 Homing Mode, Touch-Probe

GO[] – Output Source

GO[] routes digital output into the Strobe output of quadrature module 0 or 1 of the drive.

CANopen/CoE

Attributes

Attribute	Description
Type	Unsigned integer, Read/Write
Source	RS232, USB, TCP, EoE
Restrictions	None
Range	**GO[1]** to **GO[4]**: 0 to 2 **GO[14]** to **GO[15]**: 0 to 5 (6 is reserved for compatibility)
Index range	1 to 4 14 to 16
Default	Refer to the table below.
Unit modes	All
Non-volatile	Yes
Attribute	None

Remarks

The drive includes two quadrature encoder modules, Quad 0 (Port B) and Quad 1 (Port A). These modules are used, among other things, to generate pulses according to position reading.

The drive includes port C output that is used (among other things) for feedback emulation, daisy chain and gantry functions.

Each quadrature module includes a Strobe hardware signal. This signal is used to generate pulses according to a given position.

The **GO[*N*]** command enables routing between the digital output (denoted by *N*) and the quad module's strobe output.

This command allows the user to select the following for each of the supported outputs:

- General-purpose output. Here the output is controlled by the functionality defined in the **OL[*N*]** command.
- Quad module 0 Strobe. Here the Output Compare is activated on Quad module 0, and the output is routed to the Strobe pin.

- Quad module 1 strobe. Here the Output Compare is activated on Quad module 1, and the output is routed to the Strobe pin.
- Daisy chain Quad module 0. Here the output is connected to Quad module 0.
- Daisy chain Quad module 1. Here the output is connected to Quad module 1.
- Emulation. Here the output is connected to the encoder emulation output allowing either of the following emulation signals which are generated from port C according to the emulated feedback (socket):
  - AqB pulses
  - Pulse and Direction
  - Up Down format
  - Halls
- Gantry. Here the output is used by the gantry socket for master/slave communications.

In GCON-based products (e.g. Gold Whistle) outputs 1 to 4 and 14 to 16 are supported.

Unsupported outputs will return a Bad Index error.

Indices

The following table details each of the **GO[*N*]** output options:

<table>
<tr><th>Index/ Output#</th><th>Description</th><th>Default</th><th colspan="2">Value/Options</th></tr>
<tr><td>0</td><td>Reserved</td><td>0</td><td colspan="2">None</td></tr>
<tr><td rowspan="4">1 to 4, 14 to 16</td><td rowspan="4">1 to 4
Denotes the functionality of the digital output

14 to 16
1. Denotes the functionality of the digital output
2. Routes the function of Port C for emulation, daisy chain or gantry.</td><td rowspan="4">0</td><td>Value</td><td>Description</td></tr>
<tr><td>0</td><td>General-purpose Output (GPO) according to OL[i] functionality (i: output number). (If selected, output does not participate in Output Compare or emulation)</td></tr>
<tr><td>1</td><td>Output Compare 1
Output is used as an output signal when position compare function (OC[]) is active on Quad 0 (Port B)</td></tr>
<tr><td>2</td><td>Output Compare 2
Output is used as an output signal when position compare function (OC[]) is active on Quad 1 (Port A)</td></tr>
</table>

14 to 16 only	**14 to 16** 1. Denotes the functionality of the digital output 2. Routes the function of Port C for emulation, daisy chain or gantry.	0	3	**Daisy chain 1** (DC1) Output is used to buffer AQB signals from Quad 0 (Port B) ***For index 14-16 only**
			4	**Daisy chain 2** (DC2) Output is used to buffer AQB signals from Quad 1 (Port A) ***For index 14-16 only**
			5	**Emulation** Output is used to emulate feedback according to EA[] ***For index 14-16 only**
			6	**Gantry** Output is used for Gantry communication (backwards compatibility) ***For index 14-16 only**
5 to 13	Reserved	0	None	

Notes

- Consult the Administrative Guide for the actual abilities and restrictions of the outputs.
- If the function denoted by **GO[]** is higher than the value 2 (i.e., Daisy Chain 0/1 or Emulation), all three relevant outputs of Port C (14, 15, 16) must be configured for the same function.
- If the function of **GO[14]** to **GO[16]** is not configured to Daisy Chain or Emulation, the drive function will fire blanks and no pulses will be generated.

References

GV[], **GW[]**, **OC[]**

GP[*N*] – Error Mapping Correction Table Editing

GP[*N*] edits entries in the error mapping correction table.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	RS232, USB, TCP, EoE
Restrictions	An error mapping correction table must be defined first with the **PC[3]** command.
Range	None
Index range	Depends on the selected table, see the **PC[3]** command.
Default	None
Unit modes	All
Non-volatile	Yes

Remarks

GP[*N*] enables editing of the error mapping correction table.

After setting **PC[3]** to the requested array, which will be used as the error mapping correction table, the table can be filled with correction values as defined in Error Mapping manual.

The index range of the **GP[]** command depends on the table selected in **PC[3]** (for optional tables and sizes, see the **PC[3]** command).

It is recommended not to edit the selected array directly, but to use the **GP[]** command.

The **GP[]** command protects against deviations of the table index.

The **GP[]** command does not protect against editing of the table while error mapping is enabled. Ambiguity may occur when it is incorrectly used during feature activation (**PC[1]** differs from 0).

The **GP[]** values are in user units.

References

PC[]

GS[] – Gain Scheduling

GS[] defines the gain scheduling process of the position and velocity controller.

CANopen/CoE

N/A

Attributes

Attribute	Description
Type	Integer
Source	All
Restrictions	
Range	According to the table
Index range	1 to 20
Default	According to the array index, elsewhere is zero
Non-volatile	Yes
Activation	Immediate

Remarks

The Gold drives are scheduled according to the speed or according to the position. This scheduling may be necessary due to either the difference between the low-speed behavior and the high-speed behavior of the plant or a lack of feedback information in low speed, or due to mechanical changes with position dependence. The process of assessing the situation and varying the controller parameters online accordingly is called “gain scheduling”.

There are up to 63 controllers in the position/speed loop. You can use a specific controller or let the gain schedule be chosen automatically by the speed (feedback or command) or by the position of a socket. When gain scheduling is disabled, the controller (without the filters) is configured by the parameters **KP[3]**, **KP[2]**, and **KI[2]**. When gain scheduling is enabled, the controller is chosen from the 63 controllers in the **KG[*N*]** array.

The controllers can also be changed when the servo is enabled (**MO** = 1).

In general, the following applies:

The parameters **GS[2, 16, 17, 18]** define the gain scheduling mode of the controllers and filters.

The parameters **GS[1, 6, 8, 10]** are used when gain scheduling by speed is active.

The parameters **GS[18, 20]** and **CA[65]** are used when gain scheduling by position is active.

The **GS[*N*]** array is normally programmed by the EAS. Manipulate it only if you are sure of what you are doing.

Indices

The following table lists the gain scheduling parameters. Unused indices are reserved for compatibility with older drives.

<table>
<tr><th>Index</th><th colspan="2">Description</th><th>Default</th><th>Values</th><th>Restrictions</th></tr>
<tr><td>0</td><td colspan="2">Reserved</td><td></td><td></td><td></td></tr>
<tr><td>1</td><td colspan="2">When GS[2] defines the minimum speed used in gain scheduling by speed. Below this speed the gain schedule will choose the first (low-bandwidth) controller. The maximum speed for gain scheduling is defined by GS[8]. Units are counts/sec.

When GS[2]=66, defines the threshold profile speed below which the algorithm assumes that the profiler stopped and that a different controller is set.</td><td>100</td><td>>0</td><td></td></tr>
<tr><td rowspan="6">2</td><td colspan="2">Use scheduled controller according to the following:</td><td rowspan="6"></td><td rowspan="6">0 to 66</td><td rowspan="6"></td></tr>
<tr><td>0</td><td>No gain scheduling (by KP[2], KI[2], KP[3])</td></tr>
<tr><td>1 to 63</td><td>Specific controller from table</td></tr>
<tr><td>64</td><td>Gain scheduling by speed (use GS[7] to set the speed source and position error, which are multiplied by GS[10])</td></tr>
<tr><td>65</td><td>Gain scheduling by position. Use CA[65] to set the position socket and GS[19], GS[20] to set the position limits.</td></tr>
<tr><td>66</td><td>Three controllers by profiler.
1. During profiler – index 63
KI=KG[63]
KPvel=KG[126]
KPpos=KG[189]
2. After the profiler stops and the time is less than GS[11] (ms) – index 62.
KI=KG[62]
KPvel=KG[125]
KPpos=KG[188]</td></tr>
</table>

	3. The time is **GS[11]** (ms) or more after the profiler stops – index 61 KI=**KG[61]** KPvel=**KG[124]** KPpos=**KG[187]**			
3	With Quad encoders, select the minimum speed for calculating speed as 1/T. Above this speed the calculation is by 1/T. Units are counts/sec. With Hall sensors only, any value differing from zero will not calculate speed by 1/T.		>=0	
4	Upward gain of gain scheduling filter. Units are Hz.	1500	100 to 3000	
5	Downward gain of gain scheduling filter. Units are Hz.	1500	100 to 3000	
6	Defines the maximum speed used in gain scheduling by speed. Above this speed the gain schedule will choose the last (high-bandwidth) controller. The minimum speed for gain scheduling is defined in **GS[1]**. Units are counts/sec.	62	>0	
7	Speed source for gain scheduling by speed 0 – By command 1 – By feedback		0, 1	
8	Reserved			
9	Non-linear factor for position controller. This value limits the position controller output to a specific acceleration to close the position error. Abs(**KP[3]***Error) <= sqrt(2***GS[9]***abs(Error)). Units are counts/sec^2.	2×10^9	0 to 2×10^9	
10	With **GS[2]**=64. Position error coefficient for position gain scheduling to raise gains. Actual speed for gain scheduling is **GS[10]***abs(Error)+abs(speed).	54	0 to 1,200	

<table>
<tr><td></td><td colspan="2">Units are rad/sec.</td><td></td><td></td><td></td></tr>
<tr><td>11</td><td colspan="2">With GS[2]=66. Time limit for gain scheduling of three controllers by the profiler.
Sets the time after the profiler has stopped (in milliseconds) when the second controller will be used, and afterwards the third controller is used.</td><td>0</td><td>0 to 8000</td><td></td></tr>
<tr><td>12 to 15</td><td colspan="2">Reserved</td><td></td><td></td><td></td></tr>
<tr><td rowspan="7">16</td><td colspan="2">Use scheduled gains in velocity advanced filter #10, to disable zero KV[25]</td><td rowspan="7">1</td><td rowspan="7">1 to 66</td><td rowspan="7"></td></tr>
<tr><td>1 to 63</td><td>Specific controller from table</td></tr>
<tr><td>64</td><td>Gain scheduling by speed (use GS[7] to set the speed source)</td></tr>
<tr><td>65</td><td>Gain scheduling by position (use CA[65] to set the position socket)</td></tr>
<tr><td>66</td><td>Three controllers by the profiler.
1. During profiler operation – index 63.
2. After the profiler stops and the time is less than GS[11] (ms) – index 62.
3. The time is GS[11] (ms) or more after profiler stops – index 61.</td></tr>
<tr><td colspan="2">Use KV[25] to set the type of this filter (and also to cancel it).
The 63*4 parameters are set in KG[190] to KG[441].</td></tr>
</table>

<table>
<tr><td rowspan="6">17</td><td colspan="2">Use scheduled gains in velocity advanced filter #2, to disable zero KV[30]:</td><td rowspan="6">1</td><td rowspan="6">1 to 66</td><td rowspan="6"></td></tr>
<tr><td>1 to 63</td><td>Specific controller from table</td></tr>
<tr><td>64</td><td>Gain scheduling by speed (use GS[7] to set the speed source)</td></tr>
<tr><td>65</td><td>Gain scheduling by position (use CA[65] to set the position socket)</td></tr>
<tr><td>66</td><td>Three controllers by profiler.
1. During profiler operation – index 63.
2. After the profiler stops and the time is less than GS[11] (ms) – index 62.
3. The time is GS[11] (ms) or more after the profiler stops – index 61.</td></tr>
<tr><td colspan="2">Use KV[30] to set the type of this filter (and also to cancel it).
The 63*4 parameters are set in KG[442] to KG[693].</td></tr>
</table>

18	Use scheduled gains in position advanced filter, to disable zero **KV[50]**: 1 to 63 – Specific controller from table 64 – Gain scheduling by speed (use **GS[7]** to set the speed source) 65 – Gain scheduling by position (use **CA[65]** to set the position socket) 66 – Three controllers by profiler. 1. During profiler operation – index 63. 2. After the profiler stops and the time is less than **GS[11]** (ms) – index 62. 3. The time is **GS[11]** (ms) or more after the profiler stops – index 61. Use **KV[50]** to set the type of this filter (and also to cancel it). The 63*4 parameters are set in **KG[694]** to **KG[945]**.	1	1 to 66	
19	First position boundary for gain scheduling by position. This value together with **GS[20]** defines the position that is divided by 63 for gain scheduling. Units are counts.			
20	Second position boundary for gain scheduling by position. This value together with **GS[19]** defines the position that is divided by 63 for gain scheduling. Units are counts.			

Notes

When setting the gain schedule filter (**KV[25]**, **KV[30]** and **KV[45]**), all the filters in the table are checked and set. An output error is only produced depending on the relevant indexes in (**GS[16]**, **GS[17]** and **GS[18]**). It is therefore advisable to first set GS and then KV.

References

CA[], **KG[]**, **KV[]**

GV[*N*] – Output Compare Editing Table

GV[N] edits entries in the position table for output compare in module 0.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	RS232, USB, TCP, EoE
Restrictions	The output compare table must be defined first in **OC[7]**.
Range	None
Index range	Depends on the selected table, see the **OC[7]** command.
Default	None
Unit modes	All
Non-volatile	No

Remarks

The **GV[]** command enables editing of the compare positions table for output compare table based modes regardless of the selected table.

After setting **OC[7]** to the requested array to be used as the position compare table, the table can then be filled with compare positions as defined in the output compare manual.

The index range of the **GV[]** command depends on the table selected in **OC[7]** (for the optional tables and sizes, see the **OC[7]** command).

It is recommended not to edit the selected array directly, but to use the **GV[]** command.

The **GV[]** command protects against deviation of the table index.

The **GV[]** command does **NOT** protect against editing the table while output compare – 0 is operational, i.e., **OC[1]** can report 1 or 2.

The **GV[]** values are in user units.

References

OC[], **GO[]**, **GW[]**

GW[*N*] – Output Compare Editing Table

GW[*N*] edits entries in the position table for output compare in module 1.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	RS232, USB, TCP, EoE
Restrictions	The output compare table must be defined first in **OC[27].**
Range	None
Index range	Depends on the selected table, see the **OC[27]** command.
Default	None
Unit modes	All
Non-volatile	No

Remarks

The **GW[]** command enables editing of the compare positions table for output compare table based modes.

After setting **OC[27]** to the requested array to be used as the position compare table, the table can then be filled with compare positions as defined in the output compare manual.

The index range of the **GW[]** command depends on the table selected in **OC[27]** (for the optional tables and sizes, see the **OC[27]** command).

It is recommended not to edit the selected array directly, but to use the **GW[]** command.

The **GW[]** command protects against deviation of the table index.

The **GW[]** command does **NOT** protect against editing the table while output compare – 1 is operational, i.e., **OC[21]** can report 1 or 2.

The **GW[]** values are in user units.

References

OC[], **GO[]**, **GV[]**

GX[] – Capture Array Value from HM

GX[] retrieves captured values from the capture array defined by **HM**.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read-only
Source	RS232, USB, TCP, EoE
Restrictions	The captured buffer must be defined by the **HM[11]** command.
Range	None
Index range	**HM[12]** to **HM[13]**
Default	None
Unit modes	All
Non-volatile	No

Remarks

During position capture, the set of captured positions can be defined by **HM[11]** in an array. In that case the **GX[]** command can be used to retrieve the position inputs.

This command reads the captured array, saving the user the need to know how the software stores the captured values in the array.

Note: If capture to array is selected (1 <= **HM[11]** <= 5), the user can read the captured values from the array via the **GX[]** command.

If **HM[]** is not configured, **GX[]** will return an error. Otherwise, it will return the value that is currently in the selected array, according to the specified index.

The next index to be filled is indicated in **HM[9]**.

References

HM[]

GY[*N*] – Capture Array Value from HF

GY[*N*] retrieves captured values from the capture array defined by **HF**.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read-only
Source	RS232, USB, TCP, EoE
Restrictions	The captured buffer must be defined by the **HF[11]** command.
Range	None
Index range	**HF[12]** to **HF[13]**
Default	None
Unit modes	All
Non-volatile	No

Remarks

During position capture, the set of captured positions can be defined by **HF[11]** in an array. In that case the **GY[]** command can be used to retrieve the position inputs.

This command reads the captured array, saving the user the need to know how the software stores the captured values in the array.

Note: If capture to array is selected (1 ≤ **HF[11]** ≤ 5), the user can read the captured values from the array via the **GY[]** command.

If **HF[]** is not configured, **GY[]** will return an error. Otherwise, it will return the value that is currently in the selected array, according to the specified index.

The next index to be filled is indicated in **HF[9]**.

References

HF[]

HL[] /LL[] – High/Low Feedback Limit (Reserved)

HL[]/LL[]

CANopen/CoE

Attributes

Attribute	Description
Type	
Source	
Restrictions	
Range	
Index range	
Default	
Unit modes	
Non-volatile	

Indices

The following table describes the **HL[]/LL[]** entries.

Index	Description	Type	Values	Restrictions
0				
1				
2				

References

HM[*N*]/HF[*N*] – Main/Aux Homing

HM[*N*] /HF[*N*] enables the ability to capture input events and to execute a predefined operation when an event occurs.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer
Source	USB, RS232, TCP, EoE
Restrictions	According to array index
Range	According to array index
Index range	1 to 13
Default	None
Unit modes	All
Non-volatile	No
Attribute	None

Remarks

The **HM[*N*]** command defines the main homing parameters, always work on the main feedback sensor.

The **HF[*N*]** command defines the auxiliary (Aux) homing parameters. The sensor is selected by the user (see **HF[10]**).

Note: The following text refers to the **HM[*N*]** command. It also applies to the **HF[*N*]** command unless stated otherwise.

The command sets and gets parameters of the Main/Aux homing and capture process, which the drive uses to set a trap for a user-defined event. When the event occurs, the drive can perform one of the following tasks:

1. Modify the main feedback position counter.
2. Log the event position counter.
3. Flag a digital output.

An event is a change in a digital input signal. The polarity of the change is defined by the **IL[]** command.

Currently a drive supports two types of capture accuracy:

1. Index, Home. Fast accurate HW capture. Using these inputs must be configured by the **GI** command.
2. All other inputs. SW capture with a delay configured in the **IF** command.

Values defined in **HM[3]** are duplicated for compatibility reasons.

A drive currently supports six inputs. If **HM[3]** >= 21, the behavior will be unpredictable.

All **HM[*N*]** entries that are not read-only, except **HM[1]**, can be changed during the homing procedure. The activation of these values will be performed at the next homing activation, that is, when **HM[1]** != 0.

If **HM[2]** is set to a value beyond **XM[1]** and **XM[2]**, the actual main position will not be updated when homing is complete. (Feature not available at this time)

Each homing event is attached to a predefined functionality (FLS, RLS, general-purpose, home, and so on). If the corresponding input is not defined first, the homing procedure may never end. Refer to the **IL[*N*]** command.

IN5 and IN6 can be captured by HW. In order to use this feature, use the **GI[*N*]** command to direct IN5 or IN6 into home capture, and configure the homing parameters to a home event. Refer to the **GI[*N*]** command.

The homing and capture procedures can be carried out in any unit mode (**UM** = 1, 2, 3, 4, 5).

- In external reference mode (**RM** = 1), when **HM[4]** = 0, the software portion of the reference is stopped, while the external portion is not. In such cases, the motor continues to move according to the analog reference. (Feature not available at this time)
- When capturing more than one event is configured, i.e., when **HM[1]** > 1, the delta between each event must be (4***TS**) for digital inputs, and (2***TS**) for home input or Index.
- When both **HM[*N*]** and **HF[*N*]** are run, and using the main feedback sensor is selected (**HF[10]** = 1), the results are unpredictable.
- When **HM[11]** = 5, capture is added to the data recording array, and data recording should not be used.
- If **HM[11]!** = 0 and **HM[1]**/**HF[1]** is finite and exceeds the number of possible entries in the array, the value of **HM[1]**/**HF[1]** value will be saturated to the following value: (**HM[13] –HM[12]** + 1).
- The capture delay is input type-dependent. For Index or Home inputs the delay is 2***TS**. For other inputs it is dependent on the configured input filter (see the **IF** command).
- DS-402 homing and DS-402 Touch Probe cannot work while **HM** or **HF** are running.

Indices

The following table describes the **HM[*N*]/HF[*N*]** entries.

Index	Purpose	Value	Description
0	Reserved	None	Reserved
1	Activation mode	0	Stop the homing process. **HM[1]** is automatically reset to 0 when homing is complete.
		≥1	Number of events. **HM[4]** is performed at last event.
		32000	Infinite number of event captures. **HM[4]** is ignored. If Capture to array is selected (see **HM[11]**), the array will be filled between **HM[12]** and **HM[13]** in an infinite loop.
2	Absolute /relative value		Value to load, according to the method specified in **HM[5]**. Absolute value is limited to the position counter range.
3	Event definition	0 default	Immediate: The trigger is the receipt of **HM[1] = 1**.
		1/2	Event according to main home switch capture. The first event is always high transition.(High transition - level change from low to high (rising edge)). The home switch is selected according to the **GI** command.
		3	High transition (High transition - level change from low to high (rising edge)) of index pulse (capture).
		4	Low transition (Low transition, level change from high to low (falling edge)) of index pulse (capture).
		5/6	Event according to the FLS switch.
		7/8	Event according to the RLS switch.
		9/10	Event according to the DIN1 switch.
		11/12	Event according to the DIN2 switch.
		13/14	Event according to the DIN3 switch.
		15/16	Event according to the DIN4 switch.
		17/18	Event according to the DIN5 switch.
		19/20	Event according to the DIN6 switch.
		21/22	Event according to the DIN7 switch.
		23/24	Event according to the DIN8 switch.
		25/26	Event according to the DIN9 switch.
		27/28	Event according to the DIN10 switch.
4	After event behavior. Executed when **HM[1]** reaches 0.	0 default	In **UM** = 1,2,3,4,5: stop immediately using the **SD** deceleration value.
		1	Set digital output. Equivalent to **OP** = **HM[6]**.
		≥2	Do nothing.

Index	Purpose	Value	Description
5	What to set for PX during event	0 default	Absolute setting of the position counter: **PX** = **HM[2]**.
		1	Relative setting of the position counter: **PX** = (**PX** at event) – **HM[2]**
		==2	**HM[5]** - Do nothing. **HF[5]** – Absolute setting of the sensor position counter to **HF[2]** value. In case the sensor is defined as Gantry, this will adjust all relevant Gantry sockets.
		>2	Do nothing.
6	Output value		Digital output value to set if **HM[4]** = 1. Only outputs defined as general outputs are affected.
7	Captured value (PX)		The capture value of **PX** (read-only). The position value is captured before **PX** is changed according to **HM[5]**.
8	Reserved	None	None
9	Next capture array index		The next index in the capture array for inserting the next captured value. The last valid captured value is: The captured value can be read via the **GX**/**GY** command.
10	Socket selection	1 to 4	Selects the socket to be used for **HF**. This command is available only in **HF[]**.
11	Capture array selection	1	The ZX array is used for capture in the range from 1 to 1023.
		2	The NT array is used for capture in the range from 1 to 255.
		3	The ET array is used for capture in the range from 1 to 2048.
		4	The UI array is used for capture in the range from 1 to 24.
		5	The BH array is used for capture in the range from 1 to 16383 (When this array is used, data recording is not available).
		All others	No array is selected.
12	Capture array low index		Low capture array index, to be filled with capture values.
13	Capture array high index		High capture array index, to be filled with capture values.

Examples

Example 1

The following example uses capture on the main home switch when input-5 is routed into encoder-1 strobe input by the **GI[]** command.

HM[1] will be set to 0 in case that the digital input 5 will be logically set to '1'.

If the example is used on **HF**, **HF[10]** must be configured (**HF[10]** = 1).

Command	Description
HM[1]=0	Disable the ongoing homing sequence
HM[2]=1000	Home position is been offset by 1000 counts from input
IL[5]=17	Input-5 is configured as homing switch, active high
GI[4]=6	Route input-5 into Encoder-1 strobe input
HM[3]=2	Wait for the event on home signal (the first rising edge)
HM[1]=1	Start searching for a single event

Example 2

The following example uses capture on the main home switch when input-5 is routed into encoder-1 strobe input by the **GI** command, and event values are added to the **BH** array infinitely.

If the example is used on **HF**, **HF[10]** must be configured (**HF[10]** = 1).

Command	Description
HM[1]=0	Disable the ongoing homing sequence.
HM[2]=0	Home position is offset by 0 from the strobe input.
IL[5]=17	Input 5 is configured as homing switch, active high.
GI[4]=6	Route input-5 into Encoder-1 strobe input.
HM[3]=2	Wait for the event on home signal (first signal is always a rising edge).
HM[11]=5	Select the **BH** array for event storage.
HM[12]=1	The low **BH** index is 1.
HM[13]=16000	The high **BH** index is 16000.
HM[1]=32000	Start capturing events and add them into the **BH** array. **HM[9]** will indicate the next **BH** array index to be filled. The values are read using the **GX** command.

Reference

GI, GX, GY, IL, IF

HP – Halt Program

HP halts the user program.

CANopen/CoE

Attributes

Attribute	Description
Type	Command
Source	All, except the user program
Restrictions	No
Range	None
Default	None
Unit modes	All
Non-volatile	No

Remarks

The command halts execution of the user program.

A subsequent **XC** command resumes the program from the instruction at which the program was halted. A pending Auto-Routine will remain pending.

An **HP** command issued when no program is running does nothing and sets no error.

The **XC** command resumes execution after a halt.

Program status (**PS**) is 0 when the user program is halted.

References

KL, **XQ**, **XC**

HT[] – Open Loop Torque

HT[] specifies the open loop torque for stepper 2-phase

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Range	According to array index
Index range	1 to 3
Default	0
Unit modes	UM=6 (open loop stepper)
Non-volatile	Yes

Remarks

The **HT[]** command sets the open loop torque for 3 different movement types: no movement, movement at certain speed without acceleration and movement with some acceleration.

In addition to the specified torque, there is an option to add gain proportional to the velocity or the acceleration using **FF[]** command.

Indices

The following table describes the available options for **HT[]**.

Index	Description	Units	Values	Restrictions
1	Holding torque if there is no speed and no acceleration	Amperes	**-CL[1] to CL[1]**	
2	Torque when there is speed and no acceleration. Receives the maximum value when between HT[1] and HT[2]. If HT[1] > HT[2] the maximum value obtained is HT[1].	Amperes	**-CL[1] to CL[1]**	

3	Torque when there is acceleration. Receives the maximum value when between HT[1] and HT[3]. If HT[1] > HT[3} the maximum value obtained is HT[1].	Amperes	**-PL[1] to PL[1]**	

References

CL[], **FF[]**, PL[], **UM**

HX – Hexadecimal Mode

HX specifies a parameter that allows hexadecimal numbers to be displayed, set and indicated.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	USB, TCP, RS232. Has no influence on other.
Restrictions	No
Range	0, 1
Default	0
Unit modes	All
Non-volatile	No

Remarks

The **HX** parameter allows a hexadecimal reply to the host when USB, RS232 and TCP are used. The format eases the reading and understanding of bit-field variables, such as the digital inputs port (**IP**), servo drive status (**SR**) and more.

When **HX** = 0, integers are reported as decimal numbers.
When **HX** = 1, integers are reported as hexadecimal numbers.

The **HX** parameter is not required for setting values. The commands `BH=1024` and `BH=0x400` are equivalent, as 0x400 equals its decimal equivalent 1024.

Floating point numbers cannot be presented in hexadecimal format.

IA[] – Index Analog Sensor

IA[] specifies an interrupt for capturing the index of the Sine/Cosine analog encoder and specifies the encoder position at the index location for fast index capture. This procedure should be performed once, when setting an analog encoder to a drive.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	Use for an analog sin/cos encoder with an index. This procedure should be managed by EAS.
Range	According to the table.
Index range	1 to 4
Default	0
Unit modes	Any
Non-volatile	Yes

Indices

The following table describes the **IA[]** entries.

<table>
<tr><th>Index</th><th colspan="2">Description when set</th><th colspan="2">Description when read</th><th>Values</th></tr>
<tr><td rowspan="3">1</td><td colspan="2">Enable/disable the index interrupt.</td><td colspan="2">The index interrupt is enabled/ disabled.</td><td rowspan="3">0, 1</td></tr>
<tr><td>0</td><td>Disable the interrupt</td><td>0</td><td>The index interrupt is disabled.</td></tr>
<tr><td>1</td><td>Enable the interrupt</td><td>1</td><td>The index interrupt is enabled.</td></tr>
<tr><td rowspan="3">2</td><td colspan="2">Clear the interrupt flag.</td><td colspan="2">The interrupt flag is cleared/ set.</td><td rowspan="3">0, 1</td></tr>
<tr><td>0</td><td>No effect</td><td>0</td><td>The interrupt flag is cleared.</td></tr>
<tr><td>1</td><td>Clear the interrupt flag.</td><td>1</td><td>The interrupt flag is set (the position was captured).</td></tr>
</table>

3	Set the angle at index rising edge interrupt.	Captured angle at index rising edge interrupt.	
4	Set the angle at index rising edge interrupt.	Captured angle at index rising edge interrupt.	

References

IB[] – Digital Input Bits

IB[] reads a digital input bit.

CANopen/CoE

0x60FD

Attributes

Attribute	Description
Type	Integer Indices 17 to 32 R/W, other Read-only
Source	All
Restrictions	None
Range	0, 1
Index range	1 to 32
Default	None
Unit modes	All
Non-volatile	No

Remarks

IB[1] to **IB[32]** reflect the **IP** register bits 0 to 31, respectively.

Refer to the **IP** command for more details about the digital input function and state.

The 'Write" function clears a fetched input (sticky bit). Refer to **IL[]** command "sticky bit" section.

Example

IB[1] reflects **IP** bit 0, indicating a general-purpose function state is active (1)or not active (0).

IB[17] reflects **IP** bit 16, indicating that digital input 1, regardless to its function, is active (1)or not active (0).

Indices

Index	Description	Type	Values	Restrictions
1	General purpose input is active	Integer	0..1	
2	Safety (o.k.)	Integer	0..1	
3	Main home switch	Integer	0..1	

4	Auxiliary home switch	Integer	0..1	
5	Soft stop	Integer	0..1	
6	Hard stop	Integer	0..1	
7	Forward limit (FLS)	Integer	0..1	
8	Reverse limit (RLS)	Integer	0..1	
9	INH (enable) switch	Integer	0..1	
10	Hardware BG (begin)	Integer	0..1	
11	Abort function	Integer	0..1	
12..16	Not used. Always zero.	0		
17..22	Digital input 1..6 logical pins state	Integer	0..1	Write 1 to clear sticky bit
23..30	Reserved. Always 0	0		
31..32	Digital input 15..16 logical pins state	Integer	0..1	Write 1 to clear sticky bit

References

IP, **IL[*N*]**

ID, IQ – Active/Reactive Current

ID and **IQ** get the active (**IQ**) and the reactive (**ID**) components of the motor current, in amperes.

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read-only
Source	All
Restrictions	None
Range	N/A
Index range	N/A
Default	N/A
Unit modes	All
Non-volatile	No

Remarks

A brushless motor carries alternating currents in its phases. The alternating currents in the motor phases create a rotating magnetic field, which can be projected in two directions. The first magnetic field component is aligned with the magnetic direction of the rotor; it produces no mechanical torque. The other magnetic field component is perpendicular to the magnetic direction of the rotor and produces all the mechanical torque.

IQ [ampere] is the component of the motor phase current that creates the effective torque. The current controller attempts to make **IQ** equal to the current command. **ID** is the component of the motor phase current that does not create torque. Usually the current controller tries to null **ID**.

When the motor is off (**MO** = 0), **IQ** and **ID** are not calculated and return 0.

References

- Language and User Program Manual: Chapter 10, "The Current Controller"
- **AN[*N*]**, **MC**, **PL[*N*]**, **CL[*N*]**

IF[] – Digital Input Filter

IF[] defines the time period of the digital input filter.

CANopen/CoE

N/A

Attributes

Attribute	Description
Type	Parameter, Float
Source	All
Restrictions	None
Range	0.0 to 500.0
Index range	1 to 16
Default	0 (no filter)
Unit modes	All
Non-volatile	Yes

Remarks

The **IF[]** is defined in milliseconds. Input pulses which are shorter than **IF[]** will be discarded. Inputs pulses which are longer than **IF[]** will be sensed.

The digital filter basic time is 250 µsec by default.

Note:
The actual number of digital inputs depends on the drive hardware. Typically, 6 inputs are available. The drive firmware will allow the inputs setting even if the drive's hardware is not available.

The input filter is a deterministic period function in the firmware. The actual filter time will be the time (milliseconds) which is closest, but not shorter, to the requested time. The resolution depends on the filter period, which might differ between versions.

Example

If the IF[*x*] is set to 1.1, the filter period will be 1.25 msec, which is the filter period closest to the requested value.
In cases in which the filter basic period is 100 µsec, the actual filter time will be 1.1 msec.
In any case, pulses which are shorter than the requested value will be rejected.

In cases in which the digital input is used for position capture (homing on home switch, or touch probe) the Software filter is not respected, since the position is captured by hardware, much faster than any firmware period.

Hardware filtering of inputs is not available at this stage.

For inputs that are not supported by the product, the relevant index will be accepted but ignored.

Indices

The following table describes the **IF[]** entries.

Index	Description	Type	Values	Restrictions
1 to 6	Digital input filter in msec	Float	0.0 to 500.0	

References

IB[], **IL[]**,**IP**

IL[] – Digital Input Logic

IL[] specifies the function and logic of the specified digital input.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	In the Gold Whistle all inputs can be assigned to the Home switch. In the Gold Guitar and Gold Trombone only input 5 can be used as the Home switch.
Range	N/A
Index range	1 to 15
Default	Input 1: 0 – Active low, Inhibit Input 2 to 6: 7 – General purpose, high logic level Input 8 to 16: 5 – Ignore
Unit modes	All
Non-volatile	Yes

Remarks

IL[*N*] is a bit field command which allows the user to map digital input *N* to a desired function and to determine the logic level at which the input will be active.

Each input in the drive can be mapped to a "built-in" function. This means that when the input is logically sensed, the function will be activated. The available built-in functions are described in **Function Description**.

The logic level defines the relation between the hardware connectivity of the digital input and the activation of this input.

Positive logic (*active high*) means that the input will be sensed if current flows through the input pin.

Negative logic (*active low*) means that the input will be sensed if no current flows though the input pin. This is normally used, for example, in brake or abort functions, where the user would typically like to prevent motion when no current flows to the drive input.

The number of inputs depends on the drive HW. The software, however, is designed for 6 inputs regardless of the hardware ability.

Sticky bit

If the input is defined as sticky bit (bit 8 of the **relevant IL[]** is set to '1'), the input is latched. Writing '1' to a latched input will clear the indication. This is useful for cases the host cycle time is slower than the input triggered time.

Note:
Digital inputs 11 to 16 are read from the differential encoder entries of Port A and Port B. In order to prevent the triggering of these signals, set **IL[11]** to **IL[16]** to " Ignore " (**IL[N]** = 4 or 5).

Bit-Field Entries

The following table describes the bit-field entries for logic and function in **IL[]**.

Bit	Description	Type	Values		Restrictions
0	Logic level	Boolean	0	Active low	
			1	Active high	
1 to 4	Function number and behaviors * See detailed description in **Function Description**.	Integer	0	Shut off the servo drive, freewheel.	
			1	Stop immediately under control: soft and auxiliary stop.	
			2	No function is attached. Ignore the switch.	
			3	General-purpose.	
			4	Hard-enable forward direction only (RLS).	
			5	Hard-enable reverse direction only (FLS).	
			6	Begin: activates the **BG** command.	
			7	Stop immediately under control: soft stop.	
			8	Enable the Main Home sequence.	
			9	Enable the Auxiliary Home sequence.	
			10	Stop immediately under control: stop both the software trajectory and auxiliary reference.	

Bit	Description	Type	Values		Restrictions
			11	Abort motion. Shut off the servo drive, freewheel.	
			12	Reserved for safety function compatibility.	
			13	Additional Abort motion. Function can be used to shut off the motion, produces a different fault code. (See **MF** command)	
			14	Engage ECAM/Follower: set **RM=1**	
			15	Disengage ECAM/Follower**:** set **RM=0**	
5 -7	Reserved				
8	Sticky function	Boolean	0	Non sticky	
			1	Sticky	
9 to 15	Reserved				

Possible Values for IL[*N*]

Command Value	Logic Level	When Active
IL[*N*] = 0	Low	Shut off the servo drive, freewheel. - Inhibit. **Note:** In the auxiliary reference, motion will be activated if the switch is disabled.
IL[*N*] = 1	High	Shut off the servo drive, freewheel - Inhibit. **Note:** In the auxiliary reference, motion will be activated if the switch is disabled.
IL[*N*] = 2	Low	Stop immediately under control: soft and auxiliary stop.
IL[*N*] = 3	High	Stop immediately under control: soft and auxiliary stop.
IL[*N*] = 4	Low	No function is attached. Ignore the switch.
IL[*N*] =5	High	No function is attached. Ignore the switch.
IL[*N*] = 6	Low	General-purpose.
IL[*N*] = 7	High	General-purpose.
IL[*N*] = 8	Low	Reverse limit switch (RLS). Only forward motion is allowed.

Command Value	Logic Level	When Active
IL[*N*] = 9	High	Reverse limit switch (RLS). Only forward motion is allowed.
IL[*N*] = 10	Low	Forward limit switch (FLS). Reverse motion is allowed.
IL[*N*] = 11	High	Forward limit switch (FLS). Reverse motion is allowed.
IL[*N*] = 12	Low	Begin: activates the **BG** command.
IL[*N*] = 13	High	Begin: activates the **BG** command.
IL[*N*] = 14	Low	Stop immediately under control: soft stop only. Activates the **ST** command.
IL[*N*] = 15	High	Stop immediately under control: soft stop only. Activates the **ST** command.
IL[*N*] = 16	Low	Enable the Main Home sequence. *N* can be 5 only. (N/A for the Gold Whistle)
IL[*N*] = 17	High	Enable the Main Home sequence. *N* can be 5 only. (N/A for the Gold Whistle)
IL[*N*] = 18	Low	Reserved
IL[*N*] = 19	High	Reserved
IL[*N*] = 20	Low	Stop immediately under control: stop both the software profiler and the auxiliary reference.
IL[*N*] = 21	High	Stop immediately under control: stop both the software profiler and the auxiliary reference
IL[*N*] = 22	Low	Abort motion. Shut off the servo drive, freewheel
IL[*N*] = 23	High	Abort motion. Shut off the servo drive, freewheel
IL[N]=24	Low	Additional Abort motion. Shut off the servo drive, freewheel
IL[N]=25	High	Additional Abort motion. Shut off the servo drive, freewheel
IL[*N*] = 28	Low	Set RM =1; Engage Follower and ECAM
IL[*N*] = 29	High	Set RM =1; Engage Follower and ECAM
IL[*N*] = 30	Low	Set RM =0; Disengage Follower and ECAM
IL[*N*] = 31	High	Set RM =0; Disengage Follower and ECAM
IL[*N*] = 262	Low	General-purpose sticky input
IL[*N*] = 263	High	General-purpose sticky input

Function Description

Function 0: Inhibit (freewheel)

Servo drive is off (**MO** = 0). The motor is not under control. No current is applied through the motor phases. If the motor was previously running, it will continue to coast on its own inertia.

The motor fault code (see the **MF** command) is 0x10. If an external command is active (**RM** = 1), a motor restart will be attempted when the switch is "not active." This attempt is made within a few milliseconds during the background task of the drive.

In addition, when the motor is restarted, the #@AUTO_ENA automatic routine, if declared in a User Program, will be activated.

Warning: Use the Inhibit freewheel function with care. When the drive is shut off, the motor applies no torque. Turning off a drive might leave the motor spinning until it stops by friction. In some situations, this may be dangerous.

Function 1: Hard stop immediately under control

The drive will stop all auxiliary motion (e.g., ±10 V analog reference, follower, ECAM etc.) in the fastest possible way. If **UM** = 1 (current control), the torque command is set to 0 immediately. In any other unit modes (velocity and position) the drive will stop using the **SD** command.

The #@AUTO_STOP automatic routine, if declared in a User Program, will be activated.

When this digital input is changed to its not active state, the Hard Stop situation is terminated.

Function 2: Input is ignored

This serves no function in the drive and always reads zero in the **IP**/**IB[*N*]** indications.

Function 3: General purpose

The purpose of this function is to allow the user general use of the input. The relevant input entry will be signaled in the **IP** or **IB[*N*]** command. With the use of the User Program, the user can perform any desired action, for example, signaling an output.

If an #@AUTO_IN routine is declared in the User Program, the routine will be automatically called. For example, if digital input 3 is declared as general-purpose with active low logic, #@AUTO_IN3 will be called if no current flows though input 3 pin.

Function 4: Reverse limit switch (RLS: forward only)

When this function is active, reverse motion is not available, and any reverse command will be discarded by the Stop Manager.

The word "reverse" refers to when the current and the velocity commands have negative values.

If the motion is in the reverse direction during activation of the function, the motion will be stopped according to the following:

If **UM** = 1 (current control), the torque command is set to 0 immediately.

If **UM** = 2 or **UM** = 5 (velocity and position modes), the drive will stop using the **SD** command.

If an #@AUTO_RLS routine is declared in the User Program, the routine will be called automatically.

This function does not change the drive's reference command. When the switch is released, the reference command (speed or position) is recovered.

Function 5: Forward limit switch (FLS)

When this function is active, forward motion is not available and any forward command will be discarded by the Stop Manager.

The word "forward" refers to when the current and the velocity commands have positive values.

If the motion is in the forward direction during activation of the function, the motion will be stopped according to the following:

If **UM** = 1 (current control),the torque command is set to 0 immediately.

If **UM** = 2 or **UM** = 5 (velocity and position modes), the drive will stop using the **SD** command.

If an #@AUTO_FLS routine is declared in the User Program, the routine will be called automatically.

This function does not change the drive's reference command. When the switch is released, the reference command (speed or position) is recovered.

Function 6: Begin

This function behaves like a software **BG** command, i.e., it starts the programmed motion. See the **BG** command for more details.

If an #@AUTO_BG routine is declared in the User Program, the routine will be called automatically.

Function 7: Software Stop

Stops all software reference.

Not supported.

Function 8: Main Home switch

This function can be used as the Homing Switch in the Homing/Capture process. For that purpose, in addition to the configuration of a digital input as Main Home Switch (digital input 5 only), the user will use the **GI[*N*]** command in order to connect the Home Switch digital input to the Drive Homing Module (refer to the **GI[*N*]** command).

This function activates the #@AUTO_HM routine in the user program, depending on its declaration.

Function 10: Hard and Soft stop

This function stops the motor under control, stopping the auxiliary reference and software reference.

This function activates the #@AUTO_STOP routine in the user program.

See Function 1: Hard stop immediately under control for further details.

Function 11: Abort motion

The behavior is similar to the Inhibit function with the exception that the “Abort” input release does not start the motor automatically. After the Abort is activated, **MO** = 1 must be set either by communication or by the internal User Program.

The function activates the #@AUTO_ER routine, if it exists, in the user program.

Function 12: Additional Abort motion

The function behaves similar to the Abort Motion. It allows the user to have a second freewheeling function and to distinguish between the source of the Abort. This can be used for e.g. PTC function. The Additional Abort reports a different MF (Motor Fault) value from the Abort motion.

Function 14: Engage ECAM/Follower

This function enables the ECAM/Follower functionality of the external reference generator (**RM=1**). The function does not execute the **EI** command.

Function 15: Disengage ECAM/Follower

This function disables the ECAM/Follower functionality of the external reference generator (**RM=0**).

References

IP, **IB[]**, **HM[]**, **MF**

IP – Input Port

IP reports the status of the digital inputs.

CANopen/CoE

0x60FD

Attributes

Attribute	Description	
Type	Bit Field, Read-only	
Source	All	
Range	Bit 0	General-purpose input is active
	Bit 1	Safety (OK)
	Bit 2	Main home switch
	Bit 3	Auxiliary home switch
	Bit 4	Soft stop
	Bit 5	Hard stop
	Bit 6	Forward limit switch (FLS)
	Bit 7	Reverse limit switch (RLS)
	Bit 8	Inhibit (enable) switch
	Bit 9	Hardware motion begin (**BG**)
	Bit 10	Abort function
	Bits 11 to 15	Not used. Always zero.
	Bit 16	Digital input 1 logical pin state
	Bit 17	Digital input 2 logical pin state
	Bit 18	Digital input 3 logical pin state
	Bit 19	Digital input 4 logical pin state
	Bit 20	Digital input 5 logical pin state
	Bit 21	Digital input 6 logical pin state
	Bits 22 to 25	Reserved. Always 0
	Bit 26	Digital input 11 logical pin state. Port A, A encoder entry
	Bit 27	Digital input 12 logical pin state. Port A, B encoder entry

Attribute	Description	
	Bit 28	Digital input 13 logical pin state. Port A, INDEX encoder entry
	Bit 29	Digital input 14 logical pin state. Port B, A encoder entry
	Bit 30	Digital input 15 logical pin state
	Bit 31	Digital input 16 logical pin state
Unit modes	All	

Remarks

The **IP** command reports the logic state and the activated function of the whole digital input port.

The command is divided in to two sections of 16 bits each:

- Bits 0 to 15 report the actual function which is active e.g. Reverse Limit Switch, Homing etc.
- Bits 16 to 31 report the logic level of the input where 1 means that the input is logically active regardless of the physical state. Writing "1" to bits 16 to 31 clears sticky bit (latched input), so command **IP**=IP will clear only the sticky bits and other bits which are not sticky do not change. Writing "0" is not allowed. For more information about "sticky bits" refer to the **IL[]** command.

For example:
If digital input 2 is configured as Forward Limit (FLS), input 4 is configured as Main Home Switch and both inputs become logically active, bits 2, 6, 17 and 19 will be set to 1. In this case the **IP** command returns 655428 or 0x000A0044.

Note:
Digital inputs 11 to 16 are read from the differential encoder entries of Port A and Port B. In order to prevent the triggering of these signals, set **IL[11]** to **IL[16]** to "ignore" (**IL[N]** = 4 or 5).

References

IB[*N*], **IL[*N*]**

JP – Jog Position

JP specifies the motor speed reference for jogging in the Profile Position mode.

CANopen/CoE

0x6081, 0x6082

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	The motor must be on.
Range	-2e9 to +2e9
Default	0
Unit modes	**UM** = 2; **UM** = 5
Non-volatile	No

Remarks

On the next **BG** after applying the **JP** command:

- The motion control will be switched to the position control loop.
- The motor will jog at the speed specified by **JP** according to **AC**, **DC** and **SF**.

When **JP** is set and **BG** is commanded, the motion mode reflected in **OV**[2] (object 0x6061) is modified to 1 (Profile Position).

Objects 0x6081 and 0x6082 will be overridden by the **JP** value. Refer to the **BG** command for more details.

JP jog in Position mode is not an endless motion like **JV** jog; therefore, it will stop at the software position limits (**VH[3]**, **VL[3]**).

JP can be higher than **VH[2]** (the velocity limit). In this case, the actual speed command will be saturated by **VH[2]**.

The motor will abort if the feedback speed is higher than **HL[2]** or lower than **LL[2]**.

The **JP** value can be given in user-defined units specified by the **FC** command.

References

PA, **SP**, **AC**, **DC**, **UM**, **JV**, **FC**

JV – Jog Velocity

JV specifies the motor speed reference for jogging in the Profile Velocity mode.

CANopen/CoE

0x60FF

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	• The motor must be on. • Effective on the next call to **BG**
Range	-2e9 to +2e9
Default	0
Unit modes	**UM** = 2; **UM** = 5
Non-volatile	No

Remarks

On the next **BG** after applying **JV** command:

- the motion control will be switched to the velocity control loop.
- the motor will jog at the speed specified by **JV** according to **AC**, **DC** and **SF**.

When **JV** is set and **BG** is commanded, the motion mode reflected in **OV[2]** (object 0x6061) is modified to 3 (Profile Velocity).

Object 0x60FF will be overridden by the **JV** value. Refer to the **BG** command for more details.

Jog is an endless motion, which does not halt at any of the software limits (**VH[3]**, **VL[3]**) or the software range modulo (**XM[1]**, **XM[2]**).

JV can be higher than **VH[2]** (the velocity limit). In this case, the actual Speed command will be saturated by **VH[2]**.

The motor will abort if the feedback speed is higher than **HL[2]** or lower than **LL[2]**.

The **JV** value can be given in user-defined units specified by the **FC** command.

References

PA, **SP**, **AC**, **DC**, **UM**

KG[] – Gain Scheduled Controller

KG[] specifies the parameters of the gain scheduled speed, position controller and advanced filters. The **KG[]** parameters apply only if the controller gains are scheduled (see the **GS[]** command).

CANopen/CoE

Attributes

Attribute	Description
Type	Real
Source	All
Restrictions	None
Range	See section 15.4 "The Gain Scheduling Algorithm" in the SimplIQ Software Manual
Index range	1 to 504
Default	0
Unit modes	N/A
Non-volatile	N/A
Activation	For **KP** and **KI** immediate For an advanced filter, only at activating the filter **KV[*N*]**.

Remarks

Velocity advanced filter #1 is configured by **KG[190…441]** and activated by **KV[25]**, and gain schedule mode is activated for it by **GS[16]**.

Velocity advanced filter #2 is configured by **KG[442…693]** and activated by **KV[30]**,and gain schedule mode is activated for it by **GS[17]**.

Position advanced filter is configured by **KG[694…945]** and activated by **KV[50]**, and gain schedule mode is activated for it by **GS[18]**.

Indices

The following table details the use of the **KG[]** parameters array:

Index	KG[*N*] Value	Units	Length
1 to 63	**KI** for inner loop	Hz	63
64 to 126	**KP** for inner loop	Amperes/ (counts/sec)	63
127 to 189	**KP** for outer loop	rad/sec	63
190 to 252	Parameter 1 for scheduled velocity advanced filter #1	By filter type	63
253 to 315	Parameter 2 for scheduled velocity advanced filter #1		63
316 to 378	Parameter 3 for scheduled velocity advanced filter #1		63
379 to 441	Parameter 4 for scheduled velocity advanced filter #1		63
442 to 504	Parameter 1 for scheduled velocity advanced filter #2	By filter type	63
505 to 567	Parameter 2 for scheduled velocity advanced filter #2		63
568 to 630	Parameter 3 for scheduled velocity advanced filter #2		63
631 to 693	Parameter 4 for scheduled velocity advanced filter #2		63
694 to 756	Parameter 1 for scheduled position advanced filter #2	By filter type	63
757 to 819	Parameter 2 for scheduled position advanced filter #2		63
820 to 882	Parameter 3 for scheduled position advanced filter #2		63
883 to 945	Parameter 4 for scheduled position advanced filter #2		63

References

In the SimplIQ Software Manual: Chapter 15, ”The Controller”

GS[], **KV[]**, **KP[]**,**KI[]**

KI[], KP[] – PI Controllers

KI[] and **KP[]** define the parameters of the PI controllers without the second-order filters.

CANopen/CoE

TBD

Attributes

Attribute	Description
Type	Parameter, Real
Source	All
Restrictions	None
Range	**KI[*N*]** > 0 **KP[*N*]** > 0
Index range	**KP[1** to **6]**, **KI[1**, **2, 4]**
Default	0
Unit modes	See below.
Non-volatile	Yes
Activation	Immediate, only when the motor is turned on

Remarks

KI[1] and **KP[1]** define the PI current control filter. The units of **KP[1]** are volt/ampere.

KI[2] and **KP[2]** define the PI velocity control filter. The units of KP[2] are ampere/(counts/sec)

KP[3] defines the gain of the position controller. The units of **KP[3]** are rad/sec.

KI[4] and **KP[4]** define the PI gantry velocity control filter. The units of **KP[4]** are ampere/(counts/sec).

KP[5] defines the gain of the gantry position controller. The units of **KP[5]** are rad/sec.

KP[6] defines the factor of the yaw control to the Y axes in a planar motor. A factor of 1 should transmit the same current command of X yaw to the Y axes.

The parameters **KP[2]**, **KI[2]** and **KP[3]** apply only if gain scheduling is not used: **GS[2]** = 0.

Indices

The following table describes the **KI[]** and **KP[]** entries.

Index	Description	Type	Unit Modes	Units
1	Defines the PI current controller		All	**KP**: volt/ampere **KI**: Hz
2	Defines the PI velocity controller		Position and velocity (when **GS[2]** = 0)	**KP**: ampere/(counts/sec) **KI**: Hz
3	Defines the gain of the position controller		Position (when **GS[2]** = 0)	**KP**: rad/sec
4	Defines the PI gantry velocity controller		Position and current with gantry	**KP**: ampere/(counts/sec) **KI**: Hz
5	Defines the gain of the gantry position controller		Position and current with gantry	**KP**: rad/sec
6	Define the factor of the Yaw control to Y axes in planar motor		Position and current with planar motor	**KP**: N/A

References

KV[], **GS[*N*]**, **KG[]**

KL – Kill User Program

KL stops execution of the user program and turns the servo off.

CANopen/CoE

Attributes

Attribute	Description
Type	Command
Source	All, except the user program
Restrictions	None
Range	None
Default	None
Unit modes	All
Non-volatile	No

Remarks

The **KL** command permanently stops the user program.

The program can run again from the start by using the **XQ** command.

The program status after the **KL** command (refer to **PS** command) is -1.

The **KL** command issued when no program is running does nothing and sets no error code.

KL differs from **HP**, which halts the program and allows it to resume from the same point.

References

HP, **XQ**, **XC**, **PS**

KR – Kill Motion Repetitive

KR command stops the ongoing special motion.

CANopen/CoE

Attributes

Attribute	Description
Type	None
Source	USB, RS232, TCP, EoE
Restrictions	PTP mode only
Range	None
Index range (used in vectored commands)	None
Default	0
Unit modes	5
Non-volatile	No
Attribute	None

Remarks

The **KR** command stops the special motion mode (i.e. repetitive motion), after the current motion is completed.

This command does not stop the ongoing motion.

Where the buffer mode (MR[1]=4) or blended mode (MR[1]=5) are used, the motion stops when the last set point is completed, regardless of the command **KR**.

Special motion mode includes the repetitive modes. Please refer to the **MR[]** command for more details.

Note:
If the special motion mode is not enabled, this command is ignored.

At the next **BG** command, the special mode is re-evaluated.

The **ST** command stops the ongoing motion and the repetitive mode (set MR[1]=0) immediately. The next **BG** does not initiate the repetitive motion i.e. MR[1] should be set again.

The **KL** command disables the servo while stopping the special motion mode.

References

UM, **SR**, **MR[*N*]**, **ST**

KV[] – High-Order Controller Filter Parameters

KV[] specifies the parameters of the second-order 2x2 filters. This filter has DC gain (gain is equal to one in zero frequency).

Indexes equal to 5n are specify the filter type according to Table 7 All advanced filters in drive. When changing filter type drive check the validity of filter parameters. This parameter must be changed at motor disabled.

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Restrictions	The motor must be off.
Range	See the tables below.
Index range	1 to 90
Default	0
Unit modes	All
Non-volatile	Yes

Remarks

Each filter has five parameters. The first four parameters are the physical parameters of the filter, and the fifth parameter is the filter type. The filter is enabled after the fifth parameter **KV**[5**n*] is set.

The following table describes the parameters.

Filter Location	Filters	Parameters	Gain Schedule Parameters
High-order speed controller filters (which filter the PI output)	Filter #1	**KV[1]** to **KV[5]**	
	Filter #2	**KV[6]** to **KV[10]**	
	Filter #3	**KV[11]** to **KV[15]**	
	Filter #4	**KV[16]** to **KV[20]**	
	Gain schedule filter #1	**KV[25]**	**KG[190…441], GS[16]**

Filter Location	Filters	Parameters	Gain Schedule Parameters
	Gain schedule filter #2	**KV[30]**	**KG[442…693]**, **GS[17]**
High-order position controller filters (which filter the proportional output)	Filter #1	**KV[31]** to **KV[35]**	
	Filter #2	**KV[36]** to **KV[40]**	
	Gain schedule filter #1	**KV[45]**	**KG[694…945]**, **GS[18]**
		KV[46] to KV[50]	**Reserved**
Velocity feedback	Filter #1	**KV[51] to KV[55]**	
	Filter #2	**KV[56] to KV[60]**	
Velocity External Reference	Filter #1	**KV[61] to KV[65]**	
Acceleration External Reference	Filter #1	**KV[66] to KV[70]**	
Analog input #1	Filter #1	**KV[71] to KV[75]**	
		KV[76] to KV[80]	**Reserved**
Gantry	Filter #1	**KV[81] to KV[85]**	
	Filter #2	**KV[86] to KV[90]**	
Velocity presentation	Filter #1	**KV[91] to KV[95]**	
Reserved		**KV[21…24]** **KV[26…29]** **KV[41…44]**	

Table 7 All advanced filters in drive

The following table describes the parameter options for each filter and the indices. There are five parameters for each filter. For a specific filter *n* (*n*=1 to 19), there are four filter parameters (5n-4, 5n-3, 5n-2, 5n-1), except for the filters that run in gain schedule, where there P1 to P4 parameters are in **KG[*N*]** parameters.

Filter Type Value P5**n*	Filter Type	P1 = 5**n* – 4	P2 = 5**n* – 3	P3 = 5**n* – 2	P4 =5**n* – 1
0	Filter is canceled				
1	Second-order low pass	Frequency [Hz]	Damping		
2	First-order lead/lag	Frequency [Hz]	Phase [deg]		
3	Second-order lead/lag	Frequency [Hz]	Phase [deg]		
4	Notch filter	Frequency [Hz]	Quality factor	Attenuation [dB]	
5	Anti Notch	Frequency [Hz]	Quality factor	Amplification [dB]	
6	General Bi-Quad	Numerator frequency [Hz]	Numerator damping	Denominator frequency [Hz]	Denominator damping

Notes

- To get the lead filter, in the lead/lag filter, the phase should be positive.
- When setting gain schedule filter (KV[25], KV[30] and KV[45]) it checks and set all the filters in table, but output an error only according to relevant indexes in (GS[16], GS[17] and GS[18]), so better to set first GS and later KV

References

KG[],**GS[]**

LC – Current Limit Flag

LC reports the status of the current limiting process.

CANopen/CoE

TBD

Attributes

Attribute	Description
Type	Integer, Read-only
Source	All
Restrictions	None
Range	0, 1
Default	0
Unit modes	All
Non-volatile	No

Remarks

Two different current limits are in use. The peak limit **PL[1]** specifies how much current can be applied to the motor during short time periods (**PL[2]**) and the continuous limit **CL[1]** specifies how much current can be applied to the motor continuously.

To protect the drive the following condition should be met:
$MC[A]^2 * 3\ [sec] >= PL[1][A]^2 * PL[2]\ [sec]$

LC returns values according to the following table:

Value	Description
0	The motor current is limited by the limit **PL[1]**, or the motor is off.
1	The motor current is limited by the continuous limit **CL[1]**.

References

- **MC**, **PL[]**, **CL[]**

LD – Load Data

LD retrieves the load parameters from non-volatile memory and resets the volatile memory to their default values.

CANopen/CoE

0x1011 with data bytes 0-3 'l', 'o', 'a', 'd'

Attributes

Attribute	Description
Type	Command
Source	All, except User Program
Restrictions	• The motor must be off. • The user program must be at rest. • Wizard mode must not be active.
Range	None
Default	None
Unit modes	All
Non-volatile	No

Remarks

The **LD** command is used to restore non-volatile application parameters from the flash memory to the RAM. After successful loading, the parameters are processed to their internal and real-time values.

The **LD** command resets volatile parameters to their default values.

During the **LD** sequence, relevant parameters are processed automatically by the drive in a procedure which is similar to what the interpreter would do if the parameter arrived from the communication channel. This assures that the loaded parameter is ready for any function, it was designed for.

In cases in which an error occurs during this post processing, the drive performs an automatic **RS** command, which forces all parameters to take their default values. The reason for the failure can then be retrieved by **CD** command.

To avoid a loss of communication, the **PP[]** parameters which define the communication attributes should not be processed.

The **SV** command saves the parameters in the flash memory.

The **LD** procedure may take a long time (tens of milliseconds), and during that time no other command can be processed.

References

SV, **CD**

LP[] – Load Program Info

LP[] gets user program information.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read-only
Source	All
Restrictions	None
Range	None
Index range	1 to 4
Default	0
Unit modes	All
Non-volatile	No

Remarks

If a user program is loaded into the drive, the **LP[*N*]** command provides information about relevant properties of this program.

The **PS** command can assure validation or non-validation of **LP[*N*]** data. See the **PS** command information.

Indices

The following table describes the **LP[*N*]** entries.

Index	Description	Type	Values	Restrictions
0	Reserved			
1	User program code segment **address**. Points to the location of the user program in the non-volatile memory for uploading purposes.	Integer	-	See note below.
2	Total **length** of the code required for program execution (as in **LP[4]**) and the **program text**.	Integer	-	See note below.

3	Program text segment **address** from where the program can be uploaded for debugging purposes.	Integer	-	See note below.
4	Non-text total **length:** code, symbols, functions, variables.	Integer	-	See note below.

Note: The **LP[*N*]** command can be used even when there is no program in the drive. In that case, no error is shown.

References

CC, **XQ**, **PS**

MC – Maximum Current

MC reports the maximum phase current (peak current) allowed for the drive, in amperes. This command informs the software about the rate of the servo drive used with the controller.

The **MC** value is burned in during the production of the drive and cannot be modified by the user.

The drive can run at the current specified by **MC** for 3 seconds.

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read-only
Source	All
Restrictions	No
Range	N/A
Default	According to the servo drive. It cannot be changed.
Unit modes	All
Non-volatile	N/A
Activation	N/A

Remarks

The current can be limited with the **PL[1]** and **CL[1]** commands.

References

IQ, **ID**, **CL[]**, **PL[]**

MF – Drive Fault

MF reports and latches the reason that caused the motor to be disabled (**MO** = 0).

CANopen/CoE

An EMCY message is transmitted when **MF** occurs. The message contains the fault reason. EMCY messages are valid to any DS-402 channel either by EtherCAT or CANopen.

Note that the CANopen "Fault reset" command does not clear the **MF** value and allows the next motor enable command.

Attributes

Attribute	Description
Type	Bit field, Read-only
Source	All
Restrictions	None
Range	None
Default	0
Unit modes	All
Non-volatile	No

Remarks

MF will not report the reason if the motor was shut off from any of the interpreting channels, such as User Program, Serial Communication, CANopen, EtherCAT or TCP/IP.

MF is automatically set to 0 on the next motor enable command from any source: an **MO** = 1 interpreter command, the DS-402 state machine or INH/ENA input.

After the motor is shut down due to a fault, the drive should prevent enabling of the motor for 7.5 miliseconds.

If the fault is caused by an amplifier fault, the red LED will be set, and the **AOK** function will be activated (see the **OL[]** command).

The AUTO_ER routine of the user program should be activated upon an **MF** event.

The following table details the bit-field structure with respect to the fault reason.

MF Value (Hex)	Description	Type CAN EMCY (Hex)	Notes
1 (0x1)	Main feedback error	81 7300	• For analog feedbacks check the threshold level in **CA [##]** • For an absolute encoder check the reason in **EE[1]**. • The fault causes a commutation search on the next motor enable.
2 (0x2)	Commutation process fail during motor on		• For looking the phase • Planar motor when on alignment process
4 (0x4)	Hall main feedback mismatch	81 7380	Illegal Halls
8 (0x8)	Current exceeded peak limit	21 8311	The current has exceeded the value of **MC** but has not yet reached the level of a short. This is typically caused by instability of the current loop.
16 (0x10)	External Inhibit was triggered (INH/ENB)	21 5441	See **IL[]** for more details about the Inhibit/Abort functions. Note the Additional Abort function.
32 (0x20)	Reserved		
64 (0x40)	Halls sensor speed is too high.	81 7381	
128 (0x80)	Speed tracking error	81 8480	• The difference between the commanded speed to the control loop and the feedback exceeded the value defined in **ER[2]**. • This indication is not related to the Max Slippage Error as defined in the DS-402 Profile Velocity mode.

MF Value (Hex)	Description	Type CAN EMCY (Hex)	Notes
256 (0x100)	Position tracking error	81 8611	• The difference between the commanded position to the control loop and the feedback position exceeded the value in **ER[3]**. • This indication is not related to the Following Error as defined in the DS-402 Profile Position mode.
512 (0x200)	Reserved		
1024 (0x400)	Reserved		
2048 (0x800)	Heartbeat event (communication)	11 8130	The motor was shut due to a heartbeat event according to CANopen DS301 object 0x1016.
4096 to 32768 (0x1000 to 0x8000)	Amplifier problem	(See table below)	Indicates the problem that the power section of the drive has encountered.
65536 (0x10000)	Reserved		
131072 (0x20000)	Overspeed indication	81 8481	• The motor speed has exceeded the value which is defined in **HL[2]** or **LL[2]**. • The motor main speed is reported in **VX**.
262144 (0x40000)	Reserved		
524288 (0x80000)	Reserved		
1048576 (0x100000)	Reserved		

MF Value (Hex)	Description	Type CAN EMCY (Hex)	Notes
2097152 (0x200000)	Motor is stuck	21 7121	A stuck motor indication can be requested by using **CL[2]**, **CL[3]** and **CL[4]** according to the following format: If the motor speed is lower than **CL[2]** (in counts/sec) and the measured current is higher than **CL[3]** (in amperes), and if this is observed for more than **CL[4]** msec, the motor is considered to be in the "Motor Stuck" state.
4194304 (0x400000)	Feedback is out of position limits	81 8680	• The main position feedback exceeded the **HL[3]** or **LL[3]** limit. • The main feedback is reported in **PX**.
8388608 (0x800000)	Numeric overflow - ambiguity in results	81 FF30	An internal mathematical problem occurred.
16777216 (0x1000000)	Gantry slave disabled		Gantry master disable because gantry slave is not enabled at current mode
0x2000000-0x8000000	Reserved		
268435456 (0x10000000)	Reserved		
536870912 (0x20000000)	Failed to start motor	81 FF10	• Commutation auto-phasing failed, and the motor could not be started. • A request to initiate the motor using a CANopen control word failed. Possible problems may be: • Inhibit/abort switches are active. • Commutation auto-phasing failed. • The PAL is not initiated/burned. • Too little time has passed since the last fault (typically 7.5 msec) or the last motor disable. • Profiler initiation failed due to conflicts between one of the

MF Value (Hex)	Description	Type CAN EMCY (Hex)	Notes
			profiler parameter/objects (reason in **EE[2]**).
1073741824 (0x40000000)	Reserved		
2147483648 (0x80000000)	Reserved		

The following table details the Amplifier Status bits indication.

MF Indication 0x1000 to 0x8000 Value (Hex)	Description	Type CAN EMCY (Hex)	Notes
0	All OK		
12288 (0x3000)	Undervoltage: The amplifier is not measuring the minimum required voltage.	5 3120	• The minimum allowed value is reported in the **WI[37]** (burnt) and **WI[38]** (actual) command. • Actual bus voltage is reported **AN[6]**.
20480 (0x5000)	Overvoltage: The amplifier is measuring a voltage which is higher than the allowed threshold.	5 3310	• The maximum allowed voltage is reported in the **WI[35]** (burnt) and **WI[36]** (actual) command. • The actual bus voltage is reported in **AN[6]**.
28672 (0x7000)	Safety: One or two of the safety inputs are in safety state.	5 FF20	The safety indications are reported in **SR** bits 14 and 15.
45056 (0xB000)	Short Protection: The current has exceeded a range which is considered as a phase-to-phase or phase-to-ground short.	3 2340	This instantaneous fault is measured by the hardware and typically cannot be recorded or indicated outside of the **MF** command.

53248 (0xD000)	Over-temperature: The drive is sensing a temperature which exceeds the maximum allowed temperature limit.	9 4310	The actual temperature is reported by the **TI[1]** (**TI[2]** in Fahrenheit) command.
61440 (0xF000)	Additional Abort was activated. The drive sensed an input switch that is defined as Additional Abort (refer to IL[] command)	81 5442	The fault is similar to "Abort" function with different value report. This allows user to distinguish between two different faults states such as Inhibit and PTC.

References

SR

MI – Mask Interrupts

MI masks the execution of specified automatic routines in the user program.

CANopen/CoE

Attributes

Attribute	Description	
Type	Bit field, Read/Write	
Source	All	
Restrictions	The AUTO_PERR routine is not maskable.	
Range	Any bit:	
	0	The automatic routine is allowed.
	1	The automatic routine is masked.
	0 to 65535	
Default	0 (All allowed)	
Unit modes	All	
Non-volatile	Yes	

Remarks

A user program may include main code and some automatic routines.

When the program runs, the conditions for calling these routines are checked continuously. If the conditions for running an automatic routine are met, it is called. At certain times, you may want to block some of the automatic routines.

For example:

- An AUTO_RLS automatic routine may be deactivated in a homing process.
- It may be required that a certain code sequence is un-interruptible.

Note: MI masks the execution but does not prevent it. The routine is executed after **MI** allows it.

The bit field characteristic of the **MI** command allows blocking of several automatic routines in a single command.

MI prevents calling of the routine while the specific bit is set. A blocked routine will be called when the specific bit in **MI** is reset to 0.

If AUTO_PERR is activated, all other interrupts are automatically masked (**MI** = 0x7fff).

When an automatic routine is called, the first executable line is performed under a "critical section," allowing the user to set **MI** in the same instance of the routine called.

The **MI** bits are detailed in the following table. The routines are listed in order of descending priority.

MI Value	Masked Interrupt	Relevant Routine
1 (0x1)	Not used	**0**
2 (0x2)	Abort	AUTO_ER
4 (0x4)	Soft stop	AUTO_STOP
8 (0x8)	Soft begin	AUTO_BG
16 (0x10)	RLS	AUTO_RLS
32 (0x20)	FLS	AUTO_FLS
64 (0x40)	Switch enable	AUTO_ENA
128 (0x80)	Digital input 1	AUTO_I1
256 (0x100)	Digital input 2	AUTO_I2
512 (0x200)	Digital input 3	AUTO_I3
1024 (0x400)	Digital input 4	AUTO_I4
2048 (0x800)	Digital input 5	AUTO_I5
4096 (0x1000)	Digital input 6	AUTO_I6
8192 (0x2000)	Main Home event	AUTO_HM
32,768 (0x8000)	User program error	AUTO_PERR

References

XQ, **XC**

MO/SO – Motor On, Servo On

MO enables and disables the motor, and **SO** checks the servo state.

CANopen/CoE

DS-402 state machine using the Control Word (0x6040) and Status Word (0x6041) objects.

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	None
Range	0, 1
Default	0
Unit modes	All
Non-volatile	No

Remarks

If **UM** = 3, an automatic torque can be applied while **MO** = 1 by setting **SC[8]**.

In auxiliary mode (**RM** = 1) **MO** = 1 will be called automatically if one of the inputs is defined as Inhibit/Enable function (**IL[]** = 0 or 1).

In auxiliary mode (**RM** = 1) the motor might move immediately with respect to the auxiliary reference, which might be an analog input.

Disabling the motor by setting **MO** = 0 or by any motor fault (**MF**) will cause a delay for 7.5 milliseconds. Any attempt to set the motor (**MO** = 1) during this time will be delayed until this time has elapsed.

The interpreter motor enable, which uses **MO** = 1, and the CANopen (EtherCAT) motor enable, which uses the DS-402 state machine, can basically live side by side, indicating the correct state regardless of the source of the command. However, mixing the two methods is not recommended.

When the motor is disabled (**MO** = 0) by the interpreter, the CANopen state is "Switch on Disable".

MO = 1 overrides the need for "Fault reset" as required by DS-402 after a fault state.

Enabling the motor

MO = 1 is the operative state of the servo drive, driving the motor and activating and executing the required motion. The software runs a set of tests to ensure that all conditions for running the motor are met.

If **MO** is set to 1 and the motor is already on, nothing happens.

When the motor is enabled, the drive reinitializes the internal parameters and motion drivers.

The drive may fail to start if the setup data is found to be inconsistent (for example, if **XM[2]** < **XM[1]**). In this case, the **CD** command indicates the reason for the failure.

CD may suggest to look for a more detailed error, such as **EE[]**.

During the sequence the last captured motor fault (**MF**) is reset to 0.

The motor is always started so that it does not jump. In case of Position Mode (**UM=5)**, the complete position control command — which consists of the internal position command and the external position command — is set to the actual present position of the motor in order to prevent the motor from jumping.

Note that between two consecutive motor enable and motor disable calls, the motor enabling will be delayed for 7.5 milliseconds.

The SO command on motor on

The Motor On request returns to the interpreter almost immediately. This, however, does not mean that the motor can be controlled by the application/profiler.

If the commutation was not found yet, the motor on procedure will indicate that to the real time, where a commutation search procedure will take place. During this procedure, which might take a long time (a few hundred milliseconds), the profiler or auxiliary reference cannot command the motor to move. During this time the **SO** command will return zero.

The **SO** command indicates whether the servo is enabled, allowing the user (profiler) to command the motion, or is not yet enabled, preventing any reference command to be executed.

After the application initiates motor enable, it must continually check the **SO** command until the value is 1.

Another example where the motion is prevented for a long time while the motor is being set on is when a brake is defined. Brake is defined by **OL[]** which sets the relevant digital input and **BP[]** command that sets the brake time. In this case **SO** indicates 0 until the brake time is exhausted and the drive is ready for profiling.

Disabling the motor

MO = 0 disables the motor. This is the idle state of the drive. The power stage is disabled, and no current flows in the motor. In this mode, the servo drive can perform various tasks that are impossible when the motor is on, including the following:

- Resetting the drive to default (RS command)

- Calculating and checking the integrity of the drive profiler database
- Downloading new firmware or User Programs
- Saving or loading parameters in the flash memory
- Modifying setup data that cannot be modified on-the-fly, such as the commutation parameters (**CA[*N*]**) and unit mode (**UM**)

The servo drive is automatically disabled when a motor fault (**MF**) is captured. An attempt to enable the motor may fail if the conditions of the fault still exist.

The SO command on motor off

SO is set immediately to 0 when the motor is off. **SO** remains set to 1 in cases in which a brake is applied. A brake is defined by **OL[]** which sets the relevant digital input and **BP[]** command that sets the brake time. In this case **MO** indicates 0 while **SO** indicates 1, informing the application that the servo is on.

References

MF, **SR**, **CD**, **BP[]**, **EE[]**

MP[] – Motion Parameters (Reserved)

MP[]

CANopen/CoE

Attributes

Attribute	Description
Type	
Source	
Restrictions	
Range	
Index range	
Default	
Unit modes	
Non-volatile	

Remarks

Indices

The following table describes the **MP[]** entries.

Index	Description	Type	Values	Restrictions
0				
1				
2				

References

MR[*N*] – Motion Repetitive

MR[*N*] are parameters for the special motion mode. Special motion modes are enhancement modes to the point-to-point motion mode (UM=5).

CANopen/CoE

Attributes

Attribute	Description
Type	Integer
Source	USB, RS232, TCP, EoE
Restrictions	According to array index
Range	According to array index
Index range (used in vectored commands)	1-4
Default	0
Unit modes	5
Non-volatile	No
Attribute	None

Remarks

The following options are supported special modes:

- Point-to-point repetitive motion. In this mode the point-to-point motion is repeated until the user stop command (e.g. **ST** command) . Delay can be defined between motions.
- Point-to-point set of set points. In this mode every new motion is added to a buffer, and starts when the previous motion ends.
- Point-to-point set of set points blend mode. In this mode every new motion is added to a buffer, and is blended with the previous motion, i.e. the present motion decelerates or accelerates according to the next speed performing motion with a smooth blend between motions.

The motion begins by setting the MR[] command and initiating **BG** command. The following commands are applied to stop this mode:

- **KR** command stops the motion after the last segment is finished. The command does not stop the mode (**MR[1]** remains active)
- **ST** command stops the motion immediately and resets the mode (**MR[1]**=0)
- **KL** command disables the servo and resets the mode (**MR[1]**=0)

Indices

The following table describes the **MR[*N*]** entries.

Index	Description		Type	Default	Restrictions
1			Integer	0	UM=5
	0	Disable			
	1	Point-to-point repetitive mode, from present position (profiler) to target position. The target position can be either absolute (PA) or relative (PR) according to the last commanded values.			
	2	Point-to-Point repetitive mode, from MR[3] absolute position to MR[4] absolute position.			
2	Delay between motions (in addition to target time) Applicable only if MR[1]==1/2/3		Integer	0	0-16777215 [mSec]
3	First position, depend on MR[1] value Applicable only if MR[1]==2/3		Integer	0	
4	Second position, depend on MR[1] value Applicable only if MR[1]==2/3		Integer	0	

References

UM, **KR**, **SR**

MS – Motion Status

MS reports the status of the motion with respect to the profiler state and the actual feedback.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read-only
Source	All
Restrictions	None
Range	0 to 3
Default	2
Unit modes	All
Non-volatile	No

Remarks

MS refers to the state of the motion according to the specific profiler and the auxiliary reference.

Motion status is set to 3 to indicate that the motor is disabled and no profiler is active.

Motion status is set to 2 when the profiler is initiated and is on the move.

Motion status is set to 1 when the profiler is at rest (the software command has reached the target).

Motion status is set to 0 when all the following conditions hold:

- No motion occurs.
- The profiler is at rest.
- The velocity command is 0.
- The target and feedback are within the target window boundaries.

The motion status values are mode-dependent according to the following description:

Torque modes:

- The CANopen DS-402 Profile Torque mode
- Elmo's **TC** command

MS Value	Description	Note
0	N/A	
1	The torque command (**DV[10]**) has reached the torque target.	
2	The torque command differs from the torque target.	
3	The motor is disabled.	

Velocity modes:

- The CANOpen Profile Position mode
- Elmo's **JV** command

MS Value	Description	Note
0	The actual velocity has reached the target within the velocity target radius range (**TR[3]**, **TR[4]**)	**TR[3]** and TR[4] are reflected in objects 0x606D and 0x606E.
1	The velocity command (**DV[2]**) has reached the velocity target (the profiler is at rest).	
2	The velocity command is in motion.	
3	The motor is disabled.	

Position modes:

- The CANopen Profile Position mode
- Elmo's **PA** and **PR** commands

MS Value	Description	Note
0	The actual position has reached the target within the position target radius (**TR[1]**, **TR[2]**)	**TR[1]** and **TR[2]** are reflected in objects 0x6067 and 0x6068.
1	The position command (**DV[3]**) has reached the position target.	
2	The position command is in motion.	
3	The motor is disabled.	

Time-dependent motion modes

In motion modes which are time-dependent (Interpolated Position, Cyclic Synchronous Position) **MS** reports 2.

References

NF[] – Non-Linear Float

NF[] specifies the non-linear float table used for various implementations.

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Restrictions	None
Range	See the table below.
Index Range	1 to 2
Default	Zero or see the table below.
Unit modes	See the table below.
Non-volatile	Yes

Remarks

NF[1] and **NF[2]** are available for velocity and for position control. The value is the current, used at beginning of motion to overcome friction.

Indices

The following table details the **FF[]** entries:

Index	Description	Default	Values	Restrictions
0	Reserved			
1	Specifies the current that will be used at the beginning of positive motion. In amperes.	0	0 to **PL[1]**	
2	Specifies the current that will be used at the beginning of negative motion. In Amperes.	0	-**PL[1]** to 0	1

3	Type of cogging compensation Bit 0 – cogging by sine wave Bit 1 – cogging by **NT[N]** table Bit 2 – flag to use the cogging compensation current (1 use; 0 disable)	0	0 to 6	
4	Cogging compensation sine amplitude in Ampere	0	0 to **CL[1]**	
5	Cogging compensation sine harmonics in electrical cycle	0	0 to 12	
6	Cogging compensation sine offset in internal electrical angle	0	-511 to 512	

References

PL[]

NT – Non-Linear Table

NT specifies the non-linear table used for various implementations.

At the present time it is used for cogging compensation for each phase in one electrical cycle.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	
Range	N/A
Index range	1 to 256
Default	0
Unit modes	All but stepper mode
Non-volatile	Yes

Remarks

To change from current units (amperes) to the internal units of the **NT[*N*]** array, you should multiply the number of amperes in **WS[22]** and set **XA[5]** to 1.

Indices

The following table describes the **NT[]** entries.

Index	Description	Type	Values	Restrictions
1 to 256	Current added to control for cogging compensation	float	Internal units	Can be changed only at motor off

References

CL[*N*], **PL[*N*]**, **WS[*N*]**, **NF[N]**

OB[*N*] – Output Bits

OB[*N*] sets and resets a general-purpose output bit.

When **OB[*N*]** is queried, it reflects the **OP** command bits 0 to 31, respectively.

CANopen/CoE

Attributes

Attribute	Description
Type	Parameter, Array, Read/Write
Source	All
Restrictions	Reflect the **OP** command in a bit oriented manner. **OB**[1] reflects bit 0 in **OP** and **IB**[32] reflects bit 31 in **OP**.
Range	0, 1
Index range	1 to 32.
Default	0
Unit modes	All
Non-volatile	Yes

Remarks

The **OB[B]** command allows the setting or getting of specific bits of the **OP** parameter.

For example, if digital output 2 is defined as general-purpose (refer to the **OL[]** command), the command **OB[2]** = 1 will set the digital output to active position (depending on the logic level of this output).

OB[1] to **OB[32]** represent the **OP** command register bits 0 to 31.

For example:

OB[1], like **OP** bit 0, represents the General-Purpose Output 1 level value.

OB[4], like **OP** bit 3, represents the General-Purpose Output 4 level value.

Gold drives support a variety of digital outputs. The number and details of these digital outputs is specified in the drive’s Installation Guide. The value of **OB[*N*]** varies according to the logic state of the output even if the output does not exists.

The **OB[*N*]** syntax may be more convenient than OP for setting individual outputs. However, it is not appropriate for the synchronized setting of several output bits.

Setting of **OB[*N*]** will not affect the digital outputs which is not defined as General Purpose. Setting a none General Purpose output does not affect the output, does not burst and alarm.

OB[*N*] reflects the logical values of the outputs. It does not however inform the physical level. Physical level depends on the way the output is connected externally.

When the drive reboots (power-up or during firmware download), output ports are set internally to 'not conduct'. This way no transition will occur during boot up.

Outputs 14, 15 & 16 are connected to hardware PORT C differential outputs, where output 15 is the 'A'&'~A', output 16 is 'B'&'~B' and output 17 is 'Index'&'~Index'.Indices.

The following table describes the **OB[*N*]** entries.

Index	Description	Type	Values	Restrictions
1 to 4	Returns the value of a digital output (1 to 4), if the digital output is defined as a general-purpose output. Otherwise, it returns 0.	Integer	0, 1	
5 to 13	Reserved, return 0	0		
14 to 16	Returns the value of digital output (14 to 16), if digital output is defined as general purpose output.			
17	If at least one of the digital outputs is mapped to the **Amplifier OK** function: **OB[7]** returns 1 if the drive is ready to be enabled (no amplifier exception such as under voltage which prevents **MO**=1). **OB[7]** returns 0 if the drive is not ready. If none of the digital outputs is mapped to the **Amplifier OK** function, **OB[7]** returns 0.	Integer	0, 1	
18	If at least one of the digital outputs is mapped to the **Brake** function: **OB[8]** returns 1 if the brake is engaged, or 0 if the brake is released. If none of the digital outputs is mapped to the **Brake** function, **OB[8]** returns 0.	Integer	0, 1	

19	If at least one of the digital outputs is mapped to the **Motor enable/disable** function: **OB[9]** returns 1, if **SO** = 1, or 0 if **SO** = 0. If none of the digital outputs is mapped to the **Motor enable/disable** function, **OB[9]** returns 0.	Integer	0, 1	
20 to 32	Reserved, return 0	0		

References

OP, **OL[]**

OC[] – Output Compare

OC[*N*] allows the generation of pulses when given sensor positions.

Note:

Mode is supported in GCON based drives only (not supported in G-GUI).

CANopen/CoE

Attributes

Attribute	Description
Type	Integer
Source	USB, RS232, TCP, EoE
Restrictions	According to array index
Range	According to array index
Index range	1 to 12, 21 to 32
Default	See table below
Unit modes	All
Non-volatile	No

Remarks

OC[1 to 12] – Output pulses when comparable terms are encountered at quad module-0 (Port B).

OC[21 to 32] – Output pulses when comparable terms are encountered at quad module-1 (Port A).

Note:
The following text refers to OC[1 to 12], but also applies to the OC[21 to 32] command unless stated differently.

The **OC** command generates a train of pulses according to the sensor position values.

The **OC[1]** enables and disables the mode and operates in three sub-mode options:

- **OC[1]=1**: Absolute position mode. The first pulse is generated by the initialized absolute position defined by **OC[2]** (i.e. **PX**=**OC[2]**) and continues at position intervals specified by the **OC[3]** value (i.e. **PX**=**OC[2]**+k***OC[3]**, where k is the number of successful compared occurrences (k=1,2,…).
- **OC[1]=3**: Table position based mode. In this mode the positions are taken from a table. The table includes pairs of absolute positions. Each pair includes a “start position” – the

absolute position to start the pulse, and an "end position" – the absolute position to end the pulse.

- **OC[1]=4**: Table time based mode. In this mode the positions are taken from a table. The table includes absolute positions. These positions indicate the absolute positions to start a pulse, and the duration of the pulse is indicated in **OC[4]**.

Notes:
"Generate pulse" or "Active high" means that the hardware will set the output to '1' – causing current to flow through the opto-coupler.

Be aware that long time duration pulses can cause a number of pulses to overlap. The drive gives no warning on such occasions.

When going in the direction, opposite to the specified direction with the same **OC[3]** value, the compare will occur every 0xFFFFFFFF-**OC[3]** counts.

In order to prevent the output of an additional pulse during activation or deactivation of the feature, it is recommended to configure the high logic level for the output (Set **OL[i]=1**) before the **OC[]** is activated.

The various homing and capture modes (i.e. DS-402 Homing mode, DS-402 Touch-Probe, **HM[1]** or **HF[1]**) will not operate when the output comparison operates on the same sensor/feedback source. Exception to this rule is the configuring of **HM**/**HF** source other than Index (i.e. **HM[3]/HF[3]**!=1 and **HM/HF[3]**!=2), in all the other cases output comparison and **HM/HF** can be used on the same sensor at the same time.

In the table single direction mode, the pulses are generated according to the order in the table. If the direction is changed during the comparison mode in the middle of the table, the next generated pulse will be the next position in the table before the change of direction.

In all modes, the first position must be at least **TS** time from the activation position (the drive position when sending the **OC[1]** command). If the table is to be converted, the first pulse must be at least 70[msec] from the activation position (please see Administrative Guidefor more information).

In table mode, if both directions are selected, the comparison operates infinitely and will only end with the **OC[1]=0** command.

In table modes compare, the table's data is converted during the activation of the comparison mode. An additional activation of comparison in the table mode must be preceded by refilling the table with the user unit's positions, or the setting of **OC[11]\OC[31]** to 1.

In table modes in both directions, the pulse generating sector is in the table boundaries (the table's index is not rolled). If the movement passes the last position in the table, the next pulse position to generate is the last table position. If the movement passes the first position in the table, the next position generated is the first position in the table.

Changing sensor position during an output comparison, will **not** re-evaluate the comparison points in the compare table automatically. After sensor position change the output compare mast be reinitialized to prevent offset, and if table mode is selected the compare table positions must be converted.

Output 14-16 are fast outputs allowing accurate pulse length and response time. Outputs 1-4 are slow outputs (relatively) and might distort the pulse timing, adding additional time to the pulse. In many cases this does not matter but user should be aware of the HW abilities.

In case that the compare feedback is not AQB encoder, the none AQB feedback must be emulated to one of the AQB modules and then the compare function can be activated on this last module.

Indices

The following table describes the **OC[]** entries.

OC Module-0 Index	OC Module-1 Index	Description	Default Value
1	21	0: disable output compare.	0
		1: Accept last changes in **OC[*N*]** and enable output compare beginning at absolute position **OC[2]**.	
		2: Reserved (returns out of range)	
		3: Accept last changes in **OC[*N*]** and enable table position based output compare mode.	
		4: Accept last changes in **OC[*N*]** and enable table time based output compare mode.	
2	22	The absolute position of the first pulse (depend on the feedback selected). Applicable only in absolute position compare mode. This value cannot exceed the modulo limit in the same direction of motion i.e. |**OC[2]**|-|**PX**| must be positive.	0
3	23	The hardware position intervals between subsequent pulses (in FP[x] units). The positive/negative value of **OC[3]** should be set according to the encoder motion. When the direction is positive (increasing PX/Sensor-position value) **OC[3]** should be positive; otherwise it should be negative.	1000

4	24	N: Pulse duration calculated in the following algorithm:
		If(1<=N<=127) – pulse duration value is N[µSec].
		If(127<N<=253) – pulse duration value is:
		((N-127+1) *100) [µSec].
		Minimum pulse length is 1[µSec].
		Maximum pulse length is 12,700[µSec].
			4

5	25	N: number of pulses to generate. 0: infinite output compare mode (train of pulses will end only with the **OC[1]**=**0** command). Not applicable in table compare modes **OC[1]**=3|4. Any 32 bits value is applicable	0
6	26	Output compare source signal: 0: Output compare on Position-Feedback. 1-4: Output compare on socket number (1-4). In both cases the socket must be configured to Port-A or Port-B (AQB sensor).	0
7	27	Array selection to be used as position table: 1: ZX array is used for compare, range 1-1022 (The array is used during the EAS Wizard and should be refilled after the Wizard is used) 2: NT array is used for compare, range 1-254 3: ET array is used for compare, range 1-2048 4: UI array is used for compare, range 1-24 5: BH array is used for compare, range 1-16382 (When using this array data recording is not available. Sending start recording during this option will fail). Applicable in table modes **OC[1]**=3|4. **Notes:** Sizes of the tables are smaller than the array's actual sizes. **GV[N]/GW[N]** command will limit the user. To fill the array with compare positions data, use **GV[N] /GW[N]**command, and do not fill the array directly with array command. NT, ET & UI are none volatile arrays. SV command shall store these arrays. When reactivating the **OC[]** need to note the **OC[11]/OC[31]** state before.	0
8	28	Tables first position index. Applicable in table modes **OC[1]**=3|4. Validated only at mode enable.	0

9	29	Tables last position index. Applicable in table modes **OC[1]**=3|4. Validated only at mode enable.	0
10	30	Axis direction: Both directions. In this case the compare mode is infinite (the train of pulses will end only with the **OC[1]=0** command). Positive direction only. Negative direction only. Applicable in table modes **OC[1]**=3|4.	0
11	31	Convert table positions 0 – convert table position from user units to Hardware (sensor) units (including error mapping if enabled (**PC[]**)), at **OC[1]** enable command. 1 – Do not convert the table positions from Sensor units to Hardware units. Applicable for table modes only, i.e. **OC[1]**=3|4. **Note:** The converted values are stored in the same table of the position which the user sets. If 0 is selected, each activation table values shall be converted. It is recommended to set 1 after the first conversion (first OC[1]=3|4). This will reduce enable mode time.	0
12	32	Number of pulses generated since mode enabled. **Notes:** Applicable in all modes. When mode is enabled (**OC[1]**!=**0** command) the value is cleared (i.e. = 0). The user can change the value to any number even when compare is enabled.	0

The following table describes the **OC[1]/OC[21]** report values:

OC[1]/OC[21] Value	Description
-1	No more pulses are being generated because the number of pulses/table entries specified in **OC[5]** has been reached.
0	Output compare module is disabled.
1	Sub mode depended: **Absolute position** sub mode: output compare function has started but absolute position has not yet been reached; therefore, the train of pulses has not begun. **Table position** based mode: output compare function has started but the first table position has not yet been reached. **Table time** based mode: output compare function has started but first table position has not yet been reached.
2	The train of pulses is being generated now.

References

GO[], **GV[]**, **GW[]**, **EA[N]**

OF[] – CAN Objects to Flash memory (Reserved)

OF[]

CANopen/CoE

Attributes

Attribute	Description
Type	
Source	
Restrictions	
Range	
Index range	
Default	
Unit modes	
Non-volatile	

Remarks

Indices

The following table describes the **OF[]** entries.

Index	Description	Type	Values	Restrictions
0				
1				
2				

References

OL[] – Output Logic

OL[] specifies the digital output function and logic.

CANopen/CoE

Attributes

Attribute	Description
Type	Parameter, Bit field, Read/Write
Source	All
Restrictions	None
Range	0 to 9
Index	1 to 16
Default	0
Unit modes	All
Non-volatile	Yes

Remarks

OL[*N*] is a bit-field command which allows the user to map any of the uncommitted digital outputs of the drive to any function and desired logic level.

The function description is given in Digital Output Function Description.

The *logic level* determines the relation between the output activation and the current flow from the drive. The actual HW functionality of the output is hardware-dependent.

Positive logic (active high) means that if the function is activated, the drive sets 1 for the relevant digital output pin and the opto-coupler is conducting.

Negative logic (active low) means that if the function is activated, the drive sets 0 for the relevant digital output pin and the opto-coupler is not conducting.

Bit-Field Entries

The following table describes the bit-field entries of **OL[]** for logic and function.

Bit	Description	Type	Values		Restrictions
0	Logic level	Boolean	0	Active low	
			1	Active high	
1 to 4	Function behavior * See the detailed description in **Digital Output Function Description** below.	Integer	0	General purpose	
			1	**AOK** function	
			2	**Brake**	
			3	Servo State (**MO**)	
			4	Motor Fault (**MF**)	
			15	Ignore	
5 to 15	Reserved				

Possible Values of OL[*N*]

The following table lists the possible values of **OL[*N*]**.

Command Value	Logic Level	When Active . . .
OL[*N*] = 0	Low	Output is general-purpose.
OL[*N*] = 1	High	Output is general-purpose.
OL[*N*] = 2	Low	**AOK** indicates that the drive is ready for use.
OL[*N*] = 3	High	**AOK** indicates that the drive is ready for use.
OL[*N*] = 4	Low	Brake feature is active.
OL[*N*] = 5	High	Reserved
OL[*N*] = 6	Low	**Motor enable/disable** indication
OL[*N*] = 7	High	**Motor enable/disable** indication
OL[*N*] = 8	Low	Motor was disabled due to a fault.
OL[*N*] = 9	High	Motor was disabled due to a fault.

Digital Output Function Description

Function 0: General purpose

The output is general-purpose (has no special automatic function) and can be set or reset by the **OP** or **OB[*N*]** command from any source.

Function 1: AOK

The **Amplifier OK** function indicates that the physical condition of the drive allows the motor to be enabled. If a digital output is assigned to **AOK** it will be automatically reset to 0 if an amplifier fault occurs. An amplifier fault includes any of the following:

- Short protection
- Overvoltage
- Overtemperature
- Overvoltage
- Safety state

For more details about amplifier faults please refer to the **MF** command.

Function 2: Brake

The **Brake** function is an automatic function which logically sets the output according to the brake parameter time definition when the motor is either enabled (to disengage the brake) or disabled (to engage the brake). Refer to the **BP[*N*]** command for more details.

Function 3: Motor Enable

The output will be logically set in cases in which the servo is enabled. Refer to the **MO** command for more details.

Function 4: Motor Fault

The output will be logically set in cases in which the motor aborted to freewheel. The cause of the fault is in the **MF** command. The output is reset to 0 when the motor is re-enabled or when a "Fault Reset" request is sent from the "Fault State" in the CANopen state machine.

In cases in which motor enable is requested by the **MO** = 1 command or by the CANopen state machine, the fault is reset even if the motor on procedure returned an error. For example, if during the motor on procedure it was detected that the safety switch is not conducting, the pervious fault indication is cleared and the output is set to 0.

Notes

- The Output Compare function is HW-dependent and requires handling with the **OC[*N*]** and **GO[*N*]** commands. Please refer to these relevant commands.
- For outputs that are not supported by the product, the relevant index (the **OL** index) will be accepted, but ignored.
- In Emulation function, the logic level of the relevant output (**OL[14]** to **OL[16]**) determines the emulation logic level. Refer to the **EA[*N*]** command for more details.
- If **GO[*N*]** is defined for Emulation, the relevant **OL[*N*]** function should not be activated.

References

OP, **OB[*N*]**, **MO**, **OC[*N*]**, **GO[*N*]**, **EA[*N*]**

OP – Output Port

OP specifies the digital output port.

CANopen/CoE

Attributes

Attribute	Description	
Type	Integer, Read/Write	
Source	All	
Range	Bit 0	General-purpose output 1 level
	Bit 1	General-purpose output 2 level
	Bit 2	General-purpose output 3 level
	Bit 3	General-purpose output 4 level
	Bit 4	Reserved
	Bit 5	Reserved
	Bit 6	**Amplifier OK** output indication
	Bit 7	**Break** output indication
	Bit 8	**MO ON** output indication
	Bit 9-15	Reserved
Default	0	
Unit modes	All	
Non-volatile	Yes	

Remarks

Sets values for all general -purpose digital outputs as defined in the **OL[]** command.

Querying **OP** indicates which digital output is logically activated.

For example, if digital output is defined as general-purpose and the user sets **OP** = 8, digital output 4 becomes active, and, depending on its logic level configuration, the specific output is set or reset.

OP does not affect the digital output pins otherwise defined as general-purpose.

The **OB[*N*]** command can be used to access (set and read) individual digital outputs rather than the whole port.

When any of the uncommitted digital outputs is defined as general-purpose, the physical state of the output depends on the previous **OP** command setting.

The **OB[*N*]** syntax may be more convenient than **OP** for setting individual outputs. However, it is not appropriate for the synchronized setting of several output bits. If a synchronized setting of several digital outputs is desired, use the **OP** command.

The output compare function depends on the drive. In the Gold Whistle any output can be used, while in case of the Gold Trombone and the Gold Guitar only output 1 can be used.

If the **OC[*N*]** function is active, the defined output compare output is overridden by this function.

References

OB[*N*], **OL[*N*]**

OV[] – Set CANopen Objects

OV[] specifies alias values in the CANopen object list.

It is mainly used during test procedures.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	None
Range	See the table below.
Index range	1 to 60
Default	N/A
Unit modes	All
Non-volatile	No

Remarks

The **OV[]** is subject to the CANopen object dictionary list. Thus, the range, type and attribute of the objects are not listed here. Refer to the CANopen manual for more details about the attributes of the objects.

All the objects are volatile objects, which are reset to their default values when the drive is booted up by NMT or power-up.

It is highly recommended not to use the **OV[]** command along with setting of the CANopen object. This will lead to ambiguous behavior.

Indices

The following table describes the **OV[]** entries.

Reserved and unused entries will return 0 and do nothing.

Index	Object (hex)	Description	Notes
1	603F	Read Error Code DS402	
2	6061	Read Mode of operation display	
3	6064	Read actual position in User Units	
4	6069	Read the actual speed in counts/sec	Similar to **VX**. Value can be read from Speed sensor or Position sensor. See 606A.
5	606A	Velocity Sensor selection code. Selects between Position or Velocity sensor	
6	606B	Speed demand value in counts/sec	The output of the profiler converted to counts/sec.
7	606C	Read the actual velocity in User Units	Value can be read from Speed sensor or Position sensor. See 606A.
8	6081	Read, Write Profile Velocity used in Profile position mode	Object is override by SP value when Elmo's Begin Motion (**BG**) is set.
9	60B0	Read, Write Position Offset for Cyclic Synchronous Position (CSP) mode	
10	60B1	Read, Write Velocity Offset for CSP and Cyclic Synchronous Velocity (CSV) mode	
11	60B2	Read, Write Torque Offset for CSP, CSV and Cyclic Synchronous Torque (CST) mode.	
12	60B8	Read, Write Touch Probe configuration function	
13	60B9	Read Touch Probe Status	
14	60BA	Read Touch Probe 1 Positive edge capture	In User Units

Index	Object (hex)	Description	Notes
15	60BB	Read Touch Probe 1 Negative edge capture	In User Units
16	60BC	Read Touch Probe 2 Positive edge capture	In User Units
17	60BD	Read Touch Probe 2 Negative edge capture	
18	2E10	Touch Probe Home	
19	60C1.1	Read, Write Interpolated Data Record Sub index 1, used in Interpolated Position mode (IP)	Typically the position set point for IP mode
20	60C1.2	Read, Write Interpolated Data Record Sub index 2, used in Interpolated Position mode (IP)	Typically the velocity set point for IP mode
21	Reserved		
22	Reserved		
23	60C2.1	Read, Write Interpolated Time period for IP mode and cyclic synchronous modes	
24	60C2.2	Read, Write Interpolation Time index for IP mode and cyclic synchronous modes	
25	60C4.4	Read, Write Buffer Position in Interpolation data configuration object. Points where the next set point shall be written to in a ring buffer organization. Used in IP mode	
26	60C4.5	Read, Write Size of data record in bytes. Used IP mode.	
27	60C4.6	Write Clears and allows the set point buffer of the IP mode.	
28	60F2	Read, Write Positioning option code. Used in position control modes to determine the behavior of the relative position, modulo and set point ACK in Profile Position mode.	

Index	Object (hex)	Description	Notes
29	60FA	Read Control Effort	Returns 0: reserved for this object.
30	60FC	Position Demand Value internal units	Value is the actual entry to the control loop including real time Auxiliary Position w/o modulo calculations.
31	60FD/20FD	Digital input/Clear digital inputs	
32	607A	Target Position. Used in Profile Position mode and CSP mode.	
33	Reserved		
34	6082	End Velocity used in Profile Position mode	Object is override by FS when Elmo's Begin Motion (BG) is set
35	6098	Homing Method	
36	6099.1	Homing Speed during search of switch	Typically fast speed when looking for edge or limit
37	6099.2	Homing Speed during search of zero	Typically slow speed when searching for the final homing location
38	609A	Homing Acceleration	Used for Deceleration when required
39	607C	Homing offset	
40	Reserved		
41	Reserved		
42	60F4	Position Following Error in User Unit	
43	60FF	Read, Write Target Velocity. Used in Profile Velocity mode.	
44	6071	Reserved for Target torque	
45	Reserved		
46	6077	Read Torque Actual Value	Torque units are in 1/1000 of rated torque (6076)
47	6078	Read Current Actual Value	Current units are in 1/1000 of rated current (6075)

Index	Object (hex)	Description	Notes
48	2E00	Reserved for Position Compensation Index	
49	2E0A	Reserved for Torque Compensation value	
50	6062	Read Position Demand value by User Unit	
51	6079	Read DC link voltage. Bus voltage in millivolts	
52	Reserved		
53	20A0	Read Additional Position	
54	20B0.8	Homing Sensor selection	
55	2020.1	Homing Torque Limit	
56	2020.2	Homing Position Limit	
57	2020.3	Homing Time Limit	
58	60FE.1	Digital output mask	
59	60FE.2	Digital output value	
60	2082	CAN Controller status	

References

OF[], **BG**

PA – Position Absolute

PA specifies the absolute position in counts.

CANopen/CoE

0x607A

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	• The motor must be on. • Effective on the next call to **BG** • **PA** must be within the ranges defined by **VH[3]** and **VL[3]**.
Range	-2^{31} to $(2^{31} - 1)$
Default	0
Unit modes	**UM** = 3; **UM** = 5
Non-volatile	No

Remarks

On the next call to **BG** after applying the **PA** command:

- The motion control will be switched to the position control loop.
- The drive motion mode will be changed to the Point-2-Point (PTP) motion mode according to **AC**, **DC** , **SF**, **SP** and **FS**.

When **PA** is set and **BG** is commanded, the motion mode reflected in **OV[2]** (object 0x6061) is modified to 1 (Profile Position).

Object 0x607A will be overridden by the **PA** value. Refer to the **BG** command for more details.

PTP is limited by software limits. Motion will stop at these limits.

If **UM** = 3, **PA** determines the target position in electrical angle units (1 pole pair is 360 electrical degrees, which are denoted by 512 electrical ticks).

The motor will abort if the feedback position is higher than **HL[3]** or lower than **LL[3]**.

The **PA** value can be given in user-defined position units specified by the **FC** command.

References

JV, **FS**, **SP**, **AC**, **DC**, **UM**, **PR**, **PO**, **FC**

PB – PAL Burn

PB reports the burned PAL number.

Changing the PAL is required in case of special features, such as special encoders implementation.

Note: Some of the drives do not support PAL burn. Refer to the **PB[]** description table in this command.

CANopen/CoE

Attributes

Attribute	Description
Type	Signed integer
Source	USB, TCP, EoE, RS232
Restrictions	N/A
Range	N/A
Default	1
Unit modes	All
Non-volatile	Yes

Remarks

The **PB** command reports current burned PAL with the user-selected PAL in the drive.

PAL burning sequence:

- Start downloading PAL via EAS software or via EtherCAT-FoE protocol.
- After PAL is burned the PB reports -1, until the drive is rebooted.
- When the process is completed successfully, the **PB** command returns the number of the PAL burned in the drive. If the process fails, the **PB** command returns -1, and the drive will not allow the user to perform the following:
  - The user program auto-exe will not be performed at power-up.
  - The user program cannot be executed.
  - The servo cannot be turned on.

Using a PAL burn with a version that is not supported by the firmware will prevent the motor from being enabled.

Power-up must be performed after the burn is completed.

The following is the description table:

Value	Description	PAL Description		
		Gold Guitar (WS[8]==0)	**Gold Whistle/ Gold Trombone Rev-A (WS[8]==1)**	**Gold Whistle/ Gold Trombone Rev-C (WS[8]==2)**
-1	PAL is not burned or after new PAL was downloaded (reboot needed). The servo cannot be set (**MO** = 1), and the user program is disabled.	N/A	N/A	N/A
1	PAL #1 is burned	Port A: Biss Port B: None absolute	PAL Version 20 (Default)	PAL Ver 42.1 (Default) Base Biss EnDat SSI Panaasonic Tamagawa
2	PAL #2 is burned	**Port A**: Panasonic, Mitutoyo , Tamagawa **Port B**: None Absolute OR **Port A**: None Absolute **Port B**: None Absolute	N/A	PAL Ver 42.2 Base Sanyo
3	PAL#3 is burned	**Port A**: EnDat **Port B**: None absolute OR **Port A**: None Absolute **Port B**: None Absolute	N/A	N/A
0x5AA5	The PAL cannot be replaced.	The actual PAL version was burned during manufacturing.		

References

VP, WS

PC[*N*] – Error Mapping

PC[*N*] enables the configuration and operation of error mapping.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer
Source	USB, RS232, TCP, EoE
Restrictions	According to array index
Range	According to array index
Index range	1 to 8
Default	**PC[3]** = 1
Unit modes	All
Non-volatile	Yes
Attribute	None

Remarks

Error mapping is used to correct non-linear mechanical position errors.

This command enables the user to set error mapping parameters and to enable/disable error mapping mode.

The mode can be activated using a linear table, where the socket position is taken as an absolute entry in the table. The mode can also be activated as a cyclic mode (modulo), where the socket position is calculated for each modulo value, as defined by the user, producing endless cyclic position correction.

The table entries are the corrections which need to be made in a specific feedback location. These values can be treated as values in user units, in which case a conversion procedure is executed, or as values in sensor units, in which case no conversion is calculated.

For more details on different error mapping options, please refer to the "Error mapping user manual" document.

Error mapping cannot be enabled during the following scenarios:

- While DS-402 homing is in progress (object 0x6060 sub mode 6)
- **HM**/**HF** are operational with homing mode, i.e., **HM[1]** > 0 and **HM[5]** != 2 or **HF[1]** > 0 and **HF[5]** != 2.

The following operations cannot be performed when error mapping is enabled:

- DS-402 homing
- Setting the position of the main position sensor (**PX** = xx)
- Activating the **HM**/**HF** with homing mode, i.e., **HM[2]** != 2 or **HF[2]** != 2.

While error mapping is enabled, all captured positions in **HM**/**HF** or Touch-probe are after the correction was considered. (i.e., positions after correction).

To edit the correction table use the **GP[]** command.

The overall corrected position (abscissa + error) must be rising monotonously. This limitation directly implies that the correction table uniquely defines all positions, i.e., for each corrected position there is one and only one actual sensor reading that satisfies the following relation:

Corrected Position = Actual Position + ErrorCorrection

This limitation is not checked by the drive before enabling error mapping. It is up to the user to verify this.

Indices

The following table details the **PC[]** entries.

Index	Description		Type	Default	Restrictions
1	**Value**	**Operation**	Integer	0	See notes below.
	0	Disable			
	1	Enable linear mode **without** converting correction table values from user units (UU) to sensor units. In this mode it is assumed that the position correction values are in sensor units.			
	2	Enable cyclic (modulo) mode **without** converting correction table values from user units (UU) to sensor units. In this mode it is assumed that the position correction values are in sensor units.			
	3	Enable linear mode **with** conversion of the correction table values from user units (UU) to sensor units.			
	4	Enable cyclic (modulo) mode **with** conversion of the correction table			

		values from user units (UU) to sensor units.			
2	Socket number to which error mapping will be applied		Integer	1	1 to 4
3	**Value**	**Direction**	Integer	0	1 to 3
	1	The NT array is selected as the correction table. The range is from 1 to 254.			
	2	The ET array is used as the correction table. The range is from 1 to 2048.			
	3	The UI array is used as the correction table. The range is from 1 to 24.			
4	Low index of the correction table		Integer	1	≤1
5	High index of the correction table		Integer	2	≤ Table size
6	Correction table position grid of size 2^N, where $3 \le N \le 19$		Integer	3	$3 \le N \le 31$
7	Error mapping start position (in sensor units)		Integer	0	
8	N-Modulo value, where the modulo is between 0 and N		Integer	100	Positive value ≤1

References

GP[]

PE – Position Error

PE reports the position error.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read-only
Source	All
Restrictions	None
Range	0 to $2^{31} - 1$
Default	0
Unit modes	**UM** = 5
Non-volatile	No

Remarks

The **PE** command returns the instantaneous position tracking error, in counts.

In main feedback position mode (**UM** = 5), **PE** reports the following:

PE = DV[3] – PX

If the absolute value of **PE** exceeds **ER[3]**, the motion is aborted, and the motion fault code **MF** = 256 (0x100) is set. If **MO** = 0, or if the position controller is not used for example in velocity or current or stepper mode (**UM** = 1, 2 or 3), **PE** returns 0.

References

XM[*N*], **ER[*N*]**, **MF**, **UM**

PL[*N*] – Peak Limit

PL[] specifies the peak limit current peak limit duration.

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Restrictions	None
Range	**PL[1]**: **0** to **MC** **PL[2]**: 0.1 to 30
Index range	1, 2
Default	**PL[1]** = 0, **PL[2]** = 3 (**RS**)
Unit modes	All
Non-volatile	Yes

Remarks

This parameter is used to protect the motor (or the drive) from overcurrent and to protect the load from excessive torque. The motor current (torque) command is normally limited to its peak limit, as defined by **PL[1]**. After a short period of torque demand higher than **CL[1]**, the torque command limit is decreased to **CL[1]**. If the current command has been raised to **PL[1]** from 0 after the time specified for the peak duration (**PL[2]**, in seconds), the motor current command will be limited to **CL[1]**. The motor current command remains limited to **CL[1]** until enough time has passed for the average requested torque command to fall below 90% of **CL[1]**.

The **LC** flag indicates that the current is limited to its continuous limit.

The torque limits **PL[1]** and **CL[1]** may be changed dynamically while the motor is on.

Indices

The following table describes the **PL[*N*]** entries.

Index	Description	Units	Range
0	Reserved		
1	Defines the motor maximum peak current, in amperes.	Amperes	0 to **MC**
2	Defines the motor maximum peak duration, in seconds.	Seconds	0.1 to 30

Notes

- It is recommended to define a **PL[1]** value that can be achieved. It is not recommended to set **PL[1]** > **BV**/R, where **BV** is the DC power supply voltage and R is the motor resistance. The **PL[1]** value should be small enough so that at peak current there is enough voltage to drive current changes. Otherwise, at large currents the drive's response rate will be limited by voltage saturation, and the controller's performance will decrease.
- The allowed peak current may be saturated at a level lower than the **PL[1]** value when the PWM frequency is increased with the **XP[2]** command.
- The peak duration **PL[2]** specifies the time that is required to switch from the peak limit to the continuous limit, when the current **PL[1]** and **PL[1]** = **MC**. The actual time period during which the peak current may be applied can, however, differ significantly from **PL[2]**.
    - If **PL[1]** < **MC**, a longer time may be allowed for the peak current in order to protect the drive itself. The command values may be used to calculate this period, but with protection of the drive.
      $\mathbf{(MC)^2 \times 3\ sec\ (always) \geq (PL[1])^2 \times PL[2]}$
      The user should make sure that the calculated value of $\mathbf{(PL[1])^2 \times PL[2]}$ does not exceed the overload value of the motor.
    - If, prior to the high current demand, the current demand was very close to **CL[1]**, the switch will occur almost instantaneously.
    - If the current demand is marginally greater than **CL[1]** and significantly less than **PL[1]**, the switch may take a very long time. The exact time required can be calculated from the previous formulas.
- If **CL[1]** > **PL[1]**, **PL[1]** will be the torque limit in effect at all times, and **PL[2]** will be ignored.

References

CL[], **LC**, **MC**, **TC**

PO – Positioning Options

PO specifies the optional behavior of the **PA** and **PR** commands.

CANopen/CoE

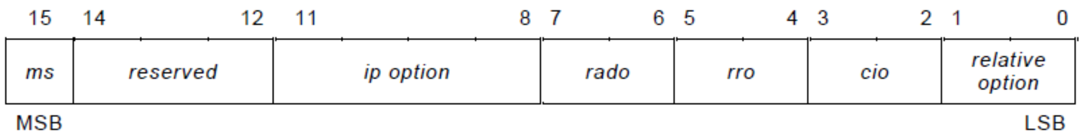
0x60F2

Attributes

Attribute	Description
Type	Unsigned Integer 16, Read/Write
Source	All
Restrictions	Effective on the next call to **BG**
Range	According to bit-field definitions
Default	0
Unit modes	**UM** = 5
Non-volatile	Yes

Remarks

The following diagram shows the bit-field structure of a **PO** value:



LEGEND
ms = manufacturer-specific
rro = request-response option
cio = change immediately option
rado = rotary axis direction option

The *relative option* bits control the behavior of the **PR** command as detailed in the following table:

Value of Bits 0 and 1	Remarks
0x0 (default)	Positioning moves are performed relative to the preceding (internal absolute) target position or relative to the present location if there is no preceding target position (after Motor On).
0x1	Positioning moves are performed relative to the actual position demand value, that is, the output of the trajectory generator.

Value of Bits 0 and 1	Remarks
0x2	Positioning moves are performed relative to the position actual value (**PU**).
0x3	Reserved.

The *change immediately option (cio)* bits are described in the following table:

Value of Bits 2 and 3	Remarks
0x0 (default)	The drive device readapts the actual motion to the new target position immediately.
0x1	The actually performed positioning task will be continued and blended with the newly commanded task when the target position is reached.
0x2	Reserved.
0x3	Reserved.

The *rotary axis direction options (rado)* bits define the behavior of the position modulo:

Mode	Value of Bits 6 and 7	Remarks
Normal positioning	0x0 (default)	Normal rotary positioning is similar to linear axis positioning. If the position range limits (object 0x607B or **XM[1]** and **XM[2]**) are achieved or exceeded, the input value wraps automatically to the other end of the range. Movement greater than the modulo is possible only with this bit combination.
Negative movement	0x1	Positioning only in the negative direction. If the target position is higher than the actual position, the axis moves over the minimum limit of the position range (**XM[1]** or object 0x607B.1) to the target position.
Positive movement	0x2	Positioning only in positive direction. If the target position is lower than the actual position, the axis moves over the maximum limit of the position range (**XM[2]** or object 0x607B.2) to the target position.
Positioning with shortest way	0x3	Positioning with the shortest way to the target position. **NOTE:** If the difference between the actual value and the target position in a 360° system is 180°, the axis moves in the positive direction.

References

PA, PR, BG, XM[]

PP[*N*] – Protocol

PP[] programs all communication parameters for the RS232 and CANOpen protocols.

CANopen/CoE

Not supported

Attributes

Attribute	Description
Type	Parameter, Integer
Source	RS232, CANOpen
Restrictions	**MO** = 0 for **PP[1]**
Index range	1 to 15
Default	See the table below.
Unit modes	All
Non-volatile	Yes

Remarks

Indices

The following table describes the **PP[*N*]** entries.

Index	Description	Type	Values		Restrictions
1	Type of Communication	Integer	1	RS232	PP[1] serves as "Enter Communication Parameters" for RS-232. PP[2] and PP[4] come into effect only when PP[1] is written. The response to PP[1]= # is not the same as the response to all other commands, because the communication type switches while processing the command.
			2	RSVD	

2	RS232 Baud rate	Integer	0	4,800	This parameter has no immediate effect.
			1	9,600	
			2	19,200	
			3	38,400	
			4	57,600	
			5	115,200 (default)	
3	RSVD				
4	RS232 parity	Integer	0	None (default)	
			1	Even	
			2	Odd	
5–12	RSVD				
13	CANOpen device ID	Integer	1 to 127		The default is 127.
14	CANOpen Baud rate	Integer	0	1,000,000	
			1	500,000 (default)	
			2	250,000	
			3	125,000	
			4	100,000	
			5 to 7	50,000	
			8	800,000	
15	CANOpen group	Integer	1 to 128		The default is 128.

Notes

- The number of RS232 stop bits has a fixed value of 1.
- The group ID number for CAN (**PP[15]**) defines the ID of the received message object. The response is transmitted by each node with its own ID (**PP[13]**). Setting **PP[15]** = 128 allows the user to cancel the CAN group ID.
- Unused **PP[*N*]** parameters are reserved for compatibility with other Elmo drives.

References

PR – Position Relative

PR specifies the relative target position.

CANopen/CoE

0x607A

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	• The motor must be on. • Effective on the next call to **BG** • Target position after adding the **PR** value must be within the ranges defined by **VH[3]** and **VL[3]**.
Range	-2^{31} to $(2^{31} - 1)$
Default	0
Unit modes	**UM** = 3; **UM** = 5
Non-volatile	No

Remarks

The **PR** command behaves according to the relative option bits of the **PO** command.

On the next call to **BG** after applying the **PR** command, the following occur:

- The motion control will be switched to the position control loop.
- The drive motion mode will be changed to the Point-2-Point (PTP) motion mode according to **AC**, **DC** , **SF**, **SP** and **FS**.

When **PR** is set and **BG** is called, the motion mode reflected in **OV[2]** (object 0x6061) is changed to 1 (Profile Position).

Object 0x607A will be overridden by the **PR** value. Refer to the **BG** command for more details.

PTP is limited by software limits. Motion will stop at these limits.

If **UM** = 3, **PR** determines the target position in electrical angle units (1 pole pair is 360 electrical degrees, which are denoted by 512 electrical ticks).

The motor will abort if the feedback position is higher than **HL[3]** or lower than **LL[3]**.

The **PR** value can be given in user-defined position units specified by the **FC** command.

References

JV, **FS**, **SP**, **AC**, **DC**, **UM**, **PA**, **PO**, **FC**

PS – Get Program Status

PS reports the status of the user program.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read-only
Source	All, except the user program
Range	-2 to 1 (See the table below.)
Unit modes	All

Remarks

A drive can be in one of the following situations with respect to the user program:

1. No program was ever loaded, or no program exists.
2. The user program was downloaded. It is not running, but it is ready to execute a command.
3. The user program was halted by the **HP** command.
4. The user program is currently running.

PS indicates one of the above descriptions. The following table details these indications:

PS value	Description
-2	The program has not been compiled or does not exist. This would also be the indication in cases in which the program exists but could not be loaded from the non-volatile flash memory during boot-up (power-up or drive reset).
-1	The program exists, is at rest, and is ready to be executed.
0	The program is in the halted state. This can be either a break point (in debug mode) or after HP command.
1	The program is running. This case would also be indicated in **SR** command bit 12.

References

CC

PT[] – Position Table (Reserved)

PT[]

CANopen/CoE

Attributes

Attribute	Description
Type	
Source	
Restrictions	
Range	
Index range	
Default	
Unit modes	
Non-volatile	

Remarks

Indices

The following table describes the **PT[]** entries.

Index	Description	Type	Values	Restrictions
0				
1				
2				

References

PU – Main Position in User-Defined Units

PU reports the present position of the position socket in user-defined units.

CANopen/CoE

0x6064

Attributes

Attribute	Description
Type	Integer, Read-only
Source	All
Restrictions	None
Range	-2^{31} to $2^{31} - 1$
Default	According to the value of the sensor. An incremental encoder starts counting from zero.
Unit modes	All
Non-volatile	No

Remarks

PU returns the position of the position socket in user units after applying the position scaling factor (**FC[]**) and the position modulo (**XM[]**).

References

PX, **FC[]**, **XM[]**

PV – Position Velocity Time setting (Reserved)

PV[]

CANopen/CoE

Attributes

Attribute	Description
Type	
Source	
Restrictions	
Range	
Index range	
Default	
Unit modes	
Non-volatile	

Remarks

Indices

The following table describes the **PV[]** entries.

Index	Description	Type	Values	Restrictions
0				
1				
2				

References

PX – Main Position in Counts

PX reports the present position of the position socket in internal units (counts).

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	None
Range	-2^{31} to $2^{31} - 1$
Default	According to the value of the sensor. An incremental encoder starts counting from zero.
Unit modes	All
Non-volatile	No

Remarks

The position sensor must be set by pointing to the relevant sensor (**CA[45]**) and setting the sensor **CA[41-44]**.

References

FP[], **OV[], CA[]**, **PU**

PY – Auxiliary Position in Counts

PY sets/ reports the present position of the external (auxiliary) reference in counts.

CANopen/CoE

0x20A0

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	None
Range	-2^{31} to $2^{31} - 1$
Default	According to the value of the connected sensor/profile.
Unit modes	All
Non-volatile	No

Remarks

The auxiliary position **PY** can be referenced by pointing to:

- the relevant sensor (**CA[79]**) set by **CA[41-44]**, if **EM[11]:bit3**=0.
- the main profiler (including CAN Encoder) output, if **EM[11]:bit3**=1.

PY counts cyclically (refer to the **YM[N]** command).

If the **PY** value setting is outside the range [**YM[1]…YM[2]],** the requested value will be ignored and **PY** will not change

References

YM[N], EM[N], CA[N]

RC – Recorder Variables

RC specifies which of the mapped signals are to be recorded.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All, except CoE
Restrictions	Recorder inactive (RR=0 or RR=-1)
Range	Bit field [bits 0 to 15]
Default	0
Unit modes	All
Non-volatile	No

Remarks

The drive can record a range of signals for performance verification and debugging. The first step of the recording process is to define the recorded variables by assigning a value to **RC** (a bit field). Each activated bit in the representation of **RC** defines a signal to be recorded. A valid **RC** value defines at least one recorded variable. **RC** can map up to sixteen variables that are to be recorded.

The host can map many optional variables to any bit of **RC** from bit 0 to bit 15.

If the drive has stored previously recorded data, setting **RC** will invalidate this data. Invalidated data cannot be retrieved.

The total number of data points that can be recorded is fixed. Therefore, the number of points per signal depends on the number of signals that are recorded simultaneously: the more signals recorded, the fewer are the points that are available for each signal.

References

RG, **RL**, **RP[*N*]**, **RR**, **BH**

RG – Recorder GAP

RG specifies the frequency per sampling time that the recorder is activated.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All, except CoE
Restrictions	The recorder must be inactive (**RR** = 0 or **RR** = -1).
Range	1 to 65535
Default	1
Unit modes	All
Non-volatile	No

Remarks

Because the recorder has a limited storage capacity (16K), if it operates at the sampling time of the drive, the recorder will operate for a very short time. To achieve longer recording times, the time interval between consecutive data recordings must be increased. The **RG** parameter trades recording resolution for increased recording time. When **RG** = 1, the sampling time of the recorder is given by the **WS[29]** command.

Be aware that the recorder sampling time depends on the **TS** value and the specific unit mode (**UM**).

If the drive has stored a previously recorded data vector, setting **RG** will invalidate this data. Invalidated data cannot be retrieved.

References

RC, **RL**, **RP[*N*]**, **RR**, **BH**, **TS**, **WS[29]**, **UM**

RL – Recorder Length

RL specifies the total length of the recorded data.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All, except CoE
Restrictions	The recorder must be inactive (**RR** = 0 or **RR** = -1).
Range	1 to 16384
Default	16384
Unit modes	All
Non-volatile	No

Remarks

If the recorder is set with a length larger than the maximum value (16K), each signal will be set according to the formula **RL**(16K)/number of vectors.

For example, in the case of **RL** = 16384, the maximum recorder length for each signal will be as follows:

Number of recorder signals	Maximum recorder length per signal
1	16,384
2	8,192
…	
8	2048
9	1820
10	1638
…	
15	1092
16	1024

The actual size of the recorded data is returned by the **WI[21]** command.

If the drive has stored a previously recorded data vector, setting **RL** will invalidate this data. Invalidated data cannot be retrieved

References

RC, **RG**, **RP[*N*]**, **RR**, **BH**

RM – Reference Mode

RM specifies the use of an external reference generator (ECAM/Follower).

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All, except CoE
Restrictions	None
Range	0, 1
Default	0
Unit modes	All
Non-volatile	Yes

Remarks

RM also specifies whether the motor can be enabled by using the Inhibit/Enable function of a digital input. Refer to the **IL** command for more details.

The following table describes the possible values of the **RM** command.

Command Value	Description
0	External reference generator is disabled. If a digital input is assigned to the Inhibit /Enable function, the motor will not be enabled by the switch.
1	External reference generator is enabled. If a digital input is assigned to the Inhibit /Enable function, the motor will be enabled by the switch.

Notes

- When setting **RM=1**, the ECAM/Follower position output jump is prevented. The Velocity output can be changed in steps, e.g. in Follower mode or when the ECAM is enabled in the middle of the table.

- After setting **RM=0**, the ECAM/Follower position output maintains the latest value, with the velocity and acceleration outputs set to 0.
- Setting **RM**=1 resets ECAM LPF and FIR filters. For filters details refer to **EM[N]** and **KV[N]** commands.

References

IL, **MO**, **KV[N]**

RP[] – Recorder Parameters

RP[] enables complete specification of how the recorder is triggered and how the recorded data is transferred to the host.

CANopen/CoE

Attributes

Attribute	Description
Type	Parameter, Long
Source	All, except CoE
Restrictions	• According to the description table • The recorder must be inactive (**RR** = 0 or **RR** = -1).
Range	According to description table
Index range	0 to 15
Default	According to the description table
Unit modes	All
Non-volatile	No

Remarks

Trigger definitions:

The recorder is started by a trigger event, which may be one of the following:

- *Immediate*:
  The recorder starts immediately after the recording request is issued.
- *Triggered by an analog signal*:
  The recorder starts upon one of the following events:
  - *Positive slope*:
    The signal crosses a prescribed level with a positive slope.
  - *Negative slope*:
    The signal crosses a prescribed level with a negative slope.
  - *Window*:
    The signal exits a window of two prescribed signal levels.
  - *Digital inputs*:
    Digital inputs are switched to their active logic state as defined by the **IL[*N*]** command.

- *Motion begins:*
  A **BG** command, or activation of a hardware **BG** command (refer to the **IL[*N*]** command).
- *Time*:
  The recorder starts after a requested time (in milliseconds) is reached.
- *On the fly*:
  The recorder begins to record immediately after the recording request is issued and uploads data until it is stopped by request.

Trigger delay:

The trigger defines when the recorder is started. The recorder can be programmed to start before the trigger event, so that the trigger event can be caught “in the middle of the action.” This is possible because the recorder starts to record at the instant when it is launched by the **RR** command, so that when the trigger event occurs, the pre-trigger information is already recorded.

Indices

The following table describes the **RP[*N*]** entries.

<table>
<tr><th>Index</th><th>Description</th><th>Type</th><th colspan="2">Values</th><th>Restrictions</th></tr>
<tr><td rowspan="2">0</td><td rowspan="2">Time quantum base</td><td rowspan="2">Integer</td><td>0</td><td>Time quantum is 2*TS</td><td rowspan="2"></td></tr>
<tr><td>1</td><td>Time quantum is TS</td></tr>
<tr><td>1</td><td>Trigger variable, which is defined similarly to RC, but only 1 bit may be non-zero. The trigger variable does not need to be one of the recorded variables.</td><td>Integer</td><td colspan="2">1 to 65535</td><td></td></tr>
<tr><td>2</td><td>Pre-trigger, the percentage of the recorded signal that is recorded before the trigger event [%]</td><td>Integer</td><td colspan="2">0 to 100</td><td></td></tr>
</table>

3	Trigger type	Integer	0 Immediate 1 Analog signal 2 Positive slope 3 Negative slope 4 Window 5 Reserved 6 Reserved 7 Digital input 8 Motion begin	
4	Level 1 for a positive-slope trigger, or the high side for a window trigger	Long		
5	Level 2 for a negative-slope trigger, or the low side for a window trigger.	Long		
6	Level of digital input when used as trigger for the recorder	Integer	1 to 0xFFFFFFF	
7	Digital input mask, defines which digital inputs trigger the recorder	Integer	1 to 0xFFFFFFF	
8	Lower buffer index for recorded data upload transmission	Integer	0 to 16384	When **RP[9]** = **RP[8]** = 0, all of the buffer is transmitted.
9	Higher buffer index for recorded data upload transmission.	Integer	0 to 16384	
10	Time value for start recording [msec]	Integer	Reserved	Used only if **RP[4]** = 5
11	Selected recorded signal for the **BS[]** command. Defined similarly to **RC**, but only 1 bit may be non-zero.	Integer	1 to 65535	Recorder ready (**RR**==0); Selected signal is recorded.
12	Reserved			
13	Reserved			
14	Reserved			
15	Reserved			

Notes

If the drive has stored a previously recorded data vector, setting **RP[*N*]** (with *N* equal to a number other than 8 or 9) will invalidate this data. Invalidated data cannot be retrieved.

When the recorder trigger is set to Digital input **(RP[3]=5)** the trigger is armed when at least one of the masked digital input indication (IP command) marked by **RP[7]** was changed and the value of masked inputs are equal to **RP[6]: (IP&RP[7]) == RP[6])**.

References

RC, **RG**, **RL**, **RR**, **BH**, **BS[]**

RR – Activate Recorder / Recorder Status

RR launches the recorder, kills an on-going recording process or retrieves the recorder status.

CANopen/CoE

Attributes

Attribute	Description
Type	Parameter, Integer
Source	All, except CoE and Program
Restrictions	The recorder must be inactive (**RR** = 0 or **RR** = -1)
Range	0 to 3
Default	-1
Unit modes	All
Non-volatile	No

Remarks

The **RR** command has the following options:

Value	Description
0	Kill the recorder
1	Start recording upon the next **BG** command
2	Start recording immediately
3	Arm the recorder with the trigger setting of the **RP[3]** command
4	Reserved

The **RR** command may report the following values:

Value	Description
-1	There is no valid data in the recorder.
0	The recorder is ready or has finished and is ready with valid data.
1 to 3	The recorder is waiting for completion of the trigger event, respectively.

Notes

The recorder buffer is shared with UL command that uploads data from the drive. When UL is used the RR is set automatically to -1.

References

BH, **RP[*N*]**, **RC**, **RG**, **RL**

RS – Soft Reset

RS initializes the drive parameters to their factory defaults and resets all volatile variables to their power-on defaults.

CANopen/CoE

Attributes

Attribute	Description
Type	Command
Source	All, except CoE
Restrictions	**MO** = 0. The user program must be inactive (PS=-1 or -2).
Unit modes	All
Non-volatile	No

Remarks

RS does not change the communication settings; therefore, after executing **RS**, it is still possible to communicate with the drive. The communication parameters, however, are reset.

The **RS** command disables the communication routines for a few milliseconds. If **RS** is executed by a USB command, the EtherCAT message may be lost in the execution interval.

The **RS** command modifies only the RAM contents; it does not affect the flash memory.
If necessary, use the **SV** command to store the default **RS** permanently.

The default parameters are designed so that after RS is enabled, the motor should not produce a torque command. Normally, the Motor On command should return an error.

References

LD, **SV**

RV[*N*] – Recorder Variables

RV[*N*] maps recorded variables to the recorder through the **RC** command. By setting **RV[*N*]** = *X*, the variable *X* is assigned to bit *N* - 1 of **RC** in the variable static table. The default mapping (power on) of **RV[1]** to **RV[16]** behaves similarly to those in previous product lines. The full list of variables available to the recorder is stored in the serial flash memory of the *Gold* or *SimplIQ* drive and can be uploaded using the **LS** command.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Bit field
Source	Program, all except CoE
Restrictions	The recorder must be inactive.
Range	According to the index in the static variable table
Index range	1 to 16
Default	1 to 16, respectively
Unit modes	All
Non-volatile	Yes

Remarks

Text

Index	Description	Type	Values	Restrictions
1 to 16	Recorder variable entry	Integer	1 to 16, respectively	None

References

RC, BH, RR

SC[] – Stepper Commutation

SC[] gets and sets parameters used for some functions in stepper mode.

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Restrictions	None
Range	Define in Table
Index Range	1 to 8
Default	See the table below.
Unit Modes	All modes (except for **SC[8]**, used only in stepper mode)
Non-volatile	Yes

Remarks

With stepper commutation (**CA[17]** = 2) this command increases the current in the stepper angle to a certain value (30 degree), waits until the speed is stable and then sets the stepper angle to 0, and waits again until the speed is stable. At this point the commutation phase is known and saved. After the current decrease back to 0, the unit mode changes to the original mode that was requested.

With binary search commutation (**CA[17]** = 3) the commutation angle converges until it is found. There is almost no movement in this process.

With the motor on in stepper mode (**UM** = 3), it automatically sets the current command for stepper mode.

The activation of **SC[]** occurs in the next motor enable.

Indices

The following table describes the **SC[]** entries.

Index	Description	Unit	Default	Range	In Process
1	Desired current in the process	Percentage of the maximum current (**PL[1]**)	50	0 to 100	Stepper and binary search commutation
2	Time to increase from 0 to the desired current	Seconds	1.0	0.001 to 6	Stepper and binary search commutation
3	Time to stabilize the motor. Stepper commutation: If the speed is below the value of **SC[5]** for a period of **SC[3]** seconds, the motor is stable, and the commutation angle can be calculated. Binary search: After the algorithm finds the angle, it increases the current for **SC[3]** seconds, t o make sure that the position in this angle will stabilize.	Seconds	1.0	0.001 to 6	Stepper and binary search commutation
4	Time to decrease the current back to 0	Seconds	1.0	0.001 to 6	Stepper commutation
5	Low speed defined as motor not moving	Electrical cycles/seconds	1.0	0.1 to 10	Stepper commutation
6	Threshold phase for stepper method	Electrical angle/512	5.0	1 to 43	Binary search commutation
7	Minimum movement used to define the direction of motor movement	Electrical angle [degree]	2.0	0 to 180 (zero not included)	Binary search commutation
8	Automatic set current command at motor on in stepper mode (**UM** = 3)	Amperes	0	-**CL[1]** to **CL[1]**	Motor on in stepper mode

References

PL[], **CA[]**, **MO**, **UM**

SD – Stop Deceleration

SD specifies the deceleration used during emergency stops.

CANopen/CoE

0x60C5, 0x60C6, 0x6085

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	None
Range	100 to2e9
Default	1e9
Unit modes	All
Non-volatile	Yes

Remarks

The **SD** value is the deceleration which is used during emergency stops, such as Limit Switch and ST.

The **SD** value is used to limit the total acceleration and deceleration of the profiler and auxiliary references.

The **SD** value should be set to the maximum acceleration and deceleration that the motor can force on the load.

On **BG**, **SD** overrides the CANopen maximum acceleration/deceleration (object 0x60C5/0x60C6) and the Quick Stop deceleration (object 0x6085).

AC or **DC** will be overridden by **SD** values if they are higher.

The **SD** value can be given in user-defined units specified by the **FC** command.

References

AC, **DC**, **FC[]**

SE[*N*] – Sine Excitation

SE[*N*] generates an internal signal command as a reference to the controller.

CANopen/CoE

Attributes

Attribute	Description
Type	Float
Source	All
Restrictions	None
Range	According to the **SE[]** table below
Index range	1 to 7
Default	According to the S**E[]** table below
Unit modes	All
Non-volatile	No

Remarks

Set a virtual sensor. The sine excitation generates a signal that consists of three elements:

- Two frequencies sinus signals denoted as A amplitude and B amplitude,
- An offset DC signal denote as DC value.

These signals can be used as position, velocity or current for reference, emulation or any other general purpose according to **SE[1]**.

The units are counts, counts/seconds or amperes, respectively.

Setting **TW[80]** = 1 activates the signal (can be changed on the fly). It first ramps to the **DC** value, and then builds the sine waves.

Setting **TW[80]** = 0 stops the signals. It first stops the high and low sine, and then the **DC** offset ramps to zero according to **SE[7]**.

The sine excitation has a function ID of 8, and it must be mapped to a socket. For example, to map socket 4 for sine excitation set **CA[44]** = 8.

The socket then can be used to any other function that can operate on sockets.

When the command is used for sine excitation, it is directed to the relevant control loop (according to **CA[68, 69, 70]**) and is not limited by the Stop Manager limits and no limits of command is checked.

The function starts to work regardless of the presence of a brake in the system.

Indices

Following table describes the **SE[*N*]** entries:

Index	Description	Units	Default	Values	Notes
1	Units of virtual sensor	1. Current [ampere] 2. Velocity [counts/sec] 3. Position [counts]	1	1 to 3	
2	A amplitude	Depends on input For example: Position: 1000 counts Velocity: 10,000.0 counts/sec Current: 0.25 amperes	0	N/A	Values can be positive or negative. Drive does not limit the entry.
3	A frequency	Hz	100	0 to 5000	
4	B amplitude	Same units as A amplitude	0	N/A	Same as A amplitude
5	B frequency	Hz	0	0 to 5000	
6	DC Value	Same units as A amplitude	0	N/A	Same as A amplitude
7	Slope to DC value	Same as DC value/sec	0	N/A	The time that takes to ramp from 0 to DC carrier value or to 0 when mode is disabled

References

TW[*N*], **CA[*N*]**

SF – Smooth Factor

SF specifies the motion smoothing factor, in milliseconds, for PTP and jogging.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	None
Range	0 to 63
Default	0
Unit modes	All
Non-volatile	Yes

Remarks

The **SF** command smooths the motion to prevent sharp and high dynamic speed changes. **SF** actually builds the acceleration for the specified milliseconds, allowing the acceleration to effect the motion in moderate portions. When **SF**=0, all the requested acceleration is used by the profiler to build the speed command.

A total of 63 msec is allowed, this is actually limited by the moving average buffer of 255 entries of 250 µsec each.

References

AC, DC, PA, PR, JV, JP

SO – Servo Enabled

Refer to **MO**/**SO** in this manual.

CANopen/CoE

When **SO** is set to 1, the Operation Enable state is reported in object 0x6041.

Attributes

Attribute	Description
Type	Integer, Read
Source	All
Restrictions	None
Range	0 1 (when the drive is ready to handle a profiler set point)
Default	
Unit modes	N/A
Non-volatile	No

References

MO

SP – PTP Profiler Speed

SP specifies the configured velocity normally reached at the end of the acceleration ramp during a profiled motion. **SP** is valid for both directions of motion.

CANopen/CoE

0x6081

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	Effective on the next call to **BG**
Range	1 to 2e9
Default	100000
Unit modes	**UM** = 5
Non-volatile	Yes

Remarks

SP is used during PTP motion to limit the speed of the profiler. In cases in which the **SP** value is too high for the position target, a triangular motion will be performed by the profiler. If the **SP** value is low enough, a trapezoidal motion will be performed.

On **BG**, **SP** overrides CANopen object 0x6081.

The **SP** value can be given in user-defined units specified by the **FC** command.

References

AC, **DC**, **PA**, **BG**, **FS**, **FC**

SR – Status Register

SR reports the status of different functions in the drive. It returns a snapshot of the system status. Most of the information returned by **SR** can be retrieved by other commands, and the purpose of the **SR** command is to assemble this status in a single variable.

CANopen/CoE

Object 0x1002 returns the same function.

Note: Object 0x6041 returns the status with respect to DS-402 functionalities.

Attributes

Attribute	Description
Type	Read, Bit-field
Source	All
Restrictions	None
Range	None
Default	None
Unit modes	All
Non-volatile	No

Remarks

The following table details the different functions in the bit-field format.

Bits	Description	Values		Notes
0 to 3	Amplifier Status - reports the instantaneous state of the power drive.	See details in the table below.		In cases in which the value differs from 0, the Red LED is set, and the **AOK** function is active. See **OL[]** for details regarding the **AOK** function.
4	The motor is enabled and ready for a profiler (**SO**)	0	The servo is not enabled.	.In some cases the **MO** value can be 1 while **SO** is still 0. The status differs between the two.
		1	The servo is enabled	
6	A fault occurred while the	See the **MF** command.		This bit is cleared.

	motor was enabled.			during the Motor Enable procedure.
7	In Elmo's homing or capture sequence	0	**HM[1]** and **HF[1]** are not active.	The command does not reflect the DS-402 homing mode or the DS-402 touch probe function.
		1	**HM[1]** or **HF[1]** is active.	
8 to 11	Reports the actual profiler according to the motion mode.	0	No motion was selected.	Bits actually reflect the "Mode of operation display" - object 0x6061 of CANopen DS-402. Profilers are activated after **BG** or according to DS-402 profiling method.
		1	Profile position mode (PTP)	
		2	N/A	
		3	Profile Velocity mode (JV)	
		4	Profile Torque mode (TC)	
		5	N/A	
		6	Homing mode (DS-402 only)	
		7	Interpolated position mode (PV)	
		8	Cyclic sync position mode (DS-402 only)	
		9	Cyclic sync velocity mode (DS-402 only)	
		10	Cyclic sync torque mode (DS-402 only)	
12	User Program is running	0	No user program or program at rest	The bit is set according to the **PS** command.
		1	The user program is running.	
13	Current Limit is on	0	No Current Limit	
		1	Current is limited to **CL[1]**.	

14	Safety Input 1 (STO_DSP)	0	The drive is in safety state. The motor cannot be activated.	Safety state is reflected in bits 0 to 3. If motor was on during the safety state, **MF** latches the event and the motor will be turned off.
		1	The drive is not in safety state. The motor may be enabled.	
15	Safety Input 2 (STO_PWM)	0	The drive in safety state. The motor cannot be activated.	
		1	The drive is not in safety state. The motor may be enabled.	
16 to 17	Recorder Status	0	The recorder is not active.	
		1	Waiting for a trigger.	
		2	The recorder has completed its task. Valid data is ready for uploading.	
		3	Recording is now active. Data is been fetched by the drive.	
18	Target Reached	According to **TR[]** and the relevant motion mode (**OV[2]**) for Profile Velocity or Profile Position mode, the same bit will be set:		**Note: BG** clears this bit.
		0	The target is not reached.	
		1	The target is within the **TR[]** boundaries.	
24 to 26	Hall A, Hall B, Hall C state			
28	The profiler stopped due to a switch.	Either a Hard Stop, FLS, RLS or Soft Stop function caused the profiler to stop.		

* Unused bits are reserved and are set to 0.

The following table details the indication of the amplifier status bits in **SR** command.

A CAN EMCY message will be transmitted if the motor was enabled prior to the indication.

SR bits 0 to 3 Value (Hex)	Description	Type CAN EMCY (Hex)	Notes
0	All OK		
3 (0x3)	Undervoltage: The amplifier is not measuring the minimum required voltage.	5 3120	• The minimum allowed value is reported in the **WI[38]** command. • The actual bus voltage is reported in **AN[6]**.
5 (0x5)	Overvoltage: The amplifier is measuring a voltage which is higher than the maximum allowed value.	5 3310	• The maximum allowed voltage is reported in **OV** command. • The actual bus voltage is reported in **AN[6]**.
7 (0x7)	Safety: One or two of the safety inputs are in safety state.	5 FF20	Safety indications are reported in **SR** bits 14 and 15.
11 (0xB)	Short Protection: The current has exceeded a range which is considered as a phase to phase or phase to ground short.	3 2340	This instantaneous fault is measured by the hardware and typically cannot be recorded or indicated outside of the **MF** command.
13 (0xD)	Overtemperature: The drive is sensing a temperature which exceeds the maximum allowed temperature limit.	9 4310	The actual temperature is reported by the **TI[1]** (**TI[2]** in Fahrenheit) command.

Notes

- In the case of the Safety bits in the Gold Guitar and the Gold Trombone there might be situations in which STO 2 is not reported to the CPU. In these cases, when Safety 2 is activated (safety state), the PWM HW will be inhibited, causing the motor to be in a freewheel state, but the CPU will not be aware of it and will not report it to the user. Typically, a tracking error would be triggered.
- The indication that the HW is suffering from that the user can detect by **PB**. If the value is 0x5AA5 (23205), STO 2 will not be reported.

- STO 1 works normally in all drives.

References

MF

ST – Stop Profiler

ST stops the profiler in stop deceleration.

CANopen/CoE

Attributes

Attribute	Description
Type	Command
Source	All
Restrictions	None
Range	N/A
Default	N/A
Unit modes	**UM** = 2, 3, 5
Non-volatile	N/A

Remarks

The **ST** command will stop the profiler (software) immediately at the value specified by the **SD** command.

ST has no effect over the auxiliary command.

When profiler is in the Torque mode (**TC**), **ST** will stop the profiler by forcing torque command to 0.

References

BG

SV – Save Parameters

SV saves non-volatile parameters from the RAM to the flash memory.

CANopen/CoE

0x1010 data bytes 0 to 3 's', 'a', 'v', 'e'

Attributes

Attribute	Description
Type	Command
Source	All, except the user program
Restrictions	**MO** = 0, and the user program is not running.
Range	None
Default	N/A
Unit modes	All
Non- Volatile	None

Remarks

The **SV** burns the parameters into the none volatile memory after it checks the parameters integrity. If one of the parameters fails and an error is produced, the saving procedure is not performed. The **CD** command details the parameter which failed and the reason for the failure.

The **SV** command may take a few hundreds of milliseconds to execute, during which the communication drivers are disabled.

References

LD, **CD**

SW – Status Word

SW informs the user regarding the status of the DS-402 (ox6041), via the Status Word.

CANopen/CoE

0x6041

Attributes

Attribute	Description
Type	Bit-Field, Read/Write
Source	USB, TCP
Restrictions	None
Range	N/A
Default	0
Unit modes	All
Non- Volatile	No

Remarks

The Status Word is used in conjunction with the Control Word in the DS-402 CANopen standard for drives and motion. The Status Word is received with the CANopen or EtherCAT communication channel with object 0x6041.

The **SW** 'get' command should normally be followed by the **CW** 'set' command, in order to be synchronized with the DS-402 state machine principles.

For more details about the **SW** bit field, refer to the object 0x6041 in the CANopen DS-402 manual.

References

CW

TC – Torque Command

TC specifies the torque command and switches to the current control loop.

CANopen/CoE

Attributes

Attribute	Description
Type	Float, Read/Write
Source	All
Restrictions	The motor must be on.
Range	Torque limits (-PL[1] to PL[1])
Default	0, Volatile. Cleared automatically when **MO** = 1
Unit modes	All
Non-volatile	N/A

Remarks

The **TC** command sets the torque (motor current) command, in amperes, in all modes. When the drive is in position or velocity mode, the **TC** command transfers it to current mode.

TC commands are accepted in the range permitted by the present torque command limits (refer to the **PL[*N*]** and **CL[*N*]** commands). If **TC** is set greater than **CL[1]**, after a few seconds, the current limit of the servo drive will drop to **CL[1]**. In this case **LC** will indicate 1, notifying that there is a current saturation state.

If **TC** is higher than **CL[1]** while at the limit (**LC** = 1), the command will fail.

TC defines the reference value **IQ** (the **ID** command is zero unless phase advanced was used).

References

MO, **UM**, **IQ**, **ID**, **CL[]**, **PL[]**,**MC**, **LC**

TI[] – Temperature Information

TI[] gets temperature information from the drive hardware.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read-only
Source	All
Restrictions	None
Range	**TI[1]** range: –40°C to 100°C (Celsius) **TI[2]** range: –40°F to 212°F (Fahrenheit) **TI[3]** range: –40°C to 120°C (Celsius)
Index range	1 to 3
Default	N/A
Unit modes	All
Non-volatile	No

Remarks

The drive hardware includes temperature sensors to measure the temperature of the driver's heat sink.

In cases in which the temperature exceeds 85°C, the servo will be shut off (**MO** = 0) with an over-temperature indication in the **MF** command and in the **SR** command.

The servo, in this case, can be enabled again when the temperature is ≤80°C.

In some cases when absolute encoders are used, the encoder manufacturer provides the sensor temperature and some more information about possible errors. Please refer to the **EE[]** command for more information.

In cases in which the sensor reports an error, the motion will be aborted with the proper **MF** command value.

Indices

The following table describes the **TI[]** entries.

Index	Description	Type	Values	Restrictions
1	Reads the heat sink temperature in Celsius	Integer	Celsius	
2	Reads the heat sink temperature in Fahrenheit	Integer	Fahrenheit	
3	Reads the BISS Absolute encoder temperature	Integer	Celsius	

References

TM – Internal Time

TM reads and writes the 32-bit system time, in microseconds.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	
Range	0 to 2^{32}
Default	0
Unit modes	All

Remarks

The **TM** command is based on the Real Time process internal counter, which counts time in microseconds.

This counter can be set and read using the **TM** command.

The **tdif( )** function uses the **TM** internal counter to calculate time differences.

Users can measure time periods of activities using the **TM** command and the **tdif()** function.

For example (using parameter **UI[*N*]**):

```
UI[1]=TM
```

… any activity

```
UI[2]=tdif(UI[1])
```

UI[2] holds the time period of the measured activity, in milliseconds.

In the absence of CANopen SYNC signals and Stamp time objects, the microsecond counter runs freely, completing a cycle approximately once every 71.5 minutes.

In a CANopen system the high-resolution time stamp protocol may modify the internal time in order to synchronize between drives on the CAN bus.

The **tick** function also gets the system time of the Gold drive, but it is based on a different timer counter (hardware timer) and returns the time in microseconds.

TR[] – Target Radius

TR[] specifies the threshold which determines whether the load has reached its target.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	None
Range	See the table below.
Index range	1 to 4
Default	See the table below.
Unit modes	**UM** = 5, **UM** = 2
Non-volatile	Yes

Remarks

The Target Reached indication informs the host that the load has reached the target with respect to the criteria of window and time around the target.

TR[] is the interface for specifying the target and time window in the target.

It is with conjunction of the CANopen target reached definition for position and velocity. The difference is that the CANopen objects are declared in User Units, while **TR[]** is given in counts and counts/sec.

When the feedback count meets the desired target window for the window time, the Target Reached bit is set.

Note: The Target Reached indication is the same for either profile velocity mode or profile position mode. Depending on the motion mode (**OV[2]** or object 0x6061), the relevant procedure is evaluated.

The Target Reached bit is indicated in CANopen status word object 0x6041 bit 10. It can also be retrieved by the **SW** command, which is a mirror image of the status word.

Target Reached is also indicated in **SR** register bit 18.

Target Reached is evaluated every 250 µsec. In between readings, no indication is given.

Indices

The following table describes the **TR[]** entries and the relevant CANopen objects.

Index	Description	Values [units]	Default	CANopen object
0	Reserved			
1	Target Position window [counts]	-1 not active 0 to $2^{31} – 1$ [counts]	100 counts	Overrides object 0x6067
2	Target Position Window time	> 0 [msec]	20 msec	Overrides object 0x6068
3	Target Velocity window	> 0 [counts/sec]	100 counts/sec	Overrides object 0x606D
4	Target Velocity window time	> 0 [msec]	20 msec	Overrides object 0x606E

References

SR

TS – Sampling Time

TS specifies the shortened sampling time of the drive, in microseconds, which is used as the update time of the current controller.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	**MO** = 0, and the user program is not running.
Range	40 to 120
Default	50 (RS), Non-volatile
Unit modes	All
Non-volatile	Yes

Remarks

TS is the sampling time of the current loop. The sampling time of the velocity and position controller is two times **TS**. For example, if **TS** = 50, the torque/commutation controller runs once every 50 microseconds, the speed and position controller executes every 100 microseconds.

The selection of **TS** is a compromise between high servo performance and the scan loop (background) operations, such as the user program and interpreter responses. A low **TS** enables the drive to achieve more control bandwidth, but at the same time, it increases the computational burden on the CPU, so that less computing power remains for executing interpreter and user program commands.

The drive does not allow an excessively low value for **TS** to prevent an overflow of the required CPU computing power.

For all unit modes, **WS[28]** gives the actual sampling time of the speed controller, and **WS[55]** gives the actual sampling time of the position controller.

When **TS** is modified, the controllers loop gains must be retuned in order to prevent instability of the controllers (current, velocity and position), which may damage the drive and/or the motor.

TS must be an even number (when **XP[2]** = 4)

All speed values are calculated with multiplication by a factor of **TS**.

The PWM frequency (in Hz) is calculated using the formula **XP[2]**/(2***TS**), and the current ripple frequency at the motor is twice as large.

References

WI[], **WS[]**

TW – Wizard Internal Identification

TW[*N*] is used to set the internal parameters during running of a Wizard and Simulation.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Write-only
Source	All
Restrictions	
Range	
Default	Depends on the variable
Unit modes	
Non-volatile	All

Remarks

All values and functions set by **TW[]** are reset to default, during:

- Power-up
- The **RS** command
- The **SV** command

Indices

The following table describes the **TW** entries.

Index	Description	Units	Default	Restrictions
3	Minimum speed to activate dynamic brake	Counts/ sec	0	
4	Current loop identification. Starts a new recorder session with a different value for **XP[6]**	Hz	**XP[6]**	The motor must be on.
6	ADC instant timing	150 MHz clock	**WS[9]**	The motor must be off.
7	Sets speed experiment			The motor must be on.

8	Select s profile for the **BG** command 0: Main profile 1, 2: Phasing profile 3: Socket profile (**ID** = 26)		0	
14	Sets command to Gantry Yaw	Counts	0	The motor must be on
15	Socket number (1 to 4) of resolver for starting calibration of the resolver excitation signal. The resolver read function is changed.			The motor must be off
16	Sets the main status LED of the drive: 0: Default normal use 1: LED always off 2: LED always green 3: LED always red			
17	Socket number (1 to 4) of Hiperface to reset position to the sensor serial absolute position.			The motor must be off or in stepper mode
18	Socket number (1 to 4) of Panasonic or Tamagawa to reset the sensor serial absolute position.			The motor must be off
23	Resets the 16 Analog In the offsets entry of **DI**			
24	Sets the 16 Analog In the gains entry of **DI** to 1			
31	DC_EXP 1: Voltage command experiment 2: Switch gate experiment 3: Skip profiler 4: Cancel skip profiler			Not for user
32	User-generated motor fault. A way to simulate a Motor Fault Event: Bit 0:			Use this command in simulation mode to simulate the behavior of the drive during an exception.

	Main - Encoder error. Bit 1: Unused (reserved for Aux encoder error). Bit 2: Encoder - Hall sensor mismatch. Bit 3: 1 Peak over current Bit 4: External inhibit, abort Bit 5: FPGA alarm Bit 6: Digital Hall sensor problem Bit 7: Speed error limit Bit 8: Position error limit Bit 9: Cannot start because of bad database Bit 10: Bad ECAM table Bit 11: Motor was disabled due to a node guarding event. Bit 12: Bridge failure from analog ASIC Bits 12 to 15: 3 – Undervoltage 5 – Overvoltage 7 – Safety error 11 – Short 13 – Overtemperature Bit 16: Cannot find zero position without DHalls			

	Bit 17: Overspeed Bit 18: Stack overflow Bit 19: Null interrupt Bit 20: Timing error Bit 21: Motor stuck Bit 22: Out of position limits Bit 23: Out numerical overflow Bit 28: Cannot tune current offsets Bit 29: Cannot start motor because of an internal problem Bit 31: Reference exceeded the Modulo			
33	Defines minimum **TS**			
34	Under voltage low limit. Use this command to set the Under Voltage protection to 0. This allows the servo (**MO** = 1) to be enabled without the main power.			
36	Determines **OF[*N*]** command report style 0: Report **OF[*N*]** from the **OF[*N*]** array 1: Report **OF[*N*]** from the relevant CANopen object which is reflected			Values: 0, 1

40	Used for Motor Simulation process. Set PWM command to zero.			
41	Used to simulate a sensor in Motor Simulation Mode. Set Kt/J value			
60	Sets shoot through. Value can be read by **WS[12]**	150 MHz clock		The motor must be off.
71	Sets 0x2F42 sync object			
75	Sets **ZX** low index for experiment			
76	Sets **ZX** high index for experiment			
77	Sets **ZX** start index for experiment			
78	Sets array size for the **ZX** reference table for experiment			
79	Time to repeat velocity profile during the speed identification phase			
80	Launch sine profile **SE[*N*]**, sensor **ID** = 8			
81	Launches the routine to identify the quality of the frequency during wizard identification . Bits are according to recorded vectors.			

References

UF[*N*] – Float User Interface

UF[*N*] specifies the float user variable array.

CANopen/CoE

Attributes

Attribute	Description
Type	Parameter, Float, Read/Write
Source	All
Restriction	None
Range	9.999999E+36 to -9.999999E+36
Default	None
Index range	1 to 24
Unit modes	All
Non-volatile	Yes

Remarks

Users can use the 24 indexed entries of **UF[]** to keep floating-point values in the non-volatile memory.

UF[] can be used for the user program as well. Users can modify the value and manipulate the user program flow in a simple way.

References

UI[*N*]

UI[*N*] – Integer User Interface

UI[*N*] specifies the integer user variable array.

CANopen/CoE

Attributes

Attribute	Description
Type	Parameter, Long, Read/Write
Source	All
Restriction	None
Range	-2147483647 to 2147483647
Default	None
Index range	1 to 24
Unit modes	All
Non-volatile	Yes

Remarks

Users can use the 24 indexed entries of **UI[]** to keep integer values in the non-volatile memory.

UI[] can be used for the user program as well. Users can modify the value and manipulate the user program flow in a simple way.

References

UF[*N*]

UM – Unit Mode

UM specifies the higher allowed control loop.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	The motor must be off. Note that control loops can be freely switched without disabling the motor.
Range	1 to 6 (excluding 4)
Default	3
Unit modes	AN
Non-volatile	Yes

Remarks

The following table describes the possible values and the modes associated with them.

UM Value	Description
1	Torque Control loop. The reference is set directly by **TC**. Values are processed immediately in the next control loop.
2	Speed Control loop. The reference is set by **JV**. The values are processed by the controller only after the next **BG**. Cyclic Synchronous velocity mode can also be used in this mode. Setting **TC** forces torque loop.
3	Stepper. No control loop beside the current. Use open loop electrical degrees given by **PA** for absolute movement, use **PR** for relative movement, and use **JV** for constant speed movement. The units are 512 ticks for 1 pole pair. **TC** must be set to excite the motor phases that allow the movement.
4	Reserved.

5	Position Loop, Single or Dual. **PA** and **PR** are used to reference the control loop. Cyclic Synchronous Position mode can be used in this mode as well. Setting **JV** will force velocity control loop and velocity profile reference. Setting **TC** will force torque control loop and the amount of torque. This method can be used for a welding application.
6	Stepper open loop. No control loop beside the current, uses open loop electrical degrees. Current during movement profile or standstill are according **HT[]** command and **FF[]** command. The units are 512 ticks for one pole pair.

References

US[] – User Saturation Parameters

US[] specifies the User Saturation Parameters for the controller output.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Range	0 to 100
Index range	1 to 4
Default	0
Unit modes	According to array index
Non-volatile	Yes

Remarks

The **US[]** command sets user saturation parameters to limit the controller output. for example: the current controller output is voltage, which is limited by **US[1]**.

Indices

The following table describes the available options for **US[]**.

Index	Description	Units	Values	Restrictions
1	Limit PWM (voltage) command max range is WS[54] when US[1]=100	% of max PWM range	0 to 100	All unit modes
2 to 3	Reserved			
4	Limit gantry's Yaw current	% of PL[1]	0 to 100	Only in gantry master

References

PL[], UM, WS[]

VB – Software Boot Version

VB returns the drive's boot version.

CANopen/CoE

Remarks

Each drive includes boot software that loads the FW after power-up and enables updating of the firmware version. The boot software version can be retrieved using the **VB** command.

The version format is as follows: DSP Boot <version number><date>;

For example: "DSP Boot 1.1.1.2 21Apr2010;"

Attributes

Attribute	Description
Type	String, read-only.
Source	RS232, USB, TCP, EoE
Restrictions	None
Range	None
Default	None
Unit modes	All
Non-volatile	No

References

VE – Velocity Error

VE gets the velocity tracking error.

CANopen/CoE

Attributes

Attribute	Description
Type	Status report, Integer
Source	Program, RS-232, CANopen
Restrictions	None
Range	N/A
Index range	N/A
Default	N/A
Unit modes	Position and velocity: **UM** = 2, 5
Non-volatile	N/A
Activation	

Remarks

VE reports the present velocity tracking error: **VE** = **DV[2]** – **VX**

If a dual loop is used, the units of the velocity error are in the position sensor and not in the velocity sensor.

If the absolute value of **VE** exceeds **ER[2]**, the motion is aborted, and the motion fault code **MF** = 128 (0x80) is set.

If **MO** = 0, or if the speed controller is not used (**UM** = 1, 3), **VE** returns 0.

References

PE, **VX**, **DV[*N*]**

VH[]/VL[] – High/Low Reference Limit

VH[] and **VL[]** specify the minimum and maximum speed and position reference limits.

CANopen/CoE

VH[2] – 0x607F

VL[3] – 0x607D.1

VH[3] – 0x607D.2

Attributes

Attribute	Description
Type	Integer32, Read/Write
Source	All
Restrictions	• The motor must be off. • **VH[3]** > **VL[3]** • **VH[3]** = **VL[3]** = 0 in the case of 32-bit modulo mode
Range	**VH[2]**: 0 to (2^{31} – 1) **VH[3]:** –2^{31} to (2^{31} – 1) **VL[3]:** –2^{31} to (2^{31} – 1)
Index range	**VH[*N*]**: N=2, 3 **VL[*N*]**: N=3
Default	**VH[2]** = 2000000000 **VH[3]** = 900000000 **VL[3]** = -900000000
Unit modes	**VH[2]**: **UM** = 2, 3, 5 **VL[3],VH[3]**: **UM** = 3 ,5
Non-volatile	Yes

Remarks

In position mode (**UM** = 5) motor movement is enabled in both directions within the defined position reference range [**VL[3]**…**VH[3]**]. Position commands outside of the range set by the **VL[3]** and **VH[3]**) values are not accepted.

If feedback has been extended beyond those limits, the motor can be enabled by the user (**MO** = 1), but the motion can only be in the direction toward the reference limit range.

Speed commands outside of the range (-**VH[2]**…**VH[2]**) are truncated.

The final velocity command limit is influenced by the following parameters: **VH[2]** (object 0x607F) and Max Motor Speed (object 0x6080). The logic of the velocity limit depends on the motor type:

- In the case of a rotary motor: Velocity Limit = min(**VH[2]**, object 0x6080)
- In the case of a linear motor: Velocity Limit = **VH[2]**

Indices

The following table describes the **VH[]/VL[]** entries.

Index	Description	Units	Range
0	Reserved		
1	Reserved		
2	The reference to the speed controller is limited to the range [-VH[2]…VH[2]].	User-defined	0 to $(2^{31} - 1)$
3	The reference to the position controller is limited to the range [VL[3]…VH[3]].	User-defined	-2^{31} to $(2^{31} - 1)$

Notes

- **VH[3]** and **VL[3]** should be set to 0 in 32-bit position modulo mode. In this mode all position limits are ignored.
- The **VH[]** and **VL[]** values should be given in user-defined position units specified by the **FC** command.

References

XM[*N*], HL[*N*], LL[*N*]

VO – Software OTP version

VO retrieves the drive's OTP (boot loader) version.

CANopen/CoE

Attributes

Attribute	Description
Type	String, read-only.
Source	RS232, USB, TCP, EoE
Restrictions	None
Range	None
Default	None
Unit modes	All
Non-volatile	No
Attribute	None

Remarks

Each drive includes an OTP version that is burned during the manufacturing process.

This command provides a way to retrieve the OTP version that resides in the drive.

The version format is as follows: DSP OTP <version number><date>;

For example: "DSP OTP 2.08 21Apr2010;"

References

VP – PAL Version

VP gets the PAL version.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer , read-only.
Source	RS232, USB, TCP, EoE
Restrictions	None
Range	0 to 255
Default	None
Unit modes	All
Non-volatile	No
Attribute	None

Remarks

This command returns the PAL version that is directly read from the PAL.

In cases where the PAL version cannot be supported by the drive firmware version, the motor enable (MO=1) will not operate, producing the error code 182.

Note:
In cases in which the version cannot be read (e.g., Gold Guitar drives), the value will be 0.

Reference

VR – Software Firmware Version

VR retrieves the drive's firmware version.

CANopen/CoE

Attributes

Attribute	Description
Type	String, read-only
Source	RS232, USB, TCP, EoE
Restrictions	None
Range	None
Default	None
Unit modes	All
Non-volatile	No
Attribute	None

Remarks

The **VR** command reports the firmware software version as a vstring.

The string includes:

1. The product name
2. The software version
3. The software release date
4. Hardware core type:
    d. None – SCORE hardware core
    e. G – GCON hardware core

The format is as follows: <Product name> <Software version> < Release date> <Core Type>

For example: “Whistle 01.01.04.34 04Oct2010G”

References

VU – Main Feedback Velocity in User-Defined Units

VU retrieves the velocity of the position sensor or velocity sensor, in user-defined units.

CANopen/CoE

0x606C

Attributes

Attribute	Description
Type	Float, Read-Only
Source	All
Restrictions	None
Range	–2e9 to 2e9
Default	0
Unit modes	All
Non-volatile	No

Remarks

In a velocity loop or a position loop with one sensor **VU** returns values in user-defined units.

In a dual loop, **VU** returns values in user-defined units of the position sensor. The sensor that returns is according to object 0x606A. If it is equal to zero, **VU** returns the velocity of the position sensor. If it is equal to 1, **VU** returns the velocity of the velocity sensor, but multiplied by a factor to be in the same units as the position sensor.

VX returns the same quantity as **VU**, but in counts per seconds.

References

VX, **FC[]**

VX – Main Feedback Velocity in Counts per Second

VX retrieves the velocity of the position sensor or velocity sensor, in counts per seconds.

CANopen/CoE

0x6069

Attributes

Attribute	Description
Type	Float, Read-Only
Source	All
Restrictions	None
Range	–2e9 to 2e9
Default	0
Unit modes	All
Non-volatile	No

Remarks

In a velocity loop or a position loop with one sensor **VX** returns values in counts/sec.

In a dual loop, **VX** returns values in counts/sec in the units of the position sensor. The sensor that returns is according to object 0x606A. If it is equal to zero, **VX** returns the position sensor. If it is equal to 1, **VX** returns the velocity of the velocity sensor, but multiplied by a factor to be in the same units as the position sensor.

VU returns the same quantity as **VX**, but in user units.

References

VU, **FC[]**, **XM[]**

WI[] – Wizard Integer Parameters

WI[] gets internal information.

CANopen/CoE

Attributes

Attribute	Description
Type	See the table below.
Source	All
Restrictions	None
Range	See the table below.
Index range	According to the table below
Default	N/A
Unit modes	All
Non-volatile	No

Remarks

Indices

The following table describes the **WI[*N*]** entries.

Index	Description	Type	Values
16	Returns the state of the voltage experiment	Integer	0, 1
20	Returns the maximum allowed recorder length	Integer	16384
21	Returns the actual recorder length, depending on the number of selected signals	Integer	1 to 16384
35	Returns the overvoltage maximum threshold	Integer	0 to <**BV**
36	Returns the actual overvoltage threshold	Integer	0 to ≤**WI[35]**
37	Returns the undervoltage minimum threshold	Integer	≥0
38	Returns the actual undervoltage threshold	Integer	≥**WI[37]**

* All other indices are reserved.

References

WS – Miscellaneous Reports

The **WS** command provides information about the state and internal variables of the drive. Mainly used for internal use only.

CANopen/CoE

Attributes

Attribute	Description
Type	Floats, Integers, Read only
Source	All
Restrictions	
Range	
Index range	Refer to the table below
Default	
Unit modes	All
Non-volatile	No

Remarks

WS[*N*] provides service personnel with a fairly comprehensive report, but these details are not normally required for defining an application.

Indices

The following table describes the **WS** entries.

Index	Description	Type	Units\Values	Notes
3	CPU main clock frequency. Default value: 150000000. Refer to the product manual.	Integer	Hz	
4	PWM frame Default=150x**TS/XP[2]**	Integer	150 MHz clock	
5	Bits per ampere (MaxADCvalue/MC)	Float		

Index	Description	Type	Units\Values	Notes
8	HW board type	Integer	0: SCORE board 1:GCON revision A 2: GCON revision C	SCORE: GGUI GCON: GWHI, GDRU, GTRO
9	ADC instant delay. Units in 150 MHz clock.	Integer	150 MHz clock	For internal use only.
12	Dead Band delay. Units in 150 MHz clock.	Integer	150 MHz clock	For internal use only.
16	Planar: bit 0 – commutation known bit 1 – Amplitude at motor off over CA[52]	Integer		For planar only Also in object 0x2085
17	Correlation in current experiment $\Sigma(0.25*fTorqueCmdOld^2)$	Float		For internal use only.
18	VDC average Σ(stStateVec.SAMPLE_VBUS)	Integer		For internal use only.
19	ERR_AVG stWizCor.fErrorAvg	Float		For internal use only.
22	1/Torque scale	Float		
23	Factor to convert user A/D bits to volts: basically, 10.0/(2048.0 - 205.0)	Float		
24	Returns the internal offset of Analog Input 1.	Float	Volts	
28	Returns the velocity loop recording cycle time.	Integer	microseconds	For compatibility
29	Returns the position loop recording cycle time.	Integer	Default: 2 * **TS**	For compatibility

Index	Description	Type	Units\Values	Notes
30	Product information **Bits 0 to 7** Product Name **Bits 8 to 11** Reserved for product name **Bits 12 to 13** Always 0 **Bit 14** Project (always 1 for Gold Line) **Bit 15** Always 0 **Bit 16** CAN integrated **Bit 17** 0: EtherCAT 1: TCP\IP **Bits 18 to 20** Feedback type: 0: E type (Encoder + Encoder, Analog sensor) 1: A type (Absolute + Encoder, Analog sensor) 2: R type (Encoder, Analog sensor + Resolver) 3: M type (Absolute + Resolver) **Bit 21** Define R type drive: Current saturation stays on **PL** **Bit 22** EtherCAT ID switches **Bits 23** Reserved **Bit 24** Ethernet hardware: 0: Absent	Integer		

Index	Description	Type	Units\Values	Notes
	1: Exist **Bits 25 to 31** Reserved			
31	Gantry cycle time	Integer	microseconds	
33	The instantaneous saturated high torque value	Float	Amperes	Nominal **PL[1]**
34	The instantaneous saturated low torque value.	Float	Amperes	Nominal **CL[1]**
35	Gantry differential position	Integer	Counts	
36	ADC value at nominal Bus voltage	Integer	ADC	
50	Sum of Reference Signal used in tuner Σ(fTorqueCmdOld)	Float	Internal torque units	For internal use only
51	Wizard state counter used in tuner	Integer		For internal use only
52	Minimum Bus Voltage for schedule current controller	Integer	Internal A2D units	For internal use only
53	Convert values from internal units to bus voltage	Float		
54	Maximum PWM (voltage) command allowed in drive	Integer	150 MHz clock	
55	Position Loop cycle time	Integer	microseconds	
56	Minimum PWM command allowed in drive	Integer	150 MHz clock	
57	PWM command range (**WS[54]**-**WS[56]**)	Integer	150 MHz clock	
72	Angle of Sin/Cos encoder	Integer		For internal use only

Index	Description	Type	Units\Values	Notes
75	Mode of sine virtual sensor: 0 – stopped 1 – ramp 2 – start the two sine signals 3 – wait for high frequency to stop 4 – wait for low frequency to stop 5 – ramp down	Integer		For internal use only
91	Signal #1 sine coefficient	Float		Provide Valid info after **TW[81]**=N. N is a bit field that determines the vectors to identify. For internal use only
92	Signal #1 cosine coefficient	Float		
93	Signal #2 sine coefficient	Float		
94	Signal #2 cosine coefficient	Float		
95	Signal #3 sine coefficient	Float		
96	Signal #3 cosine coefficient	Float		
97	Signal #4 sine coefficient	Float		
98	Signal #4 cosine coefficient	Float		
99	Signal #1 quality function	Float		
100	Signal #2 quality function	Float		
101	Signal #3 quality function	Float		
102	Signal #4 quality function	Float		
103	Offset of signal #1	Float		
104	Offset of signal #2	Float		
105	Offset of signal #3	Float		
106	Offset of signal #4	Float		
111	Temperature ADC Out	Integer	ADC digital out	For internal use only

References

TW[*N*], WI[*N*]

XA[] – Extra (Current Loop) Parameters

XA[] specifies some extra and special parameters, which are used in the current control loop.

CANopen/CoE

Attributes

Attribute	Parameter, Integer
Type	Integer, Read/Write
Source	All
Restrictions	**MO** = 0
Range	See below.
Index range	1 to 5
Default	See below.
Unit modes	All
Non-volatile	Yes

Remarks

Do not modify the **XA[]** values. These values are automatically programmed into the drive during current loop tuning.

Indices

The following table describes the **XA[*N*]** entries.

Index	Description	Default	Values	Restrictions
0	Reserved			
1	Reserved			
2	Reserved			
3	Reserved			
4	Cancel stop manager features: bit 0: bypass position software limits bit 1: bypass acceleration limiting bit 2: bypass switch handling	0	0 to 7	

<table>
<tr><th>Index</th><th colspan="2">Description</th><th>Default</th><th>Values</th><th>Restrictions</th></tr>
<tr><td rowspan="3">5</td><td colspan="2">Enable/disable cogging compensation:</td><td rowspan="3">0</td><td rowspan="3">0, 1</td><td rowspan="3">If cogging compensation is enabled, NT[N] should be filled (by EAS).</td></tr>
<tr><td>0</td><td>Disable cogging compensation</td></tr>
<tr><td>1</td><td>Enable cogging compensation with a value in the NT[N] array</td></tr>
</table>

References

NT[]

XC – Resume Program

XC resumes a halted user program.

CANopen/CoE

Attributes

Attribute	Description
Type	Command
Source	All, except the user program
Restrictions	**PS** ≥ 0
Range	None
Default	None
Unit modes	All
Non- Volatile	None

Remarks

While the user program is running, the user can halt the program temporarily, using the **HP** command.

The **XC** command is used to continue running the user program from the point at which it was halted.

The **XC** command cannot release the program from a breakpoint. For that purpose, use the **DB##GO[0]** command.

References

HP, **XQ**, **KL**

XM[] – Position Modulo

XM[] specifies the counting range for the main feedback, which is [**XM[1]**...**XM[2]** - 1].

The main feedback can be retrieved by object 0x6064 or the **PU** command.

CANopen/CoE

XM[1] – 0x607B.1

XM[2] – 0x607B.2

Attributes

Attribute	Description
Type	Integer 32, Read/Write
Source	All
Restrictions	• The motor must be off. • Homing must not be active (**HM[1]** = 0).
Range	-2^{31} to $(2^{31} - 1)$
Index range	1, 2
Default	**XM[1]** = −1000000000 **XM[2]** = 1000000000
Unit modes	All
Non-volatile	Yes

Remarks

A profiler block can be used for generating:

- **Non-modulo motion**. The motion is SW limited to within a finite position range (object 0x607D or **VH[3], VL[3]**). The non-modulo motion is enabled, if one of the following conditions holds:
  - **XM[2]** = **XM[1]** = 0;
  - **XM[1]** <= **VL[3]** and **XM[2]** >= **VH[3]** .
- **Modulo motion**. The position counts in a position range limit (object 0x607B or **XM[1]**, **XM[2]**). The modulo motion is enabled, if
  - **XM[2]** > **XM[1]** and **XM[1]** > **VL[3]** and **XM[2]** < **VH[3]**.
- **32-Bit modulo motion**. If **XM[2] = XM[1] = VH[3] = VL[3] = 0**, neither the software position limit nor the position range limit affects the motion.

In the modulo mode, the feedback (object 0x6064 or **PU**) is always counted cyclically. This means that after the position is counted to its maximum value, the next position count will reset the position counter back to its minimum value.

The speed reading is not affected by the position jump.

Example

If **XM[1]** = -5 and **XM[2]** = 5, the main position is counted in a cycle with a length equal to **XM[2] - XM[1]** = 10.

The main position will always be in the range [-5…4]. If the main feedback rotates in the positive direction, the main position count will proceed from 0, 1, 2, 3, 4 to -5, -4, -3, -2, -1, 0, 1 . . . and so on.

Optional modulo movements

The following modulo modes are optional. In all modes positioning can be relative or absolute.

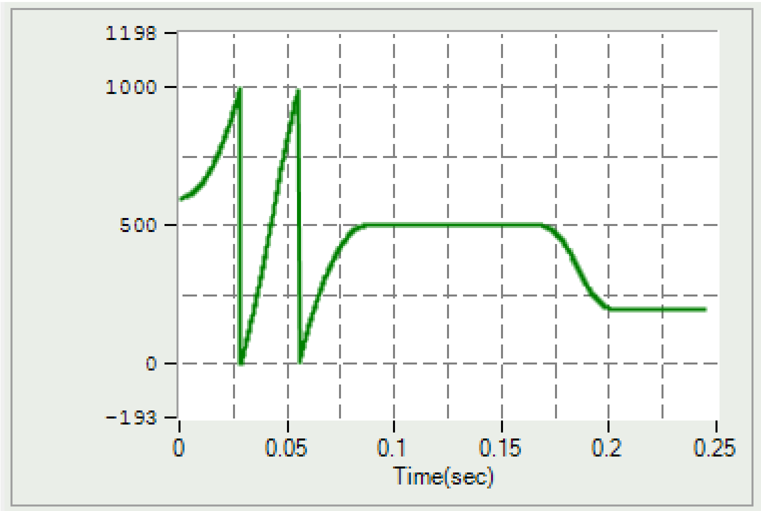
Using object 0x60F2 bits 7:6 (or **PO** bits 7:6):

Mode	Bits 7:6 Value	Remarks
Normal positioning	0x0 (default)	Normal rotary positioning is similar to linear axis positioning. If the position range limits (object 0x607B or **XM[1]**, **XM[2]**) are reached or exceeded, the input value wraps automatically to the other end of the range. Movement greater than a modulo is possible only with this bit combination.
Negative movement	0x1	Positioning only in the negative direction. If the target position is higher than the actual position, the axis moves over the minimum position range limit (**XM[1]** or object 0x607B.1) to the target position.
Positive movement	0x2	Positioning only in the positive direction. If the target position is lower than the actual position, the axis moves over the maximum position range limit (**XM[2]** or object 0x607B.2) to the target position.
Positioning with shortest path	0x3	Positioning with the shortest path to the target position. **Note.** If the difference between actual value and target position in a 360° system is 180°, the axis moves in positive direction.

Example of normal modulo positioning

The profile generator does not remember the modulo rolling that the position demand value passed. Every new movement is based on the location of the current demand value within the modulo range.

In the figure below, **XM[1]** = 0, **XM[2]** = 1000, the initial position is **PX**=600, and two absolute motions are applied: **PA** = 2500 and, then, **PA** = 200. In the first motion, the PTP trajectory travels through 600…999, 0, 1…999, 0, 1…499, 500 over a total distance of 1900 counts. In the second motion, it travels through 500, 499…201, 200 over a total distance of (-300) counts.



Example of positioning with shortest path:

If **XM[1]** = -512, **XM[2]** = 512, the initial position is **PX** = -500, and the target absolute position is **PA** = 500, the PTP trajectory will travel through -500…-512, 511…500 over a total distance of 23 counts.

Default modulo behavior versus Elmo legacy modulo behavior

The drive default modulo, as well as the DS402 default modulo, is the *normal positioning* mode.

In order to maintain the Elmo legacy mode, the modulo mode needs to be set to *positioning with shortest path* by a **PO** command.

In the case where object 0x607A (or **OV[32]**) is set using the control word (object 0x6040), the setting of object 0x60F2 (**OV[28]**) will be used.

Note that by default the drive is not in a modulo state, because the Software Position limit (object 0x607D or **VH[3]**, **VL[3]**) is lower than the Range limit (object 0x607B or **XM[1]**, **XM[2]**).

32-Bit modulo motion

If **XM[2]** = **XM[1]** = **VH[3]** = **VL[3]** = 0, neither the Software Position limit nor the Range limit affects the motion. Modulo mode should be set to normal positioning.

This mode is designed in order to support an interface to a 64-bit master. On the master side, there could be two possible approaches for supporting this mode:

- The position command passes a 32-bit modulo (0…2^{32} - 1);
- The position command is non-modulo, but the drive receives only the lower 32 bits of the position target.

The master command is subject to the following restriction: the difference between two consecutive 64-bit position targets must not exceed 2^{31} - 1.

Example

In order to move 6000000000 counts, three consecutive movements should be requested:

```
0x607A (or OV[32]) = 0x77359400;   // set abs. target 2000000000
0x607A (or OV[32]) = 0xee6b2800;   // set abs. target 4000000000
0x607A (or OV[32]) = 0x65a0bc00;   // set abs. target
                                   // 6000000000=0x165a0bc00
```

Indices

The following table describes the **XM[]** entries.

Index	Description	Type	Values
1	Minimum position range limit	User-defined	-2^{31} to $(2^{31} - 1)$
2	Maximum position range limit	User-defined	-2^{31} to $(2^{31} - 1)$

Notes

- If **XM[1]** or **XM[2]** is set so that the main feedback (**PU**) is outside of the range [**XM[1]**…**XM[2]**], **PU** is be set to the range by taking the modulo:

  PU = (**PU** - **XM[1]**) mod (**XM[2]** - **XM[1]**) + XM[1], if **PU** > **XM[2]**, or

  PU = -(-**PU** + **XM[2]**) mod (**XM[2]** - **XM[1]**) + **XM[2]**, if **PU** < **XM[1]** .

  This is done at power-up and at motor on (**MO** = 1).

- A new **XM[1]**/**XM[2]** setting is activated after the setting of **XM[2]**, at power-up or upon issuing the **MO**=1 command.
- If **XM[2]** = **XM[1]** = 0, the position modulo functionality is disabled. **VH[3]** and **VL[3]** are used for positioning limiting.
- If the **XM[*N*]** value is selected low and the main speed is too high, more than one full revolution of the main counter may occur within a single sampling time. This will cause the main position counter to behave unpredictably. This happens when the modulo cycle completes in less than 250 µsec. The drive will not indicate this as a failure, and it is up to the user to take care of this situation.
- The **XM[*N*]** values should be given in user-defined position units specified by the **FC** command.

References

VH[]/VL[], **PO**, **FC**

XP[] – Extra General Parameters

XP[] specifies extra general parameters for adapting the drive to special situations.

CANopen/CoE

Attributes

Attribute	Description
Type	Integer, Read/Write
Source	All
Restrictions	The motor must be off.
Range	See the table below.
Index range	1 to 13
Default	See the table below.
Unit modes	All
Non-volatile	Yes

Remarks

Indices

The following table details the **XP[]** entries.

Index	Description	Default	Values	Notes
0	Reserved			
1	Defines the overvoltage threshold in volts. Can only be reduced from the default value (**BV**)	**WI[35]**	0 to **WI[35]**	

<table>
<tr><td rowspan="7">2</td><td colspan="2">Defines the PWM frequency as a factor of the controller sampling time:
TS/TPWM = TS*fpwm = XP[2]/2
The current ripple frequency at load is equal to XP[2]/TS.</td><td rowspan="7">2</td><td rowspan="7">1 to 6</td><td rowspan="7">Could heat the drive.
Some values are blocked by the drive.</td></tr>
<tr><td>1</td><td>The current controller sampling time is at the top and bottom of the PWM triangle.</td></tr>
<tr><td>2</td><td>The current controller sampling time is at the top of the PWM triangle.</td></tr>
<tr><td>3</td><td>The current controller sampling time is at the end of every 1.5 cycles of PWM.</td></tr>
<tr><td>4</td><td>The current controller sampling time is at every other top of the PWM triangle.</td></tr>
<tr><td>5</td><td>The current controller sampling time is at the end of every 2.5 cycles of PWM.</td></tr>
<tr><td>6</td><td>Current controller sampling time is at every third of top of PWM triangle.</td></tr>
<tr><td>3</td><td colspan="2">Reserved</td><td></td><td></td><td></td></tr>
<tr><td>4</td><td colspan="2">Filter constant of bus voltage measurements in Hz.</td><td>1500</td><td>100 to 3,000</td><td></td></tr>
<tr><td>5</td><td colspan="2">Maximum current command rate in one cycle in percent of MC.</td><td>20</td><td>10 to 200</td><td></td></tr>
<tr><td>6</td><td colspan="2">Low-pass constant for pre-filter of current loop in Hz.</td><td>3000</td><td>0 to 20000</td><td>If it equals zero, the filter is bypassed.</td></tr>
<tr><td>7 to 8</td><td colspan="2">Reserved</td><td></td><td></td><td></td></tr>
<tr><td rowspan="4">9</td><td colspan="2">Sync PWM to other drives</td><td rowspan="4">0</td><td rowspan="4">0 to 2</td><td rowspan="4"></td></tr>
<tr><td>0</td><td>No sync</td></tr>
<tr><td>1</td><td>Sync as master</td></tr>
<tr><td>2</td><td>Sync as slave</td></tr>
</table>

10 to 12	Reserved			
13	Defines the under voltage	**WI[38]**		

Notes

- The current loop must be tuned after a modification of **XP[2]**.
- When **XP[2]** is used to multiply the PWM frequency from the default, the current saturation (**CL[1]** and **PL[1]**) might be reduced. The actual values of **PL[1]** and **CL[1]** are reported in **WS[33]** and **WS[34]**, respectively.

References

PL[], **CL[]**, **BV**, **TS**

XQ – Execute User Program

XQ executes the user program from a specified label or runs a specified function.

CANopen/CoE

Attributes

Attribute	Description
Type	Command , String
Source	All, except the user program
Restrictions	**PX** = –1 **CC** was executed correctly. Wizard mode is not active.
Range	`XQ##func_name` For example, if function to be run is **main**, the syntax is: `XQ##main` (or `xq##main`)
Default	None
Unit modes	All
Non-volatile	None

Remarks

XQ## executes a valid user program.

This command is typically sent after sending a successful **CC** command.

The general format is:

```
XQ##[function name]
```

Examples

- **XQ##** runs from the start of the user program code.
- **XQ##MyFunction(a,b,c)** runs the function **MyFunction ()** with a,b and c as arguments to the function.
- XQ##LABEL runs from ##LABEL.

The **XQ** command clears the error status of the program along with the run-time error flags.

It does not reset program variables and does not clear the interrupt mask.

Notes

- **XQ** must include **##**. If it is omitted, an error is returned.
- **XQ##**, beside acknowledge, does not return a value.

References

CC, **CP**, **HP**, **KL**, **PS**, **XC**

YM[] – External Reference Modulo

YM[] specifies the counting range for the external reference, which is **[YM[1]…YM[2] - 1]**.

The external reference can be retrieved by object 0x20A0 or the **PY** command.

CANopen/CoE

YM[1] – 0x207B.1

YM[2] – 0x207B.2

Attributes

Attribute	Description
Type	Integer 32, Read/Write
Source	All
Restrictions	• The motor must be off.
Range	-2^{31} to $(2^{31} - 1)$
Index range	1, 2
Default	**YM[1]** = −1000000000 **YM[2]** = 1000000000
Unit modes	All
Non-volatile	Yes

Remarks

- The position of the external reference is counted cyclically, and therefore after the position is counted to its maximum value, the next position count will reset the position counter back to its minimum value. The speed reading is not affected by the position jump. For example: If **YM[1]**=-5 and **YM[2]**=5, the external reference is counted in a cycle length of 10. The external reference will always be in the range [-5…4]. If the external reference rotates in the positive direction, the external reference count will proceed from 0, 1, 2, 3, 4 to -5, -4, -3, -2, -1, 0, 1….. and so on.
- A new **YM[1]**/**YM[2]** setting is activated after the setting of **YM[2]**, at power-up or upon issuing **MO**=1 command.
- **YM[2]** must be bigger than **YM[1]** (**YM[2]**>**YM[1]**).
- If **YM[2]** = **YM[1]** = 0, the external reference modulo functionality is disabled. Actually, this means that **PY** counts in the Integer32 counting range (-2^{31} to $(2^{31} - 1)$).

- If **YM[1]** or **YM[2]** is set so that **YM[2]** > **YM[1]**, but **PY** is out of the range [**YM[1]**…**YM[2]**], **PY** will be set to range by taking the modulo: **PY** = (**PY** - **YM[1]**) mod (**YM[2]** - **YM[1]**) + **YM[1]**
- If the **PY** value setting is out of the range [**YM[1]**…**YM[2]**] which is requested, the request will be ignored and **PY** will not change
- If the **YM[N]** value is selected low and the external reference speed is too high, more than one full revolution of the counter may elapse within a single sampling time. This will cause the **PY** counter to behave unpredictably.

References

PY