DSC2P 903 ver. F

MOTION TECHNOLOGY

Digital position controllers

DSCxx

Operation & Software Manual

HEADQUARTERS
SWITZERLAND
ETEL S.A.
Zone industrielle
CH-2112 Môtiers
Phone : +41 (0)32 862 01 23
Fax : +41 (0)32 862 01 01
E-mail : etel@etel.ch
http://www.etel.ch

AMERICAS
ETEL Inc.
333 E. State Parkway,
US - Schaumburg
IL 60173-5337
Phone : +1 847 519 3380
Fax : +1 847 490 0151
E-mail : info@etelusa.com

FRANCE
ETEL S.A.
89 rue Henri Rochefort
FR-91025 Evry cedex
Phone : +33 (0)1 64 97 98 05
Fax : +33 (0)1 64 97 30 56
E-mail : etel@etel.fr

GERMANY
ETEL GmbH
Schillgasse 14
DE-78661 Dietingen
Phone : +49 (0)741 17453-0
Fax : +49 (0)741 17453-99
E-mail : etel@etelgmbh.de

ITALY
ETEL S.A.
Piazza della Repubblica 11
IT-28050 Pombia
Phone : +39 0321 958 965
Fax : +39 0321 957 651
E-mail : etel@etelsa.it

SWITZERLAND
ETEL S.A.
Zone industrielle
CH-2112 Môtiers
Phone : +41 (0)32 862 01 33
Fax : +41 (0)32 862 04 12
E-mail : sales@etelsa.ch

THE NETHERLANDS
ETEL B.V.
Copernicuslaan 34
NL-6716 BM Ede
Phone : +31 (0)318 495 200
Fax : +31 (0)318 495 210
E-mail : etel@etelbv.nl

AUSTRIA (Sales rep)
I+L ELEKTRONIK GmbH
Vibrütteweg 9
AT-6840 Gotzis
Phone : +43 55 23 645 42
Fax : +43 55 23 645 424
E-mail : IUL@uta1002.at

ISRAEL (Sales rep)
MEDITAL COMOTECH Ltd
7 Leshem St. - Ramat Siv
IL-49170 Petach tikva
Phone : +972 3 923 3323
Fax : +972 3 923 1666
E-mail : comotech@medital.co.il

THIS PAGE IS INTENTIONALLY LEFT BLANK

Chapter A:
Internal functioning & architecture

Chapter B: System setup and tuning

Chapter C: System functions

Chapter D: Programming

Chapter E: Appendixes

THIS PAGE IS INTENTIONALLY LEFT BLANK

Introduction

This document concerns the following ETEL digital position controllers: **the DSC2P, DSC2V, DSCDP, DSCDL and DSCDM** also called 'position controller' or simply 'controller'.
The purpose of this manual is to give details regarding the system's functioning, installation, tuning, functions and programming possibilities. For electrical specifications, interfaces and hardware items, please refer to the corresponding **'Hardware Manual'**.

The information given in this manual is valid for:
- **the DSC2P and DSC2V with a firmware from version 1.22 (firmware identical for both products)**
- **the DSCDP with a firmware from version 1.15**
- **the DSCDL with a firmware from version 1.06**
- **the DSCDM with a firmware from version 1.02**

If a DSMAX motion controller is used, its firmware must be from version 1.20A for the DSMAX1 and DSMAX2 and from version 1.10A for the DSMAX3.
If a DSTEB motion controller is used, the DLL used must be from version EDI2.13A (usable with ETT4.11A or above).

All the functions described in this manual have not been implemented in all the controllers. This is why a table is present at the beginning of all the paragraphs describing a function to indicate in which product(s) this function is present.
Example:

DSC2P	Function present in the DSC2P

~~DSC2P~~	Function not present in the DSC2P

Remark: If **a part of** a function is not present in a controller, the name of the corresponding controller will be crossed out in the comment of the function (e.g: **~~DSCDP~~**).

Remark: The updates between two successive versions are highlighted with a modification stroke in the margin of the manual.

General operating conditions

The controllers are designed to operate in a non-aggressive and clean environment, with a humidity rate ranging between 10% and 85%, an altitude < 2000m (6562 ft), and a temperature ranging between +15°C (59°F) and +30°C (86°F) or +40°C (104°F) depending on the product (refer to the corresponding **'Hardware Manual'** for more information). They must be connected to an electrical network of overvoltage category 2 (refer to EN 50178 and UL 840 standards for more information). The DSC2P, DSCDP and DSC2V are suitable for use on a circuit capable of delivering not more than 5000 Arms, symmetrical amperes, 400 volts maximum. The DSCDM must be connected to a power supply with SELV outputs. The electronics must be in an enclosure respecting a pollution degree of 2 (refer to UL508C and EN 50178 standards for more information).

The controllers are not designed or intended for use in the on-line control of air traffic, aircraft navigation and communications as well as critical components in life support systems or in the design, construction, operation and maintenance of any nuclear facility.

Safety

Please, read all the safety precautions listed in this manual before handling the controller:

Warning: Signals a danger of electrical shock to the operator.
Can be fatal for a person.

Caution: Signals a danger for the controller. Can be destructive for the material.
A **danger** for the operator can result from this.

Caution: Indicates electrostatic discharges (ESD), dangerous for the controller. The components must be handled in an ESD protected environment only.

- Before installing or operating the controller, all the corresponding documentations listed page 15 as well as the one related to the motor(s) used with it.
- Never use the controller for purposes other than those described in this manual.
- A competent and trained technician must install and operate the controller, in accordance with all specific regulations of the respective country concerning both safety and EMC aspects.
- Troubleshooting and servicing are permitted only for ETEL's technicians and agreed distributors.
- Operating the controller will make the motor move. **Keep away from all moving parts to avoid injuries!**
- High voltage may be present on the power and motor connectors.
- Before connecting or disconnecting a cable on one of these connectors or touching the controller, **turn off all the power supplies and wait 10 minutes** (**2min** for the DSCDM) to allow the internal DC bus capacitors to discharge.
- All the connector must be handled in an ESD protected environment, only.
- The safety symbols placed on the controller or written in the manuals must be respected.
- In the controller, the **leakage current** through the protective conductor to the GND is greater than a.c. 3.5 mA.

If the controller is integrated into a machine, the manufacturer of this machine must establish that it fulfils the 89/336/EEC directive on EMC before operating the controller.

How to use this manual

If you are not an experienced user, read first the **Chapter A** to catch the **basis** about the **controller's internal functioning** and commands' syntax.

Then, follow step-by-step the **Chapter B** to successfully realize the **first installation and setup** of the controller.

In **Chapter C,** §11. and §12., all the **basic functions** necessary to operate the controller are described in details. **Chapter C**, §13. is reserved for experienced users and describes **advanced functions** only used in specific applications.

Chapter D describes how to **program** the controller (**movements** sequences).

The appendixes in **Chapter E** include the registers and commands **references lists** (with commands **examples**), as well as **error and warning messages** lists and units conversion formulas.

Remark: **ETEL can provide its customers with training courses including theoretical presentation and practice in real conditions, at our facilities in Môtiers (Switzerland).**

Working principle

These digital position controllers have been designed for direct drive applications. They can work in interpolated mode if they are mastered by a DSMAX motion controller (refer to the **'DSMAX User's Manual'**). They can drive single-phase, two-phase and three-phase motors (two motors for the DSCDP, DSCDL and DSCDM). You can obtain brushless torque and linear motors from ETEL as well as moving coils and moving magnets. They can also drive brushless motors, DC motors, steppers (only if three-phase motors are star-connected). They must be implemented with analog (incremental or absolute (EnDat2.1)) or TTL encoders available on the market.

Record of revisions, document # DSC2P 903 x

Documents revisions		
Issue (x)	Date	Modified
A	30.05.01	Operation & Software DSC2P, PRELIMINARY EDITION for firmware ver. 1.xx
B	06.11.01	Updated version with firmware from version 1.03
C	06.06.02	Updated version with firmware from version 1.10 for DSC2/6-P and with firmware from version 1.0 for DSCDP
D	03.03.03	Updated version with firmware from version 1.14 for DSC2P and with firmware from version 1.04 for DSCDP
E	12.11.03	Updated Version with firmware from version 1.17 for DSC2P and with firmware from version 1.07 for DSCDP - Extended functions for K141 parameter (refer to §12.4.4) - Dynamic braking (refer to §12.5.5) - Extended functions for K32 parameter and new parameter K100 (refer to §12.9) - New homing modes: K40 = 34 to 39 (refer to §12.9.2) - New parameters and monitorings managing the analog I/O of the DSO-HIO optional board (refer to §13.6) - Real-time monitorings (refer to §13.12.2) - New mapping mode: the rotary mapping (refer to §13.13.2)
F	03.06.05	Updated Version with firmware from version 1.22 for DSC2P/DSC2V, 1.15 for DSCDP, 1.06 for DSCDL and 1.02 for DSCDM. - New interpolation mode: ITP = 2 (refer to §13.2.1.3) - New small movement phasing mode: K90 = 6 (refer to §12.7.2.4) - Stepper mode in open loop (refer to §12.13) - New monitorings: M110 (refer to §13.2.1.3), M239 and M241 (refer to §12.3), M240 (refer to §12.2) and M145, M146, M147, M148 and M149 (refer to §12.3.2) - Extended possibilities with the AUT command (refer to §12.8.1) - New command: INI (refer to §12.7.1.2) - Extended functions for K32 parameter (refer to §12.9) - Extended description of command WTW (refer to §14.1.1.3) - Digital Hall effect sensor available for DSCDP and DSCDM (refer to §12.7.2.3) - Increase of the triggers number and actions (refer to §13.9)

Documentation concerning the position controllers:

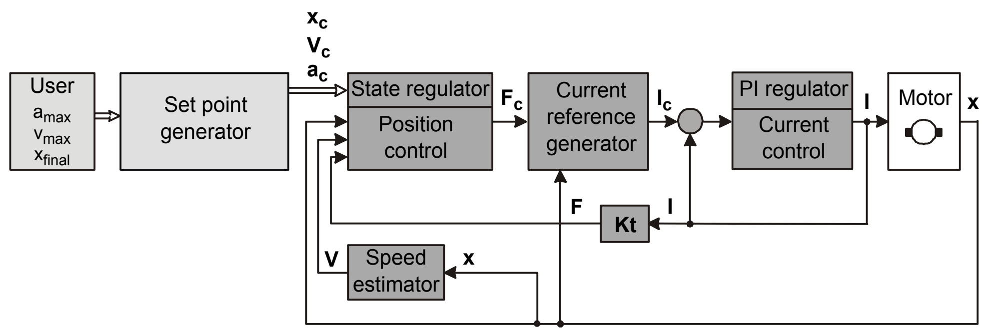
• **Operation & Software Manual**	**(Controller's setup, use & programming)**	**# DSC2P 903 F**
• DSC2P Hardware Manual	(Specifications & electrical interfaces)	# DSC2P 904 x
• DSC2V Hardware Manual	(Specifications & electrical interfaces)	# DSC2V 904 x
• DSCDP Hardware Manual	(Specifications & electrical interfaces)	# DSCDP 904 x
• DSCDL Hardware Manual	(Specifications & electrical interfaces)	# DSCDL 904 x
• DSCDM Hardware Manual	(Specifications & electrical interfaces)	# DSCDM 904 x
• EBL2 communication Manual	(EBL2 principal, messages mapping)	# EBL2 908 x
• DSO-PWS User's Manual	(Power module installation and specifications)	# DSOPWS 902 x
• DSO-RAC2 Hardware Manual	(DSO-RAC2 principal)	# DSORAC2 904 x

THIS PAGE IS INTENTIONALLY LEFT BLANK

Chapter A:
Internal functioning & architecture

1. The regulators

Two Digital Signal Processors (DSP, Sharc from Analog Devices) manage the controller. The first manages the movements profiles and the second is used for the regulation loops. There are two regulation loops, a **position loop** and a **current loop**, controlled by the second DSP. The position loop calculates the **reference force F_c** that the motor supplies to follow the position reference calculated in a separated generator, called **set point generator**. This force is afterwards sent to the current loop where the **current reference generator** calculates the reference currents I_{cx} for each motor phase. Those references are eventually sent to a PI (proportional-integral) regulator which controls the current in the motor phase(s). The diagram below shows the complete regulation process.



x	:	Motor real position	x_c	:	Position reference
V	:	Motor speed	V_c	:	Speed reference
F	:	Force supplied by the motor	F_c	:	Force reference
I	:	Current in the motor	I_c	:	Current reference in the motor
Kt	:	Motor force constant	a_c	:	Acceleration reference

Movement manager processor

Regulation (current & position) processor

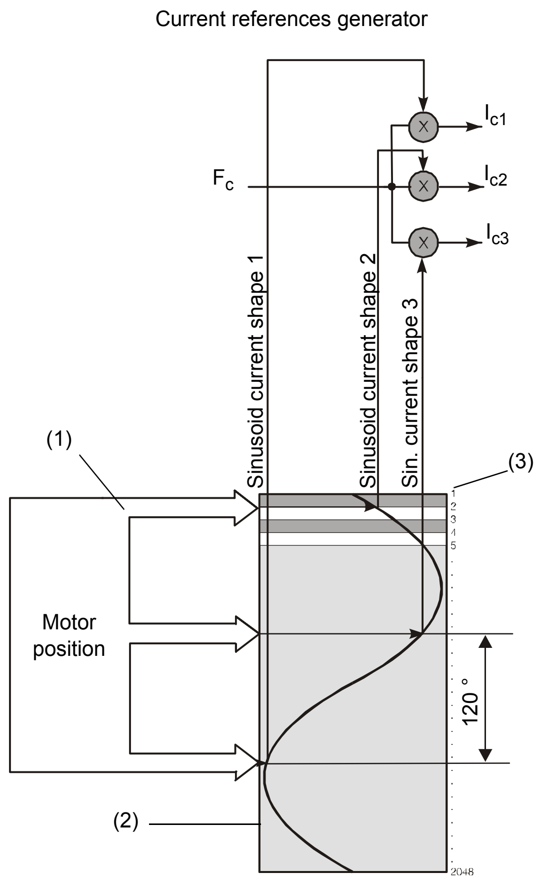
The elements of this regulation general diagram are detailed in §9.1(for beginners) and in §13.1(for advanced users).

2. Current references generator

The motor has to deliver a force $F = F_c$ independently of its position with respect to the magnets poles. All phases must then be fed with sinusoidal type currents, in phase with the magnetic field. For a three-phase motor, 3 sinusoidal currents must have a 120^o phase-shift (for a two-phase motor, it has to be a 90^o phase-shift). Three phases motor will be considered, as it is the most commonly used. The **current reference generator** multiplies first the force reference F_c by the motor position on the sinusoid, making out reference currents:
$\mathbf{I_{c1}}= \sin(X + 0^o)$ in phase 1, $\mathbf{I_{c2}}= \sin(X + 120^o)$ in phase 2 and $\mathbf{I_{c3}}= \sin(X + 240^o)$ in phase 3.

The motor currents calculation is as follows:

Three pointers (1), with a 120° electrical phase-shift, point at in a table, according to the motor position. This table, called **commutation look-up table** (2), contains 2048 points (3) forming a sinusoidal function period. Motor position sine-forms are thus immediately read on the numbers of the table. The force F_c is then multiplied by each of the two values giving I_{c1}, I_{c2} and I_{c3}.



Remark: When one of the pointers reaches the end of the table, it goes on from the other end.

Pointing at the right places in the table when powering on the motor is important, because its position with respect to the magnets is not known at the beginning. The **initialization** procedure allows the user to know the initial position.

3. Set point generator

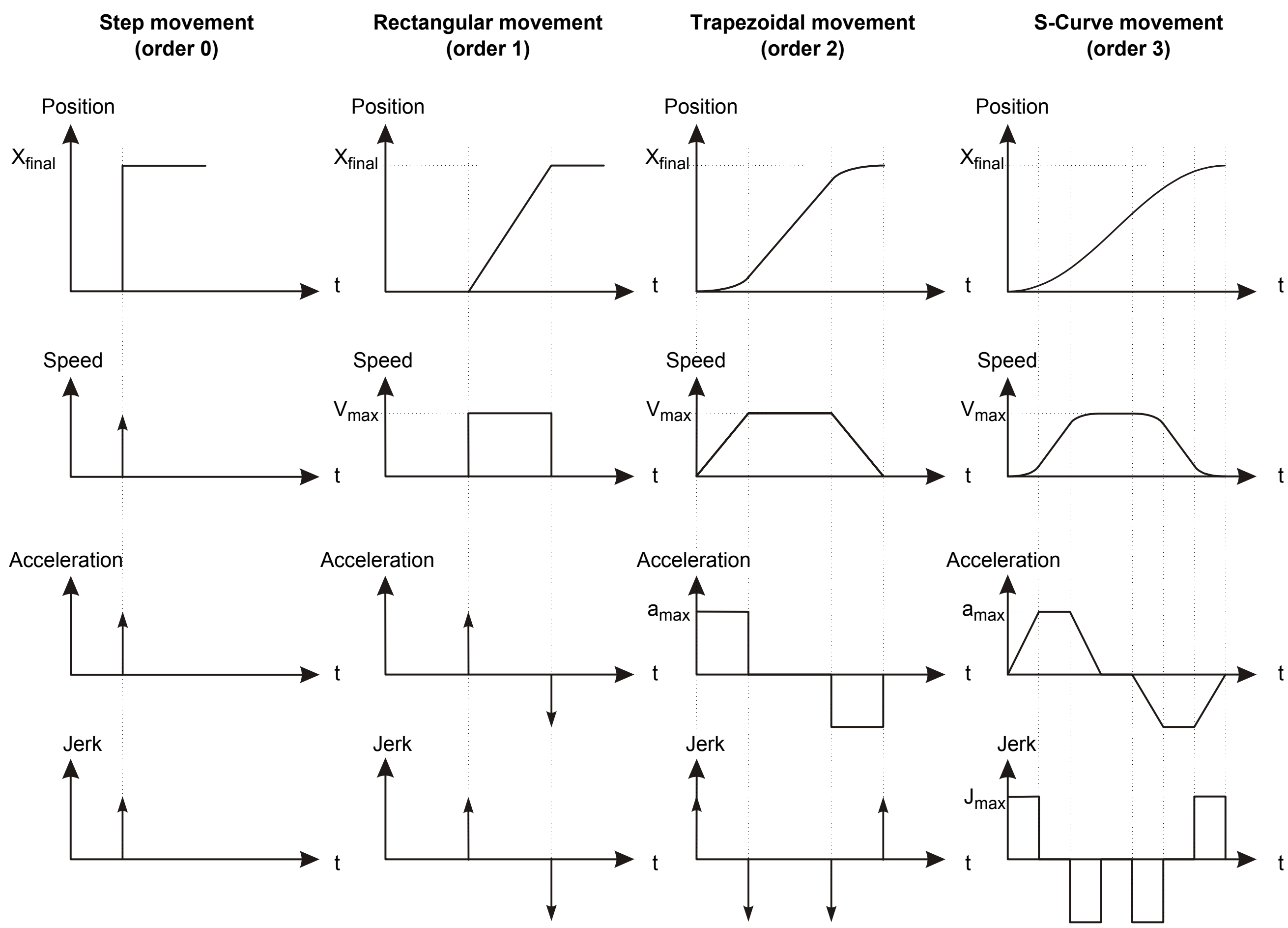
The set point generator calculates the motor position, speed and acceleration references. These references are introduced in the position regulator. This calculation is made according to the type of requested movement, the final position to reach, maximum authorized speed and acceleration. The set point generator carries out one of the most important functions of the controller: the **movement** calculation.

3.1 Introduction to movements trajectories

Note: From the movements described below, the controller uses only the **step movement** (for tuning) and the **S-Curve movement** (for motors movements in applications).
The users interested in the movements equations (order 0,1,2 and 3) can refer to Chapter C (§13.3.11).

A movement trajectory is a function which represents the position of a mobile in one direction versus time. The first derivative of this function gives the **speed trajectory** of the movement (speed versus time). The second derivative determines the **acceleration trajectory** of the movement (acceleration versus time). The third derivative is called the **jerk trajectory** of the movement (jerk versus time).

The **step movement** is a very abrupt movement in which the motor position changes instantaneously. The **rectangular movement** whose speed trajectory is a rectangle, is a specific case of trapezoidal movement. The **trapezoidal movement** is a movement whose speed trajectory is a trapezium and the **S-Curve movement** is a movement which is a step ahead from the trapezoidal movement (trapezoidal acceleration), it is the smoother movement, but for an identical maximum speed, the movement takes more time.



3.1.1 Movements trajectories used in the controller

There are 8 types of movements trajectories available in the controller

Four of them are linear movements:

- Step movement
- S-Curve movement
- Look-up table movement (refer to §4.)
- Calculated movements with predefined profiles

And four of them are rotary movements:

- Rotary S-Curve movement
- Infinite rotary movement
- Rotary look-up table movement (refer to §4.)
- Rotary calculated movements with predefined profiles

The **Step movement** is used by ETEL Tools for the controller & system tuning only (refer to §10. for more information).

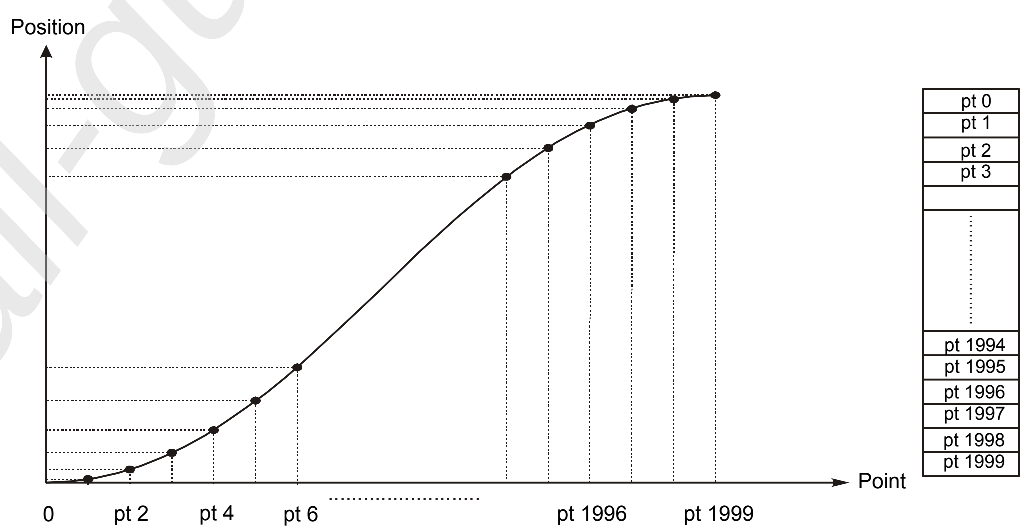
The **S-Curve** and **Rotary S-Curve movements** are used in most applications (refer to §12.10. for more information).

The **Look-up table**, **infinite rotary**, **rotary look-up table** and **calculated** movements are used in some specific applications, by advanced users only (refer to §13.3. for more information).

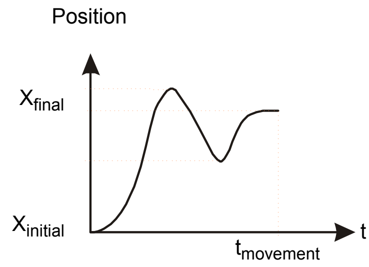
3.1.1.1 Look-up table movement

The **look-up table movements** are movements whose trajectories are freely set by the user. It is possible to save in the controller up to 8 different trajectories.

The **look-up table movement's trajectory** is kept in a table of **2000 points** memorized in a controller, so its name.



For specific applications, the user himself may create the movement trajectory he wishes to use. It can be a complex movement, with back and forth movements, where the controller is used as an electronic cam. Software tools are available to help to create complex movements.

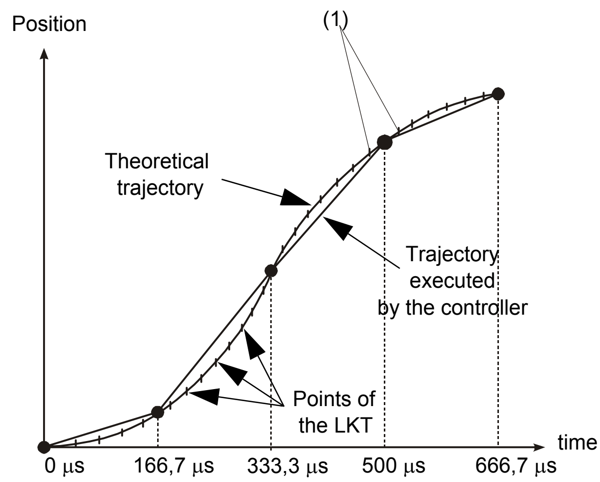


Example of complex movement

When a look-up table movement is required, only select the table with the requested movement and the total time of the movement $t_{movement}$ with LTN and LTI commands respectively. Then the POS command selects the trajectory final point position x_{final} and starts the movement. If a long movement is requested during a very short lapse of time, speed and acceleration may attain very high values, that can even exceed the capacity of the system.

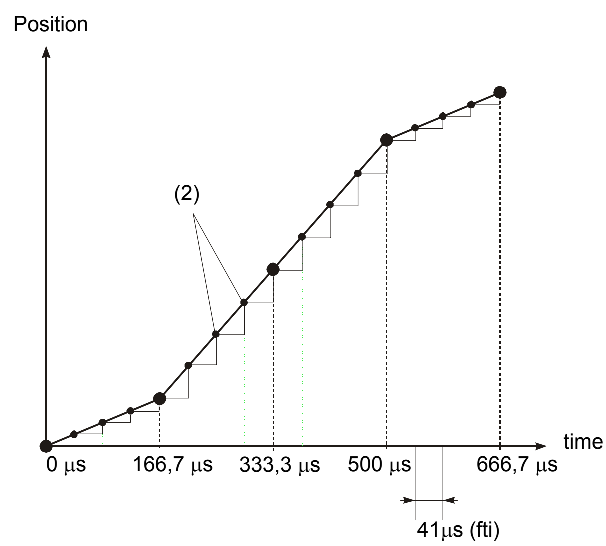
3.1.1.2 *Look-up table movement*

Each sti (refer to §4.), the controller takes a point of the look-up table. The point depends on the value of the execution time (LTI) of the look-up table. If this value should fall between two points of the table, it will be linearly interpolated between the two adjacent points of the table (1).



Example with a DSC2P

Then the points read by the controller in the table are interpolated a second time (2) every fti (refer to §4.). The trajectory is thus made of segments of a sti (refer to §4.). Refer to §13.3.2 for more information about the LKT.



Example with a DSC2P

4. Controller timing

This explanation is for your global understanding of the controller. It is not a critical point to understand for the controller operation.
The controller works on interrupts. Only the interrupts with an impact on the controller timing are explained here:

4.1 STI (Slow Time Interrupt)

The STI is used for:

- Execution of commands coming from several sources: controller internal sequences, TEB communication, optional boards (DSO-CAN, e.g.)
- Trajectory calculation (Set point generator)
- RTI (Real Time Interrupts) and TRI (Triggers) management

Default value:

	DSC2P / DSC2V	DSCDP / DSCDL / DSCDM
Slow Time Interrupt	1/6kHz = 166.67µs	1/2kHz = 500µs

To obtain the actual STI value in seconds, divide M245 by M242: STI [s] = M245 / M242

4.2 FTI (Fast Time Interrupt)

The FTI is used for:

- Linear interpolation between the trajectory points calculated by the STI
- Position loop regulation and position encoder interpolation.
  Calculation of force reference (Fc) delivered to the current loop regulator.
- EBL2 communication, transmission/reception

Default value:

	DSC2P / DSC2V	DSCDP / DSCDL / DSCDM
Fast Time Interrupt	1/24kHz = 41.67µs	1/18kHz = 55.56µs

To obtain the actual FTI value in seconds, divide M244 by M242: FTI [s] = M244 / M242

4.3 CTI (Current Time Interrupt)

The CTI is used for:

- Current loop regulation

Default value:

	DSC2P / DSC2V	DSCDP / DSCDM	DSCDL
Current Time Interrupt	1/24kHz = 41.67µs	1/18kHz = 55.56µs	1/72kHz = 13.89µs

To obtain the actual CTI value in seconds, divide M243 by M242: CTI [s] = M243 / M242

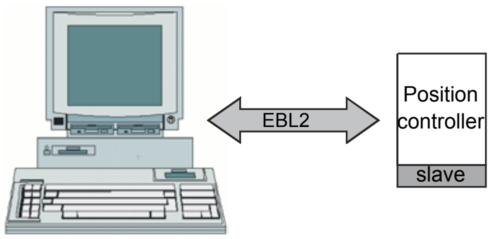
5. Communication with the controller

The communication works with a PC. Three configurations are possible: either a single-axis configuration, or a multi-axis one with DSMAX or DSTEB, or a multi-axis one with micro-master.

5.1 Single-axis configuration

In this configuration, the PC and the controller are linked through the ETEL-Bus-Lite2 (EBL2) protocol (serial communication) whose default communication speed is 115'200 bps.

This protocol is open. Refer to the **'EBL2 Communication Manual'** for more information.



5.1.1 Baud rate configuration

Available on	DSC2P	DSC2V	~~DSCDP~~	~~DSCDL~~	DSCDM

The baud rate of the serial communication (EBL2) can be modified with parameter **K195**.

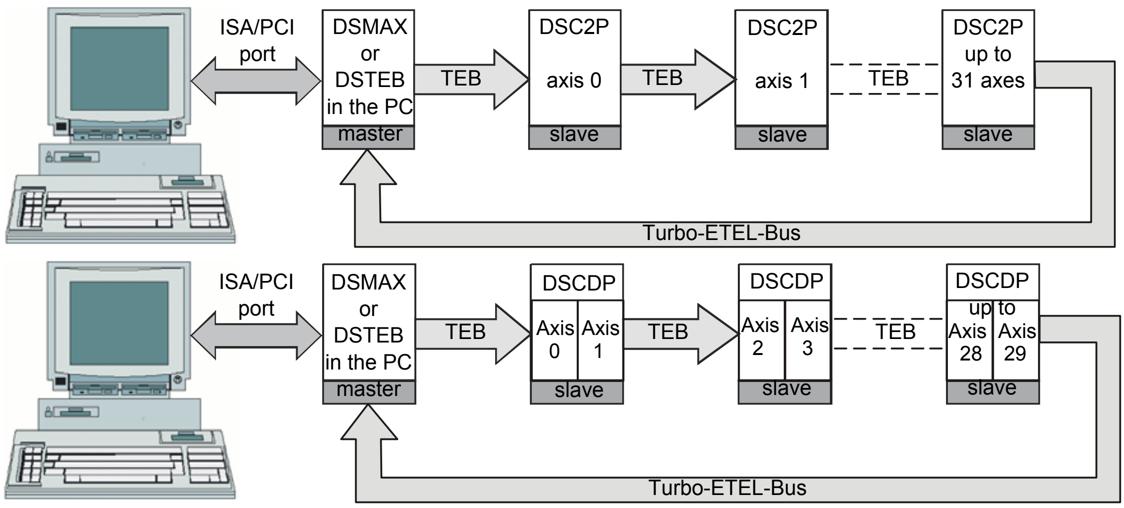
K	Function	Value	Comment
K195	Enables the selection of the ETEL-Bus-Lite2 rate. It is taken into account only at the first switch on. If the user wants to change it, the value must be set in parameter K195, saved into the controller with SAV.<axis>=2 and then the controller must be switched off and on.	0	ETEL-Bus-Lite2 at 115200 Bps
		9600	ETEL-Bus-Lite2 at 9600 Bps
		19200	ETEL-Bus-Lite2 at 19200 Bps
		38400	ETEL-Bus-Lite2 at 38400 Bps
		57600	ETEL-Bus-Lite2 at 57600 Bps
		115200	ETEL-Bus-Lite2 at 115200 Bps

Remark: On the DSCDM, parameter K195 must be saved with the **same value on both axes**.

5.2 Multi-axis configuration with DSMAX or DSTEB

In this configuration, the controllers are linked in daisy-chain. The **master** is a **DSMAX or a DSTEB** motion controller, with whom the PC communicates via the PC ISA or PCI connector. The axes are called **slaves**. All axes are linked together by the Turbo-ETEL-Bus also called TEB (ETEL proprietary protocol). One of the roles of the master is to dispatch the orders he receives from the PC or (sent by itself) to the slaves. Each axis has a personal number, and if several axes are chained, every number must be different from the others (from 0 to 30); the master will always have the number 31. It is then possible to link up to 31 axes.

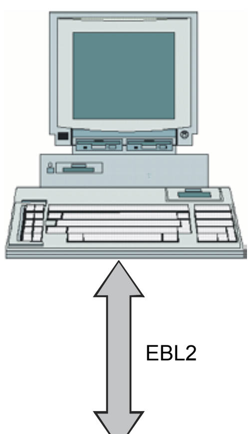
Example:



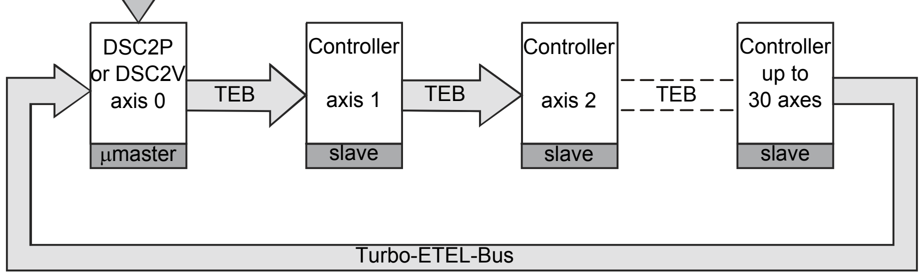
Remark: All the position controllers (slave axes) can be connected together. The only rule to respect is to have 31 slaves maximum.

5.3 Multi-axis configuration with micro-master

In this configuration, the controllers are also linked in daisy-chain. The **master** is a **DSC2P or DSC2V** called **micro-master** (and written µ-master) with whom the PC communicates via the EBL2. The µ-master can also have a sequence which send commands to other controllers. It always has the axis number 0. The other axes are called **slaves**. All axes are linked together by the Turbo-ETEL-Bus also called TEB (ETEL proprietary protocol). One of the roles of the master is to dispatch the orders he receives from the PC or (sent by itself) to the slaves. Each axis has a personal number, and if several axes are chained, every number must be different from the others (from 0 to 30). It is then possible to link up to 31 axes.



The µ-master is a DSC2P or a DSC2V.
The slaves can be any controller



The functioning mode is chosen with the **MDE** command which is an alias of parameter **K170 (only available on the DSC2P and DSC2V)**

K	Alias	Value	Comment
K170	MDE	0	Select slave mode (default mode)
		1	Select µ-master mode

This parameter is taken into account when the controller is switched on. Then, its value must be modified and saved with the SAV command and the controller reset before using this parameter.

Remark: To check if the axis number 0 is in μ-master mode, all you need is to link the TEB IN with the TEB OUT and the led besides 'TEB OK' must be lighted. You can also check the state of the bit# 6 of the M60 monitoring (SD1). If the bit is equal to 0, the controller is in the slave mode and if it is equal to 1, the controller is in the μ-master mode.

In the μ-master mode, the axis 0 can execute normal commands (record 20H), emergency commands (record 18H) and monitorings (record 12H) on all the axes present on the TEB. When the ETEL Tools is connected on the μ-master, all the slaves present on the TEB are visible and can be questioned. The μ-master can execute a sequence and receive normal commands, emergency commands or monitorings via the EBL2 at the same time.

In the configuration with the μ-master you do not have a DSMAX or DSTEB board in the PC, then you do not have access to the interpolation mode.

In the μ-master, monitoring **M81** represents the mask of all the other axes present.

M	Name	Comment
M81	Mask of the axes	Mask of all the axes present and given by the u-master only (axis 0)

Remark: The μ-master mode cannot be used with a field buses optional board.

6. Commands & registers syntax

6.1 Commands

6.1.1 Sending commands

Examples:

```
WTM.1            ;Command without parameter, sent to the axis 1.
AUT.2=1          ;Command with one parameter, sent to the axis 2.
TST.1=X1.1,10    ;Command with two parameters, sent to the axis 1.
```

Syntax:

All the commands which can be sent to the controller are given under the following format:

<cmd_name>.<axis> [=<p1>] [,<p2>]

Fields put in 'square brackets' (like: **[=<p1>]**) are **optional**. All commands do not use them.

<cmd_name> Command name. All controller's commands names have three letters.

.<axis> Axis or group of axis number which have to execute the command.

Possible values:

- **Integer from 0 to 30** according to the number of axis if the command refers to one single axis.
- Symbol **!** if the command refers to all linked axes.
- Some selected axes numbers between commas (refer to §14.4.2).
- Symbol **%** and the axes mask total number, to select some axes (refer to §14.4.2).

[=<p1>][,<p2>] The command can have zero, one or two parameters. If no value is defined, it is automatically the value 0 by default which is selected for a command which needs one or two parameters. (Note: the = link sign is needed only if at least one **[<px>]** field is present).

Possible values:

- **Integer**
- Value contained in any register, at any depth, [px] syntax is similar to a command syntax: **<register>[:<depth>].<axis>**

Remark: An alias like POS, SPD, MMD... is a more user-friendly term representing a parameter K. Their syntax is identical to the one of the corresponding register. Refer to §16. to know the list of the alias.

Some commands calculate parameter values. For example AUT calculates K80, K56 and K53 value. Others use the value contained in the parameters when they are executed. For example INI has a different action according to the value contained in parameter K90. In the commands description which follows, the list of the **calculated parameters** and those of the **read parameters** are given from case to case.

In this manual, only the axis 1 will be mentioned at, to make it simple; for commands explanations all commands have compulsory axis number.

Exception: The STE command (ex: STE.1+=X4.1) uses **+** or **-** operators, with the following syntax:
<cmd_name>.<axis>[<operator>][=<p1>][,<p2>]

Remark: Refer to §8.5 to know how to send a command.

6.1.2 Accumulator operations

Example:

`XAC.0=150/X23.0` ;In the accumulator, divide 150 by the value contained in the X23 variable.

Refer to §14.3 for more information about XAC command (functions with 2 operators).

Syntax:

Operations with the accumulator are given under the following format:

XAC.<axis> = <p1> <operator> <p2>

.<axis> Axis or group of axis number which have to execute the command.

Possible values:

- **Integer from 0 to 30** according to the number of axis if the command refers to one single axis.
- Symbol**!** if the command refers to all linked axis.
- Some selected axes numbers between commas (refer to §14.4.2).
- Symbol **%** and the axes mask total number to select some axes (refer to §14.4.2)

<p1> and **<p2>** Accumulator parameters. If no value is defined, the value 0 by default is automatically selected.

Possible values:

- **Integer**
- Value contained in any register, at any depth, [px] syntax is similar to a command syntax: **<register>[:<depth>].<axis>**

<operator> Mathematical operator. **Possible values:**

- Arithmetical and logical operators (+, -, &,...). Refer to §14.3 for more information.

6.1.3 Sequence labels

When programming a sequence in the controller, labels are used (refer to §14.2).

Example: :11.2

Syntax:

: <label_#>.<axis>

.<label_#> Label distinctive number, defining a part of the controller sequence

Possible values:

- **Integer from 0 to 511**. Labels **:79** and **:80** are specific.

.<axis> Axis number that contains the sequence.

Possible values:

- **Integer from 0 to 30** depending on the axis used.
- Symbol **%** if the command refers to the axis (controller) where the sequence is stored

6.2 Registers groups

The **registers** are accessible to the user, they store all the controller's internal values. Each register has an identification number preceded by a letter corresponding to its group. To have the motor working correctly, it is necessary to set the values of the registers belonging to the **K** group, called **parameters**. There are 6 main types of registers, 3 are basic (always used) and 3 are advanced (for specific applications only):

6.2.1 Basic registers

K, for **parameters** — Also called ***setting parameters***. They define the motor, encoder, regulator gains and the protections (maximum current, maximum position error, over-temperature, etc). Each of them may be modified, by ETEL engineers and trained personal only.

M, for **monitorings** — Also called ***monitoring variables***. They are exclusively used to monitor the controller's internal values such as motor speed, acceleration, motor current, etc.

Note: They can only be read, and no value can be assigned.

X, for **variables** — Also called ***user variables***. They are variables that the user may freely use for programming. Each user variable function may be defined in a program, according to the user's needs. Values can be stored in variables and read at any time.

Examples: Parameter K1 describes the proportional gain value of the position loop.
X13 variable describes the 14th user variable.
M11 monitoring indicates the motor real speed.

6.2.2 Advanced registers

R, for **RTIs** — Also called ***real time interrupts*** *(for **advanced** users only)*, they allow the execution of an immediate function. The sequence execution may jump to a defined label, under some conditions. Refer to §13.8 for more information.

E, for **triggers** — Also called ***triggers events*** *(for **advanced** users only)*, they are used if the user's sytem has to react specifically when the motor reaches some defined positions. Refer to §13.9 for more information.

F, for **float** — Also called ***floating-point variables*** (float-32 registers), used by the controllers mathematical functions (all other registers in the controller are integer). Refer to §14.6 for more information.

L, for **LKT** — Also called ***look-up table movement*** (*for **advanced** users only*), they allow the execution of a movement with a user-defined trajectory. Refer to §3.1.1.1 and §13.3.2 for more information.

T, for **trace** — It allows the acquisition of the registers X, K, M and L of the controller according to the time, used by the 'Scope' of ETEL Tools. Refer to **'EBL2 Communication Manual'** for more information.

S, for **sequence** — It allows the user to write/read a user's programmed sequence. A 'download/upload' menu is available from each tool of ETEL Tools to use the sequence without using the S registers (transparent for the customer). Refer to §8.5.3 for more information.

6.3 Register value attribution

Examples of registers definition:

```
K210:2.10=10000    ;Integer value attributed to parameter K210, sent to axis 10.
K211:3.2=X21.2     ;User variable X21 attributed to parameter K211, both from axis 2.
```

The syntax and operations described below are also valid for XAC value attribution (accumulator).

Syntax:

The syntax giving a value to a register is as follows:

<register> [:<depth>].<axis> [<operator>] = <p1>

Fields put in 'square brackets' (like: **[<operator>]**) are **optional**. They are not always used.

<register>

Defines the register used; is made up of the register's **type** and **number**:

Type possible values:

- **K, X, R, E, S, T, L, F**

Number possible values:

- Integer from **0** to **510** if the register type is **K** or **X**
  Integer from **0** to **7** if the register type is **R**
  Integer from **0** to **191** if the register type is **E**
  Integer from **0** to **8190** if the register type is **S**
  Integer from **0** to **999** if the register type is **T**
  Integer from **0** to **1999** if the register type is **L**
  Integer from **0** to **255** if the register type is **F**
- **Y** (indirect parameterization, refer to the **'EBL2 Communication Manual'** for more information)

[:<depth>]

K parameters may contain up to 4 different values simultaneously. Each value is stored at a different **depth** numbered 0, 1, 2,and 3. If no depth is defined, depth 0 is automatically programmed by default.
For **R** and **E**, **[:<depth>]** is another RTI priority (R) or another trigger line (E).

Possible values:

- Integer from **0** to **3** if the register type is **K** or **E**
- Integer from **0** to **11** if the register type is **R**
- Integer from **0** to **1** if the register type is **T**
- Integer from **0** to **7** if the register type is **L**
- **X** and **F** = 0

.<axis>

Axis or group of axis number whose registers need to be modified.

Possible values:

- **Integer from 0 to 30** according to the number of axis if the command refers to one single axis.
- Symbol **!** if the command refers to all linked axes.
- Some selected axes numbers between commas (refer to §14.4.2).
- Symbol **%** and the axes mask total number, to select some axes (refer to §14.4.2).
- **Y** (indirect parameterization, refer to the **'EBL2 Communication Manual'** for more information)

[<operator>] Mathematical sign for arithmetic and logical operations, only with **K** and **X** registers.

Possible values:

- \+ addition.
- \- subtraction.
- * multiplication.
- / division.
- ~ logical not (**not** for **F** registers).
- **&** logical and (**not** for **F** registers).
- I logical or (**not** for **F** registers).
- **&~** "logical and" and "logical not" (**not** for **F** registers).
- I~ "logical or" and "logical not" (**not** for **F** registers).
- \>> Arithmetical shift to the right (**not** for **F** registers).
- << arithmetical shift to the left (**not** for **F** registers).

= obligatory link sign.

<p1> Set the register's value.

Possible values:

- Integer (immediate value)
- Value contained in another register at any depth (indirect value). It can be taken from any axis.
  <p1> syntax: **<register_name>[:<depth>].<axis>**

Remark: For all registers a **maximum value** and a **minimum value** are defined. If a higher value than the maximum value or a smaller value than the minimum value is given to a register, the register value will automatically be restricted by those two limits at any depth. A **default value** is also defined for each register but only for depth 0. The default value for depth 1, 2 and 3 is the same for all registers and corresponds to 0.

For floating-point variables (F), refer also to §14.6.

6.4 Register value reading

Example of registers reading:

`K1:1.2` ;Read a register (parameter K1, depth 1), from axis n° 2.

Syntax:

The command syntax which allows the reading of a value inserted in a register is as follows:

<register1> [:<depth1>].<axis> [, <register2> [:<depth2>].<axis>]

Fields put in 'square brackets' (like: **[:<depth1>]**) are **optional**. They are not always used.

<register1> and **2** Defines the register used; is made up of the register's **type** and **number**:

Type possible values:

- **K, X, M, R, E, S, T, L, F**

Number possible values:

- Integer from **0** to **510** if the register type is **K** or **X**
  Integer from **0** to **255** if the register type is **M**
  Integer from **0** to **7** if the register type is **R**
  Integer from **0** to **191** if the register type is **E**
  Integer from **0** to **8190** if the register type is **S**
  Integer from **0** to **999** if the register type is **T**
  Integer from **0** to **1999** if the register type is **L**
  Integer from **0** to **255** if the register type is **F**

[:<depth1>] and **2** If no depth is defined, depth 0 is automatically programmed by default. **K** parameters may contain up to 4 different values simultaneously. Each value is stored at a different **depth** numbered 0, 1, 2 and 3. If no depth is defined, depth 0 is automatically programmed by default.
For **R** and **E**, **[:<depth>]** is another RTI priority (R) or another trigger line (E).

Possible values:

- Integer from **0** to **3** if the register type is **K** or **E**
- Integer from **0** to **11** if the register type is **R**
- Integer from **0** to **1** if the register type is **T**
- Integer from **0** to **7** if the register type is **L**

.<axis> Number of the axis whose register value may be read, whatever the register type.

Possible values:

- **Integer from 0 to 30** according to the number of axis if the command refers to one single axis.
- Some selected axes numbers between commas (refer to §14.4.2).
- Symbol **%** and the axes mask total number, to select some axes (refer to §14.4.2).

Note: Reading simultaneously two registers is possible if both commands are separated by a coma but it only works with two registers belonging to the same axis.
Example: M6.1, M7.1

6.5 Bit fields or numerical values for registers and commands

The registers and commands values can be of two types: **bit fields** or **numerical values**.

6.5.1 Bit fields

Some registers and commands values are defined as bit fields. Thanks to this feature, their corresponding functions may be combined by using a **binary mask**.

How to recognize them? If the numbers (0, 1, 2, ...) are present in the table under the header **Bit#**, the register or the command value is defined as **a bit field**.

For example, K68 is a parameter with 3 functions, defined as bit fields. Each function corresponds to a bit. Thanks to the binary mask, its 3 functions may be combined in 2^3=8 possibilities.

K	Name	Value	Bit#	Comment
K68	Encoder reading way and force reference inversion	1 2 4	0 1 2	Analog 1Vptp encoder reading way is inverted. TTL encoder reading way is inverted. Force reference from the Macro optional board is inverted

Binary mask example:

The user can invert the analog position encoder (bit# 0, value of **K68=1**) **and also** the TTL encoder reading way (bit# 1, value of **K68=2**).

Simply add the two bits: **K68=3**, and both functions are combined.

Remark: When all bits = 0, their functions are not active. Thus, K68=0 is not described in the table above; this is the case in this document for all registers and commands values defined as bit fields.

6.5.1.1 Numerical values

Most registers and commands are defined as simple numerical values. Their corresponding functions cannot be combined.

How to recognize them? If **no numbers** are present in the table under the header **Bit#**, the register or the command value is defined as **a numerical value**.

For example, K90 is a parameter with 7 functions, each function corresponds to a different numerical value.

K	Name	Value	Comment
K90	Phasing mode and commutation	0 1 2 3 4 5 6	No phasing (with 1-ph. motor or EnDat 2.1 encoder) Phasing by current pulses (ironcore motors only) **(~~DSCDL~~)** Phasing by constant current in the motor phases Phasing with digital Hall effect sensor (mode 1) **(~~DSCDL~~)** Phasing with digital Hall effect sensor (mode 2) **(~~DSCDL~~)** Phasing and commutation with digital Hall effect sensor only **(~~DSCDL~~)** Small movement Phasing

6.5.2 Examples of use

This chapter is dedicated to examples which show how to set a register and to read the value contained in it. To set a register with an immediate value or with a value contained in any other register (K, X or M) or even adding a value of one register to the one of another register, etc, is possible

Do not forget that it is not possible to write a value in a monitoring M and those depths are only defined for parameters K.

Take parameters K4 and K1 of axis 1 and K4 and K32 of axis 2, each containing the values shown in this table. Examples are gathered in the table below and divided into two groups, the first with register value reading examples, and the second with register setting examples.

	K4	K1	K4	K32	M4
Depth 0	0	1000	4	1	32
Depth 1	8	23	0	2	
Depth 2	0	37	5	2	
Depth 3	1	9	0	8	
	Axis 1	**Axis 1**	**Axis 2**	**Axis 2**	**Axis 2**

The controller always sends in response the line entered by the user unless the register has an alias which is sent back in this case.

	Input command	Controller response	Comment
Reading register value	K1.1	K1.1=1000	Reads the value contained in the K1 user register of the axis 1 at depth 0
	K4:0.2	K4:0.2=4	Reads the value contained in the K4 user register of the axis 2 at depth 0.
	K32:2.2	K32:2.2=2	Reads the value contained in the K32 register of the axis 2 at depth 2.
	M4.2	M4.2=32	Reads the value contained in the M4 register of the axis 2.
	K32:1.2, K4:2.2	K32:1.2=3, K4:2.2=5	Simultaneous reading of two registers. This function only works when both registers belong to the same axis.
	M73.1	SER.1=1245	The M73 register contains the serial number of the controller. Since the SER alias is defined for this register, the controller responds with the alias.
Register setting	K4.2=15	K4.2=15	The immediate value 15 is stored at depth 0 (by default) of the axis 2 and replaces the value 4.
	K4:2.2=23	K4:2.2=23	The immediate value 23 is stored at depth 2 of the axis 2 and replaces the value 5.
	K4:1.1=K32:3.2	K4:1.1=K32:3.2	The value contained in register K32 of axis 2 at depth 3 is copied in register K4 of the axis 1 at depth 1. 12 replaces 8.
	K4:3.1+=M4.2	K4:3.1+=M4.2	The K4 register of the axis 1 contains at depth3 the value 33=1+32
	K1:1.1/=7	K1:1.1/=7	The K1 register of axis 1 contains at depth 1 the value 3=23/7. It is the integer part of the division.

Remark: Refer to §14.5 for more information about the mathematical operators +, -, *, /, ~, &, |, &~, |~, >> and <<, .

Chapter B: System setup and tuning

7. Initial system installation

This chapter helps a new user to install a controller for the first time (in its minimal configuration). In case of a more specific application, refer to the corresponding **'Hardware Manual'** for more information about the electrical interfaces.

7.1 Controller connection

Every new controller should first be connected separately (without Turbo-ETEL-Bus) to check its interfaces.

Caution: Do not switch on the controller before all connections are wired! Never plug or unplug an encoder connector while the controller is switched on; this could damage the encoder's reading head!

⚠ First of all, **always connect the protective earth (PE)** before any other connection!

- Use the cables delivered with the controller(s). If you manufacture your own cables, refer to the corresponding **'Hardware Manual'** for more information about the pin assignment and the **shielding**.

7.1.1 Stand-alone configuration

For a stand-alone configuration, plug the connectors as listed below:

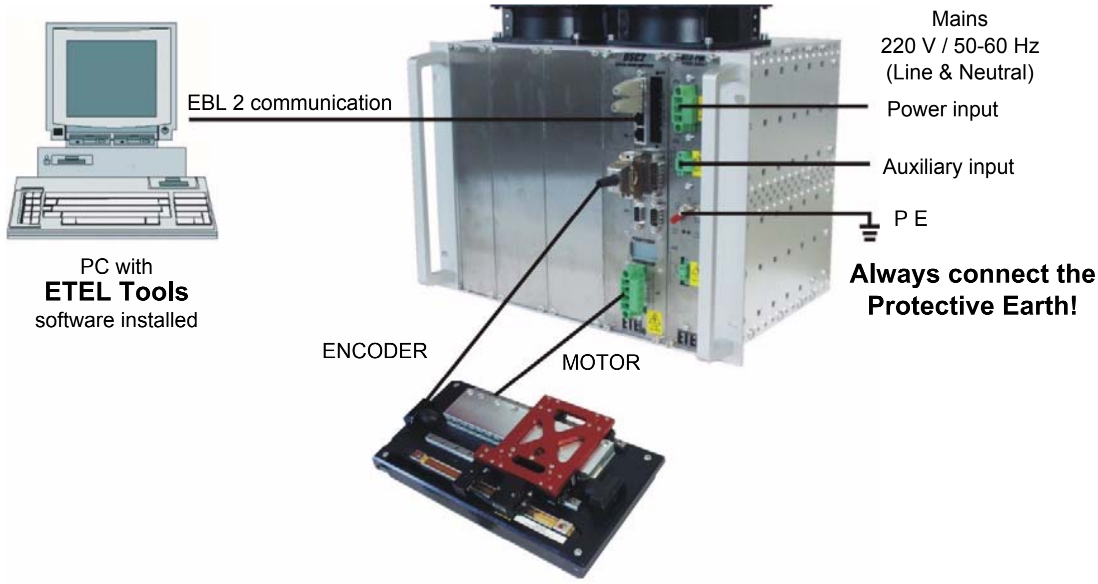
Interfaces to be connected	Required	Optional
1) EBL2 serial communication (refer to the corresponding **'Hardware Manual'** to know the connector number). No other application should run on the PC port used for the EBL2 serial communication!	X	
2) Position encoder (refer to the corresponding **'Hardware Manual'** to know the connector number)	X	
3) Motor connection (refer to the corresponding **'Hardware Manual'** to know the connector number)	X	
4) Power supply connection (refer to the corresponding **'Hardware Manual'** to know the connector number)	X	
5) Motor protection (refer to the corresponding **'Hardware Manual'** to know the connector number)		X
6) Customer I/O (refer to the corresponding **'Hardware Manual'** to know the connector number)		X

- When the connections are realized, turn on the power supply.
- The controller performs a SELF TEST and will display on the LCD its: controller family, product number, firmware version present in the controller, axis number, presence of an optional board,....
- A message (for example: DSC2P READY) appears on the LCD display (the controller is not in the 'Power On' mode).
  Never connect the motor if 'Power On' **already appears on the LCD display!**

Remark: If there is no physical display on the controllers, the user can use M95.<axis> (the conversion is automatically done by the DLLs) or can select 'scope drive LCD display' in the 'scope' menu of the scope tools (ETT) to display a software display indicating the status, the error and the warning messages.

- The system is ready to work.

Example with a DSC2P:



Warning: **Risk of electrical shock!** High voltage may be present on the motor connector. Turn off the power, **wait 10 minutes (or 2min for the DSCDM)** and **before touching them,** check with a voltmeter that no residual voltage remains on these connectors!

7.1.2 Configuration with a DSMAX or a DSTEB

For a configuration with a DSMAX or a DSTEB, plug the connectors as listed below:

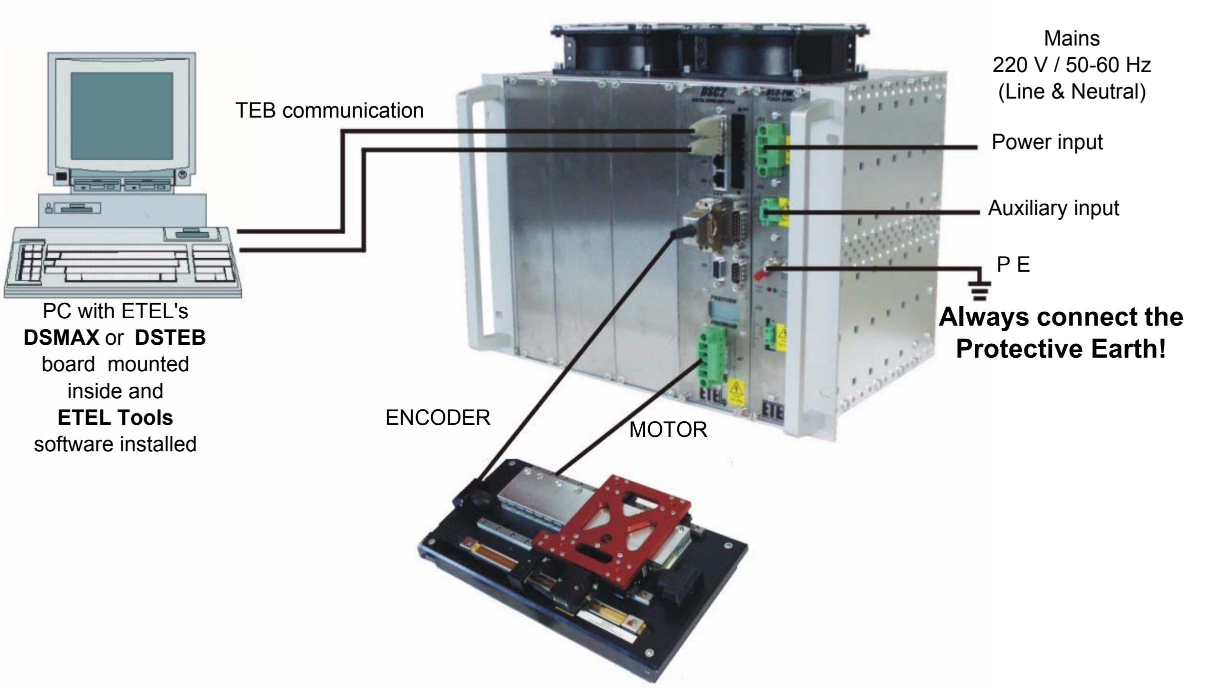
Interfaces to be connected	Required	Optional
1) TEB communication (refer to the corresponding **'Hardware Manual'** to know the connector number)	X	
2) Position encoder (refer to the corresponding **'Hardware Manual'** to know the connector number)	X	
3) Motor connection (refer to the corresponding **'Hardware Manual'** to know the connector number)	X	
4) Power supply connection (refer to the corresponding **'Hardware Manual'** to know the connector number)		
5) Motor protection (refer to the corresponding **'Hardware Manual'** to know the connector number)		X
6) Customer I/O (refer to the corresponding **'Hardware Manual'** to know the connector number)		X

- When the connections are realized, turn on the power supply.
- The controller performs a SELF TEST and will display on the LCD its: controller family, product number, firmware version present in the controller, axis number, presence of an optional board,....
- A message (for example: DSC2P READY) appears on the LCD display (the controller is not in the POWER ON mode).
  Never connect the motor if POWER ON **already appears on the LCD display!**

Remark: If there is no physical display on the controllers, the user can use M95.<axis> (the conversion is automatically done by the DLLs) or can select 'scope drive LCD display' in the 'scope' menu of the scope tools (ETT) to display a software display indicating the status, the error and the warning messages.

- The system is ready to work.

Example with a DSC2P:



Warning: **Risk of electrical shock!** High voltage may be present on the motor connector. Turn off the power, **(or 2min for the DSCDM)** and **before touching them,** check with a voltmeter that no residual voltage remains on these connectors!

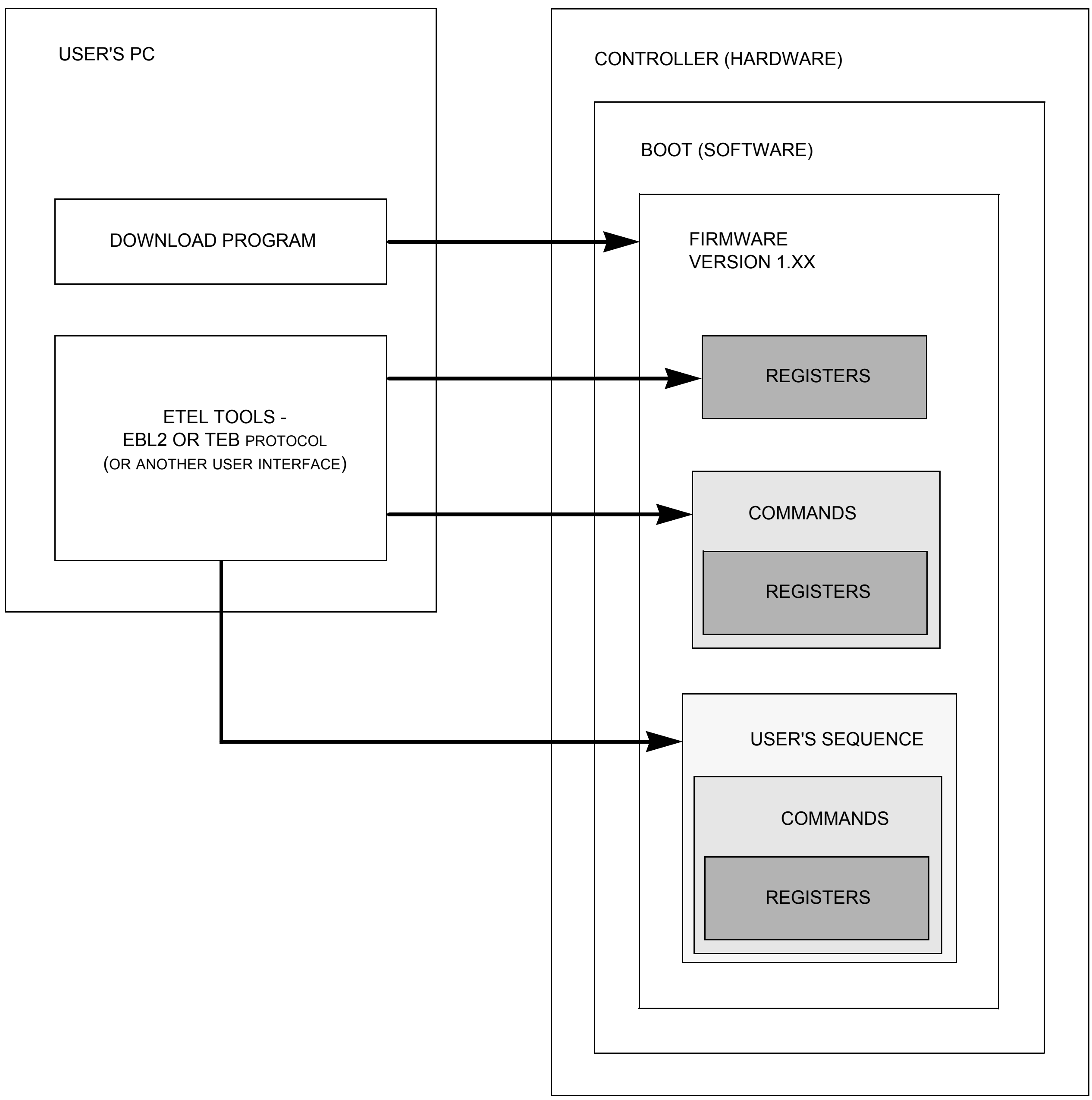
7.2 Controller setup principle

When the hardware is installed, you will have to introduce the parameters to the controller to make the system work.

The values of these parameters depend on the motor's characteristics and on your system's specifications.

The 'ETEL Tools' (ETT) software provides the necessary tools to set up the controller. It is possible to send parameters and commands (a command can include parameters) to the controller. ETT 'Scope tool' is used to monitor graphically the system's performance. ETT allows you also to build a sequence of commands, performing the movements required by the application.

The diagram below outlines this principle:



7.3 Install ETEL Tools software

7.3.1 System requirements

The PC minimal configuration to install the ETEL Tools software (ETT) is:

- Windows 9x / 2000 / NT / XP
- Pentium processor 166 MHz, or faster
- RAM 64 Mo, or greater
- Hard disk free space 250 Mo, or greater
- Screen definition 800x600 pixels, or more
- CD-ROM drive
- Mouse

7.3.2 Installing ETT

You should have received ETT on a CD-ROM. The version 4.11 or above must be used to take all these controllers and functionalities into account. To install it, proceed as follows:

- Open the directory **ETEL Tools 4.xx** (the number of the version will change with new releases)
- Run the **Setup.exe** application
- Follow the installation steps:
  a) Read and accept ETEL's **License agreement**.
  b) Choose the destination directory for ETT (by default it is: C:\Program Files\ETEL).
  c) The program folder will be **ETEL Tools 4.xx** by default.
  d) The installation will take a few minutes, depending on your PC performances.
  e) A shortcut called **ETEL Tools 4.xx** (will change with new releases) has been created on your desktop. If you look in your Windows **Start** menu under **Programs**, you will find an **ETEL Tools 4.xx** directory, containing shortcuts to 2 files: **ETEL Tools 4.xx**, and **Documentation**.
- **Restart** your PC after the installation is completed.

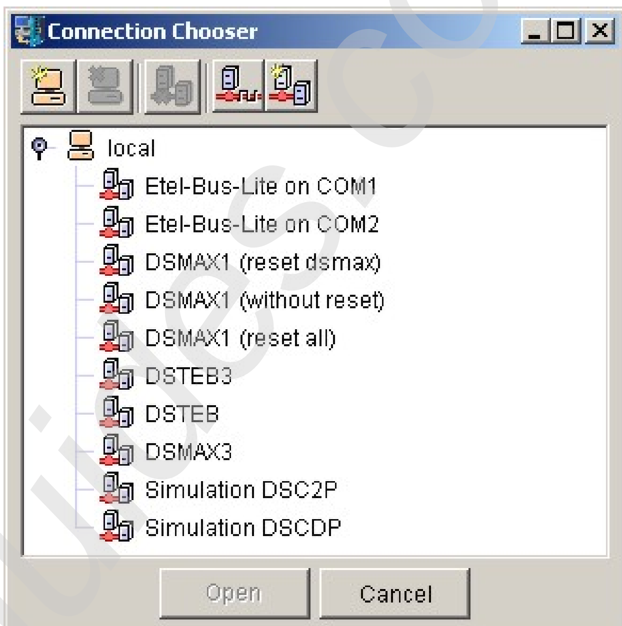
8. Controller setup with ETEL Tools

The **ETEL Tools** software **ver.4.xx** (ETT) is a user-friendly interface developed by ETEL to set up and monitor the operation of the controllers. The users who are not accustomed to work with ETEL's products should read the following paragraphs, and experience all the functionalities described below.

8.1 Run ETT and set up the communication

1° Double-click on the shortcut to **ETEL Tools 4.11** (version will change with new releases).

2° The ***Connection Chooser*** window appears, allowing the setting of the EBL2 communication parameters between the PC and the controller, or the TEB (communication with DSMAX or DSTEB if present). ETT needs this communication between the PC and the controller (through DSMAX-TEB if present).



3°a Select the port to be used by EBL2 on your PC (ETEL-Bus-Lite 2 on COM1 recommended). **No other application should run on the port selected for EBL2!** 115'200 bps is the communication speed's default setting. Click on ***Open***. Go to **4°a**.

3°b Select the corresponding DSMAX:
-DSMAX1 (reset DSMAX): the DSMAX1 is reset at each new connection.

-DSMAX1 (without reset): the DSMAX1 and the controllers keep at each new connection the configuration they had before the last disconnection.

-DSMAX1 (reset all): the DSMAX and the controllers are reset at each new connection.

-DSMAX3: the DSMAX3 and the controllers keep at each new connection the configuration they had before the last disconnection.
Click on **Open**. Go to **4°b**.

Remark: If you select 'Dummy DSC2P' and click on **Open**, the controller will be simulated. It is only used to open and display the tools.

4°a The ***Main Menu*** window (in the background) appears. ***Status: Connected to ....*** appears on the window's upper right corner:

4°b The ***Main Menu*** window (in the background) appears. ***Status: Connected to ....*** appears on the window's upper right corner:

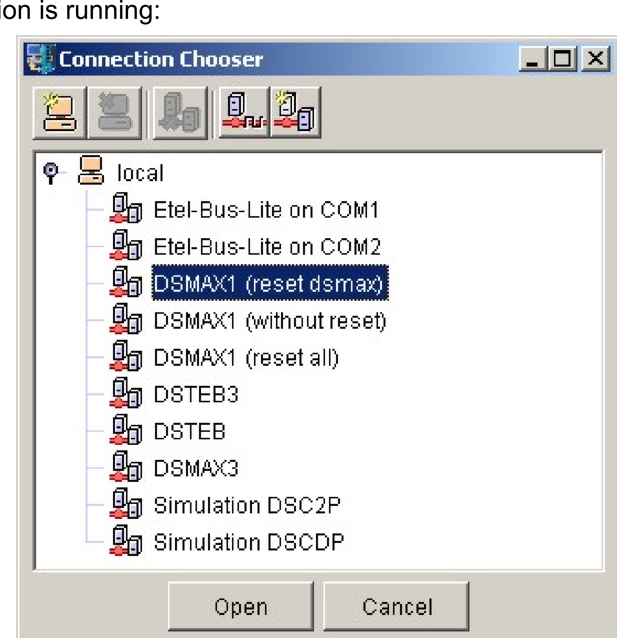
If the ***Connection Chooser*** window does not disappear, the communication is not established and ***Status: Disconnected*** appears next to the **Connection/Disconnection** icon:

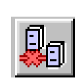
In this case, check the port selected on your PC (and your connection with the DSMAX if present), and retry to set up the communication!

 This button is used to connect a 'remote host' (PC, e.g).

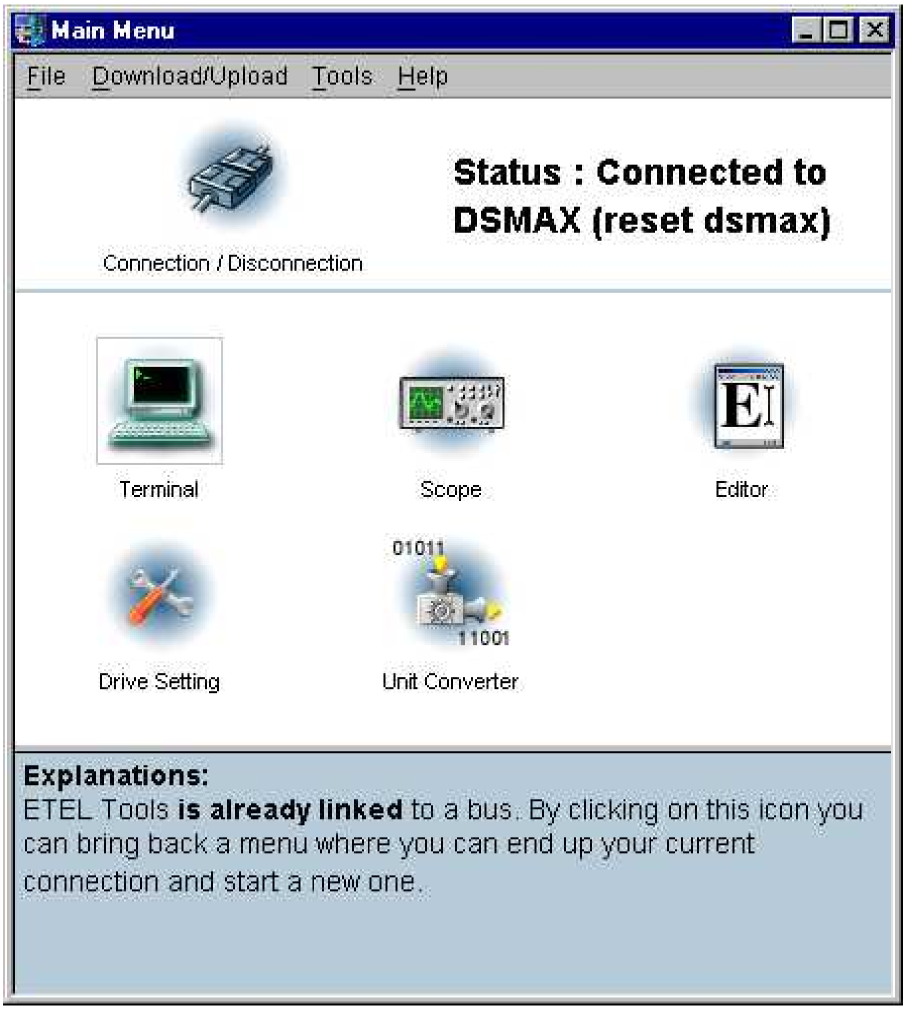
To go back to the ***Connection Chooser*** window from the ***Main Menu*** window, click on the button *Connection/ Disconnection,* as shown above (or from any other tool, except the *Download Wizard*), click on ***File*** in the menu bar and select ***Connection Chooser***. (shortcut: Ctrl-E).

If you go back to the Connection Chooser window, ***DSMAX1 (reset DSMAX)*** is in bold, indicating that the DSMAX1 communication is running:



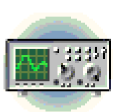
To stop the DSMAX1 (reset DSMAX) communication, click on the following button: 

8.2 Main Menu window

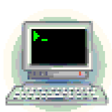


When you click on one of these five icons, ETT launches subprograms called tools and opens new windows. If you put the pointer on an icon, a short tool's description appears at the bottom under ***Explanations:***

The **'Drive Setting'** tool is used first, to quickly tune the controller's parameters and to automaticallyset the system's safeties. The user needs only to follow a step-by-step process by answering the questions asked by the 'Drive Setting'.

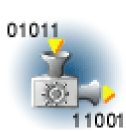


The **'Scope'** tool is used after the 'Drive Setting' tool, to accurately tune the controller's parameters. The controller's main registers are accessible and a graphical 2-channels oscilloscope allows the monitoring of the system's performance (position reference, real position, speed, current, ...).



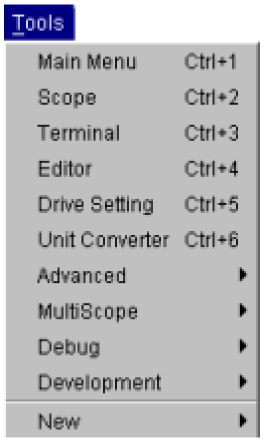
The **'Terminal'** tool is used to directly send commands to a controller or a group of drives. The ’Terminal' reacts to these commands and indicates if the last command has been acknowledged by the controller(s) or not, and displays the result(s).

The **'Editor'** tool is used to write a sequence for a controller. It includes all classical features of a basic text processing tool and an automatic syntax coloring system. The sequence should be saved later with the .SEQ extension for a proper handling by all the other tools.



The **'Unit Converter'** tool is used to convert position, speed and acceleration values from ISO units into controller increments units. The reverse conversion is also possible. To use the 'Unit Converter', the ETT communication with a drive has to be established.

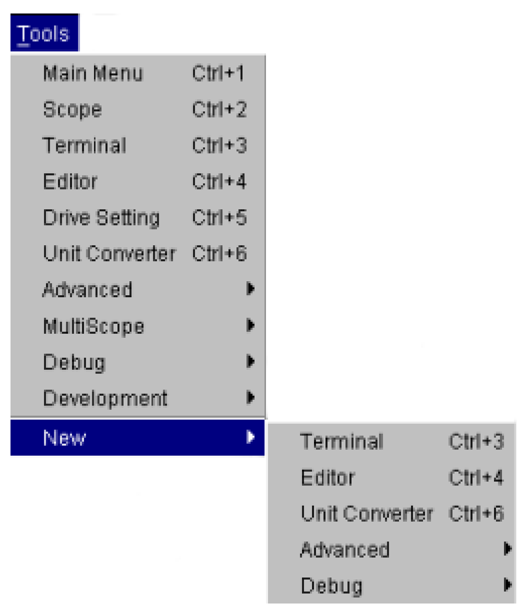
8.2.1 Tools menu



If you click on ***Tools*** in the menu bar (in the **Main Menu window,** or in other tools), you will first see all basic tools (***Main Menu*** refer to §8.2, ***Scope*** tool refer to §8.4, ***Terminal*** tool refer to §8.5, ***Editor*** tool refer to §8.6, ***Drive Setting*** tool refer to §8.3, ***Unit Converter*** tool refer to §8.7).

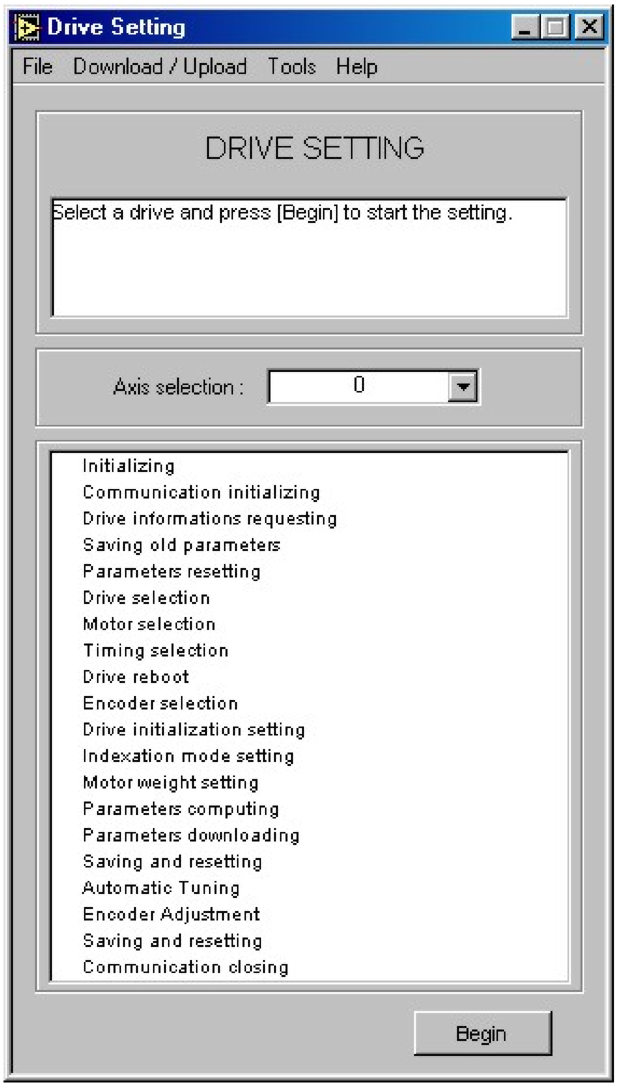
The MultiScope and all the tools listed under ***Advanced*** are reserved to **experienced users only**!

Some tools may be opened several instances simultaneously. They are listed under ***New***:



8.3 Drive Setting tool

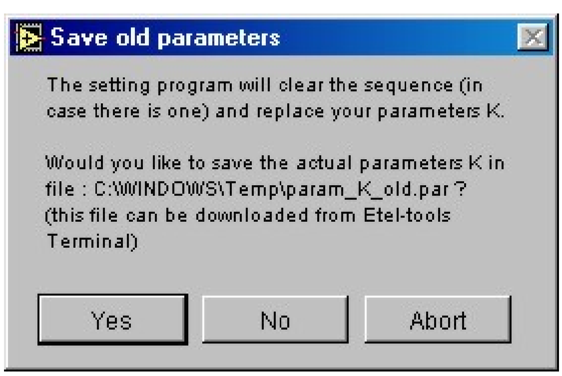
Remark: Starting with the drive setting process is possible only if the TEB communication is established between the DSMAX board and the controller(s) or if the EBL2 communication is established between the PC and the controller.



The ***Axis selection*** detects automatically (with EBL2 communication) the axis number; this number cannot be changed (with TEB communication, the axis number can be selected).

- Click on the ***Begin*** button to start the drive setting process.

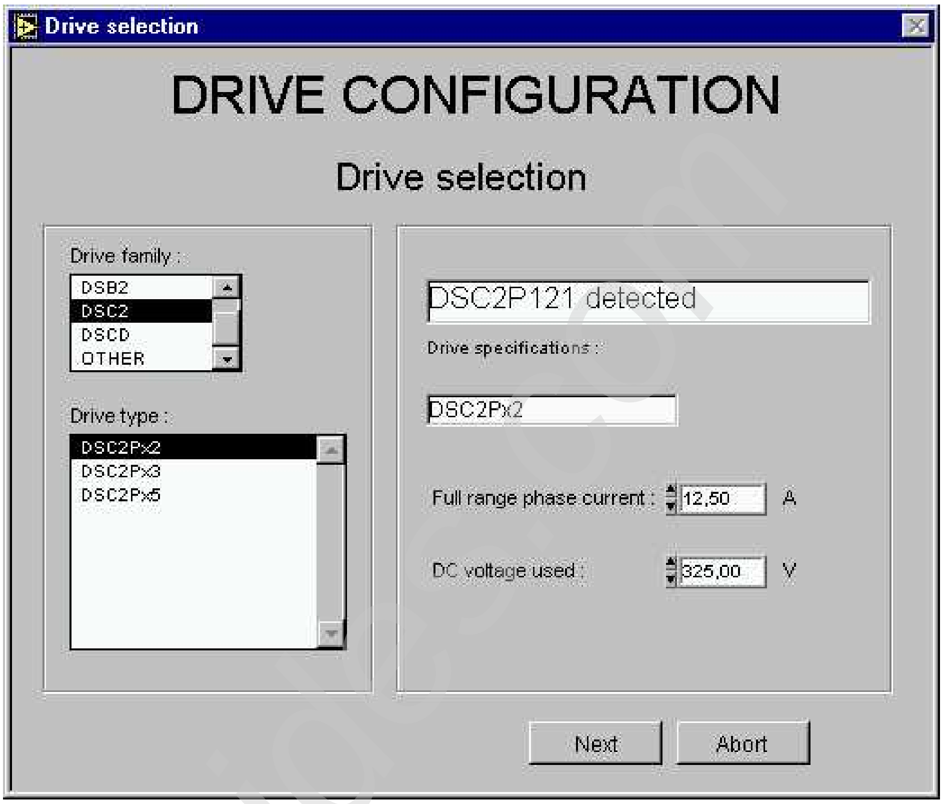
ETT proposes to save the 'old' parameters stored in your controller; you should click on ***YES***.
Caution: By clicking on ***NO***, you will loose all parameters you were previously using with the controller!



- Click on the ***YES*** button and the next window will appear.

8.3.1 Drive selection

ETT detects automatically which controller type is connected to the DSMAX (with the TEB). The drive type (here: DSC2Px2) and its main specifications (bus voltage = 325 VDC / Max. measurable current = 12.5 A) are displayed:



- Check that all data fits to your controller's characteristics.
- Click on the ***NEXT*** button and the next window will appear.

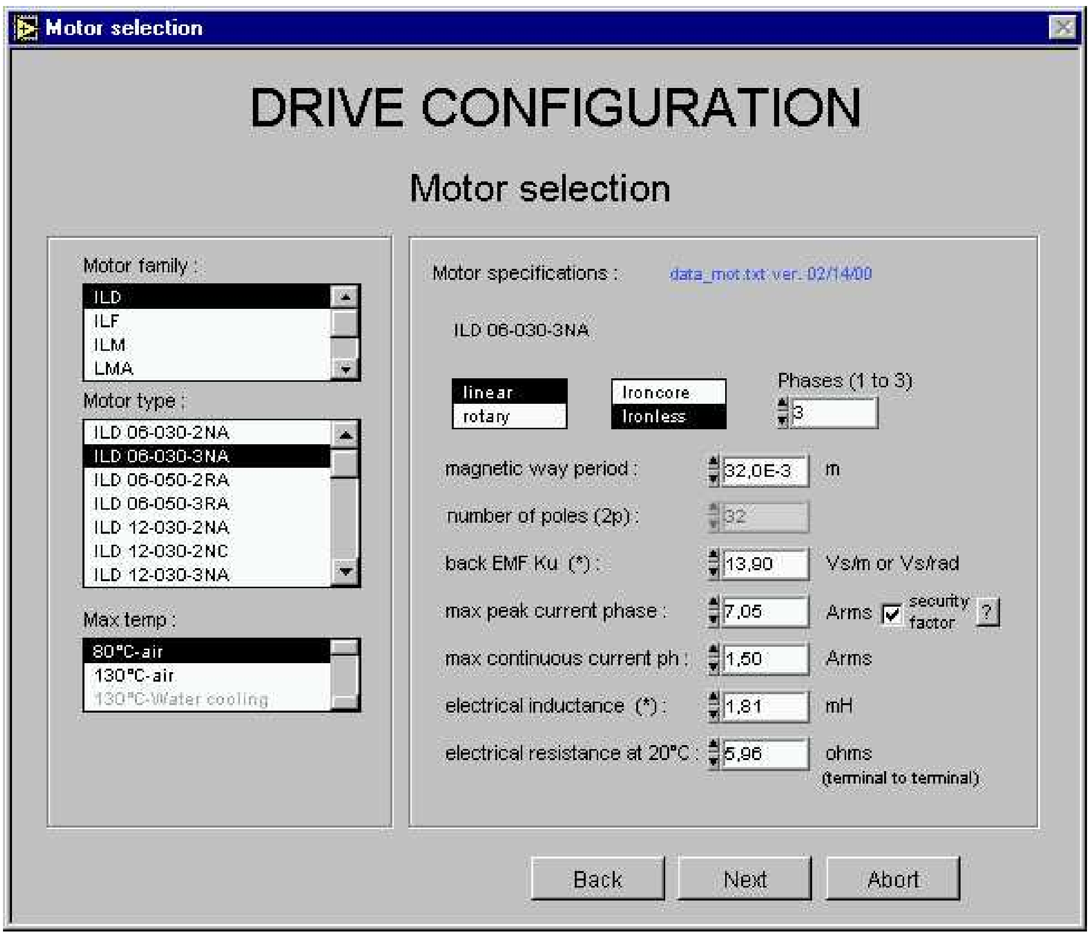
Remark:

	Full range phase current (measurable)	Peak current
DSC2Px2	12,5 [A]	11 [A] for 2 [sec] (with 3-phase motor)
DSC2Px3	25 [A]	21 [A] for 2 [sec] (with 3-phase motor)
DSC2Px5	66,67 [A]	56 [A] for 1[sec] (with 3-phase motor)
...		
DSCDPx2	12,5 [A]	11 [A] for 2 [sec] (with 3-phase motor)
...		

The *DC voltage used* value will influence the phasing parameters (K90 to K98 and K101).

8.3.2 Motor selection

- The motors types manufactured by ETEL are listed in the scroll menu.
- Click on your motor family (here: *ILD*) and type number (here: *ILD 06-030-3NA*).
- The motor specification will appear in the adjacent fields on the right.
  Here, the specifications are: *Linear - Ironless - 3 phases*
  Magnetic way period = 32.0E-3 [m]
  Back EMF / Ku = 13.90 [Vs/m] or [Vs/rad]
  Max peak current in a phase = 7.05 [Arms]
  Max continuous current in a phase = 1.50 [Arms]
  Inductance per phase = 1.81 [mH]
  Resistance per phase at 20 °C = 5.96 [Ω]
- If you are using a motor type not present in the scroll menu, click on *OTHER*.
  Then, type the applicable values to complete the motor's specification fields.
- Select the motor **max. working temperature** (here: *80°C - air cooling*).
  Caution: This max. temperature may be reached only if the system cooling is designed accordingly! Read the motor specifications in details to determine the cooling issues.

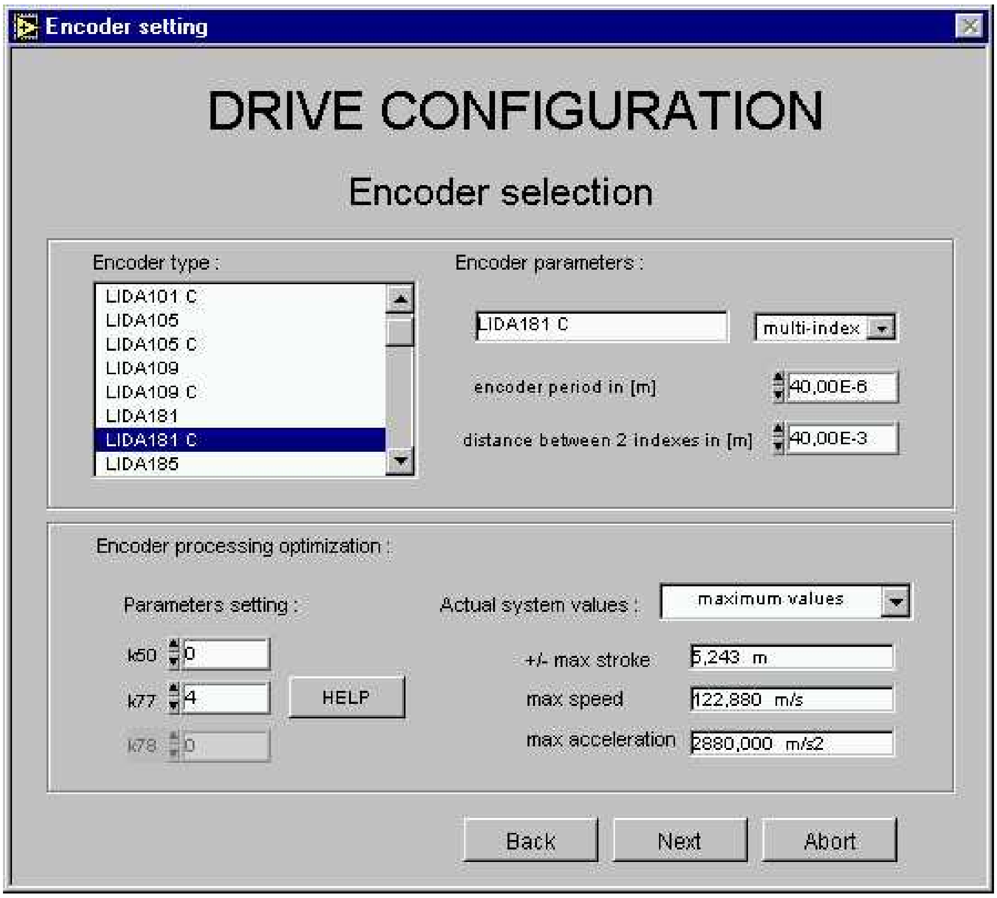


- Control the motor's specifications before performing the next step.
- Click on the ***NEXT*** button and the next window will appear.

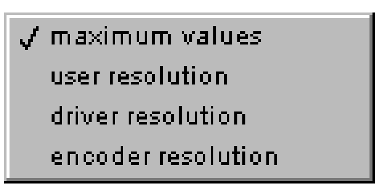
Caution: If you select a wrong motor type, the current limitation and protection parameters may not be correctly set up in the controller. If this occurs, **the motor could burn** or the wirings may be irreversibly damaged!

8.3.3 Encoder Selection

- The encoder types recommended by ETEL are listed in the scroll menu.
- Click on your encoder type number (here: *LIDA 181 C*).
- The selected encoder specification will appear in the adjacent fields on the right.
  Here, the specifications are: *Multi-index encoder*
  Encoder period = 40.00 E -6 [m]
  Distance between 2 indexes = 40.00 E -3 [m]
- If you are using an encoder type not present in the scroll menu, click on ***analog OTHER*** or ***TTL OTHER***. Then, type the values to complete the encoder's specification fields.
- Under ***Encoder processing optimization***, three ***Parameters setting*** fields allow the setting of the controller parameters K50 and K77. Default values are set by ETT. Your system will work with these values. Ideally, you should optimize them (read this **'Operation & Software Manual'** to do it).
  Click on ***HELP*** for a short explanation about these parameters.



- Under ***Actual system values***, you can read some system data directly linked to parameters K50 and K77. Click on the black arrow to choose which set of values you want to see:



- Check the ***Encoder parameters*** and the ***Actual system values*** before performing the next step.
  Note: Parameter K77 is the encoder interpolation factor. With K77=3, the best interpolation (11 bits) is selected. Most applications use it to have the highest position measurement resolution. However, if the application requires a lower precision and higher speed and acceleration, you may reduce the value of parameter K77.
- Click on the ***NEXT*** button and the next window will appear.

8.3.4 Initialization mode

There are two ways of initializing a motor:

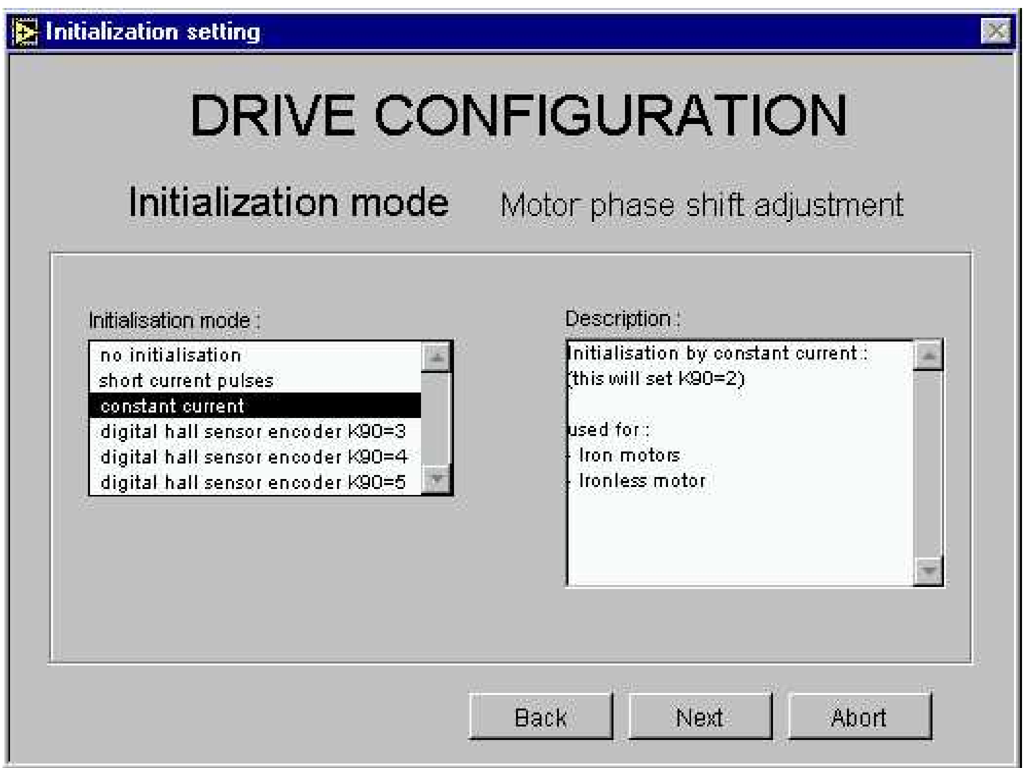
The first consists in sending a **constant current** in the phases. In that case, the motor moves until it reaches a 0 force position. Then, it stops on that **stable balance point**. Its position with respect to the magnets way (motor's fixed part) is known because the stable balance point is always the same between two poles (as long as the relation between the currents in the phases is constant). The longest stroke covered during the initialization is equivalent to one magnetic period (32 mm for most ETEL's motors).

The 2nd way, which **only works with ironcore motors**, consists in sending **current pulses** to the motor phases. It determines the motor position in relation with the magnets. The advantage of this method is that the motor does not move, but it is not as precise as the first one (20% off the optimum phase shift adjustment). This 2nd way should be especially used with vertical linear motors, or if the system mechanical friction is high.

After the initialization procedure, the motor position in relation with the magnets is known with a sufficient accuracy to work out some movements. Sometimes, the accuracy may not be sufficient. The phase shift adjustment quality may vary from an initialization to another depending on the initial position of the motor. These problems will be solved with the **homing process** (refer to the next page).

The ***constant current*** mode is selected by default, this will always work for all motors with an encoder.

- If you are an experienced user, you may modify this default setting (refer to the explanations above).

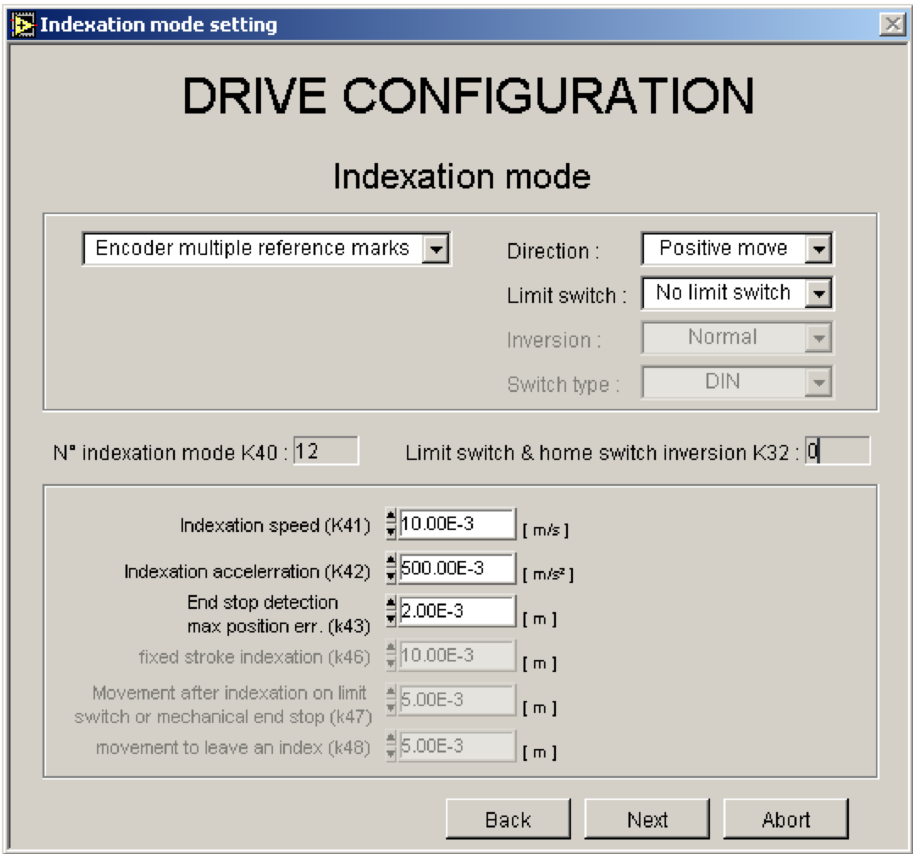


- Click on the ***NEXT*** button and the next window will appear.

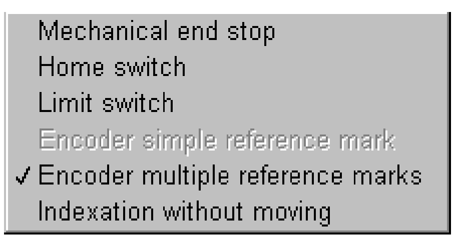
8.3.5 Homing mode

After the initialization, a **homing** (also called **indexation**) has to be carried out, moving the motor until it reaches a fix reference mark called **index.** The motor movement can highly vary during a homing according to the type of encoder used. Some display several indexes at regular distance from each other which enable the user to know the motor absolute position (multi-index encoder) and others display only one (mono-index encoder). Finally some encoders have no indexes, and homing has to be done against a fix **mechanical end stop** or against a **limit-switch** or **home switch**. The homing procedure has a double aim: first, to determine the motor absolute position, and secondly, to find a constant phase shift adjustment value for each system set up.

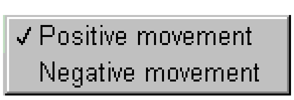
- You will have to select, with the black arrows, the correct options in three scroll menus on the window:

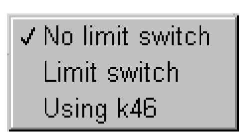


- Choose against which kind of reference should be performed the homing (depends on your hardware):



- Choose the movement direction for the homing and say if a limit switch is present or not:





- Four parameters (K40, 41, 42, 43) automatically defined by ETT are also displayed.
  The parameters values depend on both the encoder and the homing mode you have selected.
  It is possible to modify K41, 42, 43. **Never modify these values unless you are an experienced user!**

- Click on the ***NEXT*** button and the next window will appear.

8.3.6 Mass or inertia

The controller needs the mass (linear system) or the inertia (rotary system) value to correctly set up its regulator. The **whole moving mass** should be included in this value (not only the motor). Take the **mechanical frictions** into account to give accurate values to the controller! **With a wrong value your system will not work correctly!**

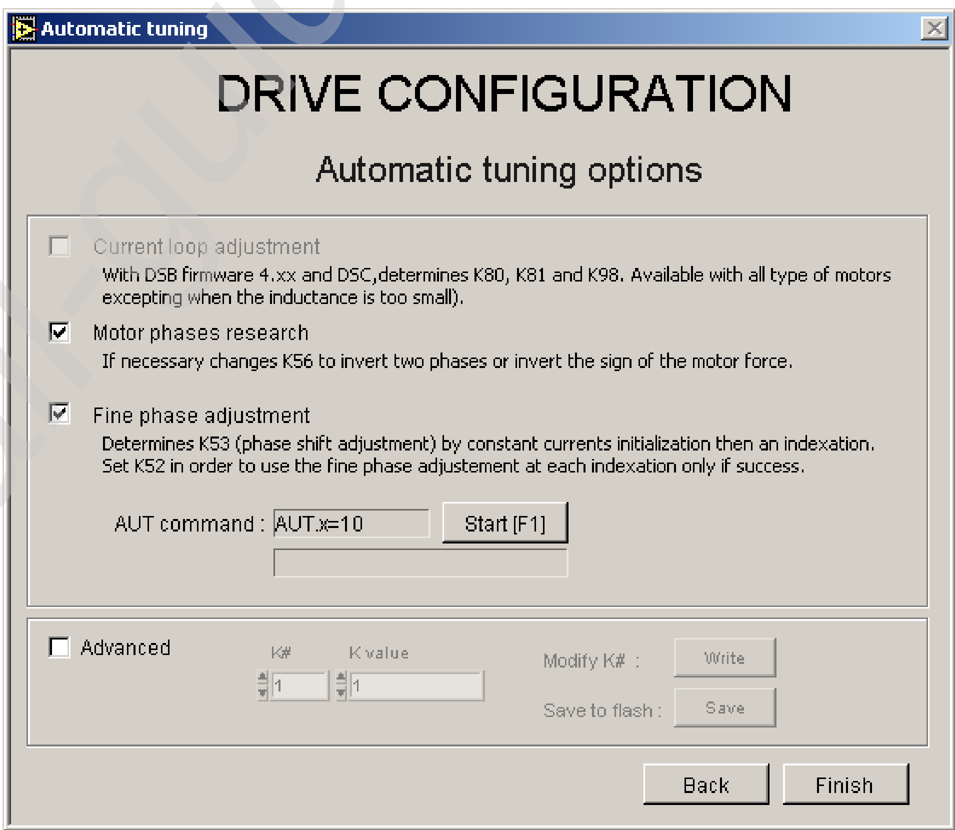
- Click in the dedicated field and enter the value (here: *500.00 E -3 [kg], we have a linear system)*

- Click on the ***NEXT*** button, wait a few seconds until the next window appears.

8.3.7 Automatic tuning options

These options are displayed for your information only (***CURRENT LOOP ADJUSTMENT, MOTOR PHASES RESEARCH, FINE PHASE ADJUSTMENT****)*, leave them set by default, unless you are an expert.

Note: The *Current loop adjustment* button is deactivated only when an ironless motor is used (low inductance).



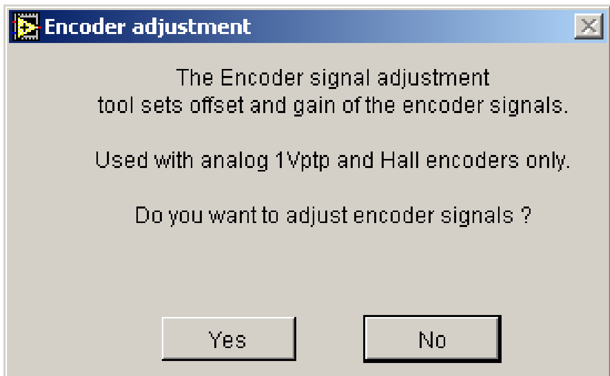
- Do not click on the ***advanced*** button unless you are an experienced user!.

- Put the motor in the middle of the stroke to avoid problems and click on ***START***. The message ***Automatic tuning proceeding!*** appears. Wait until the next window appears (in case of problem, click on ***Abort***).

8.3.7.1 *Tuning successful*

The information window above appears if the automatic tuning was successful.

- Click on ***OK*** to go back to the ***Automatic tuning options*** window and then on ***Finish***. The following windows appears:

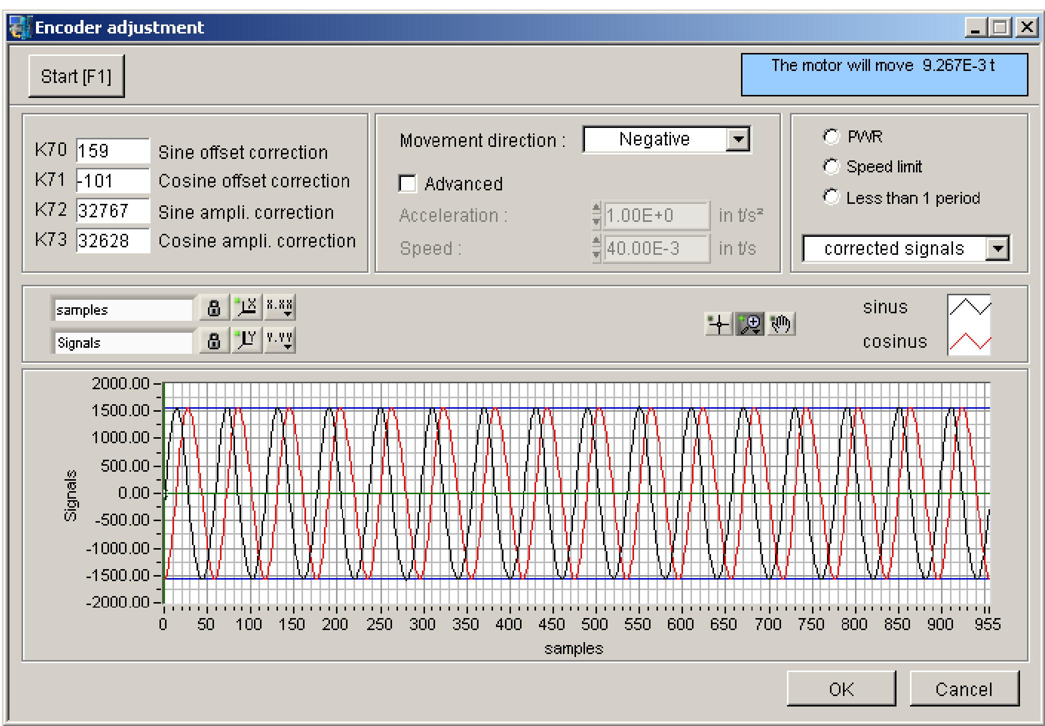


1° If you do not want to adjust the encoder's signal:

- Click on ***No***. The parameters are saved in the controller and the setting process is finished.

2° If you want to adjust the encoder's signal:

- Click on ***Yes.*** This function is used to correct the analog encoder's amplitude and the offset errors.



- Click on ***Start*** to automatically adjust the value of parameters K70 to K73 and then click on ***OK.***

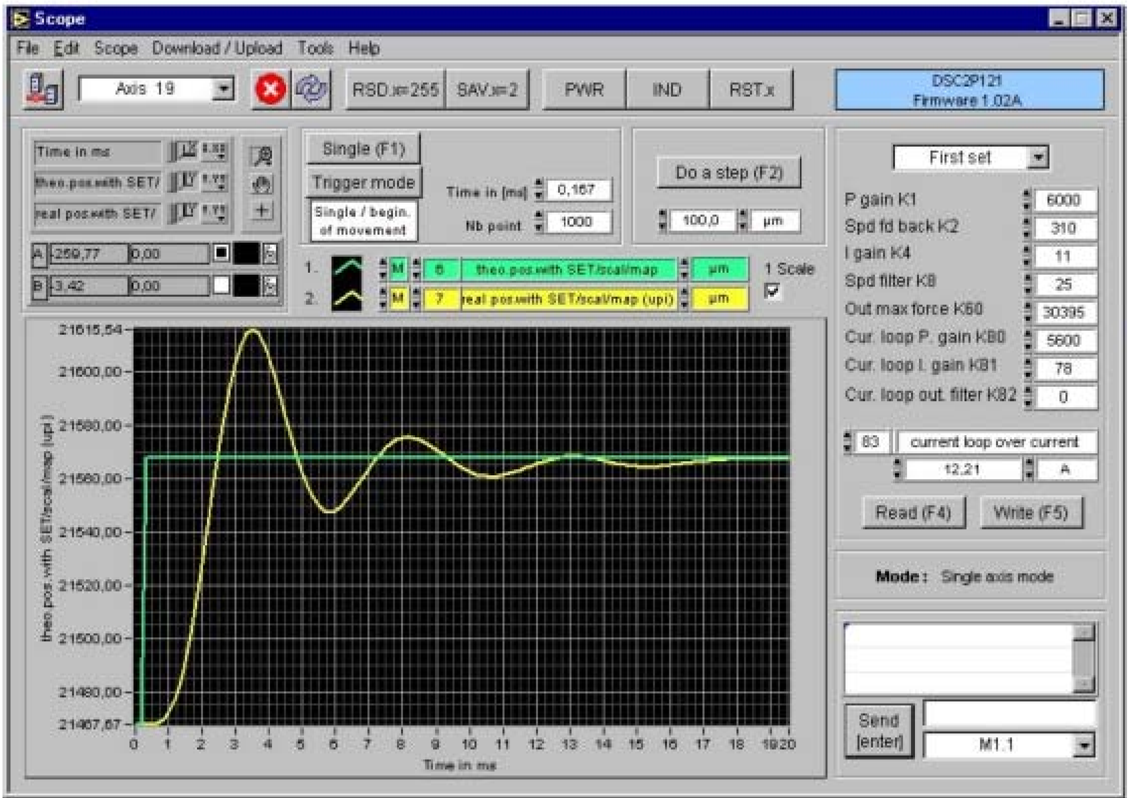
8.3.7.2 *Tuning not successful*

If a problem has occurred during the automatic tuning, the following window appears:

Note: The AUT command is necessary to automatically calculate the regulators parameters and perform the phases adjustment (currents), (read this **'Operation & Software Manual'** for more information).

- Read the window and click on ***OK*** to go back to the ***Automatic tuning options*** window.
- Check if no external perturbation has disturbed the tuning, and try to click again on ***START***
- If the tuning is still not successful, an **experienced user** should use the ***Scope Window*** (§8.4) to modify the settings until it is possible to successfully perform the tuning. Click on ***End*** (all options like CURRENT LOOP ADJUSTMENT, MOTOR PHASE ADJUSTMENT, FINE PHASE ADJUSTMENT are not performed yet).
- Wait a few seconds and click on  in the ***Drive Setting*** window to quit the tool.

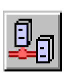
8.4 Scope tool



The ***Scope*** tool is used as a digital oscilloscope to set the controller's regulator parameters and to monitor their influence on the motor's movements. It also includes a terminal (to send commands and read/write the controller parameters). An ***Online Help*** window will appear if you click on ***Help*** in the menu bar. Point on the items in the window to get help.

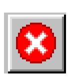
8.4.1 The icons bar

Below the menu bar, several buttons with icons or commands have their functions described below:



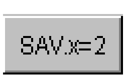
Open the ***Connection Chooser*** window (refer to §8.1).

Reset completely the **EBL2 communication** between the controller(s) and the host (PC with ETT, e.g.). It refreshes the **trigger** and the oscilloscope's **display**.



The **'emergency stop'** button (similar to HLO command) stops the progression of the user's sequence and switches off the power in the motor phases. It is used in case of emergency (unexpected motor noise, e.g.). The **Esc** key on your keyboard has the same effect.

The **RSD.x=255** button resets the controller (similar to RSD.x=255 command).



The **SAV.x=2** button saves in the flash the registers K, X, E, R, F, as well as the axis number (similar to SAV.x=2 command).

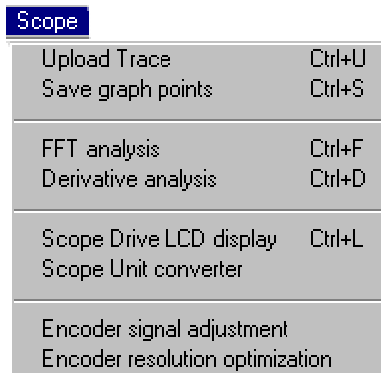
The **PWR** button initializes the controller and sends power into the motor phases (controller in POWER ON mode), or switches off the power in the motor (controller in DSC2 READY mode for example). Similar to PWR command.

The **IND** button starts the motor's homing process (reference marks search). Similar to IND command.

The **RST.x** button resets the errors, (except **HARDWARE OVER CURRENT** and errors which loose the position, like **ENCODER POS LOST**), the LED lights green and the LCD screen displays for example **DSC2 READY**, as long as the cause which has produced the error is not there any more. The motor position is kept and the instruction PWR.[axis]=1 is sufficient to reset the motor under control (similar to RST.x command).

8.4.2 Scope menu

The Scope menu allows the user to have access to several useful tools.



8.4.2.1 Upload Trace tool

This function allows the user to display the last acquisition done with the scope.

8.4.2.2 Save graph points tool

Thanks to this function, the user can save in .txt file all the points of a plot which can be used later for calculations.

8.4.2.3 FFT analysis tool

This tool has exactly the same function as the *Spectrum* tool described in §2.6.5 of the **'ETEL Tools Setup Software Manual'**.

8.4.2.4 Derivative analysis

This tool enables the user to obtain the 1st, 2nd and 3rd order derivative of a plot. For example, with the plot of a position, the user can display the corresponding speed, acceleration and jerk.

8.4.2.5 Scope Drive LCD display tool

This tool shows, on a small window, exactly the same information which appear on the display of the controller.

8.4.2.6 Scope Unit converter tool

This tool has exactly the same function as the *Unit Converter* tool described in §8.7. However, it is available only for the axis number displayed on the top left corner of the scope.

8.4.2.7 Encoder signal adjustment tool

This tool allows the user to modify the adjustment of the encoder's signals (offset and amplitude). The same window already appeared in the controller setting (refer to §8.3.7.1), but this tool enables the user to have access to these parameters (K70, K71, K72, K73) without doing the whole setting again.

8.4.2.8 Encoder resolution optimization tool

Thanks to this tool, the user can set the calculation limits of the controller (parameter K50 and K77) which determine the maximal stroke, speed and acceleration of the system plugged on the controller.

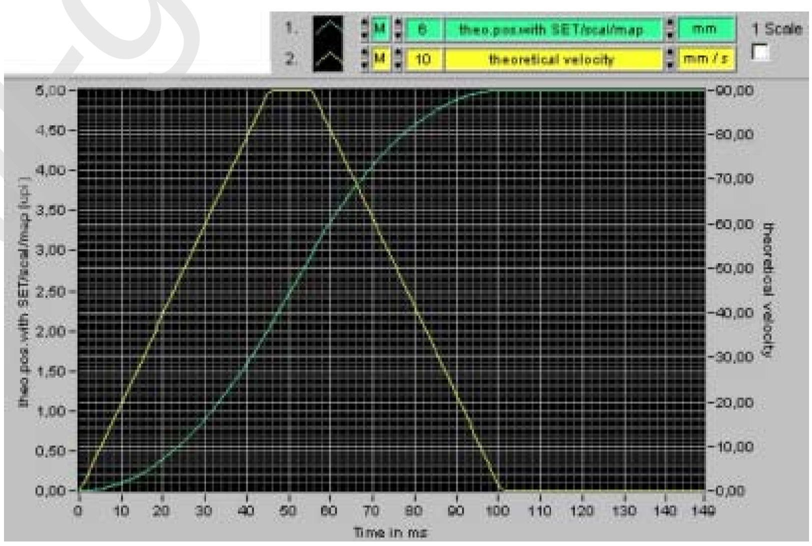
8.4.3 Digital oscilloscope

The two-channel digital oscilloscope allows the user to view the value versus time of any M, K or X register (M = monitoring, K = parameter, X = user variable). The register's type (ex: *M*), number (ex: *6*), and name (ex: *theoretical position with SET/scal/map*) and unit (ex: μm) are also displayed.

The *Single scale* display is selected by default:

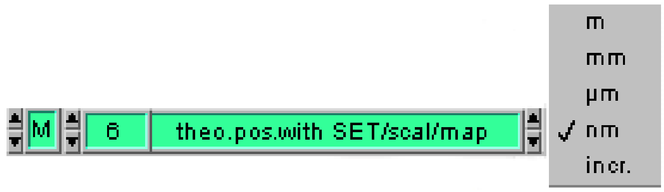
To measure the two registers displayed, it is possible to have a dual scale by unticking the box above.

In this window, the *Dual scale* mode is selected. The 2nd scale is on the right side of the Scope:



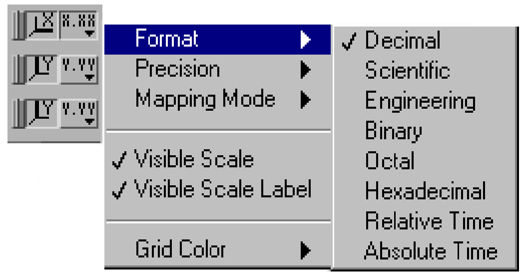
8.4.3.1 Units change

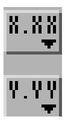
It is possible to change the units of any register displayed on the scope. Just click on the unit in the dedicated field and a menu will appear, enabling you to select a new unit (For a position, for ex: *m / mm / µm / nm / incr.*):



8.4.3.2 Scale format change

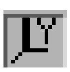
In the scope's upper left corner, some buttons open scroll menus to change both the *X* axis and *Y* axis scale's *Format*, *Precision*, *Mapping Mode*, and Grid color:





The scale *Format* buttons give you run-time control over the format of the X and Y scale markers respectively.

By pressing the *X autoscale* button, ETT autoscales the X data of the graph.



By pressing the *Y autoscale* button, ETT autoscales the Y data of the graph.

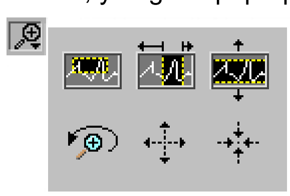
If you want the graph to autoscale both the X axis and Y axis continuously, click on the *lock switch* to lock the autoscaling mode.

8.4.3.3 Zoom and panning tools

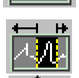
Two buttons are used to control the operating mode for the graph:

By pressing on the panning tool, you switch to a mode in which you can scroll the visible data by clicking and dragging sections of the graph.

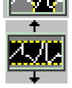
By pressing on the zoom tool, you get a pop-up menu you can choose the method of zooming:



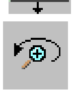
Zoom by rectangle: Zoom in on a section of the graph by dragging a selection rectangle around that section.



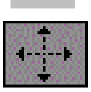
Zoom by rectangle on a restricted area of the X data (the Y scale remains unchanged).



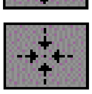
Zoom by rectangle on a restricted area of the Y data (the X scale remains unchanged).



Undo last zoom. Resets the graph to its previous setting.



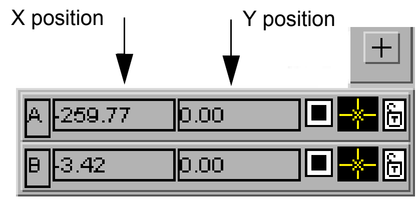
Zoom in on a point. If you hold down the mouse on a specific point, the graph continuously zooms in until you release the mouse button.



Zoom out from a point. If you hold down the mouse on a specific point, the graph continuously zooms out until you release the mouse button.

8.4.3.4 *Cursors*

Two cursors are available; by default, their position is (0;0) and their color is yellow for cursor A and blue for cursor B. They may be used to measure the positions of the signals monitored on the Scope.



Cursor modes:

Click on the little locker and a menu will appear to change the mode for each cursor:

Free, the cursor may be placed anywhere on the Scope display.

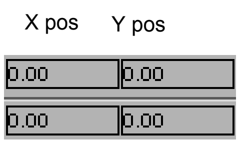
Snap to point, the cursor may be placed only on the signals traces (equally on signal 1. or on signal 2.)

Lock to plot 1, the cursor may be placed only on the signal trace 1.

Lock to plot 2, the cursor may be placed only on the signal trace 2.

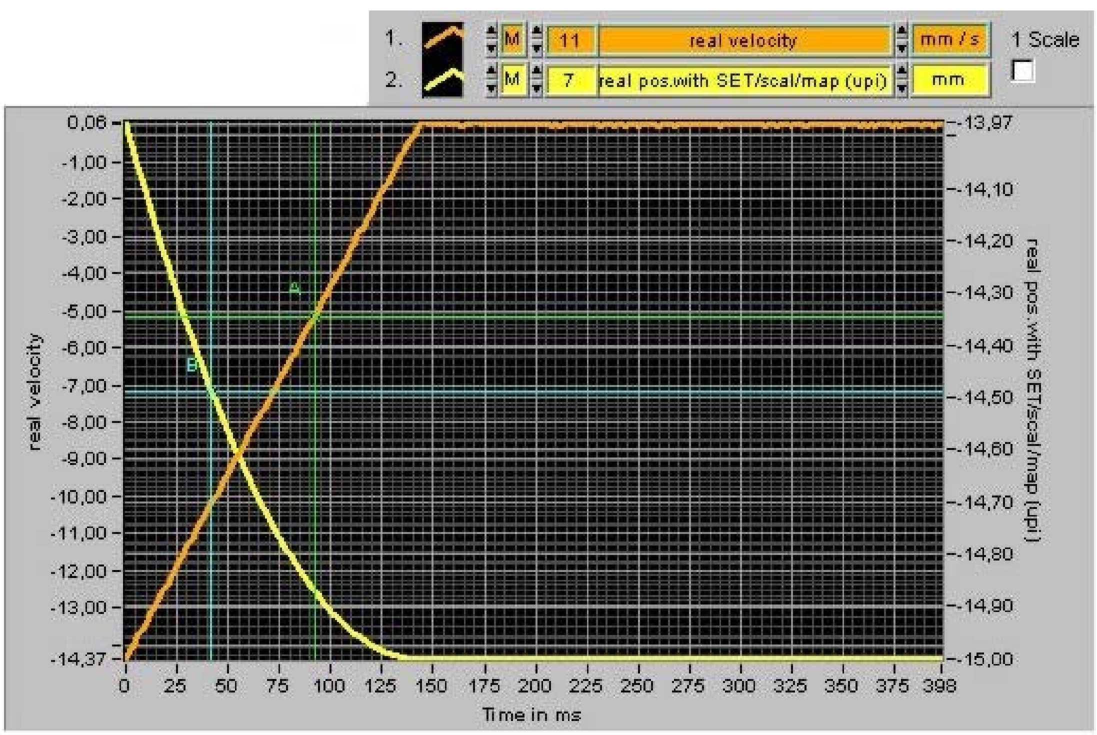
Move a cursor:

The "cross" button is used to move the cursors on the Scope's display:

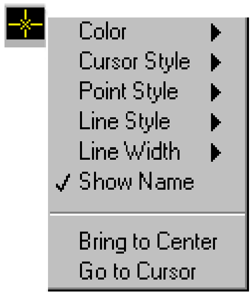


Another way to move a cursor is to type its coordinates directly in the position fields. In ***Snap to point*** and in ***Lock to plot*** modes, the cursor goes automatically to the trace's closest point (of the entered value).

In the window below, the two cursors are in ***Lock to plot*** mode. *Cursor A* is on monitoring M11 (signal trace 1, **Lock to plot 1.**), *Cursor B* is on monitoring M7 (signal trace 2, **Lock to plot 2.**):



Modify the cursors:



To change the cursor's aspect, click on the yellow target.

Note: To **suppress the cursors** from the display, select Color in the scroll menu, and click on the little "T" (like transparent) in the colors palette bottom left corner.

8.4.4 Scope's Trigger

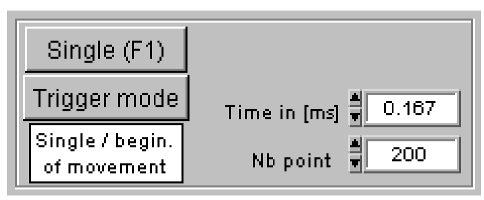
This window includes also a ***Trigger mode*** button. The trigger settings enable the user to completely define the scope's acquisition parameters and acquisition mode.

8.4.4.1 Total acquisition time

- Type in the '*Nb point'* field the number of points you want to be sampled (max. 1000).
- Type in the '*Time'* field the time between sampled points.

The total acquisition time displayed on the scope's *X* axis is:

X = *Time* x 1E-3 x *Nb point*



- If we take the example of the window displayed above, we have:

0.167 x 1E-3 x *200* = 33.4 milliseconds displayed on the *X* axis.

8.4.4.2 Change trigger mode

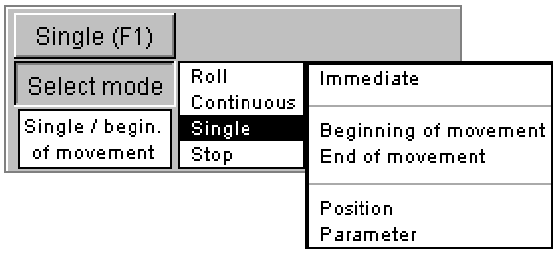
- Click on the *Trigger mode* button: *Select mode* appears on the button with a pop-up menu on its right side. It is now possible to set the trigger to four different acquisition modes: *Roll, Continuous, Single* or *Stop* (refer to the next picture).
- The *Roll* mode lets the scope run all the time.
- The *Continuous* and *Single* trigger modes have a 2nd level pop-up menu to set the acquisition's start.
- *Stop* will stop the acquisition on the last image displayed. If the trigger waits for a signal which is not coming, use the *Reset* button  to release the trigger and modify the settings.

While the button is pressed in *Select mode*, all the Scope functionalities are disabled. You must click again to release the button in *Trigger mode* to activate the scope again.

8.4.4.3 Simple trigger

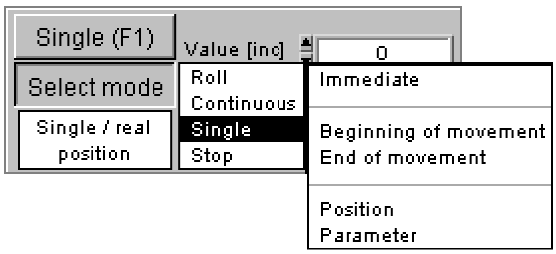
- The second level of the pop-up menu enables the acquisition start setup of the *Continuous* and *Single* triggers.
- The first three triggers are simple:
  The acquisition start can be: *Immediate*, at a *Beginning of movement,* or at an *End of movement*.

Note: Reference values (theoretical) are taken into account by the trigger (not the real measured values):

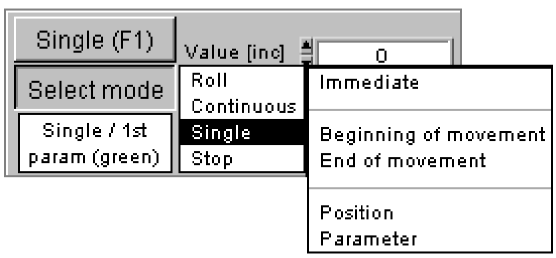


8.4.4.4 *Complex trigger*

- The last two triggers, *Position and Parameter*, are more complex.
- The second pop-up menu shows that the motor's *Position* can define the acquisition start.
- Type the position in the '*Value [incr]*' field.



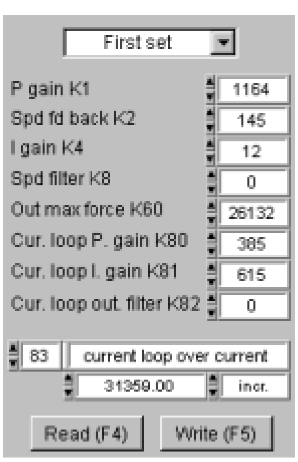
- The acquisition can also start when a *Parameter* displayed on the scope reaches a defined level.
- The 1st parameter (*green)* is set if you click on *Parameter* and if you type the *Value* you want in the dedicated '*Value [incr]'* field (can be read with monitoring M7 [incr]).

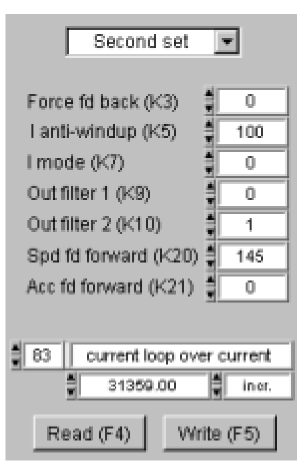


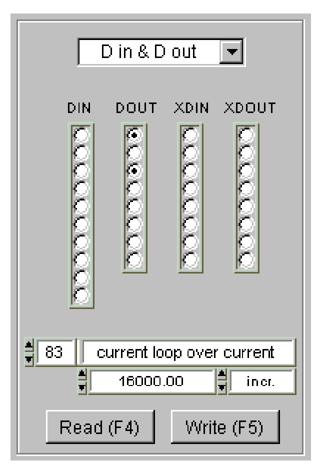
8.4.5 Parameters

8.4.5.1 *On the monitor window*

Fifteen parameters that you may use to set up the controller's regulation can be displayed in the '*Parameter*' field. They will influence either the **position state regulator**, the **current reference generator** or the **PI current regulator** response. These parameters are listed in two tables and DIN/DOUT status in a third one:





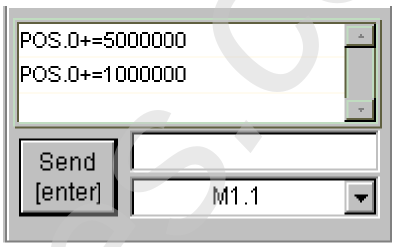


Scroll the menu to choose between ***First set***, ***Second set***, or ***Din&Dout*** table to be displayed. It is possible to type new parameter values in the dedicated fields.
Click on *Write F5* (or press F5 on your keyboard) to accept parameter changes.
Click on *Read F4* (or press F4 on your keyboard) to read the actual parameter values in the controller.
In the text field, you can work on any other parameter (here: *K83*). Only experts use this field; it has priority on all other fields.

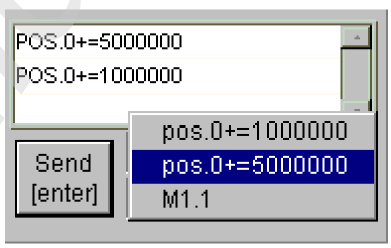
Remark: Refer to this **'Operation & Software Manual'** if you want to understand more about these parameters. Reading these paragraphs will also help you to understand which possible numbers can be entered in the dedicated fields. These numbers will set the parameters to the desired values.

8.4.6 Commands

The Scope window also includes, at the bottom right corner, a small terminal to send commands to the controller. Firstly, type the command in the dedicated field and then press on the *Send [enter]* button (or type Enter on your keyboard) to send the command to the controller.



If you want to enter one of the commands already sent, click on the black arrow and you will be able to select one of the 11 last commands sent.



8.4.7 Step movement

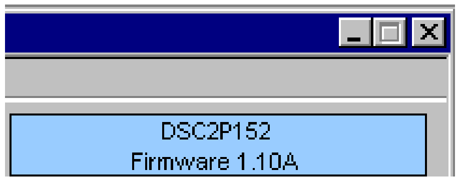
In the upper part of the ***Scope*** tool window (under the icons bar), the *'Step'* field is used to provide a step movement command to the motor (back and forth step). This movement is used by experienced users to proceed to the regulator's fine setting. Fine setting is not explained in this manual.

To perform a step movement, the controller must be in 'Power On' mode (status on the LCD). Then, select the movement size in the dedicated field (typically, *1.0 mm*), and press on the *Do a step (F2)* button (or type F2 on your keyboard) to perform the step.

8.4.8 System identification

In the upper right corner of the Scope window, several details are displayed:

- Controller type. Here: *DSC2P152*.
- Firmware version installed in the controller. Here: *Firmware 1.10A*.

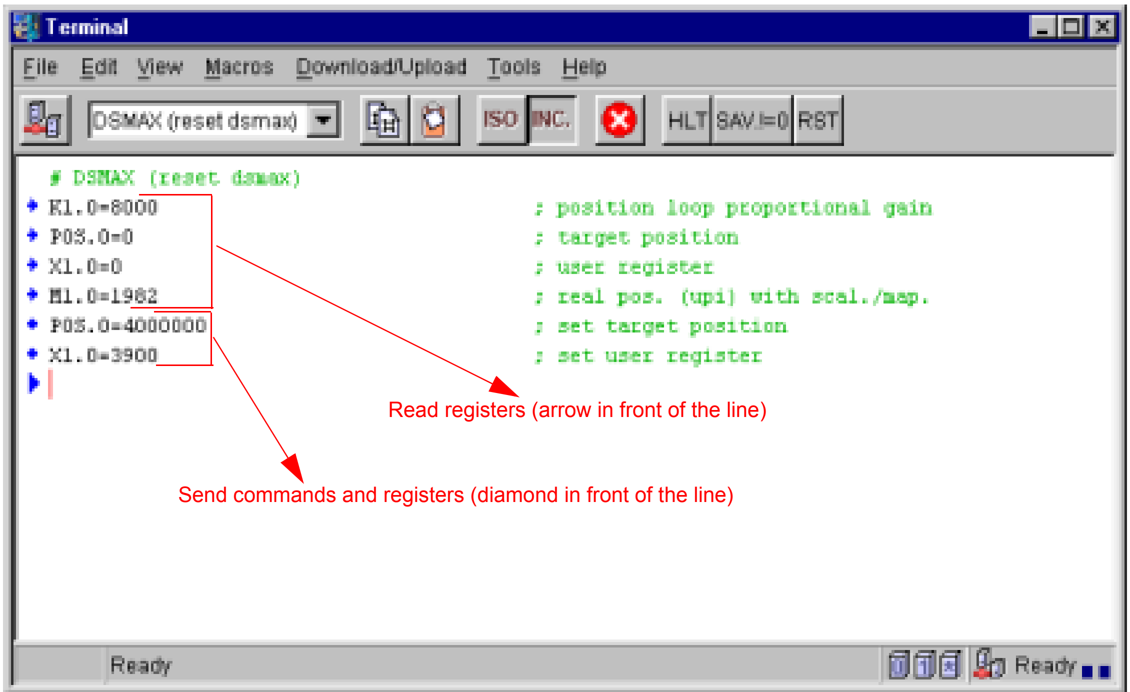


In the upper left corner of the Scope window, the following details are displayed:

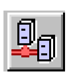
- Controller axis number. Here: *Axis 0*.
- Daisy chain (TEB): the A*xis present* scroll menu shows *Axis 0, Axis 1 and *(DSMAX) (Axis 31)*. *Axis 31* is reserved for the DSMAX; the axes number may be from 0 to 30.

8.5 Terminal tool

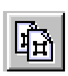
The **Terminal** tool is used to communicate with the controller. It allows the user to read registers values stored in the controller, or to send commands or registers values into the controller. All commands and the registers syntaxes are described in this **'Operation & Software Manual'**.



8.5.1 The icons bar



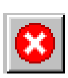
Open the ***Connection Chooser*** window (refer to §8.1).



Copy the selected text on the terminal.

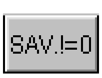
Paste the last text selection on the terminal.

Unit conversion (ISO <> increments) for all values displayed on the terminal.



The **'emergency stop'** button (similar to HLO command) stops the progression of the user's sequence and switches off the power in the motor phases. It is used in case of emergency (unexpected motor noise, e.g.). The **Esc** key on your keyboard has the same effect.

The **HLT.!** stops the progression of the user's sequence and stops the motor with the maximal deceleration possible (similar to HLT.! command).



The **SAV.!=0** button saves in the flash the registers K, X, E, L, S, R, F, as well as the axis number for all axes present (similar to SAV.!=0 command).

The **RST.!** button reset the errors of all the axes present, (except **HARDWARE OVER CURRENT** and errors which lose the position, like **ENCODER POS LOST**), the LED lights green and the LCD screen displays for example **DSC2P READY**, as long as the cause which has produced the error is not there any more. The motor position is kept and the instruction PWR.[axis]=1 is sufficient to reset the motor under control (similar to RST.! command).

8.5.2 Read / send registers and commands to the controller

It is possible to read / send registers values and to send commands:

Read Syntax: **<register_name1>[:<depth1>].<axis>[,<register_name2>[:<depth2>].<axis>]**, **Enter** (on keyboard). The register's value appears on the same line (arrow in front of the line).

Send Syntax: **<register_name>[:<depth>].<axis>[<operator>] = <p1>**, **Enter** (on keyboard)
Syntax: **<cmd_name>.<axis>[<operator>][=<p1>][,<p2>]**, **Enter**
The register's new value (or command) appears on the same line (diamond in front of the line).

After each validation (**Enter** on keyboard), a comment about the function appears in green on the same line.

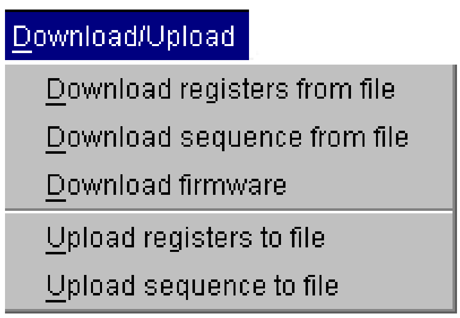
Note: The fields put in 'square brackets' (like: **[:<depth>]**) are **optional.** (refer to §6. for more explanations about the syntax described above).

8.5.3 Download sequence / registers into the controller

8.5.3.1 *Download sequence*

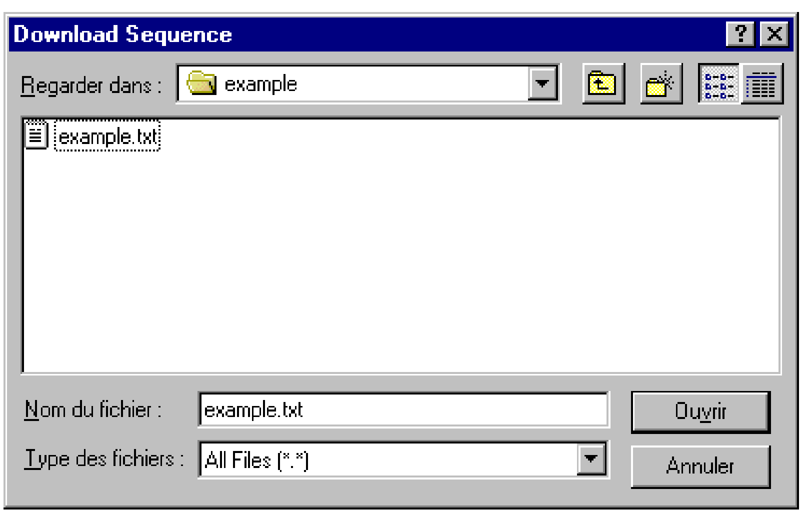
When you write a 'user sequence', you save it as a text (.txt) file on your PC (refer to §6. and §14. for the commands syntax and for programming). Then, you will have to copy it into the controller, as follows:

- Click on ***Download/Upload*** then click on ***Download sequence from file***, in the **Terminal** menu bar.

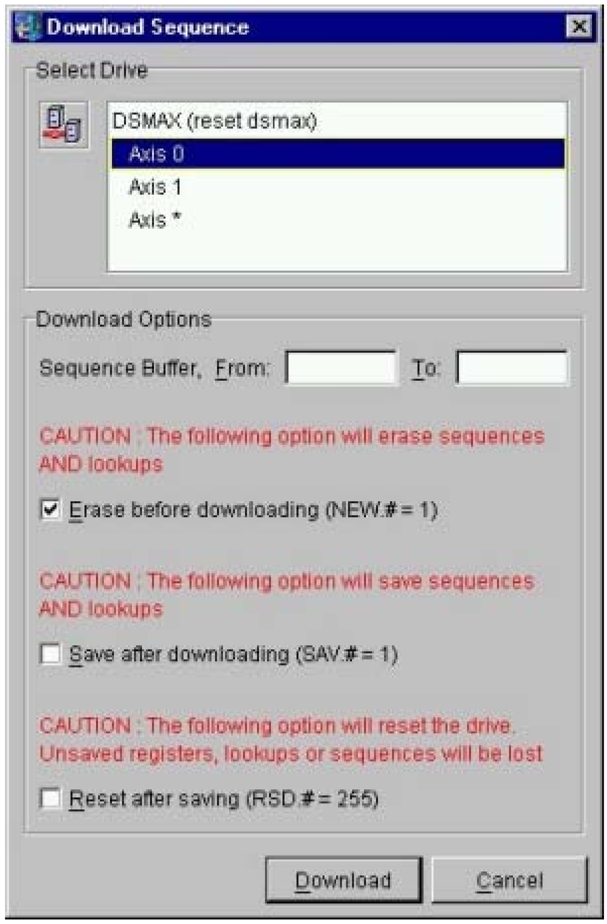


- The ***Download Sequence*** window appears. Browse your PC to find the file you want. For sequences, use ***.seq** files, for K parameters ***.par** files; ***.txt** files is also possible for both.

- Select the file (here, the file example.txt in the example directory), and click on the 'ouvrir' (Open) button.

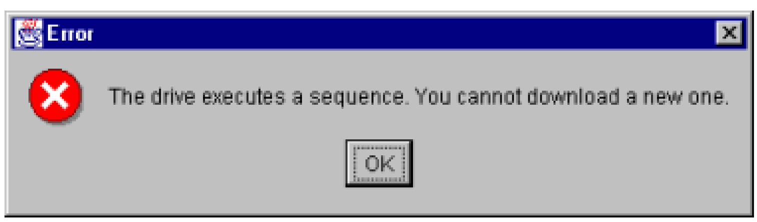


- The previous window disappears and a new ***Download Sequence*** window appears.



- Select an axis and click on the ***Download*** button. Wait until the sequence is completely downloaded. (The **Axis *** is the DSMAX).

Remark: If a sequence is already running in the controller, the ***Error*** window below appears:



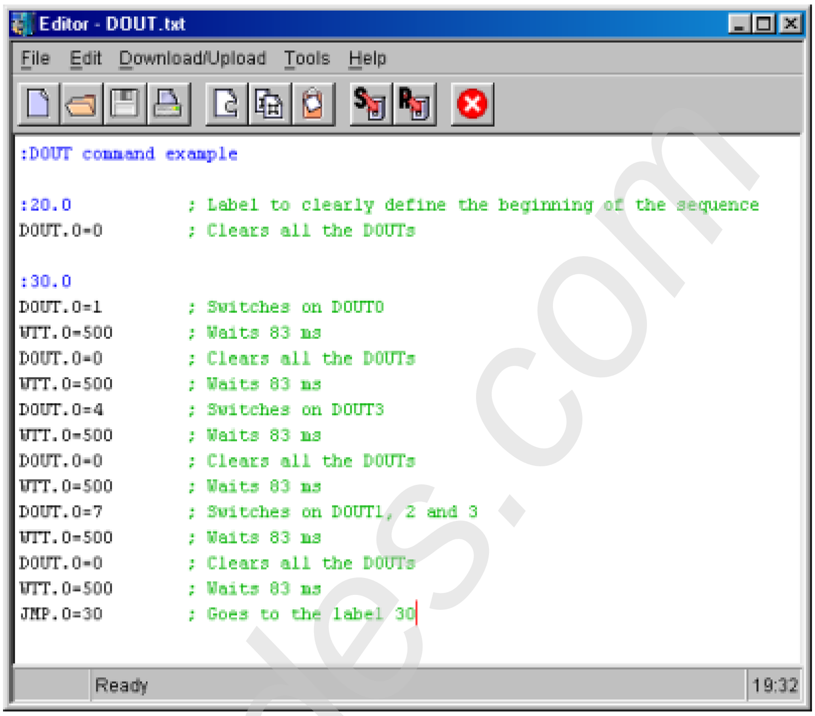
8.5.3.2 Download registers

For registers, follow the same procedure as for a sequence (refer to §8.5.3.1).

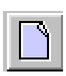
Remark: The window's name will be ***Download Registers***. The ***Error*** window will never appear.

8.6 Editor tool

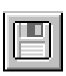
The **Editor** tool is used to communicate with the controller. It allows the user to create (write), open or modify the sequences / registers (*.par, *.seq, or *.txt files) and to download them into the controller. It allows the user to also upload the sequences / registers present in the controller. All commands and the registers syntax are described in this **'Operation & Software Manual'**.



8.6.1 The icons bar

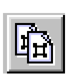

New blank editor window.


Open a text file (.txt) in the editor window.


Save the editor window contents.

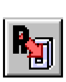

Print the editor window contents.

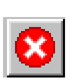

Cut the selected text on the editor.


Copy the selected text on the editor.


Paste the last text selection on the editor.


Download the Sequence present on the editor window into the controller.


Download the Registers present on the editor window into the controller.

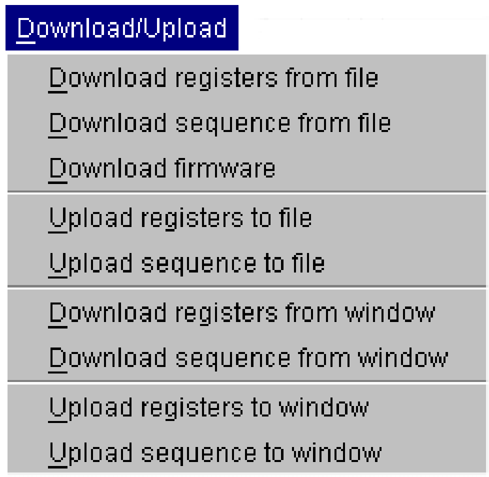

The **'emergency stop'** button (similar to HLO command) stops the progression of the user's sequence and switches off the power in the motor phases. It is used in case of emergency (unexpected motor noise, e.g.). The **Esc** key on your keyboard has the same effect.

8.6.2 Download (create, open, modify) sequences / registers

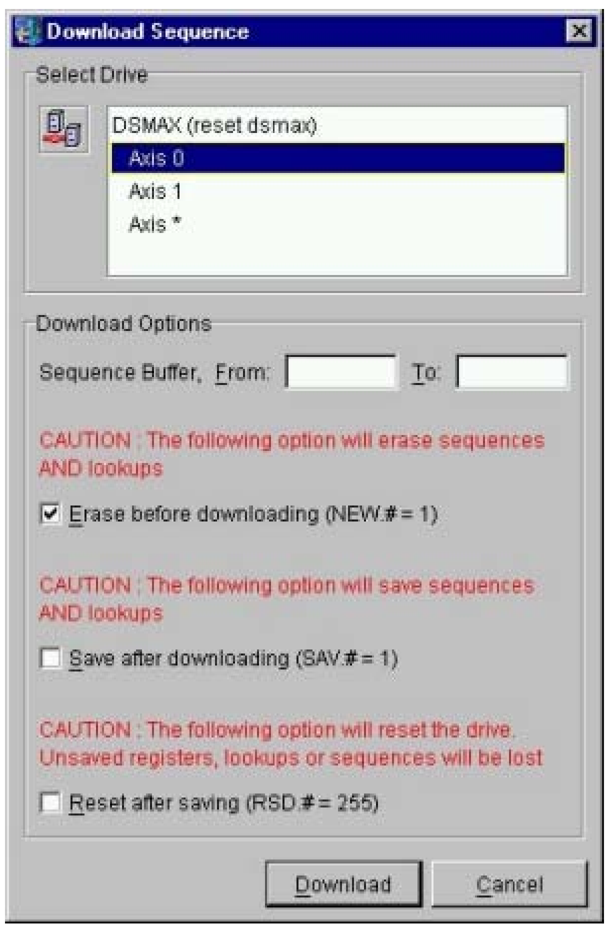
It is possible to directly type sequence lines or registers in the editor window. It is also possible to open sequences and registers saved on your PC (*.par, *.seq, or *.txt files). The sequence (including or not including registers values), or the registers may only be downloaded into the controller.

8.6.2.1 Download a sequence into the controller

- Click on **Download/Upload,** then on **Download sequence from window**, in the menu bar.

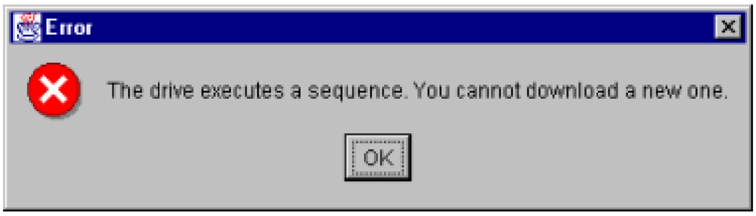


- The ***Download Sequence*** window appears



- Select an axis and click on the ***Download*** button. Wait until the sequence is completely downloaded.

Remark: If a sequence is already running in the controller, the ***Error*** window below appears:



8.6.2.2 Download registers

For registers, follow the same procedure as for a sequence (refer to previous paragraph).

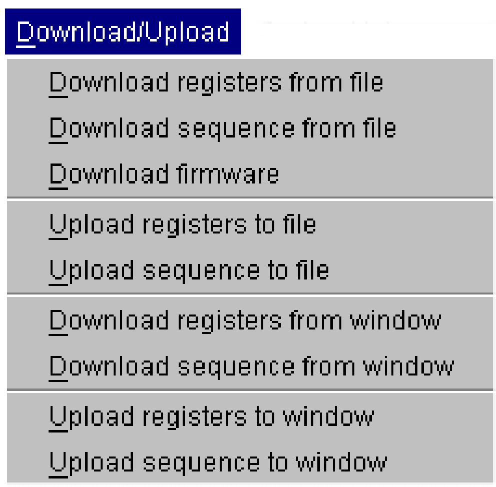
Remark: The windows names will be ***Download Registers***. The ***Error*** window will never appear.

8.6.3 Upload (create, open, modify) sequences / registers

It is also possible to upload the sequence or the registers stored in the controller. When a sequence is uploaded, a (green) comment appears on the same line. When registers are uploaded, a (green) comment appears on the same line for parameters K only (not for all of them).

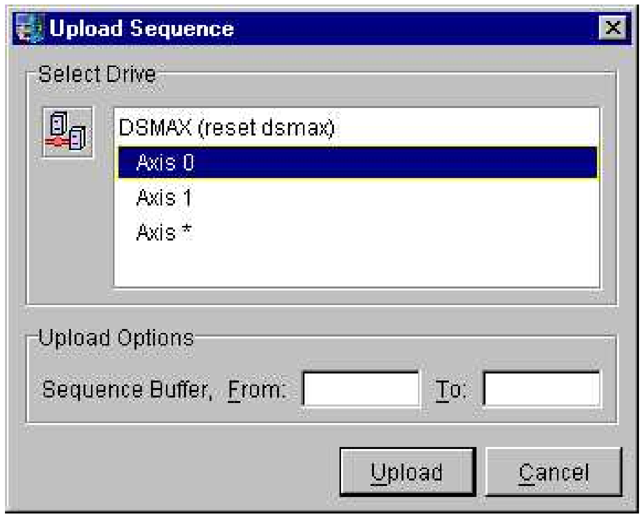
8.6.3.1 Upload a sequence from the controller

- Click on **Download/Upload,** then on **Upload sequence to window**, in the menu bar.



- It is recommended to answer **Yes** to the question below (to have a backup):

- The ***Upload Sequence*** window appears.



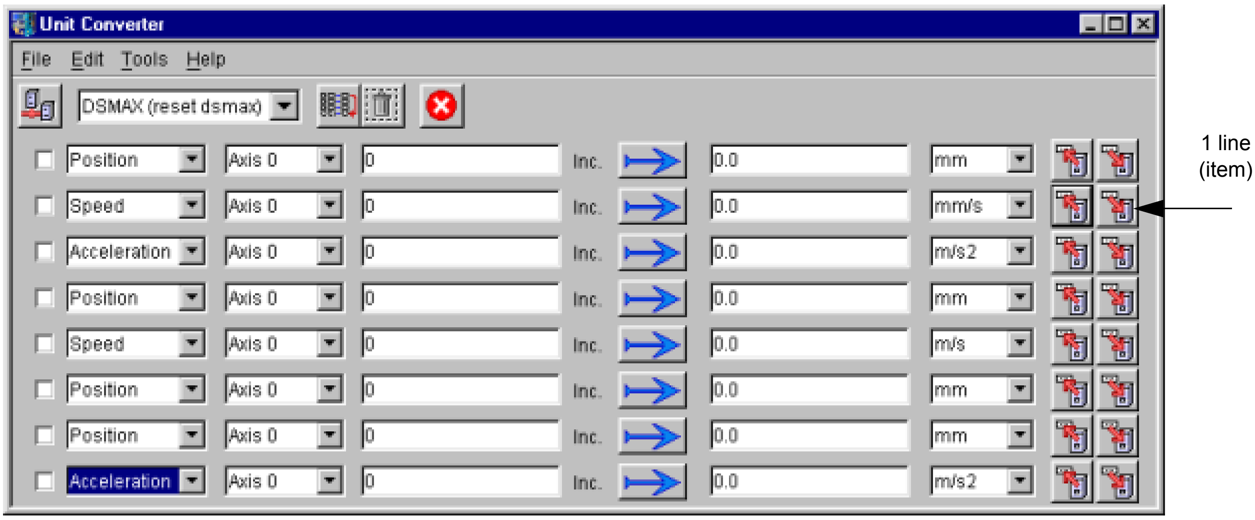
- Select an axis and click on the ***Upload*** button. Wait until the sequence is completely uploaded.

8.6.3.2 Upload registers

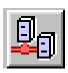
For registers, follow the same procedure as for a sequence (refer to previous paragraph).

8.7 Unit Converter tool

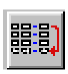
The Unit Converter tool is used to set up / read **position**, **speed** and **acceleration** values in the controllers, in ISO units or increments. It is also possible to read the real **position**, **speed** and **acceleration** values in the controllers. These functions are available for all axes present on the Turbo-ETEL-Bus daisy chain.



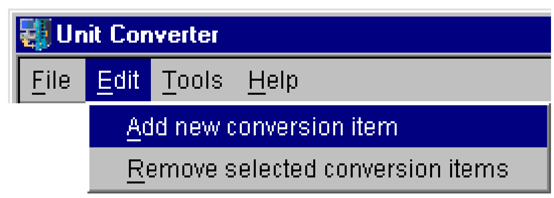
8.7.1 The icons Bar

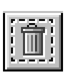


Open the ***Connection Chooser*** window (refer to §8.1).

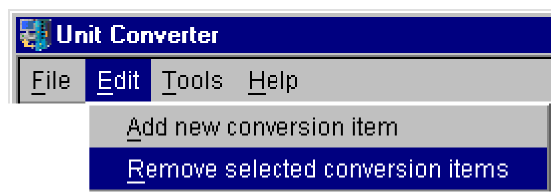


Add a new conversion item (new line).This function (**Add**) may also be realized from the menu:

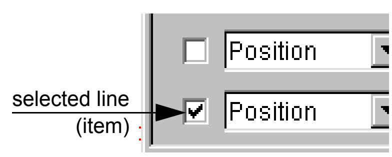


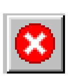


Delete a conversion item (line).This function (**Del**) may also be realized from the menu:



Remark: It is necessary to select an item (line) before deleting it.

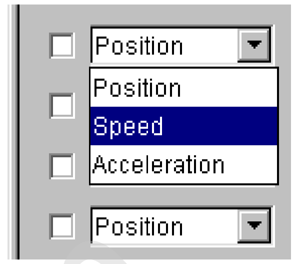


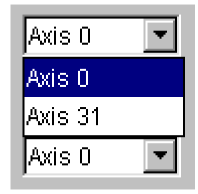


The **'emergency stop'** button (similar to HLO command) stops the progression of the user's sequence and switches off the power in the motor phases. It is used in case of emergency (unexpected motor noise, e.g.). The **Esc** key on your keyboard has the same effect.

8.7.2 Unit conversion

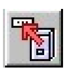
- Choose between *Position*, *Speed* or *Acceleration* in the first scroll menu.
- Select also the *Axis* number in the second scroll menu.



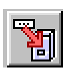


- If required, you may change the ISO unit (*Position*, *Speed* or *Acceleration*).

- With a click on the buttons below, you may upload / download *Position*, *Speed* or *Acceleration*, for each line separately, and convert their values:



Upload directly from the controller (increments) the *Position*, *Speed* or *Acceleration* value to the corresponding line (Inc and ISO). This is an **increment > ISO** value conversion.



Download directly to the controller (increments) the *Position*, *Speed* or *Acceleration* value (inc. and ISO) from the corresponding line. This is an **ISO > increment** value conversion.

Example:

The controller is in ’Power On’ mode. The user wants to **Download** into the controller the following line:

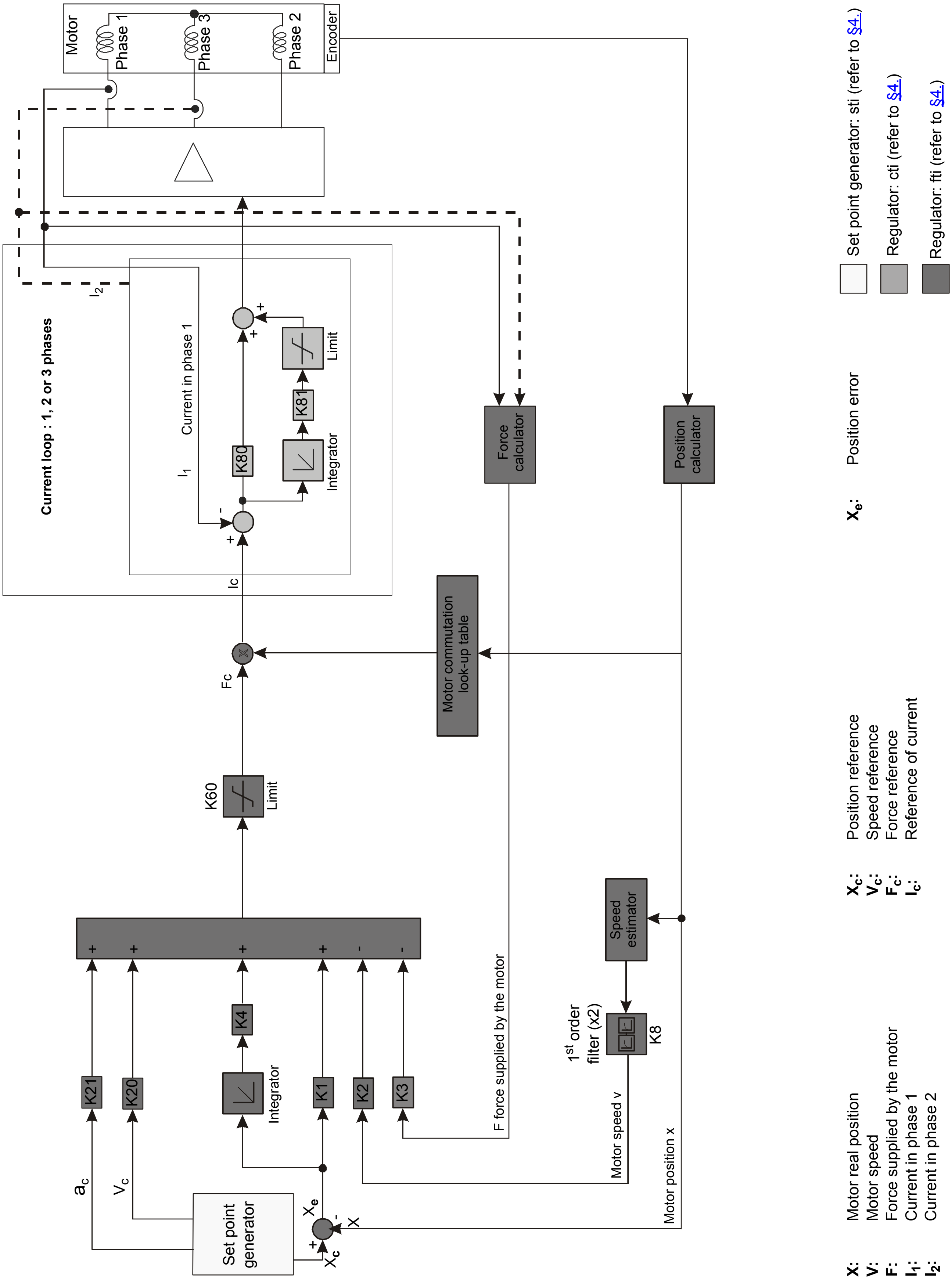
Position / Axis 0 / 65536000 [Inc] / 160 / mm (click on the **Download** button)

This is like sending the following command from the terminal: POS.0 = 65536000 (**Enter**)

Note: The position value sent is always an absolute position!

9. Simplified regulator's principle

9.1 General diagram



9.2 Parameters description

The interaction between the regulator's elements will now be looked at. These explanations are more intuitive than theoretical. They should be used to 'feel' what happens and to help the reader to set up the regulator. The reader is supposed to have basic knowledge in automatic control. For more information, please refer to one of the numerous works on the matter available on the market.
This regulator's description is not complete, it has been simplified for a better understanding. However, all the main parameters listed in this chapter are sufficient to set up the controller for all typical applications.

Remark: The principles described below are generally verified; however the results will strongly depend on the application and on the system's working conditions.

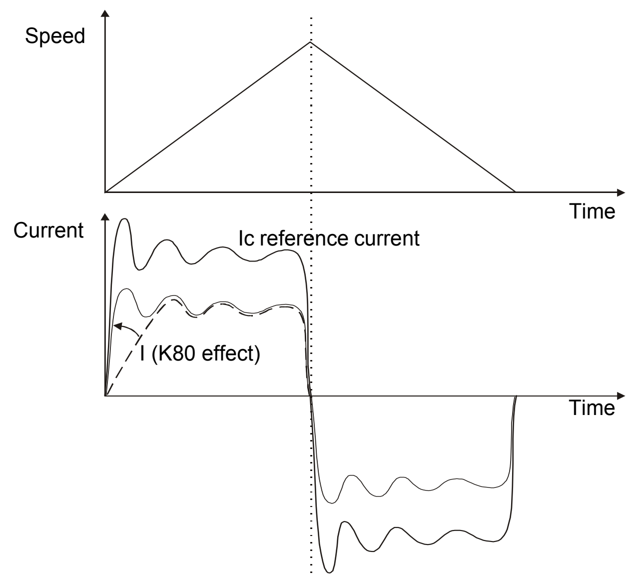
9.2.1 Current regulator

Note: Refer to §13.1.2.1 for the position regulator's K values.

A classical proportional-integral (PI) regulator is used to control the controller's current output (Refer to §13.1.2.2 for the current regulator's K values). The **PI current regulator's** parameters are:

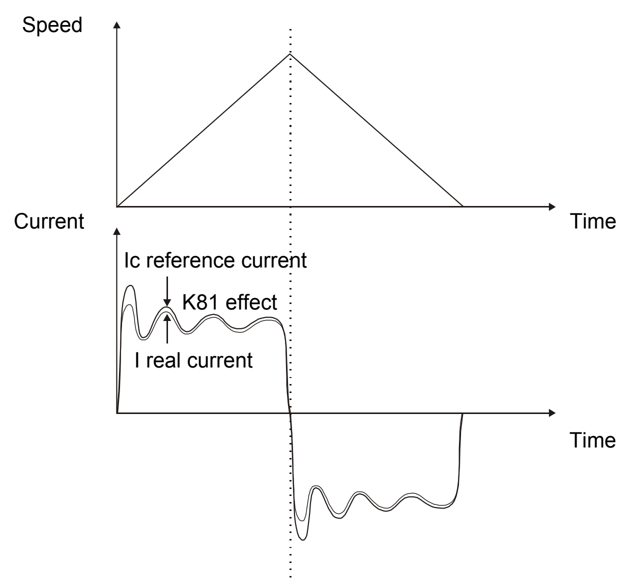
K80: Proportional gain (P).

The main effect of parameter K80 is to make the current response faster.



K81: Integral gain (I).

The main effect of parameter K81 is to suppress the current permanent error.



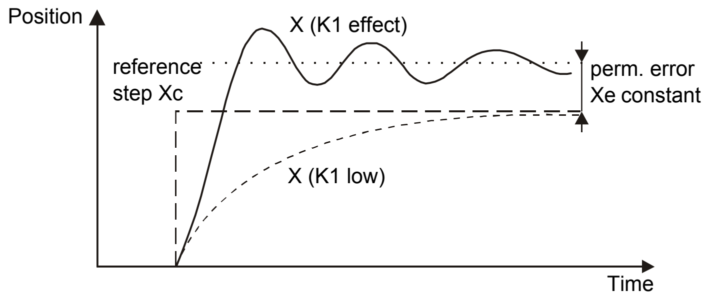
9.2.2 Position regulator

Note: Refer to §13.1.2.1 for the position regulator's K values.

9.2.2.1 PID gains

The position feedback is controlled by a state regulator, which can be approximated by a proportional-integral-derivative (PID) regulator. The **position state regulator's** parameters are:

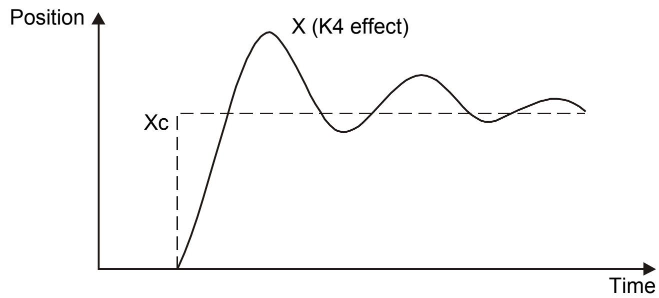
K1: Proportional gain (P).



The main effect of parameter K1 is to make the position response faster, but it may also create an overshoot, oscillations and a permanent error.

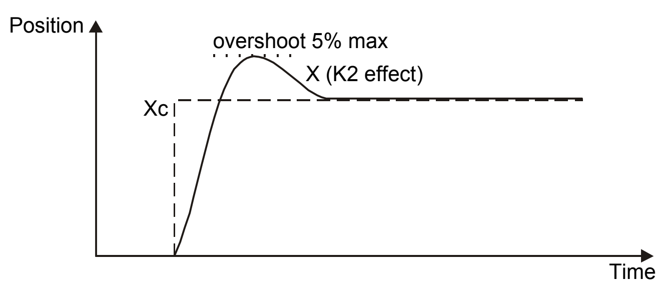
Position error : Xe = Xc - X

K4: Integral gain (I).



The main effect of parameter K4 is to suppress the position permanent error (Xe constant).

K2: Speed feedback gain, or pseudo-derivative gain (D). This is not a real position derivative gain, since the input to *K2* is not directly stemmed from the position error (Xe), but derived from the calculated speed (V).



The main effect of parameter K2 is to reduce (suppress) the overshoot and the low-frequency oscillations.

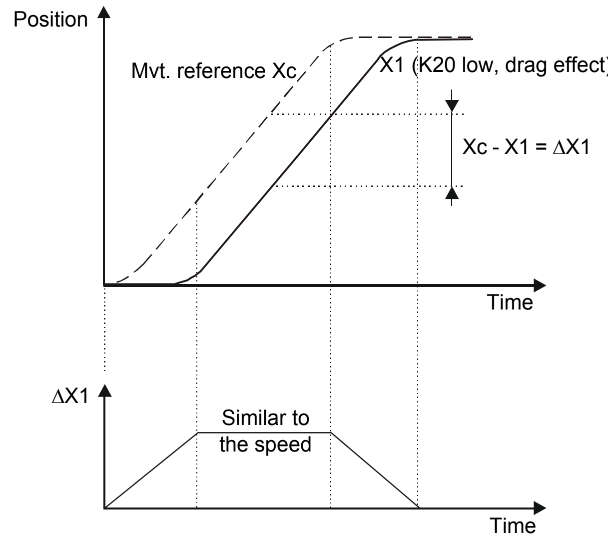
9.2.2.2 *Feedforwards*

During a movement, a permanent error may appear between the position reference and the motor's real position. This drag can be due to mechanical frictions. It is possible to compensate it with the feedforward parameters. These parameters will increase 'a-priori' (with anticipation, without reading the real drag) the speed and acceleration command inputs to the state regulator. The **feedforward** parameters are:

K20: Speed feedforward gain.

The difference between the position reference (Xc) and the real position (X1) is similar to the speed's profile (ΔX1).

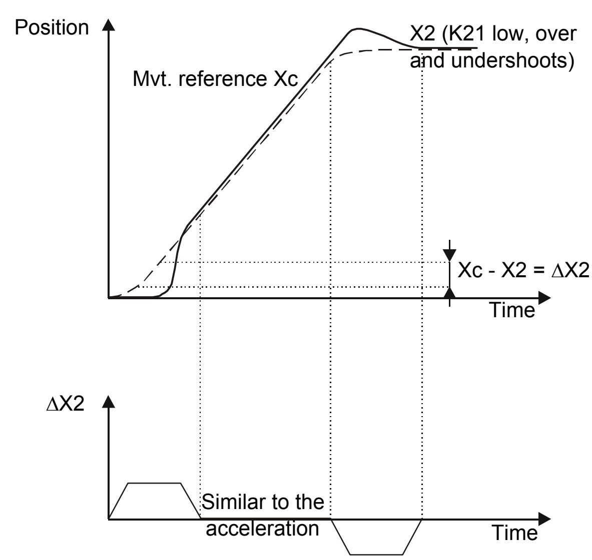
The speed feedforward will compensate the drag's constant part. the value of parameter K20 value should be set equal to the one of parameter K2.



K21: Acceleration feedforward gain.

The difference between the position reference (Xc) and the real position (X2) is similar to the acceleration's profile (ΔX2).

The acceleration feedforward will compensate the undershoot and overshoot remaining after the speed feedforward compensation.



9.2.2.3 *Speed filter*

The speed feedback (input to *K2*) to the state regulator can be filtered. This speed filter is made up of 2 low-pass 1st order filters, with a -20dB/decade slope each. The **state regulator's speed filter** parameter is:

K8: Speed filter (no filter when *K8* = 0).
Should be used when an audible high frequency perturbation occurs on the speed (f > 700 Hz).

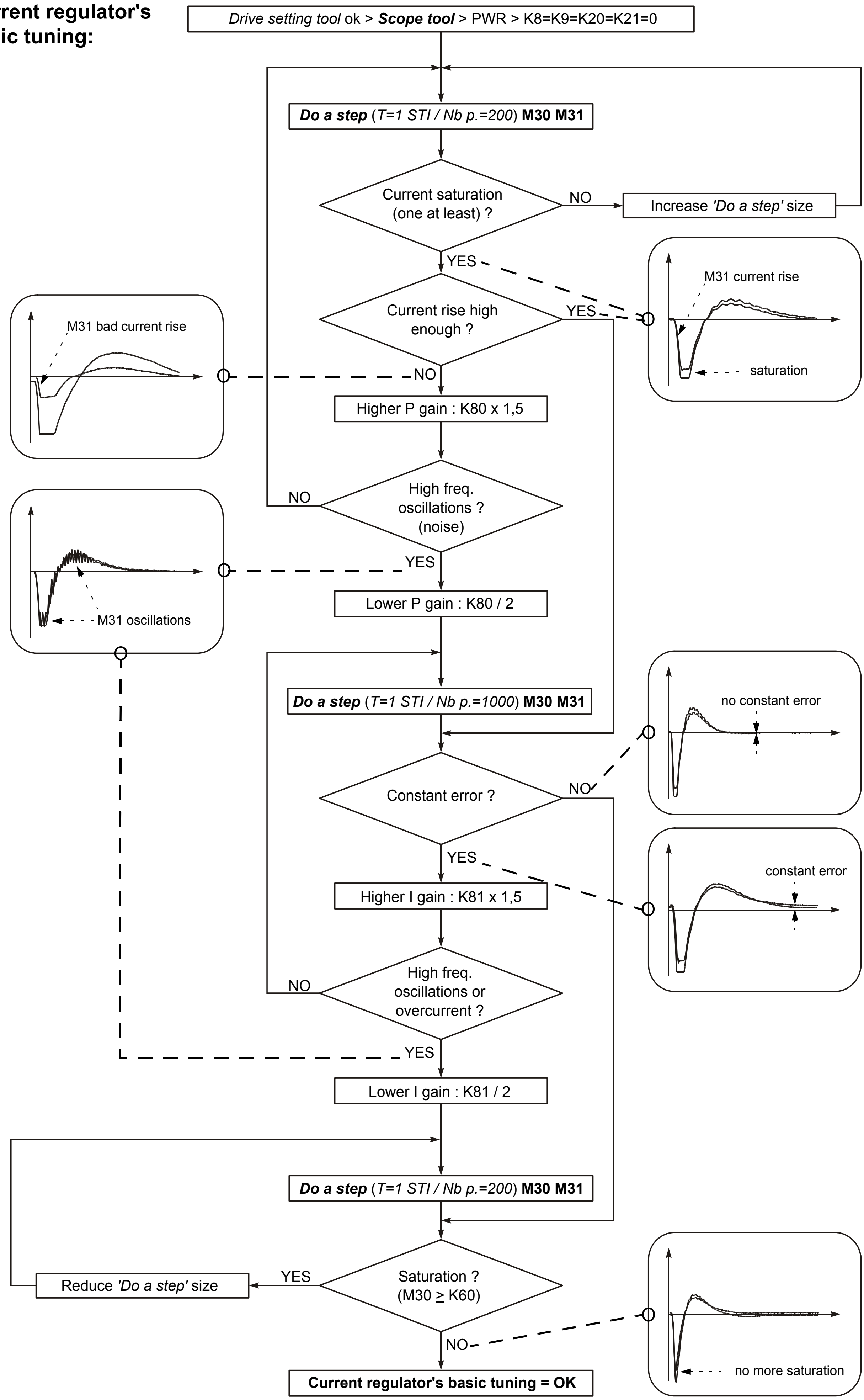
Remark: Advanced users may refer to §13.1 for a more complete regulation loop description.

10. Controller regulators tuning principle

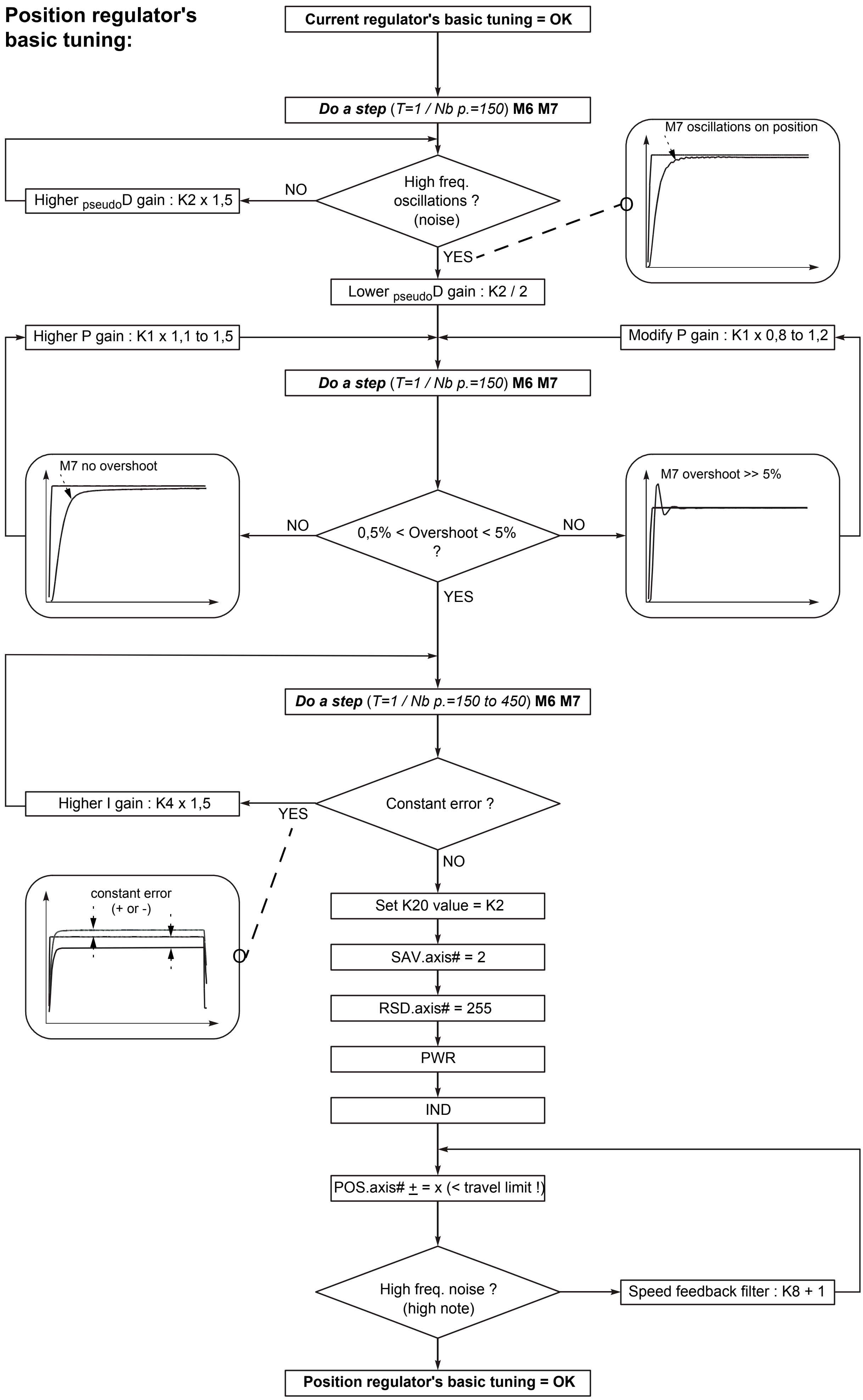
Tuning the controller's regulators gains is necessary to adapt them to the characteristics of the motor and load that it will drive. You will use **ETEL Tools** for this operation, especially its **Scope Tool** (refer to §8.4).

Caution: The following diagram describe, in a basic way, the principles to tune the controller's current and position regulators. Of course, these procedures are general and cannot be considered as perfect for setting all applications. Therefore, receiving a training by an ETEL support engineer is generally necessary for the user, to be able to accurately tune a controller.

Current regulator's basic tuning:



Position regulator's basic tuning:

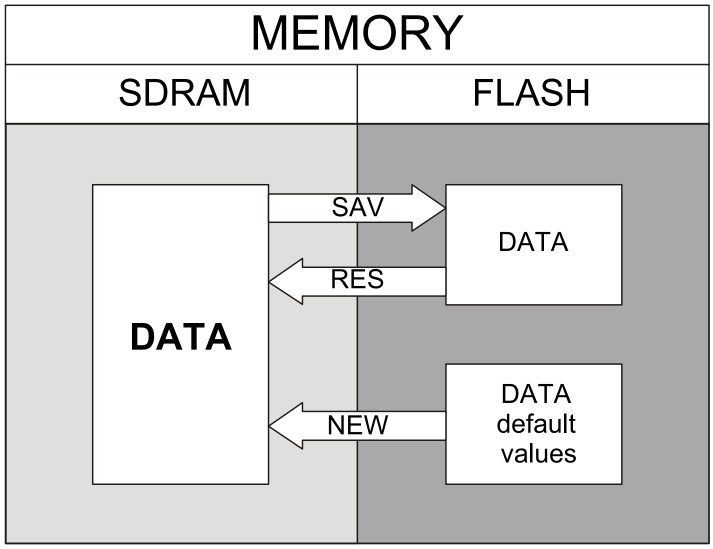


Chapter C: System functions

11. Save the settings

The controller is set with 4Mbytes (1Mx32bits) of **SDRAM memory** and 2Mbytes of **flash memory**. The SDRAM is a volatile memory which is erased each time the controller is switched OFF, whereas the flash is a non-volatile memory and the data which is stored in it is not lost when the controller is switched OFF. **All calculations and operations done by the controller are made with SDRAM values**.

SAV, RES and NEW commands allow the user to transfer data from the SDRAM to the flash and vice-versa.



The **SAV** command (**SAV**e) **saves into the "flash"** memory the controller's data (like registers, sequences, ...), so that they are not lost when the controller is switched off and on again. Saved data is defined by the first parameter.

The **RES** command (**RES**tore) rereads the controller's data (like registers, sequences,...), previously saved with SAV command into the "flash" memory, and **restores** them into the **SDRAM** memory. Restored data is defined by the first parameter.

The **NEW** command **reloads** in the **SDRAM** the default values of parameters K, or **erases** the S user sequence stored in the SDRAM, depending on the first parameter's value.

Command format	<P1>	Comment
SAV.<axis> = <P1>	0 1 2	Saves sequence (S register), user look-up tables (L register), user variable X, parameters K, trigger (E register), real-time interrupt (R register), float (F register) and axis number in flash memory Saves sequence (S register) and user look-up tables (L register) in flash memory Saves user variable X, parameters K, trigger (E register), real-time interrupt (R register), float (F register) and axis number in flash memory
RES.<axis> = <P1>	0 1 2	Restores sequence (S register), user look-up tables (L register), user variable X, parameters K, trigger (E register), real-time interrupt (R register), float (F register) and axis number from flash to ram memory if all the switches of the DIP switch are set to 1 (refer to §12.1.1 for more information) Restores sequence (S register) and user look-up tables (L register) from flash to ram memory Restores user variable X, parameters K, trigger (E register), real-time interrupt (R register), float (F register) and axis number from flash to ram memory if all the switches of the DIP switch are set to 1 (refer to §12.1.1 for more information)
NEW.<axis> = <P1>	0 1 2	Clears sequence in ram memory and sets default K value in ram memory Clears sequence in ram memory Sets default K value in ram memory

Caution: After executing RES and NEW commands, SDRAM ordinary values are replaced by the values read in the "flash" memory and are definitively lost. Similarly, the SAV command **crushes** the values contained in 'flash' with those contained in the SDRAM. **To avoid possible problems, SAV, RES and NEW commands should be executed if the controller is in 'Power Off'.**

Remark: When the controller is **switched on**, data is automatically restored, like with a **RES.<axis>=0** command (so that it is not necessary to do it manually).

All parameters K depths are saved and reread when SAV, RES and NEW commands are executed. Moreover, these commands are generally not used in a sequence but only when sending on-line commands or with ETEL Tools.

Example:

Parameter K1 is the position loop proportional gain. The default value of this parameter is 100. The table below shows the state of the SDRAM and the 'flash' after realizing different actions one after the other.

Actions	State of the SDRAM after acting (the controller uses these values for all its calculations)	State of the flash after acting
Switch on the controller	Parameter K1 is equal to 100, value by default	Parameter K1 is equal to 100, value by default.
K1.1 = 5000	Parameter K1 is equal to 5000.	Parameter K1 is equal to 100, value by default.
Switch off and on the controller	Parameter K1 is equal to 100, value by default.	Parameter K1 is still equal to 100, value by default.
K1.1 = 4000	Parameter K1 is now equal to 4000.	Parameter K1 is equal to 100, value by default.
SAV.1 = 2	Parameter K1 is still equal to 4000.	Parameter K1 is now equal to 4000.
NEW.1 = 2	Parameter K1 is now equal to 100, value by default.	Parameter K1 is still equal to 4000.
RES.1 = 2	Parameter K1 is equal to 4000, value contained in the flash.	Parameter K1 is still equal to 4000.

12. Basic functions and settings (necessary to operate the controller)

All functions defined in this chapter are necessary to operate the controller and are used for all applications. Advanced functions in §13. are not used in most applications. Only advanced users will have to use them.

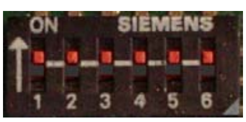
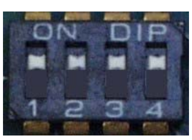
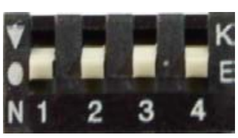
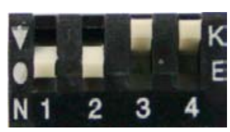
12.1 Axis number selection

The axis number of a position controller can be set either with a **DIP switch** or **by software** with the AXI command.

12.1.1 Selection with DIP switch

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

It is possible to assign or to change the axis number of the controller with a DIP switch. After each starting, the controller takes the axis number given by the DIP switch except when all the switches are in the high position (or low position for the DSCDM) which means set to 1 like in the pictures below. In this case the axis number is set by the AXI command or the value previously saved in the controller or by the default value always equal to 1 for a single axis controller (DSC2P, DSC2V) or 0 and 1 for a dual axes controller (DSCDP, DSCDM, DSCDL). This default value is used when no AXI command has been executed or no save has been done. **For the DSCDP, DSCDL and DSCDM**, as there are 16 possible values on the DIP switch for 30 axes maximum (0 to 29), the number of the first axis of a controller will be equal to the value given by the DIP switch multiplied by 2. The second axis number of the same controller will be automatically incremented by one.

	Default position		Example	
DSC2P DSC2V		The value given on the DIP switch represents a binary value. As the switch number 6 is not used, there are 32 possibilities. The axes are numbered from 0 to 30 because the axis 31 is reserved for the DSMAX.		The axis number given by the following DIP switch will be equal to: $2^0 + 2^1 = 3$.
DSCDP DSCDL		The value (16 possibilities) given on the DIP switch represents a binary value.		The value given on this DIP switch is equal to: $2^0 + 2^1 = 3$. Then, the first axis of this controller will have the number 6 and the second one the number 7.
DSCDM		The value (16 possibilities) given on the DIP switch represents a binary value.		The axis number given by this DIP switch is equal to: $2^0 + 2^1 = 3$. Then, the first axis of this controller will have the number 6 and the second one the number 7.

If there are two or more controllers on the same TEB ring with the same axis number, the **BAD NODE TEB ERR** error (M64=59) will appear. This error will be generated only on the duplicated node(s) and its (their) LED 'TEB OK' will blink.

12.1.2 Selection with command AXI

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **AXI** (**AXI**s) command is used to change an axis number. This command asks for the controller serial number and the new axis number. The serial number is asked by security to avoid an inappropriate change of the axis number. The AXI command **MUST** be used **only** if all the switches (of the DIP switch) are set to 1 (refer to §12.1.1) and in case of a dual axes controller (DSCDP, DSCDL and DSCDM), it must be used on an even axis number with an even value.

Command format	<p1>	Controller	<p2>	Comment
AXI.<axis> = <p1>, <p2>	controller serial number	DSC2P DSC2V	Axis number 0-30	Changes the current axis number of a controller; <P1> is the serial number, <P2> is the new axis number.
		DSCDP DSCDL DSCDM	Axis number 0-29	

Example:
The user wants to change the axis number 2 into the axis number 6:

`SER.2` ;The controller gives its serial number, for example 4059.
`AXI.2=4059,6` ;The axis has now the number 6 (for the **DSCDP, DSCDL and DSCDM**, the second axis has now the number 7)
The new number has to be saved once for all future applications with the SAV command.
`SAV.6=2` ;The new axis number has been saved (for the **DSCDP, DSCDL and DSCDM**, the second axis number is calculated with regards to the first one).

Remark: The SAV command is given for the axis 6 and not for axis 2 (refer to §11. for more information). In case of dual axes controller (DSCDP, DSCDL and DSCDM), the SAV command **must** be used only on the even axis number.

If there are two or more controllers on the same TEB ring with the same axis number, the **BAD NODE TEB ERR** error (M64=59) will appear. This error will be generated only on the duplicated node(s) and its (their) LED 'TEB OK' will blink.

12.1.3 Serial number and firmware version

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **SER** command (**SER**ial number) which is an alias of monitoring M73 is used to know the serial number of the position controller. It can be requested by the AXI command before changing the axis number.
The **VER** command (**VER**sion) which is an alias of monitoring M72 is used to know the firmware version contained in the controller (refer to §12.12 for more information).

M	Alias	Name	Comment
M71	-	Software boot version	Gives the software boot version number installed in the controller.
M72	VER.<axis>	Firmware version	Gives the firmware version installed in the controller.
M73	SER.<axis>	Serial number	Gives the serial number of the controller.

12.2 Motor

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The parameters described here are set only once and are normally automatically set with ETEL Tools (see §8.)

K	Alias	Name	Controller	Value	Comment
K240	-	Motor type	-	0 1	Linear motor. Rotary motor. Depth 0 for primary encoder and depth 1 for secondary encoder **(depth 1 is ONLY available on DSC2P and DSC2V)**
K89	-	Motor phase number and PWM type selection	DSC2P DSC2V	10 20 30 11 21 31 14 24 34	One-phase motor, PWM at 24kHz **(~~DSC2V~~)** Two-phase motor, PWM at 24kHz **(~~DSC2V~~)** Three-phase motor, PWM at 24kHz **(~~DSC2V~~)** One-phase motor, PWM at 12kHz Two-phase motor, PWM at 12kHz Three-phase motor, PWM at 12kHz One-phase motor, PWM at 6kHz **(~~DSC2P~~)** Two-phase motor, PWM at 6kHz **(~~DSC2P~~)** Three-phase motor, PWM at 6kHz **(~~DSC2P~~)**
			DSCDP DSCDM	10 20 30	One-phase motor, PWM at 18kHz Two-phase motor, PWM at 18kHz Three-phase motor, PWM at 18kHz
		Motor phase number	DSCDL	10 20	One-phase motor Two-phase motor
K54	-	Pairs of poles of the motor	-	-	**For linear motors: K54=1**
K56	-	Motor phase and force inversion	-	0 1	Normal. Inverts phases and force signs.

Motor type (parameter K240):
This parameter is only used for unit calculation by ETEL Tools program and by DLL libraries (if used). If the values of this parameter is not correct, the motor will correctly work but the curves will not be displayed correctly in the units given by the ETEL Tools program (meter, amp, etc.)

Remark: Monitoring M240 indicates the type of motor given **either** by parameter K240 **or** the EnDat 2.1 encoder (depending on the encoder type selection (parameter K79)).

M	Name	Comments
M240	Motor type	Gives the type of motor connected to the position controller

Number of motor's phases and switching (parameter K89):
Tells the controller the number of motor phases. There is no difference for the user if one, two or three phase motor is used. The meaning of each parameters is unchanged, commands are the same, etc.
For especially high inductance motors, a specific switching may be selected with parameter K89 = 11, 21 or 31.

Remark: Parameter K89 is only read when the controller is **switched on**. This parameter must be saved with the SAV command when it is changed and then the controller must be switched off and on to integrate this new data. If this new value is incorrect, the **K89 BAD VALUE** error (M64=41) will appear.

Number of motor's pairs of poles (parameter K54):
Is used with rotary motors. Parameter K54 shows the number of pair of motor's magnetic poles; this parameter is used by the motor commutation look-up table (LKT). The sinusoidal currents sent in the motor phases are calculated by the current reference generator, with the LKT (refer also to §12.7.1.4). With linear motors, K54=1.

Motor phase and force inversion (parameter K56):
It enables the permutation by software of the connection of the motor phases as well as the sign of the motor force. If the phases have been inverted during the installation, any initialization will give a totally wrong parameter K53 (refer to §12.7.1.4). The AUT command allows the automatic calculation of the adequate value for parameter K56.

Remark: If you manually set K56 and K52=1, it is required to perform an AUT command after, to tune K53.

12.3 Position encoder

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The position encoder's K parameters described below are only set once. Generally, they are automatically set with ETEL Tools (refer to §8.).
The position encoder is the device enabling the motor to measure its position. It is made up of two parts: the head, connected to the motor, and the scale, which is fixed. Parameter **K79** allows the user to select the type of encoder(s) used on the controller.

K	Name	Value	Bit #	Comment
K55	Encoder position increments factor	-	-	With rotary encoder: number of [dpi] / motor revolution period With linear encoder: number of [dpi] / motor magnetic period
K68	Encoder reading way inversion	1 2 4	0 1 2	Analog 1 Vptp or EnDat 2.1 encoder reading way is inverted. TTL encoder reading way is inverted. Inverts force reference from MACRO **(~~DSCDM~~ and ~~DSCDL~~)**
K70	Encoder sine offset correction	-	-	Correction of the sine signal offset.
K71	Encoder cos offset correction	-	-	Correction of the cosine signal offset.
K72	Encoder sine ampl. correction	-	-	Correction of the sine signal amplitude.
K73	Encoder cos ampl. correction	-	-	Correction of the cosine signal amplitude.
K75	Distance between two indexes	-	-	Average distance between two indexes with the multi-ref. marks scales.
K79	Encoder type selection	0 1 4 7 20 21 23 24 100 101 104	-	1 Vptp analog encoder selection (secondary = TTL for DSC2P and DSC2V) TTL encoder selection (secondary = 1 Vptp for DSC2P and DSC2V) EnDat 2.1 encoder selection (secondary = TTL for DSC2P and DSC2V) TTL encoder selection (secondary = EnDat 2.1) **(only on DSC2P and DSC2V)** Stepper in open loop without encoder **(~~DSC2P~~ and ~~DSC2V~~)** Stepper in open loop with reading of a TTL encoder **(~~DSC2P~~ and ~~DSC2V~~)** Stepper in open loop with reading of a 1 Vptp analog encoder **(~~DSC2P~~ and ~~DSC2V~~)** Stepper in open loop with reading of a EnDat 2.1 analog encoder **(~~DSC2P~~ and ~~DSC2V~~)** Macro mode: Analog encoder selection (1 Vptp) (Secondary = TTL for DSC2P) **(~~DSCDM~~ and ~~DSCDL~~)** Macro mode: TTL encoder selection (secondary = analog encoder 1 Vptp for DSC2P) **(~~DSCDM~~ and ~~DSCDL~~)** Macro mode: EnDat 2.1 encoder selection (secondary = TTL encoder selection for DSC2P) **(~~DSCDM~~ and ~~DSCDL~~)**
K241	Encoder period	-	-	Encoder period in [nm] (linear encoder) or number of periods per turn (rotary encoder). Depth 0 for primary encoder and depth 1 for secondary encoder **(depth 1 is ONLY available on DSC2P and DSC2V)**

Remark: Monitoring **M13** allows the user to display the position given by the secondary encoder (for DSC2P and DSC2V). It is also available with K79=21, 23 and 24 for DSCDP, DSCDM and DSCDL. To interpret this value, parameters K27, K50 and K69 or K77 must be taken into account according to the type of encoder (refer to §13.3.10 and §13.10 for more information about parameters K69 and K77 and to §13.3.10 for parameter K50).

Monitoring **M5** allows the user to display the distance (dpi) covered due to the homing (used in the case of external reference with the Macro optional board for example).

If parameter K68 is modified, the command AUT=10 must be executed to re-calculate parameters K53 and K56.

Parameter K79 is only read when the controller is **switched on** and must be saved with SAV command when it is changed and then the controller must be switched off and on to integrate this

new data. If this new value is incorrect, the **K79 BAD VALUE** error (M64=40) will appear.
For the DSCDP, if parameter K79 is modified (with K79=100, 101 and 104), the SAV command must be executed on both axes. For the DSCDP, K79 = 4 and 104 is available only from DSCDP3xx-xxxE. For the DSCDL, K79 = 4 is available only from DSCDL3xx-xxxC.

Parameter K55 allows the setting of the number of increments per magnetic period (linear motors) or per turn (rotary motor); this parameter is used by the motor commutation look-up table (LKT).

Monitoring **M239** indicates the encoder period given **either** by parameter K241 **or** the EnDat 2.1 encoder (depending on the encoder type selection (parameter K79)).

M	Name	Comments
M239	Encoder period	Gives the type of motor connected to the position controller

The currents sent to the motor phases are calculated by the current reference generator with the commutation look-up table. Here is the formula to calculate the number of increment (parameter K55):

- For **rotary motor:**

$$K55 = NPCod \cdot 1024 \cdot 2^{K77} \quad \text{(for analog encoder)}$$

$$K55 = NPCod \cdot 64 \cdot 2^{K69} \quad \text{(for TTL encoder)}$$

NPCod = Encoder periods number per turn [p/r]

- For **linear motor:**

$$K55 = \frac{Pway}{PCod} \cdot 1024 \cdot 2^{K77} \quad \text{(for analog encoder)}$$

$$K55 = \frac{Pway}{PCod} \cdot 64 \cdot 2^{K69} \quad \text{(for TTL encoder)}$$

PCod = Encoder periods number [m]
Pway = magnetic period [m]

Remark: Refer to §13.3.10 and §13.10 for more information about parameters K69 and K77 and to §13.3.10 for parameter K50.

Monitoring **M241** indicates the encoder interpolation factor.

M	Name	Comments
M241	Encoder interpolation factor	Gives the interpolation factor of the encoder. In the above-mentioned formulas, monitoring M241 corresponds to 1024 * 2^{K77} (for an analog encoder) and 64 * 2^{K69} (for a TTL encoder).

12.3.1 Analog encoders (K79=0)

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **analog encoders** (1Vptp) can determine exactly the motor position thanks to two sinusoidal signals with a phase-shift of 90° (sine and cosine). The period of these signals varies according to the type of encoder used (from 128 nm to 32 mm). The smaller the period is, the bigger the precision is. These signals must be calibrated to be optimized so as to have the same amplitude and no offset. Parameters K70, K71, K72 and K73 make such corrections. Some of the scales used with the analog encoders are multi-reference mark, and in that case the average distance between reference marks must be given via parameter K75. Formula for **linear motors**:

$$K75 = \frac{1024 \cdot \text{Distance between 2 indexes [m]}}{PCod} \quad \text{with PCod = encoder period [m]}$$

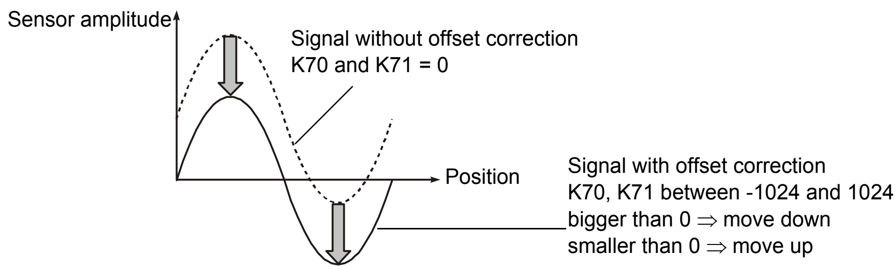
The encoder **resolution,** which is the smallest distance measured by the encoder, is given by the following formula:

$$\text{Analog encoder resolution [m]} = \frac{\text{Encoder period [m]}}{1024 \cdot \underbrace{2^{K77}}_{\text{Interpolation factor}}}$$

Parameter K75 calculation, formula for **rotary motors**:

$$K75 = \frac{1024 \cdot NPCod}{NRef} \quad \text{with NRef = nb. of encoder reference marks / turn}$$

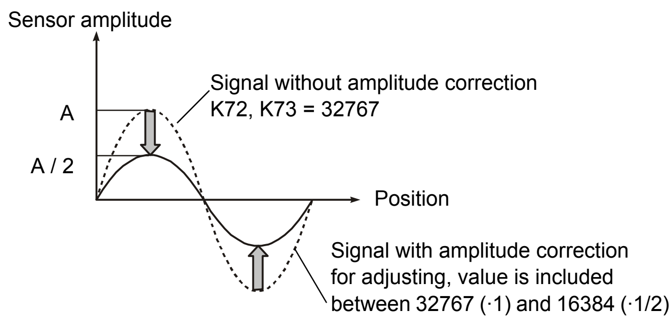
Offset correction (K70, K71):
If parameter K70 or/and K71 is smaller than 0, a positive offset is added to the corresponding signal and a value higher than 0 adds a negative offset. If parameter K70 or/and K71 is equal to 0, no offset correction is done.



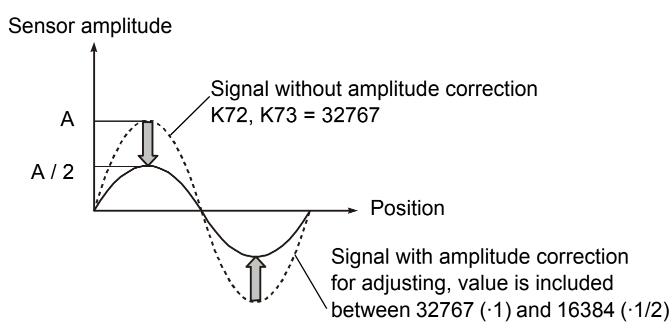
Remark: Monitorings M40 and M41 allows the user to monitor the sine and cosine signals of the analog encoder (refer to §12.3.7 for more information).

Amplitude correction (K72, K73):

The amplitude correction only allows the decrease of the signal amplitude and never the increase of it. That is why, **the encoder head tuning** is highly important because it will not be possible to correct a too weak amplitude with the software. It is possible to decrease a signal amplitude at best by a factor 2 with the value 16384. The value 32767 does not correct any encoder signals amplitude.



Remark: During offset and amplitude corrections manual research, it is advised to start first disabling individually every signal offset, and then to correct the amplitudes in order to adjust the bigger on the smaller.



12.3.2 EnDat 2.1 encoders (K79=4)

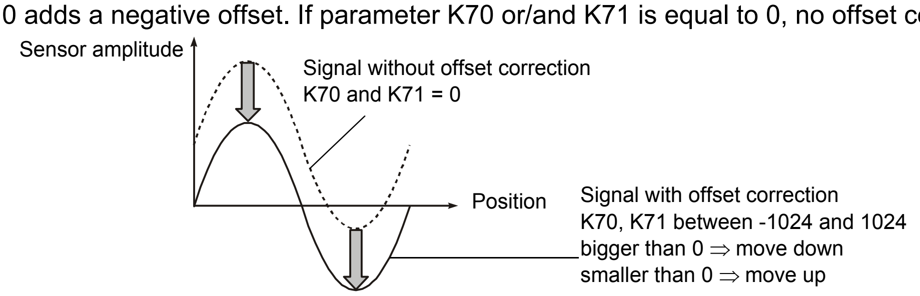
Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **EnDat 2.1 encoders** (analog 1Vptp) can determine the absolute position of the motor thanks to two sinusoidal signals with a phase-shift of 90° (sine and cosine) and the information given by its two serial lines (clock and data). The period of these signals varies according to the type of encoder used (from 128 nm to 32 mm). The smaller the period is, the bigger the precision is. These signals must be calibrated to be optimized so as to have the same amplitude and no offset. Parameters K70, K71, K72 and K73 make such corrections.

Caution: Be careful of the length (max. 150m with distributed capacitance 90 pF/m) of the encoder cable because the clock frequency of the EnDat 2.1 is equal to 500kHz. Refer to the encoder's manufacturer for more information about the EnDat 2.1 encoder's cable.

Offset correction (K70, K71):

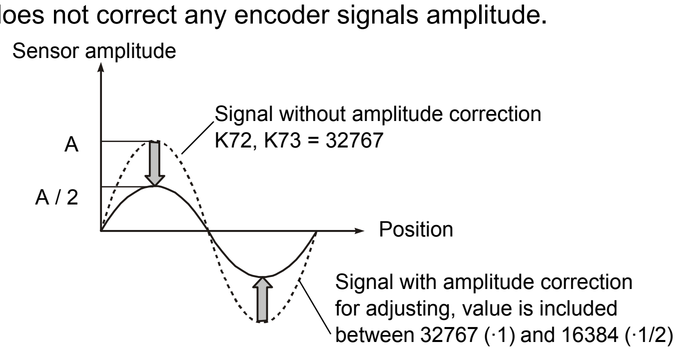
If parameter K70 or/and K71 is smaller than 0, a positive offset is added to the corresponding signal and a value higher than 0 adds a negative offset. If parameter K70 or/and K71 is equal to 0, no offset correction is done.



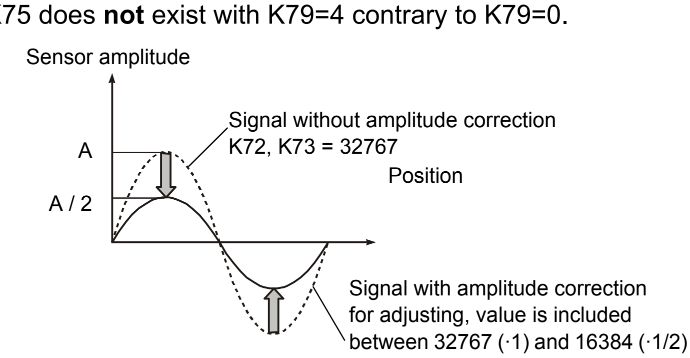
Remark: Monitorings M40 and M41 allows the user to monitor the sine and cosine signals of the analog encoder (refer to §12.3.7 for more information).

Amplitude correction (K72, K73):

The amplitude correction only allows the decrease of the signal amplitude and never the increase of it. That is why, **the encoder head tuning** is highly important because it will not be possible to correct a too weak amplitude with the software. It is possible to decrease a signal amplitude at best by a factor 2 with the value 16384. The value 32767 does not correct any encoder signals amplitude.



Remark: During offset and amplitude corrections manual research, it is advised to start first disabling individually every signal offset, and then to correct the amplitudes in order to adjust the bigger on the smaller.
Parameter K75 does **not** exist with K79=4 contrary to K79=0.



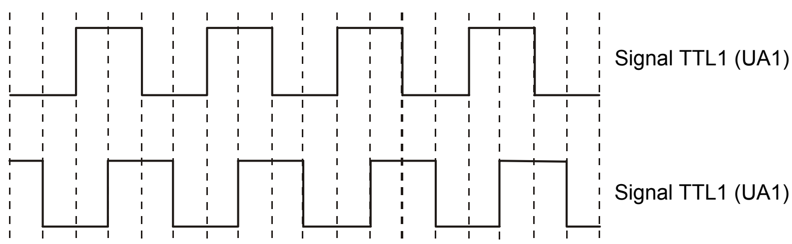
With an EnDat 2.1 encoder, additional information can be displayed thanks to the following monitorings:

M	Name	Comments
M145	Encoder type	Gives the type of EnDat 2.1 encoder (linear or rotary and single or multi-turn)
M146	EnDat measuring step	Gives the number of measuring step per turn (for rotary encoder) or in [mm] (for linear encoder)
M147	EnDat period number	Gives the number of encoder period per turn (for rotary encoder) or in [mm] (for linear encoder)
M148	EnDat pulse number	Gives the number of pulse
M149	EnDat turn number	Gives the number of turn

12.3.3 TTL encoders (K79=1 and K79=7)

Available on	**DSC2P**	**DSC2V**	**DSCDP**	**DSCDL**	**DSCDM**

TTL encoders measure the motor position with two phase-shifted TTL signals. Each change of state of one of the signals corresponds to a position increment. Parameters K70 to K73 are not used.



This formula gives the TTL encoder **resolution**, which is the smallest distance measured by the encoder, but it concerns the **encoder only**, not the resolution obtained in the controller:

$$\text{Encoder resolution [m]} = \frac{\text{Encoder period [m]}}{4}$$

The **real position reading resolution** is given in the controller by parameter **K55** (refer to §12.3).

The encoder **resolution,** which is the smallest distance measured by the encoder, is given by the following formula:

$$\text{TTL encoder resolution [m]} = \frac{\text{Encoder period [m]} \cdot 16}{1024 \cdot 2^{K69}}$$

TTL special filter: With a TTL encoder, oscillations may happen when the motor stops and stays on a position. When the motor moves, they disappear. They are due to the encoder weak resolution. A special speed filter, also named “smooth filter” has been created to reduce them. It can be tuned via parameter K11.

Parameter K11 filter is taken into account only when the real speed is equal to 0. As soon as the real speed is different from 0, parameter K11 filter is deactivated.

K	Name	Comment
K11	TTL speed smooth filter	TTL encoder special filter on the speed (K79=1).

Remark: The difference between K79=1 and K79=7 is the type of the secondary encoder. With K79=1, the secondary encoder, which can be used only on a DSC2P and DSC2V, is a 1 Vptp analog encoder. With K79=7 which can be used only on a DSC2P and DSC2V, the secondary encoder is an EnDat 2.1 encoder giving an absolute position.

12.3.4 Stepper in open loop (K79=20, 21, 23 and 24)

Available on	~~DSC2P~~	~~DSC2V~~	**DSCDP**	**DSCDL**	**DSCDM**

Refer to §12.13 for more information about the stepper in open loop.

12.3.5 Macro modes (K79=100, 101 and 104)

Available on	DSC2P	DSC2V	DSCDP	~~DSCDL~~	~~DSCDM~~

Refer to the **'DSO-MAC User's Manual'** for more information.

12.3.6 Position factors for DLLs

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The parameters below are only used by the DLLs, to calculate the position unit with indirect encoder.

K	Name	Comment
K242	Position multiplication factor	Used by DLLs only to calculate the position unit with indirect encoder
K243	Position division factor	Used by DLLs only to calculate the position unit with indirect encoder

In ETEL Tools, all the quantities representing a position given in ISO unit can be multiplied by the Kpos-iso factor. Kpos-iso = K242 / K243.
That way, it is possible to make the position scale bigger or smaller. The other ISO quantities (speed, acceleration...) will be adjusted accordingly. It is really interesting for an indirect measurement of position (for example if the user wants to measure the position on a linear movement which is caused by a rotary motor including the encoder).

12.3.7 Encoder monitorings

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

They measure the signal values given by the **analog** position encoder.

M	Name	Comment
M40	Analog encoder sine signal	Encoder sine signal value
M41	Analog encoder cosine signal	Encoder cosine signal value
M42	Analog encoder index signal	Encoder index value **(ONLY available on DSC2P and DSC2V)**
M43	Analog encoder amplitude signal	Analog encoder $\text{sine}^2 + \text{cosine}^2$

The conversion formula to know the measured encoder signal value in ISO units is:

Controller	For monitorings M40, M41 and M42	For monitoring M43
DSC2P DSC2V DSCDP DSCDM	$\text{Encoder value } [\hat{V}] = \dfrac{\text{Encoder value [inc]}}{4096 \cdot x}$	$M43\ [\hat{V}] = \dfrac{\sqrt{M43[inc]}}{2048 \cdot x}$
DSCDL	$\text{Encoder value } [\hat{V}] = \dfrac{\text{Encoder value [inc]}}{65536 \cdot x}$	$M43\ [\hat{V}] = \dfrac{\sqrt{M43[inc]}}{32768 \cdot x}$

The x value depends on the type of position controller:

	Value	Position controller
X	0.83	DSC2P and DSC2V
	0.74	DSCDP, DSCDL and DSCDM

Remark: To convert ISO units ([m/s], f.e.) and ETEL units ([upi], f.e), refer to §19.
K66=4 allows the user to display on the scope the amplitude of the encoder's analog signals as well as the position of the index.

12.4 Precaution parameters - IMPORTANT

The parameters K described below are set only once. Generally, they are set automatically by ETEL Tools during the Drive setting process (refer to §8.).
These parameters are added up to the existing material protections (fuses, mechanical end stops, etc) and protect the controller, the machine, its motor and its compounds. There are three types of protection parameters: 1st: those which introduce movement limits, 2nd: those which protect the motor from an overcurrent, 3rd: the general protection parameters which control the interactions between the motor and the rest of the machine.

12.4.1 Movements limits

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Parameters K34 and K35 limit the linear motors movements. If the value of a position to reach programmed with the POS command is higher than the one contained in parameter K35, the programmed position will automatically be brought back to parameter K35. And it is the same with a value smaller than parameter K34. In both cases the movement takes place with the limit value and **the controller does not display an error message**. Parameter K36 enables or disables the activation of the protection of parameters K34 and K 35; an error may appear (depends on the value of parameter K36) if the motor goes over the parameters K34 and K35. Parameters K30 and K31 **switch automatically off the power** of the motor in case of position error or in case of overspeed of the values contained in parameters K30 and K31. In that case the controller **switches in error mode**, lights a red error LED and displays a message to identify the error.

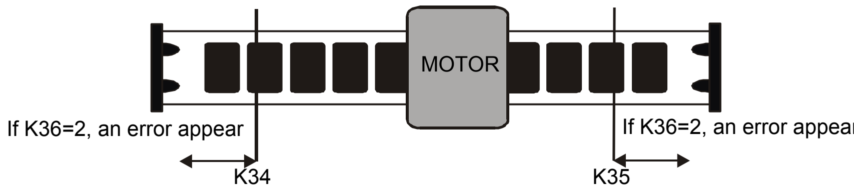
Valid for	K	Name	Value	Bit #	Comment	Units
All motors	K30	Motor position tracking error limit	-		When the tracking error is > K30, the controller generates the **TRACKING ERROR** error (M64=23)	[dpi] [rdpi]
	K31	Motor real speed limit.	-		When the speed is > K31, the controller generates an **OVER SPEED** error (M64=24)	[dsi] [rdsi]
	K36	Motor position limitation mode	1 2 4	0 1 2	Use of parameters K34 and K35 as limit on the target the motor can reach. These limits are tested on every sti only if a homing has been previously done. Used with K61=1 Use of parameters K34 and K35 as limit on the actual position of the motor. If the motor reaches these limits, it generates an **OUT OF STROKE** error (M64=65). These limits are tested on every sti only if a homing has been previously done. Used for all values of parameter K61 Use of parameters K34 and K35 as limit on the target to generate an **REF OUT OFSTROKE** error (M64=66) when the movement starts. These limits are tested only if a homing has been previously done. Used with K61=1. This error is generated when one of the following command is executed: POS=, POS+=, POS-=, STE=, STE+=, STE-=, STA= and STI=	-
	K34	Minimum software position limit.	-		Depending on the values of K36 and K61 parameters, the motor cannot go lower than parameter K34	[upi] [rupi]
	K35	Maximum software position limit	-		Depending on the values of K36 and K61 parameters, the motor cannot go higher than parameter K35	[upi] [rupi]
	K145	Search limit stroke mode selection (SLS mode)	0 1 2 3		SLS on mechanical end stop, positive movement. SLS on mechanical end stop, negative movement. SLS on limit switch, with positive movement SLS on limit switch, with negative movement	-
Rotary motors only	K27	Maximum position value	-	-	Defines the maximum position range limit (K27). Depth 0 for primary encoder and depth 1 for secondary encoder (refer also to §12.10.4.2)	[upi] [rupi]

Remark: If K145 = 0 or 1, parameters K41, K42, K43, K44 and K47 are taken into account (refer to §12.9 for more information).
If K145 = 2 or 3, parameters K41, K42, K47 and K58 are taken into account (refer to §12.9 for more information).

SLS (**S**earch **L**imit **S**troke) command is:

Command format	Comments
SLS.<axis>	Searches the limit stroke according to parameter K145. Limit position is returned in monitorings M36 and M37 (given in [upi]). K47 is taken into account by SLS command but not in M36 and M37.

Remark: A homing must be done (with IND command) before sending the SLS command.
The SLS command is available only when K61 = 1.



Note: With K36 = 1 the target of a POS / STA / STI movement is limited by parameters K34 and K35 without generating an error.

12.4.2 Current limits

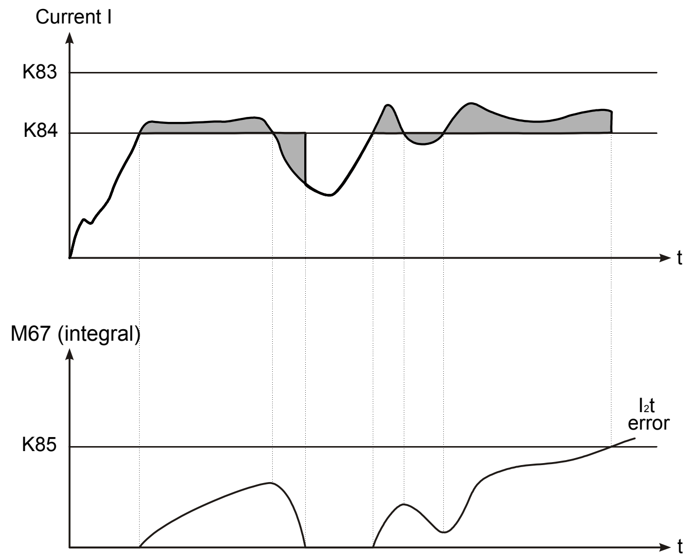
Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

An overcurrent in the motor phases can destroy it. Parameters **K83**, **K84** and **K85** help to avoid it. Parameter **K60** will limit the force/torque reference (theoretical) at the position regulator's output. Thus, the current in the motor should also be limited. However, the real force/torque in the motor may oscillate and go over parameter K60.

K	Name	Comment	Units
K83	Motor overcurrent limit	If the current in the phases is > K83, the **OVER CURRENT** error appears (M64=2 or 3)	[ci]
K84	I2t rms current limit	The integration starts when the motor current is > K84	-
K85	I2t time limit	If the integral value is > K85, the **I2T ERROR** error (M64=4) appears	-
K60	Force/torque limit	Max. force/torque reference at the position regulator output (will limit the current in the motor)	[foi] [toi]

M	Name	Comment	Units
M67	I2t integration limit test	When monitoring M67 is greater than parameter K85, the controller generates an **I2T ERROR** error (M64=4)	-
M82	Controller maximum current	Maximum current [A] = M82/100. Should never be reached! This theoretically maximum value is used for calculations only	[A]*100

Parameter K83 protects the motor from an **instantaneous overcurrent**. If the motor current gets higher than the value of parameter K83, the controller changes to error mode. **OVER CURRENT1** (M64=2) or **OVER CURRENT2** (M64=3) message is displayed.
Reaching the current in monitoring **M82 is not permitted**; this value is a theoretical measurable value, for calculations only! (refer to §20.2).
Parameters K84 and K85 protect the motor from a **too high current during a too long time**, whose **energy** could raise the temperature high enough to burn the motor. This i^2t (energy) limit is given by parameter K85. When the instantaneous current value increases over the value of parameter K84, an integrator is activated. As long as the current stays over the value of parameter K84, it is integrated, but when it passes under it the integrator empties progressively. If the integral value is higher than parameter K85, the controller displays an **I2T ERROR** error (M64=4).



Remark: It may happen that, during a periodic movement (machine cycle), parameter K84 may temporarily be overcome, and the integrator may not entirely be empty at the end of the cycle. In that case, after several cycles, an **I2T ERROR** error (M64=4) is displayed. The cycle is then called an **unstable** cycle.

Caution: All software protections described above cannot protect at 100% a motor against overheating; calculation of the energy balance in the whole system is required to avoid overheating.
Thus, ETEL cannot be hold as responsible in case of system failure due to motor overheating.

The next example explains how to calculate parameters K83, K84, K85. Some information concerning the system configuration has to be known.

Motor:

I_m peak:	Motor peak current [A] (this is also the max current of the application).
I_m continuous:	Motor continuous current [A].
t:	Maximum time at I_m peak before an i^2t error [s].
I_{over} current:	Ultimate motor current value, for a motor **OVER CURRENT** error [A] (= I_m peak [A] + 20%)

Controller:

I_{max} controller: Maximum current of controller [A] = M82 / 100

Parameter K83 motor over current limit:

$$K83 = \frac{I_{over}\text{current} \cdot 32768}{M82/100}$$

Parameter K84 i^2t rms current level:

$$K84 = \left(\frac{I_m\text{continous}}{M82/100} \cdot 100\right)^2 \cdot 0.8192$$

Caution: If a **2-phase** linear motor performs a **back and forth movement with very short strokes**, only one phase will be used. Thus, thermal load will be concentrated on the half of the motor surface. Take parameter K84, (value obtained with formula above) and **divide it by 2: K84 short_stroke = K84 / 2**.

Parameter K85 i^2t integration limit (energy):

$$K85 = 9830 \cdot t \cdot \left(\left(\frac{I_m\text{ peak}}{M82/100} \cdot 100\right)^2 - \left(\frac{I_m\text{continuous}}{M82/100} \cdot 100\right)^2\right)$$

Example:

Motor:	I_m peak	= 7.96 A
	I_m continuous	= 3.196 A
	t	= 5 s
	I_{over} current	= I_m peak + ~20% = ~1.2 · I_m peak = 9.55 A
Controller:	DSC2P1**2**1xx	I_{max} controller = (M82 / 100) = 12,5 A

$$K83 = \frac{9,55 \cdot 32768}{12,5} = 25000$$

$$K84 = \left(\frac{3.196}{12,5} \cdot 100\right)^2 \cdot 0.8192 = 536$$

$$K85 = 9830 \cdot 5 \cdot \left(\left(\frac{7.96}{12,5} \cdot 100\right)^2 - \left(\frac{3.196}{12,5} \cdot 100\right)^2\right) = 167000000$$

12.4.3 Safety signals on DIN and DOUT

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

When a motor is integrated in a complex machine and an error is detected in the controller, it is important to send a message about it to the rest of the machine so that the other elements can adequately react. This is possible with the digital outputs (the DSO-HIO optional board's outputs are not concerned). Parameter K37 allows the user to select the digital outputs to be used. When an error occurs, the outputs selected by parameter K37 change from the logical value '1' to '0'. For example, DOUT1 can be connected to a relay which short-circuits the motor phases in case of error, making a **magnetic brake**. The outputs that have to be activated in case of error are chosen via the binary value of parameter K37.

The number of digital outputs is different from a position controller to another. Here is a table giving the number of digital outputs present on each position controller (refer to the corresponding **'Hardware Manual'** for more information):

DSC2P / DSC2V	**DOUT #**	4	3	2	1
DSCDP / DSCDL	**DOUT #**	-	-	2	1
DSCDM	**DOUT #**	-	3	2	1

Caution: On the DSCDM, the digital inputs and outputs are on the same pin. The pin must be selected in order to have an input **or** an output. **It is NOT possible to have both on the same pin.** To use a pin as a digital input, the bit corresponding to this input **MUST** be equal to 0 in parameter K171.

Example:

To activate DOUT3 of the DSC2P or DSC2V, parameter K37 has to be set to 4, because it corresponds to 0000000000000100 binary. Several outputs can be activated simultaneously, for example, DOUT2 and DOUT1 are selected with value 3 (0000000000000011 bin).

K37	Bits# 3 to 15 not used	Bit# 3 DOUT4	Bit# 2 DOUT3	Bit# 1 DOUT2	Bit# 0 DOUT1
3	0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	1	1
4	0 0 0 0 0 0 0 0 0 0 0 0 0	0	1	0	0
Decimal	**Binary**				

On the contrary, parameter **K33** switches the controller in error mode by adding the number 0 and switches off the power if DIN1 digital input is not set to 1. The error displayed on the LCD screen shows **POWER OFF/ON**

(M64=26). An external error can also be detected by the controller, with this system. It is important to notice that only DIN1 input can be used to perform this function. If the PWR instruction is given while parameter K33 contains the value 0 and that DIN1 is not set to 1, the controller also switches in **POWER OFF/ON** error mode (M64=26).

K	Name	Value	Comment
K37	Select cleared DOUT if error	-	Mask of the digital output (DOUT), that must be cleared when the controller is in error. When the controller is not in error any more, the digital outputs have DOUT value.
K33	Power on/off with DIN1	0 125	Enabled signal is necessary to power up the controller on DIN1. In this case this input must be at 1 when a PWR.<axis>=1 command is executed. If this input is cleared, the controller generates a **POWER OFF/ON** error (M64=26). Enabled signal not used (DIN1 is not taken into account). PWR.<axis>=1 command powers up the motor.

Remark: Refer to §13.4 for more information about the digital inputs and outputs.

12.4.4 Motor temperature, TEB time-out and analog encoder error check

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

- When the bit# 0 of parameter K141 is equal to 1, the motor temperature is checked by the DSC2P or/and DSC2V, and the **MOTOR OVERTEMP** error (M64=29) will appear in case of overheating. **(ONLY available on DSC2P and DSC2V)**.
- When the bit# 1 of parameter K141 is equal to 1, the **TIMEOUT TEB ERR** error (M64=56) will appear if a communication problem is detected by the controller on the TEB.
- When the bit# 2 of parameter K141 is equal to 1, the **ENCODER AMPLITUD** error (M64=20) or **ENCODER POS LOST** (M64=21) will appear if a problem is detected on the analog encoder as secondary encoder **(ONLY available on DSC2P and DSC2V)**.
- When the bits# 3, 4, 5 and 6 (of parameter K141) associated to its corresponding digital input, is equal to 0, the controller generates a **MOTOR OVERTEMP** error (M64=29) **(~~DSC2P~~ and ~~DSC2V~~)**.

K	Name	Value	Bit #	Comment
K141	Motor overtemperature protection	1 2 4 8 16 32 64	0 1 2 3 4 5 6	Enables test of motor overtemperature protection (PTC), connected on TSD signal **(Only available on DSC2P and DSC2V)** Enables time-out TEB error test Enables the control of the error on the analog encoder if secondary encoder **(Only available on DSC2P and DSC2V)** Enables test of motor's overtemperature protection connected to DIN1 (**~~DSC2P~~** and **~~DSC2V~~**) Enables test of motor's overtemperature protection connected to DIN2 (**~~DSC2P~~**, **~~DSC2V~~** and **~~DSCDM~~**) Enables test of motor's overtemperature protection connected to DIN9 (**~~DSC2P~~** and **~~DSC2V~~**) Enables test of motor's overtemperature protection connected to DIN10 (**~~DSC2P~~** and **~~DSC2V~~**)

Caution: On the DSCDM, the digital inputs and outputs are on the same pin. The pin must be selected in order to have an input **or** an output. **It is NOT possible to have both on the same pin.** To use a pin as a digital input, the bit corresponding to this input **MUST** be equal to 0 in parameter K171 (otherwise the hardware of the controller and the one of the user can be damaged).

Remark: For a dual axes controller (DSCDP, DSCDL and DSCDM), if parameter K141 is modified the SAV command must be executed on both axes.
DIN1, 2, 9 and 10 are standard inputs then be careful when choosing the type of temperature sensor (digital only). Refer to the corresponding **'Hardware Manual'** for more information about the inputs.

12.4.5 Vpower DC bus voltage

Available on	DSC2P	DSC2V	~~DSCDP~~	DSCDL	~~DSCDM~~

Parameters **K146, K147, K148** and **K149** are used to give the errors and warnings limits values of the input voltage.

K	Name	Value	Comment	Units
K146	Vpower undervoltage warning	0 > 0	Disables test on Vpower Warning activated: Vpower level is tested. If Vpower[V]*100<K146, the controller generates a **W UNDER VOLTAGE** warning M66=10)	[V]*100
K147	Vpower undervoltage error	0 > 0	Disables test on Vpower Error activated: Vpower level is tested. If Vpower[V]*100<K147 the controller generates an **UNDER VOLTAGE** error (M64=9)	[V]*100
K148	Vpower overvoltage warning	0 > 0	Disables test on Vpower Warning activated: Vpower level is tested. If Vpower[V]*100> K148, the controller generates a **W OVER VOLTAGE** warning (M66=4)	[V]*100
K149	Vpower overvoltage error	0 > 0	Disables test on Vpower Error activated: Vpower level is tested. If Vpower[V]*100 > K149, the controller generates an **OVER VOLTAGE** error (M64=6)	[V]*100

Remark: Each axis of the DSCDL has its own parameters K146, K147, K148 and K149 whose values is used for +Vpower and -Vpower.

- **For the DSC2P / DSC2V**, monitoring **M91** is used to indicate the DC input voltage level (Vpower).

M	Name	Comment	Units
M91	Vpower measurement	Gives the DC input voltage level (Vpower): Vpower[V] = M91 / 100	[V]*100

- **For the DSCDL**, monitorings **M91** and **M92** are used to indicate the positive and negative DC input voltage level (respectively +Vpower and -Vpower).

M	Name	Comment	Units
M91	+Vpower measurement	Gives the positive DC input voltage level (+Vpower): +Vpower[V] = M91 / 100	[V]*100
M92	-Vpower measurement	Gives the negative DC input voltage level (-Vpower): -Vpower[V] = M92 / 100	[V]*100

12.4.6 Fuse check

Available on	DSC2P	DSC2V	~~DSCDP~~	DSCDL	DSCDM

Parameter K140 allows the user to enable or disable the check of the fuse protecting the supply of the encoders

K	Name	Value	Comment
K140	Mask for fuse control	0 1	Enables the test of the fuse Disables the test of the fuse

Remark: On the DSC2P, this parameter can be used only from the DSC2Pxxx-xxxC version.

It is also possible to monitor the state of this fuse with monitoring M140.

M	Name	Value	Comment
M140	Fuse status	0 1	Fuse is not broken Fuse is broken

Remark: If the check of the fuse is enabled and if the fuse is broken, the **ENCODER FUSE KO** error (M64=35) will appear.

12.5 Errors and warnings handling

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Errors may be detected by the controller if a precaution limit is exceeded, or if a hardware failure occurs, e.g. Each error corresponds to a value of monitoring **M64**, readable by the user. It shows which error has taken place in order to handle it adequately. Some limits will give a **warning** before the error appears, to allow the user to solve the problem. Each warning corresponds to a value of monitoring **M66**, readable by the user. Messages also appear on the DSC2P and DSC2V LCD display (16 characters).

When an error is identified, the **Errors reference list** (§19.) is used to enable the user to identify the cause of the problem.

M	Name	Comment
M64	Error code	Number of the occurred error
M66	Warning code	Number of the occurred warning

12.5.1 Troubleshooting

Note: Troubleshooting is permitted only for ETEL technicians and agreed distributors!

Points to check in the **Errors reference list** are indicated as follows:

Enc = **x** Error may be due to the **encoder** and its cable.
Mot = **x** Error may be due to the **motor** and its cable.
Hrd = **x** Error may be due to the **hardware of the controller**.
Hrd Error may be due to the a part of the hardware of the controller (F7= fuse7, DSP = Sharc,...)
K value(s) Error may be due to a bad setting of the listed parameter(s).
PS = **x** Error may be due to the **power supply** (DSO-PWS).
TEB = **x** Error may be due to the **Turbo-ETEL-Bus** communication protocol.
EBL2 = **x** Error may be due to the **ETEL-Bus-Lite 2** communication protocol.
Other Error may be due to the reason described in the cell.
SW Res = **x** It is **possible to reset** the error by software (RST command).
HW Res = **x** It is **recommended or compulsory to reset** the error by hardware (RSD command or switch off/on)
Brk = OFF/ON OFF or ON means that this error activates or deactivates the dynamic braking (when used).

Example:

M64	Displayed message	Comment	Enc	Mot	Hrd	K	PS	TEB	EBL2	Other	SW res	HW res	Brk
2	OVER CURRENT1	The measured current in phase 1 is greater than K83		x	x	K60, K80, K81, K82, K83, K98					x		OFF
3	OVER CURRENT2	The measured current in phase 2 is greater than K83		x	x	K60, K80, K81, K82, K83, K98					x		OFF
4	I2T ERROR	This occurs when M67 becomes greater than K85. This is a power protection (of the motor and/or the controller)				K1, K2, K4, K52, K53, K56, K84, K85				Friction / duty cycle	x		OFF
5	OVER TEMPERAT	The temperature of the controller is greater than 70°C. This is measured by a thermostat mounted on the heat sink.								Heat evacuation	x		ON

12.5.2 Reset errors: RST and RSD

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

RST command (**R**e**S**e**T**) **resets most of the errors** that can happen in the position controller.
RSD command (**R**e**S**et **D**rive) resets the hardware board.

Remark: RSD command consists in switching off and on the controller, so if an autorun sequence exists (label n°79), it will start again after executing RSD command.

In a sequence, these commands are generally placed in the labels n°80 which is the label in which the program execution goes on in case of error.

Command RST, used with a single axis, starts automatically the instruction CPE described underneath.

Command format	Comment
RST.<axis>	Resets the error flags of the controller (bit# 10 of SD1)
RSD.<axis> = 255	Hardware resets of the controllers.

Example:

If during a movement, the position error x_e becomes too big. The controller switches then to error mode. The power is cut off, the LED lights red and the LCD display shows **TRACKING ERROR** error (M64=23). This error can be reset with one or other of the RST or RSD commands:

`RST.1` The error is reset, the LED lights green and the LCD screen displays for example **DSC2P READY**, as long as the cause which has produced the error is not there any more. The motor position is kept and the instruction PWR.1=1 is sufficient to reset the motor under control.

`RSD.1=255` The board is reset and the motor position is erased. A new initialization has to be redone in order to find again the absolute motor position.

12.5.3 Clear errors: CPE

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

CPE command (**C**lear **P**ending **E**rror) resets to 0 the 'pending' error bit when required.

Command format	Comment
CPE.<axis>	Pending error bit set to 1 in case of error

When an error happens, the execution of a sequence goes on automatically on label n°80 (if it exists) and an internal bit of the controller called 'pending' error, is set to 1 (bit# 0 of SD2 'Status Drive', alias of monitoring M61). As long as the 'pending' error bit is set to 1, it is forbidden for the controller to jump to label 80, so that it can execute the line following the label 80, and tries to correct the error. Without this 'pending' bit and as long as the error is there, the sequence would jump endlessly on label 80 and gets stuck there without going on with the execution of the rest of the sequence.

In the code contained in label 80, the error is generally handled with monitoring M64 and the RST command. This command, apart from resetting the error, resets the 'pending' error bit to 0. This way, when the following error takes place, it enables again the controller to jump to label 80.

12.5.4 Errors management with the μ-master

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

When an axis enters in error mode, the μ-master **(DSC2P and DSC2V only)** is immediately informed about it thanks to the information running on the frame of the TEB. In this case, the μ-master sends an error in all the other axes by sending the !ERR.! command in emergency record via the TEB. All the axes which were not in error will go in **EXTERNAL ERROR** error (M64=116). The !ERR.! command is sent only once by the μ-master. Only the execution of the RST command allows the μ-master to send again the !ERR.! command (when an axis is in error).

All this is possible as long as the TEB works properly. If there are problems on the TEB (error,...), it is possible to send the **TIMEOUT TEB ERR** error (M64=56) in all the axis if the bit# 1 of parameter K141 is set to 1. This error can be activated or deactivated with a parameter because if the user wants to work with the controller in single mode (without TEB connection), he does not want it to be in error. **In all the modes using the TEB, it is highly recommended to enable this error by setting the bit# 1 of parameter K141 to 1.**

During the initialization of the TEB, if the μ-master tests that the TEB is not at least once in 'Ready' mode (which means no error) in the first 5 seconds, it will enters in **SELFTEST TEB ERR** error (M64=54).

12.5.5 Dynamic braking

Available on	DSC2P	DSC2V	DSCDP	~~DSCDL~~	DSCDM

The dynamic braking is used to quickly stop the motor when an error arises and allows the user to protect the mechanical system during an emergency braking. It is based on the short-circuit principle at the terminals of the motor.

Caution: **This function cannot be considered as a safety system because there is no mechanical protection (relay), the function is managed by a software and a voltage is present at the motor's terminal! (continuous alternating switching of the transistors (GND - Vpower)). Due to this voltage, it is strictly forbidden to touch the system (controller, cable and motor) as long as the mains is not switched off.**

Bit 4 of parameter K32 is used to enable or disable the braking mode (refer also to §12.9 for more information about parameter K32). Once enabled (K32¦ = ∅x10), 2 types of error having an opposite effect on the braking, can arise. These two types of error (refer to §19. for the list) are: **'brake on'** activating the braking mode and **'brake off'** cancelling the braking. **The deactivation takes always priority over the activation.**

- If a 'brake off' error type arises, the braking will not be reactivated by a 'brake on' error type. It can be done only after a 'PWR.<axis>=1' or RSD.<axis>=255 command.
- If a 'brake on' error type arises, followed by a 'brake off' one, this second error type will not be displayed but the braking will be deactivated.
- The 'RST' command does not have any effect on the braking mode. The latter can be reset only after a PWR.<axis>=1 or RSD.<axis>=255 or by resetting and setting again bit# 4 of parameter K32. If the braking is activated, a vertical motor will not suddenly fall against the mechanical end stop after the 'RST' command.
- During the braking, there is a protection against the overcurrent in order to protect the controller and the system. The braking is deactivated (power bridge opened) when the current in one phase is bigger than 75% of the software overcurrent error (parameter K83). The braking is reactivated when the current in all phases is again smaller than this limit.
- During the braking, there is also a protection against the I2t in order to protect the controller and the system. The braking is deactivated when monitoring M67 (I2t value) is bigger than 75% of I2t error (parameter K85). The braking is reactivated when monitoring M67 is again smaller than this limit.

12.6 Basic reference mode (K61=1)

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Parameter **K61** allows the user to set the reference mode. The standard reference mode (also called normal or basic reference mode) is set when K61 = 1. This mode is the most frequently used mode (described up to now in this manual) To use it, the user needs to set:

- a type of movement defined by parameter K202 (or the MMD command which is an alias of parameter K202). Refer to §13.3.1.3 for more information.
- the references of the movement which are (refer to §12.10.3 for more information):
    - a final position to reach defined by parameter K210 (or the POS command which is an alias of parameter K210)
    - the maximum speed defined by parameter K211 (or the SPD command which is an alias of parameter K211)
    - the maximum acceleration defined by parameter K212 (or the ACC command which is an alias of parameter K212)
    - the jerk time defined by parameter K213 (or the JRT command which is an alias of parameter K213) when a S-curve movement is selected

All these references are calculated by the set point generator generating the movement trajectory. Refer to §13.1.1 for more information about the complete diagram of the regulation loop.

Remark: Refer to §13.2 for more information about the other values of parameter K61.

12.7 Initialization

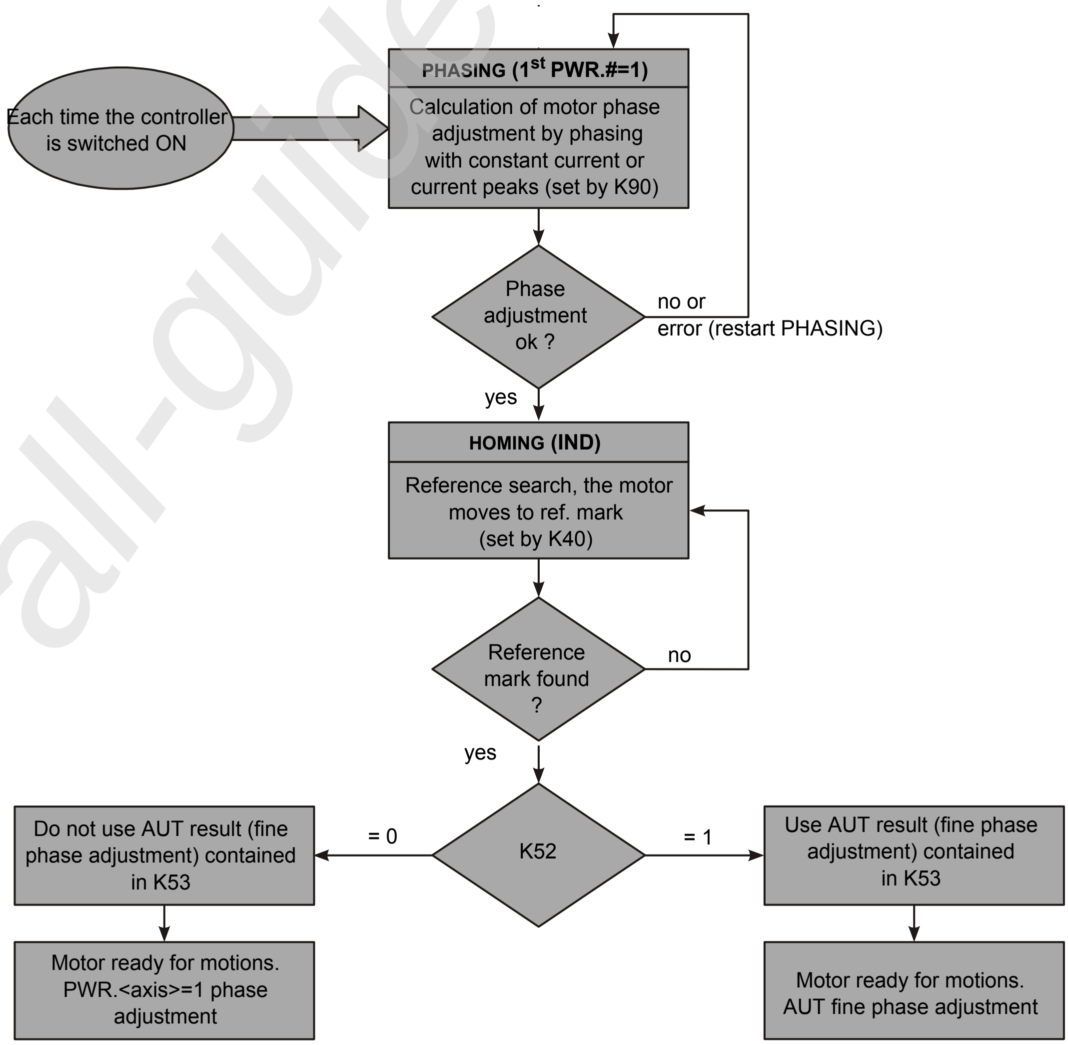
12.7.1 Phasing and homing basics

To have a direct drive motor working properly, it has to go through several processes:

- The phasing (parameter K90)
  After the controller's first 'Power On', we have to define the position of the motor's coils towards the magnets. This is made with a measurement.

- The homing (parameter K40)
  When the phasing is done, the motor is able to move. Then, we want to define the motor's real position with regard to its own physical environment. To do that we have to detect a mark. It can be a reference mark (also called index), a home switch, a limit switch or a mechanical end stop (refer to §12.9).

- Phase shift adjustment
  During the next 'Power On', it is necessary to do a new phasing. The obtained measurements can vary in a range of +/- 10% from an phasing to another (especially with an phasing by pulses). Of course, this would influence the system's performances in the same range. To avoid this problem, before the first initialization we have to make a phase shift adjustment.
  A phase shift adjustment is done during the first 'Power On' of the motor. This command will do a phasing then an homing and will store the coil position towards the magnets and the absolute position. Thus, after each homing we have the same value of phasing.



12.7.1.1 *Phasing use*

The motor must be initialized before powering it if an incremental encoder is used. This is the case in most controllers applications.

On the other hand we do not need a phasing if we use a single-phase motor or when we use an EnDat 2.1 encoder. In all these cases K90 = 0.

12.7.1.2 *Phasing purpose*

The Back-EMF is induced in the motor by the magnets' magnetic field. For an optimum working of the motor, the current into the coils has to be in phase with the B-EMF.

The phasing procedure determines the position of the motor's coils against the magnets poles with the aim of injecting the good current shape into the coils (it will give the maximum force to the motor). In fact, it avoids to calculate the initial position of the pointer in the look-up table of the phases commutation.

The look-up table of the phases commutation is a continuity of points making a sine, used to inject the currents into the coils in function of the position of the magnets.

The phasing is done at each first ’Power On’ of the motor or after each INI command.

After a phasing, the position of the motor against the magnets is precise enough to allow displacements. However, if the phasing is not correctly done, it is possible that the motor cannot give the maximum force. Sometimes the motor gives no force at all or worse, it gives a maximum force but in the wrong direction. In those cases, we will talk about phase shift adjustment and, in the last case, about an inverted phase shift adjustment. Next paragraph will show you the phase shift adjustment more precisely.

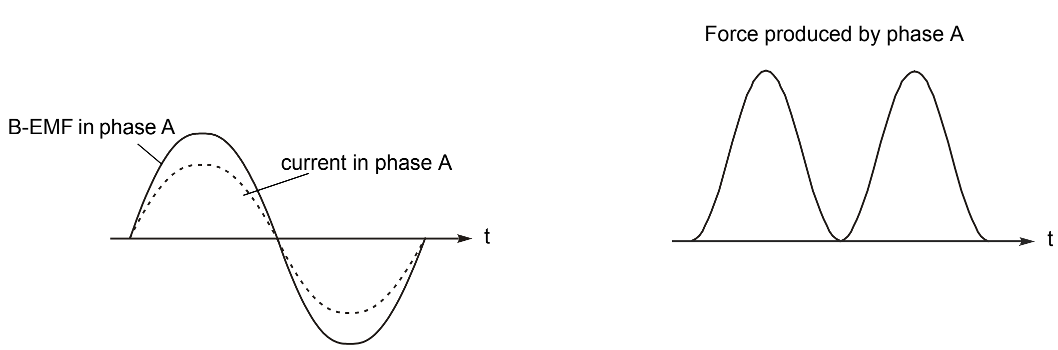
To resolve this problem, we have to do a fine phase shift adjustment (refer to §12.7.1.3 and §12.9 for more information).

Remark: With the **INI** (phasing previously called **INI**tialization) command, it is possible to restart a phasing cycle at anytime but the controller must be in ’Power Off’ mode first. After the INI command, the controller is in the same configuration than after the first ’Power On’. If K52=1 after the INI command, the fine phase adjustment (parameter K53) does not work because the IND command must be sent after the first ’Power On’ or the INI command.

Command format	Comment
INI.<axis>	Starts the phasing procedure

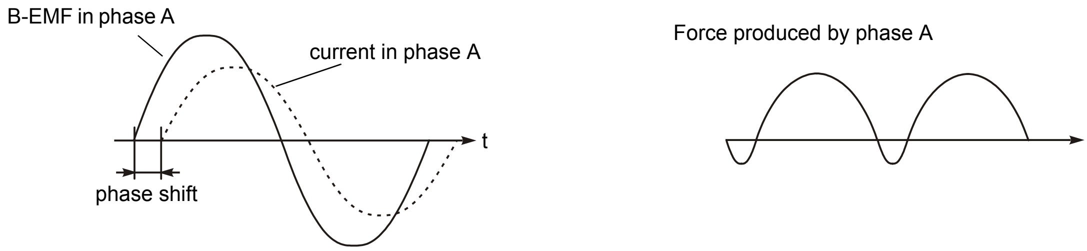
12.7.1.3 *Phase shift adjustment*

In order to obtain the **maximum force** from the motor, the current injected in the coils (phases) by the controller must be in phase with the motor induced voltage (B-EMF): ideally, this **phase shift should be null**. This offset correction is called **phase shift adjustment**, sometimes abridged **PSA** in this document.



Good phase shift adjustment

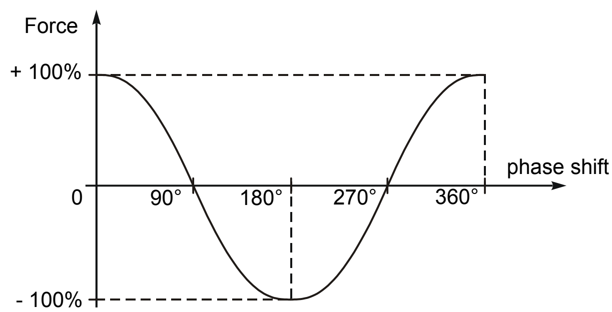
If the motor **PSA is not correct**, the motor will not give its maximum force. It can even happen that no force at all is supplied, or even worse, the force is provided in the wrong direction.



Bad phase shift adjustment

If the motor phase shift is only between 1° to 5° off zero, the motor will still deliver around 95% of its nominal force. A higher phase shift will significantly reduce the force.

You can see below that for a phase shift as bad as 90° to 270° off the correct value, a force is provided in the wrong direction (with 180° the maximum force is provided in the wrong direction). In this case, the position loop becomes unstable and the **motor speeds suddenly up in the wrong direction!** The expression used is: “there is an inverted phase shift adjustment”.



Phase shift error effect

This is the speed error v_e which pushes the motor to speed up in the case of an inverted phase shift adjustment. In a regular process, the position loop regulator usually tries to decrease v_e, but as in our case the force F provided by the motor is opposed to the reference force F_c, the motor will move in the direction that tends to increase v_e rather than to decrease it.

12.7.1.4 Commutation look-up table parameters

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The currents sent in the motor phases have a sinusoidal form. These types of current are calculated with the current reference generator. It is important to remember that the motor look-up table has a sinusoidal function period.

Parameter **K53** allows the user to have a repetitive adjustment after a homing:

- with an EnDat 2.1 encoder, parameter K53 is directly taken into account whatever the value of parameter **K52** (no homing is needed).
- AUT = 8 allows the user to find parameter K53 for all that K52 = 1 (refer to §12.8.1 for more information about the AUT command).

K	Name	Value	Comment
K52	Look-up table mode	 0 1	**After a homing:** Leave out parameter K53 Replaces the phase shift adjustment by parameter K53
K53	Look-up table phase adjustment	-	Motor phase shift adjustment value according to the ref. mark position

12.7.2 Phasing processes (K90)

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Several phasing processes may be performed. Parameter **K90** allows the user to choose the phasing type.

K	Name	Value	Comment
K90	Phasing mode and commutation	0 1 2 3 4 5 6	No phasing (with 1-ph. motor or EnDat 2.1 encoder) Phasing by current pulses (3-phase ironcore motors only) **(~~DSCDL~~)** Phasing by constant current in the motor phases Phasing with digital Hall effect sensor (mode 1) **(~~DSCDL~~)** Phasing with digital Hall effect sensor (mode 2) **(~~DSCDL~~)** Phasing and commutation with digital Hall effect sensor only **(~~DSCDL~~)** Small movement phasing

Remark: The depth 1 of parameter K90 is used for AUT.x=8.

12.7.2.1 K90=1: Phasing by pulses

Available on	DSC2P	DSC2V	DSCDP	~~DSCDL~~	DSCDM

K	Name	Comment	Units
K91	Phasing pulse; current level	Pulse amplitude when phasing by pulses	[ci]
K98	Phasing voltage rate	Bus voltage during the phasing by pulses	%

In this type of phasing (available only with 3-phase ironcore motor), we send some pulses with an amplitude given by parameter **K91**. During the phasing, the voltage present in the motor can be reduced with parameter **K98** (useful if you have a low inductance motor drove by a 300VDC position controller).

- If K98=100: full DC bus voltage (100%)
- If parameter K98 is included between 25 and 100 : the voltage is equal to: $\frac{DCbus \cdot K98}{100}$

Example:
K98 = 50
For a controller working at 300V, the voltage used for the phasing is 150V (half of the DC bus voltage).

The method by pulses has the advantage of practically not moving the motor (some microns) but it is less precise (about 10%) than a phasing by constant current.

12.7.2.2 K90=2: Phasing by constant current

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Parameters **K92**, **K93**, **K94** and **K97** concern only the phasing by constant currents.

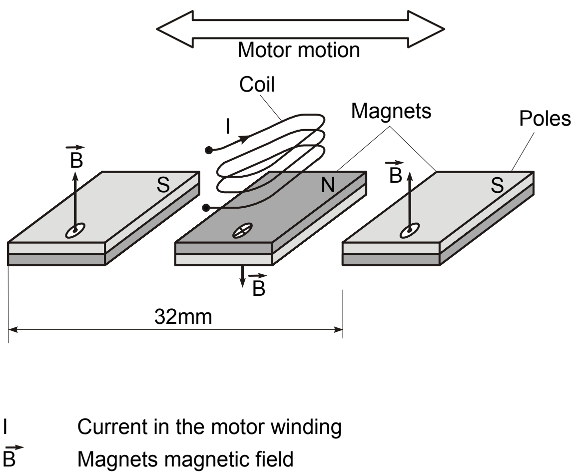
K	Name	Comment	Units
K92	Phasing motor current level	Maximum current level when phasing with constant current.	[ci]
K93	Phasing motor maximum adjustment	Pointers final position in the current loop look-up table.	-
K94	Phasing motor maximum time	Maximum time allowed before the motor is in a stable balance point. $\text{Time [s]} = K94 \cdot 10^{-3}$	[ms]
K97	Phasing motor minimum adjustment	Pointers initial position adjustment in the current loop look-up table.	-

Remark: Refer to §12.8.1 for more information about the autosetting with constant current.

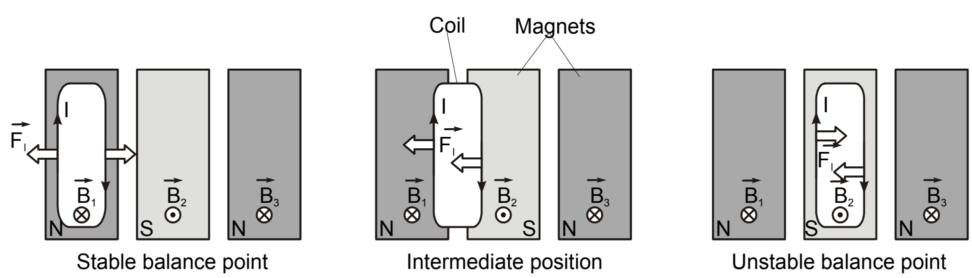
Phasing by constant current:

Supposing the user does a phasing by applying a constant current to one of the phase and a 0 current to the others. The Laplace force F1 generated by the current and the magnetic field will move the motor. When the powered coil reaches a magnet pole, this force becomes null and the motor stops.

There are two possible configurations: either the coil has reached a stable balance point, or a unstable one. In both cases the motor stops, but if the balance point is unstable, an external force, even weak, is enough to eject it from this position to the next stable balance point.
Stable and unstable balance points alternate every 16mm (for most ETEL's motors) if the magnetic period is equal to 32mm or every 32mm if the magnetic period is equal to 64mm.

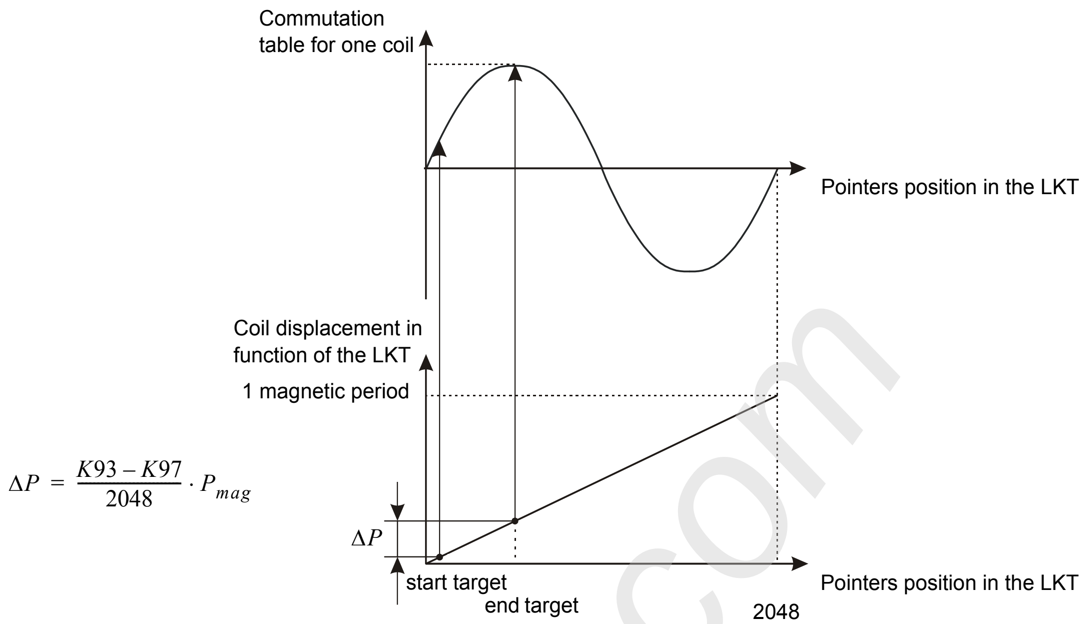


In fact, during a phasing, distinct constant current are sent into each phase of the motor instead of a single current in a single phase. In this case the principle remains exactly the same but the stable and unstable balance points are shifted with respect to the magnet pole according to the ratio between the injected currents.



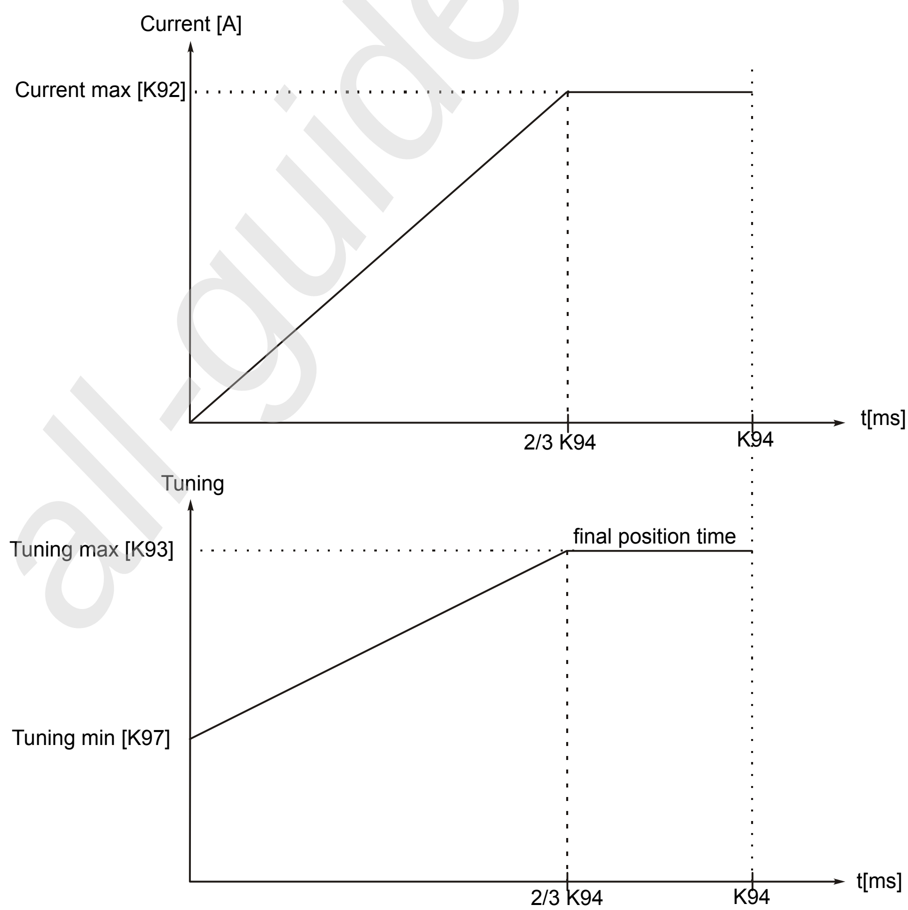
If a motor has reached an unstable balance point, it means that was already in this position at the beginning of the phasing. This case must be avoided because the motor will not move when constant current are sent and it will be in 'inverted phase shift adjustment' (force F generated by the motor in opposition to force reference Fc).

To avoid that problem, we move linearly in function of the time, the pointers position in the look-up table. Which amounts to move the position of the stable and unstable balance points against the magnets. To simplify the drawing, the look-up table of one phase is represented below:



Moreover, when the phasing has to be done near a mechanical end stop, those same parameters allow the user to move the final stable balance point if it is outside of the admitted stroke of the motor to bring it back inside.

To avoid any sudden movement, the constant current is not immediately applied to the phases but it is linearly increased until the maximum value set by the parameter **K92** is reached.



The time elapsed since the start of the phasing determines the end of the phasing by constant current. It is determined by parameter **K94** and is expressed in seconds:

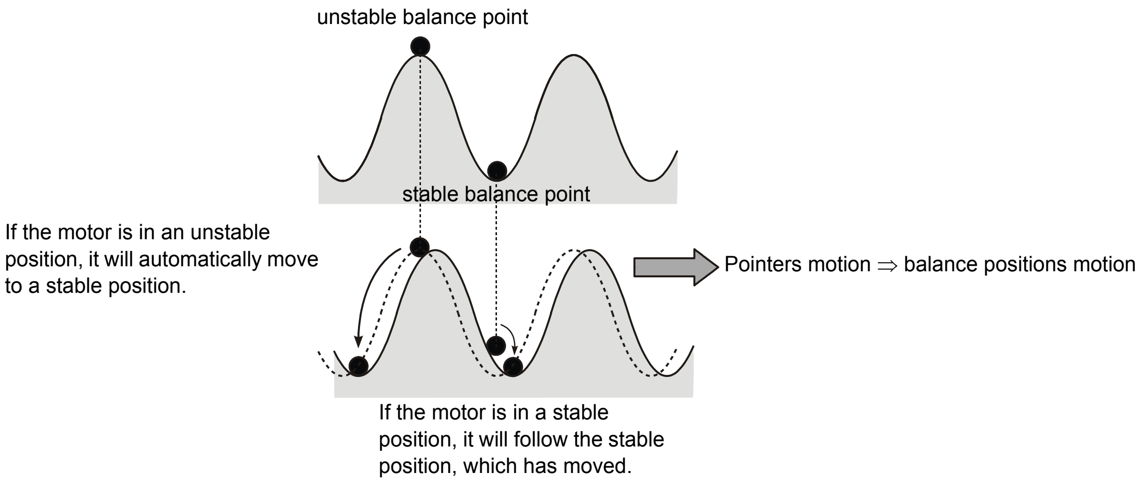
$$Time[s] = K94 \times 10^{-3}$$

We consider that the motor has reached a stable balance point position after this lapse of time.
To sum up, these are the three possible initial configurations:

Case 1: The motor is not in a balance point position (stable or unstable) and moves to a stable balance point position when the current in the phase is increased.

Case 2: The motor is in a stable balance point position and when the current increases, it follows the shift of the balance point position.

Case 3: The motor is in an unstable balance point position and when the balance point moves, the motor is confronted to the same problem as case 1 and moves when the current increases to a stable balance point position.



As we see, in all cases, the motor reaches at the end of the phasing a stable balance point position. Moving the pointers in the motor look-up table is useful when the motor is initially in the same configuration as case 3.

12.7.2.3 Phasing with digital Hall effect sensor

Available on	**DSC2P**	**DSC2V**	**DSCDP**	~~**DSCDL**~~	**DSCDM**

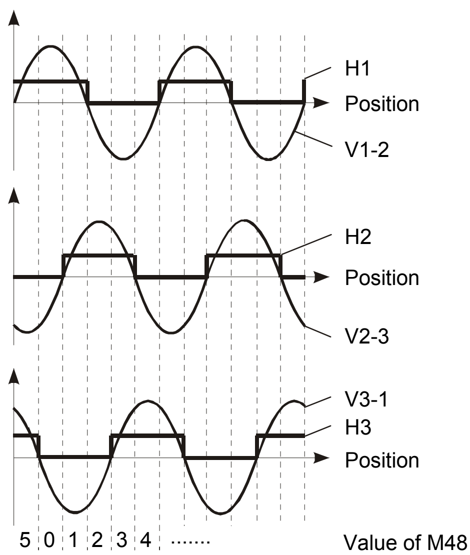
It is possible to interface digital Hall effect sensor only with three-phase motors. Three digital inputs (H1, H2 and H3), specific or not, can be used to connect a digital Hall effect sensor (except on the DSCDL) if K90 = 3, 4 or 5. The digital Hall effect sensor is connected to these 3 inputs (refer to the corresponding **'Hardware Manual'** for more information).

Caution: On the **DSCDM**, these 3 inputs are **only** dedicated to a digital Hall effect sensor and **cannot** be used as a standard digital input (DIN) as it is the case on the DSC2P, DSC2V and DSCDP.

Monitoring M48 allows the reading of the counter of the digital Hall effect sensor:

M	Name	Comment
M48	Digital Hall effect sensor signal	Gives the value of the digital Hall effect sensor's counter only if K90 = 3, 4 or 5 **(~~DSCDL~~)**

On the following graph, the Hall signals and the sine voltages between the motor phases (V1-2, V2-3 and V3-1) are displayed:



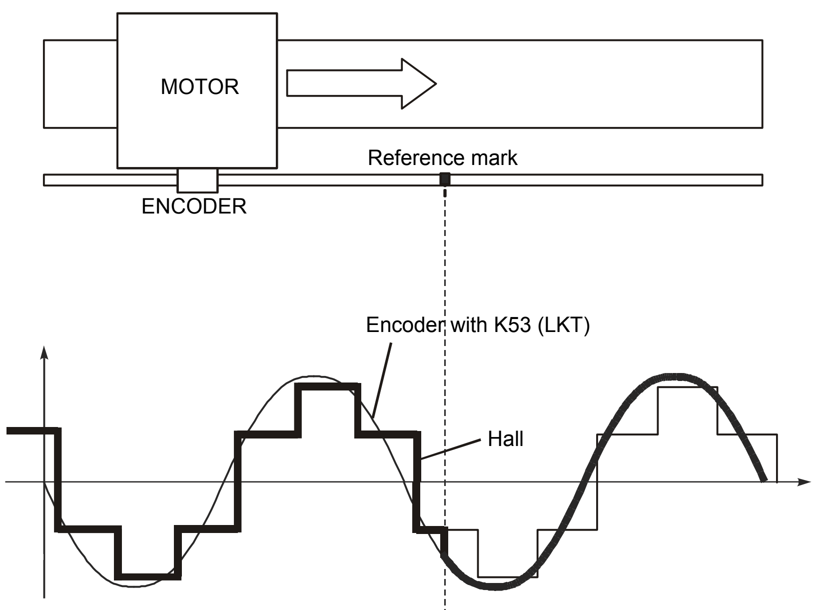
One period of the signals is divided into 6 different parts. These parts (as mentioned above) are associated to the following values of monitoring M48: 5, 0, 1, 2, 3 and 4 respectively.

Remark: M48 = 255 indicates that there is a cabling problem of the digital Hall effect sensor.

- **K90 = 3: Phasing with digital Hall effect sensor (mode 1)**

Available on	DSC2P	DSC2V	DSCDP	~~DSCDL~~	DSCDM

The commutation is done with a digital Hall effect sensor as far as the reference mark is found then the position encoder is used with the value stored in parameter K53 (phase shift adjustment).

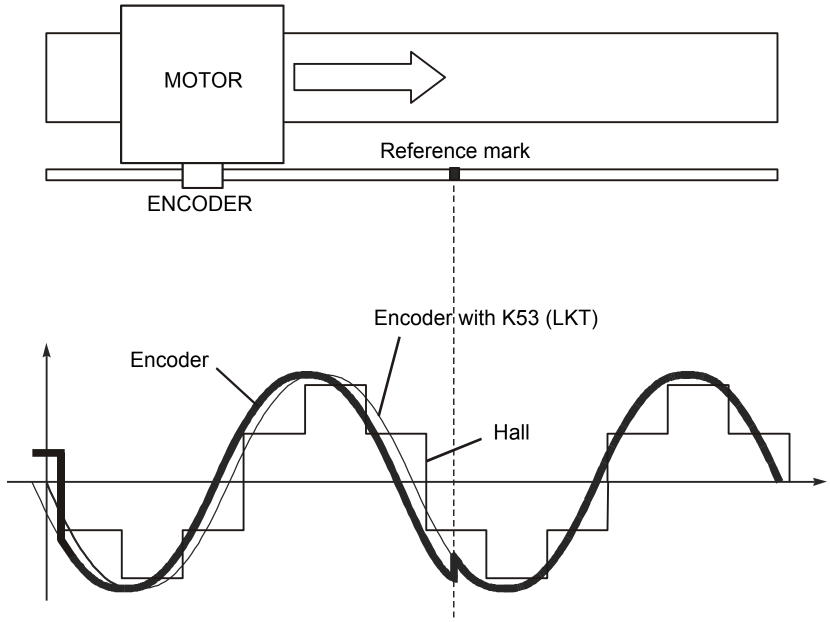


Caution: When parameter K90 or K68 is modified, the command AUT = 10 must be executed to re-calculate parameter K53.

- **K90 = 4: Phasing with digital Hall effect sensor (mode 2)**

Available on	DSC2P	DSC2V	DSCDP	~~DSCDL~~	DSCDM

The commutation is done with a digital Hall effect sensor up to the first edge and with the position encoder up to the reference mark is found. From then on, the commutation is done with the position encoder with the value stored in parameter K53 (phase shift adjustment).

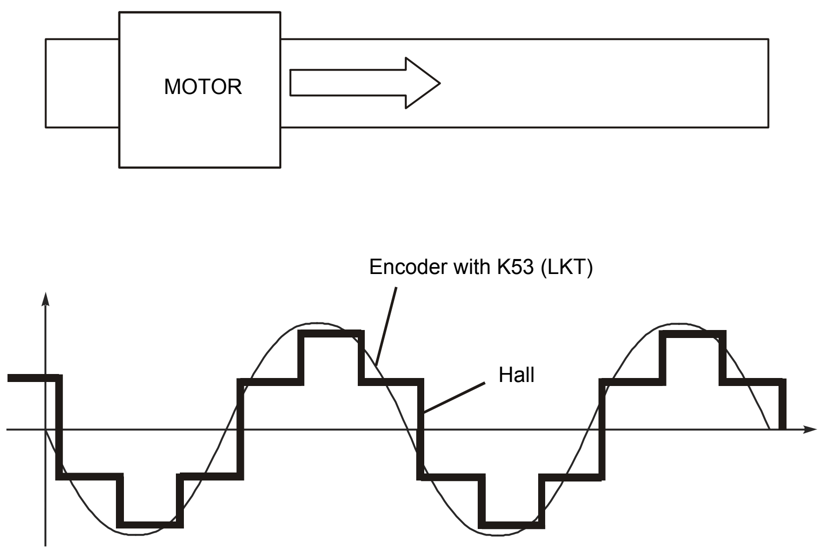


Caution: When parameter K90 or K68 is modified, the command AUT = 10 must be executed to re-calculate parameter K53.

- **K90 = 5: Phasing and commutation with digital Hall effect sensor only**

Available on	DSC2P	DSC2V	DSCDP	~~DSCDL~~	DSCDM

The commutation is only done with the digital Hall effect sensor.



Caution: When parameter K90 or K68 is modified, the command AUT = 10 must be executed to re-calculate parameter K53.

Remark: Refer to the corresponding **'Hardware Manual'** for more information about the connection of the Hall effect sensor.

12.7.2.4 K90 = 6: Small movement phasing

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

This type of phasing allows the user to initialize a motor by moving as little as possible. This type of phasing can be used for all kinds of ironless and ironcore motors. The main condition for making this phasing work is that the motor must be free of brake.

The motor is initialized by measuring the movement induced by a succession of current pulses. There are two parameters to set to achieve the phasing:

K	Name	Comment	Units
K91	Phasing pulse; current level	Pulse amplitude when phasing by pulses	[ci]
K101	Phasing time	Phasing time	[ms]

The current level used for the phasing may widely vary, mainly depending on the moving inertia. ETEL advises to begin the setting with a current level (given by parameter **K91**) at 5 % of the force/torque limit (parameter K60). The user should hear a low oscillation noise at the first 'Power On'.

Parameter **K101** is useful to slow down the motor between the different pulses when using a high inertia system. The default value of parameter K101 is 1000 [incr] = 1 [sec]. When K101 = 0, the total phasing duration is about 60ms, but a short phasing fits only a certain type of application and is not 100 % reliable with most systems.

When K90 = 6, three possible phasing errors may occur:

INITIALI LOW CUR (M64=153): this error appears when the movement is too small during the phasing process. The value of the current (parameter K91) is too low and should be increased in this case.

INITIALI HIGH CUR (M64=154): this error appears when the movement is too important (> ± 20% of the magnetic period) during the phasing process. The value of the current (parameter K91) is too big and should be lowered in this case.

INITIALI LOW TIME (M64=155): this error appears when the quality of the movements response is too bad to deduce a reliable phase shift adjustment. Typically, it will happen when the motor moves away from an unstable balance point during the phasing or when the load on the motor is high. In general, the best reaction to this error is to increase the value of parameter K101 (phasing time).

12.8 Autosetting

12.8.1 AUT command

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **AUT** command (**AUT**omatic setting) calculates automatically parameters **K80** and **K81** (current loop proportional & integral gain). Parameters **K56** (motor phases connection), **K53** (motor phase shift adjustment) and **K98** (DC bus voltage) are also set. This command **initializes** the motor. Therefore the adequate values must be conveniently stored in parameters K91 to K97.

Command format	<p1>	Bit #	Comments
AUT.<axis>=<p1>	1	0	Calculates the current loop parameters (K80, K81, K98), performs fine phase adjustment (parameter K53 if K52=1)and finds motor connection (parameter K56). Tunes the proportional and integrator gain of the current loop parameters K80, K81, and DC power voltage rate parameter K98.
	2	1	Searches motor phases and sets parameter K56.
	8	3	Sets parameter K53 (fine phase adjustment) if K52 = 1.

Remarks: The AUT command is only possible when the controller is in 'Power Off' mode (no current in the motor phases).
As the AUT command is a bits field, it is possible to execute, for example, AUT.1=10 (bits 1 & 3).
During the autosetting process, the message **'AUT CMD IN PROG'** is displayed.

Command format										
AUT.<axis>=<p1>		Calculated parameters					Read parameters			
<p1>	Bit #	K53	K56	K80	K81	K98(*)	Short movement phasing (K91 and K101 with K90:1)	Constant current phasing (K92, K93, K94 and K97)	Homing (K32, K40 to K47)	Encoder reading way inversion (K68)
1	0			OK	OK	OK				
2	1		OK					X		X
8 (**)	3	OK					X	X	X	X

(*): parameter K98 does not exist on the DSCDL.
(**): The initialization launched when the AUT.x=8 command is sent, is defined by depth 1 of parameter K90.

- If K90:1 $\neq$ 6, the phasing is executed by constant current
- If K90:1 = 6, the phasing is executed by short movement with parameters K91 and K101.

Remark: Parameter K80 can only be calculated with the AUT command for **ironcore motors**. With an ironless motor, parameter K80 value calculated with the AUT command can be completely false. Refer to §12.7.2.2 for more information about the phasing with constant current.

Parameter **K56** is first calculated when executing a phasing by constant current. Then, the motor moves of a distance equal to 45° of the electric period.

Remark: During the AUT command, the **TIMEOUT AUT CMD** error (M64=156) will appear if the motor did not move after the time defined by parameter K94 or if no value has been found for parameter K56.

Parameter **K53** (phase shift adjustment) is first calculated executing a phasing defined by the depth 1 of parameter K90, then a homing. When the index is found, the controller calculates the phase shift adjustment with respect to the index and this value is stored in the parameter K53. To perform all this, phasing parameters by constant current and homing parameters need to be correctly programmed.

Remark: With EnDat 2.1 encoder, there is no homing. Thus, there is the phasing and then the reading of the absolute position

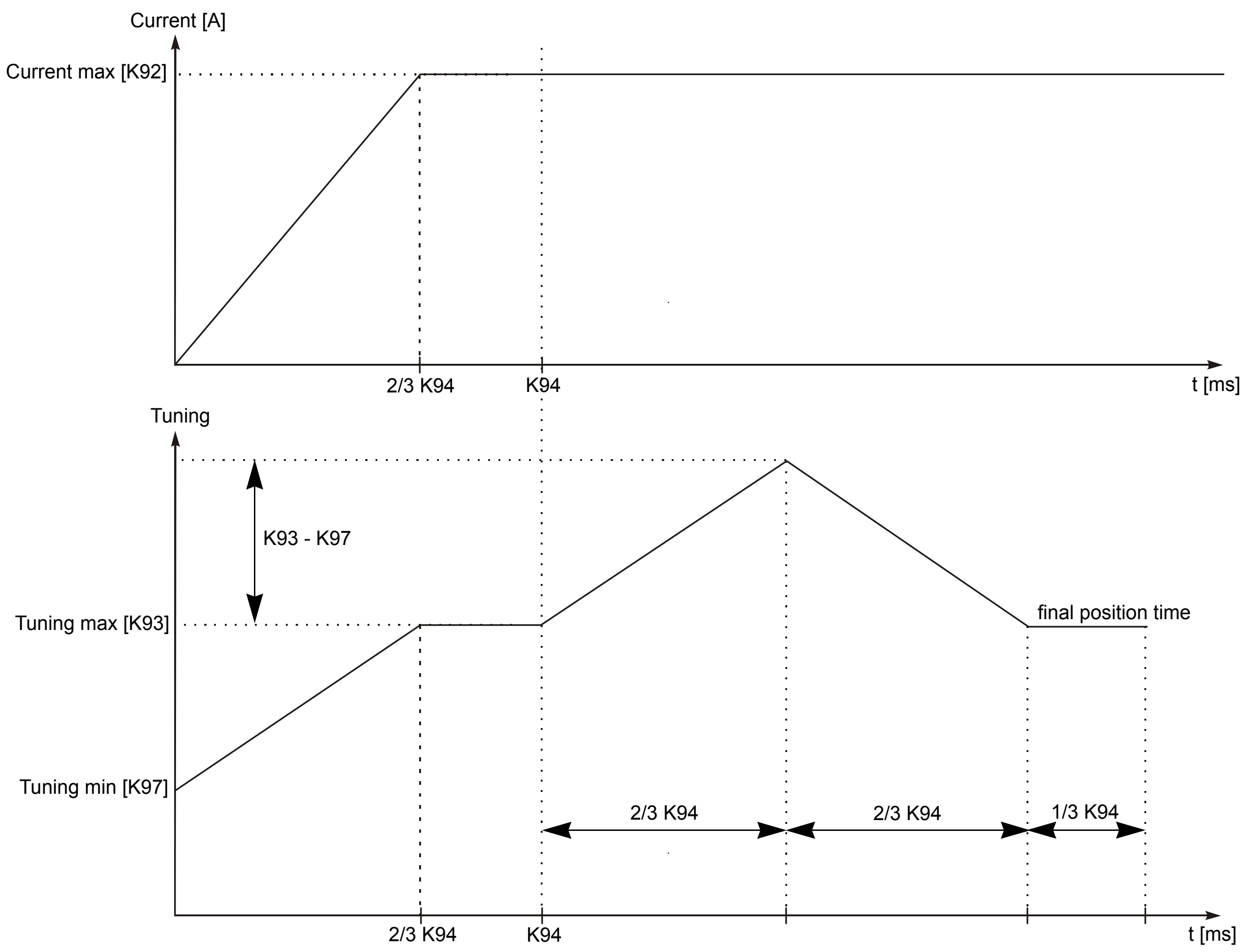
Parameters K80, K81, K98 are calculated first, then parameter K56, and finally parameter K53.

With the bit# 0 of parameter **K133**, it is possible to choose the principle for finding the fine phase adjustment (parameter K53) when using the AUT command. Parameter K133 can be used only when K90:1≠ 6. This method allows a better compensation of the friction forces.

- If bit# 0 = 0
    - Step 1: The look-up table pointer moves from parameters K97 to K93 during 2/3 of the time defined by parameter K94
    - Step 2: The process waits 1/3 of the time defined by parameter K94
    - Step 3: Calculation of the position value of the motor's coil towards the magnet (after the homing, this value is used for the calculation of the fine phase value (parameter K53))

Remark: Refer to §12.7.2.2 to see the graphical representation of the process.

- If bit# 0 = 1
    - Step 1: The look-up table pointer moves from parameters K97 to K93 during 2/3 of the time defined by parameter K94
    - Step 2: The process waits 1/3 of the time defined by parameter K94
    - Step 3: First calculation of the position value of the motor's coil towards the magnet
    - Step 4: Move away from parameter K93, in the same direction from a value given by parameters K93 to K97 during 2/3 of the time defined by parameter K94
    - Step 5: Move back to parameter K93 during 2/3 of the time defined by parameter K94
    - Step 6: The process waits 1/3 of the time defined by parameter K94
    - Step 7: Second Calculation of the position value of the motor's coil towards the magnet
    - Step 8: The average between the two values is used for the position value of the motor's coil towards the magnet (after the homing, this value is used for the calculation of the fine phase value (parameter K53)



The best calculation of parameter K53 after the AUT.x=8 command is achieved as follows:

- For horizontal movement (restoring force negligible) => K90:1.x = 2 (phasing by constant current) and K133.x = 1.
- For vertical movement (restoring force non-negligible) => K90:1.x = 6 (small movement phasing).

12.8.2 PWR command

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **PWR** command (**P**o**W**e**R**) **initializes** the motor then **supplies the motor phases** ('Power On') (it can also power it off). The phasing is done according to parameters K90 to K98. If a phasing has already been executed, it is not executed again and only the power in the phases is applied. When the PWR command is sent, the position loop must be correctly regulated because the motor is under control.

The PWR command must be executed each time the motor is switched on.

Remark: If parameter K33 contains the value 0, the power is only supplied if the digital input DIN1 is set to 1. If not, the controller enters in **POWER OFF/ON** error (M64=26).

When the power is cut off with the PWR.1=0 command, the motor position keeps on being calculated permanently. The motor position value is not erased.

Command format	<p1>	Comment	Read parameters
PWR.<axis> = <p1>	0 1	Motor power switched off. Phasing and switches on the power in the phases.	K90 to K98 and K33

Example:

`PWR.1=1` ;Phasing and then power in the phases, the motor is set in position, it is ready to make a move. '**Power On**' is displayed on the LCD screen of the controller.

`PWR.1=0` ;The power is switched off, it is possible to freely move the motor with the hand, but the position value is always calculated.

12.9 Homing

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

After phasing (refer to §12.7.1), a **homing** has to be carried out, moving the motor until it reaches a fix **reference mark** (also called **index)**. The motor movement can highly vary during a homing according to the type of encoder used. Some display several reference marks at regular distance from each other to know the motor absolute position (multi-reference mark encoder) and others display only one (mono-ref. mark encoder). Finally some encoders have no ref. marks, and homing has to be done against a fix **mechanical end stop** or against a **limit switch** or **home switch**. The **IND**.<axis> command starts the homing (it can be used only if the power is on and K61=1). With EnDat 2.1 encoder, there is no homing because it gives an absolute position.

The homing procedure (not available with EnDat 2.1 encoder) determines the motor absolute position, and secondly finds a constant phase shift adjustment value for each system powered (therefore, the motor's force will be the same after each homing).

- **Determination of the motor absolute position:**

For **multi-reference marks encoders**, the motor moves and knows its absolute position as soon as it finds two successive reference marks (they are coded). It means that the position origin, also called **'0 machine'**, is set. If during a next homing the motor uses two different successive reference marks, the '0 machine' remains set in the same position.

Remark: With a multi-reference marks encoder linear motor, the '0 machine' can appear outside the total motor stroke.

Immediately after having set the '0 machine', the controller adds up the value contained in parameter K45 (with EnDat 2.1 encoder, parameter K45 is added to the absolute position). This value is given in [upi]. This procedure places the '0 machine' in any position set by the user according to the application. Move the '0 machine' to the position of the motor is possible with the **SET** command if the power is on and K61=1.

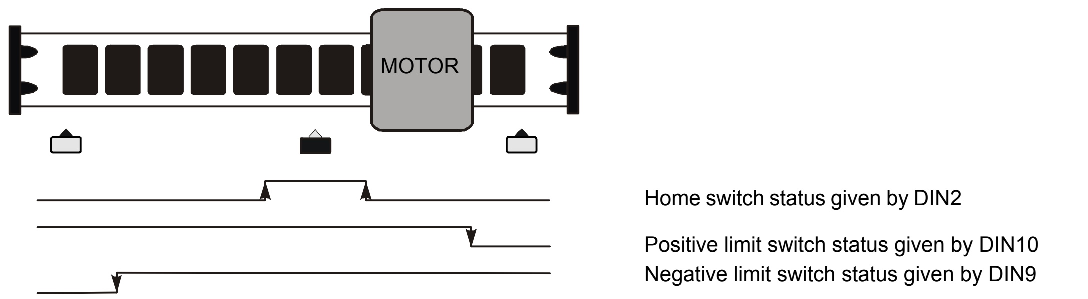
For **mono-reference mark encoders** or for a **mechanical end stop**, **home switch** or **limit switch** homing, the motor moves until it reaches the reference mark and places there the '0 machine'. Then the controller also adds up the value contained in parameter K45 [upi]. The motor does not exactly stop on the reference mark.

- **Calculation of the constant phase shift adjustment for each setting:**

The phasing determines the motor position according to the magnets by initializing the pointers in the current loop look-up table. This value can vary from one time to another. The **AUT** command and parameters K52 and K53 (refer to §12.7.1.4) enable the controller to avoid that. When the motor is **first** switched on, the AUT command allows a precise calculation of the motor phase shift adjustment **according to the reference mark** (AUT phase shift adjustment). This phase shift adjustment value is stored in parameter K53. Then each time the motor is switched on, the INI command calculates a phase shift adjustment value and the IND command moves the motor until the reference mark is reached. At that moment, the INI phase shift adjustment value is replaced (according to parameter K52) by the value contained in the parameter K53 (AUT phase shift adjustment). Therefore the motor phase shift adjustment is always the same each time the motor is switched on, and no longer depends on the phasing (refer to §12.7.1).

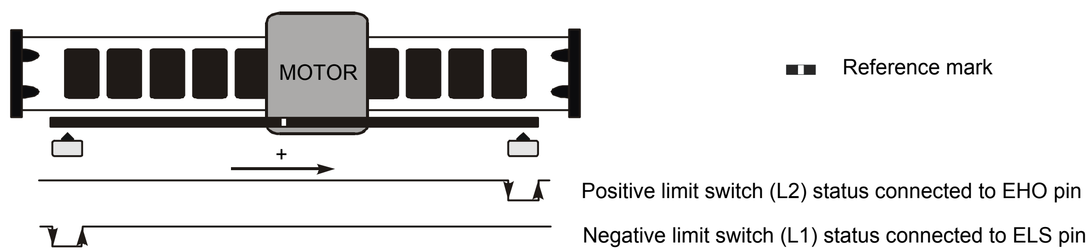
The controller allows the user to manage two types of home switch and limit switch:
- External home/limit switch (not integrated in the encoder). In that case, the positive limit switch (reached with a positive movement of the motor) must be connected to DIN10 (digital input), the negative limit switch (reached with a negative movement of the motor) must be connected to DIN9 (digital input) and the home switch must be connected to DIN2 (digital input). There can be only limit switches or only home switch or both together.

Caution: As there is no DIN2 on the DSCDM, it is necessary to define from which digital input the home switch signal is coming. To do so, bits# 2 and 3 of parameter K58 must be used to determine the origin of the home switch signal. In that case, bit# 0 and 1 of K58 must be equal to 0.

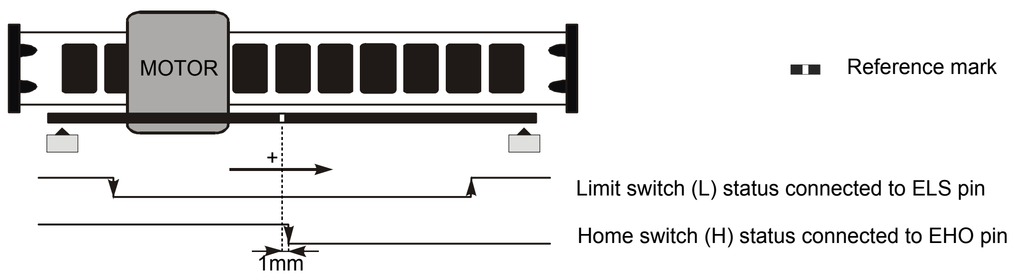


- Encoder's home/limit switch. In that case, they are integrated in the encoder and the controller allows the user to connect them directly to the EHO and ELS pins of the encoder's connector (refer to the corresponding **'Hardware Manual'** for more information about these pins).

  •If there are two limit switches, generally called L1 for the negative one and L2 for the positive one, they can be connected as follows:



  •If there are one limit switch and one homing switch(*), generally called L for the limit switch and H for the home switch, they can be connected as follows:



(*): The home switch is positioned within a gap of 1mm.

Bit# 1, 2 and 3 of parameter K32 allow the user to invert either the home switch or the limit switch (which means to choose the polarity) depending on the homing mode selected by parameter K40.

K	Name	Value	Bit #	Comment
K32	Limit switch and home switch inversion	1 2 4 8 16 32	0 1 2 3 4 5	Enables the errors when using the limit switches. Inverts the polarity of the home switch. Inverts the polarity of the limit switches from the encoder. Inverts the polarity of the limit switches from the digital inputs. Enables the use of the dynamic braking controlled by the transistor (**~~DSCDL~~**) Enables the limitation of parameters K60 and K31 according to the state of the digital inputs.

Remark: If the bit 0 is enabled, the controller gives a **SWITCH LIMIT** error (M64=30) when the limit switches are reached during a movement (except during the IND or SLS command). This error is generated only if the controller is in 'Power On'.
To reset the error **when K61=1**, the following procedure is recommended:
- Reset the error (with RST.<axis> command)
- Send the command: PWR.<axis>=1
- Move the motor out of the limit switch (with POS.<axis> command).

To reset the error **when K61≠1**, the following procedure is recommended:
- Clear bit# 0 of K32 (otherwise during the next 'Power On', the same error will come back)
- Reset the error (with RST.<axis> command)
- Send the command: PWR.<axis>=1
- Move the motor out of the limit switch (with POS.<axis> command)
- Set to 1 the bit# 0 of parameter K32

To activate bit 5 of parameter K32, it is possible to choose the state and the digital input that will activate the limitation. To do so, the depth 2 of parameters **K178** and **K179** is used:

- K178:2 defines the mask of the digital input that must be at 1 to activate the limitation (bit0=DIN1, bit1=DIN2, bit2=DIN3, bit3=DIN4, bit4=DIN5, bit5=DIN6, bit6=DIN7, bit8=DIN9 and bit9=DIN10). Refer to the corresponding **'Hardware Manual'** to know the number of the digital inputs present.
- K179:2 defines the mask of the digital input that must be at 0 to activate the limitation bit0=DIN1, bit1=DIN2, bit2=DIN3, bit3=DIN4, bit4=DIN5, bit5=DIN6, bit6=DIN7, bit8=DIN9 and bit9=DIN10). Refer to the corresponding **'Hardware Manual'** to know the number of the digital inputs present.

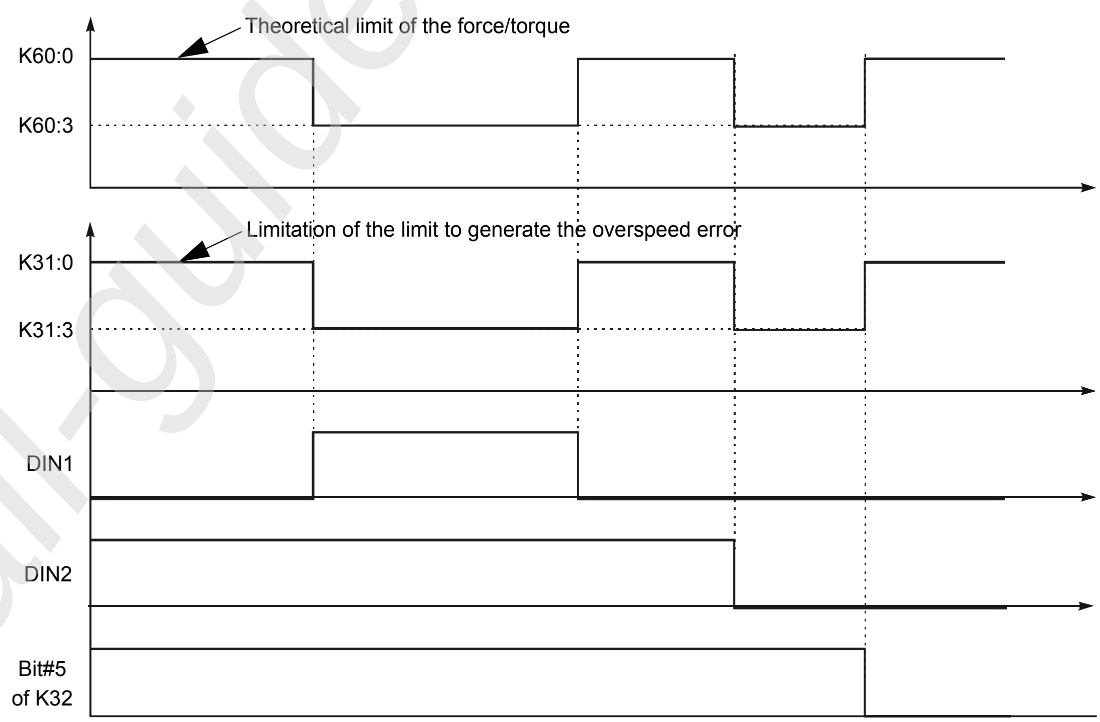
Bit# 5 of K32=1

- If at least one of the conditions defined by K178:2 and K179:2 is true, depth 3 of parameter K60 (K60:3) is kept as the limitation of the force/torque and depth 3 of K31 (K31:3) as the value of the overspeed.
- If none of the conditions defined by K178:2 and K179:2 is true, depth 0 of parameter K60 (K60:0) is kept as the limitation of the force/torque and depth 0 of K31 (K31:0) as the value of the overspeed.
- If K178:2=0 and K179:2=0 (state 0 of DIN2 activates the limit), depth 3 of parameter K60 (K60:3) is kept as the limitation of the force/torque and depth 3 of K31 (K31:3) as the value of the overspeed.

Bit# 5 of K32=0

- The limitation is not activated, depth 0 of parameter K60 (K60:0) is kept as the limitation of the force/torque and depth 0 of K31 (K31:0) as the value of the overspeed.

Example:
Here the limitation is activated if DIN1=1 (K178:2=1) or if DIN2=0 (K179:2=2):



When a motor is not positioned on a home switch, the latter is set to "0", and when it is positioned on it, it is set to "1". However these values can be inverted as well as for the home switch.

Parameter **K58** allows the user to define the type of home switch and/or limit switch used because it is also possible to select the limit switches of the encoder.

K	Name	Value	Bit #	Comment
K58	Home switch and limit switch mode	0 1 2 4 8	- - - Bit 2 Bit 3	Limit switches on DIN9 and DIN10 Limit switches L1 / L2 (L1 = ELS pin signal and L2 = EHO pin signal) Limit switches L / H (L = ELS pin signal and H = EHO pin signal) Home switch on DIN1 **(for DSCDM only)** Home switch on DIN9 **(for DSCDM only)**

Remark: If the hardware version of the controller is before DSC2Pxxx-xxxC-xxxA or DSCDP3xx-xxxE-xxxA version, parameter K58 is forced to 0.
If K32 = 1, the choice of the limit switches is set by parameter K58.
Parameter K58 can be used for the homing as well as for precaution. However it is not possible to manage two different limit switches (one for precaution, one for homing) at the same time.

Monitoring M44 shows the status of the home switch and limit switch signals coming from the encoder's connector (these signals are present only from DSC2Pxxx-xxxC-xxxA and DSCDP3xx-xxxE-xxxA versions).

M	Name	Value	Bit #	Comment
M44	Limit switch and home switch status	1 2	0 1	Indicates the state of the EHO pin signal Indicates the state of the ELS pin signal

The homing parameters define the type of reference mark used (parameter K40) as well as the behavior of the motor when, for instance, it reaches a mechanical end stop before a ref. mark (parameters K47 and K48). Thanks to these parameters the required speed and acceleration for the search of a ref. mark (parameters K41 and K42) are set as the movement during a homing is a trapezoidal movement. Current and maximum position error for an end stop detection are set via parameters K43 and K44.
By default (K100 = 0), after a homing, the speed (parameter K211), the acceleration (parameter K212) are modified by the values stored in parameters K41 and K42 and the movement type is modified to 1. However, if bit# 1 of K100 is set (K100=2) before sending the IND command, the speed, the acceleration and the movement mode at the end of the homing will be identical to the ones before it (the speed and the acceleration takes parameter K205 (CAM command) into account (refer to §13.3.7 for more information)).

K	Name	Comment	Unit
K40	Homing mode	Chooses the required homing mode (refer to §12.9.2).	-
K41	Homing speed	Motor speed during a homing.	[usi] [rusi]
K42	Homing acceleration	Motor acceleration during a homing.	[uai] [ruai]
K43	Max pos error for end stop detection	Maximum position error for mechanical end stop detection (K43 < K30).	[dpi] [rdpi]
K44	Max current for end stop detection	Maximum current limit for mechanical end stop detection (K44 < K60).	[ci]
K45	Offset for absolute position	Offset added to the position after having found the index or the absolute position with EnDat 2.1.	[upi] [rupi]
K46	Homing movement stroke	The motor will cover the distance defined by parameter K46 for K40=20, 21, 24 to 27, 36 to 39.	[upi] [rupi]
K47	Mvt. at init. on limit switch or end stop	Movement after homing on mechanical end stop or limit switch.	[upi] [rupi]
K48	Mvt to go out of an index or home switch	Movement to leave an index or a home switch if the motor is positioned on the top of it at the start or find it from the wrong way.	[upi] [rupi]
K100	Speed, accel. and MMD after homing	The speed, the acceleration and the movement mode (MMD) after the homing is identical to the ones before it (for K100=2)	-

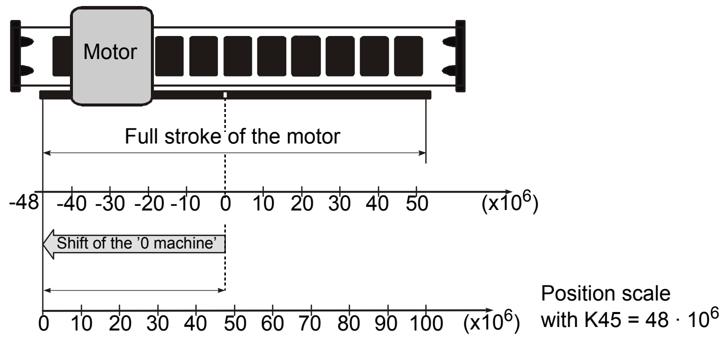
12.9.1 K45 parameter

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Once the homing procedure done (according to parameter K40) or the absolute position known (in case of EnDat 2.1 encoder), the position of the '0 machine' is determined and modified by an offset given by parameter K45 to place the origin position in the requested position.

Example:

A homing with a mono-ref. mark encoder linear motor is realized. The user wants to position the '0 machine' on the left, at the beginning of the motor stroke. He can do it entering the value $48{\cdot}10^6$ in parameter K45 as follows:



12.9.2 Homing modes

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

In the table hereafter, the symbol (OK) indicates the device used for the homing and the cross (x) indicates another existing device. For example, K40=6 stands for a homing with a positive movement towards a home switch, and a system provided with limit switches to detect stroke ends.

K40	Direction	Mechanical end stop	Home switch (DIN2) *	Limit switch (DIN) K58=0	Limit switch (L1/L2) K58=1	Limit switch (L/H) K58=2	Mono-ref. mark encoder	Multi-ref. mark encoder	Remark
0	+	OK							
1	-	OK							
2	+		OK						
3	-		OK						
4	+			OK	OK	OK			
5	-			OK	OK	OK			
6	+		OK	X	X	X			
7	-		OK	X	X	X			
8	+						OK		
9	-						OK		
10	+			X	X	X	OK		
11	-			X	X	X	OK		
12	+							OK	
13	-							OK	
14	+			X				OK	
15	-			X				OK	
16	+						OK+DIN2*		Particular homing in which the controller only selects a ref. mark if the input DIN2 is set to 1.
17	-						OK+DIN2*		
18	+			X	X	X	OK+DIN2*		Particular homing in which the controller only selects a ref. mark if the input DIN2 is set to 1.
19	-			X	X	X	OK+DIN2*		

K40	Direction	Mechanical end stop	Home switch (DIN2) *	Limit switch (DIN) K58=0	Limit switch (L1/L2) K58=1	Limit switch (L/H) K58=2	Mono-ref. mark encoder	Multi-ref. mark encoder	Remark
20	+							OK	The trip is defined by parameter K46
21	-							OK	The trip is defined by parameter K46
22									Immediate homing. The actual position is the ref. mark.
24	+						OK		The trip is defined by parameter K46
25	-						OK		The trip is defined by parameter K46
26	+	X					OK		The trip is defined by parameter K46 if mechanical end stop
27	-	X					OK		The trip is defined by parameter K46 if mechanical end stop
28 - 33	Reserved for future use								
34						X	OK		The trip is defined by the status of the limit switch and home switch
35						X	OK		The trip is defined by the status of the limit switch and home switch
36	+	X	OK						The trip is defined by parameter K46 if mechanical end stop
37	-	X	OK						The trip is defined by parameter K46 if mechanical end stop
38	+	X					OK		The trip is defined by K46 after having found the mechanical end stop
39	-	X					OK		The trip is defined by K46 after having found the mechanical end stop
40 - 45	Reserved for future use								

(*)Caution: As there is no DIN2 on the DSCDM, it is necessary to define from which digital input the home switch signal is coming. To do so, bits# 2 and 3 of parameter K58 must be used to determine the origin of the home switch signal. In that case, bit# 0 and 1 of parameter K58 must be equal to 0.

Remark: When a DSCDM is used with K40 = 2, 3, 6, 7, 16, 17, 18, 19, 36 and 37, the **HOME NOT POSSIBLE** error (M64=69) will appear if bits# 2 **and** 3 of parameter K58 are equal to 0. The same error will occur with K40 = 6, 7, 18 and 19 if bits# 0 and 1 of parameter K58 are equal to 0 while bit# 3 = 1.

12.9.3 K40 parameter:

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Parameter **K40** sets the **homing mode** according to the devices in the system. All homing modes are precisely explained below. Everything is summarized in the above-mentioned table.

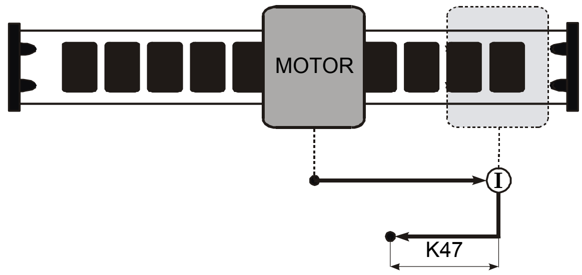
Here is a description of the symbols used in this paragraph:

Reference position
Reference mark
Zero machine
Motor position
Motor trip

Remark: In the following examples, the reading head is positioned in the middle of the motor and the limit switch covers the distance between the middle of the limit switch and the mechanical end stop.

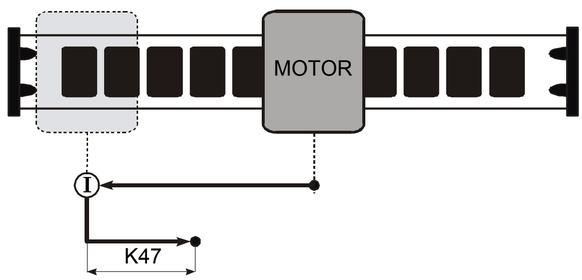
K40 = 0:

Homing against a mechanical end stop with a positive movement. After having found the mechanical end stop, the motor moves back the distance given by parameter K47.



K40 = 1:

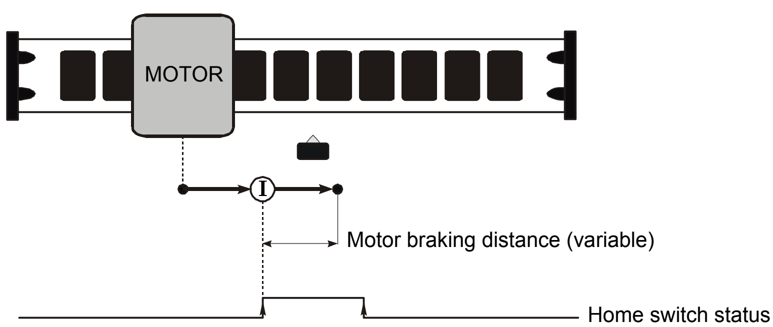
Homing against a mechanical end stop with a negative movement. After having found the mechanical end stop, the motor moves back the distance given by parameter K47.



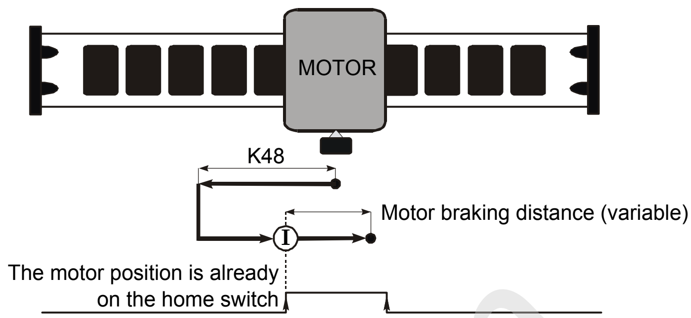
K40 = 2:

Homing on a home switch with a positive movement. To have the home switch always in the same position the motor must find it when moving in the determined direction. There are three possibilities:

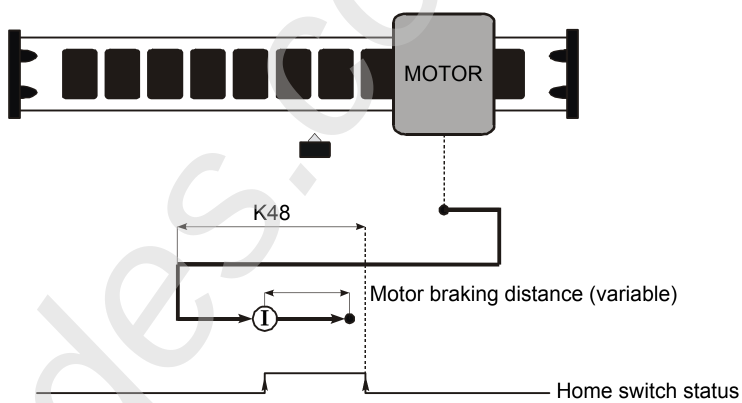
1. The motor is on the left of the home switch at the beginning of the homing. It moves and directly meets the home switch.



2. The motor is on the home switch at the beginning of the homing. It moves the distance given by parameter K48 in the opposed direction of the homing, to leave the home switch. Then it moves back towards the home switch in the right direction. Parameter K48 must be bigger than the home switch length.



3. The motor is on the right of the home switch at the beginning of the homing. After having found the mechanical end stop, it comes back. When it finds the home switch (in the wrong direction), it keeps moving the distance given by parameter K48 before changing direction a second time to find the home switch in the right direction. Parameter K48 must be bigger than the home switch length.



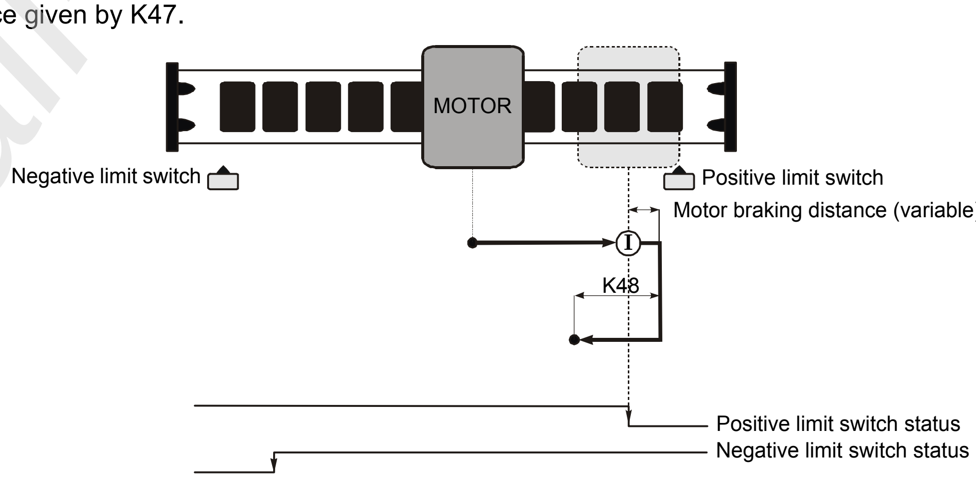
Remark: When a DSCDM is used with K40 = 2 and 3, the **HOME NOT POSSIBLE** error (M64=69) will appear if bits# 2 **and** 3 of parameter K58 are equal to 0.

K40 = 3:

Same as K40 = 2 but with a negative movement.

K40 = 4:

Homing on a limit switch with a positive movement. After having found the limit switch, the motor moves back the distance given by K47.



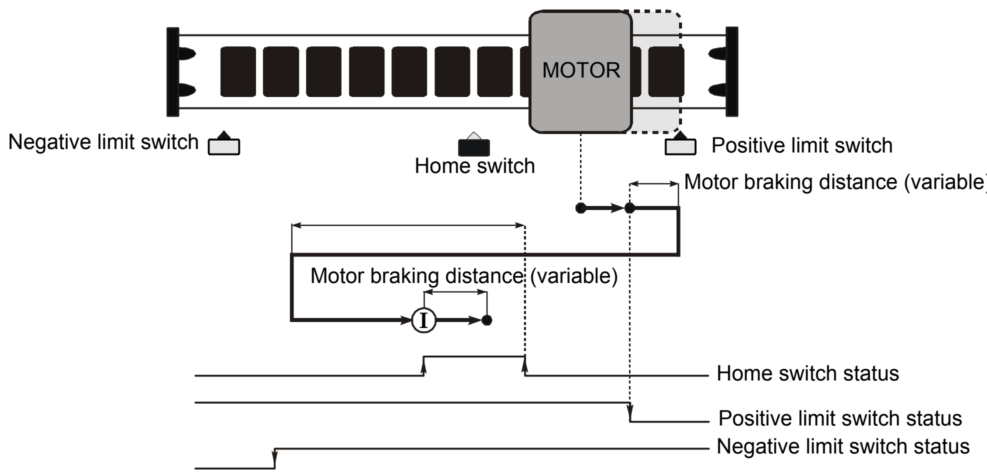
If at the beginning of the homing the motor is on the positive limit switch, the motor moves back the distance given by K48.

K40 = 5:

Same as K40 = 4 but with a negative movement.

K40 = 6:

Same as K40 = 2 but the motor changes direction when meeting a limit switch instead of a mechanical end stop. Only case 3 is shown. Parameter K48 must be bigger than the home switch length.



If the motor is on the home switch at the beginning of the homing, the motor moves of the value given by parameter K48.

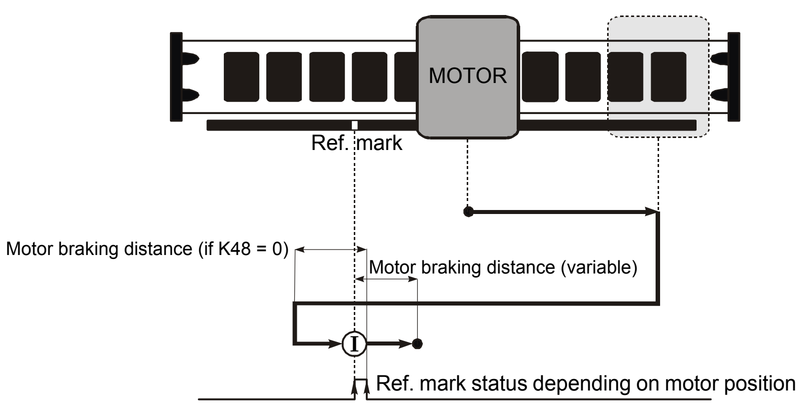
Remark: When a DSCDM is used with K40 = 6 and 7, the **HOME NOT POSSIBLE** error (M64=69) will appear if bits# 2 **and** 3 of parameter K58 are equal to 0. The same error will occur if bits# 0 and 1 of parameter K58 are equal to 0 while bit# 3 = 1.

K40 = 7:

Same as K40 = 6 but with a negative movement.

K40 = 8:

Same as K40 = 2 but with a mono-ref. mark with positive movement. As the width of a ref. mark is very small with respect to a home switch, the motor braking distance is big enough to move away from the ref. mark in case 3. So parameter K48 can be set to 0. Only case 3 is shown.

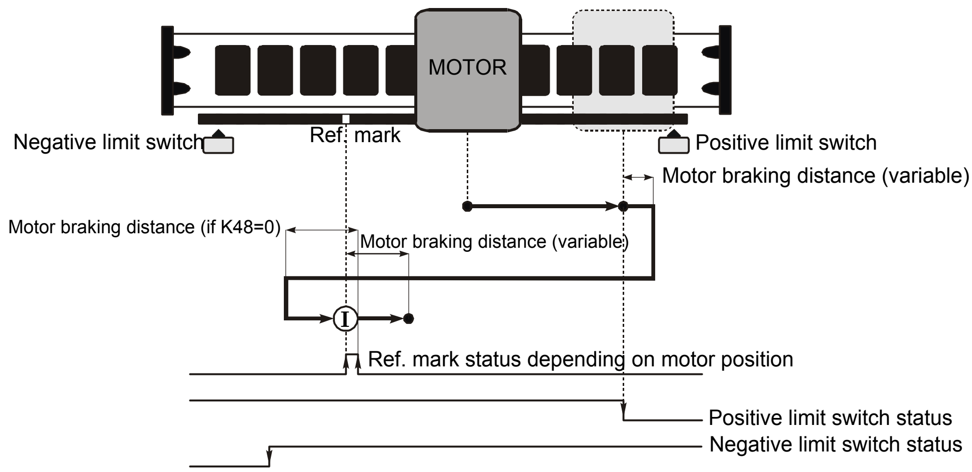


K40 = 9:

Same as K40 = 8 but with a negative movement.

K40 = 10:

Same as K40 = 8 but the motor changes direction when meeting a limit switch instead of mechanical end stop. Only case 3 is shown.



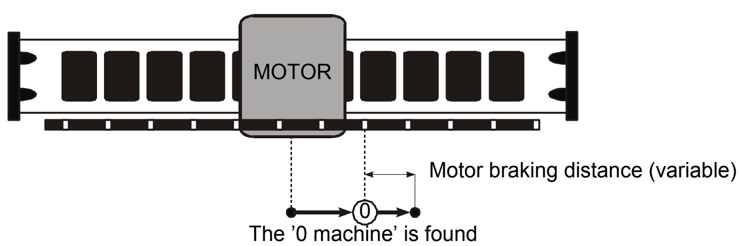
If the motor is on the limit switch, it moves back the distance given by parameter K48.

K40 = 11:

Same as K40 = 10 but with a negative movement.

K40 = 12, 13, 14 or 15:

Homing with a multi-reference mark is realized. The motor needs to find only two successive reference marks to determine its **absolute position** (the '0 machine' is always positioned at the same place regardless the two ref. marks found). If the motor finds one reference mark followed with a mechanical end stop (K40 = 12 or 13) or with a limit switch (K40 = 14 or 15) the reference mark is not considered and the motor starts the homing in the opposite direction.



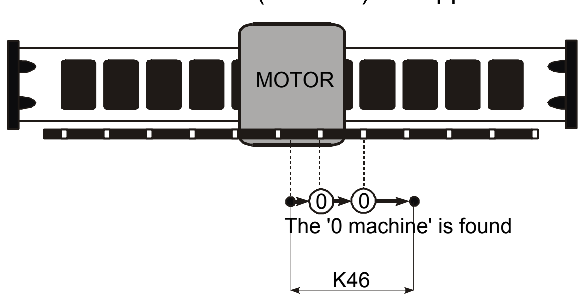
K40 = 16, 17, 18 or 19:

They are specific homings where a reference mark is detected by the controller only if DIN2 input is set to 1.

Remark: When a DSCDM is used with K40 = 16, 17, 18 and 19, the **HOME NOT POSSIBLE** error (M64=69) will appear if bits# 2 **and** 3 of parameter K58 are equal to 0. The same error will occur with K40 = 18 and 19 if bits# 0 and 1 of parameter K58 are equal to 0 while bit# 3 = 1.

K40 = 20:

Homing with a multi-reference mark. The motor moves the distance defined by parameter K46 to find 2 successive ref. marks. If the motor does not find two successive ref. marks (or a mechanical end stop) during this movement, the **MULT IDX SEARCH** error (M64=61) will appear on the LCD display.

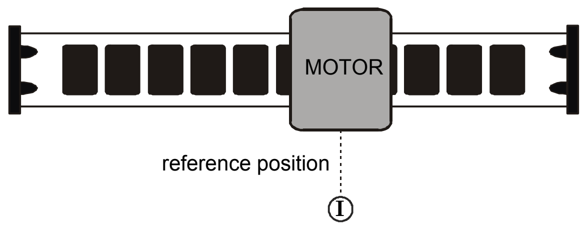


K40 = 21:

Same as K40 = 20 but with a negative movement.

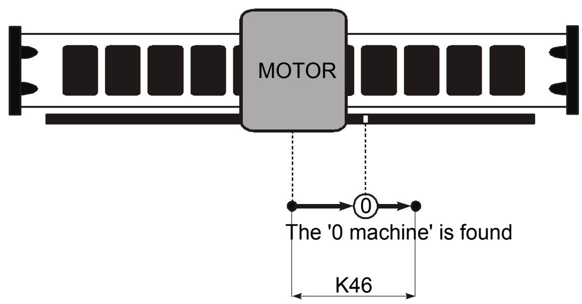
K40 = 22:

The present position is the reference position.



K40 = 24:

Same as K40 = 20 but with a single ref. mark. If the motor does not find the ref. mark (or a mechanical end stop) during this movement (defined by parameter K46), the **SING IDX SEARCH** error (M64=62) will appear on the LCD display.



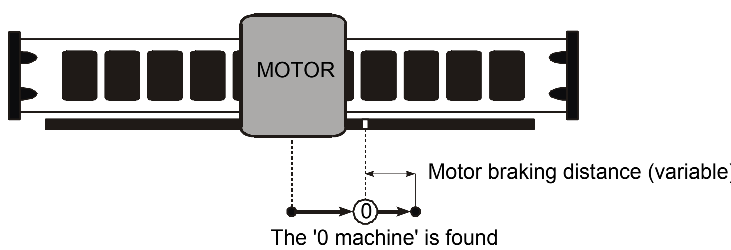
K40 = 25:

Same as K40 = 24 but with a negative movement.

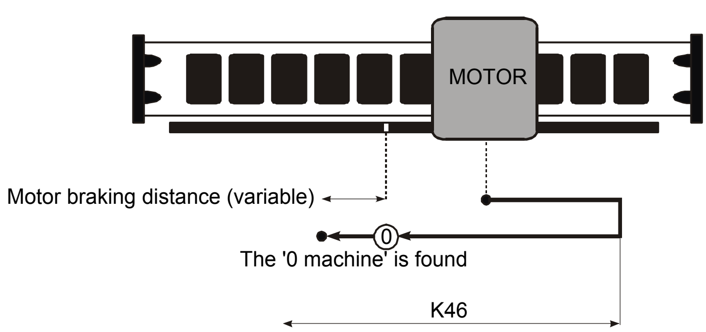
K40 = 26:

Homing on a mono-reference mark with a positive movement. There are two possibilities:

1. The motor is on the left of the index at the beginning of the homing. It moves and meets the index.



2. The motor is on the right of the index at the beginning of the homing. It moves up to the mechanical end stop and then comes back the distance given by parameter K46. If the index is not found during the return, the **SING IDX SEARCH** error (M64=62) appears.



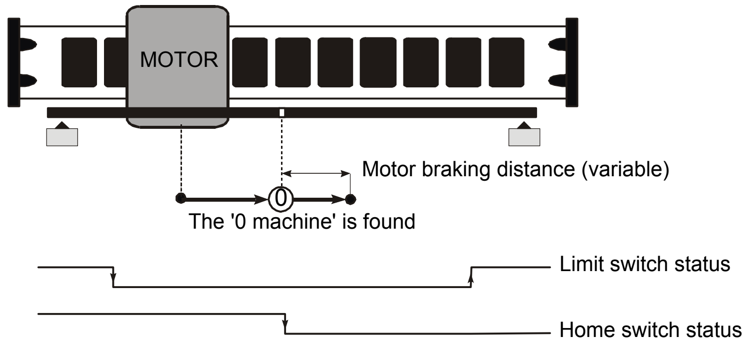
K40 = 27:

Same as K40 = 26 but with a negative movement.

K40 = 34:

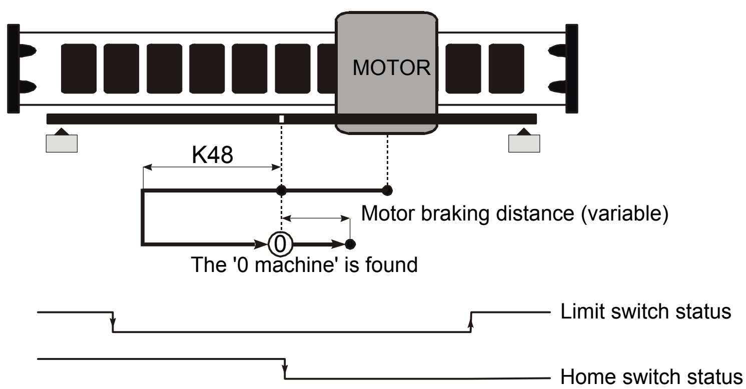
Homing on a mono-reference mark with a home switch and limit switch signal. It is possible only with controller from version DSC2Pxxx-xxxC-xxxA, DSCDP3xx-xxxE-xxxA, DSCDL3xx-xxxC-xxxA and DSCDMxxx-xxxB-xxxA and with certain types of encoder. There are three possibilities:

1. If the home switch signal is equal to 1 and on the left hand side of the reference mark, the motor moves in the positive direction up to the reference mark.



2. If the home switch signal is equal to 1 and on the right hand side of the reference mark (due to the gap: refer to §12.9 for more information), the motor moves in the positive direction. When the homing signal changes to 0, the movement goes it the opposite direction until the reference mark is found. Once it is found, a movement in the negative direction of the distance of parameter K48 is done and then another movement in the positive direction is done up to the reference mark.

3. If the home switch signal is equal to 0, the motor moves in the negative direction until the reference mark is found. Once it is found, a movement in the negative direction of the distance of parameter K48 is done and then another movement in the positive direction is done up to the reference mark. If the limit switch is found before the reference mark, the **SING IDX SEARCH** error (M64=62) appears.

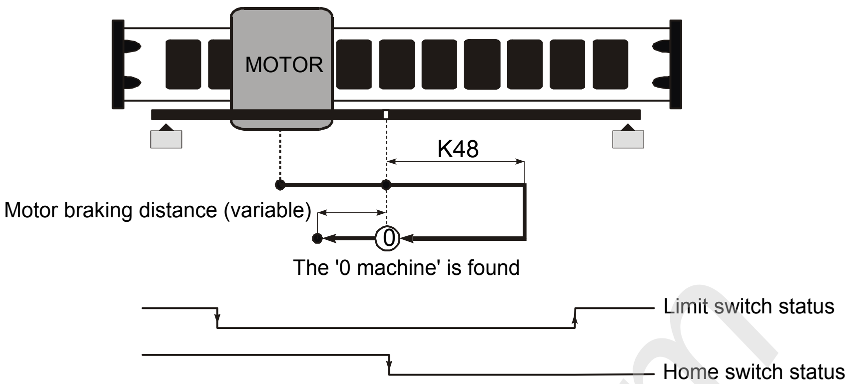


Remark: If K58≠2 when K40=34, the **HOME NOT POSSIBLE** error (M64=69) will appear instead of the **SING IDX SEARCH** error (M64=62).

K40 = 35:

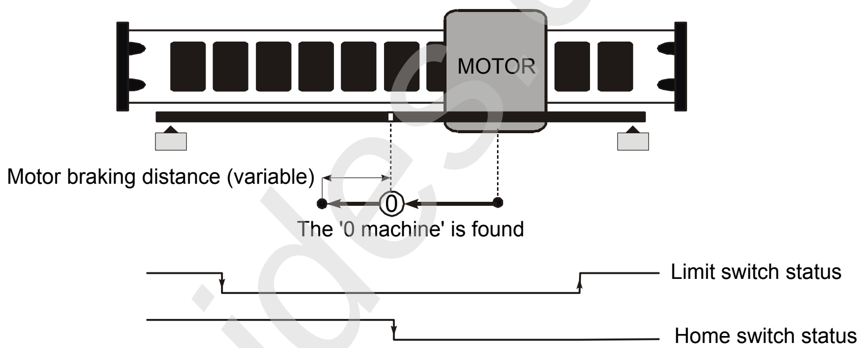
Homing on a mono-reference mark with a home switch and limit switch signal. It is possible only with controller from version DSC2Pxxx-xxxC-xxxA, DSCDP3xx-xxxE-xxxA, DSCDL3xx-xxxC-xxxA and DSCDMxxx-xxxB-xxxA and with certain types of encoder. There are three possibilities:

1. If the home switch signal is equal to 1 and on the left hand side if the reference mark, the motor moves in the positive direction up to the reference mark. When the reference mark is found, the motor still moves, in the positive direction, the distance given by parameter K48 and then the motor moves back in the negative direction up to the reference mark. If the limit switch is found before the reference mark, the **SING IDX SEARCH** error (M64=62) appears.



2. If the home switch signal is equal to 1 and on the right hand side of the reference mark (due to the gap: refer to §12.9 for more information), the motor moves in the negative direction until the reference mark is found.

3. If the home switch signal is equal to 0, the motor moves in the negative direction until the reference mark is found.

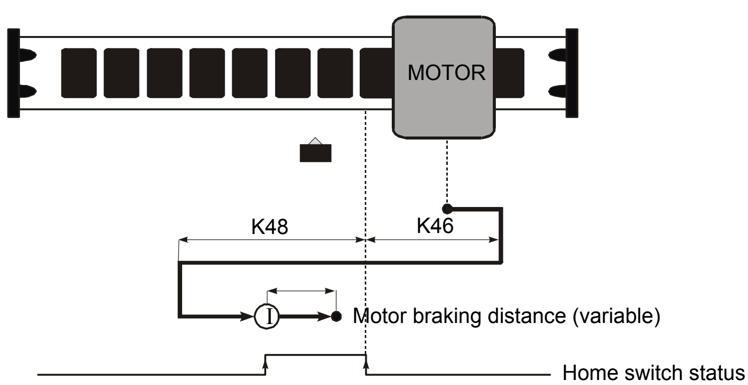


Remark: If K58≠2 when K40=35, the **HOME NOT POSSIBLE** error (M64=69) will appear instead of the **SING IDX SEARCH** error (M64=62).

K40 = 36:

Same as K40 = 2 except for case 3 which is different.

The motor is on the right of the home switch at the beginning of the homing. It moves up to the mechanical end stop and then comes back the distance given by parameter K46. If the index is not found during the return, the **SING IDX SEARCH** error (M64=62) appears. If the home switch is found (in the wrong direction), the motor moves the distance given by parameter K48 in the opposite direction of the homing to leave the home switch (parameter K48 must be bigger than the home switch length). Then, it comes back towards the home switch in the right direction.



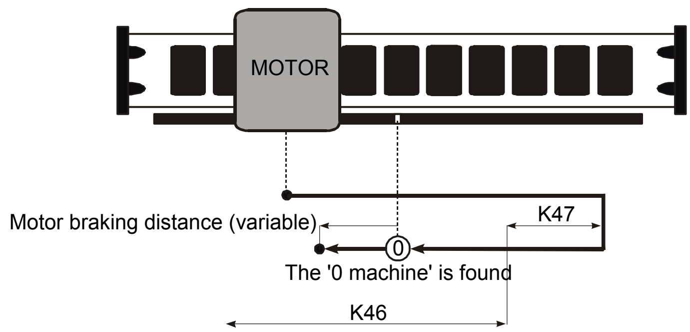
Remark: When a DSCDM is used with K40 = 36 and 37, the **HOME NOT POSSIBLE** error (M64=69) will appear if bits# 2 **and** 3 of parameter K58 are equal to 0.

K40 = 37:

Same as K40 = 36 but with a negative movement.

K40 = 38:

Homing on a mono-reference mark but only after having found the mechanical end stop. Once found, the motor moves back the distance given by parameter K47 and finds the reference mark with the distance given by parameter K46. If the index is not found within the distance given by parameter K46, the **SING IDX SEARCH** error (M64=62) appears (the distance given by parameter K46 must be smaller than the one between both mechanical end stop).



K40 = 39:

Same as K40 = 38 but with a negative movement.

12.10 Basic movements

A movement is made between two points, on a trajectory limited by the speed, the acceleration and the jerk. The basic movements features, described in this part of the manual, are using only **S-curve** and Rotary **S-curve** movements types. Advanced users may also refer to §13.3, for other movements types description.

12.10.1 SET command: zero machine

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **SET** command (**SET** '0 machine') defines the motor current position value.

Command format	<p1>	Comment	Units
SET.<axis> = <p1>	-2^{31} to $(2^{31}-1)$	'0 machine' positioning.	[upi], [rupi]

This command is generally used at the beginning of a sequence (after the homing) to place the '0 machine' in a different position than the reference mark because after a homing, the '0 machine' is automatically placed where the reference mark is (in case of a mono-reference mark encoder). This command must be used if the power is on and if K61=1.

Caution: After the homing, the '0 machine' is set on the reference mark but **the motor does not exactly stop.** The motor braking distance is determined by its speed when it crosses the reference mark and with parameters K41 and K42 (homing speed and acceleration). Therefore. it is really important to execute a POS command (POS.1=0, for instance) to place the motor in a precise pre-defined position before starting the SET command.

12.10.2 Linear or rotary movement

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Parameter **K202** (also available via the **MMD** alias) defines the **movement type**.

K202 = 1 (or MMD = 1): defines a linear movement (S-curve movement)

K202 = 17 (or MMD = 17): defines a rotary movement (rotary S-curve movement)

Remark: These two movements are available only with the standard reference mode (K61=1).

12.10.3 Movement trajectory parameters

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

In these parameters are memorized final position to reach, maximum speed, acceleration and jerk time values when an (S-curve) movement is executed. They define the movements trajectory and are available with K61=1.

K	Alias	Name	Comment	Units
K210	POS	Target position	Starts the movement (only depth 0) and gives the target position, with POS command	[upi], [rupi]
K211	SPD	Maximum speed	Pre-programmed maximum speed, with SPD command	[usi], [rusi]
K212	ACC	Max. acceleration	Pre-programmed maximum acceleration, with ACC command	[uai], [ruai]
K213	JRT	Jerk time	Pre-programmed jerk time, with JRT command	[sti]

Remark: ACC, SPD, POS and JRT alias use the same syntax than their corresponding parameter K. Refer to §6.3 for more information about the syntax and the possible operators.

The **ACC** command (**ACC**eleration) defines the maximum acceleration a_{max} during a movement.

The **SPD** command (**SP**ee**D**) defines the maximum speed v_{max} for a movement.

The **POS** command (**POS**ition) has two functions: to define the position x_{final} to reach during a movement and to start the movement.

The **JRT** command (**J**e**R**k **T**ime) defines the jerk time value for S-curve movements. This value corresponds to the extra number of times *a sti (166.67*µs for the DSC2P and DSC2V and 500µs for the DSCDP, DSCDL and DSCDM) that will take the S-Curve movement with respect to a trapezoidal movement at same speed, same acceleration and same final position. The JRT max value: Tmax = 500 x 166.67µs = 83ms for the DSC2P and DSC2V and 500 x 500µs = 250ms for the DCSDP, DSCDL and DSCDM.

Here is the formula giving the jerk J, according to the acceleration A and the jerk time T (with ISO units):

$$J \ [m/s^3] = \frac{A \ [m/s^2]}{T \ [s]}$$

Remark: The jerk (J) is not always the same. If the movement reaches a constant speed, the above-mentioned formula can be applied. However, if a constant speed is not reached, the following formula must be used:

$$\frac{A \ [m/s^2]}{T \ [s]} < J \le 2 \cdot \frac{A \ [m/s^2]}{T \ [s]}$$

Example:

`PWR.1=1`	;Current is supplied in the phases (after a phasing, if it was the 1st PWR).
`IND.1`	;The motor moves up to the reference mark.
`WTM.1`	;Waits until the movement is finished
`POS.1=0`	;The motor moves exactly on the reference mark.
`WTM.1`	;Waits until the movement is finished
`SET.1=100000`	;'0 machine' definition at 100000 [upi] of the reference mark position
`MMD.1=1`	;Selects S-curve movement.
`ACC.1=500000`	;Definition of a_{max}.
`SPD.1=200000`	;Definition of v_{max}.
`JRT.1=200`	;Definition of the jerk time: = 200x166.67µs = 33,2 ms for the DSC2P/DSC2V and 200 x 500µs = 100 ms for the DSCDP, DSCDL and DSCDM

Until now the motor is still positioned on the reference mark.

`POS.1=300000`	;The motor moves to position 300000 [upi] with a speed of 200000 [usi] and an acceleration of 500000 [uai].
`WTM.1`	;Waits until the movement is finished
`POS.1=10000`	;The motor moves to position 10000 [upi] with the same speed and acceleration than before.
`WTM.1`	;Waits until the movement is finished
`POS.1=-15000`	;The motor moves to an absolute negative position -15000 [upi] with the same speed and acceleration than before.
`WTM.1`	;Waits until the movement is finished.

12.10.3.1 Relative and absolute movements

The arithmetical operations +, -, * and / are allowed with the POS, SPD, ACC and JRT commands. It is then possible to realize not only absolute movements, such as the one described above, but also relative movements using the arithmetical symbols + and -.

`POS.1=300000`	***Absolute movement***:	The motor moves to the absolute position 300000. The movement can be positive or negative according to the motor position with respect to the 300000 point.

`POS.1+=300000`	***Relative movement***:	The motor moves in the positive direction of 300000 increments with respect to its actual position.
`POS.1-=300000`	***Relative movement***:	The motor moves in the negative direction of 300000 increments with respect to its actual position.

12.10.4 Rotary S-Curve movement

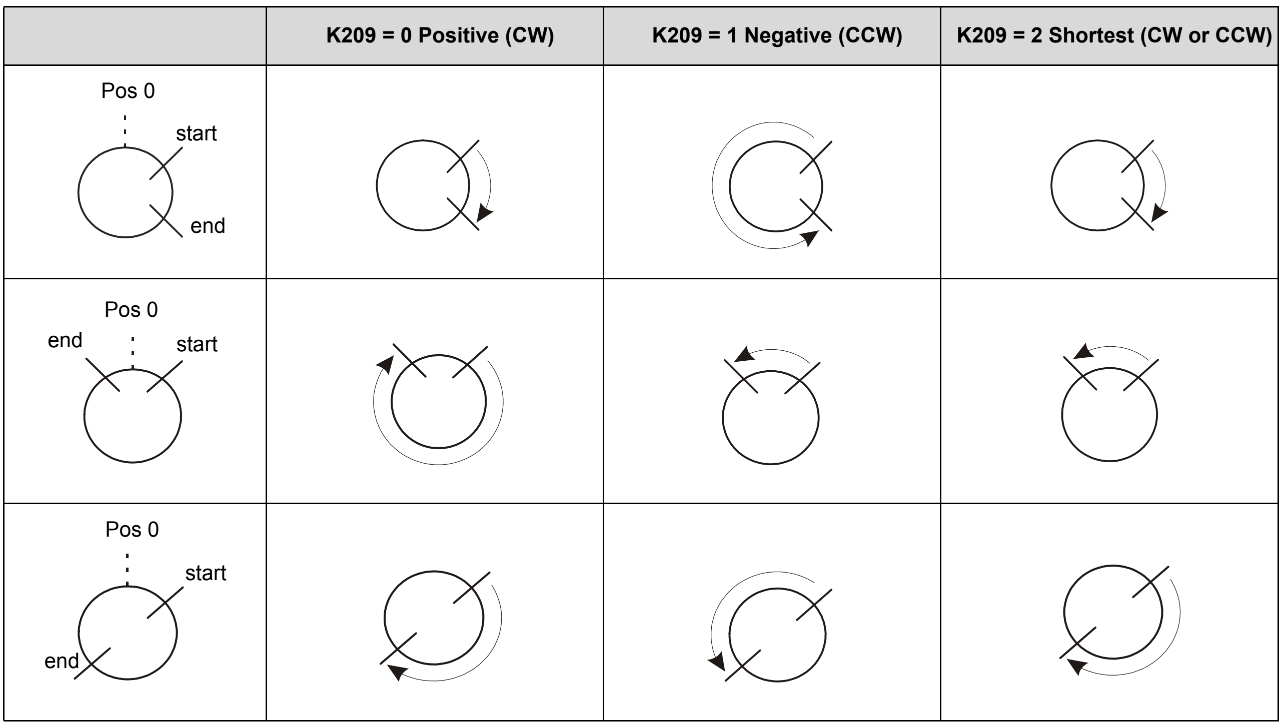
Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

When a rotary movement type is selected, it is also necessary to define the rotation way with parameter **K209** (CW or CCW) and the controller position counter's limit value with parameter **K27** (refer to §12.4.1).

Note: These parameters are also used for advanced rotary movements types (refer to §13.3 if you are an advanced use).

K	Name	Value	Comment
K209	Rotation way selection	0 1 2	Positive (CW) rotary movement Negative (CCW) rotary movement Shortest way to reach the target (CW or CCW)

12.10.4.1 Rotation way (parameter K209)



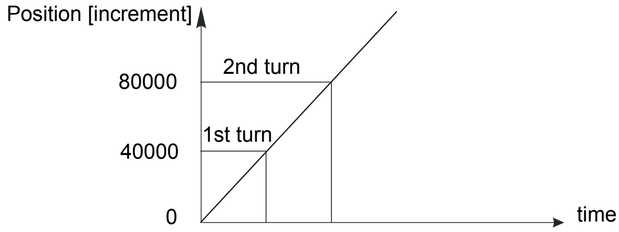
Caution: The following condition must be met so that a rotary motor works correctly:

$$\frac{K211^2}{2 \cdot K212} < 2^{30} - (2 \cdot K27) \quad \text{with} \quad \begin{array}{l} \text{K211: speed; SPD alias command} \\ \text{K212 : acceleration; ACC alias command} \end{array}$$

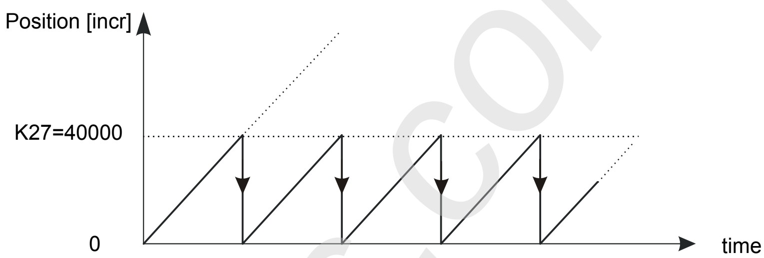
12.10.4.2 Position counter's limits (parameter K27)

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

A maximum limit is required for the position counter (refer to §12.4.1). For a rotary motor revolving at constant speed:



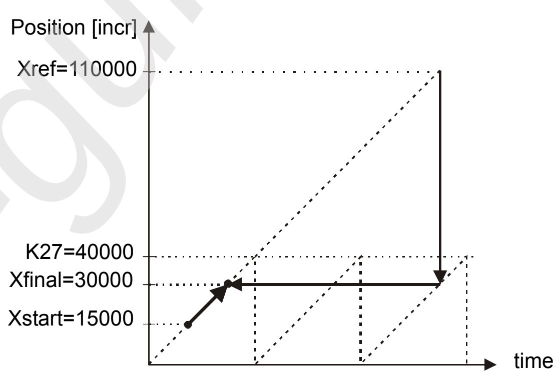
The measured position increases continuously and after a lapse of time, it will be so big that the controller's position counter will not be able to handle it (counter overflow).
To avoid it, as soon as the motor reaches the position programmed in parameter K27, the position counter is brought back to the value 0.
If K27= 40000 (user defined number of increments), the position measurement versus time will be:



Remark: It is recommended to set parameter K27 to the value corresponding to one complete machine cycle.
If a position target is given over the limit (|Xref| > K27), it is brought back to a value: 0<Xfinal<K27

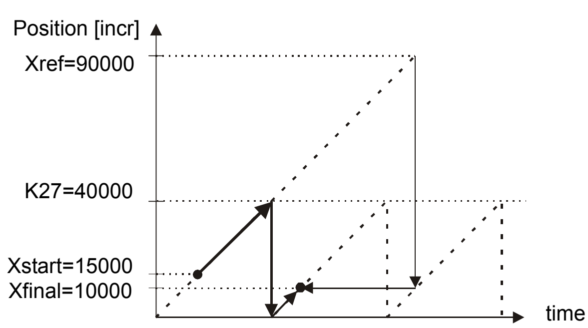
Example 1:

Xstart=15000
POS.1=110000 or POS.1+=95000 (means: Xref=110000 > K27=40000)
Xfinal=30000 (brought back to a value < K27)



Example 2:

Xstart=15000
POS.1=90000 or POS.1+=75000 (means: Xref=90000 > K27=40000)
Xfinal=10000 (brought back to a value < K27)



12.11 Monitorings

12.11.1 Monitoring registers

The monitorings M are exclusively used to monitor the controller's internal values, like motor speed, acceleration, current, etc...refer to §17.

12.11.2 LCD display

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

K	Name	Value	Comment
K66	LCD display mode	1 2 4 8 16 32	Displays normal information. Displays the temperature [°C] of the controller. Displays the amplitude and the index of the encoder's analog signals Displays sequence line number (in process). Displays the last message output by the optional board (DSO-XXX) message (refer to the corresponding **'User's Manual'** for more information) **(~~DSCDM~~)** Displays DC bus voltage Vpower [V]. **(ONLY available on DSC2P and DSC2V)**

Display modes (parameter K66):

The controller's LCD screen displays the controller's temperature, type of error, etc. These different modes are selected with parameter K66. The DSC2P LCD display has 2 lines of 8 characters. As there is no physical display on a dual axes controller, the user can use M95.<axis> (the conversion is automatically done by the DLLs) or can select 'scope drive LCD display' in the 'scope' menu of the scope tools (ETT) to display a software display indicating the error and warning messages.

In a **normal display mode** (K66=1), the controller's status appears on the LCD display. When the controller is switched on, it executes a self-test.

SELFTEST and the boot version (01.00B, eg.) appear (1.). Then, the axis number is given from the boot (2.). Then briefly, the type of controller (3.). Then, the controller tells if there is an optional board or not (4.). Then the firmware version (5.) is displayed. Then, the axis number is indicated (6.) as well as the baud rate of EBL2 (7). Finally, if no error has been found until then, DSC2P READY for example is displayed on the screen (8.), otherwise the corresponding error message appears (9.) and lights the SERVO ERROR red LED on the front panel of the controller.

Caution: Error and warning messages are only entirely displayed in normal mode (K66 = 1).

In the **temperature display mode** (K66=2), the controller's internal temperature in Celsius degrees [^{o}C] (10.) is displayed.

In the **analog encoder mode** (K66=4), the amplitude of the encoder's signal [Vptp] (11) is displayed.

In the **sequence display mode** (K66=8), the line number being executed by the controller (12.) is displayed.

In the **voltage display mode** (K66=32), the controller's internal DC bus voltage in [V] (13.) is displayed **(ONLY available on DSC2P and DSC2V)**

1.	SELFTEST 01.00B
2.	BOOTAXIS NUM 1
3.	DSC2P131 111B000B
4.	NO EXT BOARD
5.	DSC2P V1.00A
6.	DSC2P AXIS 1
7.	EBL2 115200Bps
8.	DSC2P READY
9.	ENC POS LOST
10.	DSC2P T = +40°C
11.	1.00 Vptp ■■I■■
12.	0023>
13.	DSC2P U=313 V

12.12 Controller software characteristics

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The following monitorings indicate useful data for the user (they cannot be modified).

M	Alias	Name	Value	Comment	Units
M70	-	Controller type	6 7 8 10	DSC2P / DSC2V controller DSCDP controller DSCDL controller DSCDM controller	-
M71	-	Controller boot software version	-	Number of the Boot software version installed in the controller (same format as monitoring M72).	-
M72	VER	Controller firmware version	-	Number of the firmware version installed in the controller.	-
M73	SER	Controller serial number	-	Controller's serial number	-
M76	-	Optional board type	 0 4 7 16 24 32	Optional board installed in the controller **(~~DSCDM~~)** No optional board in the controller DSO-MAC: Macro field bus interface **(~~DSCDL~~)** DSO-HIO: Extension board with 8 digital I/O DSO-CAN: CAN field bus interface, with CANetel protocol **(~~DSCDL~~)** DSO-SER: Sercos field bus interface **(Only on DSC2P and DSC2V)** DSO-PRO: Profibus field bus interface **(~~DSCDL~~)**	-
M77	-	Optional board info boot revision	-	Number of the boot revision in the optional board **(~~DSCDM~~)**	-
M78	-	Optional board firmware	-	Number of the firmware version installed in the optional board **(~~DSCDM~~)**	-
M79	-	Optional board serial number		DSO-XXX board (installed in the controller) serial number **(~~DSCDM~~)**	-
M85	-	Controller article number	-	Controller's article number (16 strings using 4 depths of M85)	-
M86	-	Optional board article number	-	DSO-XXX article number (16 strings using 4 depths of M86) **(~~DSCDL~~** and **~~DSCDM~~)**	-
M87	-	Controller axis number	-	Controller's axis number	-
M90	-	Controller temperature	-	Controller's temperature (heat sink).	[°C]
M91	-	Controller +Vpower measurement	-	Gives the DC bus Vpower level **(~~DSCDP~~** and **~~DSCDM~~)** Read M91: +Vpower[V] = M91 / 100	100·[V]
M92	-	Controller -Vpower measurement		Gives the negative DC bus Vpower level **(Only on DSCDL)** Read M92: -Vpower[V] = M92 / 100	
M95	-	Controller display message	-	LCD display message (16 strings using 4 depths of monitoring M95)	-
M96	-	Sequence line	-	Number of line in the sequence currently processed by the controller	-

Monitorings M71, M72 and M78:
Monitorings M72 and M78 contain, respectively, the firmware version numbers installed in the controller and in an optional board if there is any. Monitoring M72 can directly be read with **VER** command (refer to §12.1.3). Monitoring M71 gives the software boot version of the controller.

The controller sends a firmware version, as a hexadecimal number, under the following form:
VER.1 = 0x**WWWXXYY**, with the following possible values:

	WWW: version # (3 digits)				**XX: revision index (2 digits)**										**YY: status (2 digits)**	
Values examples	100	...	120	...	00	01	...	40	41	...	80	81	82	...	00	80
Meaning	1.00	...	1.20	...	α_0	α_1	...	β_0	β_1	...	A	B	C	...	released	in dev.

For example, the hexadecimal value 0x**1208000** means that the firmware version 1.20A, released, is in the controller.

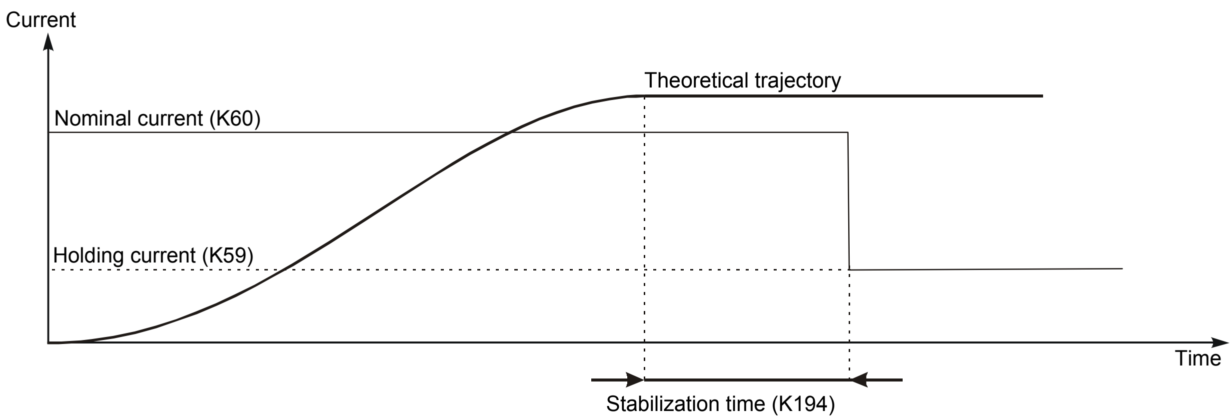
Monitoring M90 shows the controller temperature in Celsius degrees. If the temperature inside the controller is greater than 70°C (75°C for the DSCDM), the **OVER TEMPERAT** error (M64=5) will appear. If the thermostat inside the controller is faulty (or if the power bridge of the DSC2V is overheating), the **SENSOR TEMP ERR** error (M64=13) will appear.

12.13 Stepper in open loop

Available on	~~DSC2P~~	~~DSC2V~~	DSCDP	DSCDL	DSCDM

It is possible to manage two or three-phase stepper motor in open loop which means that an encoder is not needed. It is then not possible for the user to know if some steps have been lost or not and all the registers and command referring to the real position cannot be used any more.

During the movement in standard reference mode (K61=1), a current called 'nominal current' and set by parameter K60 is injected into the phases of the motor to follow the theoretical trajectory. When the movement is finished, another current called 'holding current' and set by parameter K59 is injected into the phases of the motor to keep the position. It means there is always current present in a stepper motor when it is powered on. The holding current (parameter K59) must then be high enough to keep the position of the motor but not too high to avoid higher temperature in the phases of the motor.
At the end of the theoretical trajectory, it is possible to set a stabilization time with parameter K194 before changing from parameter K60 to parameter K59.



In standard reference mode (K61=1, refer to §12.6 for more information), a filter on the current reference set by parameter K9 allows the user to have smooth current changes. The current is injected into the motor as soon as the motor is in 'Power On' (PWR.<axis>=1). The 'moving' bit (bit# 4 of SD1, refer to §13.11.1 for more information) is set to 1 during the movement and the stabilization time. In external reference mode (K61≠1, refer to §13.2 for more information), **only** parameter K60 is taken into account and the user has to change its value during the movement.

In one turn, the number of drive increment [dpi], given by parameter K55 is equal to 2048 x p with p is the number of pairs of poles (this number of pairs of poles is given by parameter K54). Then, 1 dpi = 360°/(p x 2048) = 360°/(K54 x 2048).

- **Phasing mode**

The only phasing mode available with stepper in open loop is the phasing by constant current (K90=2) which means that parameters K92, K93, K94 and K97 must be set correctly (Refer to §12.7.2.2 for more information). During the phasing with stepper in open loop, the motor will move of maximum one magnetic period.

- **Homing mode**

The homing mode is normally used to define an absolute position. As there is no encoder used with stepper in open loop, the only homing mode which can be used are as follows:

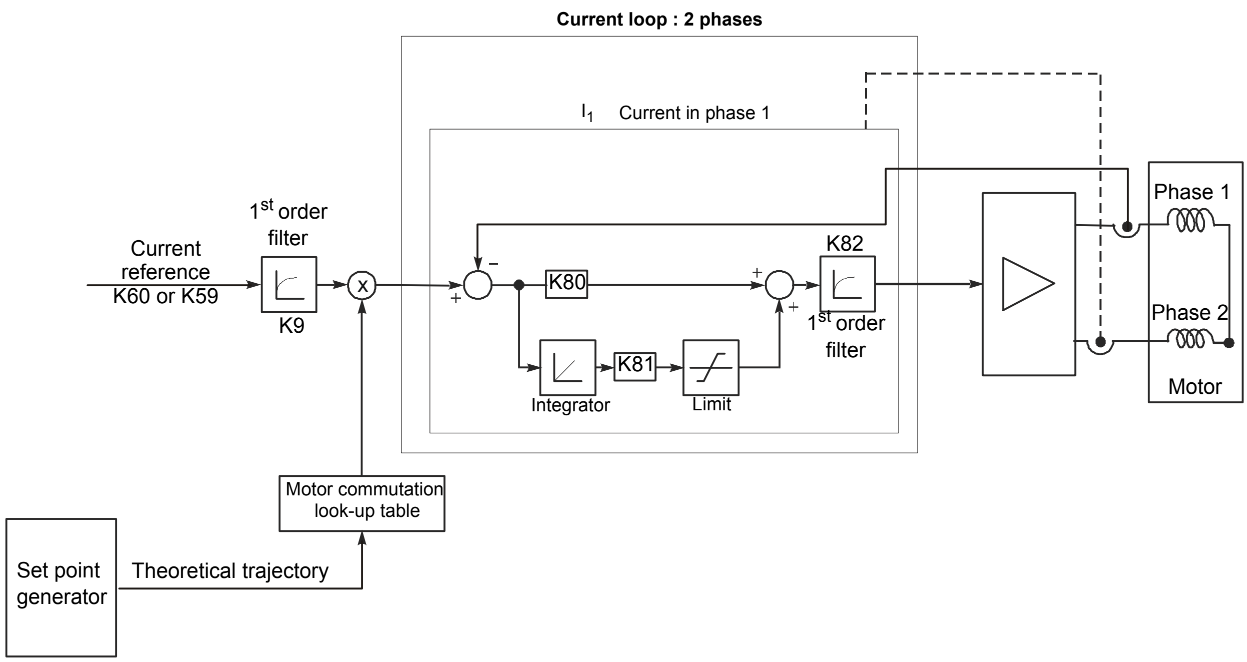
- Homing mode with limit switch: K40=2, 3, 6 and 7 (refer to §12.9.3 for more information)
- Homing mode with home switch: K40=4 and 5 if bit# 0 and 1 of parameter K58 are equal to 0 (refer to §12.9.3 for more information)
- Immediate homing: K40=22 (refer to §12.9.3 for more information)

The homing is made on the FTI interrupt (refer to §4.2 for more information). The home switch and limit switch are tested at each FTI. As the DSCDM does not have a DIN2 (which is normally used for home switch), bit# 2 and 3 of parameter K58 allows the user to select DIN1 or DIN9 to connect the home switch (refer to §12.9 for more information). It is not possible to make a homing on mechanical end stops as there is no way to detect them.

- **Reference mode**

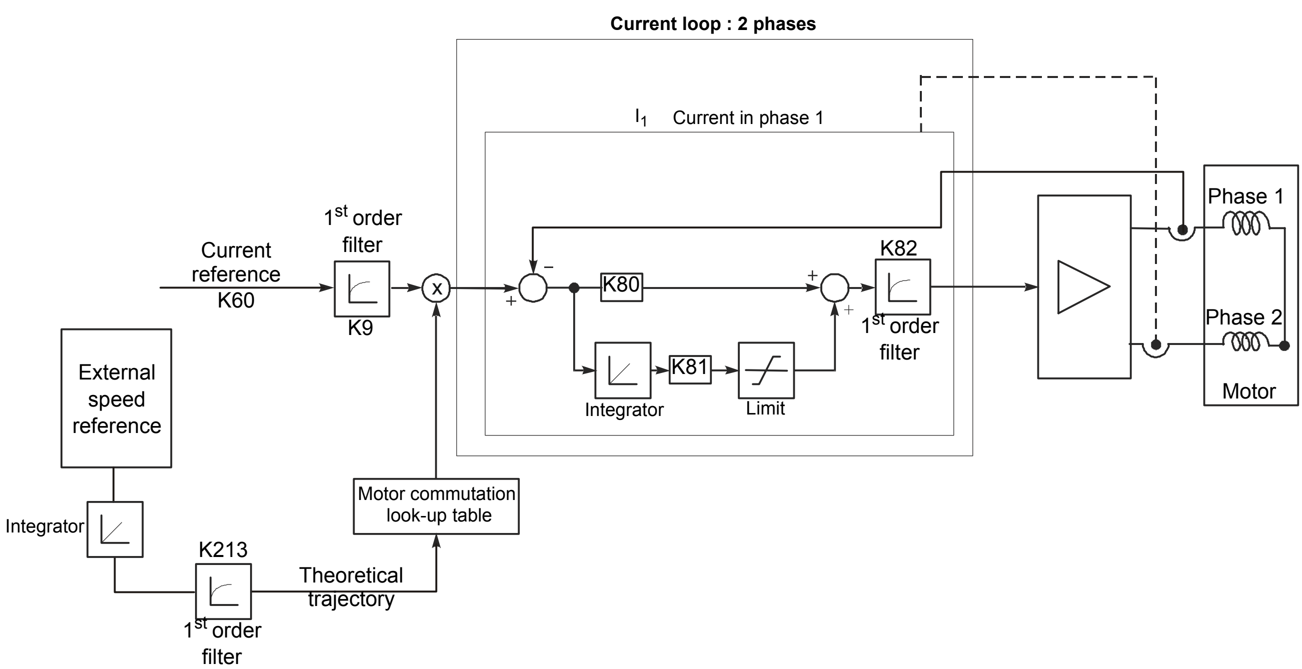
With stepper in open loop, the reference mode (defined by parameter K61) which can be used are:

- Standard reference mode: K61=1 (refer to §12.6 for more information)



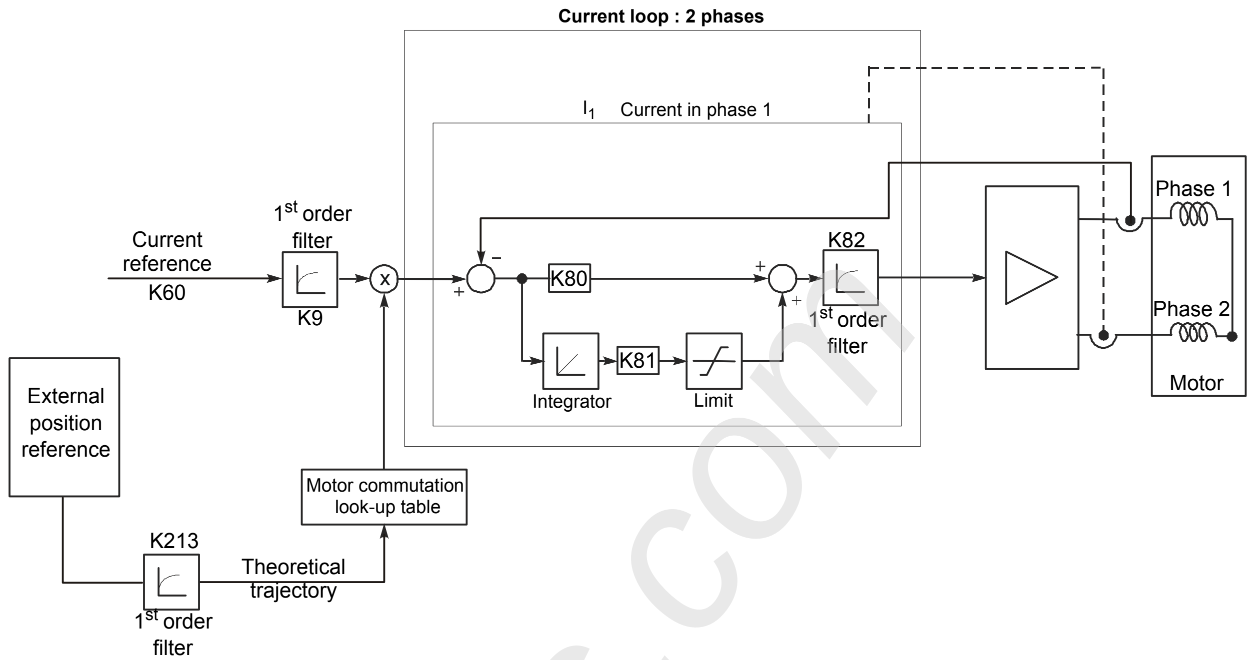
Remark: The regulation loop principle is identical with a three-phase stepper motor.

- Speed reference mode: K61=3 (refer to §13.2.1 for more information)



Remark: The regulation loop principle is identical with a three-phase stepper motor.

- External position reference mode: K61=36 (refer to §13.2.1 for more information)



- **Other parameters**

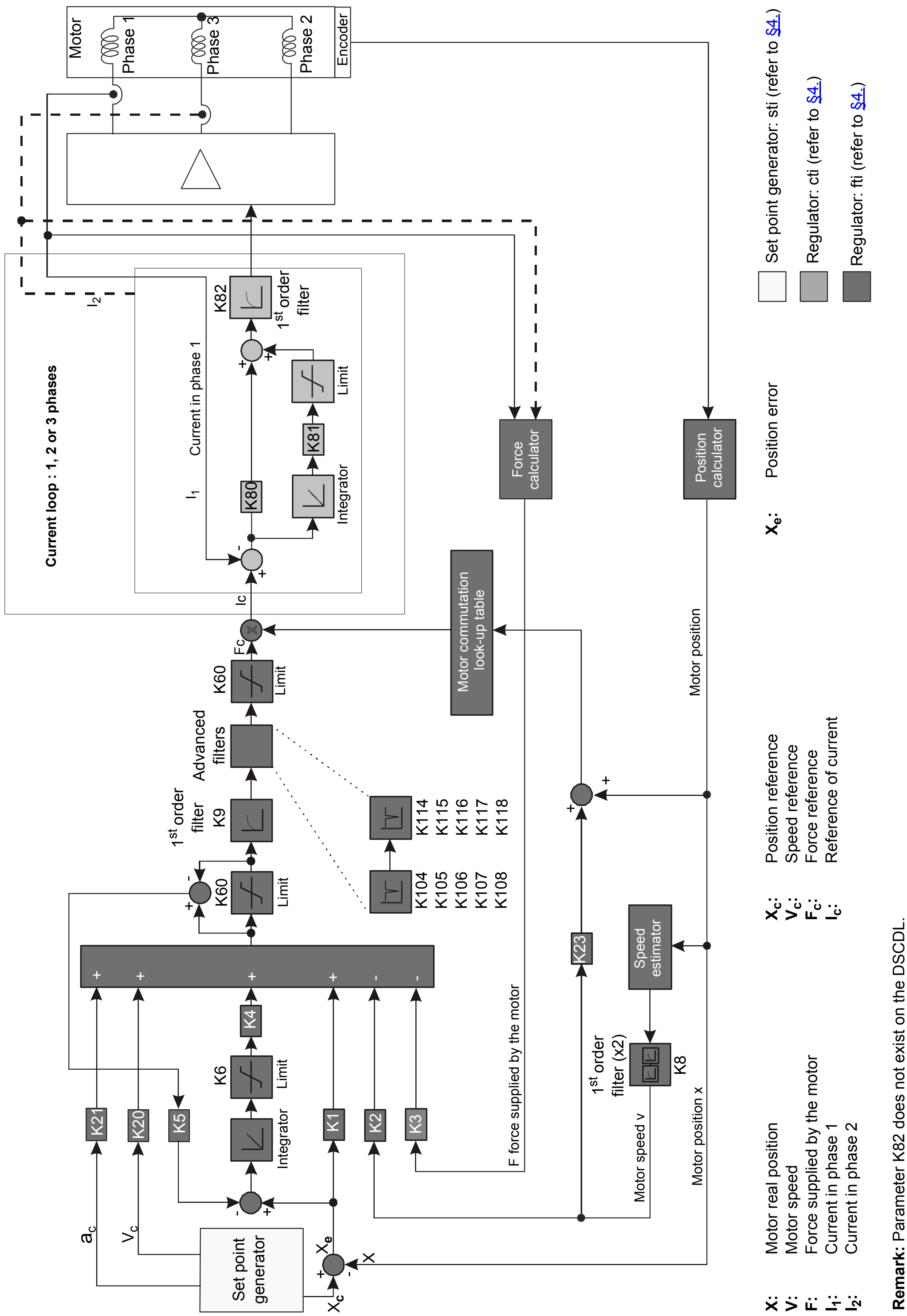
Some parameters (other than those mentioned above), previously described in this manual, must be used with stepper in open loop:

- Commutation look-up table parameters: K52 and K53 must be equal to 0 (refer to §12.7.1.4 for more information)
- Force inversion given by parameter K56 (refer to §12.2 for more information). Changing the force in stepper mode as the same effect than changing the movement direction
- Encoder interpolation shift value given by parameter K77 must be equal to 0 (refer to §13.10 for more information)
- Encoder type selection given by parameter K79 must be equal to 20, 21, 23 or 24 (refer to §12.3 for more information)
- Current loop parameters given by parameters K80, K81 and K82 (refer to §13.1.2.2 and §13.1.2.3 for more information)
- Current limits given by parameters K83, K84 and K85 (refer to §12.4.2 for more information)
- Motor phase number given by parameter K89 must be equal to 20 or 30 (refer to §12.2 for more information)
- The number of period per turn given by parameter K241 which is equal to K54 x 2 (refer to §12.3 for more information)

13. Advanced functions (only for advanced users)

13.1 Regulators in details - advanced tuning

13.1.1 Regulators diagram



13.1.2 Regulators parameters

The position feedback is controlled by a **state regulator**, which can be approximated to a proportional-integral-derivative (PID) regulator. The **position state regulator's** parameters are:

13.1.2.1 Position regulator's gains and limits

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

K	Name	Comment	Units
K1	P gain position	Position loop's proportional gain	-
K2	D gain position	Position loop's speed feedback gain	-
K3	Force feedback gain position	Position loop's force/torque feedback gain	-
K4	I gain position	Position loop's integrator gain	-
K5	anti-windup gain	Position loop's anti-windup gain.	-
K6	Integrator limit	Position loop's integrator limit	-
K20	Speed feedforward	Position loop's speed feedforward gain	-
K21	Acceleration feedforward	Position loop's acceleration feedforward gain	-
K60	Maximum force limit	Theoretical software force/torque limit (regulator output)	[foi], [toi]

Parameters **K1**, **K2** and **K4** define the position regulator's PID. Parameter K1 is proportional to the motor position error x_e; it is used to make the position regulator more reactive. The integral gain (parameter K4) will reduce oscillations and suppress a permanent error on the position. Parameter K2 is proportional to the speed (and not the speed error), thus it acts as a pseudo-derivative gain.

Parameter **K3** (motor force feedback gain) is rarely used. It may be used sometimes when the advanced filters and the filter set by parameters K9 and K10 are not able to suppress low frequency oscillations (range: around 500Hz) on the position loop output (force reference Fc).

Parameter **K5** (anti-windup) works together with parameter K60 (the force (or current) reference limitation).

Parameter **K60** is automatically set to protect the motor against a too high force reference (Fc) coming from the state regulator. For an optimal operation, parameter K60 should never be reached, or should limit Fc only for very short times. If parameter K60 is often reached and for long times, it means that the motor is under-dimensioned in comparison with its load. In that case, parameter K5 should be used to compensate the state regulator saturation. When parameter K60 is active, the low motor’s force needs a too long time to reach a far position. During that time, the integrator reaches a high value. When the position is reached, Fc should invert its direction, but it is impossible because the integrator is full. To avoid the problem, parameter K5 will subtract a value from the integrator’s input (parameter K4), but only when parameter K60 is active (state regulator saturated).

Parameter **K6** is the integrator's max. and min. values.

Parameters **K20** and **K21** are feedforward parameters. During a movement, a permanent error may appear between the position reference and the motor's real position. This drag can be due to mechanical friction. It is possible to compensate it with the feedforward parameters. These parameters will increase 'a-priori' (with anticipation, without reading the real drag) the speed and acceleration command inputs to the state regulator. The **speed feedforward** will compensate the drag's constant part and the **acceleration feedforward** will compensate the undershoot and overshoot remaining after the speed feedforward compensation. To have the optimum value, set K20 = K2 (they have the same unit).

Note: Parameters K20 and K21 can be used with all reference modes (parameter K61)

The integrator's effect (parameter K4) may also be disabled with parameter **K7**.

K	Name	Value	Comment
K7	Integrator mode	0 2	Integrator gain (parameter K4) always on Integrator gain (parameter K4) always off

13.1.2.2 *Current regulator's gains*

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **current reference generator**, with the motor look-up table, calculates the input current into each motor phase from the reference force F_c delivered by the position loop and the real motor position. The current loop itself makes sure that this reference is reached.

K	Name	Comment
K23	Commutation phase advance versus speed	Advance of phase commutation versus speed
K80	Current loop proportional gain	Proportional gain of the current loop.
K81	Current loop integrator gain	Integrator gain of the current loop.

To do that, a classical PI regulator is used. Parameter **K80** is proportional to the force reference F_c. It is used to make the current regulator more reactive. The integral gain given by parameter **K81** will reduce oscillations and suppress a permanent error on the current (and on the force).

Parameter **K23** is the phase advance. The current in the motor should always be in phase with the B-EMF; in normal cases, the motor look-up table guarantees it. Nevertheless, when a motor moves at high speed (>2m/s) the current tends to be late with respect to the B-EMF. This delay can be compensated with a positive phase shift of the pointers in the motor look-up table (as the motor currents are late, one has to point ahead in the look-up table). Parameter K23 corresponds to the phase shift defined for the pointers.

The formula below gives the value of parameter K23 according to the requested phase shift expressed in degrees, and the motor speed:

$$K23 = \frac{2^{30} \times \text{Phase shift [degrees]}}{360 \times M11}$$

Monitoring M11 is the motor real speed. The application nominal speed will be chosen.

Remark: With a direct drive, the quality of the current loop has to be very good, otherwise the noise due to the transistor commutation or the current bad measurement due to a ripple, will not be removed with the position loop. A special algorithm has been developed in the controller to reduce the noise.

13.1.2.3 *Regulators filters*

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Numeric filters have been integrated in the regulators. They are set with parameters **K9** and **K82** in the current regulator and parameter **K8** in the position regulator. These are low-pass and first order filters.

The same type of formula is valid to calculate cut off frequencies f_c at –3dB for parameters K8, K9 and K82:

$$fc = \frac{10}{2\Pi \cdot SamplingTime \cdot Gain} \qquad [Hz]$$

K	Name	Comment
K8	Speed filter	Speed feedback filter of the position loop: $fc = \frac{10 \cdot \sqrt{(\sqrt{2}-1)}}{2\pi \cdot 41.67\times10^{-6} \cdot \sqrt{(10+K8)\cdot K8}}$ For DSC2P and DSC2V $fc = \frac{10 \cdot \sqrt{(\sqrt{2}-1)}}{2\pi \cdot 55.56\times10^{-6} \cdot \sqrt{(10+K8)\cdot K8}}$ For DSCDP, DSCDL and DSCDM
K9	Force reference output filter	Force reference filter (1st order) of the position regulator output: $fc = \frac{10}{2\pi \cdot 41.67\times10^{-6} \cdot \sqrt{(10+K9)\cdot K9}}$ For DSC2P and DSC2V $fc = \frac{10}{2\pi \cdot 55.56\times10^{-6} \cdot \sqrt{(10+K9)\cdot K9}}$ For DSCDP, DSCDL and DSCDM
K82	Current filter	Current output filter (1st order) of the current regulator: $fc = \frac{10}{2\pi \cdot 41.67\times10^{-6} \cdot \sqrt{(10+K82)\cdot K82}}$ For DSC2P and DSC2V $fc = \frac{10}{2\pi \cdot 55.56\times10^{-6} \cdot \sqrt{(10+K82)\cdot K82}}$ For DSCDP and DSCDM

With parameter K8, the speed feedback (or pseudo-derivative gain) to the state regulator (parameter K2) will be filtered. Parameter K8 is a speed filter made up of 2 low-pass 1st order filters.

Parameter K9 is the filter on the position regulator's output (force reference), a low-pass 1st order filter, with a -20dB/decade slope.

Parameter K82 is the filter on the current regulator's output (current in the motor).

Remark: Parameter K82 does not exist on the DSCDL.

There are also two advanced filters (second order filter) on the position regulator's output (force reference). The default values below make the advanced filters inactive. To make it active (for specific applications), please refer to the ETEL technical support.

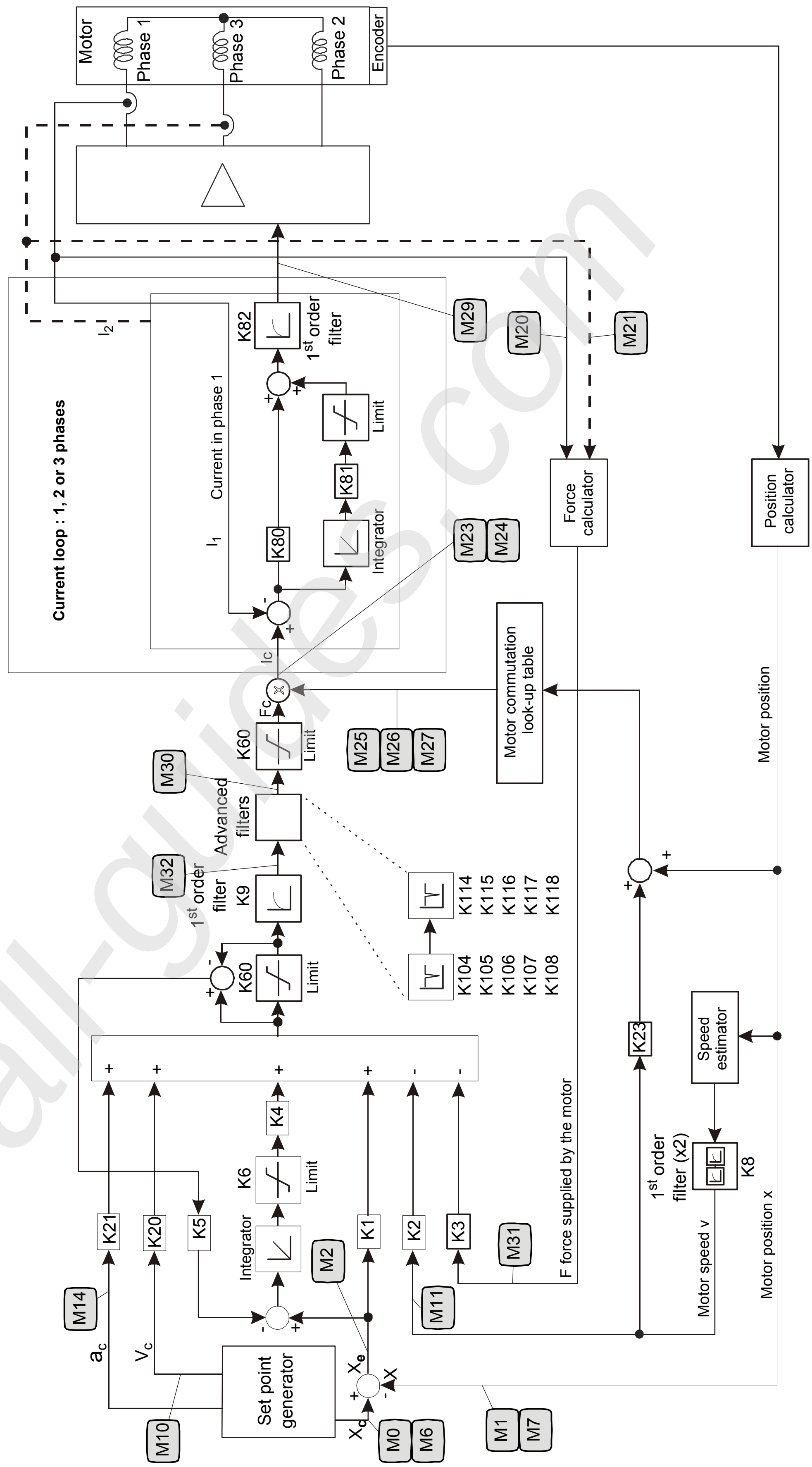
The parameters of the first advanced filter are:

K	Name	Comment
K104	Force reference output filter	Parameter of the first digital filter
K105	Force reference output filter	Parameter of the first digital filter
K106	Force reference output filter	Parameter of the first digital filter
K107	Force reference output filter	Parameter of the first digital filter
K108	Force reference output filter	Parameter of the first digital filter

The parameters of the second advanced filter are:

K	Name	Comment
K114	Force reference output filter	Parameter of the second digital filter
K115	Force reference output filter	Parameter of the second digital filter
K116	Force reference output filter	Parameter of the second digital filter
K117	Force reference output filter	Parameter of the second digital filter
K118	Force reference output filter	Parameter of the second digital filter

13.1.3 Monitorings diagram



Remark: Parameter K82 does not exist on the DSCDL.

13.1.3.1 Monitorings description

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

M	Name	Comment	Units
M0	Theoretical position	Theoretical position Xc. Takes the scaling/mapping correction into account, but does not take care about SET command and parameter K50.	[dpi], [rdpi]
M1	Real position	Real position X. Takes the scaling/mapping correction into account, but does not take care about SET command and parameter K50	[dpi], [rdpi]
M2	Position control error	Tracking position control error Xe. This is the difference between monitorings M0 and M1	[dpi], [rdpi]
M6	Theoretical position in user scale	Theoretical position Xc in user scale. Takes SET command, parameter K50 and the scaling/mapping correction into account	[upi], [rupi]
M7	Real position in user scale	Real position X in user scale. Takes SET command, parameter K50, and the scaling/ mapping correction into account	[upi], [rupi]
M10	Theoretical velocity	Theoretical velocity Vc	[dsi], [rdsi]
M11	Real velocity	Real velocity V	[dsi], [rdsi]
M14	Theoretical acceleration	Theoretical acceleration Ac	[dai], [rdai]
M20	Real current in phase 1	Real current in phase 1	[ci], [A]
M21	Real current in phase 2	Real current in phase 2	[ci], [A]
M22	Real current in phase 3	Real current in phase 3 **(~~DSCDL~~)**	[ci], [A]
M23	Current reference in phase 1	Current reference in phase 1	[ci], [A]
M24	Current reference in phase 2	Current reference in phase 2	[ci], [A]
M25	Current loop look-up table value phase 1	Current loop look-up table value phase 1	Incr.
M26	Current loop look-up table value phase 2	Current loop look-up table value phase 2	Incr.
M27	Current loop look-up table value phase 3	Current loop look-up table value phase 3 **(~~DSCDL~~)**	Incr.
M29	PWM value of phase 1	PWM value of phase 1	Incr.
M30	Theoretical force (after advanced filter)	Theoretical force Fc (after advanced filter 2)	[foi], [toi]
M31	Real force	Real force F	[foi], [toi]
M32	Theoretical force (before advanced filter)	Theoretical force Fc after K9-filter and before advanced filter 1	[foi], [toi]

13.2 Advanced reference modes (K61≠1)

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Parameter **K61** selects the input reference given to the controller. When K61≠1, an Advanced reference mode is selected.

K	Name	Value	Comment
K61	Reference mode	0 1 3 4 36	Force/torque reference mode defined by parameters K220 to K224. Standard mode with set point generator movement profiles. Controller controlled by a speed reference defined by parameters K220 to K224 Controller controlled by a position reference defined by parameters K220 to K224. Like K61 = 4, but the actual position is kept as a reference when the controller is switched on.

13.2.1 External reference modes (K61=0, 3, 4 or 36)

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **external reference** mode includes all the cases where the reference is not given by the set point generator but by an external source. It can be any controller's register (K, M or X).
In all cases the external source can be used as position (K61=4) or force (K61=0) reference. Parameters used with the external reference mode are given in the following table:

K	Name	Value	Comment
K220	Reference type	1 2 3	The external reference is a user variable X The external reference is a parameter K The external reference is a monitoring M.
K221	Reference index	-	External reference index.
K222	Reference multiplication factor	-	External reference multiplication factor.
K223	Reference offset	-	External reference offset correction.
K224	Reference amplitude	-	External reference amplitude correction.

Parameters **K220** and **K221** determine which parameter will be used as an external reference. For the DSC2P, DSC2V and DSCDL, the analog input is usually used and is accessible via monitoring M51 (refer to §13.6) **(monitoring M51 does not exist on the DSCDP and DSCDM)** because parameters K220 and K221 default values are 3 and 51 respectively, corresponding to AIN controller's analog input. For the dual axes controller, the user's variable X0 is used by default.

Caution: The external position reference must be given in user position increments [upi], the external speed reference in user speed increments [usi], the external force reference in force increments [foi] and the external torque reference in torque increments [toi]. The SET command is disabled. When one of the external reference modes is used, the user must set first parameters K220 to K224 **before** parameter K61.

13.2.1.1 *External reference setting:*

Parameters **K222** to **K224 'set up'** the external reference.
The external reference formula taken from the 'unset' external reference (value of the register given by parameters K220 and K221) is:

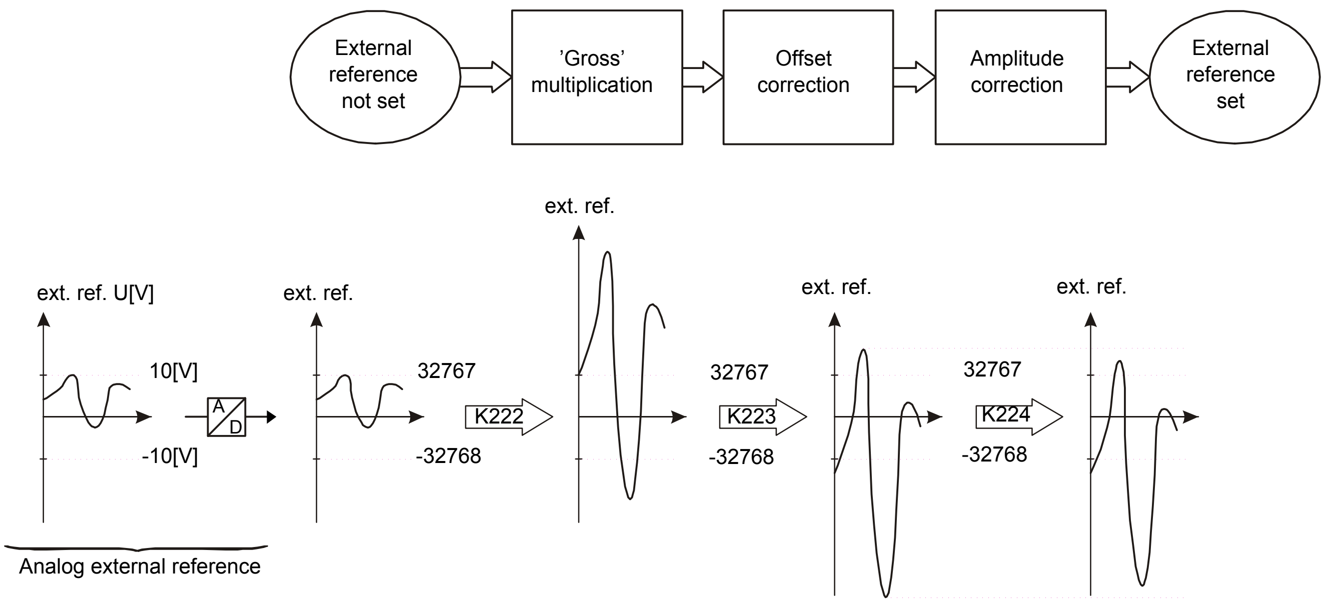
$$\text{external ref. set} = (\text{'unset' external ref. not set} \cdot 2^{K222} - K223) \cdot \frac{K224}{2^{24}}$$

Parameter K222 makes the first **'gross' multiplication** from the unset external reference. This multiplication is useful when a voltage on an analog input is used as reference because the analog input possesses an A/D converter which can convert a voltage between +10V and –10V into a value between -32767 to +32767. For instance 32767 is a very low value for a position reference that can easily reach millions of increments. This value must be multiplied by a very high number if a movement of more than a few millimeters is wished.

After that first multiplication, the value contained in parameter K223 is subtracted to the external reference. This **offset** moves the origin according to the user's needs.

It is then multiplied by $\frac{K224}{2^{24}}$. It tunes the reference amplitude. Unlike the first operation which only increases the amplitude signal, it can decrease it (K224<2^{24}) and invert the reference sign (K224<0).

The following diagram shows the different steps. They are the same for all external reference modes. In the following example, a signal sent to the analog input has been chosen for the external reference.

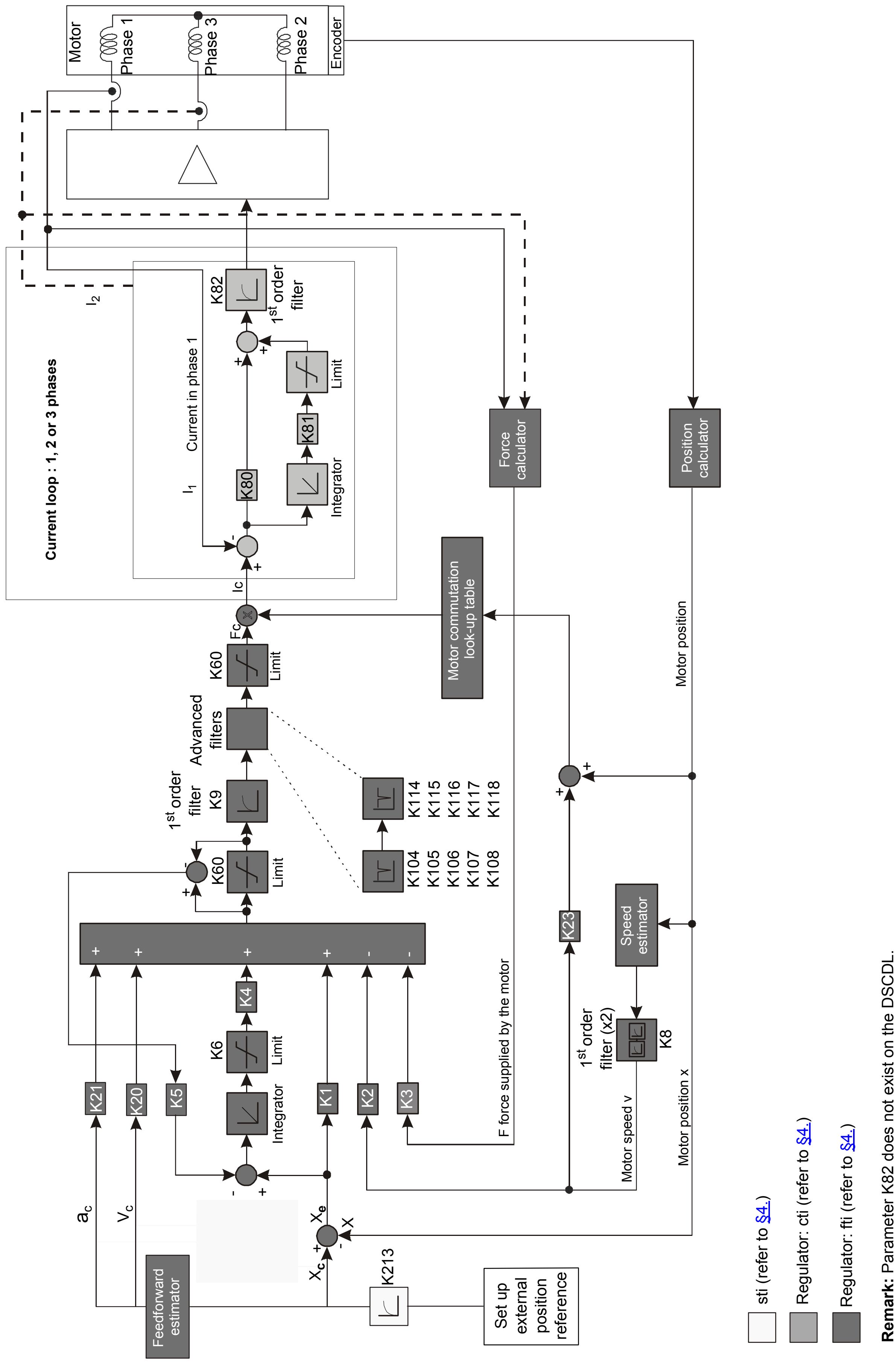


Remark: Refer to §20. for more information about the units of the external reference

Each external reference will now be reviewed in the next paragraphs.

13.2.1.2 *Position reference mode (K61=36):*

K61 = 36: Same as K61 = 4, but the actual position is kept as reference when the controller is switched on. The graph below shows the regulation loop of the position reference mode:



Remark: The external position reference is given in [upi] and [rupi]. Refer to §20. for more information.

13.2.1.3 *DSMAX position reference mode*

Until the firmware version **1.14A** for the DSMAX1 and 2, when the interpolation mode is activated, the DSMAX automatically sends (by default) the ITP=**1** command to put the controller into position reference mode. **From** the firmware version **1.20A** for the DSMAX1 and 2 and firmware **1.10A** for the DSMAX3, the DSMAX automatically sends (by default) the ITP=**2** command to put the controller into position reference mode. Refer to the **'DSMAX User's Manual'** for more information.

The Turbo-ETEL-Bus (TEB) cycle is 166,67µs (6kHz). The DSMAX motion controller works at the same frequency as the TEB and sends a point of interpolation at each cycle (sti).

- **ITP=1 mode**

In this mode, the controllers use the interpolation points sent by the DSMAX at each sti interrupt that is to say every 166,67µs for the DSC2P/DSC2V and every 500 µs for the DSCDP, DSCDL and DSCDM. It means that a dual axes controllers (DSCDP, DSCDL and DSCDM) does not take the interpolation point at each cycle of the DSMAX unlike the DSC2P/DSC2V. The trajectory with a dual axes controller is then not as precise as with a DSC2P/DSC2V. The advantage of the ITP=1 mode is the possibility to use the encoder scaling/mapping (refer to §13.13 for more information) and the filter set by parameter *K600 (refer to the **'DSMAX User's Manual'** for more information).

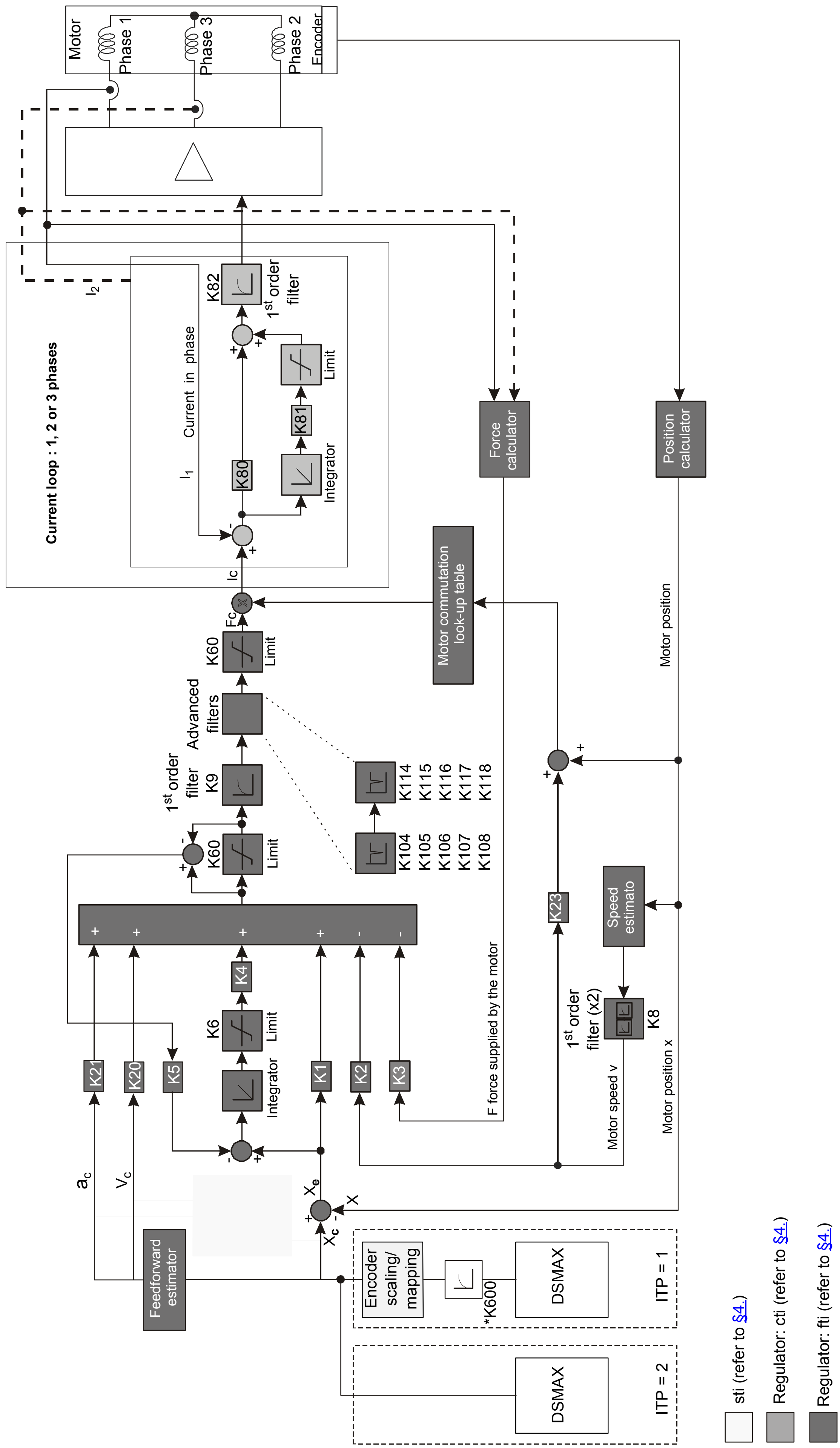
- **ITP=2 mode**

This mode allows all the controllers (even the dual axes controllers) to use all the interpolation points sent by the DSMAX. These points are read every three fti cycles (refer to §4.) for the DSCDP, DSCDL and DSCDM and every four fti cycles for the DSC2P and DSC2V. However, it is not possible any more to use the encoder scaling/mapping and the filter set by parameter *K600

Monitoring **M110** allows the user to know the interpolation mode selected in the position controller:

M	Name	Value	Comment
M110	Interpolation mode type	0 1 2	Interpolation mode disabled Interpolation mode at every sti interrupt. Takes jerk time (IJT) and encoder scaling / mapping into account Interpolation mode on fti interrupt. Does not take jerk time (IJT) and encoder scaling / mapping into account

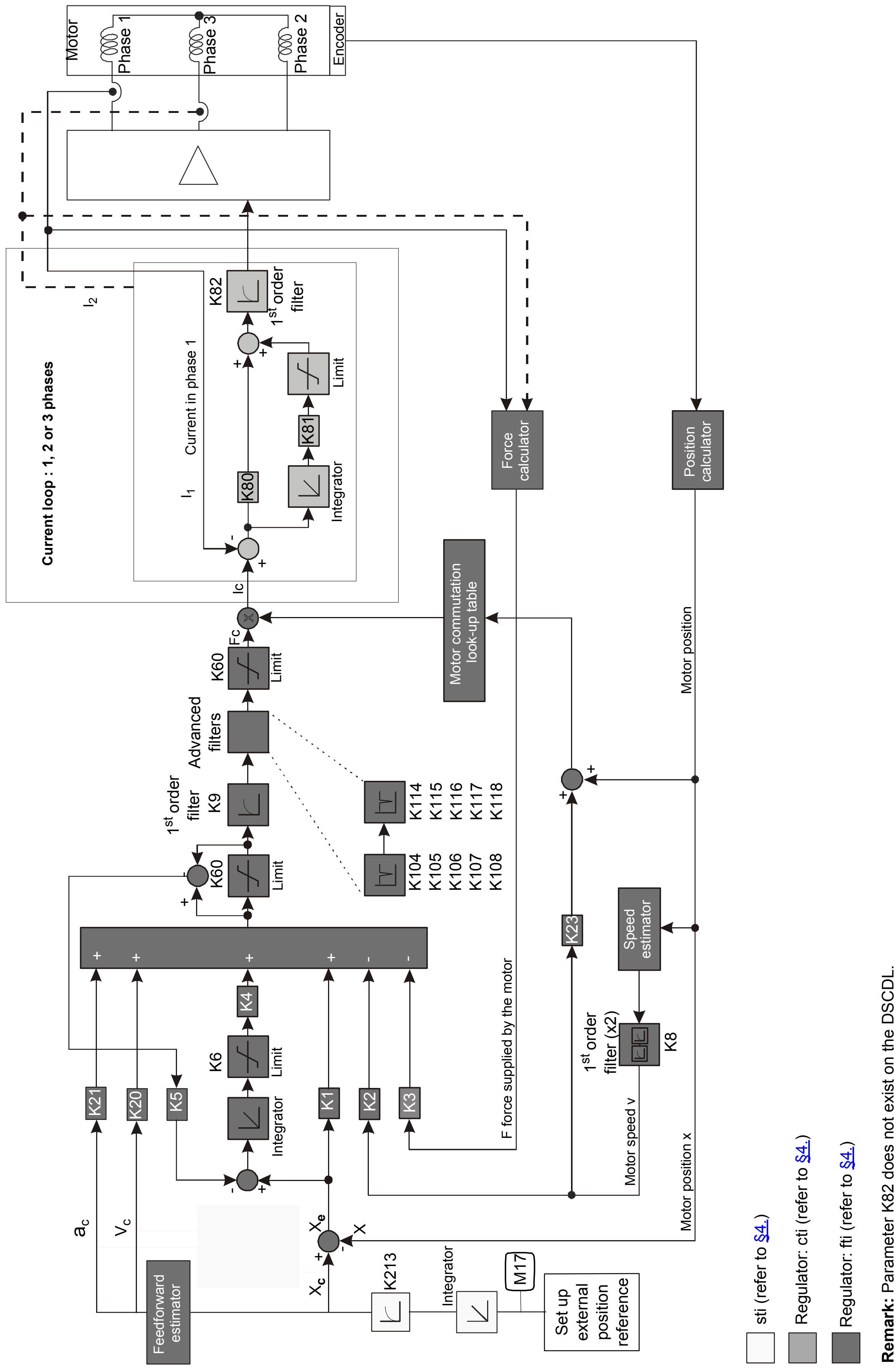
The graphs below show the regulation loop of the DSMAX position reference mode:



Remark: Parameter K82 does not exist on the DSCDL.

13.2.1.4 Speed reference mode (K61=3):

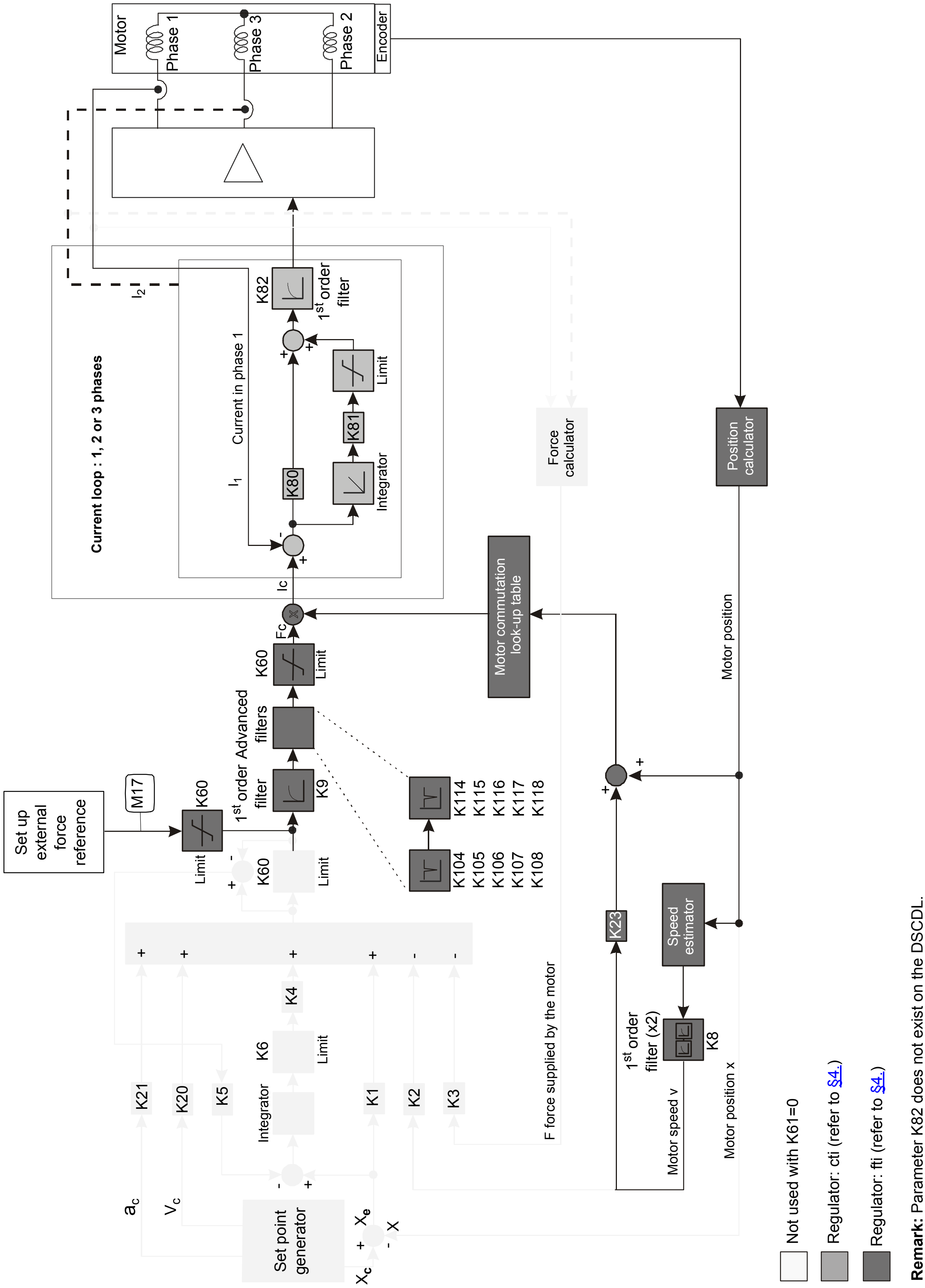
The external reference is used as a speed reference. An integrator calculates the position reference from this reference. The graph below shows the regulation loop of the speed reference mode:



Remark: The external speed reference is given in [usi] and [rusi]. Refer to §20. for more information.

13.2.1.5 *Force reference mode (K61=0):*

The external reference directly gives the motor reference force F_c without going through the position loop regulator. The limit given by parameter K60 is used and limits the force reference value after being set. The graph below shows the regulation loop of the force reference mode.



Remark: The external force reference is given in [foi] and [toi]. Refer to §20. for more information.

13.3 Advanced movements

Here are the advanced movements usable in standard reference mode (K61 = 1).

13.3.1 Movements types

Advanced movement types are:

- Look-up table (refer to §4.) movement
- Calculated movement with predefined profile
- Rotary look-up table movement
- Rotary infinite movement
- Rotary calculated movement with predefined profile

Note: S-Curve and rotary S-curve movements have been explained in §12.10, as basic movements.

13.3.1.1 Movements definition

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

All parameters allowing the definition of advanced movements are summarized below. Detailed explanations are given from paragraph §13.3.2.
Some of the following parameters are available via alias commands.

K	Alias	Name	Value	Comment	Units
K202	MMD	Movement type	1 3 10 17 19 24 26	S-curve (jerk time) movement Calculated movement with predefined profile. Parameter K230 gives the profile and parameter K229 the time. Look-up table movement (sti) Rotary S-curve (sti) movement Rotary calculated mvt. with predefined profile. Parameter K230 gives the profile and parameter K229 the time. infinite rotary movement Rotary look-up table movement	-
K201	MMC	Concatenated movements selection	0 1 2 3	Concatenated movements disabled Concatenated movements for MMD=1 & 17 Infinite back and forth movement for MMD=3, 10, 19 & 26 One back and forth movement. for MMD=3, 10, 19 & 26	-
K203	LTN	Number of the LKT	0-7	Number of the user look-up table movement selected.	-
K204	LTI	Execution time of LKT movement	-	Time to execute a look-up table movement	[sti]
K205	CAM	Cam value	-	Cam value (in percent). Stretches the time scale	-
K206	-	Quick stop acc. pre-programmed	-	Brake acceleration used with BRK and HLB commands.	[uai] [ruai]
K207	-	LKT starting and end positions	0 1	Starting and end position are different (target defined by POS command) Starting and end position are the same (amplitude defined by parameter K208)	-
K208	-	LKT movement max. stroke	-	Max. stroke for LKT movement with K207=1 and MMD=10 and 26	[upi]
K229	-	Calculated movement time		Execution time of the movement selected by parameter K230. Time = K229 x 166.67 µs or K229 x 500 µs	[sti]
K230	-	Calculated movement profile	 0 1 2 3	Predefined profiles for MMD=3 or 19 Triangular (speed) movement S-Curve (full jerk) movement Sine modified (Cam) movement Real sine movement	-

Remark: LTI and CAM alias use the same syntax than their corresponding parameter K. Refer to §6.3 for more information about the syntax and the possible operators.

13.3.1.2 Movements concatenation: MMC

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

MMC = 1: Concatenated movement activated for MMD = 1 & 17 (S-curve and rotary S-curve movement respectively).

MMC=2 or 3: The time needed by the motor to reach the end of the LKT (point# 1999) is: LTI x STI. (STI = 166.67µs for DSC2P/DSC2V and 500µs for DSCDP, DSCDL and DSCDM).
Then, the motor goes back to its start point. For the way out like the way back, the LKT points are read from 0 to 1999.

MMC = 2: The back and forth movement is infinite; BRK, STP, HLT, HLB, HLO commands may stop it (the POS command will stop the movement at the end of the current back and forth movement).

MMC = 3: One back and forth movement and the motor stops. It may also be stop with BRK, STP, HLT, HLB, HLO commands.

Remark: Refer to §12.3.6 for more information.

13.3.1.3 Movement definition: MMD

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Command **MMD** (**M**ove**M**ent mo**D**e) is an alias which chooses the requested type of movement. If MMD command is not executed before starting a movement, the S-curve movement will be selected by default.

Alias	**Parameter**		
MMD.<axis> = <p1>	**K202**		
<p1>	**Type of movement**	**Usable functions**	**See also**
1	S-curve (jerk time) movement (linear motor)	POS, SPD, ACC, JRT, K206, MMC	§12.10.3
3	Calculated mvt. with predefined profile (linear motor)	POS, K206, K229, K230, MMC, K213	§13.3.4
10	Look-up table movement (linear motor)	POS, K203, K204, K206, K207, K208, MMC	§13.3.2
17	Rotary S-curve (jerk time) movement	POS, SPD, ACC, JRT, K27, K206, K209, MMC	§12.10.4
19	Rotary calculated movement with predefined profile	POS, K27, K206, K209, K229, K230, MMC	§13.3.4
24	Infinite rotary movement	POS, SPD, ACC, JRT, K27, K206, K209	§13.3.3
26	Rotary look-up table movement	POS, K27, K209, K203, K204, K206, K207, K208, MMC	§13.3.2.2

Example:

`MMD.1=1` ;Select S-curve movement.

`MMD.1=24` ;Select Rotary infinite movement.

13.3.2 Look-up table movements

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **L**oo**K**-up **T**able (**LKT**) movements functioning principle has already been explained in §3.1.1. Here will be exposed how to set them up.

MMD.x	Type of movement	Usable functions
10	Look-up table movement (sti) (linear motor)	POS, K203, K204, K206, K207, K208, MMC

13.3.2.1 Start and end at the same point, max. stroke

It is possible with the look-up tables to begin and finish a movement at the same point (when K207=1).Refer to the drawing below.

In that case, the 'position to reach' parameter (K210, alias of POS command) is not used any more, as the end is the same as the start position and it **only** starts the LKT movement.

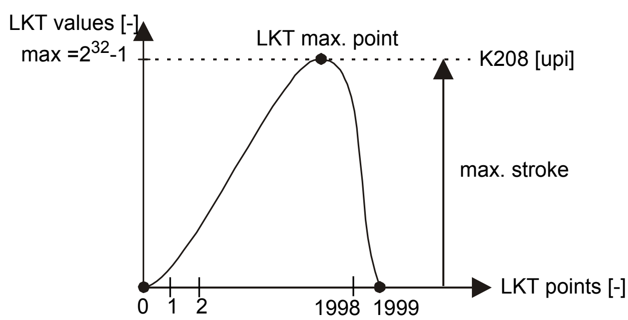
Instead, the **movement's maximum stroke** is defined by parameter **K208** in [upi]. This parameter defines the furthest point (with respect to the start/end point) reached during the LKT movement.

13.3.2.2 Look-up table points definition

There are 2000 points in the LKT (0 to 1999 stored in the register L) which must be not signed. The LKT points max. possible value is: 2^{32}-1. Ideally, the point corresponding to the movement's maximum stroke should bear this value, to have the highest possible resolution with the LKT movement. Position during the movement is given by the formula:

$$\text{Position[upi]} = \frac{\text{LKTvalues}}{2^{32}-1} \cdot \text{K208[upi]}$$

So, the LKT max. value for the max. stroke will use the whole LKT resolution:



13.3.2.3 LTN and LTI commands

To realize a look-up table movement, the movement must first be selected with the MMD command. Then the requested table is selected with the **LTN** command (**L**ook-up **T**able **N**umber) and the total movement duration $t_{movement}$ with the **LTI** command (**L**ook-up table **TI**me). Finally the movement is executed with the POS command which also determines the final position x_{final} to reach.

K	Alias	<p1>	Comment	Units
K203	**LTN.<axis> = <p1>**	0-7	Number of user look-up table movement selected.	-
K204	**LTI**	-	Time to execute a look-up table movement. Time = K204 x 166.67µs or K204 x 500µs	[sti]

Remark: LTI alias use the same syntax than the corresponding parameter K. Refer to §6.3 for more information about the syntax and the possible operators.

The LTI value is limited to 1500000 for the DSC2P/DSC2V and to 500000 for DSCDP, DSCDL and DSCDM for technical reasons. This means that it is not possible to realize look-up table movements of more than 166,67µs·1500000 = 250s for the DSC2P/DSCD2V or more than 500µs·500000 = 250s for the DSCDP, DSCDL and DSCDM.

It is also possible to use the **CAM** command (refer to §13.3.7) to increase the duration of look-up table movements, but it is not possible to increase the time over 250s whatever the controller.

Caution: The reader must be aware that the shorter time $t_{movement}$ is, the bigger speed and acceleration become.

Once the LKT are programmed into the controller, you can use them as explained below:

`LTN.x = y`	;y: # of the look-up table to use.
`LTI.x = y`	;y: execution time of the LKT in [sti].
`K207.x = y`	;y: 0 to a normal LKT and 1 to a LKT with the begin and the end at the same place.
`K208.x = y`	;y: amplitude of the movement in user increments (when K207=1)
`POS.x = y`	;y: not taken into account, only starts the LKT movement
`STA/STI.x = y, k`	;y: value in a depth to start a movement ;k: mask of bit(s)

Example:

The following example shows how to realize a look-up table movement with a total movement of 200000 increments in 2.5s.

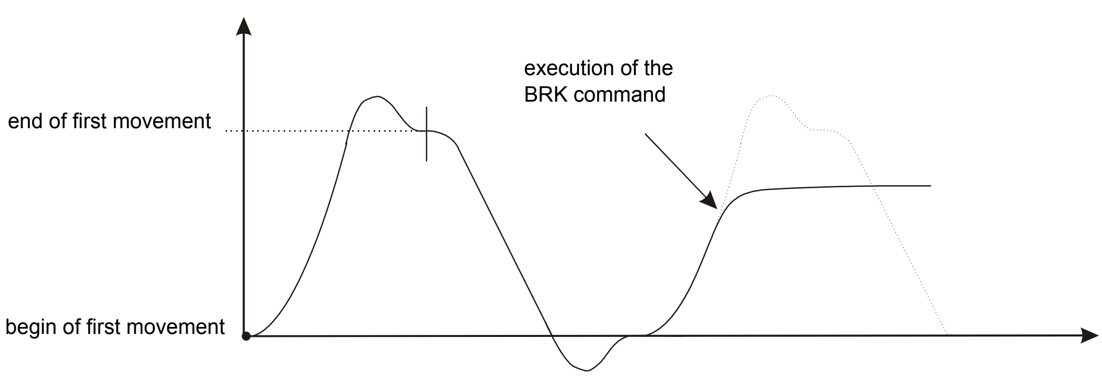
`MMD.1=10`	;Select the look-up table movement.
`LTN.1=3`	;Select the look-up table movement (LKT 3 is user-defined).
`LTI.1=15000`	;The movement duration is: 15000 x 166,67µs = 2.5s for the DSC2P/DSC2V and 15000 x 500µs = 7.5s for the DSCDP, DSCDL and DSCDM.
`POS.1=200000`	;The motor moves by following the trajectory (defined in LKT 3) to the position 200000 and reaches this position 2.5s after leaving.

13.3.2.4 *Rotary look-up table movement*

MMD.x	Type of movement	Uses functions
26	Rotary look-up table movement	POS, K27, K209, K203, K204, K206, K207, K208, MMC

This movement type works like a 'Rotary S-curve' type but it executes look-up table type movements.

Look-up table movements may be concatenated with MMC=2 or 3. Example:



Remark: The example above is also valid for MMD=10 (linear look-up table movement).
To stop a concatenated look-up table movement, use a STP or BRK command.
The POS command will also stop the movement at the nearest start or end point of the LKT.

13.3.3 Infinite rotary movements

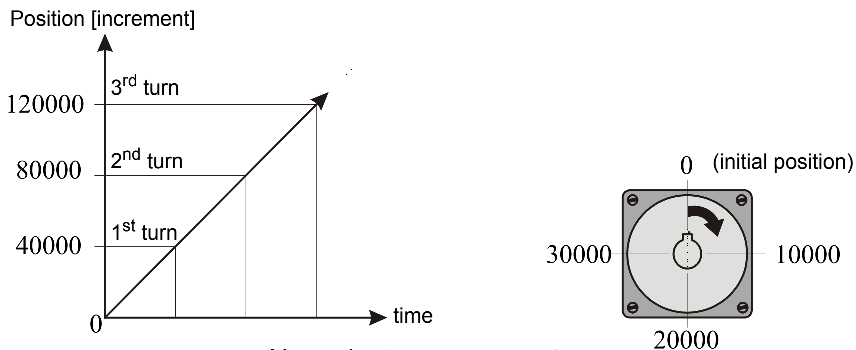
Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

A rotary motor can make an infinite rotation in one or the other direction if there is no mechanical end stop to limit its stroke. This type of movement is selected with the MMD command. To avoid the motor real position value to become too important and exceed the controller's position counter limit, it must be reset at regular intervals via the parameter K27.

The motor turns indefinitely in the way defined by parameter K209. Its position is always included between 0 and parameter K27.

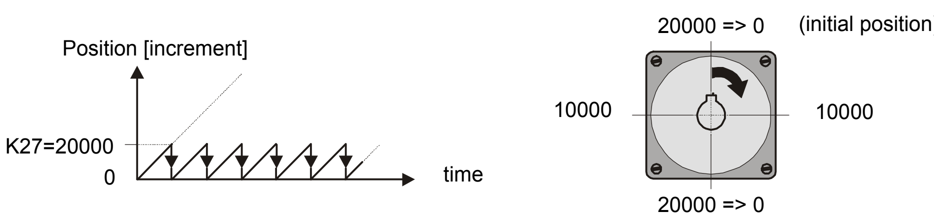
13.3.3.1 Functioning principle

Supposing that a rotary motor revolves at constant speed, the measure of the position versus time is:



Normal rotary movement

The position measure increases continuously and after a lapse of time it will be so big that the controller will not be able to handle it. To avoid it, as soon as the motor reaches the position programmed with parameter K27, the position is brought back to 0. Thus, the motor position is never bigger than parameter K27. If the value 20000 is inserted in parameter K27 (1/2 a turn), the motor position measure versus time would likely be:



Infinite rotary movement

The movement undertaken before the distance measured has been reset is called a cycle. In the case above, a cycle corresponds to half a turn. The size of a cycle definition depends on the application.

Remark: In infinite rotary mode (MMD = 24), the acceleration can be modified only when the speed is constant (and whatever the value of MMC).

Infinite rotary movement example:

`K209.1=0`	Selects a positive movement
`MMD.1=24`	Selects an infinite rotary movement
`K27.1=20000`	Defines parameter K27
`ACC.1=300000`	Defines the movement acceleration
`SPD.1=200000`	Defines the movement speed
`POS.1=1`	**Starts an infinite movement** whatever the value of the POS.1

Remark: K209 = 0 would give a positive movement, and K209 = 1 would give a negative movement (refer to §12.10.4.1)

The motor starts revolving endlessly in the positive direction with an acceleration of 300000 and a speed of 200000. It is a **trapezoidal** movement.

There are two ways to stop an infinite rotary movement. The BRK command can be used so that the motor brakes with the deceleration contained in K206 and stops. The final position is not determined by the user.

`BRK.1` The motor brakes with the deceleration contained in K206 and stops.

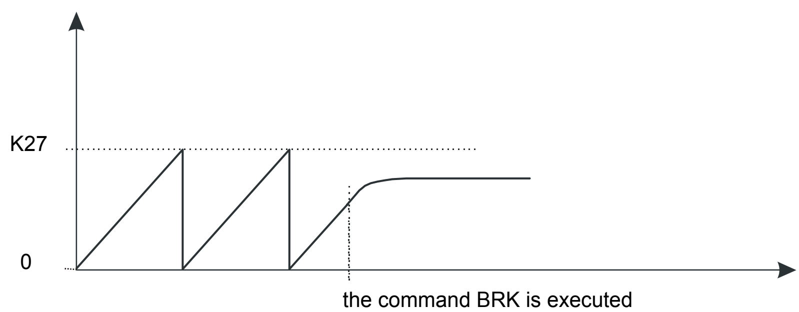
The POS command can also be used to stop the motor in a determined position with the acceleration defined with ACC command in parameter K212.

`POS.1=15000` The rotary movement stops with the acceleration defined with ACC in position 15000. If this position has already been passed in the present cycle, or if the programmed acceleration is not big enough to reach it, the motor will stop there during the next cycle.

13.3.3.2 Stopping an infinite rotary movement

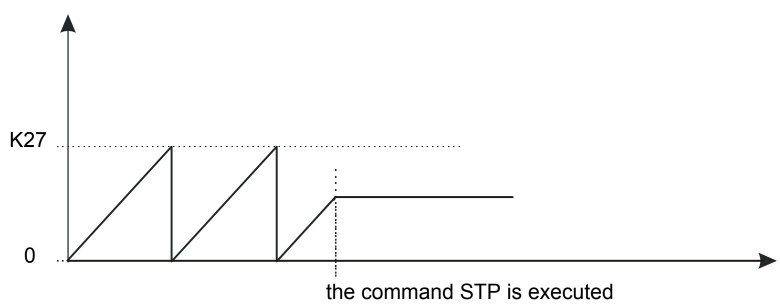
a) Using BRK command

Note: Using the BRK command is generally recommended, as it is smoother for the machine (takes parameter K206 into account).



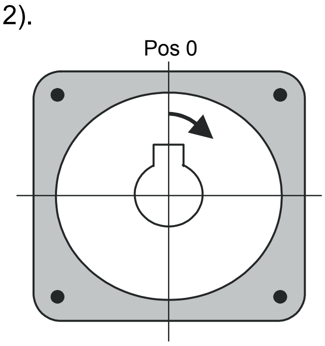
b) Using STP command

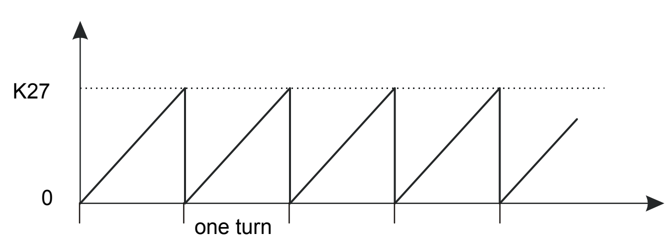
Note: Using the STP command is generally not recommended, as it is very sharp for the machine.



c) Using POS command

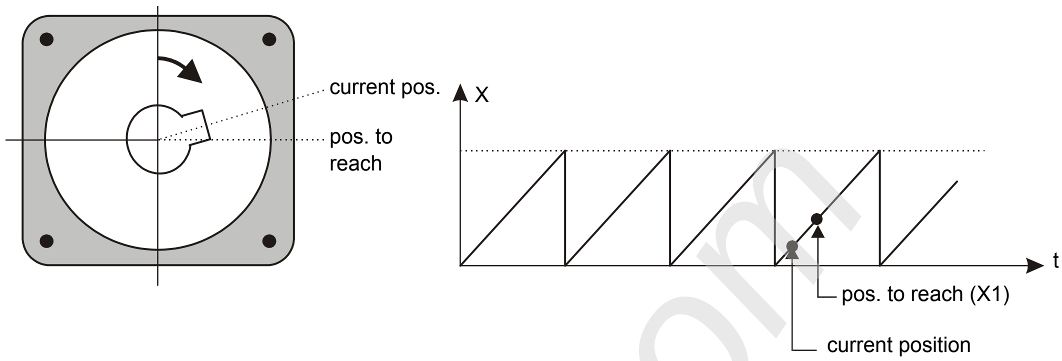
The motor is revolving with a constant speed (parameter K211) and its acceleration is limited (parameter K212).



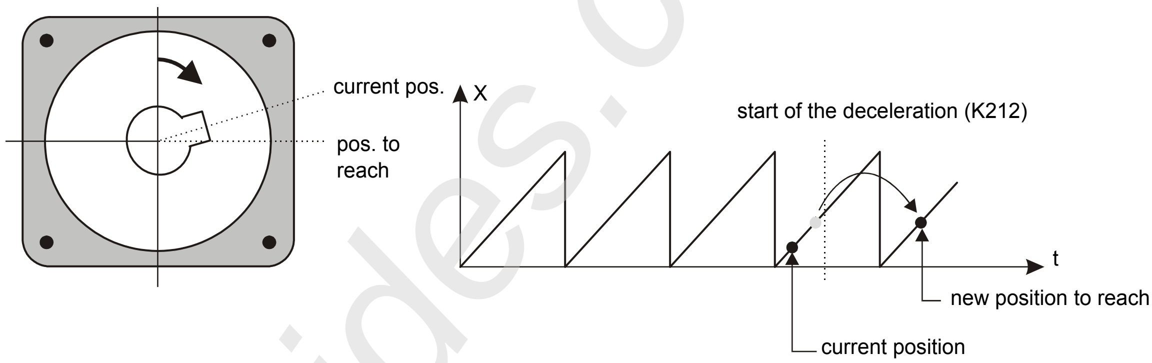


The user can stop the motor in the position X1 by sending the command POS.x = X. If the distance between the current motor position and the position to reach (X1) is too small to stop the motor with the acceleration in parameter K212 (negative = deceleration), the motor will make one more turn to stop correctly.

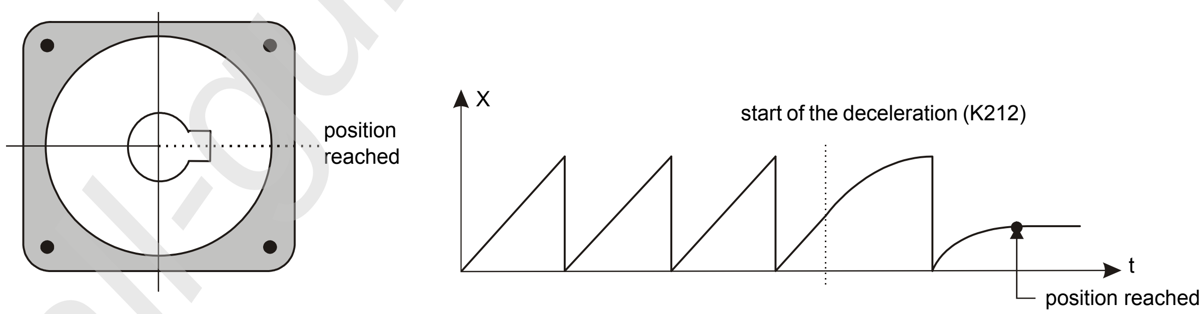
Step 1: POS.x = X



Step 2



Step 3



13.3.4 Movement with predefined profile

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

This chapter describes the two new mathematical movement types available with the controller.

The working principles are as follows:

- Like a look-up table, the duration of the whole movement is written in parameter K229. The maximum value is 100000 STI which represents 16.7 sec for the DSC2P/DSC2V or 50s for the DSCDP., DSCDL and DSCDM
- There is no linear interpolation but at each STI (refer to §4.), a point is calculated (position versus time).
- Parameter **K230** gives the type of movement to use (triangular, S-curve (full jerk), sine modified, real sine.

K230	Movement
0	Triangular
1	S-curve (full jerk)
2	Sine modified (*)
3	Real sine

(*): it is almost a sine but the beginning and the end of the movement is done without acceleration.

13.3.4.1 Movement calculated with predefined profile (MMD=3)

The type of trajectory depends on the value of parameter K230. The target is defined by parameter K210 and the duration by parameter K229.

MMD.x	Type of movement	Used functions
3	Movement calculated with a predefined movement	POS, BRK, STP, K206, K209, K210, K229, K230, MMC

Remark: This movement cannot be concatenated with MMC = 1.

The movement stops either when the target has been reached or when a command such as POS, STP or BRK has been used. If such a command is used, the position is not lost and the movement can start again from this point.

13.3.4.2 Rotary movement calculated with a predefined movement (MMD=19)

This type of movement works as the previous one (with MMD=3) but a position range limit (parameter K27=maximum position) has been programmed. Parameter K209 allows the user to define the way of the rotation (negative, positive or the shortest).

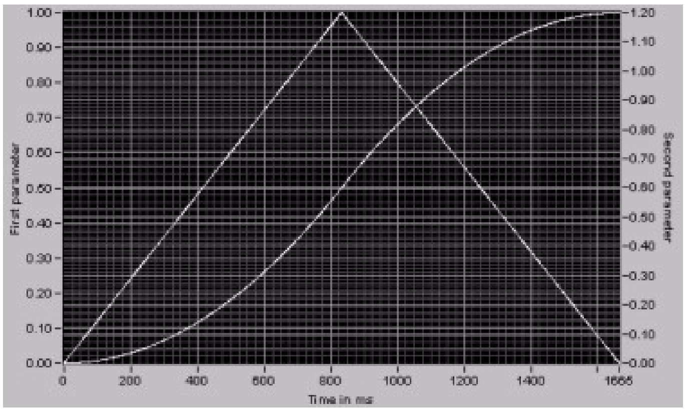
MMD.x	Type of movement	Used functions
19	Rotary movement calculated with a predefined movement	POS, BRK, STP, K27, K206, K209, K210, K229, K230, MMC

Remark: This movement cannot be concatenated with MMC = 1.

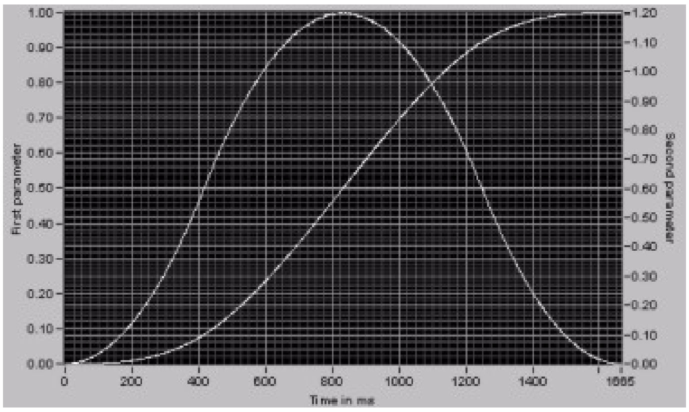
The movement stops either when the target has been reached or when a command such as POS, STP or BRK has been used. If such a command is used, the position is not lost and the movement can start again from this point.

13.3.4.3 Predefined movements examples with the different values of parameter K230 (MMD=3 or 19)

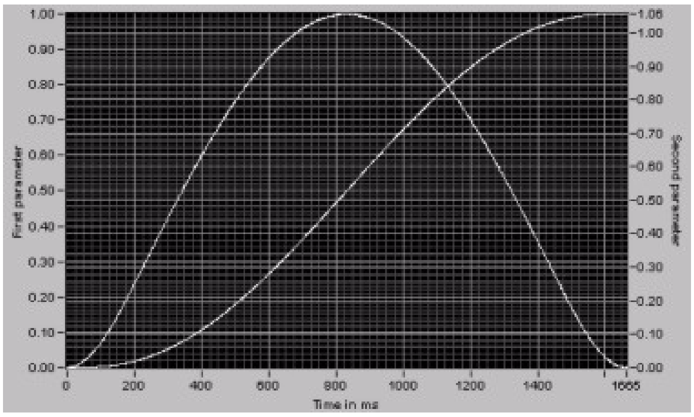
K230=0: Triangular movement (the graph represents the theoretical speed and position)



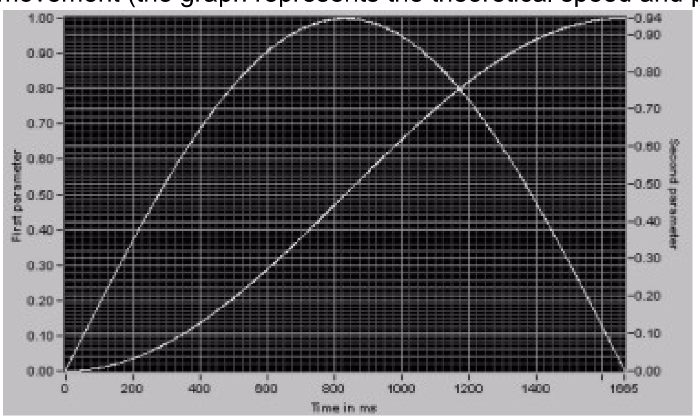
K230=1: S-curve movement (the graph represents the theoretical speed and position)



K230=2: Sine modified movement (the graph represents the theoretical speed and position)



K230=3: Real sine movement (the graph represents the theoretical speed and position)



13.3.5 Start movements: STA and STI commands

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

These commands start a movement with movement parameters (alias of POS, SPD, ACC, JRT,... command) stored at another depth than 0, as well as look-up table movements parameters stored at another depth than 0.

The **STI** command allows the user to start a movement according to the status of the digital input(s) defined by parameters K160 and K161 (refer to §13.3.5.1). As soon as the STI command is executed, the digital output(s) are set to the state defined by parameters K162 and K163 (refer to the next §). A **SYNCHRO START** error (M64=63) is sent to the controller if the movement does not start within the time-out defined by parameter K164 (time-out = K164 x 166.67µs for the DSC2P/DSC2V or K164 x 500 µs for the DSCDP, DSCDL and DSCDM).

<table>
<tr><th>Command format</th><th>Comments on commands</th><th><p1> values (depth)</th><th><p2> values</th><th><p2> bit #</th><th>Comments on bits mask <p2></th></tr>
<tr><td rowspan="11">STA.<axis> = <p1>, <p2>

STI.<axis> = <p1>, <p2></td><td rowspan="11">Starts a movement by using the movement parameters of the sub-indexes (depth 1 to 3) specified by <P1>, and <P2> is a mask of the movement parameter to be loaded

Synchronous start (sti) of a movement on an input by using the movement parameters of the sub-indexes specified by <P1>, and <P2> is a mask to enable movement variable to load</td><td rowspan="11">1 - 3</td><td>1</td><td>0</td><td>Gets target position (POS = K210) from the specified index. Also possible during a trapezoïdal movement if MMC=1.</td></tr>
<tr><td>2</td><td>1</td><td>Gets profile velocity (SPD = K211) from the specified index. Also possible during a trapezoïdal movement if MMC=1.</td></tr>
<tr><td>4</td><td>2</td><td>Gets profile acceleration (ACC = K212) from the specified index. Also possible during a trapezoïdal movement if MMC=1.</td></tr>
<tr><td>8</td><td>3</td><td>Gets jerk filter time (JRT = K213) from the specified index.</td></tr>
<tr><td>64</td><td>6</td><td>Gets profile type (MMD = K202) from the specified index.</td></tr>
<tr><td>128</td><td>7</td><td>Gets user look-up table number (LTN = K203) and the look-up table number selection mode (parameter K207) from the specified index.</td></tr>
<tr><td>256</td><td>8</td><td>Gets look-up table time (LTI = K204) from the specified index.</td></tr>
<tr><td>512</td><td>9</td><td>Gets user amplitude for look-up table (parameter K208) in mode K207=1.</td></tr>
<tr><td>1024</td><td>10</td><td>Gets rotary movement type selection (K209), only when the motor is not moving (bit# 4 of SD1 at 0).</td></tr>
<tr><td>2048</td><td>11</td><td>Gets the value of parameter K230 from the specified index.</td></tr>
<tr><td>4096</td><td>12</td><td>Gets the value of parameter K229 from the specified index.</td></tr>
</table>

Note: The depth where the movement characteristics are stored is defined in <p1>.
A mask of bits is used in <p2> to select which movement parameters are used among all parameters present in the depth defined in <p1>.

If **<p2> value = 0**, the controller considers that **all <p2> bits = 1** (bits# 0-12: 8191 decimal)

Do not use values 16 (bit 4) or 32 (bit 5) for <p2> of STA and STI commands.

STA and STI commands: bits mask description for values in <p2>				
Movement parameter in the mask	**Bit value**	**Bit #**	**Parameter function** (refer to §12.10.3 & §13.3.1.1)	**Behavior with S-Curve concatenated movements (MMC=1)**
K210	1	0	Position to reach (alias is POS command)	Recaptured
K211	2	1	Limit speed (alias is SPD command)	Recaptured
K212	4	2	Limit acceleration (alias is ACC command)	Recaptured
K213	8	3	Jerk time (alias is JRT command)	Ignored
-	-	4	Reserved for future use	-
-	-	5	Reserved for future use	-
K202	64	6	Movement type (alias is MMD command)	Ignored
K203 / K207	128	7	LKT number (alias is LTN command) / LKT start and end positions	Ignored
K204	256	8	Execution time of the LKT (alias is LTI command)	Ignored
K208	512	9	LKT maximum stroke	Ignored
K209	1024	10	Way of rotation movement	Ignored
K230	2048	11	Calculated movement (predefined profile) time	Ignored
K229	4096	12	Calculated movement (predefined profile) type	Ignored

The bits set to 1 mean that the values programmed at the defined depth will be kept to replace the values at the depth 0. The reserved bits must be set to 0.

Caution: When the whole **bits field is equal to 0**, the controller considers that **all bits are equal to 1!**

Example:

PWR.1=1, IND.1, WTM.1 commands are from now on omitted to avoid example overloading.

`MMD.1=1`	;Selects the S-Curve movement
`ACC.1=500000`	;Defines a_{max} at the depth 0
`SPD.1=200000`	;Defines v_{max} at the depth 0
`ACC:1.1=700000`	;Defines a_{max} at the depth 1
`SPD:1.1=400000`	;Defines v_{max} at the depth 1
`POS:1.1=200000`	;Defines x_{final} at the depth 1. The motor does not move. Only a POS at the depth 0 can move the motor.
`ACC:2.1=600000`	;Defines a_{max} at the depth 2.
`SPD:2.1=300000`	;Defines v_{max} at the depth 2.
`POS:2.1=800000`	;Defines x_{final} at the depth 2.

At this moment the motor has not moved yet but all the values are stored at different depths.

`POS.1=300000`	;The motor moves to the position 300000 with an acceleration of 500000 and a speed of 200000 increments.
`STA.1=2,7`	;Starts a movement with the values contained at the depth 2. The motor moves to the position 800000 with an acceleration of 600000 and a speed of 300000 increments.
`STA.1=1,7`	;Starts a movement with the values contained at the depth 1. The motor moves to the position 200000 with an acceleration of 700000 and a speed of 400000 increments.

Remark: STA and STI commands copy at the depth 0, the POS, SPD, ACC and JRT values contained at the depth where the movement will be executed. Therefore each STA or STI execution will see the **depth 0 values crushed**. In the above example, after executing the first STA command (STA.1=2,7), the acceleration of 500000, the speed of 200000 which were initially at the depth 0 are lost. However, all the others at the depths other than 0 remain unchanged.

If for any kind of reason, one of the axis cannot execute the movement started by **STI** within the time defined in parameter K164 (because another movement is being realized, eg.), all controllers switch to error mode and the **SYNCHRO START** error (M64=63) is displayed on the LCD screen of this axis. This will never happen with the STA command, where each available axis immediately executes the movement and the others do it as soon as they can.

13.3.5.1 DIN and DOUT with the STI command

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

With command STI, the synchronized start of the movements on one or several axis depends on the status (0/I) of one or several DIN(s). They are defined by parameters **K160** and **K161**.

When a synchronized movement starts on one or several axis with command STI, one or several DOUT(s) are set or reset (I/0). They are defined by parameters **K162** and **K163**.

K160 and K161: DIN mask and status

K	Name	Comment
K160	STI DIN mask	Defines with a mask of bits which DIN will be tested to execute a STI command.
K161	STI DIN status	Defines with a mask of bits which DIN status is required to start the movement after a STI command.

Remark: Parameter K160 is always $\geq$ K161

The controller's DINs are represented by a mask of bits, to be used with the STI command. The number of digital inputs is different from a position controller to another. Here is a table giving the number of digital inputs present on each position controller (refer to the corresponding **'Hardware Manual'** for more information):

DSC2P / DSC2V	**DIN #**	10	9	-	7	6	5	4	3	2	1
DSCDP / DSCDL	**DIN #**	10	9	-	-	-	-	-	-	2	1
DSCDM	**DIN #**	-	-	-	-	-	-	-	3	2	1
Bit #		9	8	-	6	5	4	3	2	1	0
Decimal value		512	256	-	64	32	16	8	4	2	1
Hexadecimal value		200	100	-	40	20	10	8	4	2	1

Caution: On the DSCDM, the digital inputs and outputs are on the same pin. The pin must be selected in order to have an input **or** an output. **It is NOT possible to have both on the same pin.** To use a pin as a digital input, the bit corresponding to this input **MUST** be equal to 0 in parameter K171 (otherwise the hardware of the controller and the one of the user can be damaged).

Example:

`K160.1 = 513` ;513 = (1 + 512) means that the inputs DIN1(=1, bit# 0) and DIN10 (= 512, bit# 9) will be tested to execute a STI command
`K161.1 = 512` ;the movement will start when DIN1 = 0 and DIN10 = 1 (bit# 9, value 512)

K162 and K163: DOUT mask and status

K	Name	Comment
K162	STI DOUT mask	Defines with a mask of bits which DOUT will be modified when a STI command is executed.
K163	STI DOUT status	Defines with a mask of bits which DOUT status will be set after a STI command.

Remark: Parameter K162 is always $\geq$ K163

The controller's DOUTs are represented by a mask of bits, to be used with the STI command. The number of digital outputs is different from a position controller to another. Here is a table giving the number of digital outputs present on each position controller (refer to the corresponding **'Hardware Manual'** for more information):

DSC2P / DSC2V	**DOUT #**	4	3	2	1
DSCDP / DSCDL	**DOUT #**	-	-	2	1
DSCDM	**DOUT #**	-	3	2	1
Bit #		3	2	1	0
Decimal and hexadecimal value		8	4	2	1

Caution: On the DSCDM, the digital inputs and outputs are on the same pin. The pin must be selected in order to have an input **or** an output. **It is NOT possible to have both on the same pin.** The state of the digital output is given by the value of parameter K171.

Example:

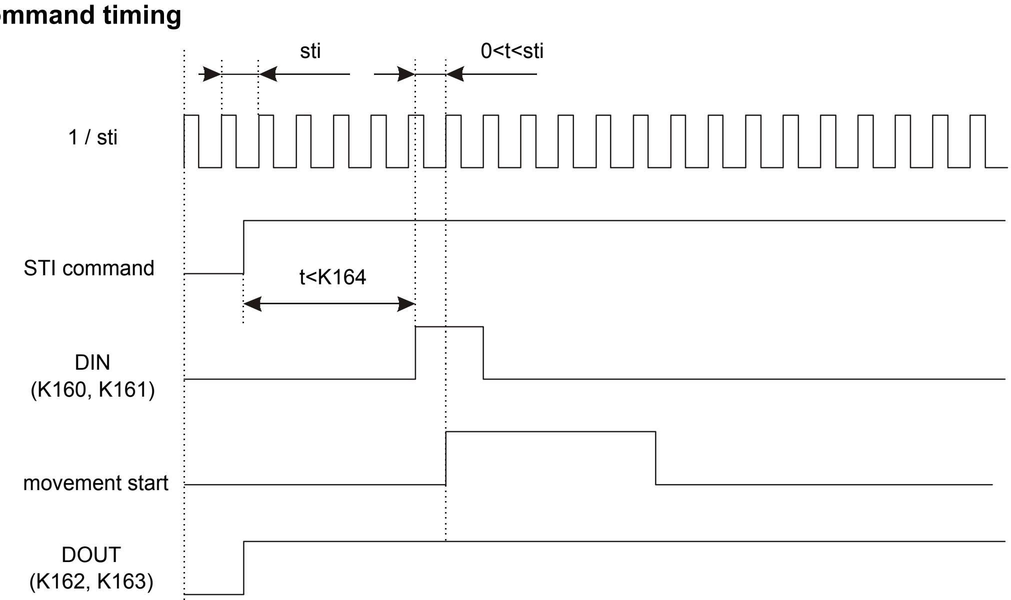
`K162.1 = 15` ;15=(1+2+4+8) means that all the DOUTs of the controller will be modified when a STI command is executed.
`K163.1 = 2` ;the status will be DOUT2=1 (bit# 1, value 2), and all the other DOUTs = 0.

Parameter **K164**: time limit for STI start

K	Name	Comment	Units
K164	STI time-out limit	Defines the time limit between the moment when the STI command is sent and the moment when the DIN (defined in mask K160) are set to the required values (defined in mask K161). If this time is exceeded, a **SYNCHRO START** error (M64=63) appears on the controller.	[sti]

The ISO time is for the DSC2P/DSC2V: K164 x 166.67µs. Thus, K164 max. value = 100[sec].
DSCDP, DSCDL and DSCDM: K164 x 500µs. Thus, K164 max. value = 300[sec].

STI command timing



STI use example:

The user wants the movement to begin when DIN1=0 and DIN10=1.
Furthermore, the user wants DOUT2 to move to 1 and DOUT1 to 0 when the movement begins.
`K160.1 = 513` ;513 = (1 + 512) means that the inputs DIN1(=1, bit# 0) and DIN10 (= 512, bit# 9) will be tested to execute a STI command

`K161.1 = 512`	;the movement will start when DIN10 = I (bit# 9, value 512) and DIN1 = 0
`K162.1 = 3`	;3 = (1 + 2) means that the outputs DOUT1(=1, bit# 0) and DOUT2(=2, bit# 1) will be modified when a STI command is executed.
`K163.1 =2`	;the status will be DOUT1=0, DOUT2=I (bit# 1, value 2).
`K164 = 6000`	;time-out of 1 sec. (for DSC2P/DSC2V) and 3 sec. (for DSCDP,DSCDL and DSCDM)
	....
	STI.1 = 1,1
	....

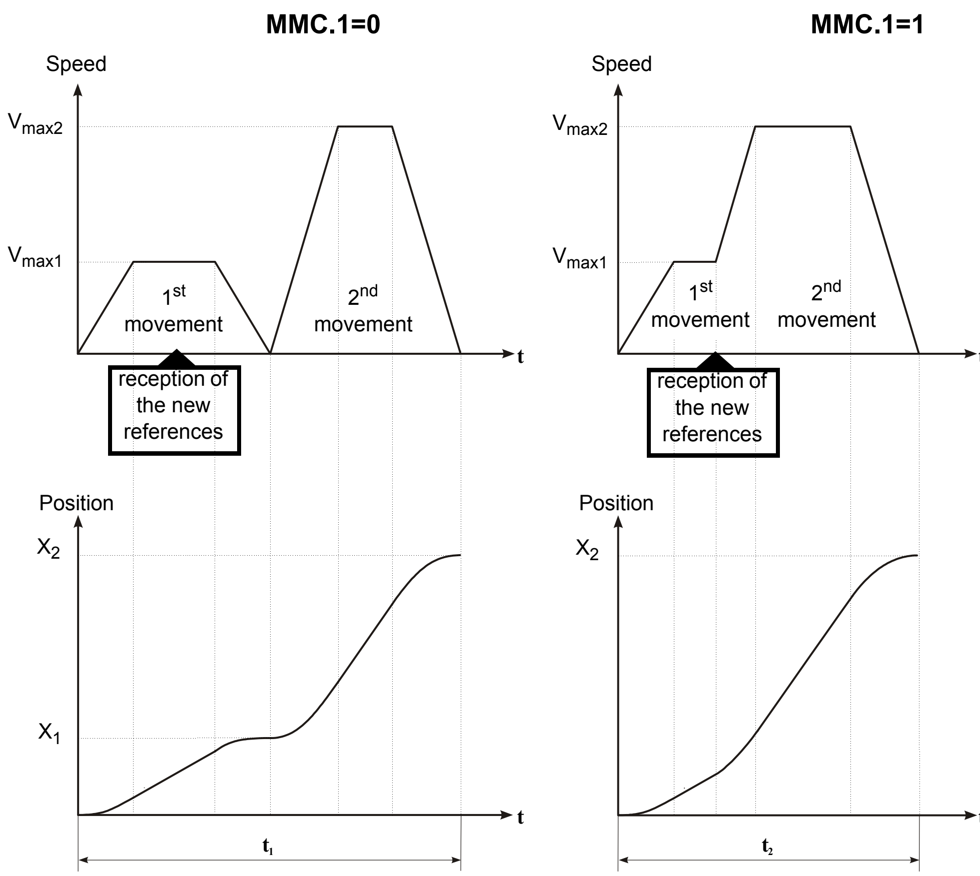
13.3.6 Concatenated movements: MMC command

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

If two POS commands follow each other, the movements are made one after the other, so that the first must be finished before the second starts. However, some applications need to change the final position (x_{final}) to reach during a movement. The **MMC** command (**M**ovement **M**ode **C**oncatenated), which is an alias of parameter **K201**, allows to realize this function and even more, since also speed and acceleration can be changed during the movement.

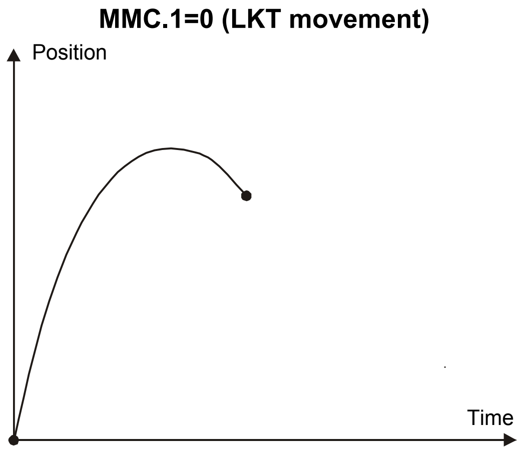
K	Alias	<p1>	Comment
K201	MMC	0 1 2 3	Concatenated movements disabled Concatenated movements for MMD=1, 17 & 24 Infinite back and forth movement for MMD=3, 10, 19 & 26 One back and forth movement. for MMD=3, 10, 19 & 26

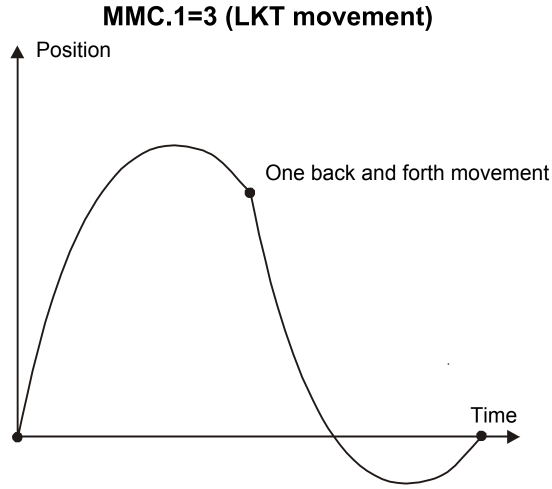
When concatenated movements mode is selected (MMC.1=1), and a second movement starts, the controller immediately calculates the new speed trajectory and the motor switches from one movement to the other without going through the 0 speed. The first movement does not end and the controller immediately considers the new position, the new maximum speed and the new maximum acceleration.

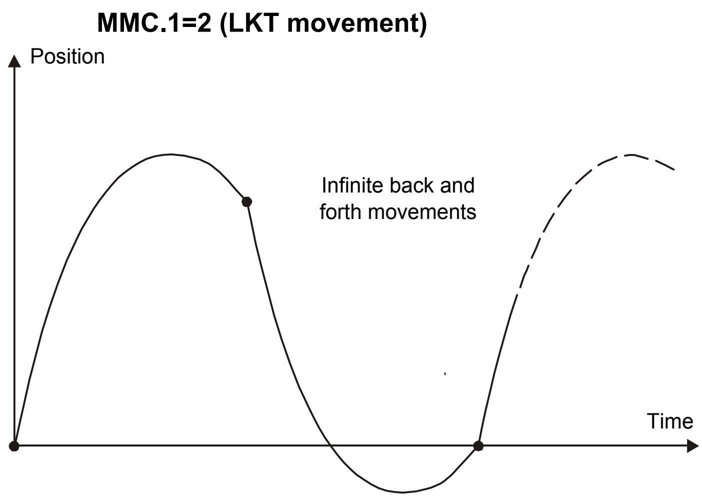


Remark: For the same final position, times t_1 and t_2 are not identical; time t_2 is shorter.

Examples with LKT movements:







Remark: If the MMC command does not appear in the sequence, the movements without concatenation mode are used by default.

Example:

Concatenated movements are simple to execute. Just select the concatenated movements mode and execute the memorized movements at different depths.

`MMD.1=1`	;Selects the S-curve movement.
`MMC.1=1`	;Selects the concatenated movements mode.
`ACC.1=500000`	;Defines a_{max} at the depth 0.
`SPD.1=200000`	;Defines v_{max} at the depth 0.
`ACC:1.1=700000`	;Defines a_{max} at the depth 1.
`SPD:1.1=400000`	;Defines v_{max} at the depth 1.
`POS:1.1=200000`	;Defines x_{final} at the depth 1. The motor does not move. Only the POS command at the depth 0 can move the motor.
`POS.1=100000`	;The motor moves to the position 100000
`STA.1=1,...`	;The memorized movement at the depth 1 is started. If the first movement is not finished, it is interrupted. The motor will increase its speed because the movement speed at the depth 1 is higher than that of the previous movement and it will finally stop at the position 200000.

13.3.7 CAM command

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **CAM** command is an alias of parameter **K205** which globally diminishes by a certain percentage the **cinematic quantities** such as speed, acceleration and jerk time of a movement (but not the position to reach). Similarly, **time quantities** such as time, lapse of time between two movements with the WTT command, execution time of a look-up table movement, etc., are increased in the same proportion. The formulae which calculates the new values of these quantities according to the number included in the CAM command are given below.

For cinematic quantities the formula is as follows:

- speed: $\frac{\text{Cinematic quantity} \cdot \text{CAM}}{100}$
- acceleration: $\frac{\text{Cinematic quantity} \cdot \text{CAM}^2}{100^2}$

For time quantities the formula is: $\frac{\text{Time quantity} \cdot 100}{\text{CAM}}$

K	Alias	Name	Comments	Units
K205	CAM	CAM value	Global diminution of the characteristics of a movement.	[%]

Remark: CAM alias use the same syntax than parameter K205. Refer to §6.3 for more information about the syntax and the possible operators.

The value programmed with the CAM command remains always active. In order to come back to the basic values the command CAM.1=100 must be entered.

Example:

`MMD.1=1`	;S-curve movement selection
`ACC.1=500000`	;Definition of a_{max}.
`SPD.1=200000`	;Definition of v_{max}.
`POS.1=300000`	;The motor moves to the position 300000 with an acceleration of 500000 and a speed of 200000.
`CAM.1=20`	;Global diminution to 20%.
`POS.1=5000`	;The motor moves to the position 5000 with a speed and an acceleration decreased of 80%. Only 20% of these values are kept. The new speed will be:

$$\frac{200000 \cdot 20}{100} = 40000$$ [usi] and the new acceleration will be:

$$\frac{500000 \cdot 20^2}{100^2} = 8000$$ [uai].

13.3.8 STE command

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **STE** command (**STE**p) is a command which enables the motor to make jump movements. It is used by the ETEL Tools program to find out the right values of the regulator different gains.

Caution: This command must be used with the **utmost prudence**, since movement speed is very high.

Command format	[s]	<p1>	Comment	Units
STE.<axis>[s] = <p1>	Sign + or - or nothing	-2^{31} to $(2^{31}-1)$	Performs a step movement of the length contained in <p1>	[upi] [rupi]

Although the STE command is not an alias, the syntax STE.1+=1000 or STE.1-=1000 are allowed, in order to make **relative** jump movements, because an absolute jump movement can be dangerous if the motor is far away from the position to reach.

Examples:

`STE.1+=1000` ;**Relative** jump movement. the motor realizes a jump movement of 1000 increments in the positive direction.

`STE.1=1000` ;**Absolute** jump movement. The motor makes a jump movement from its actual position to the position of 1000 increments.

13.3.9 BRK and STP commands

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The **BRK** command (**BR**a**K**e) stops a movement with the deceleration programmed in parameter K206. It is also used with infinite rotary movement (refer to §13.3.3).

The **STP** command (**ST**o**P**) behaves the same way as BRK but the deceleration is infinite. The braking is then very sudden. This command is to be used only in case of emergency.

Caution: The sudden deceleration set with the STP command can damage the mechanical parts of the system.

Note: BRK and STP commands do not stop the sequence; thus, the movements defined in a sequence will be executed.

Command format	Comment
BRK.<axis>	Stops the movement using the deceleration programmed in parameter K206 but it does not stop the sequence. Stops an infinite rotary movement.
STP.<axis>	Stops the movement using an infinite deceleration but it does not stop the sequence.

13.3.10 Parameters defining units scales

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Parameter **K50** defines the unit scale movements:

K	Name	Comment
K50	Set point calculator shift	Enables the diminution of the cinematic quantities values. 1dpi = 2^{K50} * 1upi Depth 0 for primary encoder and depth 1 for secondary encoder **(depth 1 is ONLY available on DSC2P and DSC2V)**

Parameter K50:

The POS, SPD and ACC commands are used respectively to determine a position to reach, a speed or an acceleration. The arguments of these commands must be given in a unit called **user increment**. The conversion formulae of these units into the IS0 system are given in §19. Parameter K50 appearing at the denominator of the formula decreases the position, speed and acceleration **user resolution** in case of the values exceed the maximum controlled by the controller without going over the system's performances.

Example (for the DSC2P/DSC2V):

Supposing that a movement with a speed of 1 m/s is planned with a linear motor with a 4µm period encoder. Parameters K77 and K50 are chosen with respective values of 8 and 0, and a sampling period of h=166.67µs. The conversion equation of the user speed increment unit [usi] gives the speed to be programmed in user increment:

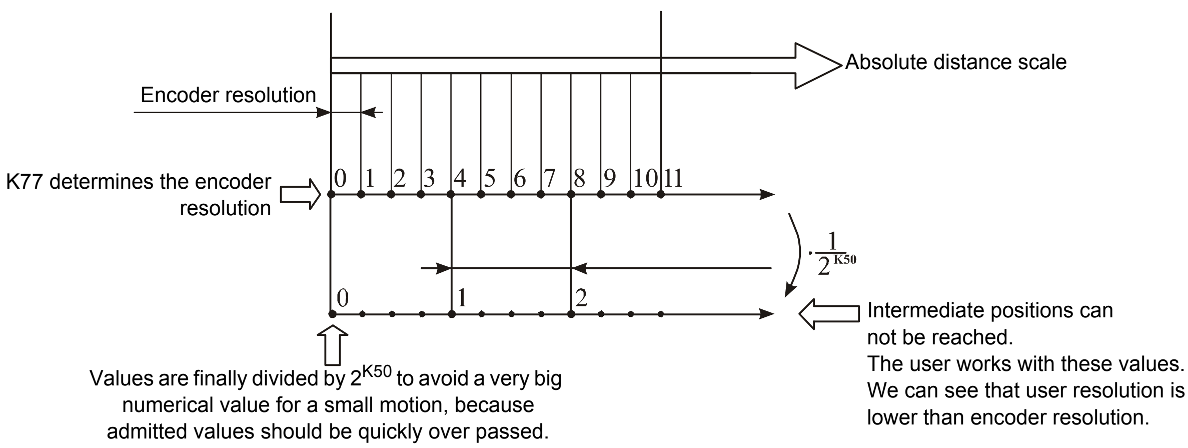
$$\text{Speed [usi]} = \text{Speed [m/s]} \cdot \frac{256 \cdot 1024 \cdot 2^{K77} \cdot h}{\text{PCod} \cdot 2^{K50}} = 1 \cdot \frac{256 \cdot 1024 \cdot 2^{8} \cdot 166 \cdot 10^{-6}}{4 \cdot 10^{-6} \cdot 2^{0}} = 2,8 \cdot 10^{9}$$

Remark: The same calculation can be done for the DSCDP, DSCDL and DSCDM but h = 500µs.

A speed of 2,8 billion increment exceeds the maximum value the controller can control, which is of 2 billion. It is therefore impossible to reach the 1m/s requested in this configuration even though the motor and the encoder system easily allow it. To change this, the value 1 is introduced in parameter K50 of the above formula. A speed of $1{,}4\times10^{9}$ [usi] for 1 m/s is then obtained and this value is accepted by the controller.

Remark: Increasing parameter K50 diminishes the system's position, speed and acceleration resolution. However the controller's internal calculations (the regulator e.g.) do not include this parameter and the resolution is not decreased.

Parameter **K77** (refer to §13.11) determines the smallest distance an **analog encoder** (resolution) can measure (this is parameter **K69** for **TTL encoders**). Increasing parameter K77 (or K69) allows also to diminish gain values in the position regulator if they are too big. In the end only the user distance is divided by 2^{K50} in order to avoid to be too high and to exceed the maximum limit. The same operations can be made with the other cinematic quantities such as speed, acceleration and jerk.



13.3.11 Movements equations

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Note: The understanding of these equations is not necessary to operate the controller, they are given only for information.

13.3.11.1 Constantly accelerated linear movement (CALM)

A **CALM** is a movement with a constant acceleration. The movement trajectory of a constantly accelerated linear movement is described by the following equation:

$$x(t) = \frac{1}{2}at^2 + v_0 t + x_0$$

with x = position versus time
v_0 = initial speed
x_0 = initial position

The derivatives of this equation versus time determine the CALM speed, acceleration and jerk trajectory:

$$\begin{cases} v(t) = at + v_0 \\ a(t) = a = \text{constant} \\ j(t) = 0 \end{cases}$$

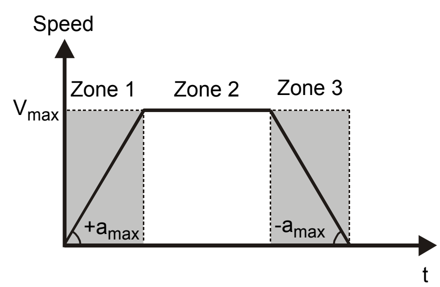
13.3.11.2 Step movement

The **step movement** is a very blunt movement in which the motor is asked to go immediately from a position to another. In such a movement, the speed, acceleration and jerk are infinite, therefore, no system is able to perfectly perform it. A step movement is entirely defined by the final position x_{final} to reach (STE command). This movement is generally only used to find the optimal values of the position and current loop regulators.

13.3.11.3 Trapezoidal movement

(For principle understanding - not used in the controller, S-curve movement is preferred).

A **trapezoidal movement** is a movement whose speed trajectory has a trapezoidal shape. There are three distinct zones. In zone 1, a CALM (Constantly Accelerated Linear Movement) is used with a positive constant tangential acceleration $+a_{max}$. In zone 2, there is no tangential acceleration (because the tangential speed is constant). In zone 3, there is a negative tangential acceleration (deceleration) $-a_{max}$. Therefore, the mobile can slow down and reach its aim without speed.



A trapezoidal movement is entirely defined by the length of the segment to reach (set up by the controller with the POS command), the maximum tangential speed V_{max} (given with the SPD command), and the tangential acceleration a_{max} (ACC command). Those three values can be displayed on the speed trajectory graph. V_{max} corresponds to the maximum of the curve and the tangential accelerations ($+a_{max}$, 0, $-a_{max}$) are the slopes of the three segments which make up the trapezium. The final segment length is given by the surface of the trapezium.

Here are the equations found in a trapezoidal movement:

Zone 1 $$\begin{cases} x(t) = \frac{1}{2}a_0 t^2 + v_0 t + x_0 \\ v(t) = a_0 t + v_0 \\ a(t) = a_0 = \text{cst with } a_0 > 0 \\ j(t) = 0 \end{cases}$$

Zone 2 $$\begin{cases} x(t) = v_0 t + x_0 \\ v(t) = v_0 = cst \\ a(t) = 0 \\ j(t) = 0 \end{cases}$$

Zone 3: same as zone 1 with a < 0

Remark: The maximum speed is not necessarily reached with a trapezoidal movement. In that case, we have a **triangular movement**. This movement is optimal in time.

13.3.11.4 Rectangular movement

(For understanding principle - not implemented in the controller).

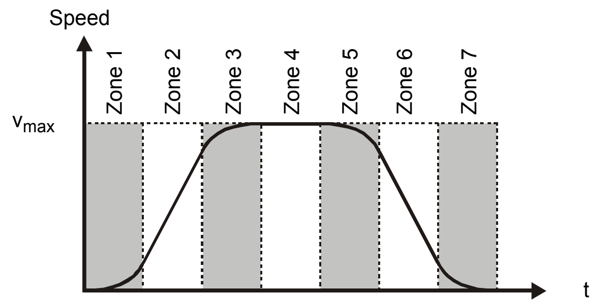
The **rectangular movement** is a special case of trapezoidal movement. Its speed trajectory has a rectangular

shape. It is very blunt because motor speed moves from 0 to v_{max} suddenly at the beginning and from v_{max} to 0 at the end. A motor cannot move exactly over this speed trajectory because it would imply infinite acceleration and deceleration. This type of movement can be used for very small movements at high speed. The rectangular movement is entirely defined by the position to reach (POS command) and by movement speed (SPD command).
A rectangular movement equation is very simple because it only has one zone with constant speed.

$$\begin{cases} x(t) = vt + x_0 \\ v(t) = v = cst \\ a(t) = 0 \\ j(t) = 0 \end{cases}$$

13.3.11.5 S-Curve movement

The S-Curve movement is even smoother than the trapezoidal movement because it can avoid infinite jerk (derivative of the acceleration). This movement avoids as much as possible the shocks in the machine and increases mechanical parts in motion and bearings lifetime. But, with the same maximum speed, acceleration and final position, the motor's S-curve movement will last longer than a trapezoidal movement.



Here are the equations of the S-Curve movement:

Zone 3: same as zone 1 with j < 0

Zone 4: $$\begin{cases} x(t) = v_0 t + x_0 \\ v(t) = v_0 = cst \\ a(t) = 0 \\ j(t) = 0 \end{cases}$$

Zone 5: same as zone 3 with a < 0

Zone 6: same as zone 2 with a < 0

Zone 7: same as zone 1 with a < 0

Zone 1 $$\begin{cases} x(t) = \frac{1}{6} j_0 t^3 + \frac{1}{2} a_0 t^2 + v_0 t + x_0 \\ v(t) = \frac{1}{2} j_0 t^2 + a_0 t + v_0 \\ a(t) = j_0 t + a_0 \\ j(t) = j_0 = \text{cst with } j_0 > 0 \end{cases}$$

Zone 2 $$\begin{cases} x(t) = \frac{1}{2} a_0 t^2 + v_0 t + x_0 \\ v(t) = a_0 t + v_0 \\ a(t) = a_0 = cst \\ j(t) = 0 \end{cases}$$

As the S-curve movement uses 3rd degree equations, it needs one more parameter than the trapezoidal movement to be defined. This parameter is called **jerk time** which is programmed with the JRT command and defines the additional time (number of times of sti (refer to §4.) that the S-Curve movement will last with respect to a trapezoidal movement.

Example:
If a movement of 10cm takes 0.5 seconds with a trapezoidal profile, the same movement will last 20 x 166.67µs for the DSC2P/DSC2V or 20 x 500µs for the DSCDP, DSCDL and DSCDM longer if a jerk time of 20 [sti] is programmed, so the total time will be: 0.5s + (20x166.67µs) = 0.503s for the DSC2P/DSC2V or 0.5s + (20x500µs) = 0.51s for the DSCDP, DSCDL and DSCDM.

Caution: If a very big jerk time is used, the maximum programmed speed will not be reached.

Remark: A S-curve movement without any jerk time is equal to a trapezoidal movement.

13.4 Digital inputs / outputs

13.4.1 Digital inputs

The number of digital inputs is different from a position controller to another. Here is a table giving the number of digital inputs present on each position controller (refer to the corresponding **'Hardware Manual'** for more information):

DSC2P / DSC2V	**DIN #**	10	9	-	7	6	5	4	3	2	1
DSCDP / DSCDL	**DIN #**	10	9	-	-	-	-	-	-	2	1
DSCDM	**DIN #**	-	-	-	-	-	-	-	3	2	1

Caution: On the DSCDM, the digital inputs and outputs are on the same pin. The pin must be selected in order to have an input **or** an output. **It is NOT possible to have both on the same pin.** To use a pin as a digital input, the bit corresponding to this input **MUST** be equal to 0 in parameter K171 (otherwise the hardware of the controller and the one of the user can be damaged).

13.4.1.1 Digital inputs on position controller

Available on	**DSC2P**	**DSC2V**	**DSCDP**	**DSCDL**	**DSCDM**

The DIN command enables the reading of the digital inputs states. This command is an alias of monitoring **M50**.

M	Alias	Name	Comment
M50	DIN	DIN status	Gives the status of the digital inputs: DIN1 (bit# 0), DIN2 (bit# 1)...

A bit equal to 1 means that the corresponding digital input is activated and equal to 0 means that the corresponding digital input is deactivated. The digital inputs bits values are as follows:

DIN #	10	9	-	7	6	5	4	3	2	1
Bit #	9	8	-	6	5	4	3	2	1	0
Decimal value	512	256	-	64	32	16	8	4	2	1
Hexadecimal value	200	100	-	40	20	10	8	4	2	1

Example:

A simple conversion in binary of the value shown by DIN (in hexadecimal) is enough to know which digital inputs are activated and deactivated.

DIN values		**DIN10**	**DIN9**	**-**	**DIN7**	**DIN6**	**DIN5**	**DIN4**	**DIN3**	**DIN2**	**DIN1**
		Bit# 9	**Bit# 8**	**-**	**Bit# 6**	**Bit# 5**	**Bit# 4**	**Bit# 3**	**Bit# 2**	**Bit# 1**	**Bit# 0**
3	**3**	0	0	-	0	0	0	0	0	1	1
769	**301**	1	1	-	0	0	0	0	0	0	1
Decimal	**Hexa.**	**Binary**									

In the first line of the above-mentioned example, DIN1 and DIN2 are activated.

`DIN.1` ;Reads the digital inputs state. The controller gives the value 3 in hexa, because 0x3 in binary is 0000000011 and each bit represents one of the digital inputs, from right to left.

In the second line of the above-mentioned example DIN1, DIN9 and DIN10 are activated.

`DIN.1` ;Reads the digital inputs state. The controller gives the value 301 in hexa, because 0x301 in binary is 1100000001 and each bit represents one of the digital inputs, from right to left.

Remark: Some digital inputs can also be used for external limit switches and external home switch, if these switches are present in the application. Refer to §12.9 for more information.

13.4.1.2 Digital inputs on DSO-HIO

Available on	DSC2P	DSC2V	DSCDP	DSCDL	~~DSCDM~~

The DSO-HIO has 8 digital inputs (refer to the **'DSO-HIO User's Manual'** for more information).

Remark: For the DSCDP and DSCDL, the 8 digital inputs are available for both axes.

The XDIN command allows the reading of the 8 digital inputs states: XDIN1 to XDIN8. This command is an alias of monitoring **M55**.

M	Alias	Name	Comment
M55	XDIN	DSO-HIO's DIN status	Gives the status of the DSO-HIO digital inputs: XDIN1 (bit# 0) to XDIN8 (bit# 1)

A bit equal to 1 means that the corresponding digital input is activated and equal to 0 means that the corresponding digital input is deactivated. The digital inputs bits values are as follows:

XDIN #	8	7	6	5	4	3	2	1
Bit #	7	6	5	4	3	2	1	0
Decimal value	128	64	32	16	8	4	2	1
Hexadecimal value	80	40	20	10	8	4	2	1

Example:

A simple conversion in binary of the value shown by monitoring M55 (in hexadecimal) is enough to know which digital inputs are activated and deactivated.

XDIN values		**XDIN8** **Bit# 7**	**XDIN7** **Bit# 6**	**XDIN6** **Bit# 5**	**XDIN5** **Bit# 4**	**XDIN4** **Bit# 3**	**XDIN3** **Bit# 2**	**XDIN2** **Bit# 1**	**XDIN1** **Bit# 0**
16	**10**	0	0	0	1	0	0	0	0
255	**FF**	1	1	1	1	1	1	1	1
Decimal	**Hexa.**	**Binary**							

In the first line of the above-mentioned example, XDIN5 is activated.

`M55.1` ;Reads the digital inputs state. The controller gives the value 0x10 (in hexadecimal), because 0x10 in binary is 00010000 and each bit represents one of the eight digital inputs, from right to left.

In the second line of the above-mentioned example, XDIN1 to XDIN8 are activated.

`M55.1` ;Reads the auxiliary digital inputs state. The controller gives the value 0xFF (in hexadecimal), because 0xFF in binary is 11111111 and each bit represents one of the eight digital inputs, from right to left.

13.4.2 Digital outputs

The number of digital outputs is different from a position controller to another. Here is a table giving the number of digital outputs present on each position controller (refer to the corresponding **'Hardware Manual'** for more information):

DSC2P / DSC2V	**DOUT #**	4	3	2	1
DSCDP / DSCDL	**DOUT #**	-	-	2	1
DSCDM	**DOUT #**	-	3	2	1

Caution: On the DSCDM, the digital inputs and outputs are on the same pin. The pin must be selected in order to have an input **or** an output. **It is NOT possible to have both on the same pin.** The state of the digital output is given by the value of parameter K171.

13.4.2.1 Digital outputs on position controller

Available on	**DSC2P**	**DSC2V**	**DSCDP**	**DSCDL**	**DSCDM**

The DOUT command enables the user to activate or deactivate the digital outputs. This command is an alias of parameter **K171**.

K	Alias	Name	Comment
K171	DOUT	DOUT value	Activates / deactivates the digital outputs according to the value

Remark: DOUT alias use the same syntax than parameter K171. Refer to §6.3 for more information about the syntax and the possible operators.

A bit equal to 1 means that the corresponding digital input is activated and equal to 0 means that the corresponding digital input is deactivated. The digital inputs bits values are as follows:

DOUT #	4	3	2	1
Bit #	3	2	1	0
Decimal and hexadecimal value	8	4	2	1

The DOUTs state can be read with monitoring **M171** which takes parameters K171 and K37 into account:

M	Name	Comment
M171	DOUT status	Gives the status of the digital outputs of the controller

Example:

A simple conversion in binary of the value shown by monitoring M171 (in hexadecimal) is enough to know which digital inputs are activated and deactivated.

DOUT values		**DOUT4** **Bit# 3**	**DOUT3** **Bit# 2**	**DOUT2** **Bit# 1**	**DOUT1** **Bit# 0**
15	**F**	1	1	1	1
4	**4**	0	1	0	0
Decimal	**Hexa.**	Binary			

In the first line of the above-mentioned example, DOUT1 to 4 are activated.

`DOUT.0=15` ;Activates DOUT1, 2, 3 and 4.

In the second line of the above-mentioned example, DOUT3 is activated and DOUT1, 2 and 4 are deactivated.

`DOUT.0=4` ;Activates DOUT3 and deactivates DOUT1, 2 and 4.

13.4.2.2 Digital outputs on DSO-HIO

Available on	DSC2P	DSC2V	DSCDP	DSCDL	~~DSCDM~~

The DSO-HIO has 8 digital outputs (refer to the **'DSO-HIO User's Manual'** for more information).

Remark: For the DSCDP and DSCDL, the 8 digital outputs are available for both axes.

The XDOUT command allows the user to activate or deactivate the 8 digital outputs of the DSO-HIO: XDOUT1 to XDOUT8. This command is an alias of parameter **K172**.

K	Alias	Name	Comment
K172	XDOUT	XDOUT value	Activates / deactivates the 8 digital outputs (XDOUT1 to XDOUT8) of the DSO-HIO optional board

Remark: For the DSCDP and DSCDL, if parameter K172 is modified, the SAV command must be executed on both axes.
XDOUT alias use the same syntax than parameter K172. Refer to §6.3 for more information about the syntax and the possible operators.

A bit equal to 1 means that the corresponding digital input is activated and equal to 0 means that the corresponding digital input is deactivated. The digital inputs bits values are as follows:

XDOUT #	8	7	6	5	4	3	2	1
Bit #	7	6	5	4	3	2	1	0
Decimal value	128	64	32	16	8	4	2	1
Hexadecimal value	80	40	20	10	8	4	2	1

Remark: On the DSCDP and DSCDL, if parameter K172 of one axis is changed, K172 of the other axis will be automatically changed.

Example:

XDOUT value		**XDOUT8 Bit# 7**	**XDOUT7 Bit# 6**	**XDOUT6 Bit# 5**	**XDOUT5 Bit# 4**	**XDOUT4 Bit# 3**	**XDOUT3 Bit# 2**	**XDOUT2 Bit# 1**	**XDOUT1 Bit# 0**
FF	**255**	1	1	1	1	1	1	1	1
1	**1**	0	0	0	0	0	0	0	1
AA	**170**	1	0	1	0	1	0	1	0
Hexa.	**Decimal**	**Binary**							

In the first line of the above-mentioned example, XDOUT1 to 8 are activated.

`K172.1=255` ;Activates XDOUT1 to 8.

In the second line of the above-mentioned example, XDOUT1 is activated and XDOUT2 to 8 are deactivated.

`K172.1=1` ;Activates XDOUT1 and deactivates XDOUT2 to 8.

In the third line of the above-mentioned example, XDOUT2, 4, 6 and 8 are activated and XDOUT1, 3, 5 and 7 are deactivated.

`K172.1=170` ;Activates XDOUT2, 4, 6 and 8 and deactivates XDOUT1, 3, 5 and 7.

13.5 Position capture on digital inputs

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The number of digital inputs is different from a position controller to another. Here is a table giving the number of digital inputs present on each position controller (refer to the corresponding **'Hardware Manual'** for more information):

DSC2P / DSC2V	**DIN #**	10	9	-	7	6	5	4	3	2	1
DSCDP / DSCDL	**DIN #**	10	9	-	-	-	-	-	-	2	1
DSCDM	**DIN #**	-	-	-	-	-	-	-	3	2	1

Caution: **On the DSC2P and DSC2V, the digital inputs 5, 6 and 7 are NOT tested for the position capture.**

On the DSCDM, the digital inputs and outputs are on the same pin. The pin must be selected in order to have an input **or** an output. **It is NOT possible to have both on the same pin.** To use a pin as a digital input, the bit corresponding to this input **MUST** be equal to 0 in parameter K171 (otherwise the hardware of the controller and the one of the user can be damaged).

The inputs are tested every FTI (FTI = 41.67µs for the DSC2P/DSC2V and 55.56µs for the DSCDP, DSCDL and DSCDM) and when the condition(s) defined by parameters **K178** and/or **K179** is true the position is captured and saved in monitoring **M12** [upi].

K	Name	Comment
K178	Mask of the DINs on rising edge	Defines the mask of the digital inputs on the rising edge: K178, depth 0, bit 0, 1, 2, 3, 8 and 9 (bit0=DIN1, bit1=DIN2, bit2=DIN3, bit3=DIN4, bit8=DIN9 and bit9=DIN10)
K179	Mask of the DINs on falling edge	Defines the mask of the digital inputs on the falling edge: K179, depth 0, bit 0, 1, 2, 3, 8 and 9 (bit0=DIN1, bit1=DIN2, bit2=DIN3, bit3=DIN4, bit8=DIN9 and bit9=DIN10)

Remark: At least one condition defined by parameter K178 or K179 must be met to have a position capture.

M	Name	Comment
M12	Real position on DIN	Gives the real position (upi) captured on digital inputs

Parameter **K159** allows the user to invert the digital inputs for position capture

K	Name	Comment
K159	Inversion of the DINs	Inverts the digital inputs for position capture: K159, depth 0, bit 0, 1, 2, 3, 8 and 9 (bit0=DIN1, bit1=DIN2, bit2=DIN3, bit3=DIN4, bit8=DIN9 and bit9=DIN10)

Remark: If parameter K159 inverts the digital inputs, the rising edges become falling edges and vice versa. DIN1 cannot be inverted by parameter K159 if K33 = 0 because DIN1 is a safety input. Parameter K159 does not modify monitoring M50. It is only for the position capture test that the digital inputs are inverted.

Parameter **K182** allows the user to activate the position capture on the digital inputs.

K	Name	Comment
K182	Position capture on DIN	Activates the position capture on digital input according to parameters K178 and K179. Writing a 1 in parameter K182 activates the capture and resets (=0) bit# 2 of SD2 (monitoring M61) and bit# 30 of monitoring M63. Writing a 0 stops the process.

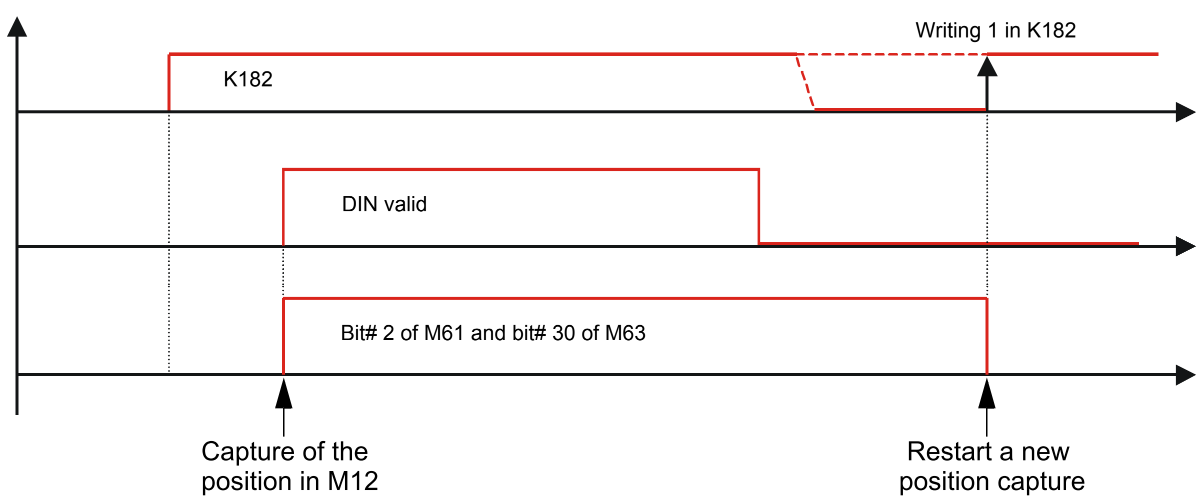
Remark: Parameter K182 is automatically set to 0 when the controller is switched on. This parameter cannot be saved in the controller. Writing 1 in parameter K182 starts the position capture process even if the value was already equal to 1. To start a new position capture, the value 1 must be rewritten in parameter K182.

Bit# 2 of monitoring M61 and bit# 30 of monitoring M63 are set to 1 when the conditions of the digital input allow the capture of the position. They are reset (to 0) when 1 is written in parameter K182.

13.5.1 Description of the position capture process on the digital inputs

- Set to 1 the bits of parameter K159 corresponding to the digital inputs which must be inverted.
- Set to 1 the bits of parameter K178 corresponding to the digital inputs which must be detected on a rising edge.
- Set to 1 the bits of parameter K179 corresponding to the digital inputs which must be detected on a falling edge.
- Set parameter K182 to 1 to execute a new position capture.
- The position controller monitors the digital inputs according to parameters K178, K179 and K159. When one of the conditions is met, the position controller stores the position (upi) in monitoring M12 and sets to 1 bit# 2 of monitoring M61 and bit# 30 of monitoring M63 (bits of monitorings M61 and M63 are processed every 166.67µs for the DSC2P/DSC2V and 500µs for the DSCDP, DSCDL and DSCDM).
- To start a new position capture, the last two points above-mentioned must be re-executed.

Position capture diagram on digital inputs.



13.6 Analog input / output

Only the DSC2P, DSC2V and DSCDL include an analog input (refer to the corresponding **'Hardware Manual'** for more information). No analog output is present on the position controllers. It is possible to install in the controllers (except in the DSCDM) a DSO-HIO optional board, adding up to 4 analog inputs and 4 analog outputs (refer to the **'DSO-HIO User's Manual'** for more information).

13.6.1 Analog input

13.6.1.1 Analog input on position controller

Available on	DSC2P	DSC2V	~~DSCDP~~	DSCDL	~~DSCDM~~

The **AI1** (**A**nalog **I**nput **1**) command enables the reading of the analog input (AIN). This command is an alias of monitoring **M51**.

M	Alias	Name	Comment
M51	AI1	Analog input status	Reads the voltage on the analog input (AIN)

The input voltage of the analog input must be included between +10V and –10V:

- For the DSC2P / DSC2V: +10V corresponds to -2048 increments and -10V to 2047 increments.
- For the DSCDL: +10V corresponds to -32768 increments and -10V to 32767 increments.

Here is the formula calculating the voltage from the values contained in monitoring M51:

	DSC2P / DSC2V	DSCDL
Formulas	$U\,[V] = \frac{-M51}{2048} \cdot 10$	$U\,[V] = \frac{-M51}{32768} \cdot 10$

Example:

`AI1.1` Reads the voltage on the analog input (AIN). For example, if the controller gives the value 1718 increments, use the above-mentioned formula to calculate the corresponding voltage measurable on AIN: here, 1718 increments corresponds to ~ - 8.39V for a DSC2P or DSC2V and ~ - 0.52V for a DSCDL.

13.6.1.2 Analog inputs on DSO-HIO

Available on	DSC2P	DSC2V	DSCDP	DSCDL	~~DSCDM~~

The DSO-HIO111 has 4 analog inputs (14 bits ADC). Refer to the **'DSO-HIO User's Manual'** for more information.

Remark: For the DSCDP and DSCDL, the 4 analog inputs are available for both axes.

The XAIN1, XAIN2, XAIN3, XAIN4 commands allow the user to read the XAIN1, XAIN2, XAIN3 and XAIN4 analog input of the DSO-HIO. These commands are an alias of monitorings **M56**, **M57**, **M58** and **M59** respectively.

M	Alias	Name	Comment
M56	XAIN1	XAIN1 status	Reads the value of XAIN1 analog input of the DSO-HIO
M57	XAIN2	XAIN2 status	Reads the value of XAIN2 analog input of the DSO-HIO

M	Alias	Name	Comment
M58	XAIN3	XAIN3 status	Reads the value of XAIN3 analog input of the DSO-HIO
M59	XAIN4	XAIN4 status	Reads the value of XAIN4 analog input of the DSO-HIO

The input voltage of the analog input must be included between +10V and –10V between XAIN+ and XAIN-: +10V corresponds to -8192 increments and -10V to 8191 increments.
Here is the formula calculating the voltage from the values contained in monitoring M56, M57, M58 or M59:

$$U\,[V] = \frac{-M5x}{8192} \cdot 10$$

Example:

`M56.1` Reads the voltage on the analog input (XAIN1). For example, if the controller gives the value -5284 increments, use the above-mentioned formula to calculate the corresponding voltage measurable on XAIN1: here, -5284 increments corresponds to ~ +6,45V.

13.6.2 Analog outputs

13.6.2.1 Analog outputs on DSO-HIO

Available on	DSC2P	DSC2V	DSCDP	DSCDL	~~DSCDM~~

The DSO-HIO111 and 211 have 4 analog outputs (refer to the **'DSO-HIO User's Manual'** for more information).

Remark: For the DSCDP and DSCDL, the 4 analog outputs are available for both axes.

The analog outputs can be used from two different modes:

- A 'direct' mode
- A 'source register' mode

- **Analog outputs with 'direct' mode**

The XAOUT command enables the user to allocate a defined value (defined voltage) to the XAOUT1, XAOUT2, XAOUT3 and XAOUT4 analog outputs of the DSO-HIO. This command is an alias of parameter **K173**.

K	Alias	Name	Comment
K173	XAOUT	Analog output value	Gives the value of the analog outputs (XAOUT1 to XAOUT4) of the DSO-HIO. The 4 depths of the parameter correspond to the 4 analog outputs.

Remark: For the DSCDP and DSCDL, if parameter K173 is modified, the SAV command must be executed on both axes.

XAOUT alias use the same syntax than parameter K173. Refer to §6.3 for more information about the syntax and the possible operators.

The output voltage of the analog output is included between +20V and –20V between XAOUT+ and XAOUT-: +20V corresponds to -8192 increments and -20V to 8191 increments.

Here is the formula calculating the voltage from the values contained in parameter K173:

$$U\,[V] = \frac{-K173}{8192} \cdot 20$$

Example:

`K173:2.1=-4600` ;-4600 is converted into +11.23V on XAOUT3. This can be measured between XAOUT3+ and XAOUT3- on the DSO-HIO.

Remark: If the measure is done between +XAOUT3 and GND, the voltage will be divided by two :11.23 / 2 = ~5.61V (refer to the **'DSO-HIO User's Manual'** for more information).

The digital analog converter (DAC) may induce a small offset and amplitude error. These errors can be removed with the DAO (**DA**c **O**ffset) and DAA (**DA**c **A**mplitude) commands which are an alias of parameters **K156** and **K157**.

K	Alias	Name	<p1>	Comment
K156	DAO:[depth].<axis>=<p1>	Analog output offset	-32768 to +32767	Modifies the offset of the signal sent to the analog outputs (XAOUT1 to XAOUT4). The 4 depths of the parameter correspond to the 4 analog outputs.
K157	DAA:[depth].<axis>=<p1>	Analog output amplitude	-32768 to +32767	Modifies the amplitude of the signal sent to the analog outputs (XAOUT1 to XAOUT4). The 4 depths of the parameter correspond to the 4 analog outputs.

Remark: For the DSCDP and DSCDL, if parameter K156 or K157 is modified, the SAV command must be executed on both axes.

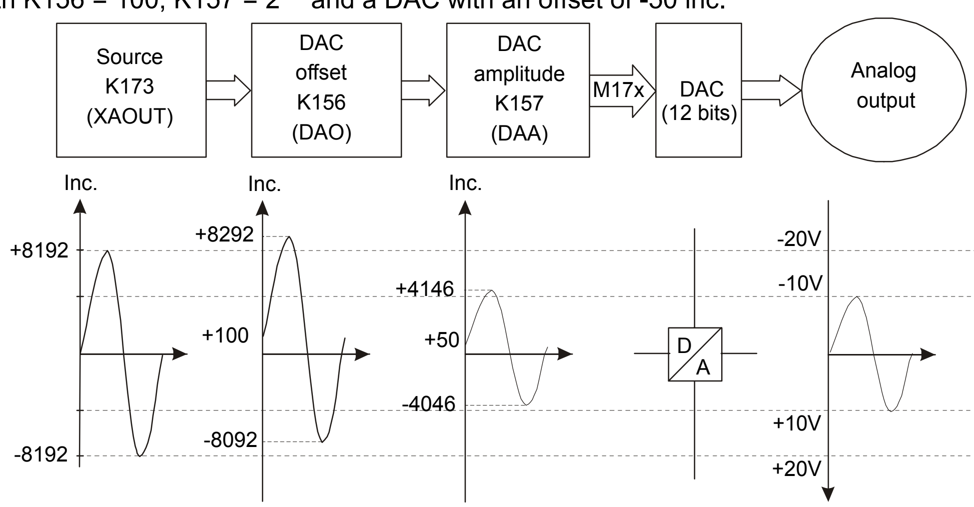
Monitorings **M173**, **M174**, **M175** and **M176** allow the user to read the value of the 4 analog outputs by taking parameters K156 and K157 into account:

M	Name	Comment
M173	Analog output1 status	Gives the value on the analog output 1
M174	Analog output2 status	Gives the value on the analog output 2
M175	Analog output3 status	Gives the value on the analog output 3
M176	Analog output4 status	Gives the value on the analog output 4

Here is the formula giving the values on the analog outputs after the amplitude and offset correction (M17x is M173, M174, M175 or M176 depending on the output number corresponding to the depth of parameter K173):

$$\mathrm{M17x} = \frac{K157}{2^{15}} \cdot (K173 + K156)$$

Example with K156 = 100, K157 = 2^{14} and a DAC with an offset of -50 inc.



- **Analog outputs with 'source register' mode**

The user can allocate to the XAOUT1, XAOUT2, XAOUT3 and XAOUT4 analog outputs of the DSO-HIO, the values of any register (X, K or M) present in the controller. To do so, the following commands (or parameters) must be used (the 4 depths of the parameters correspond to the 4 analog outputs):

K	Alias	Name	<p1>	Comment
K150	MST:[depth].<axis>=<p1>	Source register type	0 1 2 3	Disables the real-time monitoring mode. The source register is user variable X. The source register is a parameter K. The source register is a monitoring M. The 4 depths of the parameter correspond to the 4 analog outputs.
K151	MSN:[depth].<axis>=<p1>	Register index	0 - 255	Indicates the source register number. The 4 depths of the parameter correspond to the 4 analog outputs.
K158	ASE:[depth].<axis>=<p1>	Source register axis	0 1	Indicates that the source register is taken from the even-numbered axis **(ONLY available on DSCDP and DSCDL).** Indicates that the source register is taken from the odd-numbered axis **(ONLY available on DSCDP and DSCDL).** The 4 depths of the parameter correspond to the 4 analog outputs.

Remark: For the DSCDP and DSCDL, if parameter K150, K151 or K158 is modified, the SAV command must be executed on both axes.

These 3 registers allow the user to define the 'non-rectified source signal'.
It is possible to modify the offset and the amplitude of the source register signal with the MOF and MAM commands both the alias of the parameters **K154** and **K155**.

K	Alias	Name	<p1>	Comment
K154	MOF:[depth].<axis>=<p1>	Source register offset	-2^{31} to $+2^{31}-1$	Modifies the offset of the signal sent to the digital/analog converter. The 4 depths of the parameter correspond to the 4 analog outputs.
K155	MAM	Source register gain	-	Modifies the amplitude of the signal sent to the digital/analog converter. The 4 depths of the parameter correspond to the 4 analog outputs.

Remark: For the DSCDP and DSCDL, if parameter K156 or/and K157 is modified, the SAV command must be executed on both axes.

The MAM alias use the same syntax than parameter K155. Refer to §6.3 for more information about the syntax and the possible operators.

The output voltage of the analog output is included between +20V and –20V between XAOUT+ and XAOUT-: +20V corresponds to -8192 increments and -20V to 8191 increments.

Here is the formula giving the rectified source register signal value from the non-rectified value:

$$\text{Rectified source signal (=K173 =XAOUT)} = \text{(Non-rectified source signal} - \text{K154)} \cdot \frac{K155}{2^{24}}$$

From this point, the 'source register' mode is identical to the 'direct' mode described in the previous chapter.

The digital analog converter (DAC) may induce a small offset and amplitude error. These errors can be removed with the DAO (**DA**c **O**ffset) and DAA (**DA**c **A**mplitude) commands.

K	Alias	Name	<p1>	Comment
K156	DAO:[depth].<axis>=<p1>	Analog output offset	-32768 to +32767	Modifies the offset of the signal sent to the analog outputs (XAOUT1 to XAOUT4). The 4 depths of the parameter correspond to the 4 analog outputs.
K157	DAA:[depth].<axis>=<p1>	Analog output amplitude	-32768 to +32767	Modifies the amplitude of the signal sent to the analog outputs (XAOUT1 to XAOUT4). The 4 depths of the parameter correspond to the 4 analog outputs.

Monitorings M173, M174, M175 and M176 allow the user to read the value of the 4 analog outputs by taking parameters K154 to K157 into account:

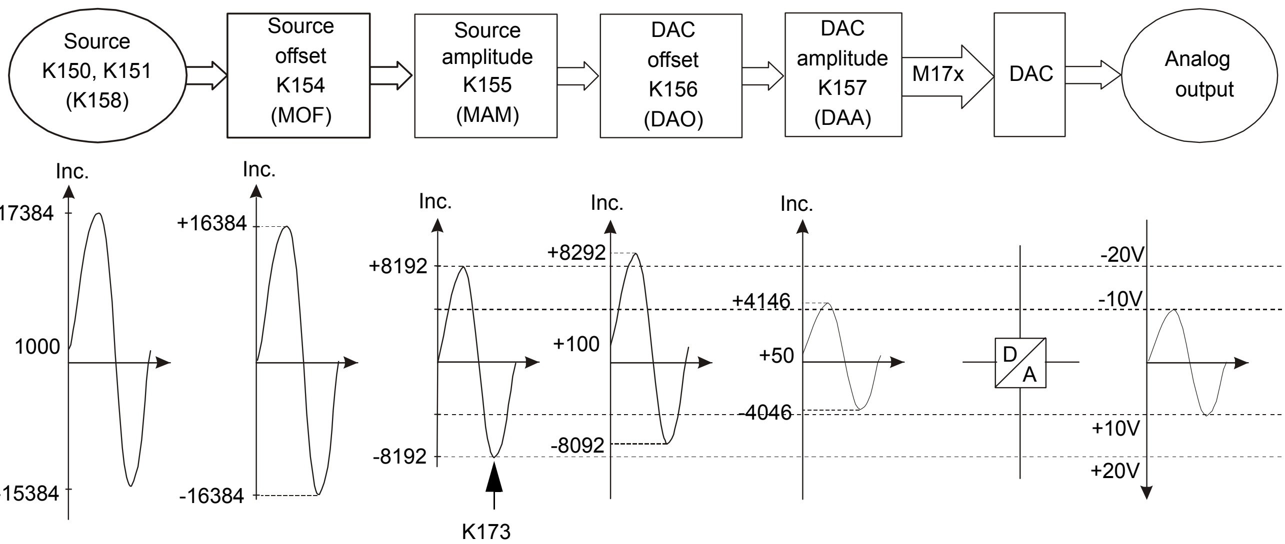
M	Name	Comment
M173	Analog output1 status	Gives the value on the analog output 1
M174	Analog output2 status	Gives the value on the analog output 2
M175	Analog output3 status	Gives the value on the analog output 3
M176	Analog output4 status	Gives the value on the analog output 4

Here is the formula giving the values on the analog outputs after the amplitude and offset correction (M17x is M173, M174, M175 or M176 depending on the output number):

$$\text{M17x} = \frac{K157}{2^{15}} \cdot (K173 + K156)$$

Remark: It is possible to invert the signal with negative values.

Example with K154 = 1000, K155 = 2^{23}, K156 = 100, K157 = 2^{14} and a DAC with an offset of -50 inc.

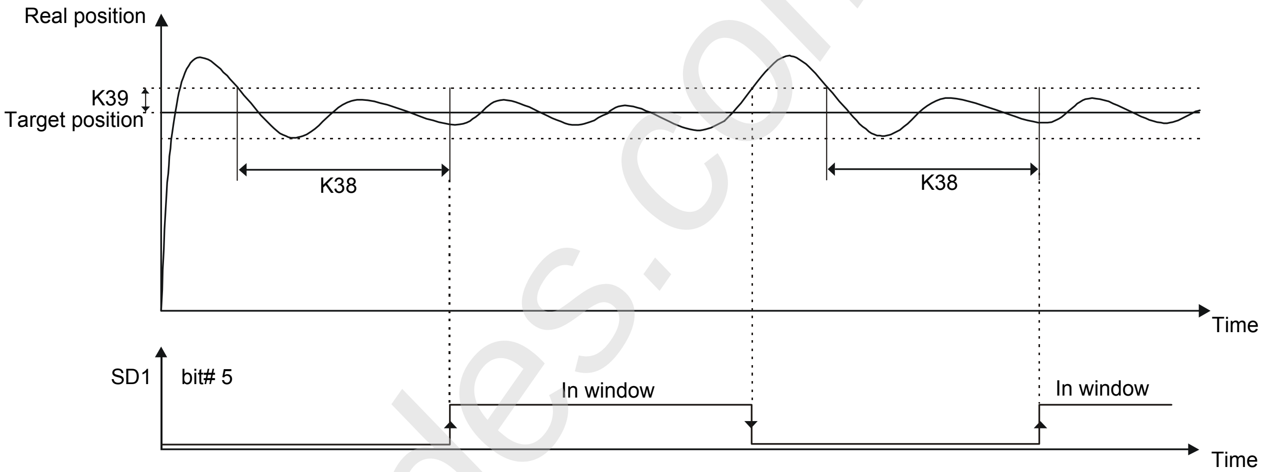


13.7 In-window

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

It is possible to define a 'window' around the target position to reach when the standard reference mode is selected (K61=1). The controller scans the real position of the motor when the position is inside the 'window' and actives a status bit SD1 (bit# 5 of monitoring M60). The user has the possibility to stop the programmed sequence until the motor arrives into the 'window' with the WTW command (refer to §14.1.1.3 for more information about the WTW command).

The motor is considered to be positioned when its real position is between [target position - **K39**] and [target position + **K39**] during a minimum time given by parameter **K38** (in [sti]) without interruption.



K	Name	Comment	Units
K38	In-window time	Minimum time for the real position to stay in the position window	[sti]
K39	In window position	Maximum (absolute) position error for the real position to stay in the position window	[upi], [rupi]

This **In-window bit** (bit# 5 of monitoring M60 (SD1)) is set to 1 when the motor reaches the position window and stays within the tolerance during the time defined by parameter K38. This bit is set to 0 during the next movement (with POS, STA, STI or WTW.<axis>=2 command).

Remark: When K38 = 0, the In-window bit (bit# 5 of monitoring M60) is set to 1 at the end of the theoretical trajectory whatever the value of parameter K39.

When K39 = 0, the In-window bit (bit# 5 of monitoring M60) is set to 1 at the end of the theoretical trajectory plus the value of parameter K38.

13.8 RTI: Real-Time Interrupts

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

A Real-Time Interrupt (RTI) allows the execution of an immediate function. The sequence execution may jump to a defined label, under some conditions. The RTI are tested each sti (each 166.67µs for the DSC2P/DSC2V or each 500µs for the DSCDP, DSCDL and DSCDM). There are different types of RTI, but each of them can be programmed only once. A maximum of 8 RTI can be programmed in the controller and they are all different from each other.

Caution: RTI functions take a lot of calculation time on the controller's micro-controllers.

RTI definition: A RTI is a set of data which defines the conditions for a function to be activated and executed at each cycle as far as they are validated (parameter K191) and enabled (parameters K190 and K192).

A RTI defines the following points:

- The conditions to test (in order to activate a RTI)
- The operations to carry out when a RTI is executed
- The label where the sequence has to jump in case of execution

Functioning principle:

The user sets the RTI he will use with the corresponding parameters. He declares this **RTI valid (1)**. From this moment onwards, the **activation conditions (2)** of the valid **RTI** are tested. This test proceeds increasingly from the lowest line number of the RTI table where they are stored (refer to §13.8.1.1). As soon as the test condition is true, the RTI is activated, independently from the fact that the test condition remains true or becomes false. If the **RTI is enabled (3)**, the **RTI is executed (4)**.

Definitions:

1) Valid: A RTI is declared valid by setting the corresponding bit in parameter **K191** (refer to §13.8.3.1). It can be declared valid only if its type, mode, label and parameters define a coherent function (refer to §13.8.1).

2) Active: A RTI is active if its activation conditions have been or are true. *The RTI automatically changes to a valid non active RTI when it is executed.*

3) Enable/disable: As the functioning of the interrupts on a micro-controller, the RTI can be 'enabled ' or 'disabled'. A RTI can become enabled (parameter K190) only when the controller is in the RTI enable mode (RIE). Furthermore, the RTI table line itself must be set as enable in parameter **K192**.

4) Executing: This action consists in executing the preset functions of a RTI when the RTI is in an active and enable mode. If the RTI label is set, the execution of the sequence goes on from the specified label. In the case of a jump to a label, the controller automatically disables the RTI (automatic execution of command RID).

Note: A **RTI** must be **valid**, **enabled** and **active** to be **executed** (refer to §13.8.4).

13.8.1 RTI structure

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The RTI is defined by a specific register: **R**

Syntax: **R** W: X .Y = Z

13.8.1.1 W: Table line number

W defines the table line number. Value (RTI priority definition): W = 0 to 7.

	Type	Label	Mode	Status	Param. 1	Param. 2	Param. 3	Param. 4
RTI0 (table line 0)	R0: 8.Y = Z	R0: 9.Y = Z	R0:10.Y = Z	R0:11.Y = Z	R0: 1.Y = Z	R0: 2.Y = Z	R0: 3.Y = Z	R0:4.Y = Z
RTI1 (table line 1)	R1: 8.Y = Z	R1: 9.Y = Z	R1:10.Y = Z	R1:11.Y = Z	R1: 1.Y = Z	R1: 2.Y = Z	R1: 3.Y = Z	R1:4.Y = Z
RTI...	...							
RTI7 (table line 7)	...							

13.8.1.2 X: Elements

X defines the RTI elements. The RTI structure is built on 8 elements:

Type (8), **Label (9)**, **Mode (10)**, **Status (11)**, **Parameter 1**, **Parameter 2**, **Parameter 3**, **Parameter 4**. Refer to §13.8.2 for more information.

	RTI Elements							
	Type	**Label**	**Mode**	**Status**	**Parameter 1**	**Parameter 2**	**Parameter 3**	**Parameter 4**
X value	0				**1**	**2**	**3**	**4**
	8	**9**	**10**	11				
Length	8 bits	8 bits	8 bits	8 bits	32 bits	32 bits	32 bits	32 bits

Elements 8 to 11 are all included in the element 0. They are redundant.

13.8.1.3 Y: Axis number

Axis number values: Y = 0 to 30.

13.8.1.4 Z: Elements values

The possible values are different for every element.

13.8.2 RTI elements description

The RTI elements **Type**, **Label**, **Mode**, **Status**, and the **Parameters** formats, are described through examples in the following paragraphs.

13.8.2.1 Element: Type

Z= 0 ;Non active
Z= 1-99 ;Standard type
Z= 100-199 ;Customer specific RTI
Z= 200-255 ;Reserved

Example: Simple clock type

RTI Type (Z):	**40**	
Activation condition	P1 > P2. P1 is incremented by 1 unit each [sti].	
Execution function	Bit in P3 modified according to P4	
Parameters:	**Parameter function**	**Parameter format**
P1	Variable to increment. Increment by 1 unit each [sti] sampling time.	**Register Description (RD)** (refer to **§13.8.2.5**)
P2	Trigger variable. Value to reach to modify the bit in P3 according to P4	**Register Description (RD)** (refer to §13.8.2.5)
P3	Bit to modify	**Bit Description (BD)** (refer to §13.8.2.5)
P4	P3 modification mode	0: no modification of P3 1: P3 is set to 1 when P1 > P2 (state) 2: P3 is set to 0 when P1 > P2 (state)

13.8.2.2 Element: Label

Z= 255 ;No jump
Z= 0-254 ;Label where to jump

Remark: Labels where to jump are coded on 8 bits; thus it is impossible to jump to labels 256 to 511.

13.8.2.3 Element: Mode

Z= 0 ;The RTI takes all the commands 'wait' (WTT, WTM, WTP, WTW,...) previously launched into account, before jumping to a label.
Z= 1 ;Clear the commands 'wait'. All the commands 'wait' previously launched are cancelled by the RTI.
Z= 2 ;Keep the commands 'wait'. All the commands 'wait' previously launched are memorized, but not executed. Then, after a REI command, the commands 'wait' previously memorized are restored and executed.

13.8.2.4 Element: Status

Access: Reading

command: `R0:11.1`

Z= 1 ;RTI is valid.
Z= 2 ;RTI is active.
Z= 4 ;RTI is enabled.
Z= 8 ;Wait for the end of commands 'wait' or come back in the sequence before the commands 'wait'.

Used to have the state of a RTI

13.8.2.5 Elements: Parameters 1, 2, 3, 4

The three available **parameter coding formats** (**BD**, **RD**, **UD**) are described below:

Bit description (BD) parameter

Byte	Weight	Comment
0	LSB	Register index (LSB)
1		Register index (MSB)
2		Type of register: 1: type X 2: type K 3: type M
3	MSB	Bit number

Register Description (RD) parameter

Byte	Weight	Comment
0	LSB	Register index (LSB)
1		Register index (MSB)
2		Type of register: 1: type X 2: type K 3: type M
3	MSB	Always programmed to 0 (zero)

Value description (UD) parameter

Byte	Weight	Comment
0	LSB	Value (LSB)
1		Value
2		Value
3	MSB	Value (MSB)

13.8.2.6 RTI elements declaration

Below is given an **example** on how the RTI elements are structured:

```
R0:8.1=40                ;Type of RTI: "Simple clock"
R0:9.1=50                ;Jumps to label 50
R0:10.1=1                ;Clears the commands 'WAIT'
R0:1.1=0x0001000B        ;X11 variable to increment.
R0:2.1=0x0001000A        ;X10 variable as a limit.
R0:3.1=0x020200AB        ;Bit 2 from digital outputs.
R0:4.1=1                 ;Digital outputs set to 1.
```

13.8.3 Controller parameters and commands for RTI

13.8.3.1 Parameters

K	Name	Bit value	Bit #	Comment
K190	Controller RTI mode	0 1	0	disable enable
K191	Valid RTI lines	1 2 ... 128	0 1 ... 7	line 0 line 1 ... line 7
K192	Enable RTI lines			
K193	Flag of active RTI lines in the RTI table			

Remark for parameters K191 to K193: When bit# 0 = 1, line 0 is activated
When bit# 1 = 1, line 1 is activated
...and so on ...

13.8.3.2 Commands

The following controller commands may be used to set the controller general mode for the RTI process:

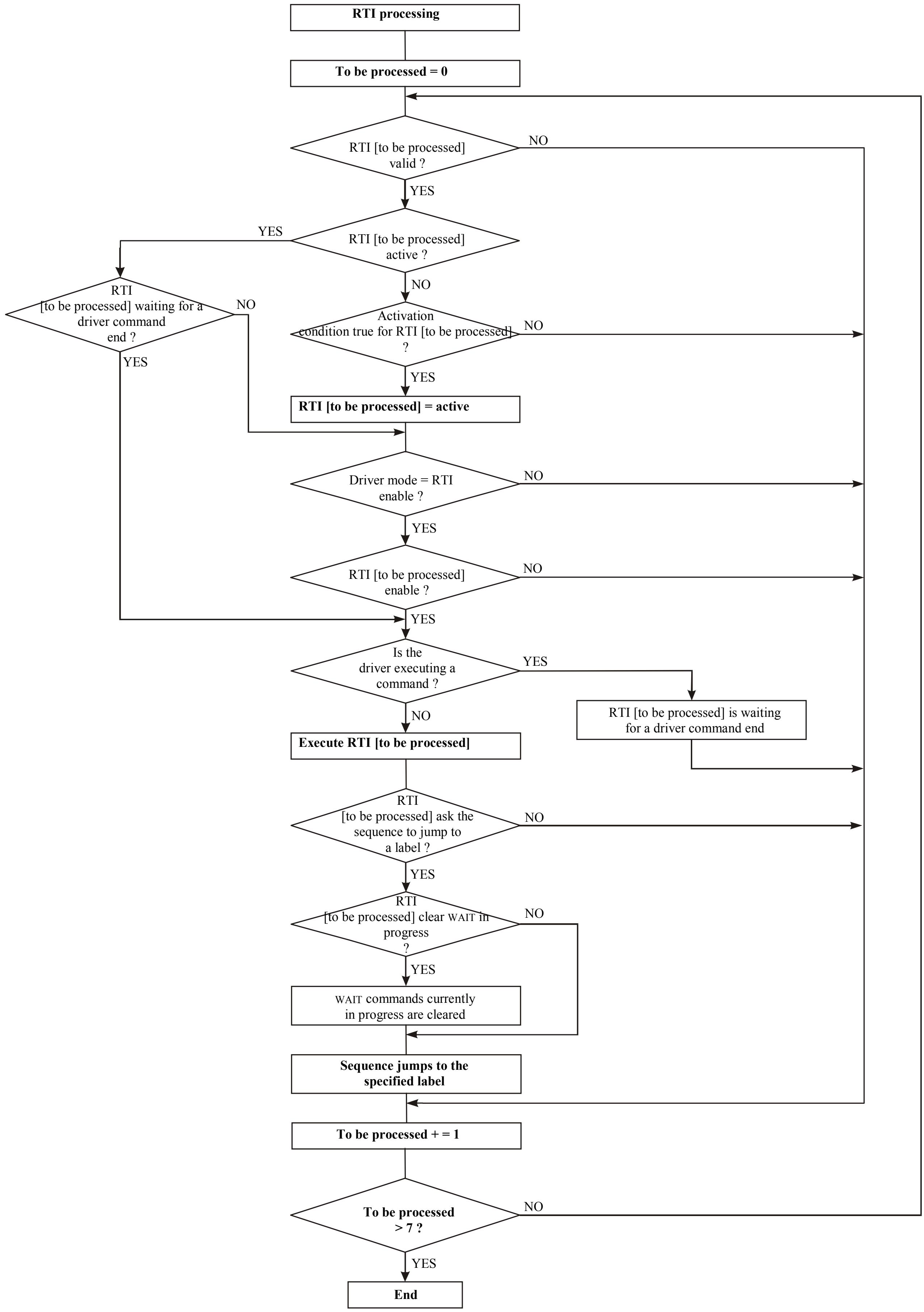
RID.<axis> ;Sets the **controller** in the **RTI disable mode**.
All valid RTI are always tested, but are not executed.

RIE.<axis> ;Sets the **controller** in the **RTI enable mode**.
If some RTI have been activated during the disable period, they are directly executed.

REI.<axis> ;Executes the RIE and RET commands within one instruction.
The sequence returns to the line which follows the last executed line before the call of the RTI label.

13.8.4 RTI process chart

The following chart describes the functioning of the RTI process at each sti (refer to §4.) sampling time.



13.8.5 RTI types

13.8.5.1 *Introduction*

All existing RTI types are detailed in the following paragraphs:

Type	Type of RTI	
Activation condition	Activation conditions to change from a non active valid status to a valid status	
Executing function	Preset functions executed by the RTI when it stands in an active and enable mode	
Parameters	**Function**	**Format**
P1	Parameter 1 description	Parameter 1 coding format
P2	Parameter 2 description	Parameter 2 coding format
P3	Parameter 3 description	Parameter 3 coding format
P4	Parameter 4 description	Parameter 4 coding format

Remark: With an 'always active' activating condition, it is not recommended to jump to a label. It does not make sense because the sequence would continuously jump to that label without executing the other lines.

13.8.5.2 *Bit copy*

Type	**2**	
Activation condition	Always active	
Executing function	Copies the register bit defined by P1 in another register bit defined by P2, according to P3 mode.	
Parameters	**Function**	**Format**
P1	Bit source	Bit description (BD)
P2	Bit destination	Bit description (BD)
P3	Mode	0: Normal 1: bit inversion
P4	Not used	Not used

13.8.5.3 *Bit test*

Type	**3**	
Activation condition	If the test of the bit P1 fulfils the condition requested in P2	
Executing function	Modification of P3 according to P4 mode	
Parameters	**Function**	**Format**
P1	Bit to test	Bit description (BD)
P2	Jumps to a label (if existing) and modifies the bit defined by P3 according to P4	0: Not test 1: Rising edge 2: Falling edge 3: Rising and falling edge 4: state 0 8: state 1
P3	Bit to modify	Bit description (BD)
P4	P3 modification mode	0: No modification of P3 1: P3 is set to 1 when jump condition is detected 2: P3 is set to 0 when jump condition is detected 3: P3 is toggled when jump condition is detected

13.8.5.4 Mask test

Type	**4**	
Activation condition	Logic operation: P2 is masked by P1 (P2 AND P1). If the result of the operation fulfils the condition requested in P3.	
Executing function	Sets the bit defined in P4 to 1	
Parameters	**Function**	**Format**
P1	Mask to apply	Value Description (UD)
P2	register to test	Register Description (RD)
P3	Jumps to a label (if existing) and modifies the bit defined by P4	0: No test 1: Rising edge 2: Falling edge 3: Rising and Falling edge 4: State 0 8: State 1 +0: at least 1 bit of the mark has changed +32: all bits of the mask have changed
P4	Bit to modify	Bit Description (BD)

13.8.5.5 Test on the value

Type	**20**		
Activation condition	Comparison of P1 (register) with P2 (immediate value) If the result of the comparison fulfils the condition requested in P3.		
Executing function	Sets the bit defined in P4 to 1		
Parameters	**Function**	**Format**	
P1	Register to test	Register Description (RD)	
P2	Value of comparison	Value Description (UD)	
P3	Jumps to a label (if existing) and modifies the bit defined by P4	0: No test 1: P1 > P2 2: P1 = P2 3: P1 >= P2 4: P1 < P2 5: P1 != P2 6: P1 <= P2 128: No test 129: P1 > P2 130: P1 = P2 131: P1 >= P2 132: P1 < P2 133: P1 != P2 134: P1 <= P2	edge edge edge edge edge edge edge state state state state state state state
P4	Bit to modify	Bit Description (BD)	

13.8.5.6 Test on the variable

Type	**21**		
Activation condition	Comparison of P1 (register) with P2 (register) according to P3 is true		
Executing function	P4 is set to 1, Allows a jump to some label.		
Parameters	**Function**	**Format**	
P1	Register to test P1	Register Description (RD)	
P2	Test register P2	Register Description (RD)	
P3	Jumps to a label (if existing) and modifies the bit defined by P4	0: No test 1: P1 > P2 2: P1 = P2 3: P1 >= P2 4: P1 < P2 5: P1 != P2 6: P1 <= P2 128: No test 129: P1 > P2 130: P1 = P2 131: P1 >= P2 132: P1 < P2 133: P1 != P2 134: P1 <= P2	edge edge edge edge edge edge edge state state state state state state state
P4	Bit to modify	Bit Description (BD)	

13.8.5.7 Window test

Type	**22**	
Activation condition	Always active	
Executing function	If P2<P1<P3 then P4=1 otherwise P4=0	
Parameters	**Function**	**Format**
P1	Register to test	Register Description (RD)
P2	Inferior window limit register	Register Description (RD)
P3	Superior window limit register	Register Description (RD)
P4	Bit to modify	Bit Description (BD)

13.8.5.8 Continuous calculation

Type	**30**	
Activation condition	Always active	
Executing function	Calculation of P3 from P1 and P2 according to the mode defined in P4	
Parameters	**Function**	**Format**
P1	Register P1	Register Description (RD)
P2	Register P2	Register Description (RD)
P3	Register P3	Register Description (RD)

P4	Type of calculation	0: No calculation 1: P3 = P1 + P2 2: P3 = P1 - P2 3: P3 = P1 * P2 4: P3 = P1 / P2 5: P3 = ~P1(not) 6: P3 = P1 & P2 7: P3 = P1 | P2 8: P3 = P1 &~ P2 9: P3 = P1 |~ P2 10: P3 = P1 >> P2 11: P3 = P1 << P2	 addition subtraction multiplication division and or 'not' and 'and' 'not' and 'or' shift to the right shift to the left

13.8.5.9 Simple Clock

Type	**40**		
Activation condition	P1 is incremented by 1 unit each [sti]. Jump to a label condition: P1>P2		
Executing function	Bit P3 is modified according to P4		
Parameters	**Function**	**Format**	
P1	Register to increment This value is incremented each [sti]	Register Description (RD)	
P2	Maximum value register P2	Register Description (RD)	
P3	Bit to modify	Bit Description (BD)	
P4	P3 modification mode	0: No modification of P3 1: P3 is set to 1 when P1>=P2 2: P3 is set to 0 when P1 3: P3 is toggled	 state state

13.8.5.10 Break control

Type	**50**	
Activation condition	always active	
Executing function	Break control especially in applications with vertical motors The output defined by P3 is changed to 1 if the controller is not in error (M60 bit 10) AND the controller has finished its phasing (M60 bit 1) AND the controller is in 'Power On' (M60 bit 0) 'operation defined by P2' Register defined by P1 The output P3 can be inverted by P4	
Parameters	**Function**	**Format**
P1	External control The control P1 is combined with P2 to modify the result P3	Bit Description (BD)
P2	Maximum value variable P2	0: P1 is not taken into account 1: P3 = !M60bit10 * M60bit1 * M60bit0 * P1 2: P3 = !M60bit10 * M60bit1 * M60bit0 + P1 3: P3 = !M60bit10 * M60bit1 * M60bit0 * !P1 4: P3 = !M60bit10 * M60bit1 * M60bit0 + !P1
P3	Bit to modify	Bit Description (BD)
P4	P3 modification mode	0: P3 not modified 1: P3 = !M60bit10 * M60bit1 * M60bit0 ? P1 2: P3 = !(!M60bit10 * M60bit1 * M60bit0 ? P1)

13.8.6 RTI programming examples

13.8.6.1 Example 1: 'Window test' type

Tests if the X1 variable is inside the limit defined by X2 and X3.
Set the DOUT1, if it is the case.

Sequence:

```
:5.1                       ;Start of the sequence.

X1.1=0                     ;Sets the X1 variable
X2.1=20                    ;Sets the X2 variable
X3.1=40                    ;Sets the X3 variable

R0:8.1=22                  ;Type of RTI.
R0:9.1=255                 ;Does not jump to a label.
R0:10.1=1                  ;Does not wait on the 'wait' command
R0:1.1=0x00010001          ;Variable to test X1
R0:2.1=0x00010002          ;Inferior limit, X2 variable
R0:3.1=0x00010003          ;Superior limit, X3 variable
R0:4.1=0x000200ab          ;Bit to modify, DOUT1 (parameter K171)

K191.1=1                   ;Valid RTI line
K192.1=1                   ;Enables the RTI line
RIE.1                      ;Sets the controller RTI enable mode
```

13.8.6.2 Example 2: 'Bit test' type

Test if the din1 comes to 1 and jump to label 200.
P3 is not included, because P4 = 0 in this case.

Sequence:

```
:5.1                       ;Start of the sequence.

R0:8.1=3                   ;Type of RTI.
R0:9.1=200                 ;Jumps to label 200.
R0:10.1=1                  ;Does not wait on the 'w..' command
R0:1.1=0x00030032          ;Bit 0 of DIN1 (M50).
R0:2.1=8                   ;State 1
R0:4.1=0                   ;No modification of P3

K191.1=1                   ;Valid RTI line
K192.1=1                   ;Enables RTI line
RIE.1                      ;Sets the controller RTI enable mode

:200.1                     ;Label 200.
…..                        ;Commands
K193.1=0                   ;Resets the flag
REI.1                      ;Back to the sequence and RTI activation
```

13.8.6.3 *Example 3: 'Simple clock' type*

- The motor is initialized and it is going to complete a series of small movements.
- A clock is switched on. It jumps to label 50 every 10 seconds.
- At label 50, DOUT 7 is set to 1.
- After some time, digital output is switched off

Sequence:

```
:79.1               ;Label 79: The following sequence starts automatically
PWR.1=1             ;Motor phasing
IND.1               ;Looking for the index

R0:8.1=40           ;Type of RTI: "Simple clock" (§13.8.2.1 and §13.8.5).
R0:9.1=50           ;Jumps to label 50 (§13.8.2.2).
R0:10.1=1           ;Does not wait for the end of the commands 'wait' (§13.8.2.3).

                    ´Simple clock array´

R0:1.1=0x0001000B   ;X11 variable to increment (hexa), (§13.8.2.5).
R0:2.1=0x0001000A   ;X10 variable as a limit (hexa), (§13.8.2.5).
R0:3.1=0x020200AB   ;Bit 2 from digital outputs (hexa), (§13.8.2.5).
R0:4.1=1            ;Digital outputs set to 1 (§13.8.2.5).

X11.1=0             ;Clock initialization
X10.1=10000         ;Clock limit initialization
K191.1=1            ;The first RTI line is valid (§13.8.3.1)
K192.1=1            ;The first RTI line is enabled (§13.8.3.1)
K193.1=0            ;Reset the flag of the activated RTI
RIE.1               ;Sets the controller in the RTI enable mode (§13.8.3.2)

:20.1               ;Label 20: Any sequence of movements
POS.1=0             ;Goes to the position 0
WTM.1               ;Waits until the movement is finished
POS.1=200000        ;Goes to the position 200000
WTM.1               ;Waits until the movement is finished
JMP.1=20            ;Jumps to label 20

:50.1               ;Label 50: Start of "Simple clock" RTI processing
X11.1=0             ;Switches the clock on again
DOUT.1=&64          ;Outputs 1 to 7 switched off
WTT.1=500           ;Waits 500ms
K171.1=0            ;Digital Output is switched off
K191.1=1            ;Every preset RTI validated (§13.8.3.1)
REI.1               ;Back to the sequence and RTI activation (§13.8.3.2).
END.1
```

13.9 Triggers

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

13.9.1 Principle

The trigger function is interesting if the system of the user has to specifically react when the theoretical trajectory (monitoring M6) reaches defined positions. This is available with the standard reference mode (K61=1). With the **triggers**, the controller may induce **two effects**, according to the theoretical trajectory:

- Modification of the state of one or more controller or optional board's (DSO-HIO) digital outputs (refer to the corresponding **'Hardware Manual'** to know the number of the digital outputs present).
- And/or modification of bits# 8-15 of 'Status Drive' **SD2** (monitoring M61) and bits# 0-15 of monitoring **M63**.

Remark: Refer to §13.11 for more information about **SD2** (monitoring M61) and monitoring **M63**.

Triggers are programmed in tabular form. Every line represents a trigger describing the action induced by the controller when a defined position [upi] is reached.

There are 3 types of triggers: **positive triggers**, **negative triggers** and **bidirectional triggers**. The first is activated only when the motor goes past the requested position with a local positive movement. The second one is activated with a negative local movement and the third one is activated at each time whatever the movement direction. Using simultaneously **192** triggers is possible. They can be subdivided into groups called 'mapping' (parameter K187 or TNB alias command).

13.9.2 Mappings definition

To program triggers, the first operation is to define the mappings with parameter **K187** (or **TNB alias command**). A mapping is a group of triggers active during a given time (active during a movement, eg.).

K	Alias	Name	<p1>	Comment
K187	TNB.<axis> = <p1>	Trigger mapping size	0-192	Number of triggers (or events) per mapping

When this operation is completed, the triggers table format is defined.

Example:

The opposite table shows the structure of a triggers table. The table is divided into **mappings**. In this example, they are 4 triggers per mapping (TNB.#=4).
The TRS command can select another mapping for each movement and therefore use a different group of triggers (refer to §13.9.6 for more information).

Position	Triggers table	
0	E_0	Mapping #0
1	E_1	
2	E_2	
3	E_3	
4	E_4	Mapping #1
5	E_5	
6	E_6	
7	E_7	
...	...	...
...	...	
189	...	Last mapping
190	...	
191	E_{191}	
192	E_{192}	

13.9.3 Triggers definition and structure

When the trigger table mapping is defined (refer to §13.9.2), the triggers themselves can be programmed. Each trigger is defined by its type, position, the induced action(s), etc...

Remark: In each mapping, the triggers are always programmed from the lowest to the highest position.

The trigger is defined by a specific parameter: **E** (for **E**vent).

Syntax: **E** W: X .Y = Z

13.9.3.1 W: Table line number

W defines the table line number. Value (trigger order): W = 0 to 191.

13.9.3.2 X: Elements

X defines the trigger elements. The trigger structure is built on 7 elements:

X=0 Res 2, 8 bits (MSB)
Action 8 bits
Res 1, 8 bits
Type 8 bits (LSB)

X=1 Parameter 1, 32 bits

X=2 Parameter 2, 32 bits

X=3 Position 32 bits

Refer to §13.9.4 for more information about the elements.

	Trigger elements						
	Res 2	Action	Res 1	Type	Parameter 1	Parameter 2	Position
X value	0				1	2	3
Length	8 bits	8 bits	8 bits	8 bits	32 bits	32 bits	32 bits

13.9.3.3 Y: Axis number

Axis number values: Y = 0 to 30.

13.9.3.4 Z: Elements values

The possible values are different for every element.

13.9.4 Elements description

Here is the description of the trigger elements: **Action**, **Type**, **Parameter 1**, **Parameter 2**, **Position**, (**Res 2**, **Res 1**).

13.9.4.1 Element: Type

Selects the trigger type

Z = 128 (0x80) **No trigger**
Z = 129 (0x81) **Positive trigger**: active on a positive movement.
Z = 130 (0x82) **Negative trigger**: active on a negative movement.
Z = 131 (0x83) **Bidirectional trigger**: active on a negative and positive movement.

Remark: It is recommended to program the element **Type** in hex. (0x81, e.g.), for a better readability.

13.9.4.2 Elements: Res 1 and Res 2

Reserved for a future use.

13.9.4.3 Element: Action

Selects the actions taken when a trigger is activated.

Z = 0	No action
Z = 1	Modify bits of **SD2** (**S**tatus **D**rive) and monitoring **M63** (TEB status) (according to parameter K184) and / or the digital outputs of the controller (according to parameter K185)
Z = 2	Modify bits of **SD2** (**S**tatus **D**rive) and monitoring **M63** (TEB status) (according to parameter K184) and / or the **DSO-HIO** digital outputs (XDOUT1 to 8) (according to parameter K183)
Z = 3	Modify bits of **SD2** (**S**tatus **D**rive) and monitoring **M63** (TEB status) (according to parameter K184) and / or the digital outputs of the controller (according to parameter K185) without changing the values of the other bits
Z = 4	Modify bits of **SD2** (**S**tatus **D**rive) and monitoring **M63** (TEB status) (according to parameter K184) and / or the **DSO-HIO** digital outputs (XDOUT1 to 8) (according to parameter K183) without changing the values of the other bits

Remark: Refer to §13.11 for more information about **SD2** (monitoring M61) and monitoring **M63**.

13.9.4.4 Element: Parameter 1

- **For action = 1 and 2**

'Parameter 1' represents a bits field defining the **digital outputs**. The bits, set to 1 in 'Parameter 1', set the corresponding digital outputs to 1 (the other outputs are set to 0).

•When **Action = 1** ➔ the bits mask defines the digital outputs of the controller (refer to the corresponding **'Hardware Manual'** to know the number of the digital outputs present)
•When **Action = 2** ➔ the bits mask defines the **DSO-HIO** digital outputs (XDOUT1 to 8)

Action = 1 (DSC2P / DSC2V digital outputs)	**DOUT #**	-				4	3	2	1
	Bit #	-				3	2	1	0
	Parameter 1 value	-				8	4	2	1
Action = 1 (DSCDP / DSCDL digital outputs)	**DOUT #**	-						2	1
	Bit #	-						1	0
	Parameter 1 value	-						2	1
Action = 1 (DSCDM digital outputs)	**DOUT #**	-					3	2	1
	Bit #	-					2	1	0
	Parameter 1 value	-					4	2	1
Action = 2 (DSO-HIO digital outputs)	**XDOUT #**	8	7	6	5	4	3	2	1
	Bit #	7	6	5	4	3	2	1	0
	Parameter 1 value	128	64	32	16	8	4	2	1

Remark: The (binary) value of 'Parameter 1' is masked by parameter K185 (if Action = 1) or by parameter K183 (if Action = 2). Refer to §13.9.5 for more information.

As there is only one DSO-HIO per DSCDP and DSCDL, both axes can change the state of the XDOUTs. It is then recommended to use the action 4 instead of 2 when using a DSO-HIO with a DSCDP and DSCDL.

- **For action = 3 and 4**

'Parameter 1' represents a bits field defining the **digital outputs**. The bits# 0 to 15, set to 1 in 'Parameter 1', correspond to the bit of the digital outputs to be set to 1 and bits# 16 to 31, set to 1 in 'Parameter 1', correspond to the bit of the digital outputs to be set to 0.

•When **Action = 3** ➔ the bits mask defines the digital outputs of the controller (refer to the corresponding **'Hardware Manual'** to know the number of the digital outputs present)

•When **Action = 4** ➔ the bits mask defines the **DSO-HIO** digital outputs (XDOUT1 to 8)

Example:

The user wants to clear DOUT3 while setting DOUT1 and 2 on a DSC2P:

Action = 3																			
	Comment	-	-	-	-	-	No change	Clear DOUT3	No change	No change	-	-	-	-	-	No change	No change	Set DOUT2	Set DOUT1
	DOUT #	Not used	-	-	-	-	4	3	2	1	Not used	-	-	-	-	4	3	2	1
	Bit #	31-24	-	-	-	-	19	18	17	16	15-8	-	-	-	-	3	2	1	0
	Parameter 1 value	0	0	0	0	0	0	262144	0	0	0	0	0	0	0	0	0	2	1

The user wants to clear XDOUT4 while setting XDOUT1 and 6 on a DSO-HIO:

Action =4																			
	Comment	-	No change	No change	No change	No change	Clear XDOUT4	No change	No change	No change	-	No change	No change	Set XDOUT6	No change	No change	No change	No change	Set XDOUT1
	XDOUT #	Not used	8	7	6	5	4	3	2	1	Not used	8	7	6	5	4	3	2	1
	Bit #	31-24	23	22	21	20	19	18	17	16	15-8	7	6	5	4	3	2	1	0
	Parameter 1 value	0	0	0	0	0	524288	0	0	0	0	0	0	32	0	0	0	0	1

Remark: The (binary) value of 'Parameter 1' is masked by parameter K185 (if Action = 3) or by parameter K183 (if Action = 4). Refer to §13.9.5 for more information.

13.9.4.5 Element: Parameter 2

- **For Action = 1 and 2**

'Parameter 2' is a bits field defining the **'Status Drive' SD2** (alias of monitoring M61) and monitoring **M63** (TEB status). The bits, set to 1 in 'Parameter 2', set the corresponding bits to 1 in SD2 and monitoring M63 (the other bits are set to 0).

Bit # of M63	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	-
Bit # of SD2	-	-	-	-	-	-	-	-	15	14	13	12	11	10	9	8	0-7
Parameter 2 value	32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	not used

Caution: The SD2 bits available are 8 to 15 but the corresponding values are programmed in 'Parameter 2' as they were 0 to 7. Example: If 'Parameter 2' = 4, bit 10 of SD2 is set to 1.
The bits# 8 to 15 of SD2 are identical to the bits# 0 to 7 of monitoring M63. Refer to §13.11 for more information.

Remark: The (binary) value of parameter 2 is masked by parameter K184. Refer to §13.9.4 for more information.

- **For Action = 3 and 4**

'Parameter 2' is a bits field defining **'Status Drive' SD2** (alias of monitoring M61) and monitoring **M63** (TEB status). The bits# 0 to 15, set to 1 in 'Parameter 2', correspond to the bits of SD2 and monitoring M63 to be set to 1 and bits# 16 to 31, set to 1 in 'Parameter 2', correspond to the bits of SD2 and monitoring M63 to be set to 0.

Example:

The user wants to clear bit# 10 of SD2 while setting bits# 8 and 9 of monitoring M63:

Action = 3	**Comment**	No change	No change	No change	No change	No change	No change	Clear bit# 10	No change	No change	No change	Set bit# 9	Set bit# 8	No change	No change	No change	No change	No change	No change	No change	No change
	Bit # of SD2	-	15	14	13	12	11	10	9	8	-	-	-	15	14	13	12	11	10	9	8
	Bit # of M63	31-24	23	22	21	20	19	18	17	16	15-10	9	8	7	6	5	4	3	2	1	0
	Parameter 2 value	0	0	0	0	0	0	262144	0	0	0	512	256	0	0	0	0	0	0	0	0

Remark: The (binary) value of parameter 2 is masked by parameter K184. Refer to §13.9.4 for more information.

13.9.5 Masks, actions selection

The values defined in 'Parameter 1' (digital outputs DOUTs or XDOUTs) and 'Parameter 2' ('Status Drive' SD2) are masked with parameters **K184** or **K185** (TMK, **T**rigger **M**as**K,** alias command). This means the actions defined in the triggers are selected (active/inactive) with those parameters. The bits set to 1 are allowed to be modified by the trigger. Refer to §13.11 for more information about 'Status Drive' (SD2) and monitoring M63.

K	Alias	Name	Bit	Comment
K183	-	Trigger mask: DSO-HIO digital outputs	0 - 7	Selection of XDOUTs (on DSO-HIO optional board) modified by the trigger
K184	-	Trigger mask: 'Status Drive' bits: monitorings M61 and M63	0 - 7 or 0 - 15	Selection of bits 8 to 15 of SD2 ('Status Drive', alias M61) modified by the trigger or Selection of bits 0 to 15 of monitoring M63 modified by the trigger
K185	TMK	Trigger mask: DSC2P / DSC2V digital outputs	0 - 3	Selection of DOUTs modified by the trigger
		Trigger mask: DSCDP / DSCDL digital outputs	0 - 1	Selection of DOUTs modified by the trigger
		Trigger mask: DSCDM digital outputs	0 - 2	Selection of DOUTs modified by the trigger

Remark: TMK alias use the same syntax than parameter K185. Refer to §6.3 for more information about the syntax and the possible operators.

For example, these masks may be used to 'protect' some DOUTs from being modified by triggers, when they are used by another function (refer to §13.4 for more information).

Remark: Parameter K184 may also be set with bits 0 - 15 of monitoring M63.

13.9.6 Mappings activation

All triggers grouped in the same mapping (refer to §13.9.2) are activated at the same time. They must be activated with the TRS command (**TR**igger **S**elected), which is an alias of parameter **K186**, otherwise the actions linked to these triggers will not occur. The TCL command (**T**rigger **CL**ear) is also used to activate/ deactivate and erase the triggers.

K	Alias	Name	<p1>	Comment
K186	TRS.<axis> = <p1>	Trigger mapping number	-1 0 - 191	No mapping activated Mapping number <p1> is activated

Command format	Comment
TCL.<axis>	All triggers tables are cleared

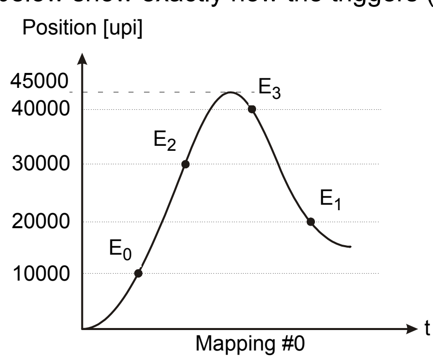
Remark: It is not possible to change the mappings during a movement.

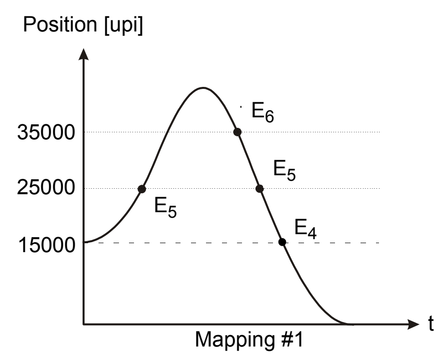
13.9.7 Programming triggers example

Let's consider 2 movements (back and forth) with a trajectory as given below:

- On the way forth (first movement, mapping #0), 4 triggers are positioned: E_0 (position 10000 [upi]), E_1 (position 20000 [upi]), E_2 (position 30000 [upi]) and E_3 (position 40000 [upi]).
- On the way back (second movement, mapping #1), 3 triggers are positioned: E_4 (position 15000 [upi]), E_5 (position 25000 [upi]) and E_6 (position 35000 [upi]).

The diagrams below show exactly how the triggers (E_x) are positioned:





The user wants to activate the digital outputs of the DSC2P (DOUTs) and the 'Status Drive' (SD2) when the motor reaches a position with a trigger, as shown below:

Mapping #0:	go past E_0	DOUT1="1"	DOUT2="0"	SD2=1
	go past E_2	DOUT1="0"	DOUT2="1"	SD2=1
	go past E_3	DOUT1="1"	DOUT2="1"	SD2=1
	go past E_1	DOUT1="0"	DOUT2="0"	SD2=0
Mapping #1:	go past E_5	DOUT3="1"		SD2=2
	go past E_6		XDOUT1="1"	SD2=3
	go past E_5	DOUT3="1"		SD2=2
	go past E_4	DOUT3="0"		SD2=0

The table below shows how to program the triggers, according to the above-mentioned list. To avoid modifying unselected outputs, a **Mask** (parameter K183 or K185) for each mapping is defined. Bits set to 1 in the mask show the selected actions.

		DSC2P outputs					DSO-HIO outputs		'Status Drive' bits		
		DOUT4 Bit 3 = 8	DOUT3 Bit 2 = 4	DOUT2 Bit 1 = 2	DOUT1 Bit 0 = 1	Dec. val. E_x:1.1	XDOUT1 Bit 0 = 1	Dec. val. E_x:1.1	SD2 Bit 9 = 2	SD2 Bit 8 = 1	Dec. val. E_x:2.1
Mapping #0	E_0	0	0	**0**	**1**	**1**	0	0	0	**1**	**1**
	E_1	0	0	**0**	**0**	**0**	0	0	0	**0**	**0**
	E_2	0	0	**1**	**0**	**2**	0	0	0	**1**	**1**
	E_3	0	0	**1**	**1**	**3**	0	0	0	**1**	**1**
	Mask	*0*	*0*	*1*	*1*	*3*	0	0	*0*	*1*	*1*
Mapping #1	E_4	0	**0**	0	0	**0**	0	0	**0**	**0**	**0**
	E_5	0	**1**	0	0	**4**	0	0	**1**	**0**	**2**
	E_6	0	0	0	0	**0**	**1**	**1**	**1**	**1**	**3**
	Mask	*0*	*1*	*0*	*0*	*4*	*1*	*1*	*1*	*1*	*3*

Remark: Masks are programmed with parameters K183, K184 and K185 (alias TMK) (refer to §13.9.5). The values selected by the Mask are in **bold** in the table above; they will be activated by the triggers. The other values are not controlled, they remain normally in their previous state.

Now, we are ready to program the triggers described above:

```
:79.1                 ;Autostart label
DOUT.1=0              ;All digital outputs of the controller are set to 0
TCL.1                 ;All triggers tables are erased
TNB.1=4               ;The table is defined with 4 triggers per mapping (alias of parameter K187)

                      ;Mapping #0 (definition triggers E0, E1, E2, E3)

E0:0.1=0x00010081     ;Positive trigger (0x81) and action 1 (0x01), controller's DOUTs & SD2
E0:1.1=1              ;Controller outputs =0001 bin. DOUT1=1 (masked by TMK), other DOUTs = 0
E0:2.1=1              ;'Status Drive' SD2 bits = 01 bin. Bit# 9=0, Bit# 8=1 (masked by parameter K184)
E0:3.1=10000          ;Trigger position = 10000 [upi]

E1:0.1=0x00010082     ;Negative trigger (0x82) and action 1 (0x01), controller's DOUTs & SD2
E1:1.1=0              ;Controller's outputs = 0000 bin. All the DOUTs = 0 (masked by TMK)
E1:2.1=0              ;'Status Drive' SD2 bits = 00 bin. Bit# 9=0, Bit# 8=0 (masked by parameter K184)
E1:3.1=20000          ;Trigger position = 20000 [upi]

E2:0.1=0x00010081     ;Positive trigger (0x81) and action 1 (0x01), controller's DOUTs & SD2
E2:1.1=2              ;Controller's outputs = 0010 bin. DOUT2=1(masked by TMK), other DOUTs = 0
E2:2.1=1              ;'Status Drive' SD2 bits = 01 bin. Bit# 9=0, Bit# 8=1 (masked by parameter K184)
E2:3.1=30000          ;Trigger position = 30000 [upi]

E3:0.1=0x00010082     ;Negative trigger (0x82) and action 1 (0x01), controller's DOUTs & SD2
E3:1.1=3              ;Controller's outputs = 0011 bin. DOUT2 & 1=1 (masked by TMK), other DOUTs
                      = 0
E3:2.1=1              ;'Status drive' SD2 bits = 01 bin. Bit# 9=0, Bit# 8=1 (masked by parameter K184)
E3:3.1=40000          ;Trigger position = 40000 [upi]
```

	;Mapping #1 (definition triggers $E_{4,}$ $E_{5,}$ E_6)
`E4:0.1=0x00010082`	;Negative trigger (0x82) and action 1 (0x01), controller's DOUTs & SD2
`E4:1.1=0`	;Controller's outputs = 0000 bin. All the DOUTs = 0 (masked by TMK)
`E4:2.1=0`	;'Status Drive' SD2 bits = 00 bin. Bit# 9=0, Bit# 8=0 (masked by parameter K184)
`E4:3.1=15000`	;Trigger position = 15000 [upi]
`E5:0.1=0x00010083`	;Bidirectional trigger (0x83) and action 1 (0x01), controller's DOUTs & SD2
`E5:1.1=4`	;Controller's outputs = 0100 bin. DOUT3=1 of DSC2P (masked by TMK), other outputs = 0.
`E5:2.1=2`	;'Status Drive' SD2 bits = 10 bin. Bit# 9=1, Bit# 8=0 (masked by parameter K184)
`E5:3.1=25000`	;Trigger position = 25000 [upi]
`E6:0.1=0x00020082`	;Negative trigger (0x82) and action 2 (0x02), DSO-HIO XDOUTs & SD2
`E6:1.1=1`	;Controller's outputs = 0001 bin. XDOUT1=1 of DSO-HIO (masked by K183)
`E6:2.1=3`	;'Status Drive' SD2 bits = 11 bin. Bit# 9=1, Bit# 8=1 (masked by parameter K184)
`E6:3.1=35000`	;Trigger position = 35000 [upi]
`PWR.1=1`	;Power on
`IND.1`	;Starts the homing process
`POS.1=0`	;Goes to position 0 [upi]
`WTM.1`	;Waits for the end of the previous movement
`:10.1`	;Label 10
`TRS.1=0`	;Mapping #0 is activated (alias of parameter K186)
`TMK.1=3`	;Controller's outputs Mask = 0011 bin. only actions linked to DOUT2 & DOUT1 are selected (TMK alias of parameter K185)
`K184.1=1`	;SD2 bits Mask = 01 bin. only action linked to Bit# 8 is selected
`POS.1=45000`	;Goes to the position 45000 [upi]
`WTM.1`	;Waits for the end of the previous movement
`POS.1=15000`	;Goes to the position 15000 [upi]
`WTM.1`	;Waits for the end of the previous movement
`TRS.1=1`	;Mapping #1 is activated (alias of parameter K186)
`TMK.1=4`	;Controller's outputs Mask = 100 bin. only action linked to DOUT3 of DSC2P is selected (TMK alias of parameter K185)
`K184.1=3`	;SD2 bits Mask = 11 bin. only actions linked to Bit# 8 & Bit# 9 are selected
`POS.1=45000`	;Goes to the position 45000 [upi]
`WTM.1`	;Waits for the end of the previous movement
`POS.1=0`	;Goes to the position 0 [upi]
`WTM.1`	;Waits for the end of the previous movement
`K172.1=0`	;All digital outputs are set to 0 (DSO-HIO)

	;(Additional part to the example, to show the usefulness of TRS; refer to §13.9.6)
`TRS.1=-1`	;The actual trigger mapping is deactivated
`POS.1=45000`	;Goes to the position 45000 [upi]
`WTM.1`	;Waits for the end of the previous movement
`POS.1=0`	;Goes to the position 0 [upi]
`WTM.1`	;Waits for the end of the previous movement

`JMP.1=10`	;Goes to the label 10

13.10 Analog encoder interpolation

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

A period is divided into four equal parts and each of them is interpolated. Parameter **K77** determines this interpolation for **analog encoders** (1Vptp and EnDat 2.1). As the encoder signals are coded on 13 bits (16 bits for the DSCDL), it is possible to choose an interpolation included between 8 and 12 bits (8 and 15 bits for the DCSDL) corresponding to an interpolation factor between 0 and 4 (0 and 7 for the DSCDL).

Example:

We have an analog encoder with a period equal to 40 μm. If the maximum interpolation factor of the sine and the cosine (K77=4 for all controllers and K77=7 for the DSCDL) is used, there are $4{\cdot}2^{12}$ = 16384 ($4{\cdot}2^{15}$ = 131072 for the DSCDL) increments on one period of encoder signals. The encoder resolution is:

	DSC2P / DSC2V / DSCDP / DSCDM	DSCDL
Formulas	$\frac{40 \cdot 10^{-6}}{16384} = 2.44 \cdot 10^{-9}\,\mathrm{m}$	$\frac{40 \cdot 10^{-6}}{131072} = 0.3 \cdot 10^{-9}\,\mathrm{m}$

Parameter K77 directly influences the motor position, speed, acceleration and jerk values (refer to §12.10.3), It also directly influences units scales (refer to §13.3.10). Refer to §12.3 for more information about the position encoder.

K	Name	Controller	Value	Comment
K77	Encoder Interpolation shift value for analog encoder (1Vptp and EnDat 2.1)	DSC2P DSC2V DSCDP DSCDM	0 1 2 3 4	Interpolation of signals on 8 bits Interpolation of signals on 9 bits Interpolation of signals on 10 bits Interpolation of signals on 11 bits Interpolation of signals on 12 bits
			5 6 7 ... 12	Signals shift of 1 bit Signals shift of 2 bits Signals shift of 3 bits Signals shift of ... bits Signals shift of 8 bits
		DSCDL	0 1 2 3 4 5 6 7	Interpolation of signals on 8 bits Interpolation of signals on 9 bits Interpolation of signals on 10 bits Interpolation of signals on 11 bits Interpolation of signals on 12 bits Interpolation of signals on 13 bits Interpolation of signals on 14 bits Interpolation of signals on 15 bits
			8 ... 12	Signals shift of 1 bits Signals shift of ... bits Signals shift of 5 bits

Parameter **K69** has a similar effect as parameter K77, but for **digital TTL encoders**.

K	Name	Value	Comment
K69	Encoder shift value for TTL encoder	0 1 2 3 4 ...	Signals shift of 4 bits (= 16 [dpi] min. resolution) Signals shift of 5 bits (= 32 [dpi] min. resolution) Signals shift of 6 bits (= 64 [dpi] min. resolution) Signals shift of 7 bits (= 128 [dpi] min. resolution) Signals shift of 8 bits (= 256 [dpi] min. resolution) (values above 4 are normally not used)

Remark: With an analog encoder (1 Vptp and EnDat 2.1), the minimum position reading resolution is 1 [dpi]. With a TTL encoder (no interpolation), the minimum position reading resolution is 16 [dpi], corresponding to the 4 bits (= 16) default shift value. Refer also to formulas in §12.3, to see the effect of this factor 16 value.

13.11 Status Drive

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The ’Status Drive’ are monitoring variables which enable the user to see if the controller is in error or warning mode. Moreover, for an i^2t error type, the integral value can be directly displayed via monitoring M67.

M	Alias	Name	Comment
M60	SD1	Status drive 1	Status drive (phasing, error, ...).
M61	SD2	Status drive 2	Status drive (process error label, ...).
M63	-	TEB status	TEB status

Monitorings **M60 (SD1)**, **M61 (SD2)** and **M63** enable the user to know if the initialization is over, if the motor is moving, if the controller has detected an error... It is necessary to convert the value shown by monitorings M60, M61 and M63 in binary. The three tables below show the meaning of each bit of monitorings M60, M61 and M63. They are used for really advanced programming. Detailed explanations are not given in this manual.

13.11.1 M60 monitoring (SD1)

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

When a bit=1, its definition is validated (the status description is verified). If a bit=0, its definition is not validated. Example: Bit 0 = 1 means 'controller switched on' and bit 0 = 0 means 'controller switched off'.

Bit #	Value	Bit definition (description when bit=1)
0	**1**	Controller in ’Power On’ mode
1	**2**	Phasing completed (first PWR.<axis> = 1)
2	**4**	First homing process completed
3	**8**	Motor (axis) present. Bit normally always set to 1.
4	**16**	Motor is executing a trajectory (moving)
5	**32**	Motor in the position/time window (In-window defined by parameters K38 and K39)
6	**64**	Controller in μ-master mode. This mode is set the controller is switched on when K170=1
7	**128**	Controller in waiting mode (commands WTM, WTP, WTW, WTB)
8	**256**	Controller executing an internal sequence (sequence in the controller)
9	**512**	Controller in edition of sequence mode (EDI command). The writing of S sequences is allowed.
10	**1024**	Controller in error mode
11	**2048**	Trace busy flag is set during a register trace acquisition
13	**8192**	Homing process in progress
23	**8388608**	Controller global warning

The EXI command clears the bit# 9 of the SD1. So, it disables the editor mode of the sequence and builds label table.

Example:

`M60.1` ;The ’Status Drive’ SD1 value is 2F; it means 101111 binary. It means that the power is on in the motor, the phasing and homing have been completed and the position is kept 'in the window' (bits 0, 1, 2, 3 and 5 are set to 1).

bits #	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Bin.	1	0	1	1	1	1
Hexa.	2		F			
Dec.	47					

13.11.2 M61 monitoring (SD2)

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Bit	Value	Bit definition	Signification when bit = 0	When bit = 1
0	**1**	Sequence error label pending (label: 80)	Error handling routine being processed: If a controller error appears, the sequence jumps to :80, to process the error handling routine. Then, the bit will be set to 1 to avoid re-entering in the error routine.	Not possible to execute the error handling routine (under label : 80).
2	**4**	Position captured on digital input	This bit is reset when 1 is written in parameter K182.	This bit is set when the conditions of the digital input allow the capture of the position.
		Bit definition (description when bit=1)		
8	**256**	User bit 0, could be modified by trigger functions or by parameter K177		
9	**512**	User bit 1, could be modified by trigger functions or by parameter K177		
10	**1024**	User bit 2, could be modified by trigger functions or by parameter K177		
11	**2048**	User bit 3, could be modified by trigger functions or by parameter K177		
12	**4096**	User bit 4, could be modified by trigger functions or by parameter K177		
13	**8192**	User bit 5, could be modified by trigger functions or by parameter K177		
14	**16384**	User bit 6, could be modified by trigger functions or by parameter K177		
15	**32768**	User bit 7, could be modified by trigger functions or by parameter K177		

13.11.3 M63 monitoring

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

Bit #	Value	Bit definition (description when bit=1)	Corresponding to SD1 or SD2 bits
0	**1**	User bit 0, could be modified by trigger functions or by parameter K177	SD2, bit# 8
1	**2**	User bit 1, could be modified by trigger functions or by parameter K177	SD2, bit# 9
2	**4**	User bit 2, could be modified by trigger functions or by parameter K177	SD2, bit# 10
3	**8**	User bit 3, could be modified by trigger functions or by parameter K177	SD2, bit# 11
4	**16**	User bit 4, could be modified by trigger functions or by parameter K177	SD2, bit# 12
5	**32**	User bit 5, could be modified by trigger functions or by parameter K177	SD2, bit# 13
6	**64**	User bit 6, could be modified by trigger functions or by parameter K177	SD2, bit# 14
7	**128**	User bit 7, could be modified by trigger functions or by parameter K177	SD2, bit# 15
8	**256**	User bit 8, could be modified by trigger functions or by parameter K177	-
9	**512**	User bit 9, could be modified by trigger functions or by parameter K177	-
10	**1024**	User bit 10, could be modified by trigger functions or by parameter K177	-
11	**2048**	User bit 11, could be modified by trigger functions or by parameter K177	-
12	**4096**	User bit 12, could be modified by trigger functions or by parameter K177	-

Bit #	Value	Bit definition (description when bit=1)	Corresponding to SD1 or SD2 bits
13	**8192**	User bit 13, could be modified by trigger functions or by parameter K177	-
14	**16384**	User bit 14, could be modified by trigger functions or by parameter K177	-
15	**32768**	User bit 15, could be modified by trigger functions or by parameter K177	-
16	**65536**	The controller is in 'Power On'	SD1, bit# 0
19	**524288**	Motor (axis) present. Bit normally always set to 1.	SD1, bit# 3
20	**1048576**	Motor is executing a trajectory (moving)	SD1, bit# 4
21	**2097152**	Motor in the position/time window (In-window defined by parameters K38 and K39)	SD1, bit# 5
23	**8388608**	Controller global warning	SD1, bit# 23
24	**16777216**	Controller executing an internal sequence (sequence in the controller)	SD1, bit# 8
26	**67108864**	Controller in error mode	SD1, bit# 10
27	**134217728**	Trace busy flag is set during a register trace acquisition	SD1, bit# 11
30	**1073741824**	Position captured on digital input (see K182/K178/K179) this bit is set when the conditions of the digital input allows the capture of the position; it is reset when 1 is written in parameter K182	SD2, bit# 2

13.12 Advanced communication

13.12.1 Synchronization

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

The synchronization is given by the TEB *synchro* signal present on the Turbo-ETEL-Bus (TEB). This signal allows several daisy-chained controllers to be synchronized to the same reference.

- For the DSC2P/DSC2V: the TEB *synchro* signal frequency is equal to 1/sti (1/sti = 1/166,67μs = 6 kHz).
- For the DSCDP, DSCDL and DSCDM: the TEB *synchro* signal frequency is equal to 6kHz but the DSCDP, DSCDL and DSCDM works at 1/sti (1/sti = 1/500μs = 2 kHz).

Remark: The *synchro* signal can also come from the Macro bus if a Macro optional board is installed on the controller. In this case, the working frequency of the controller is given by parameter K88. In Macro mode, parameter K87 does not work like described below (refer to the **'DSO-MAC User's Manual'** for more information).

The synchronization is necessary when the controller is in slave mode, receiving its reference from the master (DSMAX, DSTEB or μ-master) through the TEB. It is applied to:

- PWM (simultaneous commutation of power IGBTs)
- All the process of the controllers: reading of position reference coming from the DSMAX (interpolated mode)

The controller synchronization mode is set by parameter **K87**. It defines if the controller should be synchronized, and how.

K	Name	Bit # <p1>	Comment
K87	Controller synchronization	0 - 7	Synchronization frequency = (1/sti) / <p1>
		8 - 11	Synchronization delay

Bits# 0 to 7 define the synchronization frequency of the controller:

Controller	TEB		
	K87 value	Comment	
DSC2P DSC2V	0	The controller is not synchronized	**The optimal value for the DSC2P and DSC2V is K87 = 1**
	1	The controller is synchronized at 6 kHz	
	2	The controller is synchronized at 3 kHz	
	3	The controller is synchronized at 2 kHz	
	...		
DSCDP DSCDL DSCDM	0	The controller is not synchronized	**The optimal value for the DSCDP, DSCDL and DSCDM is K87 = 257**
	1	The controller is synchronized at 2 kHz	
	2	The controller is synchronized at 1 kHz	
	3	The controller is synchronized at 500 Hz	
	...		

Remark: With a TEB communication, parameter K87 must be equal to 0 or 1 for the bits ranging from 0 to 7. For the DSCDP, DSCDL and DSCDM, if parameter K87 is modified, the SAV command must be executed on the both axes.

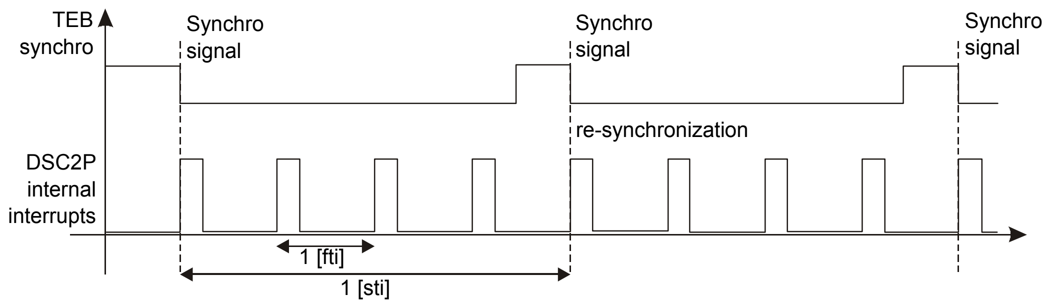
Bits# 8 to 11 define the delay of synchronization of the controller:

- If bits# 8&9 = 0: the synchronization is not delayed
- If bit# 8 = 1: the synchronization is delayed of 1 [fti]
- If bit# 9 = 1: the synchronization is delayed of 2 [fti]
- If bits# 8&9 = 1: the synchronization is delayed of 3 [fti]
- ...

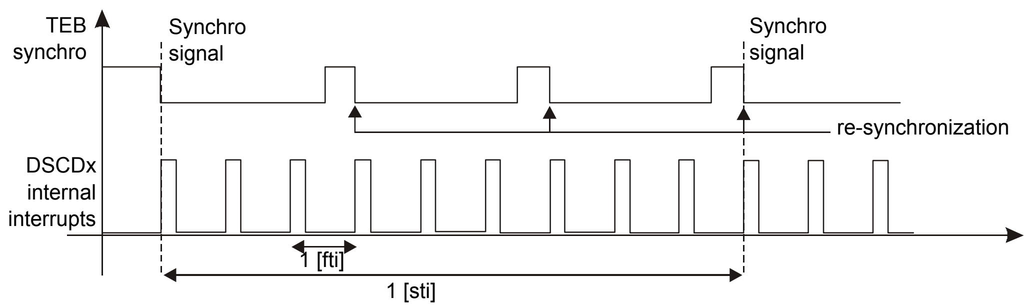
Remark: 1 [fti] = 41.67 µs for the DSC2P/DSC2V and 1 [fti] = 55.56 µs for the DSCDP, DSCDL and DSCDP.

Synchronization examples:

Example at 6 kHz with a TEB communication between DSC2Ps and/or DSC2Vs:



Example at 2 kHz with a TEB communication between DSCDPs or/and DSCDLs or/and DSCDMs:



13.12.2 Real-time monitoring (RTM)

13.12.2.1 Slave to slave

Available on	DSC2P	DSC2V	~~DSCDP~~	~~DSCDL~~	~~DSCDM~~

It is possible to send two registers from a slave to another slave through a real-time monitoring channel. To do so, a DSMAX or a DSTEB must be present as a master of the communication ring and the following parameters set (this function is then not available with a DSC2P or DSC2V in µ-master mode). There can be a maximum of 4 slaves sending 2 registers to other slaves through 4 real-time monitoring channels.

Warning: A slave must not receive registers trough the real-time monitoring channel from more than one slave.

To activate the desired RTM mode, parameter **K102** must be correctly set:

K	Name	Value	Bit #	Comment
K102	RTM channel mode	2	1	Enables the slave to slave communication mode

To enable the slave to slave mode, parameter K102 must be equal to 2 (bit 1 set) or 3 (bits 1 & 2 set).

Parameters **K231**, **K232** and **K233** are used to choose one or two registers to be sent.

K	Name	Value	Comment
K231	RTM register type	1 2 3	The register sent is a user's variable X The register sent is a parameter K The register sent is a monitoring M Parameter K231 has 2 depths: depth 0 for register 1 and depth 1 for register 2
K232	RTM register index	-	Number of the register to be sent. Parameter K232 has 2 depths: depth 0 for register 1 and depth 1 for register 2
K233	RTM destination axis	-	Mask of the destination axes

Remark: Parameters K231, K232 and K233 must be set before enabling the RTM mode (K102 = 2).

Monitoring **M230** and **M231** allows the user to read the two registers sent to the slave(s).

M	Name	Comment
M230	Register 1 value	Displays the value of the first register sent
M231	Register 2 value	Displays the value of the second register sent

Example:

X0 and M7 of the axis 0 are sent to the axes 2 and 4.

```
K233.0=0x14      ;0x14 = (10100)bin. Axes 2 and 4 are selected as the destination axes
K231:0.0=3       ;Register 1 is a monitoring M
K232:0.0=7       ;Register 1 is monitoring M7
K231:1.0=1       ;Register 2 is a user's variable X
K232:1.0=0       ;Register 2 is the X0 user's variable
K102.0=2         ;Slave to slave mode is enabled
```

Remark: **The TEB synchronization must be activated!** If it is not the case, the real-time values read by the slave could be not updated at each cycle! Then, parameter K87 must be equal to 1 for a DSC2P or DSC2V.

13.12.2.2 Slave to master

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

It is possible to send 2 registers from a slave to the master through a real-time monitoring channel. To do so, a DSMAX must be present as the master of the communication ring and the following parameters set. Maximum 16 slaves can send a register to the master and in that case the axes number must range from 0 to 15. Refer to the **'DSMAX User's Manual'** to have more information about how to collect them.

To activate the desired RTM mode, parameter K102 must be correctly set:

K	Name	Value	Bit #	Comment
K102	RTM channel mode	1	0	Disables the slave to master communication mode

To enable the slave to master mode, bit 0 of parameter K102 must be equal to 0 (bit 0 reset).

Parameter **K219** is used to choose two registers to be sent: the first depth corresponds the first register and the second depth corresponds to the second register.

K	Name	Byte #	Comment
K219	RTM register	0 1	Number of the register to be sent
		2	Depth of the register (only available for the parameters K)
		3	1 = the register sent is a user's variable X 2 = the register sent is a parameter K 3 = the register sent is a monitoring M

Remark: **The TEB synchronization must be activated!** If it is not the case, the real-time values read by the DSMAX could be not updated at each cycle! Then, parameter K87 must be equal to 1 for a DSC2P/DSC2V and to 257 for a DSCDP, DSCDL and DSCDM.

When K219 = 0, the controller does not send any value to the RTM channels.

Example:

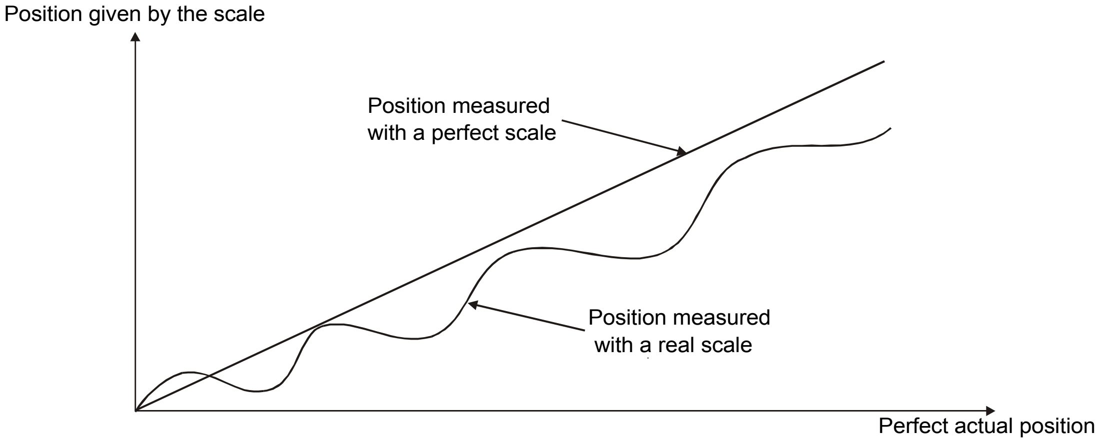
The measured position (monitoring M7) and the theoretical force (monitoring M30) of the axis 5 are sent to the master.

```
K219:0.5=0x03000007    ;Register 1 is monitoring M7 (03 = monitoring M; 07 = monitoring number)
K219:1.5=0x0300001E    ;Register 2 is monitoring M30 (03 = monitoring M; 1Eh= 30 = monitoring number)
```

13.13 Encoder scaling and mapping

This paragraph describes how to correct the position given by the encoder along an axis.

The linear and rotary encoders have a measurement error with regard to the actual position of the axis. This error comes in the form of a linear error according to the position (a μm of the encoder is not necessarily a standard μm) on which is added a random error. The following drawing gives a good idea of the error:



In the application requiring a very high absolute precision on the movement, it may be necessary to rectify the wanted position according to the known errors of the scale. These corrections will cancel the errors of the measuring system and will position the moving load with a better absolute precision.

To sum up, we will use the term of **encoder mapping** to talk about the compensation of the random error along the scale and the term of **encoder scaling** to talk about the linear correction of the scale. The second function allows the user to proportionately influence the movement references.

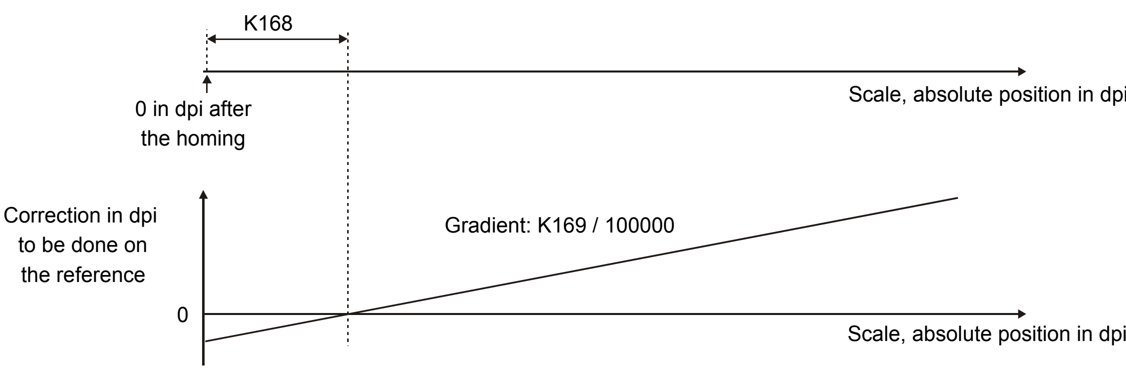
Whether it is the trajectory generator or the DSMAX which gives the position set point, the data firstly goes through the jerk filter defined by parameter K213 and then through the encoder scaling and mapping correction.

Remark: Both corrections are given with regards to the absolute position on the scale and then, they will be activated only after the homing!

The encoder scaling is not available in the rotary modes (MMD=17 or 19 or 24 or 26). If the user wants to use this correction with a rotary motor, the mode MMD = 1 must be used but in that case the movement is limited to $-2*10^9$ to $+2*10^9$ [udi].

13.13.1 Encoder scaling

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM



Parameter **K168** determines the point where the correction is null. The principle is to add a correction in dpi on the reference position. The correction is calculated as follows:

$$\text{Correction} = (\text{position} - K168) * K169 / 100000$$

The sensitivity for a linear correction is then one thousandth of per cent. Parameter K169 can be positive or negative: from - 50000 to + 50000. If K169=0, there is no correction.

Remark: If (position - K168) exceeds +/- 2^{31}, the encoder scaling correction will be discontinuous. It means that the user can move around the position given by parameter K168 from -2^{31} to $+2^{31}$ dpi.

Caution: Infinite rotary movement cannot be used!

13.13.2 Encoder mapping

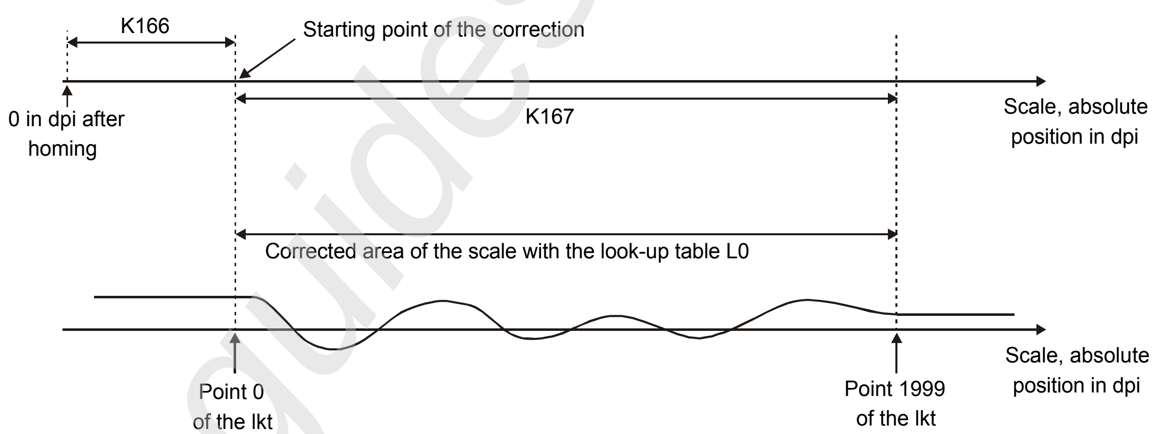
Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

A look-up table (L0) of correction with 2000 points can be programmed and saved in the controller. This look-up table contains the correction values in dpi to be applied on the reference position of the axis.

Parameter **K166** defines the position of the scale from where the correction, defined by the point 0 of the look-up table, will be applied. This parameter enables the setting of the correction table on the scale.

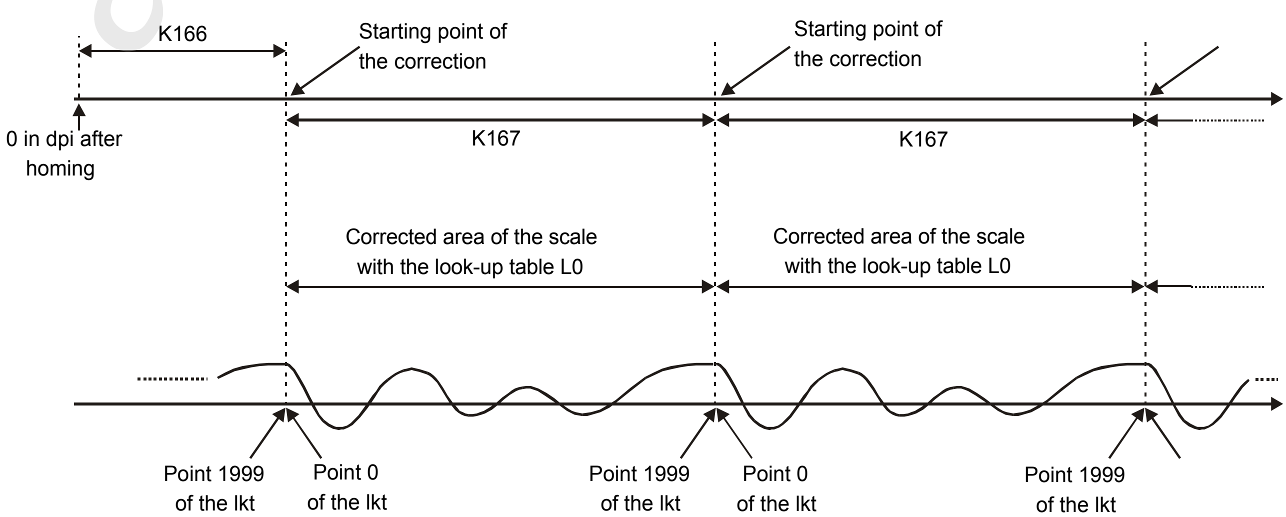
Parameter **K167** defines the length in dpi on which the look-up table corrects the set point. The value of parameter K167 is from 0 to 2^{31}-1. This correction is subtracted from the set point reference (theoretical position).

- Linear encoder mapping



- Rotary encoder mapping

To apply cyclic corrections on one motor's turn, parameter **K167** must be set to include the distance in [dpi] corresponding to exactly one turn. Thus, when a negative or positive distance given by parameter **K167** has been covered (from parameter K166), the same look-up table is applied again and again as long as the motor is moving. To activate the rotary encoder mapping, parameter K165 must be equal to 4.



13.13.3 Activation of the corrections

Available on	DSC2P	DSC2V	DSCDP	DSCDL	DSCDM

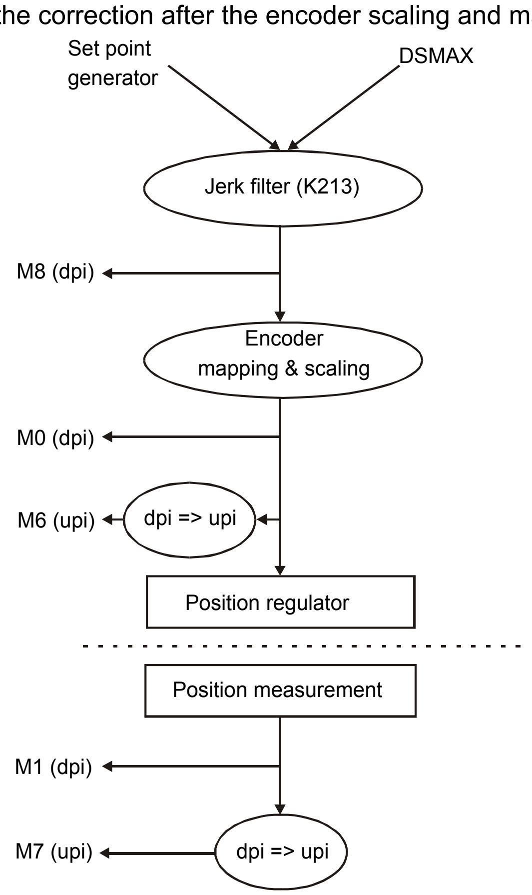
Parameter **K165** allows the user to activate these corrections:

K165	Encoder scaling correction	Standard encoder mapping correction	Rotary encoder mapping
0	No	No	No
1	No	Yes	No
2	Yes	No	No
3	Yes	Yes	No
4	No	No	Yes

By default, there is no correction because K165 = 0. If K165 is equal to 1, 2, 3 or 4, the correction will be activated only after the homing. The setting of parameters K165 to K169 will be done only after the end of a movement.

13.13.4 Use

In order to monitor the effect of the correction on the scope, a new monitoring variable has been added: monitoring M8. It represents the position reference in dpi before the encoder scaling and mapping correction. Monitoring M0 represents the correction after the encoder scaling and mapping.



Remark: Once the corrections of the position are activated, there are also valid in interpolated mode with the DSMAX (as shown above).
As the corrections are done with regard to the absolute position in dpi, the SET command which changes the offset for the variable in upi can be used without affecting the correction.
Once the controller is powered on, K168.0=M8.0 can be executed to automatically set a null correction at the current position. K169 will be modified by the user to change the correction of the encoder scaling.

Chapter D: Programming

14. Basic programming

14.1 Commands

The commands allow the communication with the controller, giving it orders, etc. They can be sent one after the other and immediately executed by the controller: this is the **on-line** mode. A sequence of commands can also be memorized in the controller, and in this case it is called a **user sequence** (refer to §14.4).

14.1.1 Wait commands

These commands temporarily stop the execution of a sequence (normal commands) during all the waiting time and then continue from the following line.

14.1.1.1 WTT command

The **WTT** command (**wa**it a **t**ime) makes a pause in the sequence progression. The pause duration is set with the command parameter.

Example:

The Slow Time Interrupt (sti) is equal to 166.67µs for the DSC2P/DSC2V and to 500µs for the DSCDP, DSCDL and DSCDM.

`WTT.1=500` ;The pause duration is equal to 500×166.67µs = 83ms for the DSC2P and DSC2V. The pause duration is equal to 500×500µs = 250ms for the DSCDP, DSCDL and DSCDM.

`WTT.1=20` ;The pause duration is equal to 20×166.67µs = 3,32ms for the DSC2P and DSC2V. The pause duration is equal to 20×500µs = 10ms for the DSCDP, DSCDL and DSCDM.

Command	<P1>	Comment	Units
WTT.<axis> = <P1>	Pause duration	Pause of <P1> * sti (166.67µs or 500µs) in the process of the sequence.	[sti]

14.1.1.2 WTM and WTP commands

When the **WTM** command (**W**ai**T** for end of **M**ovement) is executed, the controller waits for the end of the current movement before going on with the sequence execution. If several movement commands (POS, STA, etc.) are successively sent it is not necessary to introduce the WTM command between each movement to wait for the end of the preceding one because the controller does it automatically, unless **concatenated movements** are executed (refer to §13.3.1.1).

The **WTP** command (**W**ai**T** for **P**osition) allows the user to wait for the motor to cross the position specified in its parameter, in any direction, before going on with the sequence execution from the following line.

Remark: With WTP, if the specified position has not been crossed yet, and the movement ends, the command following WTP is executed at the end of the movement.

If the motor is not moving when WTM or WTP commands are sent, they are ignored. Both commands are generally preceded by the POS or STA command because the motor must be moving.

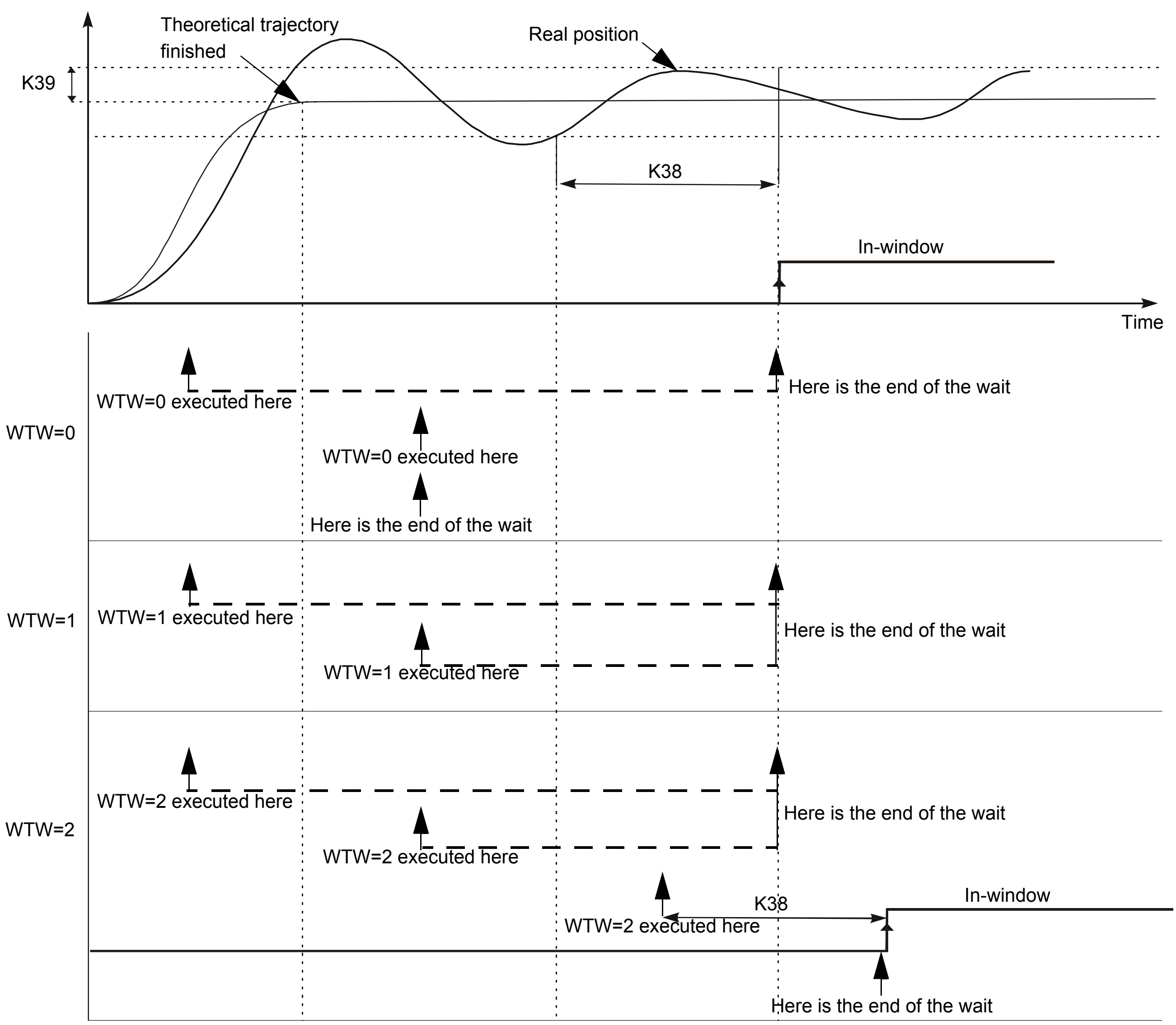
Command	<P1>	Comment	Units
WTM.<axis>	-	Waits for the end of the current movement before continuing the execution of the sequence.	-
WTP.<axis> = <P1>	Motor position	Waits for the motor to cross position <P1> before going on with the execution of the sequence.	[upi] [rupi]

14.1.1.3 WTW command

The **WTW** command (**W**ai**T** for **W**indow) allows the user to define a 'window' around the target position to reach (refer to §13.7 for more information about the definition of the window) when the standard reference mode is selected (K61=1). The WTW command stops the execution of the sequence on the used axis until the motor arrive inside the window.

Command format	<P1>	Comment
WTW.<axis> = <P1>	0	Waits for the bit 'in-window' (bit# 5 of SD1) to be at 1. If the theoretical trajectory is finished (bit# 4 of SD1 at 0) when this command is executed, the controller acknowledges the command without testing if the motor is in the window or not. WTW.<axis>=0 command must not be preceded by a WTM.<axis> command because the WTW command must be executed before reaching the theoretical position.
	1	Waits for the bit 'in-window' (bit# 5 of SD1) to be at 1 without taking into account if the theoretical trajectory is finished or not.
	2	Restarts a new test by clearing the bit 'in-window' (bit# 5 of SD1) without taking into account if the theoretical trajectory is finished or not.

Here is a graphical representation of the above-mentioned explanations:



14.1.2 Wait on bits: WBS and WBC commands

WBS (**W**ait **B**it **S**et) and **WBC** (**W**ait **B**it **C**lear) commands test one or several bits of X, K or M registers and go on with the sequence execution if the bits are set to 1 (WBS) or to 0 (WBC). It is reminded that bits are numbered from the right to the left from 0 to 31.

Command	<P1>	<P2>	Comment
WBS.<axis> = <P1>, <P2>	Register to test	Mask over the bits	Waits for the bits selected in the mask to be set to 1.
WBC.<axis> = <P1>, <P2>	Register to test	Mask over the bits	Waits for the bits selected in the mask to be cleared to 0.

The field <P2> must contain the **mask** that selects the bits to be tested in the register included in the field <P1>. This mask is obtained by transforming in binary the value contained in <P2>. The bits with 1 are those tested by WBS and WBC.

Here is an example with <P1> corresponding to monitoring M50 (value of the digital inputs). If <P2>=2 then only bit 1 is tested. If <P2>=513 bit 0 and bit 9 are tested. If <P2>=773 bits 1, 3, 8 and 9 are tested:

<P2> values		DIN10	DIN9	-	DIN7	DIN6	DIN5	DIN4	DIN3	DIN2	DIN1
		Bit# 9	Bit# 8	-	Bit# 6	Bit# 5	Bit# 4	Bit# 3	Bit# 2	Bit# 1	Bit# 0
2	**2**	0	0	-	0	0	0	0	0	1	0
513	**200**	1	0	-	0	0	0	0	0	0	1
773	**305**	1	1	-	0	0	0	0	1	0	1
Decimal	**Hexa.**	**Binary**									

Remark: Refer to §13.4 to know the digital inputs available for each controller.

Example:

WBS and WBC instructions are particularly useful to test the state of one or several digital inputs. For example, in order to continue the execution of a sequence only if the digital input DIN2 is activated, the following command will be used :

`WBS.1=M50.1,2` ;The sequence goes on only when DIN2 is set to 1. It is reminded that every bit of monitoring M50 represents one of the controller's digital inputs.
`WBS.1=DIN.1,2` ;Same effect as above but the alias DIN is used instead of monitoring M50.

Testing various inputs simultaneously by choosing the adequate mask is also possible. For example, the following command has to be used to continue the sequence when DIN1 **and** DIN2 are set to 1:

`WBS.1=M50.1,3` ;The sequence only goes on when the digital inputs DIN1 **and** DIN2 are set to 1.

14.1.3 Wait on values: WPL, WSL, WPG and WSG commands

When the controller meets the **WPL** command (**W**ait **P**arameter **L**ower than), it waits for the register specified in <P1> to be lower than the value given in <P2> to continue the sequence execution. The **WSL** command (**W**ait **S**igned **L**ower than) works like the WPL command but in this case the parameters <P1> and <P2> can be signed.

The **WPG** command (**W**ait **P**arameter **G**reater than) works the same way but the register must be greater than the value included in <P2> to continue the sequence execution. The **WSG** command (**W**ait **S**igned **G**reater than) works like the WPG command but in this case the parameters <P1> and <P2> can be signed.

Remark: This function is not only dedicated to the movements but also to all K, M or X registers. If the values specified in <P2> are never reached by the chosen register, the pause has an infinite duration. If the condition is already met when the command is executed, the sequence goes on immediately.

The value in <P2> is only taken into consideration when WPL, WSL, WPG or WSG commands are executed.

Command	<P1>	<P2>	Comment
WPL.<axis> = <P1>, <P2>	Register to test	Value of register	Waits for the register <P1> to be lower than the value of <P2>.
WSL.<axis> = <P1>, <P2>	Register to test	Value of register	Waits for the register <P1> to be lower than the value of <P2>. **<P1> and <P2> can be signed.**
WPG.<axis> = <P1>, <P2>	Register to test	Value of register	Waits for the register <P1> to be greater than the value of <P2>.
WSG.<axis> = <P1>, <P2>	Register to test	Value of register	Waits for the register <P1> to be greater than the value of <P2>. **<P1> and <P2> can be signed.**

Example:

Here is the extract of a sequence. The motor is supposed to be initially in position 0.

`ACC.1=500000`	;Definition of a_{max}.
`SPD.1=200000`	;Definition of v_{max}.
`POS.1=300000`	;The motor moves to the position 300000 at a speed of 200000 and an acceleration of 500000 increments.
`WPG.1=M11.1,10000`	;X2.1=1 command is executed only when the motor real speed given by monitoring M11 is over 10000 increments.
`X2.1=1`	;Value 1 is attributed to the user variable X2.

14.1.4 Controller busy: WTB command

Available on	DSC2P	DSC2V	~~DSCDP~~	~~DSCDL~~	~~DSCDM~~

When a controller executes a waiting command like WTM for example, it cannot do something else at the same time: the controller is **busy**. The **WTB** (**W**ai**T** **B**usy) command allows the μ-master to wait for an axis or a group of axes not to be busy any more before executing the next commands. The typical use of this command is to wait for the end of a movement on an axis before starting a movement on another axis. This command cannot be used on a dual axes controller (DSCDP, DSCDL and DSCDM).

Caution: The WTB command can only be used if several axis are chained and can only be executed by the μ-master (axis 0). It means that the command will always be WTB.**0** = <P1>.

Command format	<P1>	Comment
WTB.0 = <P1>	Mask of the axis or axes numbers	Waits for the axis or axes not to be busy any more before executing the next command.

The value contained in <P1> is a **mask** which corresponds to the axis or axes that the μ-master has to wait for. The selected axis or axes numbers are given when the bits corresponding to <P1> value are set to 1 (in binary), as shown in the following table. Value 4 only selects axis 2 and value 41 selects axis 0, 3 and 5.

<P1> values		**Axis 31**	**Axis 30 - 7**	**Axis 6**	**Axis 5**	**Axis 4**	**Axis 3**	**Axis 2**	**Axis 1**	**Axis 0**
		Bit# 31	**Bit# 30 - 7**	**Bit# 6**	**Bit# 5**	**Bit# 4**	**Bit# 3**	**Bit# 2**	**Bit# 1**	**Bit# 0**
2	**2**	0	000000000000000000000000	0	0	0	0	0	1	0
4	**4**	0	000000000000000000000000	0	0	0	0	1	0	0
41	**29**	0	000000000000000000000000	0	1	0	1	0	0	1
Dec.	**Hexa.**	**Binary (only bits set to 1 will be tested)**								

Example:

If the axes 0 and 1 are chained and it is wished to start a movement on axis 1, then when this movement is over, to start a second movement on axis 0. The following sequence extract is memorized in axis 0.

POS.1=300000	;The axis 1 goes to the position 300000.
WTM.1	;Waits for the axis 1 to finish the movement, so the axis 1 becomes busy until the end of the movement (refer to §14.1.1.2).
WTB.0=2	**;Waits for the axis 1 not to be 'busy' any more** before executing the POS.0=10000 command**.**
POS.0=10000	;When the movement of the axis 1 is over, the axis 0 goes to the position 10000. (refer to §15.29 for another example)

14.2 Tests and jumps to labels

These functions allow to go to a specific part of the sequence, and some will do it only if a condition is verified.

14.2.1 Labels

Example:

```
:11.2
```

Labels are marks which can be placed in a sequence and to which it is possible to come back with instructions of unconditional jump like JMP or conditional like IEQ (jumps if the accumulator is equal to) for instance. The label syntax is as follows, different from the commands (refer also to §6. for general syntax):

Syntax:

: <label_#>.<axis>

:<label_#>	Label distinctive number, defining a part of the controller sequence **Possible values:** **Integer from 0 to 511**. Labels **:79**, **:80** are specific
.<axis>	Axis number that contains the sequence. **Possible values:** **Integer from 0 to 30** depending on the axis used.

Examples:

`:1.1`	;Label n°1 for a sequence of axis 1.
`:10.4`	;Label n°10 for a sequence of axis 4.
`:123.2`	;Label n°123 for a sequence of axis 2.

14.2.1.1 Particular labels

Particular labels are labels in which the controller automatically goes on with the sequence execution if a particular event takes place, as long as the corresponding label is present in the sequence.

`:79`	;If the label 79 is available, the sequence is **automatically executed** from this label, as soon as the controller is switched on. It is not necessary to enter the JMP command.
`:80`	;If the **controller enters to error mode** the sequence execution automatically goes on to label 80. It deals with the error, and if necessary goes on with the sequence execution. If the label 80 is not available, then the sequence is interrupted. The value of monitoring M64 determines the current error.

Remark: For the label 80, when the error is processed, it is possible to continue the sequence execution from any label with the JMP instruction, or from the place where the error has occurred with the RET command. With the JMP command, it is recommended to erase the stack with the POP command (refer to §14.2.4).

Example:

Here is the sequence:

:79.1	**;Label n°79. the sequence is executed as soon as the controller is switched on.**
PWR.1=1	;Initialization then power supply in the phases, the motor is in closed loop.
IND.1	;The motor moves up to the index.
WTM.1	;Waits until the movement is finished
SET.1=0	;Definition of '0 machine' where the motor is.
MMD.1=1	;Selection of a S-curve movement.
JRT.1=10	;Jerk time = 10 x STI (STI = 166.67µs for the DSC2P/DSC2V and 500µs for the DSCDP, DSCDL and DSCDM)
ACC.1=500000	;Definition of a_{max}.
SPD.1=200000	;Definition of v_{max}.
:10.1	**;Label n°10.**
POS.1=300000	;The motor moves to the position 300000 at a speed of 200000 and an acceleration of 500000 increments.
WTM.1	;Waits until the movement is finished
POS.1=0	;The motor moves to the position 0 with the same speed and acceleration as before.
WTM.1	;Waits until the movement is finished
JMP.1=10	;Unconditional jump, the execution of the sequence goes on from label n°10.
:80.1	**;Label n°80. In case of error, the execution of the sequence continues from this place.**
POP.1	;The stack is erased.
RST.1	;The error is reset.
JMP.1=79	;The sequence continues at label n°79, from the beginning in this case.

All benefits that can produce the use of particular labels in the execution of a sequence, especially when handling errors and warnings, is demonstrated here.

14.2.2 Unconditional jump: JMP command

As soon as the controller meets the **JMP** command (**J**u**MP**), the sequence execution goes on from the label indicated in <P1>. The bit# 8 of SD1 is set to 1 ('Status Drive')

Sent on-line, the JMP command **starts the sequence execution** from the label indicated in <P1>.

Command	<P1>	Value of <P1>	Comment
JMP.<axis> = <P1>	n° of label	0 - 511	The execution of the sequence continues on the label indicated by <P1>

Example:

JMP.1=12 ;The sequence execution starts from label :12.1

JMP.1=X1.1 ;If X1.1=10, the sequence starts or jumps to label :10.1

Remark: If the label 79 is present, the sequence is **automatically executed** from this label when the controller is switched on.
If the label given by <P1> is not in the sequence, a **MAX SEQ LINE ERR** error (M64=37) appears and the sequence stops (refer to §18. for the warning list).

14.2.3 Conditional jump: TST, JEQ, JNE, JLT and JGT commands

To compare a register value to another register value or an immediate value with another immediate value, the **TST** command (**TeST**) is first performed.

Then, if the following '**test jump**' command is true, the sequence execution will continue from the label in <P1> (the bit# 8 of SD1 is set to 1).

The commands '**Test jump**' are: **JEQ** (**J**ump if **EQ**ual), **JNE** (**J**ump if **N**ot **E**qual), **JLT** (**J**ump if **L**i**T**tle than) and **JGT** (**J**ump if **G**rea**T**er than).

If the condition is not met, the sequence execution goes on from the line following the test.

Command	<P1>, <P2>	Comment
TST.<axis> = <P1>, <P2>	2 values	Compares the values contained in registers <P1> and <P2>, or the immediate values entered for <P1> and <P2>

Note: For floating point variables (F registers), the **FTST** command replaces **TST**.

Command	<P1>	Value of <P1>	Comment
JEQ.<axis> = <P1>	<label_#>	0-511	If <P1>=<P2> the execution of the sequence goes on from label shown by <P1>.
JNE.<axis> = <P1>			If <P1>≠<P2> the execution of the sequence goes on from label shown by <P1>.
JLT.<axis> = <P1>			If <P1> < <P2> the execution of the sequence goes on from label shown by <P1>.
JGT.<axis> = <P1>			If <P1> > <P2> the execution of the sequence goes on from label shown by <P1>.

Example:

`X2.1=20`	;Attributes the value 20 to X2 variable.
`TST.1=X1.1,25`	;Compares the X1 user variable with the number 25.
`JGT.1=10`	;Sequence execution goes on from label 10 if X1>25.
`JEQ.1=X2`	;Sequence execution goes on from label 20 (value contained in the user variable X2) if X1=25.
`POS.1=50000`	;If no test is right when X1<25, the sequence goes on from the following line and the motor moves to the position 50000.

```
:10.1
Instruction 1
:20.1
Instruction 2
```

14.2.4 Routine commands: CAL, RET and POP

The controller programming allows the use of **routines**. A routine is an independent part of a sequence, delimited by a **label** and a **RET** command (**RET**urn). When the controller meets the **CAL** command (**CAL**l), the routine is executed until it reaches the RET command, then the sequence continues from the command following the CAL command.

It is possible for a routine to call another routine (refer to the example below), which can itself call a third routine, etc. It is possible to go up to **256 levels**. The routines are said to be **embedded**. However, a routine cannot call itself.

When a routine is called, the number of the line following the CAL command is memorized in a **stack**. As soon as the routine is finished, the sequence goes on from the line stored on the top of the stack and this line is erased from the stack. With embedded routines the stack may contain a large amount of line numbers, and if the controller detects an error when the sequence is executing a high level subroutine, the sequence jumps to the particular label n°80, as seen previously. If in this label the error is processed and the sequence execution is started from the first line, then the numbers of lines which were stored in the stack will remain because they have not been erased with the RET command. A second error may appear and fill in a bit more the stack. A capacity overflow may happen if several errors happen. To avoid it, the contents of the stack is erased while treating the error on label 80 with the **POP** command (**POP** up).

Remark: The values contained in the **accumulator** are kept when a routine is over (RET) even if the routine also uses an accumulator and even if there are several embedded routines.
If the **RET** command is used in a routine; it is **not possible** to use a **POP** command in the same routine (stack erased, no address where to go back to).

Command	<P1>	<P1> value	Comment
CAL.<axis> = <P1>	<label_#>	0-511	Call of the subroutine located in label shown by <P1>.
RET.<axis>	-	-	Tells the end of the subroutine. The execution goes on at line following CAL instruction.
POP.<axis>	-	-	Clears all levels of the stack.

Example:

```
:10.1            ;Label 10, example start
CAL.1=50         ;Routine call, the sequence execution goes on from label 50.
Instruction 3
Instruction 4
END.1            ;End of main sequence, the sequence execution finishes.

:50.1            ;Label 50. First routine starts.
Instruction 5
CAL.1=100        ;The routine calls a subroutine, the execution goes on from label 100.
Instruction 6
RET.1            ;End of the first routine, the sequence goes on with instruction 3.

:100.1           ;Label 100. Second routine starts.
Instruction 7
Instruction 8
RET.1            ;End of the second routine, the sequence execution goes on with the instruction
                 which follows the CAL command, it is instruction 6.
:80.1            ;Particular label 80. In case of error, the sequence execution goes on from this label.
POP.1            ;The stack is erased.
Instruction 9    ;The error is processed.
JMP.1=10         ;Jump to label 10
END.1            ;Stop the sequence
```

In the example above, the commands have been executed in the following order: Instruction 5, 7, 8, 6, 3 and 4. Instruction 9 is only executed when an error is displayed.

14.3 Accumulator functions

The accumulator functions allow mathematical operations, tests and jumps, depending on the XAC value.

14.3.1 Set the accumulator: XAC

The **accumulator** is a particular variable which makes extra operations with respect to those that can be made on the other registers. The accumulator also accepts arithmetical and logical **dyadic** operations. Moreover, the value contained in the accumulator is retrieved at the output of a routine, even if this routine uses also the accumulator.

The **XAC** command enables the access to the accumulator. This command allows the user to store or to read the accumulator value, but also to make arithmetical and logical operations.

	Monadic operation	**Dyadic operation**
	Arithmetical operations	
Addition	XAC.1 + = <P1>	XAC.1 = <P1> + <P2>
Subtraction	XAC.1 - = <P1>	XAC.1 = <P1> - <P2>
Multiplication	XAC.1 * = <P1>	XAC.1 = <P1> * <P2>
Division	XAC.1 / = <P1>	XAC.1 = <P1> / <P2>
	Logical operations	
Not	XAC.1 ~ = <P1>	
And	XAC.1 & = <P1>	XAC.1 = <P1> & <P2>
Or	XAC.1 I = <P1>	XAC.1 = <P1> I <P2>
Not and	XAC.1 & ~ = <P1>	XAC.1 = <P1> & ~ <P2>
Not or	XAC.1 I ~ = <P1>	XAC.1 = <P1> I ~ <P2>
Shift to the right	XAC.1 >> = <P1>	XAC.1 = <P1> >> <P2>
Shift to the left	XAC.1 << = <P1>	XAC.1 = <P1> << <P2>

Examples:

`XAC.1=1029` ;The accumulator is equal to 1029.

`XAC.1=K2:2.1` ;Attributes the value contained at the depth 2 of parameter K2 to the accumulator.

`XAC.1=SPD.1-X12.1` ;The accumulator is equal to the difference between the programmed v_{max} maximum value (contained in parameter K211) and the X12 user variable value.

Remark: Accumulator max. value: XAC $\leq (2^{31}- 1)$, and min. value: XAC $\geq - 2^{31}$
Values in <P1> and <P2> may be immediate values or registers.

Caution: **XAC** is pointing on the user variable X511; thus, **never use the register X511!**

14.3.2 Test XAC value: IEQ, INE, ILT, IGT, ILE, IGE, JBS and JBC

IEQ (**I**f accumulator **EQ**ual), **INE** (**I**f accumulator **N**ot **E**qual), **ILT** (**I**f accumulator **L**ower **T**han), **IGT** (**I**f accumulator **G**reater **T**han), **ILE** (**I**f accumulator **L**ower or **E**qual), **IGE** (**I**f accumulator **G**reater or **E**qual), **JBS** (**J**ump if **B**it **S**et) and **JBC** (**J**ump if **B**it **C**lear) commands realize some tests on the **values contained in the accumulator** and pursue the sequence execution according to the test result, at any label.

Remark: If the condition is not met, the sequence execution goes on from the line following the test. For JBS and JBC commands, <P1> indicates the number of the bit to test. It is reminded that bits are numbered from the right to the left from 0 to 31.

Command	<P1>	<P2>	Comment
IEQ.<axis> = <P1>, <P2>	Reference value	<label_#>	If XAC=<P1> the execution of the sequence goes on from the label indicated by <P2>.
INE.<axis> = <P1>, <P2>	Reference value	<label_#>	If XAC≠<P1> the execution of the sequence goes on from the label indicated by <P2>.
ILT.<axis> = <P1>, <P2>	Reference value	<label_#>	If XAC<<P1> the execution of the sequence goes on from the label indicated by <P2>.
IGT.<axis> = <P1>, <P2>	Reference value	<label_#>	If XAC><P1> the execution of the sequence goes on from the label indicated by <P2>.
ILE.<axis> = <P1>, <P2>	Reference value	<label_#>	If XAC≤<P1> the execution of the sequence goes on from the label indicated by <P2>.
IGE.<axis> = <P1>, <P2>	Reference value	<label_#>	If XAC≥<P1> the execution of the sequence goes on from the label indicated by <P2>.
JBS.<axis> = <P1>, <P2>	N° of the bit to test	<label_#>	If the bit indicated by <P1> of XAC is set to 1 the execution goes on from label <P2>.
JBC.<axis> = <P1>, <P2>	N° of the bit to test	<label_#>	If the bit indicated by <P1> of XAC is set to 0 the execution goes on from label <P2>.

The instructions given above may not necessarily be directly preceded by the XAC command.

Example:

`X3.1=7` ;Attributes the value 7 to the X3 user variable.

`XAC.1=5` ;The accumulator is equal to 5 which represents 101 in binary. The bits 0 and 2 of the accumulator are then set to 1 and all the others to 0.

XAC			
Dec. value	**Bit# 2**	**Bit# 1**	**Bit# 0**
5	1	0	1

`JBS.1=1,10` ;Jumps to the label 10 if the bit 1 of the accumulator is set to 1. As it is not the case, the sequence execution goes on from the next line.

`ILT.1=X3.1,20` ;The execution of the sequence continues from label 20 because XAC=5 < X3=7.

`:10.1`

`Instructions`

`:20.1;`

`JBC.1=1,10` ;The execution of the sequence continues from label 10 because bit 1 of the accumulator is really equal to 0.

14.4 Sequences handling

A sequence of commands can be memorized in the controller, it is called a **user sequence**. A sequence may contain up to **8190 lines** for a DSC2P and DSC2V or **4095 lines** per axes for a DSCDP, DSCDL and DSCDM. The sending of commands or sequences, as well as all communications with the controller are usually realized with **ETEL Tools** graphical interface software.

14.4.1 Stop a sequence: HLT, HLB, HLO

The three commands **stop the movement** of the motor as well as the **execution** of the **user sequence**.

The **HLT command** (**H**a**LT** sequence and movement) stops the movement with the maximum deceleration that the controller can supply and interrupts the current sequence execution.

Caution: The sudden deceleration that the HLT command provokes may seriously damage the mechanical parts of the system.

The **HLB** command (**H**a**L**t sequence and **B**rake) stops the movement without overrunning the authorized motor maximum deceleration defined in parameter **K206** and interrupts the current sequence execution.

The **HLO** command (**H**a**L**t sequence and power **O**ff) stops the movement with the maximum deceleration that the controller can supply, interrupts the current sequence execution and switches off the power in the motor phases (PWR.1=0).

Caution: The motor may **keep moving** for a while, depending on the system's moving part inertia.

It is advised to use HLO command only if the controller is set with a relay which short-circuits the motor phases when the power is switched off, in order to have a magnetic brake.

Command	Comment	Read parameters
HLT.<axis>	Stops the motor with an infinite deceleration and the sequence.	-
HLB.<axis>	Stops the motor without going over the authorized maximum acceleration programmed in parameter K206 and the sequence.	K206
HLO.<axis>	Stops the motor and switch off the power in motor phases (PWR.1 = 0) and the sequence.	-

14.4.2 Group of axes

To send the same command to several axes, the simplest way is to use the following syntax (refer also to §6.):

<cmd_name>. (<axis1>,<axis2>,...,<axisn>) [<operator>] =[<P1>] [,<P2>]

Example:

POS.(1,2,7,30) = X1.1

14.4.2.1 Mask of axes

It is also possible to define a mask with several axes, using the **%** symbol.

Let's take as example the axes 1 and 4; they give in binary: 10010; this is 18 decimal or 12 Hexadecimal

Examples:

POS.%18 = X1.1
or
POS.%0x12 = X1.1

14.4.3 Clear user variables: CLX

The **CLX** command (**CL**ear **X** variables) clears (sets to 0) all user variables as well as the accumulator.

Command	Comment
CLX.<axis>	Clears all user variables and the accumulator.

Example:

```
X2.1=12     ;The X2.1 user variable contains the value 12.
X4.1=20     ;The X4.1 user variable contains the value 20.
CLX.1       ;The X2 and X4 variables (as well as all the others) contain the value 0.
```

14.4.4 End of sequence: END

When the controller meets the **END** (**END**) command, the execution stops definitively. The END command can be used to store different sequences at the same time in the controller for instance.

Remark: When the controller has executed the last line of the command, it stops even if no END command is executed.

Command	Comment
END.<axis>	Stops the sequence execution.

Example:

Given the following sequence memorized in the controller:

```
:10.1
sequence 1
END.1

:20.1
sequence 2
END.1

:30.1
sequence 3
END.1
```

Entering JMP.1=10 command is enough to start the sequence 1, JMP.1=20 command for the sequence 2, and JMP.1=30 command for the sequence 3.

14.5 Mathematical operations

The mathematical operations which can be applied to registers are divided into two groups: arithmetical operations and logical operations. The symbols used for each operation are displayed in this table.

Mathematical operations can be realized only with 3 types of registers: K parameters, F floats and X variables.

Refer to §14.6 for additional mathematical operations with F registers.

Symbol	Operation
Arithmetical operations	
+	Addition
-	Subtraction
*	Multiplication
/	Division
Logical operations	
~	Not
&	And
I	Or
&~	«Not» and «and»
I~	«Not» and «or»
>>	Shift to the right
<<	Shift to the left

14.5.1 Arithmetical operations

All operations are performed by using integer numbers. Hence, additions, subtractions and multiplications give exact answers, whereas fractional parts are lost with the division.

Examples:

In these examples, what is called result is the value memorized in X1.1 after executing the command. The value at the start of X1.1 is 23 in any case.

Initial value	Operation	Result
X1.1=23	X1.1+=7	30
	X1.1+=-3	20
	X1.1-=5	18
	X1.1*=2	46
	X1.1/=7	3 (integer of 3.2857)

14.5.2 Logical operations

Logical operations allow the user to directly work on the bits which compose the register value. The controller does not accept the values in binary writing, and the user will consequently have to do the conversions himself.

Here is the Boolean operation tables for logical operations:

Basic values		Result				
A	**B**	**¬A** **~ A**	**A ∧ B** **A & B**	**A v B** **A I B**	**A ∧ (¬B)** **A & ~ B**	**A v (¬B)** **A I ~ B**
0	**0**	1	0	0	0	1
0	**1**	1	0	1	0	0
1	**0**	0	0	1	1	1
1	**1**	0	1	1	0	1

Application example of the logical operation **and**. Given X1.1=28=11100:

`X1.1&=22` ;After executing this command. X1 variable contains the value 20 (10100) because there is:

$$\begin{array}{rl} & 28 = 11100 \\ \text{And} & \underline{22 = 10110} \\ & 20 = 10100 \end{array}$$

Description of the operation &~:

The operation **|** (or) sets to 1 one or several bits of a number according to a **mask** without changing the value of the bits which are not in the mask. The operation **&~** ('not' 'and') has been defined in order to set this or these bits to 0, always without changing the value of the other bits.

Example:

Given X1.1=43=101011 and a mask X2.1=18=010010 which selects the bits 1 and 4 of A (the position of the 1 shows the masked bits). In the following table the grey blanks show that the bits of unmasked A have not been changed and that the masked bits (bits 4 and 1) are forced either to 1 or to 0.

		Values expressed in binary					
Commands	**Result**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
X1.1=43	43	1	**0**	1	0	**1**	1
X2.1=18 (mask)	18	0	**1**	0	0	**1**	0
X1.1 I =X2.1	59	1	**1**	1	0	**1**	1
X1.1 &~ =X2.1	41	1	**0**	1	0	**0**	1

Operations >> and <<:

The **operation** >> shifts all bits a number to the right, and the shift is done over 32 bits. If the bit of higher weight (bit 31) is equal to 1 then 1 is entered, if not it will be 0. The **operation** << does the inverted operation, it shifts all bits a number to the left and 0 is always entered. These operations allow the multiplication and division by 2nd powers.

Examples:

In the examples, the result is the value which is memorized in X1.1 after executing the command. The start value of X1.1 is 23=**10111** in both cases.

`X1.1>>=2`	result:	**101**=5.	Shift 2 to the right.
`X1.1<<=3`	result:	**10111000**=184.	Shift 3 to the left.

14.6 Float functions

Most registers in the controller (like K parameters, M monitorings, X variables,...) are defined by **integer** values. A **floating-point value** can be handled only by **F registers**, also called **float**. 256 **F** registers (F0 to F255) are defined (one depth).

14.6.1 Read / write F registers

You may: convert X, K, M, L registers (integers) into an F register (float)
convert an F register (float) into an X or L registers (integers)

Remark: The copy of a F into a X is done by rounding to the closest integer, in the limits of -2^{31} to 2^{31}. XAC (accumulator) may also be used, as it is an X register.

Operations examples	Description
F7.2=F4.2	Copies an F register into another F register
F2.1=3.1415**f**	Immediate F value allocation (displayed in floating-point mode)
X3.5=F2.5	Conversion (float > integer) of an F into an X or L
F2.1=X3.1	Conversion (integer > float) of an X, K, L, M into an F
F2.1	Reads an F register (displayed in floating-point mode)
F2.1+=F3.1	Addition between 2 F registers
F2.1-=F3.1	Subtraction between 2 F registers
F2.1*=F3.1	Multiplication between 2 F registers
F2.1/=F3.1	Division between 2 F registers
FSQRT.5=F2.5,F3.5	Square root of an F register (value taken from <P2>, the result is given in <P1>)
FINV.5=F2.5,F3.5	Invert an F register (value taken from <P2>, the result is given in <P1>)
FSIN.5=F2.5,F3.5	Sine of an F register (value taken from <P2>, the result is given in <P1>)
FCOS.5=F2.5,F3.5	Cosine of an F register (value taken from <P2>, the result is given in <P1>)
FSIGN.5=F2.5,F3.5	Reads the sign of an F register in <P2> and writes in <P1> -1., 0. or 1., when the value is <0, =0, or >0
FFP.5=F2.5,F3.5	Reads the fractional part of an F register in <P2> and writes it <P1>
FIP.5=F2.5,F3.5	Reads the integer part of an F register in <P2> and writes it <P1>
FTST.5=F2.5,F3.5	Compares two F registers (similar to the TST command, but for floating-point values)

Caution: All commands **do not** accept to handle floating-point and integer values together.

Example: ~~F5.1+=X1.1~~ is **not allowed!** Thus, the conversion command (=) should be used every time when integer and float should be mixed up.

THIS PAGE IS INTENTIONALLY LEFT BLANK

Chapter E: Appendixes

15. Commands examples & reference list

All the values present in the following examples are given in ISO unit:

- Position: m, mm, µm, nm or inch
- Speed: m/s, mm/s, µm/s, nm/s or inch/s
- Acceleration: m/s^2, mm/s^2, $\mu m/s^2$, nm/s^2 or $inch/s^2$
- Jerk: s, ms, µs or ns

15.1 AXI, SAV example

The following commands should be used on-line through the Terminal or the Scope. (It is useless to have them in the middle of sequence as the rest of it will be lost because a switch on/off is needed after the SAV command).

At the beginning, the controller has the axis number 0.

SER.0	;Reads the serial # of the controller axis 0, here e.g. : 5968 (returned from the controller)
AXI.0=5968,7	;The axis 0 now bears the number 7
SAV.7=2	;Saves the new axis number (and registers X, K,...) in the flash

Remark: It is not necessary to reset all the controllers of the daisy chain. Only the axis with the new axis number can be reset.

15.2 BRK example

In this example, a rotary motor with an infinite movement is used.

```
:10.0           ;Label 10
PWR.0=1         ;Power on
WTM.0           ;Waits until the movement is finished
IND.0           ;Starts the homing
WTM.0           ;Waits until the movement is finished
SPD.0=1.0       ;Sets the maximum speed to 1 (ISO unit)
ACC.0=20.0      ;Sets the maximum acceleration to 20 (ISO unit)
MMD.0=24        ;Sets the infinite rotary movement

:30.0           ;Label 30
POS.0=1         ;Goes to the position 1 (ISO unit)
WTT.0=5.0       ;Waits for 5 (ISO unit)
BRK.0           ;Stops the movement with the acceleration programmed in parameter K206
WTT.0=5.0       ;Waits for 5 (ISO unit)
JMP.0=30        ;Jumps to label 30
```

15.3 CAL, RET, POP example

```
:10.0           ;Label 10
PWR.0=1         ;Power on
WTM.0           ;Waits until the movement is finished
IND.0           ;Starts the homing
WTM.0           ;Waits until the movement is finished
SPD.0=360.0     ;Sets the maximum speed to 360 (ISO unit)
ACC.0=200.0     ;Sets the maximum acceleration to 200 (ISO unit)

:30.0           ;Label 30
POS.0=360.0     ;Goes to the position 360 (ISO unit)
WTM.0           ;Waits until the movement is finished
CAL.0=60        ;Calls the subroutine 60
POS.0=30.0      ;Goes to the position 30 (ISO unit)
WTM.0           ;Waits until the movement is finished
CAL.0=70        ;Calls the subroutine 70
JMP.0=30        ;Jumps to label 30

:60.0           ;Label 60
X1.0=450        ;Sets user's variable X1 of axis 0 to 450
WTT.0=1.0       ;Waits for 1 (ISO unit)
RET.0           ;Returns to the calling routine

:70.0           ;Label 70
X10.0+=2        ;Adds 2 to the previous value of the user's variable X10 of the axis 0
X1.0=130        ;Sets user's variable X1 of axis 0 to 130
WTT.0=1.0       ;Waits for 1 (ISO unit)
RET.0           ;Returns to the calling routine

:80.0           ;Label 70
POP.0           ;Erases the stack
X10.0=0         ;Sets user's variable X10 of axis 0 to 0
X1.0=0          ;Sets user's variable X1 of axis 0 to 0
RST.0           ;Resets most of the errors
WTT.0=1.0       ;Waits for 1 (ISO unit)
PWR.0=1         ;Power on
WTM.0           ;Waits until the movement is finished
JMP.0=30        ;Jumps to label 30
```

15.4 CAM example

`:10.0`	;Label 10
`PWR.0=1`	;Power on
`WTM.0`	;Waits until the movement is finished
`IND.0`	;Starts the homing
`WTM.0`	;Waits until the movement is finished
`SPD.0=360.0`	;Sets the maximum speed to 360 (ISO unit)
`ACC.0=200.0`	;Sets the maximum acceleration to 200 (ISO unit)
`:30.0`	;Label 30
`POS.0=360.0`	;Goes to the position 360 (ISO unit)
`WTM.0`	;Waits until the movement is finished
`CAL.0=60`	;Calls the subroutine 60
`POS.0=30.0`	;Goes to the position 30 (ISO unit)
`WTM.0`	;Waits until the movement is finished
`CAL.0=70`	;Calls the subroutine 70
`JMP.0=30`	;Jumps to label 30
`:60.0`	;Label 60
`CAM.0=20`	;Global diminution to 20% of speed, acceleration and increases the waiting time (refer to §13.3.7)
`WTT.0=1.0`	;Waits for 1 (ISO unit)
`RET.0`	;Returns to the calling routine
`:70.0`	;Label 70
`CAM.0=100`	;Sets the default value of the CAM (no diminution of speed and acceleration)
`WTT.0=3.0`	;Waits for 3 (ISO unit)
`RET.0`	;Returns to the calling routine

15.5 CLX example

```
:10.0           ;Label 10
X1.0=1          ;Sets user's variable X1 of axis 0 to 1
X2.0=2          ;Sets user's variable X2 of axis 0 to 2
X3.0=3          ;Sets user's variable X3 of axis 0 to 3
WTT.0=1.0       ;Waits for 1 (ISO unit)
CLX.0           ;Clears all user's variables X (resets to 0) as well as the accumulator
WTT.0=2.0       ;Waits for 2 (ISO unit)
JMP.0=10        ;Jumps to label 10
```

15.6 DOUT example

```
:10.0         ;Label 10
DOUT.0=0      ;Clears all the DOUTs
WTT.0=1.0     ;Waits for 1 (ISO unit)
DOUT.0=7      ;Enables the DOUTs 1, 2 and 3 (only DOUT1 and 2 on DSCDP and DSCDL)
WTT.0=1.0     ;Waits for 1 (ISO unit)
JMP.0=10      ;Jumps to label 10
```

15.7 END example

M96 must be monitored to see what happens in this example.

```
:10.0           ;Label 10
PWR.0=1         ;Power on
WTM.0           ;Waits until the movement is finished
IND.0           ;Starts the homing
WTM.0           ;Waits until the movement is finished
SPD.0=360.0     ;Sets the maximum speed to 360 (ISO unit)
ACC.0=200.0     ;Sets the maximum acceleration to 200 (ISO unit)

:30.0           ;Label 30
POS.0=360.0     ;Goes to the position 360 (ISO unit)
WTM.0           ;Waits until the movement is finished
CAL.0=60        ;Calls the subroutine 60
POS.0=30.0      ;Goes to the position 30 (ISO unit)
WTM.0           ;Waits until the movement is finished
CAL.0=70        ;Calls the subroutine 70
JMP.0=30        ;Jumps to label 30

:60.0           ;Label 60
X1.0=450        ;Sets user's variable X1 of axis 0 to 450
WTT.0=1.0       ;Waits for 1 (ISO unit)
RET.0           ;Returns to the calling routine

:70.0           ;Label 70
X10.0+=2        ;Adds 2 to the previous value of the user's variable X10 of the axis 0
X1.0=130        ;Sets user's variable X1 of axis 0 to 130
WTT.0=1.0       ;Waits for 1 (ISO unit)
RET.0           ;Returns to the calling routine

:80.0           ;Label 80
POP.0           ;Erases the stack
X10.0=0         ;Sets user's variable X10 of axis 0 to 0
X1.0=0          ;Sets user's variable X1 of axis 0 to 0
RST.0           ;Resets most of the errors
WTT.0=1.0       ;Waits for 1 (ISO unit)
PWR.0=1         ;Power on
WTM.0           ;Waits until the movement is finished
END.0           ;Stops the execution of the sequence
```

15.8 F registers (float) example

`:10.0`	;Label 10
`X10.0=0`	;Sets user's variable X10 of axis 0 to 0
`F100.0=5.0f`	;Sets float F100 of axis 0 to 5
`F200.0=10.0f`	;Sets float F200 of axis 0 to 10
`F100.0*=F100.0`	;$F100^2$ (= 25)
`F100.0-=f200.0`	;F100 - F200 (= 15)
`F100.0/=2.0f`	;F100 / 2 (= 7.5)
`X10.0=f100.0`	;Sets user's variable X10 of axis 0 with the value of F100 (as the user's variable must include an integer, X10.0=8)
`END.0`	;Stops the execution of the sequence

15.9 HLT, HLB, HLO example

```
:10.0           ;Label 10
PWR.0=1         ;Power on
WTM.0           ;Waits until the movement is finished
IND.0           ;Starts the homing
WTM.0           ;Waits until the movement is finished
SPD.0=360.0     ;Sets the maximum speed to 360 (ISO unit)
ACC.0=300.0     ;Sets the maximum acceleration to 300 (ISO unit)
MMD.0=24        ;Sets the infinite rotary movement

:30.0           ;Label 30
POS.0=1         ;Goes to the position 1 (ISO unit)
WTT.0=5.0       ;Waits for 5 (ISO unit)
HLT.0           ;Stops the motor with an infinite deceleration and the sequence
END.0           ;Stops the execution of the sequence
```

Remark: The principle is identical for the HLB and HLO command:
Command HLB stops the motor without going over the authorized maximum acceleration programmed in parameter K206 and stops the sequence.
Command HLO stops the motor, switch off the power in the motor phases (PWR.<axis>=0) and stops the sequence

15.10 JBS, JBC example

```
:10.0           ;Label 10
PWR.0=1         ;Power on
WTM.0           ;Waits until the movement is finished
IND.0           ;Starts the homing
WTM.0           ;Waits until the movement is finished
SPD.0=360.000   ;Sets the maximum speed to 360 (ISO unit)
ACC.0=300.000   ;Sets the maximum acceleration to 300 (ISO unit)
X10.0=0         ;Sets user's variable X10 of axis 0 to 0

:20.0           ;Label 20
XAC.0=X10.0     ;Puts the value of the user's variable X10 of axis 0 in the accumulator (XAC)
JBS.0=0,30      ;If the bit# 0 of the accumulator is equal to 1, the execution goes on from label 30
JBC.0=0,40      ;If the bit# 0 of the accumulator is equal to 0, the execution goes on from label 40
JMP.0=20        ;Goes to label 20

:30.0           ;Label 30
POS.0=100.0     ;Goes to the position 100 (ISO unit)
WTT.0=1.00000   ;Waits for 1 (ISO unit)
POS.0=0.00000   ;Goes to the position 0 (ISO unit)
WTT.0=1.00000   ;Waits for 1 (ISO unit)
JMP.0=20        ;Goes to label 20

:40.0           ;Label 40
POS.0=10.0      ;Goes to the position 10 (ISO unit)
WTT.0=2.00000   ;Waits for 2 (ISO unit)
POS.0=0.00000   ;Goes to the position 0 (ISO unit)
WTT.0=2.00000   ;Waits for 2 (ISO unit)
JMP.0=20        ;Goes to label 20
```

15.11 JMP example

X10 must be monitored to see what happens in this example:

```
:10.0            ;Label 10
X10.0=0          ;Sets user's variable X10 of axis 0 to 0
WTT.0=4.0        ;Waits for 4 (ISO unit)

:20.0            ;Label 20
X10.0+=2         ;Adds 2 to the previous value of the user's variable X10 of the axis 0
WTT.0=1.0        ;Waits for 1 (ISO unit)
JMP.0=40         ;Goes to label 40

:30.0            ;Label 30
X10.0-=1         ;Substracts 1 from the previous value of the user's variable X10 of the axis 0
WTT.0=1.0        ;Waits for 1 (ISO unit)
JMP.0=20         ;Goes to label 20

:40.0            ;Label 40
WTT.0=1.0        ;Waits for 1 (ISO unit)
JMP.0=30         ;Goes to label 30
```

15.12 MMC example

M6, M7, M10 and M11 must be monitored to see what happens in this example:

`:10.2`	;Label 10
`PWR.2=1`	;Power on
`WTM.2`	;Waits until the movement is finished
`IND.2`	;Starts the homing
`WTM.2`	;Waits until the movement is finished
`POS.2=30.0000`	;Goes to the position 30 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`MMD.2=1`	;Sets the S-curve movement
`MMC.2=1`	;Selects the concatenated movements mode
`:30.2`	;Label 30
`POS:1.2=10.0000`	;Goes to the position 10 (ISO unit)
`SPD:1.2=50.0000`	;Sets the maximum speed to 50 (ISO unit)
`ACC:1.2=500.000`	;Sets the maximum acceleration to 500 (ISO unit)
`POS:2.2=0.00000`	;Goes to the position 0 (ISO unit)
`SPD:2.2=10.0000`	;Sets the maximum speed to 10 (ISO unit)
`ACC:2.2=500.000`	;Sets the maximum acceleration to 500 (ISO unit)
`:40.2`	;Label 40
`STA.2=1,0x7`	;Starts the movement specified at the depth 1 by using the parameter specified by the mask 0x7: bit0 (goes to the position 10), bit1 (with a speed of 50) and bit2 (with an acceleration of 500)
`WTT.2=0.2`	;Waits for 0.2 (ISO unit)
`STA.2=2,0x7`	;Starts the movement specified at the depth 2 by using the parameter specified by the mask 0x7: bit0 (goes to the position 0), bit1 (with a speed of 10) and bit2 (with an acceleration of 500)
`WTM.2`	;Waits until the movement is finished
`WTT.2=2.00000`	;Waits for 2 (ISO unit)
`POS.2=30.0000`	;Goes to the position 30 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`WTT.2=1.00000`	;Waits for 1 (ISO unit)
`JMP.2=40`	;Goes to label 40

15.13 MMD, LTN, LTI example

M6, M7, M10 and M11 must be monitored to see what happens in this example:

```
:10.2              ;Label 10
MMD.2=17           ;Sets the rotary S-curve movement
PWR.2=1            ;Power on
WTM.2              ;Waits until the movement is finished
IND.2              ;Starts the homing
WTM.2              ;Waits until the movement is finished
POS.2=0.00000      ;Goes to the position 0 (ISO unit)
WTM.2              ;Waits until the movement is finished
MMD.2=26           ;Sets the rotary look-up table movement
LTN.2=0            ;Selects the look-up table number 0
LTI.2=2.00000      ;Sets the execution time of the look-up table to 2 (ISO unit)

:30.2              ;Label 30
POS.2=10.0000      ;Goes to the position 10 (ISO unit)
WTM.2              ;Waits until the movement is finished
SET.2=0.00000      ;Sets the current position as the '0 machine'
WTT.2=1.0          ;Waits for 1 (ISO unit)
JMP.2=30           ;Goes to label 30
```

Remark: If this example is realized with a linear motor, the mechanical end stop could be reached quickly.

15.14 MMD=3 (calculated mvt.), SET example

The example below illustrates the use of "movements calculated with predefined profiles".
M6 and M10 must be monitored to see what happens in this example.
K230 can be changed to see the different profiles.

```
:10.2              ;Label 10
MMD.2=17           ;Sets the rotary S-curve movement
PWR.2=1            ;Power on
WTM.2              ;Waits until the movement is finished
IND.2              ;Starts the homing
WTM.2              ;Waits until the movement is finished
POS.2=0.00000      ;Goes to the position 0 (ISO unit)
WTM.2              ;Waits until the movement is finished
MMD.2=3            ;Sets the calculated movement with predefined profile
K229.2=2.00000     ;Sets the movement time: 2 (ISO unit)
K230.2=0           ;Sets a triangular movement

:30.2              ;Label 30
POS.2=10.0000      ;Goes to the position 10 (ISO unit)
WTM.2              ;Waits until the movement is finished
SET.2=0.00000      ;Sets the current position as the '0 machine'
WTT.2=1.00000      ;Waits for 1 (ISO unit)
JMP.2=30           ;Jumps to label 30
```

15.15 PWR, IND, ACC, SPD, POS example

`:79.0`	;Autostart label
`PWR.0=1`	;Power on
`IND.0`	;Starts the homing
`WTM.0`	;Waits until the movement is finished
`ACC.0=100.000`	;Sets the acceleration limit to 100 (ISO unit)
`SPD.0=10.0000`	;Sets the speed limit to 10 (ISO unit)
`:10.0`	;Label 10
`POS.0=0.00000`	;Goes to the position 0 (ISO unit) with the acceleration and speed previously set
`WTM.0`	;Waits until the movement is finished
`POS.0=20.0000`	;Goes to the absolute positive position 20 (ISO unit) with the same acceleration and speed as before
`WTM.0`	;Waits until the movement is finished
`WTT.0=1.00000`	;Waits for 1 (ISO unit)
`ACC.0/=5`	;Sets a new acceleration 5 times smaller
`SPD.0/=5`	;Sets a new speed 5 times smaller
`POS.0=30.0000`	;Goes to the position 30 (ISO unit) with the new acceleration and speed
`WTM.0`	;Waits until the movement is finished
`POS.0-=10.000`	;Moves in the negative direction of 10 (ISO unit) with the new acceleration and speed
`WTM.0`	;Waits until the movement is finished
`WTT.0=1.00000`	;Waits for 1 (ISO unit)
`POS.0=-20.000`	;Goes to the absolute negative position -20 (ISO unit) with the new acceleration and speed
`WTM.0`	;Waits until the movement is finished
`PWR.0=0`	;Power off
`END.0`	;Stops the sequence execution

15.16 REI: RTI example

`:10.2`	;Label 10
`K190.2=0`	;Disables the RTI mode of the controller
`K191.2=0`	;No RTI line validated
`K192.2=0`	;No RTI line enabled
`K193.2=0`	;Clears the flag if one RTI has been activated
`:20.2`	;Label 20
`R0:8.2=20`	;Type of RTI: 'Test on the value'
`R0:9.2=60`	;Jumps to label 60
`R0:10.2=1`	;Clears all the WAIT commands
`R0:1.2=0x10001`	;The user's variable X1 is the register to be tested (corresponds to the first parameter (P1) of this type of RTI)
`R0:2.2=0x0000A`	;10 is the value of comparison (corresponds to the second parameter (P2) of this type of RTI)
`R0:3.2=130`	;Jumps to label 60 if X1=10 (if P1=P2) (corresponds to the third parameter (P3) of this type of RTI)
`R0:4.2=0x10002`	;Sets to 1 the bit#0 of the user's variable X2
`:30.2`	;Label 30
`K190.2=1`	;Enables the RTI mode of the controller
`K191.2=0x1`	;The first line (R0) of the RTI is validated
`K192.2=0x1`	;The first line (R0) of the RTI is enabled
`:40.2`	;Label 40
`X1.2=0`	;Sets user's variable X1 of axis 2 to 0
`X2.1=0`	;Sets user's variable X2 of axis 1 to 0
`WTT.2=2.00000`	;Waits for 2 (ISO unit)
`:45.2`	;Label 45
`X1.2+=1`	;Adds 1 to the previous value of the user's variable X1 of the axis 2
`WTT.2=2.00000`	;Waits for 2 (ISO unit)
`JMP.2=45`	;Jumps to label 45
`:60.2`	;Label 60
`X100.2=25`	;Sets user's variable X100 of axis 2 to 25
`WTT.2=2.00000`	;Waits for 2 (ISO unit)
`X100.2=0`	;Sets user's variable X100 of axis 2 to 0
`WTT.2=2.00000`	;Waits for 2 (ISO unit)
`X1.2=0`	;Sets user's variable X1 of axis 2 to 0
`X2.2=0`	;Sets user's variable X2 of axis 2 to 0
`K193.2=0`	;Clears the flag if one RTI has been activated
`REI.2`	;The sequence returns to the line which follows the last executed line before the call of the RTI label

Remark: The REI command executes the RIE and the RET commands within one instruction

15.17 RSD example

```
:79.2            ;Autostart label (the sequence starts automatically)
MMD.2=17         ;Sets the rotary S-curve movement
PWR.2=1          ;Power on
WTM.2            ;Waits until the movement is finished
IND.2            ;Starts the homing
WTM.2            ;Waits until the movement is finished
POS.2=0.00000    ;Goes to the position 0 (ISO unit)
WTM.2            ;Waits until the movement is finished
X1.2=0           ;Sets user's variable X1 of axis 2 to 0

:30.2            ;Label 30
XAC.2=X1.2       ;Copies the value of the user's variable X1 of the axis 2 in the accumulator (XAC)
IEQ.2=10,50      ;Goes to label 50 if XAC = 10
POS.2=10.0000    ;Goes to the position 10 (ISO unit)
WTM.2            ;Waits until the movement is finished
POS.2=0          ;Goes to the position 0 (ISO unit)
WTM.2            ;Waits until the movement is finished
X1.2+=1          ;Adds 1 to the previous value of the user's variable X1 of the axis 2
JMP.2=30         ;Jumps to label 30

:50.2            ;Label 50
PWR.2=0          ;Power off
WTT.2=2.00000    ;Waits for 2 (ISO unit)
RSD.2=255        ;Hardware reset of the controller
```

15.18 Special labels (79, 80) example

```
:79.2             ;Autostart label (the sequence starts automatically)
MMD.2=17          ;Sets the rotary S-curve movement
PWR.2=1           ;Power on
WTM.2             ;Waits until the movement is finished
IND.2             ;Starts the homing
WTM.2             ;Waits until the movement is finished
POS.2=0.00000     ;Goes to the position 0 (ISO unit)
WTM.2             ;Waits until the movement is finished
X1.2=0            ;Sets user's variable X1 of axis 2 to 0
X100.2=0          ;Sets user's variable X100 of axis 2 to 0

:30.2             ;Label 30
XAC.2=X1.2        ;Copies the value of the user's variable X1 of the axis 2 in the accumulator (XAC)
IEQ.2=10,50       ;Goes to label 50 if XAC = 10
POS.2=10.0000     ;Goes to the position 10 (ISO unit)
WTM.2             ;Waits until the movement is finished
POS.2=0           ;Goes to the position 0 (ISO unit)
WTM.2             ;Waits until the movement is finished
X1.2+=1           ;Adds 1 to the previous value of the user's variable X1 of the axis 2
JMP.2=30          ;Jumps to label 30

:50.2             ;Label 50
PWR.2=0           ;Power off
WTT.2=2.00000     ;Waits for 2 (ISO unit)
X20.2+=1          ;Adds 1 to the previous value of the user's variable X20 of the axis 2
JMP.2=79          ;Jumps to label 79

:80.2             ;Label 80 (the sequence jumps automatically to this label if an error occurs)
PWR.2=0           ;Power off
X100.2=25         ;Sets user's variable X100 of axis 2 to 25
END.2             ;Stops the execution of the sequence
```

15.19 SLS example

M36 and M37 must be monitored to see the limit positions.

```
:20.0           ;Label 20
PWR.0=1         ;Power on
IND.0           ;Starts the homing
WTM.0           ;Waits until the movement is finished
SLS.0           ;Searches the limit stroke on a mechanical end stops or limit switch
WTM.0           ;Waits until the movement is finished
PWR.0=0         ;Power off
END.0           ;Stops the sequence execution
```

After having found the mechanical end stop, the motor moves back the distance given in K47.

15.20 STA example

The axis 0 is in μ-master mode (K170=1 or MDE=1).

M0, M1, M10 and M11 must be monitored to see what happens in this example:

```
:10.2              ;Label 10
MMD.2=17           ;Sets the rotary S-curve movement
PWR.2=1            ;Power on
WTM.2              ;Waits until the movement is finished
IND.2              ;Starts the homing
WTM.2              ;Waits until the movement is finished
POS.2=0.00000      ;Goes to the position 0 (ISO unit)
WTM.2              ;Waits until the movement is finished
POS:1.2=100.000    ;Goes to the position 100 (ISO unit)
SPD:1.2=10.0000    ;Sets the maximum speed to 10 (ISO unit)
ACC:1.2=100.000    ;Sets the maximum acceleration to 100 (ISO unit)
POS:2.2=10.000     ;Goes to the position 10 (ISO unit)
SPD:2.2=360.0000   ;Sets the maximum speed to 360 (ISO unit)
ACC:2.2=500.000    ;Sets the maximum acceleration to 500 (ISO unit)

:30.2              ;Label 30
STA.2=1,0x7        ;Starts the movement specified at the depth 1 by using the parameter specified by
                   the mask 0x7: bit0 (goes to the position 100), bit1 (with a speed of 10) and bit2 (with
                   an acceleration of 100)
WTM.2              ;Waits until the movement is finished
WTT.2=1.00000      ;Waits for 1 (ISO unit)
STA.2=2,0x7        ;Starts the movement specified at the depth 2 by using the parameter specified by
                   the mask 0x7: bit0 (goes to the position 10), bit1 (with a speed of 360) and bit2 (with
                   an acceleration of 500)
WTM.2              ;Waits until the movement is finished
WTT.2=1.00000      ;Waits for 1 (ISO unit)
JMP.2=30           ;Jumps to label 30
```

15.21 STE example

M0 and M6 must be monitored to see what happens in this example:

```
:10.2           ;Label 10
MMD.2=17        ;Sets the rotary S-curve movement
PWR.2=1         ;Power on
WTM.2           ;Waits until the movement is finished
IND.2           ;Starts the homing
WTM.2           ;Waits until the movement is finished
POS.2=0.0       ;Goes to the position 0 (ISO unit)
WTM.2           ;Waits until the movement is finished

:30.2           ;Label 30
STE.2+=1.0      ;Moves of 1 (ISO unit) in the positive direction (relative step)
WTM.2           ;Waits until the movement is finished
WTT.2=1.0       ;Waits for 1 (ISO unit)
JMP.2=30        ;Jumps to label 30
```

Remark: The STE command must be used with the utmost prudence as the movement speed is very high.

15.22 STI example

`:10.2`	;Label 10
`MMD.2=17`	;Sets the rotary S-curve movement
`PWR.2=1`	;Power on
`WTM.2`	;Waits until the movement is finished
`IND.2`	;Starts the homing
`WTM.2`	;Waits until the movement is finished
`POS.2=0.00000`	;Goes to the position 0 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`POS:1.2=100.000`	;Goes to the position 100 (ISO unit)
`SPD:1.2=10.0000`	;Sets the maximum speed to 10 (ISO unit)
`ACC:1.2=100.000`	;Sets the maximum acceleration to 100 (ISO unit)
`K160.2=1`	;DIN 1 is tested to execute the STI command
`K161.2=1`	;DIN 1 must be activated to start the movement after the STI command
`K164.2=10.0000`	;Defines the time-out limit (10 in ISO unit) between the moment when the STI command is sent and the moment when DIN 1 = 1.
`:30.2`	;Label 30
`STI.2=1,0x7`	;Starts the movement, on the rising edge of DIN 1, specified at the depth 1 by using the parameter specified by the mask 0x7: bit0 (goes to the position 100), bit1 (with a speed of 10) and bit2 (with an acceleration of 100)
`WTM.2`	;Waits until the movement is finished
`WTT.2=1.00000`	;Waits for 1 (ISO unit)
`POS.2=0.00000`	;Goes to the position 0 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`JMP.2=30`	;Jumps to label 30

15.23 STP example

In this example, a rotary motor with an infinite movement is used.

```
:10.0           ;Label 10
PWR.0=1         ;Power on
WTM.0           ;Waits until the movement is finished
IND.0           ;Starts the homing
WTM.0           ;Waits until the movement is finished
SPD.0=1.0       ;Sets the maximum speed to 1 (ISO unit)
ACC.0=20.0      ;Sets the maximum acceleration to 20 (ISO unit)
MMD.0=24        ;Sets the infinite rotary movement

:30.0           ;Label 30
POS.0=1         ;Goes to the position 1 (ISO unit)
WTT.0=5.0       ;Waits for 5 (ISO unit)
STP.0           ;Stops the movement with an infinite deceleration
WTT.0=5.0       ;Waits for 5 (ISO unit)
JMP.0=30        ;Jumps to label 30
```

15.24 TCL, TMK, TNB, TRS triggers example

:10.2	;Label 10
MMD.2=17	;Sets the rotary S-curve movement
PWR.2=1	;Power on
WTM.2	;Waits until the movement is finished
IND.2	;Starts the homing
WTM.2	;Waits until the movement is finished
POS.2=0.00000	;Goes to the position 0 (ISO unit)
WTM.2	;Waits until the movement is finished
DOUT.2=0	;All digital outputs of the controller are set to 0
TCL.2	;All triggers tables are cleared
TNB.2=2	;The table is defined with 2 triggers per mapping (alias of parameter K187)
TMK.2=3	;Selects the DOUT1 and 2 to be modified by the trigger
:20.2	;Label 20
	Mapping #0 (triggers E_0 and E_1)
E0:0.2=0x00010081	;Positive trigger (0x**81**) and action 1 (0x**01**), controller's DOUTs & SD2
E0:1.2=1	;Controller outputs =0001 bin. DOUT1=1 (masked by TMK), other DOUTs = 0
E0:2.2=1	;'Status Drive' SD2 bits = 01 bin. Bit# 9=0, Bit# 8=1 (masked by parameter K184)
E0:3.2=1137778	;Trigger position = 1137778 [upi]
E1:0.2=0x00010082	;Negative trigger (0x82) and action 1 (0x01), controller's DOUTs & SD2
E1:1.2=2	;Controller's outputs = 0010 bin. DOUT2=1 (masked by TMK), other DOUTs = 0
E1:2.2=1	;'Status Drive' SD2 bits = 01 bin. Bit# 9=0, Bit# 8=1 (masked by parameter K184)
E1:3.2=568889	;Trigger position = 568889 [upi]
TRS.2=0	;Mapping # 0 is activated (alias of parameter K186)
:30.2	;Label 30
POS.2=20.0	;Goes to the position 20 (ISO unit)
WTM.2	;Waits until the movement is finished
WTT.2=1.00000	;Waits for 1 (ISO unit)
POS.2=0.0	;Goes to the position 0 (ISO unit)
WTM.2	;Waits until the movement is finished
WTT.2=1.00000	;Waits for 1 (ISO unit)
JMP.2=30	;Jumps to label 30

15.25 TST, JGT, JEQ, JNE example

`:10.2`	;Label 10
`MMD.2=17`	;Sets the rotary S-curve movement
`PWR.2=1`	;Power on
`WTM.2`	;Waits until the movement is finished
`IND.2`	;Starts the homing
`WTM.2`	;Waits until the movement is finished
`POS.2=0.00000`	;Goes to the position 0 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`X10.2=0`	;Sets user's variable X10 of axis 2 to 0
`:20.2`	;Label 20
`TST.2=X10.2,10`	;Compares the value of X10 of the axis 2 with the value 10
`JEQ.2=30`	;Sequence execution goes on from label 30 if X10.2=10. If X10.2≠10, the sequence goes on from the following line
`JGT.2=40`	;Sequence execution goes on from label 40 if X10.2>10. If X10.2<10, the sequence goes on from the following line
`JNE.2=20`	;As long as X10.2≠10, it jumps to label 20, otherwise the sequence goes on from the following line
`:30.2`	;Label 30
`POS.2=20.0`	;Goes to the position 20 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`WTT.2=1.00000`	;Waits for 1 (ISO unit)
`POS.2=0.0`	;Goes to the position 0 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`WTT.2=1.00000`	;Waits for 1 (ISO unit)
`JMP.2=20`	;Jumps to label 20
`:40.2`	;Label 40
`POS.2=10.0`	;Goes to the position 10 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`WTT.2=2.00000`	;Waits for 2 (ISO unit)
`POS.2=0.0`	;Goes to the position 0 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`WTT.2=2.00000`	;Waits for 2 (ISO unit)
`JMP.2=20`	;Jumps to label 20

15.26 TST, JLT example

`:10.2`	;Label 10
`MMD.2=17`	;Sets the rotary S-curve movement
`PWR.2=1`	;Power on
`WTM.2`	;Waits until the movement is finished
`IND.2`	;Starts the homing
`WTM.2`	;Waits until the movement is finished
`POS.2=0.00000`	;Goes to the position 0 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`X10.2=10`	;Sets user's variable X10 of axis 2 to 10
`:20.2`	;Label 20
`TST.2=X10.2,10`	;Compares the value of X10 of the axis 2 with the value 10
`JEQ.2=20`	;Sequence execution goes on from label 20 if X10.2=10. If X10.2≠10, the sequence goes on from the following line
`JLT.2=40`	;The execution sequence goes on from label 40 if X10.2<10. If X10.2>=10, the sequence goes on from the following line
`:30.2`	;Label 30
`POS.2=20.0`	;Goes to the position 20 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`WTT.2=1.00000`	;Waits for 1 (ISO unit)
`POS.2=0.0`	;Goes to the position 0 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`WTT.2=1.00000`	;Waits for 1 (ISO unit)
`JMP.2=20`	;Jumps to label 20
`:40.2`	;Label 40
`POS.2=10.0`	;Goes to the position 10 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`WTT.2=2.00000`	;Waits for 2 (ISO unit)
`POS.2=0.0`	;Goes to the position 0 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`WTT.2=2.00000`	;Waits for 2 (ISO unit)
`JMP.2=20`	;Jumps to label 20

15.27 WBC, WBS example

```
:10.2             ;Label 10
MMD.2=17          ;Sets the rotary S-curve movement
PWR.2=1           ;Power on
WTM.2             ;Waits until the movement is finished
IND.2             ;Starts the homing
WTM.2             ;Waits until the movement is finished
POS.2=0.00000     ;Goes to the position 0 (ISO unit)
WTM.2             ;Waits until the movement is finished

:30.2             ;Label 30
WBS.2=M50.2,1     ;Waits until bit#1 of monitoring M50 (corresponding to DIN2) is equal to 1
POS.2=20.0        ;Goes to the position 20 (ISO unit)
WTM.2             ;Waits until the movement is finished
WTT.2=1.00000     ;Waits for 1 (ISO unit)
POS.2=0.0         ;Goes to the position 0 (ISO unit)
WTM.2             ;Waits until the movement is finished
WTT.2=1.00000     ;Waits for 1 (ISO unit)

:40.2             ;Label 40
WBC.2=M50.2,1     ;Waits until bit#1 of monitoring M50 (corresponding to DIN2) is equal to 0
POS.2=10.0        ;Goes to the position 10 (ISO unit)
WTM.2             ;Waits until the movement is finished
WTT.2=2.00000     ;Waits for 2 (ISO unit)
POS.2=0.0         ;Goes to the position 0 (ISO unit)
WTM.2             ;Waits until the movement is finished
WTT.2=2.00000     ;Waits for 2 (ISO unit)
JMP.2=30          ;Jumps to label 30
```

15.28 WPG, WPL, WSG, WSL example

```
:10.2               ;Label 10
MMD.2=17            ;Sets the rotary S-curve movement
PWR.2=1             ;Power on
WTM.2               ;Waits until the movement is finished
IND.2               ;Starts the homing
WTM.2               ;Waits until the movement is finished
POS.2=0.00000       ;Goes to the position 0 (ISO unit)
WTM.2               ;Waits until the movement is finished
X10.2=100           ;Sets user's variable X10 of axis 2 to 100

:30.2               ;Label 30
WPG.2=M50.2,2       ;Waits for the value of monitoring M50 of axis 2 to be greater than the value 2
POS.2=20.0000       ;Goes to the position 20 (ISO unit)
WTM.2               ;Waits until the movement is finished
WTT.2=1.00000       ;Waits for 1 (ISO unit)
POS.2=0.00000       ;Goes to the position 0 (ISO unit)
WTM.2               ;Waits until the movement is finished
WTT.2=1.00000       ;Waits for 1 (ISO unit)

:40.2               ;Label 40
WPL.2=M50.2,2       ;Waits for the value of monitoring M50 of axis 2 to be lower than the value 2
POS.2=10.0000       ;Goes to the position 10 (ISO unit)
WTM.2               ;Waits until the movement is finished
WTT.2=2.00000       ;Waits for 2 (ISO unit)
POS.2=0.00000       ;Goes to the position 0 (ISO unit)
WTM.2               ;Waits until the movement is finished
WTT.2=2.00000       ;Waits for 2 (ISO unit)
JMP.2=30            ;Jumps to label 30
```

Remark: WSL and WSG commands work respectively like WPL and WPG commands but in this case the parameters <P1> and <P2> can be signed.

15.29 WTB example

The axis number 0 is in µ-master mode (K170=1 or MDE=1). This command is only available on the DSC2P and DSC2V.

`:10.2`	;Label 10
`MMD.!=17`	;Sets the rotary S-curve movement
`PWR.!=1`	;Power on
`WTM.!`	;Waits until the movement is finished
`IND.!`	;Starts the homing
`WTM.!`	;Waits until the movement is finished
`POS.!=0.00000`	;Goes to the position 0 (ISO unit)
`WTM.!`	;Waits until the movement is finished
`:20.2`	;Label 20
`POS.(0,2,3)=20.0000`	;Goes to the position 20 (ISO unit)
`WTM.!`	;Waits until the movement is finished
`WTB.0=13`	;Waits for the axes 0, 2 and 3 not to be busy any more before executing the next command
`POS.(0,2,3)=10.0000`	;Goes to the position 10 (ISO unit)
`WTM.!`	;Waits until the movement is finished
`WTB.0=13`	;Waits for the axes 0, 2 and 3 not to be busy any more before executing the next command
`JMP.2=20`	;Jumps to label 20

15.30 WTM, WTT example

```
:10.2               ;Label 10
MMD.2=17            ;Sets the rotary S-curve movement
PWR.2=1             ;Power on
WTM.2               ;Waits until the movement is finished
IND.2               ;Starts the homing
WTM.2               ;Waits until the movement is finished
POS.2=0.00000       ;Goes to the position 0 (ISO unit)
WTM.2               ;Waits until the movement is finished

:20.2               ;Label 20
POS.2=10.0          ;Goes to the position 10 (ISO unit)
WTM.2               ;Waits until the movement is finished
WTT.2=2.0           ;Waits for 2 (ISO unit)
POS.2=20.0          ;Goes to the position 20 (ISO unit)
WTM.2               ;Waits until the movement is finished
WTT.2=1.0           ;Waits for 1 (ISO unit)
JMP.2=20            ;Jumps to label 20
```

15.31 WTP example

`:10.2`	;Label 10
`MMD.2=17`	;Sets the rotary S-curve movement
`PWR.2=1`	;Power on
`WTM.2`	;Waits until the movement is finished
`IND.2`	;Starts the homing
`WTM.2`	;Waits until the movement is finished
`POS.2=0.00000`	;Goes to the position 0 (ISO unit)
`WTM.2`	;Waits until the movement is finished
`:20.2`	;Label 20
`POS.2=10.0`	;Goes to the position 10 (ISO unit)
`WTP.2=5.0`	;Waits for the motor to cross position 5 (ISO unit) before going on with the execution of the sequence
`X10.2=1`	;Sets user’s variable X10 of axis 2 to 1
`WTM.2`	;Waits until the movement is finished
`POS.2=0.0`	;Goes to the position 0 (ISO unit)
`WTP.2=3.0`	;Waits for the motor to cross position 3 (ISO unit) before going on with the execution of the sequence
`X10.2=0`	;Sets user’s variable X10 of axis 2 to 0
`WTM.2`	;Waits until the movement is finished
`WTT.2=1.0`	;Waits for 1 (ISO unit)
`JMP.2=20`	;Jumps to label 20

15.32 WTW example

```
:10.2               ;Label 10
MMD.2=17            ;Sets the rotary S-curve movement
PWR.2=1             ;Power on
WTM.2               ;Waits until the movement is finished
IND.2               ;Starts the homing
WTM.2               ;Waits until the movement is finished
POS.2=0.00000       ;Goes to the position 0 (ISO unit)
WTM.2               ;Waits until the movement is finished
K38.2=1.0           ;Sets the minimum time to 1 (ISO unit) for the real position to stay in the position window
K39.2=0.1           ;Sets the maximum position error to 0.1 (ISO unit) in the position window

:20.2               ;Label 20
POS.2=10.0          ;Goes to the position 0 (ISO unit)
WTW.2=0             ;Waits until the movement reaches the window
POS.2=0.0           ;Goes to the position 0 (ISO unit)
WTW.2=0             ;Waits until the movement reaches the window
JMP.2=20            ;Jumps to label 20
```

15.33 XAC, IEQ, ILT, IGT example

```
:10.2           ;Label 10
X10.2=0         ;Sets user's variable X10 of axis 2 to 0

:20.2           ;Label 20
XAC.2=X10.2     ;Copies the value of the user's variable X10 of the axis 2 in the accumulator (XAC)
IEQ.2=10,30     ;Goes to label 30 if XAC = 10
ILT.2=5,40      ;Goes to label 40 if XAC < 5
IGT.2=10,50     ;Goes to label 50 if XAC > 10

:30.2           ;Label 30
X20.2=1         ;Sets user's variable X20 of axis 2 to 1
WTT.2=0.1       ;Waits for 0.1 (ISO unit)
X20.2=0         ;Sets user's variable X20 of axis 2 to 0
WTT.2=0.1       ;Waits for 0.1 (ISO unit)
X10.2+=1        ;Adds 1 to the previous value of the user's variable X10 of the axis 2
JMP.2=20        ;Jumps to label 20

:40.2           ;Label 40
X20.2=2         ;Sets user's variable X20 of axis 2 to 2
WTT.2=1.0       ;Waits for 1 (ISO unit)
X20.2=0         ;Sets user's variable X20 of axis 2 to 0
WTT.2=1.0       ;Waits for 1 (ISO unit)
X10.2+=1        ;Adds 1 to the previous value of the user's variable X10 of the axis 2
JMP.2=20        ;Jumps to label 20

:50.2           ;Label 50
X20.2=4         ;Sets user's variable X20 of axis 2 to 4
WTT.2=2.0       ;Waits for 2 (ISO unit)
X20.2=0         ;Sets user's variable X20 of axis 2 to 0
WTT.2=2.0       ;Waits for 2 (ISO unit)
X10.2+=2        ;Adds 2 to the previous value of the user's variable X10 of the axis 2
JMP.2=20        ;Jumps to label 20
```

15.34 XAC, IGE, ILE example

```
:10.2           ;Label 10
X10.2=0         ;Sets user's variable X10 of axis 2 to 0

:20.2           ;Label 20
XAC.2=X10.2     ;Copies the value of the user's variable X10 of the axis 2 in the accumulator (XAC)
ILE.2=5,30      ;Goes to label 30 if XAC ≤ 5
IGE.2=10,50     ;Goes to label 50 if XAC ≥ 10
X10.2+=1        ;Adds 1 to the previous value of the user's variable X10 of the axis 2
WTT.2=1.0       ;Waits for 1 (ISO unit)
JMP.2=20        ;Jumps to label 20

:30.2           ;Label 30
X20.2=1         ;Sets user's variable X20 of axis 2 to 1
WTT.2=0.1       ;Waits for 0.1 (ISO unit)
X20.2=0         ;Sets user's variable X20 of axis 2 to 0
WTT.2=0.1       ;Waits for 0.1 (ISO unit)
X10.2+=1        ;Adds 1 to the previous value of the user's variable X10 of the axis 2
JMP.2=20        ;Jumps to label 20

:50.2           ;Label 50
X20.2=4         ;Sets user's variable X20 of axis 2 to 4
WTT.2=2.0       ;Waits for 2 (ISO unit)
X20.2=0         ;Sets user's variable X20 of axis 2 to 0
WTT.2=2.0       ;Waits for 2 (ISO unit)
X10.2+=2        ;Adds 2 to the previous value of the user's variable X10 of the axis 2
JMP.2=20        ;Jumps to label 20
```

15.35 XAC, INE example

```
:10.2          ;Label 10
X10.2=0        ;Sets user's variable X10 of axis 2 to 0

:20.2          ;Label 20
XAC.2=X10.2    ;Copies the value of the user's variable X10 of the axis 2 in the accumulator (XAC)
INE.2=1,30     ;Goes to label 30 if XAC ≠ 1
X10.2+=1       ;Adds 1 to the previous value of the user's variable X10 of the axis 2
WTT.2=1.0      ;Waits for 1 (ISO unit)
JMP.2=20       ;Jumps to label 20

:30.2          ;Label 30
X20.2=1        ;Sets user's variable X20 of axis 2 to 1
WTT.2=0.1      ;Waits for 0.1 (ISO unit)
X20.2=0        ;Sets user's variable X20 of axis 2 to 0
WTT.2=0.1      ;Waits for 0.1 (ISO unit)
X10.2+=1       ;Adds 1 to the previous value of the user's variable X10 of the axis 2
JMP.2=20       ;Jumps to label 20
```

15.36 Commands reference list

The following commands can be used with all the controllers except for:

- the WTB command which can be used **only** with the DSC2P
- the CN1 and CN2 command which **cannot** be used with the DSCDM.

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
AUT.<AXIS>=<P1>			Calculates current loop parameters (K80, K81, K98), performs fine phase adjustment (K53 if K52=1) and finds motor connection (K56)			
	1	0	Tunes proportional and integrator gain of the current loop K80, K81, and DC power voltage rate K98			
	2	1	Searches motor phase and sets K56			
	8	3	Sets K53 (fine phase adjustment) if K52=1			
AXI.<AXIS>=<P1>,<P2>			Changes the current axis number of a controller based on its serial number; <P1> is the serial number, <P2> is the new axis number. Example: AXI.1=1245,2 (change axis 1 in axis 2, its serial number is 1245 that is given by SER or M73)			
BRK.<AXIS>			Stops the motor using programmed deceleration K206 but does not stop the sequence			
CAL.<AXIS>=<P1>			Calls a subroutine specified at the label defined by <P1>. The accumulator (XAC) is preserved (in the stack) and it is restored when RET command is executed			
CLX.<AXIS>			Clears all user variables (X register) including accumulator (XAC)			
CPE.<AXIS>			Clears pending error: is used with error process label (:80). When you process at label 80, the pending error bit is set to 1 (bit#0 of SD2). If you do not clear it (with CPE), the next time an error occurs the controller will not jump to label 80			
EDI.<AXIS>			Switches in editor mode. It allows the user to write into sequence register (S). It sets bit#9 (edit mode) of SD1			
END.<AXIS>			Stops the sequence execution and clears bit #8 (seq_on) of status drive 1 (SD1)			
ERR.<AXIS>			Puts the controller in error mode (M64=116)			
EXI.<AXIS>			Disables the editor mode of the sequence and builds label table. It clears bit#9 (edit mode) of SD1			
FCOS.<AXIS>=<P1>,<P2>			Executes the operation: <P1> = cos(<P2>). Both operands (<P1> and <P2>) are only float (type F)			

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
FFP.<AXIS>=<P1>,<P2>			Executes the operation: <P1> = fractional part of <P2>. Both operands (<P1> and <P2>) are only float (type F)			
FINV.<AXIS>=<P1>,<P2>			Executes the operation: <P1> = 1/(<P2>). Both operands (<P1> and <P2>) are only float (type F)			
FIP.<AXIS>=<P1>,<P2>			Executes the operation: <P1> = integer part of <P2>. Both operands (<P1> and <P2>) are only float (type F)			
FSIGN.<AXIS>=<P1>,<P2>			Executes the operation: <P1> = sign(<P2>). A negative number will return -1 and a positive +1. Both operands (<P1> and <P2>) are only float (type F)			
FSIN.<AXIS>=<P1>,<P2>			Executes the operation: <P1> = sin(<P2>). Both operands (<P1> and <P2>) are only float (type F)			
FSQRT.<AXIS>=<P1>,<P2>			Executes the operation: <P1> = sqrt(<P2>). Both operands (<P1> and <P2>) are only float (type F)			
FTST.<AXIS>=<P1>,<P2>			Tests 2 float registers			
HLB.<AXIS>			Stops the sequence by clearing bit#8 (seq_on) of status drive 1 (SD1) and brakes the motor using programmed deceleration K206			
HLO.<AXIS>			Stops the sequence by clearing bit#8 (seq_on) of status drive 1 (SD1) and the motor movement using an infinite deceleration and performs a power off			
HLT.<AXIS>			Stops the sequence by clearing bit#8 (seq_on) of status drive 1 (SD1) and the motor movement using an infinite deceleration			
IEQ.<AXIS>=<P1>,<P2>			Jumps to the label specified by <P2> if <P1> is equal to the accumulator (XAC). Sets bit#8 (seq_on) of status drive 1 (SD1)			
IGE.<AXIS>=<P1>,<P2>			Jumps to the label specified by <P2> if <P1> is greater or equal to the accumulator (XAC). Sets bit#8 (seq_on) of status drive 1 (SD1)			
IGT.<AXIS>=<P1>,<P2>			Jumps to the label specified by <P2> if <P1> is greater than the accumulator (XAC). Sets bit#8 (seq_on) of status drive 1 (SD1)			
ILE.<AXIS>=<P1>,<P2>			Jumps to the label specified by <P2> if <P1> is lower or equal to the accumulator (XAC). Sets bit#8 (seq_on) of status drive 1 (SD1)			
ILT.<AXIS>=<P1>,<P2>			Jumps to the label specified by <P2> if <P1> is less than the accumulator (XAC). Sets bit#8 (seq_on) of status drive 1 (SD1)			
IND.<AXIS>			Starts a homing sequence			

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
INE.<AXIS>=<P1>,<P2>			Jumps to the label specified by <P2> if <P1> is different from the accumulator (XAC). Sets bit#8 (seq_on) of status drive 1 (SD1)			
INI.<AXIS>			Starts the phasing procedure (K90)			
ITP.<AXIS>=<P1>			Sets the controller in interpolation mode reference if <P1> is equal to 1			
	0		Interpolation mode disabled			
	1		Interpolation mode at every sti interrupt. Takes jerk time (IJT) and encoder scaling/mapping into account			
	2		Interpolation mode on fti interrupt. Does not take jerk time (IJT) and encoder scaling/mapping into account			
JBC.<AXIS>=<P1>,<P2>			Jumps to the specified label if a bit in the accumulator is cleared. The bit number is specified by <P1> and the label to jump to by <P2>. Sets bit#8 (seq_on) of status drive 1 (SD1)			
JBS.<AXIS>=<P1>,<P2>			Jumps to the specified label if a bit in the accumulator is set. The bit number is specified by <P1> and the label to jump to by <P2>. Sets bit #8 (seq_on) of status drive 1 (SD1)			
JEQ.<AXIS>=<P1>			Jumps to the label specified by <P1> if the operands of the last TST function were equal. If it jumps, it sets bit#8 (seq_on) of status drive 1 (SD1)			
JGT.<AXIS>=<P1>			Jumps to the label specified by <P1> if the first operand of the last TST function was bigger than the second one. If it jumps, it sets bit#8 (seq_on) of status drive 1 (SD1)			
JLT.<AXIS>=<P1>			Jumps to the label specified by <P1> if the first operand of the last TST function was lower than the second one. If it jumps, it sets bit#8 (seq_on) of status drive 1 (SD1)			
JMP.<AXIS>=<P1>			Absolute jump to the specified label; it is EXE instruction. Sets bit #8 (seq_on) of status drive 1 (SD1)			
JNE.<AXIS>=<P1>			Jumps to the label specified by <P1> if the operands of the last TST function were different. If it jumps, it sets bit#8 (seq_on) of status drive 1 (SD1)			
:<label>.<AXIS>			Defines a label. Syntax is ":" followed by label #. Example label 10 of axis 1 is :10.1. There are two special labels :79 is an autostart label. The controller runs sequence from :79 if present; :80 if controller enters in error mode, it goes on from :80			
NEW.<AXIS>=<P1>			Restores default current parameters (K register) and clears sequence according to <P1>			

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
	0		Clears sequence in ram memory and sets default K value in ram memory			
	1		Clears sequence in ram memory			
	2		Sets default K value in ram memory			
POP.<AXIS>			Clears all levels of the call stack. Every time a CAL command is executed, or the sequence goes to label 80 (error process label), the sequence line is stacked. It is possible to stack 256 times before overflow			
PWR.<AXIS>=<P1>			Power on if <P1> = 1 or power off if <P1> = 0. The first power on does an phasing according to K90. A power off followed by a power on resets the errors			
REI.<AXIS>			Performs a RIE followed by a RET in one time (cannot be interrupted)			
RES.<AXIS>=<P1>			Restores the saved user variable (X register), parameters (K register), sequences (S register), look-up table (L register), trigger (E register), real-time interrupt (R register), float (F register) and axis number in ram memory according to <P1>			
	0		Restores sequence S, user look-up table L, user variable X, parameters K, trigger E, real-time interrupt R, float F and axis number from flash to ram memory if all the switches of the DIP switch are set to 1			
	1		Restores sequence S and user look-up table L from flash to ram memory			
	2		Restores user variable X, parameters K, trigger E, real-time interrupt R, float F and axis number from flash to ram memory if all the switches of the DIP switch are set to 1			
RET.<AXIS>			Returns from subroutine, the accumulator (XAC) is restored			
RID.<AXIS>			Disables real-time functions. K190.# = 0 has the same effect			
RIE.<AXIS>			Enables real-time functions. K190.# = 1 has the same effect			
RSD.<AXIS>=<P1>			Puts the controller in hard reset if <P1> = 255			
RST.<AXIS>			Resets the error flags of the controller (bit#10 of SD1). If this command is executed in the error label routine (:80), the pending error bit is cleared (bit#0 of SD2)			
SAV.<AXIS>=<P1>			Saves the current user variable (X register), parameters (K register), sequences (S register), look-up table (L register), trigger (E register), real-time interrupt (R register), float (F register) and axis number in flash memory in function of <P1>			

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
	0		Saves sequence (S register), user look-up table (L register), user variable (X register), parameters (K register), trigger (E register), real-time interrupt (R register), float (F register) and axis number in flash memory			
	1		Saves sequence (S register) and user look-up table (L register) in flash memory			
	2		Saves user variable (X register), parameter (K register), trigger (E register), real-time interrupt (R register), float (F register) and axis number in flash memory			
SET.<AXIS>=<P1>			Sets the current user position to the specified value (first parameter of SET command). Affects user units only (M6 + M7)			
SLS.<AXIS>			Searches limit stroke according to K145 (search limit stroke mode) and returns the limit position in M36 and M37. K47 is taken into account by SLS command but not in M36 and M37 setting			
STA.<AXIS>=<P1>,<P2>			Starts a movement by using the movement parameters of sub-index (depth 1 to 3) specified by <P1>, and <P2> is a mask of movement parameter to be loaded (0 means all movements parameters are taken into account)			
				1	0	Gets target position (POS=K210) from the specified index. Also possible during a trapezoïdale movement if MMC=1
				2	1	Gets profile velocity (SPD=K211) from the specified index. Also possible during a trapezoïdale movement if MMC=1
				4	2	Gets profile acceleration (ACC=K212) from the specified index. Also possible during a trapezoïdale movement if MMC=1
				8	3	Gets jerk time (JRT=K213) from the specified index
				64	6	Gets profile type (MMD=K202) from the specified index
				128	7	Gets user look-up table number (LTN=K203) and the look-up table number selection mode (K207) from the specified index
				256	8	Gets look-up table time (LTI=K204) from the specified index
				512	9	Gets user amplitude for look-up table (K208) in mode K207=1

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
				1024	10	Gets rotary movement type selection (K209) only when the motor is not moving (bit#4 of SD1 at 0)
				2048	11	Gets K230 from the specified index
				4096	12	Gets K229 from the specified index
STE.<AXIS>=<P1>			Performs a position step (infinite acceleration) to the specified (absolute) position			
STE.<AXIS>+=<P1>			Performs a position step (infinite acceleration) of the specified size in the positive direction			
STE.<AXIS>-=<P1>			Performs a position step (infinite acceleration) of the specified size in the negative direction			
STI.<AXIS>=<P1>,<P2>			Synchronous start (sti) of a movement on an input by using the movement parameters of the sub-indexes specified by <P1>, <P2> is a mask to enable movement variable to load (0 means all movements parameters are taken into account)			
				1	0	Gets target position (POS=K210) from the specified index. Also possible during a trapezoïdale movement if MMC=1
				2	1	Gets profile velocity (SPD=K211) from the specified index. Also possible during a trapezoïdale movement if MMC=1
				4	2	Gets profile acceleration (ACC=K212) from the specified index. Also possible during a trapezoïdale movement if MMC=1
				8	3	Gets jerk time (JRT=K213) from the specified index
				64	6	Gets profile type (MMD=K202) from the specified index
				128	7	Gets user look-up table number (LTN=K203) and the look-up table number selection mode (K207) from the specified index
				256	8	Gets look-up table time (LTI=K204) from the specified index
				512	9	Gets user amplitude for look-up table (K208) in mode K207=1
				1024	10	Gets rotary movement type selection (K209) only when the motor is not moving (bit#4 of SD1 at 0)
				2048	11	Gets K230 from the specified index

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
				4096	12	Gets K229 from the specified index
STP.<AXIS>			Stops the current motor movement with an infinite deceleration but does not stop the sequence			
TCL.<AXIS>			Clears the trigger table			
TST.<AXIS>=<P1>,<P2>			Tests 2 registers			
WBC.<AXIS>=<P1>,<P2>			Waits until the specified bits (which have their mask value included in <P2>) of the specified register (<P1>) are all cleared. Sets bit#7 (wait) of status drive 1 (SD1) as long as the condition is not true and then clears it			
WBS.<AXIS>=<P1>,<P2>			Waits until the specified bits (which have their mask value included in <P2>) of the specified register <P1> are all set. Sets bit#7 (wait) of status drive 1 (SD1) as long as the condition is not true and then clears it			
WPG.<AXIS>=<P1>,<P2>			Waits until <P1> becomes greater than <P2>. Sets bit#7 (wait) of status drive 1 (SD1) as long as the condition is not true and then clears it			
WPL.<AXIS>=<P1>,<P2>			Waits until <P1> becomes lower than <P2>. Sets bit#7 (wait) of status drive 1 (SD1) as long as the condition is not true and then clears it			
WSG.<AXIS>=<P1>,<P2>			Waits until <P1> becomes greater than <P2> (signed). Sets bit#7 (wait) of status drive 1 (SD1) as long as the condition is not true and then clears it. The test is signed			
WSL.<AXIS>=<P1>,<P2>			Waits until <P1> becomes lower than <P2> (signed). Sets bit#7 (wait) of status drive 1 (SD1) as long as the condition is not true and then clears it. The test is signed			
WTM.<AXIS>			Waits for the end of the movement. Sets bit #7 (wait) of status drive 1 (SD1) during all the movements and clears it at the end of movement			
WTP.<AXIS>=<P1>			Waits for the specified position. Sets bit #7 (wait) of status drive 1 (SD1) all the time it does not reach this position and clears it then			
WTT.<AXIS>=<P1>			Waits for the specified amount of time (sti). Sets bit #7 (wait) of status drive 1 (SD1) all the time it waits and clears it then			
WTW.<AXIS>=<P1>			Waits for the end of movement, when the real position (upi) is in a window of position (+/- K39) during a time (k38 in sti) around the target			
	0		Waits for the bit in window (bit#5 of SD1) to be at 1. If the motor is not moving (bit#4 of SD1 at 0) when this command is executed, the controller does not wait			

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
	1		Waits for the bit in window (bit#5 of SD1) to be at 1. It does not take into account if the motor is moving or not			
	2		Clears the counter defined by K38 to wait the time to be in the window when this command is executed and clears the bit in window (bit#5 of SD1). It does not take into account if the theoretical trajectory is finished or not			
XAC.<AXIS>=<P1>+<P2>			Addition of <P1> and <P2> and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 + M7.1			
XAC.<AXIS>=<P1>&<P2>			Logical AND between each bit of <P1> and <P2> and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 & M7.1			
XAC.<AXIS>=<P1>&~<P2>			Logical NOT AND between each bit of <P1> and <P2> and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 &~ M7.1			
XAC.<AXIS>=<P1>/<P2>			Division of <P1> by <P2> and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 / M7.1			
XAC.<AXIS>=<P1>*<P2>			Multiplication of <P1> by <P2> and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 * M7.1			
XAC.<AXIS>=<P1>¦<P2>			Logical OR between each bit of <P1> and <P2> and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 ¦ M7.1			
XAC.<AXIS>=<P1>¦~<P2>			Logical NOT OR between each bit of <P1> and <P2> and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 ¦~ M7.1			
XAC.<AXIS>=<P1><<<P2>			Shifts <P1> of <P2> bit to the left and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 << M7.1			
XAC.<AXIS>=<P1>>><P2>			Shifts <P1> of <P2> bit to the right and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 >> M7.1			
XAC.<AXIS>=<P1>-<P2>			Substracts <P2> of <P1> and puts the result in the accumulator (XAC). Example: XAC.1 = X1.1 - M7.1			
<REG>.<AXIS>=<P2>			Affects the value of <P2> to <REG>. This is available with all kinds of register excepted monitoring register (M)			
<REG>.<AXIS>+=<P2>			Adds <P2> to <REG> and puts the result in <REG>. The first operand can only be a K parameter or a X variable, the second one can be an immediate value or any other register. Example: X1.1 += M7.1			

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
<REG>.<AXIS>&=<P2>			Logical AND between each bit of <REG> and <P2> and puts the result in <REG>. The first operand can only be a K parameter or a X variable, the second one can be an immediate value or any other register. Example: X1.1 &= K30.1			
<REG>.<AXIS>&~=<P2>			Logical NOT AND between each bit of <REG> and <P2> and puts the result in <REG>. The first operand can only be a K parameter or a X variable, the second one can be an immediate value or any other register. Example: X1.1 &~= K30.1			
<P1>.<AXIS>=<P2>			Executes the operation: <P1> = conversion float_to_long or long_to_float <P2>. This is available with all kind of registers (<P1> could not be a monitoring register). One of the register must be a float. The syntax is for example F2.1=X3.1, or X3.1=F2.1			
<REG>.<AXIS>/=<P2>			Divides <REG> by <P2> and puts the result in <REG>. The first operand can only be a K parameter or a X variable, the second one can be an immediate value or any other register. Example: X1.1 /= K30.1			
<REG>.<AXIS>*=<P2>			Multiplies <REG> by <P2> and puts the result in <REG>. The first operand can only be a K parameter or a X variable, the second one can be an immediate value or any other register. Example: X1.1 *= K30.1			
<REG>.<AXIS>~=<P2>			Inverts <P2> and puts the result in <REG>. The first operand can only be a parameter K or a variable X, the second one can be an immediate value or any other register. Example: X1.1 ~= M7.1			
<REG>.<AXIS>¦=<P2>			Logical OR between each bit of <REG> and <P2>, puts the result in <REG>. The first operand can only be a K parameter or a X variable, the second one can be an immediate value or any other register. Example: X1.1 ¦= K30.1			
<REG>.<AXIS>¦~=<P2>			Logical NOT OR between each bit of <REG> and <P2> and puts the result in <REG>. The first operand can only be a K parameter or a X variable, the second one can be an immediate value or any other register. Example: X1.1¦~= K30.1			
<REG>.<AXIS><<=<P2>			Shifts <REG> of <P2> bit to the left and puts the result in <REG>. The first operand can only be a parameter K or a variable X, the second one can be an immediate value or any other register. Example: X1.1 <<= M7.1			

Syntax	Val <P1>	Bit # <1>	Comment for commands and parameter <P1>	Val <P2>	Bit # <P2>	Comment for parameter <P2>
<REG>.<AXIS>>>=<P2>			Shifts <REG> of <P2> bit to the right and puts the result in <REG>. The first operand can only be a parameter K or a variable X, the second one can be an immediate value or any other register. Example: X1.1 >>= M7.1			
<REG>.<AXIS>-=<P2>			Substracts <P2> of <REG> and puts the result in <REG>. The first operand can only be a K parameter or a X variable, the second one can be an immediate value or any other register. Example: X1.1 -= K30.1			
ZCL.<AXIS>=<P1>,<P2>			Sets channel (0:first param of ZTR or 1:second param of ZTR) and trigger level (when second param of ZIM is equal to 3 or 4) for trace register (T) acquisition. See EBL2 User's Manual			
ZFT.<AXIS>			Enables start trigger acquisition. It does not take trigger condition into account. When the controller has finished the acquisition, it clears the trace busy bit of SD1 (bit#11). See EBL2 User's Manual			
ZIM.<AXIS>=<P1>,<P2>			Sets period and trigger mode for trace register (T) acquisition. See EBL2 User's Manual			
ZTE.<AXIS>=<P1>			Starts trigger acquisition if first parameter is equal to 1. It directly sets the bit #11 of SD1 if bit#0 of K120 is equal to 0 and only when the condition is true if it is equal to 1. See EBL2 User's Manual			
ZTR.<AXIS>=<P1>,<P2>			Starts trigger acquisition for trace register (T) defined in the two parameters. See EBL2 User's Manual			

16. Parameters K

16.1 Parameters K for DSC2P and DSC2V

DSC2P and DSC2V parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
K1				Position loop proportional gain	100	0	2147483647
K2				Position loop speed feedback gain	20	0	2147483647
K3				Position loop force feedback gain	0	0	2147483647
K4				Position loop integrator gain	0	0	2147483647
K5				Position loop anti-windup gain	10	0	2147483647
K6				Position loop integrator limitation	1073741823	0	2147483647
K7				Position loop integrator mode	0	0	2
		0		Position loop integrator gain (K4) always on			
		2		Position loop integrator gain (K4) always off			
K8				Position loop speed filter	0	0	511
K9				Force reference filter on the position regulator output	0	0	511
K11				Speed smooth filter for TTL encoder	0	0	256
K20				Position loop speed feedforward gain	0	0	2147483647
K21				Position loop acceleration feedforward gain	0	0	2147483647
K23				Commutation phase advance according to the speed	0	0	65535
K27				Maximum position range limit for rotary movement. Depth 0 for primary encoder and depth 1 for secondary encoder	1	0	2000000000
K30				Absolute tracking error limit. When the tracking error is greater than K30, the controller generates the error M64=23	10000000	0	1073741823
K31				Absolute velocity limit. When the velocity is greater than K31, the controller generates an overspeed error M64=24	100000000	0	2147483647
K32				Limit switch and home switch inversion. Activates the dynamic brake	0	0	63
		1	0	Enables the use of the limit switches			
		2	1	Inverts home switch			
		4	2	Inverts limit switches from the encoder			
		8	3	Inverts limit switches			
		16	4	Enables the use of the dynamic braking controlled by the transistors			
		32	5	Enables limitation of K60/K31 according to DIN. K178:2 defines the mask of the DIN that should be at 1 to limit K60 and K31. K179:2 defines the mask of the DIN that should be at 0 to limit K60 and K31. K60:3 and K31:3 are the limitation			
K33				Enables input mode	125	0	125
		0		Enabled signal is necessary to power on the controller on DIN1. In this case this input must be at 1 when a PWR.#=1 command is executed. If this input is cleared, the controller generates an error (M64=26)			
		125		Enabled signal not used (DIN1 is not taken into account). PWR.#=1 command switches the power on the motor			
K34				Minimum software position limit	0	-2147483648	2147483647

DSC2P and DSC2V parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
K35				Maximum software position limit	0	-2147483648	2147483647
K36				Enables position limit (K34, K35) generating an error depending on the value	0	0	7
		1	0	Use of K34 and K35 as limit on the target the motor can reach. Used with K61=1			
		2	1	Use of K34 and K35 as limit on the actual position of the motor. If the motor reaches these limits, it generates an error (M64=65). These limits are tested on every sti but only if a homing has been previously done. Used for all values of K61			
		4	2	Use of K34 and K35 as limit on the target to generate an error (M64=66) when the movement starts. Used with K61=1			
K37				Mask of the digital output (DOUT) that must be cleared when the controller is in error. When the controller is not in error any more, the digital outputs have the DOUT value.	0	0	15
K38				Duration of the window (used with WTW command)	0	0	393210
K39				Position range of the window (used with WTW command)	0	0	1073741823
K40				Homing mode	8	0	45
		0		Homing on mechanical end stop			
		1	0	Homing with a negative movement			
		2		Homing on home switch			
		4		Homing on limit switch			
		6		Homing on home switch with detection of limit switch			
		8		Homing on a single index			
		10		Homing on a single index with detection of limit switch			
		12		Homing on a multi-index			
		14		Homing on a multi-index with detection of limit switch			
		16		Homing on a single index with DIN2 at 1			
		18		Homing on a single index with DIN2 at 1 with detection of limit switch			
		20		Homing on multi-index with a defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
		22		Immediate homing at the current position (It is used to take K53 at a known position into account)			
		24		Single index homing with defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
		26		Homing on single index. If the motor reaches a mechanical end stop before an index, it changes its movement direction. If after the mechanical end stop it does not meet an index before a stroke defined by K46, the controller generates an error (M64=62)			
		34		Homing on a single index with detection of limit switch coming from the encoder			
		36		Homing on home switch. If the motor reaches a mechanical end stop before home switch, it changes its movement direction. If after the mechanical end stop it does not meet home switch before stroke defined by K46, the controller generates an error (M64=62)			

DSC2P and DSC2V parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		38		Homing on single index after having found mechanical end stop with defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
K41				Homing speed	2000000	1	2147483647
K42				Homing acceleration	1000000	256	2147483647
K43				Homing tracking limit for mechanical end stop detection	10000000	0	2147483647
K44				Homing force limit for mechanical end stop detection	4096	0	32767
K45				Offset on absolute position	0	-2147483648	2147483647
K46				Stroke for K40 = 20, 21, 24, 25 homing mode	0	0	2147483647
K47				Movement to go out of a limit switch or mechanical end stop at the end of the homing	0	0	2147483647
K48				Movement to go out of an index or home switch if the motor is on the top of it when starting the homing	0	0	2147483647
K50				Set point calculator shift value: 1dpi = 2^K50 * 1upi	0	0	8
K52				Enables fine phase adjustment (takes K53 into account) after homing	0	0	1
K53				Motor commutation phase adjustment after homing (is taken into account only if K52=1)	0	0	2048
K54				Pairs of pole of the motor (=1 for linear motor)	1	1	2147483647
K55				Motor commutation encoder: number of dpi per revolution for rotary motor or number of dpi per magnetic period for linear motor	0	0	2147483647
K56				Motor commutation phase inversion enabled	0	0	1
K58				Limit switch mode	0	0	2
K58		0		Limit switch mode DIN9 and DIN10			
		1		Limit switch mode L1/L2			
		2		Limit switch mode L/H			
K60				Theoretical software force/torque limit (regulator output)	20000	0	31000
K61				Reference mode	1	0	39
		0		Force/torque reference mode. The reference value is defined by K220 to K224 (32767 is equivalent to the current defined by M82)			
		1		Standard position profile mode			
		3		Controller controlled by a speed reference defined by K220 to K224			
		4		Controller controlled by a position reference defined by K220 to K224			
		36		Controller controlled by a position reference defined by K220 to K224. After a power on, it takes into account the actual motor position as reference			
K66				Display mode	1	1	64
		1		Displays normal informations			
		2		Displays temperature of the controller			
		4		Displays analog encoder amplitude and index position			
		8		Displays sequence line number			
		16		Displays optional board message			
		32		Displays DC power voltage (Vpower) [V]			
K68				Inverts positive/negative way	0	0	7
		1	0	Inverts analog 1Vptp encoder			

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSC2P and DSC2V parameters			2	1	Inverts TTL encoder			
			4	2	Inverts force reference from MACRO			
	K69				Encoder shift value for TTL encoder (K79=1)	0	0	12
	K70				Analog encoder sine offset	0	-1024	1024
	K71				Analog encoder cosine offset	0	-1024	1024
	K72				Analog encoder sine factor	32767	16384	32767
	K73				Analog encoder cosine factor	32767	16384	32767
	K75				Distance between two indexes for multi-indexes encoder	0	0	2147483647
	K77				Encoder interpolation shift value for analog encoder	0	0	12
	K79				Encoder type selection	0	0	120
			0		Principal: 1Vptp, Secondary: TTL			
			1		Principal: TTL, Secondary: 1Vptp			
			4		Principal: EnDat, Secondary: TTL			
			7		Principal: TTL, Secondary: EnDat			
			100		Mode MACRO: Principal: 1Vptp, Secondary: TTL			
			101		Mode MACRO: Principal: TTL, Secondary: 1Vptp			
			104		Mode MACRO: Principal: EnDat, Secondary: TTL			
	K80				Current loop proportional gain	500	0	2147483647
	K81				Current loop integrator gain	0	0	2147483647
	K82				Current loop output filter	0	0	511
	K83				Current loop software overcurrent limit	16000	0	32000
	K84				Current loop i2t rms current limit	510	0	8192
	K85				Current loop i2t time limit	20966400	0	2147483647
	K87				Synchronization mode and frequency of the controller	0	0	16777215
	K88				Position loop frequency (MACRO mode only)	24	0	32
	K89				Motor phase number and PWM type selection	30	10	41
			10		1 phase motor, PWM at 24KHz (for DSC2P only)			
			11		1 phase motor, PWM at 12KHz			
			14		1 phase motor, PWM at 6KHz (for DSC2V only)			
			20		2 phase motor, PWM at 24KHz (for DSC2P only)			
			21		2 phase motor, PWM at 12KHz			
			24		2 phase motor, PWM at 6KHz (for DSC2V only)			
			30		3 phase motor, PWM at 24KHz (for DSC2P only)			
			31		3 phase motor, PWM at 12KHz			
			34		3 phase motor, PWM at 6KHz (for DSC2V only)			
	K90				Phasing mode. Depth 1: phasing mode for command AUT.x=8	2	0	6
			0		No phasing (with 1-phase)			
			1		Phasing with current pulses (with ironcore motors only)			
			2		Phasing by sending constant current to the motor (ironcore and ironless motors)			
			3		Phasing with digital Hall sensors until the index is found then commutation by position encoder. The value stored in K53 is used if K52 = 1			

DSC2P and DSC2V parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		4		Phasing with digital Hall sensors until the first edge of signal then commutation by position encoder. When index is found the value stored in K53 is used if K52 = 1			
		5		Phasing and commutation with digital Hall sensors only			
		6		Small movements phasing			
K91				Phasing pulse level (with K90=1 or K90=6)	12000	0	28000
K92				Phasing constant current level (with K90=2)	12000	0	28000
K93				Phasing final phase (with K90=2)	1024	0	2048
K94				Phasing time process (with K90=2)	5000	0	25000
K97				Phasing initial phase (with K90=2)	512	0	2048
K98				Phasing power voltage rate in percent with K90=1. This is useful for slow inductance motor	100	25	100
K100				JC4 output mode selection and SPD, ACC and MMD after IND selection mode	0	0	23
		2	1	Sets SPD, ACC and MMD parameter that was set before IND command at the end of the homing process			
K101				Phasing time process (with K90=6)	1000	0	25000
K102				Enables or disables the TEB Real-Time Monitoring (RTM)	0	0	3
		1	0	Disables slave to master RTM			
		2	1	Enables slave to slave RTM			
K104				First advanced filter coefficient for term yk-1	0	-2147483648	2147483647
K105				First advanced filter coefficient for term yk-2	0	-2147483648	2147483647
K106				First advanced filter coefficient for term xk	1073741824	-2147483648	2147483647
K107				First advanced filter coefficient for term xk-1	0	-2147483648	2147483647
K108				First advanced filter coefficient for term xk-2	0	-2147483648	2147483647
K114				Second advanced filter coefficient for term yk-1	0	-2147483648	2147483647
K115				Second advanced filter coefficient for term yk-2	0	-2147483648	2147483647
K116				Second advanced filter coefficient for term xk	1073741824	-2147483648	2147483647
K117				Second advanced filter coefficient for term xk-1	0	-2147483648	2147483647
K118				Second advanced filter coefficient for term xk-2	0	-2147483648	2147483647
K120				Enables trace synchronization mode. See EBL2 manual	0	0	1
K121				Trace register selection type (depth 0 for trace 0; depth 1 for trace 1,...). See EBL2 manual.	3	0	8
		1		Users variable register (X)			
		2		Parameter register (K)			
		3		Monitoring register (M)			
		8		LKT register (L)			
K122				Trace register number and depth selection (bit#0-15 for register number, bit#16-23 for depth). See EBL2 manual	0	-2147483648	2147483647
K123				Trace time in sti between two trace acquisition points. See EBL2 manual	0	0	65535
K125				Trace trigger mode. See EBL2 manual	0	0	5
		0		No trigger			
		1		Start of movement			
		2		End of movement			
		3		Trigger at a position selected by K127			

DSC2P and DSC2V parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		4		Trigger at a register value defined by K127. K126=0 means that register defined in depth 0 is chosen for trigger, and K126=1 for depth 1,…			
		5		Trigger never starts; only ZFT command can start trigger acquisition			
K126				The value indicates that the register selected for the trigger condition is defined in depth of K121 and K122 parameters. See EBL2 manual	0	0	3
K127				Value of the register selected by K121, K122 and K126 parameters for acquisition start. See EBL2 manual	0	-2147483648	2147483647
K128				Number of points for the trace. See EBL2 manual	1024	0	1024
K133				AUT command definition mode	0	0	1
		0		Principle for finding the fine phase adjustment 0:old principle 1:new principle			
K135				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K136				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K137				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K138				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K139				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K140				Mask for fuse control	0	0	1
		1	0	Disables test of F3 fuse (from DSC2Pxxx-xxxC-xxxx)			
K141				Enables test of motor overtemperature protection connected to TSD signal (see Hardware Manual), and/or time-out TEB error, and/or enables the checking of the error for the secondary encoder	0	0	7
		1	0	Enables test of motor overtemperature protection connected to TSD signal (see Hardware Manual)			
		2	1	Enables time-out TEB error test			
		4	2	Enables the checking of the error for the secondary encoder			
K145				Searches limit stroke (SLS) mode	0	0	3
		0		Positive movement to search mechanical end stop			
		1		Negative movement to search mechanical end stop			
		2		Positive movement to search mechanical limit switch			
		3		Negative movement to search mechanical limit switch			
K146				Limit of Vpower to activate undervoltage warning [V/100]. When Vpower[V] * 100 is lower than K146, the controller generates an undervoltage warning (M66=10). If K146=0, the limit detection is disabled	0	0	100000
K147				Limit of Vpower to activate undervoltage error [V/100]. When Vpower[V] * 100 is lower than K147, the controller generates an undervoltage error (M64=9). If K147=0, the limit detection is disabled	0	0	100000
K148				Limit of Vpower to activate overvoltage warning [V/100]. When Vpower[V] * 100 is greater than K148, the controller generates an overvoltage warning (Set bit#19 of SD1). If K148=0, the limit detection is disabled. For DSC2P the limit is reduced by firmware	70000	0	100000
K149				Limitt of Vpower to activate overvoltage error [V/100]. When Vpower[V] * 100 is greater than K149, the controller generates an overvoltage error (M64=6). If K149=0, the limit detection is disabled. For DSC2P the limit is reduced by firmware	80000	0	100000

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSC2P and DSC2V parameters	K150	MST			Optional board source register type. It has four depths to monitor four registers per sti (slow time interrupt)	0	0	3
			1		Source type is a user's variable X			
			2		Source type is a parameter K			
			3		Source type is a monitoring M			
	K151	MSN			Optional board source register index	0	0	255
	K154	MOF			Optional board source register offset	0	-2147483648	2147483647
	K155	MAM			Optional board source register gain	16777216	-2147483648	2147483647
	K156	DAO			Optional board analog offset. It has 4 depths for DSO-HIO analog output, each depth corresponds to the depths of K173	0	-32768	32767
	K157	DAA			Optional board analog gain. It has 4 depths, for DSO-HIO analog output, each depth corresponds to the depths of K173	32767	-32768	32767
	K159				Inverts the digital inputs for position capture or with DSO-MAC optional board. If K33=0 (DIN1 is a safety input), DIN1 cannot be inverted	0	0	783
	K160				Syncronization input mask that will be tested for synchronized start movement on digital input (STI command)	0	0	1023
	K161				Syncronization input mask value (used with STI command)	0	0	1023
	K162				Syncronization output mask (used with STI command)	0	0	15
	K163				Syncronization output value (used with STI command)	0	0	15
	K164				Syncronization time-out (used with STI command)	0	0	600000
	K165				Enables scaling/mapping correction	0	0	4
			0		Mapping and scaling deactivated			
			1		Mapping activated			
			2		Scaling activated			
			3		Mapping and scaling activated			
			4		Rotary mapping activated			
	K166				Position (dpi) where the mapping correction starts	0	-2147483648	2147483647
	K167				Length (dpi) where the mapping correction is active	0	0	2147483647
	K168				Position (dpi) where the scaling correction is null	0	-2147483648	2147483647
	K169				This value/100000 multiplied by the position M8 gives the scaling correction	0	-50000	50000
	K170	MDE			Selects the manager mode (slave, u-master,...). This value is taken into account only at the power on of the controller	0	0	2
			0		Select slave mode (default)			
			1		Select u-master mode			
	K171	DOUT			Activates or deactivates the digital outputs of the controller (DOUT1 to DOUT4). K37 (if different from 0) changes the value of the digital outputs in case of error, but not the value of DOUT (K171).	0	0	15
	K172	XDOUT			Optional board digital outputs (8 for DSO-HIO)	0	0	255
	K173	XAOUT			Optional board analog outputs of the DSO-HIO (8191 = -20V between XAOUT+ and XAOUT-, -8192 = +20V between XAOUT+ and XAOUT-). It has 4 depths that corresponds to the 4 analog outputs of the DSO-HIO	0	-8192	8191

DSC2P and DSC2V parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
K176				Enables the fast trigger mode (available in macro mode only)	0	0	3
		1	0	Enables fast trigger mode. Digital output masked by K188 are set or cleared according to bit#1 of K182 when M1 is greater or lower than K189			
		2	1	If this bit is set, digital outputs masked by K188 are cleared when M1 is greater than K189, and set when M1 is lower than K189. If this bit is cleared, the digital outputs are inverted in comparison with the previous case.			
K177				User word of M63 (bit#0 to 15) and user byte of SD2 (bit#8 to 15)	0	0	65535
K178				Depth 0: rising edge mask on digital input for position capture. Depth 1: mask on digital input. Depth 2: mask on digital input that must be at 1 for K60/K31 limitation	0	0	895
K179				Depth 0: falling edge mask on digital input for position capture. Depth 1: mask on digital input that must be at 0 for BRT command. Depth 2: mask on digital input that must be at 0 for K60/K31 limitation	0	0	895
K182				Enables position capture on input according to K178 and K179. Writing a 1 in K182 enables the capture and resets bit#2 of SD2 and bit#30 of M63. Writing a 0 stops the process	0	0	1
K183				Digital output mask on optional board for trigger	0	0	255
K184				Mask on user word of M63 (bit#0 to 15) and user byte of SD2 (bit#8 to 15) for trigger	0	0	65535
K185	TMK			Digital output mask for trigger	0	0	255
K186	TRS			Trigger mapping number. -1 deactivates the trigger	-1	-1	191
K187	TNB			Trigger mapping size	0	0	192
K188				Mask on the digital output in fast trigger mode. Available in Macro mode with K176 different from 0	0	0	15
K189				Position (dpi) that is compared to M1 to activate digital output in fast trigger mode (K176 different from 0). K188 is the mask of the digital outputs and K176 enables the selection to modify them. Avalaible in Macro mode only	0	-2147483648	2147483647
K190				Global enable of real-time function	0	0	1
K191				List of valid real-time lines in the RTI table	0	0	255
K192				List of enable real-time lines in the RTI table	0	0	255
K193				List of pending real-time lines in the RTI table	0	0	255
K195				Enables the selection of ETEL-Bus-Lite2 baudrate. It is taken into account only at the first switch on. If you want to change it, you have to set the value in K195 and then save it into the controller with SAV.#=2. Then switch off and on the controller	0	0	115200
		0		ETEL-Bus-Lite2 at 115200 bauds (default value)			
		9600		ETEL-Bus-Lite2 at 9600 bauds			
		19200		ETEL-Bus-Lite2 at 19200 bauds			
		38400		ETEL-Bus-Lite2 at 38400 bauds			
		57600		ETEL-Bus-Lite2 at 57600 bauds			
		115200		ETEL-Bus-Lite2 at 115200 bauds			
K198				The register number is given by K198 when it is equal to Y or y (ex: Ky.1=...). See EBL2 manual	0	0	65535
K201	MMC			Concatenated movement selection	0	0	3

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSC2P and DSC2V parameters			0		Concatenated movement disabled			
			1		Concatenated movement enabled for MMD=1, 17 and 24			
			2		Continuous back and forth movement enabled for MMD = 10, 26, 3 and 19 movement (could be stopped by POS, BRK, HLT, HLB and HLO commands)			
			3		One back and forth MMD = 10, 26, 3 and 19 movement enabled for look-up table movement			
	K202	MMD			Movement type	1	1	32
			1		S-curve (jerk time) linear movement			
			3		Selects a predefined linear movement according to K230 value and executes the trajectory in a time given by K229			
			10		Look-up table linear movement			
			17		S-curve (jerk time) rotary movement			
			19		Selects a predefined rotary movement according to K230 value and executes the trajectory in a time given by K229			
			24		Infinite rotary movement			
			26		LKT rotary movement			
	K203	LTN			Look-up table number movement	0	0	7
	K204	LTI			Time to execute a look-up table movement	10000	4	1500000
	K205	CAM			Came value (in percent). Stretches the user time scale	100	1	100
	K206				Brake deceleration (with BRK and HLB command)	1000000	256	2147483647
	K207				LKT mode selection	0	0	1
			0		LKT movement running to target defined by POS command			
			1		LKT movement with same starting and end point. K208 defines the amplitude of the movement			
	K208				Maximum stroke for LKT movement with K207=1 and for MMD=10 & 26	0	-2147483648	2147483647
	K209				Rotary movement type selection	0	0	2
			0		Movement always positive			
			1		Movement always negative			
			2		Movement minimum distance			
	K210	POS			Starts movement (only depth 0) and gives the target position	0	-2147483648	2147483647
	K211	SPD			Absolute maximum speed	2000000	1	2147483647
	K212	ACC			Absolute maximum acceleration and deceleration	1000000	256	2147483647
	K213	JRT			Jerk time	0	0	500
	K219				TEB real-time slave to master monitoring pointer	0	-2147483648	2147483647
	K220				Control source type	3	1	3
			1		Source type is a user's variable X			
			2		Source type is a parameter K			
			3		Source type is a monitoring M			
	K221				Control source index	51	0	255
	K222				Control source shift factor	0	0	16
	K223				Control source offset	0	-2147483648	2147483647

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSC2P and DSC2V parameters	K224				Control source gain	8388608	-2147483648	2147483647
	K229				Execution time of the movement selected by K230	10000	4	1500000
	K230				Movement type selection for MMD = 3 or 19	0	0	3
			0		Triangular speed movement			
			1		S-curve (full jerk) movement			
			2		Sine modified movement			
			3		Real sine movement			
	K231				Slave to slave RTM register type: depth 0 for slave to slave RTM1 and depth 1 for slave to slave RTM2	3	1	3
			1		Source type is a user's variable X			
			2		Source type is a parameter K			
			3		Source type is a monitoring M			
	K232				Slave to slave RTM index: depth 0 for slave to slave RTM1 and depth 1 for slave to slave RTM2	6	0	255
	K233				Slave to slave RTM destination mask axis	1	-2147483648	2147483647
	K240				Motor type. 0: linear motor, 1:rotary motor. Depth 0 for primary encoder and depth 1 for secondary encoder	0	0	1
	K241				Encoder period in [nm] for linear encoder or number of period for rotary encoder. Depth 0 for primary encoder and depth 1 for secondary encoder	1	-2147483648	2147483647
	K242				Position multiplication factor (is used by the DLL to calculate the position unit with indirect encoder)	1	1	2147483647
	K243				Position division factor (is used by the DLL to calculate the position unit with indirect encoder)	1	1	2147483647

16.2 Parameters K for DSCDP

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP parameters	K1				Position loop proportional gain	100	0	2147483647
	K2				Position loop speed feedback gain	20	0	2147483647
	K3				Position loop force feedback gain	0	0	2147483647
	K4				Position loop integrator gain	0	0	2147483647
	K5				Position loop anti-windup gain	10	0	2147483647
	K6				Position loop integrator limitation	1073741823	0	2147483647
	K7				Position loop integrator mode	0	0	2
			0		Position loop integrator gain (K4) always on			
			2		Position loop integrator gain (K4) always off			
	K8				Position loop speed filter	0	0	511
	K9				Force reference filter on the position regulator output	0	0	511
	K11				Speed smooth filter for TTL encoder	0	0	256
	K20				Position loop speed feedforward gain	0	0	2147483647
	K21				Position loop acceleration feedforward gain	0	0	2147483647
	K23				Commutation phase advance according to the speed	0	0	65535
	K27				Maximum position range limit for rotary movement	1	0	2000000000

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP parameters	K30				Absolute tracking error limit. When the tracking error is greater than K30, the controller generates the error M64=23	10000000	0	1073741823
	K31				Absolute velocity limit. When the velocity is greater than K31, the controller generates an overspeed error M64=24	100000000	0	2147483647
	K32				Limit switch and home switch inversion. Activates the dynamic brake	0	0	63
			1	0	Enables the use of the limit switches			
			2	1	Inverts home switch			
			4	2	Inverts limit switches from the encoder			
			8	3	Inverts limit switches			
			16	4	Enables the use of the dynamic braking controlled by the transistors			
			32	5	Enables limitation of K60/K31 according to DIN. K178:2 defines the mask of the DIN that should be at 1 to limit K60 and K31. K179:2 defines the mask of the DIN that should be at 0 to limit K60 and K31. K60:3 and K31:3 are the limitation			
	K33				Enables input mode	125	0	125
			0		Enabled signal is necessary to power on the controller on DIN1. In this case this input must be at 1 when a PWR.#=1 command is executed. If this input is cleared, the controller generates an error (M64=26)			
			125		Enabled signal not used (DIN1 is not taken into account). PWR.#=1 command switches the power on the motor			
	K34				Minimum software position limit	0	-2147483648	2147483647
	K35				Maximum software position limit	0	-2147483648	2147483647
	K36				Enables position limit (K34, K35) generating an error depending on the value	0	0	7
			1	0	Use of K34 and K35 as limit on the target the motor can reach. Used with K61=1			
			2	1	Use of K34 and K35 as limit on the actual position of the motor. If the motor reaches these limits, it generates an error (M64=65). These limits are tested on every sti but only if a homing has been previously done. Used for all values of K61			
			4	2	Use of K34 and K35 as limit on the target to generate an error (M64=66) when the movement starts. Used with K61=1			
	K37				Mask of the digital output (DOUT) that must be cleared when the controller is in error. When the controller is not in error any more, the digital outputs have the DOUT value.	0	0	3
	K38				Duration of the window (used with WTW command)	0	0	393210
	K39				Position range of the window (used with WTW command)	0	0	1073741823
	K40				Homing mode	8	0	41
			0		Homing on mechanical end stop			
			1	0	Homing with a negative movement			
			2		Homing on home switch			
			4		Homing on limit switch			
			6		Homing on home switch with detection of limit switch			

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP parameters			8		Homing on a single index			
			10		Homing on a single index with detection of limit switch			
			12		Homing on a multi-index			
			14		Homing on a multi-index with detection of limit switch			
			16		Homing on a single index with DIN2 at 1			
			18		Homing on a single index with DIN2 at 1 with detection of limit switch			
			20		Homing on multi-index with a defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
			22		Immediate homing at the current position (It is used to take K53 at a known position into account)			
			24		Single index homing with defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
			26		Homing on single index. If the motor reaches a mechanical end stop before an index, it changes its movement direction. If after the mechanical end stop it does not meet an index before a stroke defined by K46, the controller generates an error (M64=62)			
			34		Homing on a single index with detection of limit switch coming from the encoder			
			36		Homing on home switch. If the motor reaches a mechanical end stop before home switch, it changes its movement direction. If after the mechanical end stop it does not meet home switch before stroke defined by K46, the controller generates an error (M64=62)			
			38		Homing on single index after having found mechanical end stop with defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
	K41				Homing speed	2000000	1	2147483647
	K42				Homing acceleration	1000000	256	2147483647
	K43				Homing tracking limit for mechanical end stop detection	10000000	0	2147483647
	K44				Homing force limit for mechanical end stop detection	4096	0	32767
	K45				Offset on absolute position	0	-2147483648	2147483647
	K46				Stroke for K40 = 20, 21, 24, 25 homing mode	0	0	2147483647
	K47				Movement to go out of a limit switch or mechanical end stop at the end of the homing	0	0	2147483647
	K48				Movement to go out of an index or home switch if the motor is on the top of it when starting the homing	0	0	2147483647
	K50				Set point calculator shift value: 1dpi = 2^K50 * 1upi	0	0	8
	K52				Enables fine phase adjustment (takes K53 into account) after homing	0	0	1
	K53				Motor commutation phase adjustment after homing (is taken into account only if K52=1)	0	0	2048
	K54				Pairs of pole of the motor (=1 for linear motor)	1	1	2147483647
	K55				Motor commutation encoder: number of dpi per revolution for rotary motor or number of dpi per magnetic period for linear motor	0	0	2147483647
	K56				Motor commutation phase inversion enabled	0	0	1
	K58				Limit switch mode	0	0	2
			0		Limit switch mode DIN9 and DIN10			

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP parameters			1		Limit switch mode L1/L2			
			2		Limit switch mode L/H			
	K59				Theoretical software force/torque limit for stepper, when no movement is required	20000	0	31000
	K60				Theoretical software force/torque limit (regulator output)	20000	0	31000
	K61				Reference mode	1	0	39
			0		Force/torque reference mode. The reference value is defined by K220 to K224 (32767 is equivalent to the current defined by M82)			
			1		Standard position profile mode			
			3		Controller controlled by a speed reference defined by K220 to K224			
			4		Controller controlled by a position reference defined by K220 to K224			
			36		Controller controlled by a position reference defined by K220 to K224. After a power on, it takes into account the actual motor position as reference			
	K66				Display mode	1	1	64
			1		Displays normal informations			
			2		Displays temperature of the controller			
			4		Displays analog encoder amplitude and index position			
			8		Displays sequence line number			
			16		Displays optional board message			
	K68				Inverts positive/negative way	0	0	7
			1	0	Inverts analog 1Vptp encoder			
			2	1	Inverts TTL encoder			
			4	2	Inverts force reference from MACRO			
	K69				Encoder shift value for TTL encoder (K79=1)	0	0	12
	K70				Analog encoder sine offset	0	-1024	1024
	K71				Analog encoder cosine offset	0	-1024	1024
	K72				Analog encoder sine factor	32767	16384	32767
	K73				Analog encoder cosine factor	32767	16384	32767
	K75				Distance between two indexes for multi-indexes encoder	0	0	2147483647
	K77				Encoder interpolation shift value for analog encoder	0	0	12
	K79				Encoder type selection	0	0	120
			0		Analog sine/cosine encoder 1Vptp			
			1		TTL encoder			
			4		EnDat encoder			
			20		Stepper in open loop without encoder			
			21		Stepper in open loop with TTL encoder as secondary			
			23		Stepper in open loop with 1Vptp encoder as secondary			
			24		Stepper in open loop with EnDat encoder as secondary			
			100		Mode MACRO: 1Vptp			
			101		Mode MACRO: TTL			
			104		Mode MACRO: EnDat			
	K80				Current loop proportional gain	500	0	2147483647

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP parameters	K81				Current loop integrator gain	0	0	2147483647
	K82				Current loop output filter	0	0	511
	K83				Current loop software overcurrent limit	16000	0	32000
	K84				Current loop i2t rms current limit	510	0	8192
	K85				Current loop i2t time limit	20966400	0	2147483647
	K87				Synchronization mode and frequency of the controller	0	0	16777215
	K88				Position loop frequency (MACRO mode only)	18	0	32
	K89				Motor phase number and PWM type selection	30	10	41
			10		1 phase motor, PWM at 18KHz			
			20		2 phase motor, PWM at 18KHz			
			30		3 phase motor, PWM at 18KHz			
	K90				Phasing mode. Depth 1: phasing mode for command AUT.x=8	2	0	6
			0		No phasing (with 1-phase)			
			1		Phasing with current pulses (with ironcore motors only)			
			2		Phasing by sending constant current to the motor (ironcore and ironless motors)			
			3		Phasing with digital Hall sensors until the index is found then commutation by position encoder. The value stored in K53 is used if K52 = 1			
			4		Phasing with digital Hall sensors until the first edge of signal then commutation by position encoder. When index is found, the value stored in K53 is used if K52 = 1			
			5		Phasing and commutation with digital Hall sensors only			
			6		Small movements phasing			
	K91				Phasing pulse level (with K90=1 or K90=6)	12000	0	28000
	K92				Phasing constant current level (with K90=2)	12000	0	28000
	K93				Phasing final phase (with K90=2)	1024	0	2048
	K94				Phasing time process (with K90=2)	5000	0	25000
	K97				Phasing initial phase (with K90=2)	512	0	2048
	K98				Phasing power voltage rate in percent with K90=1. This is useful for slow inductance motor	100	25	100
	K100				SDP, ACC and MMD after IND selection mode	0	0	26
			2	1	Sets SPD, ACC and MMD parameter that was set before IND command at the end of the homing process			
	K101				Phasing time process (with K90=6)	1000	0	25000
	K102				Enables or disables the TEB Real-Time Monitoring (RTM)	0	0	1
			1	0	Disables slave to master RTM			
	K104				First advanced filter coefficient for term yk-1	0	-2147483648	2147483647
	K105				First advanced filter coefficient for term yk-2	0	-2147483648	2147483647
	K106				First advanced filter coefficient for term xk	1073741824	-2147483648	2147483647
	K107				First advanced filter coefficient for term xk-1	0	-2147483648	2147483647
	K108				First advanced filter coefficient for term xk-2	0	-2147483648	2147483647
	K114				Second advanced filter coefficient for term yk-1	0	-2147483648	2147483647
	K115				Second advanced filter coefficient for term yk-2	0	-2147483648	2147483647

DSCDP parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
K116				Second advanced filter coefficient for term xk	1073741824	-2147483648	2147483647
K117				Second advanced filter coefficient for term xk-1	0	-2147483648	2147483647
K118				Second advanced filter coefficient for term xk-2	0	-2147483648	2147483647
K120				Enables trace synchronization mode. See EBL2 manual	0	0	1
K121				Trace register selection type (depth 0 for trace 0; depth 1 for trace 1,...). See EBL2 manual.	3	0	8
		1		Users variable register (X)			
		2		Parameter register (K)			
		3		Monitoring register (M)			
		8		LKT register (L)			
K122				Trace register number and depth selection (bit#0-15 for register number, bit#16-23 for depth). See EBL2 manual	0	-2147483648	2147483647
K123				Trace time in sti between two trace acquisition points. See EBL2 manual	0	0	65535
K125				Trace trigger mode. See EBL2 manual	0	0	5
		0		No trigger			
		1		Start of movement			
		2		End of movement			
		3		Trigger at a position selected by K127			
		4		Trigger at a register value defined by K127. K126=0 means that register defined in depth 0 is chosen for trigger, and K126=1 for depth 1,…			
		5		Trigger never starts; only ZFT command can start trigger acquisition			
K126				The value indicates that the register selected for the trigger condition is defined in depth of K121 and K122 parameters. See EBL2 manual	0	0	3
K127				Value of the register selected by K121, K122 and K126 parameters for acquisition start. See EBL2 manual	0	-2147483648	2147483647
K128				Number of points for the trace. See EBL2 manual	1024	0	1024
K133				AUT command definition mode	0	0	1
		0		Principle for finding the fine phase adjustment 0:old principle 1:new principle			
K135				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K136				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K137				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K138				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K139				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K141				Enables test of motor overtemperature protection connected to DIN and test of TEB time-out error. K141 is a global parameter (for both axes), it is only available on pair axis (for example: K141.0=2 is available, K141.1=2 is not available)	0	0	122
		2	1	Enables time-out TEB error test			
		8	3	Enables test of motor overtemperature protection, connected to DIN1			
		16	4	Enables test of motor overtemperature protection, connected to DIN2			
		32	5	Enables test of motor overtemperature protection, connected to DIN9			

DSCDP parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		64	6	Enables test of motor overtemperature protection, connected to DIN10			
K145				Searches limit stroke (SLS) mode	0	0	3
		0		Positive movement to search mechanical end stop			
		1		Negative movement to search mechanical end stop			
		2		Positive movement to search mechanical limit switch			
		3		Negative movement to search mechanical limit switch			
K150	MST			Optional board source register type. It has four depths to monitor four registers per sti (slow time interrupt)	0	0	3
		1		Source type is a user's variable X			
		2		Source type is a parameter K			
		3		Source type is a monitoring M			
K151	MSN			Optional board source register index	0	0	255
K154	MOF			Optional board source register offset	0	-2147483648	2147483647
K155	MAM			Optional board source register gain	16777216	-2147483648	2147483647
K156	DAO			Optional board analog offset. It has 4 depths for DSO-HIO analog output, each depth corresponds to the depths of K173	0	-32768	32767
K157	DAA			Optional board analog gain. It has 4 depths, for DSO-HIO analog output, each depth corresponds to the depths of K173	32767	-32768	32767
K158	ASE			Optional board source register selection. It has four depths for the four analog outputs. Choose the axis to link with the monitoring source	0	0	1
K159				Inverts the digital inputs for position capture or with DSO-MAC optional board. If K33=0 (DIN1 is a safety input), DIN1 cannot be inverted	0	0	771
K160				Syncronization input mask that will be tested for synchronized start movement on digital input (STI command)	0	0	771
K161				Syncronization input mask value (used with STI command)	0	0	771
K162				Syncronization output mask (used with STI command)	0	0	3
K163				Syncronization output value (used with STI command)	0	0	3
K164				Syncronization time-out (used with STI command)	0	0	600000
K165				Enables scaling/mapping correction	0	0	4
		0		Mapping and scaling deactivated			
		1		Mapping activated			
		2		Scaling activated			
		3		Mapping and scaling activated			
		4		Rotary mapping activated			
K166				Position (dpi) where the mapping correction starts	0	-2147483648	2147483647
K167				Length (dpi) where the mapping correction is active	0	0	2147483647
K168				Position (dpi) where the scaling correction is null	0	-2147483648	2147483647
K169				This value/100000 multiplied by the position M8 gives the scaling correction	0	-50000	50000
K171	DOUT			Activates ot deactivates the digital outputs of the controller (DOUT1 and DOUT2). K37 (if different from 0) changes the value of the digital outputs in case of error, but not the value of DOUT (K171)	0	0	3

DSCDP parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
K172	XDOUT			Optional board digital outputs (8 for DSO-HIO)	0	0	255
K173	XAOUT			Optional board analog outputs of the DSO-HIO (8191 = -20V between XAOUT+ and XAOUT-, -8192 = +20V between XAOUT+ and XAOUT-). It has 4 depths that corresponds to the 4 analog outputs of the DSO-HIO	0	-8192	8191
K176				Enables the fast trigger mode (available in macro mode only)	0	0	3
		1	0	Enables fast trigger mode. Digital output masked by K188 are set or cleared according to bit#1 of K182 when M1 is greater or lower than K189			
		2	1	If this bit is set, digital outputs masked by K188 are cleared when M1 is greater than K189, and set when M1 is lower than K189. If this bit is cleared, the digital outputs are inverted in comparison with the previous case.			
K177				User word of M63 (bit#0 to 15) and user byte of SD2 (bit#8 to 15)	0	0	65535
K178				Depth 0: rising edge mask on digital input for position capture. Depth 2: mask on digital input that must be at 1 for K60/K31 limitation	0	0	771
K179				Depth 0: falling edge mask on digital input for position capture. Depth 2: mask on digital input that must be at 0 for K60/K31 limitation	0	0	771
K182				Enables position capture on input according to K178 and K179. Writing a 1 in K182 enables the capture and resets bit#2 of SD2 and bit#30 of M63. Writing a 0 stops the process	0	0	1
K183				Digital output mask on optional board for trigger	0	0	255
K184				Mask on user word of M63 (bit#0 to 15) and user byte of SD2 (bit#8 to 15) for trigger	0	0	65535
K185	TMK			Digital output mask for trigger	0	0	3
K186	TRS			Trigger mapping number. -1 deactivates the trigger	-1	-1	191
K187	TNB			Trigger mapping size	0	0	192
K188				Mask on the digital output in fast trigger mode. Available in Macro mode with K176 different from 0	0	0	3
K189				Position (dpi) that is compared to M1 to activate digital output in fast trigger mode (K176 different from 0). K188 is the mask of the digital outputs and K176 enables the selection to modify them. Avalaible in Macro mode only	0	-2147483648	2147483647
K190				Global enable of real-time function	0	0	1
K191				List of valid real-time lines in the RTI table	0	0	255
K192				List of enable real-time lines in the RTI table	0	0	255
K193				List of pending real-time lines in the RTI table	0	0	255
K194				Time added at the end of the movement where K60 is kept as force/torque reference. At the end of this time, K59 is kept as force/torque reference	0	0	600000
K198				The register number is given by K198 when it is equal to Y or y (ex: Ky.1=...). See EBL2 manual	0	0	65535
K201	MMC			Concatenated movement selection	0	0	3
		0		Concatenated movement disabled			
		1		Concatenated movement enabled for MMD=1, 17 and 24			
		2		Continuous back and forth movement enabled for MMD = 10, 26, 3 and 19 movement (could be stopped by POS, BRK, HLT, HLB and HLO commands)			

DSCDP parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		3		One back and forth MMD = 10, 26, 3 and 19 movement enabled for look-up table movement			
K202	MMD			Movement type	1	1	32
		1		S-curve (jerk time) linear movement			
		3		Selects a predefined linear movement according to K230 value and executes the trajectory in a time given by K229			
		10		Look-up table linear movement			
		17		S-curve (jerk time) rotary movement			
		19		Selects a predefined rotary movement according to K230 value and executes the trajectory in a time given by K229			
		24		Infinite rotary movement			
		26		LKT rotary movement			
K203	LTN			Look-up table number movement	0	0	7
K204	LTI			Time to execute a look-up table movement	10000	4	500000
K205	CAM			Came value (in percent). Stretches the user time scale	100	1	100
K206				Brake deceleration (with BRK and HLB command)	1000000	256	2147483647
K207				LKT mode selection	0	0	1
		0		LKT movement running to target defined by POS command			
		1		LKT movement with same starting and end point. K208 defines the amplitude of the movement			
K208				Maximum stroke for LKT movement with K207=1 and for MMD=10 & 26	0	-2147483648	2147483647
K209				Rotary movement type selection	0	0	2
		0		Movement always positive			
		1		Movement always negative			
		2		Movement minimum distance			
K210	POS			Starts movement (only depth 0) and gives the target position	0	-2147483648	2147483647
K211	SPD			Absolute maximum speed	2000000	1	2147483647
K212	ACC			Absolute maximum acceleration and deceleration	1000000	256	2147483647
K213	JRT			Jerk time	0	0	500
K219				TEB real-time slave to master monitoring pointer	0	-2147483648	2147483647
K220				Control source type	1	1	3
		1		Source type is a user's variable X			
		2		Source type is a parameter K			
		3		Source type is a monitoring M			
K221				Control source index	0	0	255
K222				Control source shift factor	0	0	16
K223				Control source offset	0	-2147483648	2147483647
K224				Control source gain	8388608	-2147483648	2147483647
K229				Execution time of the movement selected by K230	10000	4	500000
K230				Movement type selection for MMD = 3 or 19	0	0	3
		0		Triangular speed movement			
		1		S-curve (full jerk) movement			
		2		Sine modified movement			

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP parameters			3		Real sine movement			
	K240				Motor type. 0: linear motor, 1:rotary motor	0	0	1
	K241				Encoder period in [nm] for linear encoder or number of period for rotary encoder	1	-2147483648	2147483647
	K242				Position multiplication factor (is used by the DLL to calculate the position unit with indirect encoder)	1	1	2147483647
	K243				Position division factor (is used by the DLL to calculate the position unit with indirect encoder)	1	1	2147483647

16.3 Parameters K for DSCDL

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL parameters	K1				Position loop proportional gain	100	0	2147483647
	K2				Position loop speed feedback gain	20	0	2147483647
	K3				Position loop force feedback gain	0	0	2147483647
	K4				Position loop integrator gain	0	0	2147483647
	K5				Position loop anti-windup gain	10	0	2147483647
	K6				Position loop integrator limitation	1073741823	0	2147483647
	K7				Position loop integrator mode	0	0	2
			0		Position loop integrator gain (K4) always on			
			2		Position loop integrator gain (K4) always off			
	K8				Position loop speed filter	0	0	511
	K9				Force reference filter on the position regulator output	0	0	511
	K11				Speed smooth filter for TTL encoder	0	0	256
	K20				Position loop speed feedforward gain	0	0	2147483647
	K21				Position loop acceleration feedforward gain	0	0	2147483647
	K23				Commutation phase advance according to the speed	0	0	65535
	K27				Maximum position range limit for rotary movement	1	0	2000000000
	K30				Absolute tracking error limit. When the tracking error is greater than K30, the controller generates the error M64=23	10000000	0	1073741823
	K31				Absolute velocity limit. When the velocity is greater than K31, the controller generates an overspeed error M64=24	100000000	0	2147483647
	K32				Limit switch and home switch inversion	0	0	47
			1	0	Enables the use of the limit switches			
			2	1	Inverts home switch			
			4	2	Inverts limit switches from the encoder			
			8	3	Inverts limit switches			
			32	5	Enables limitation of K60/K31 according to DIN. K178:2 defines the mask of the DIN that should be at 1 to limit K60 and K31. K179:2 defines the mask of the DIN that should be at 0 to limit K60 and K31. K60:3 and K31:3 are the limitation			
	K33				Enables input mode	125	0	125

DSCDL parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		0		Enabled signal is necessary to power on the controller on DIN1. In this case this input must be at 1 when a PWR.#=1 command is executed. If this input is cleared, the controller generates an error (M64=26)			
		125		Enabled signal not used (DIN1 is not taken into account). PWR.#=1 command switches the power on the motor			
K34				Minimum software position limit	0	-2147483648	2147483647
K35				Maximum software position limit	0	-2147483648	2147483647
K36				Enables position limit (K34, K35) generating an error depending on the value	0	0	7
		1	0	Use of K34 and K35 as limit on the target the motor can reach. Used with K61=1			
		2	1	Use of K34 and K35 as limit on the actual position of the motor. If the motor reaches these limits, it generates an error (M64=65). These limits are tested on every sti but only if a homing has been previously done. Used for all values of K61			
		4	2	Use of K34 and K35 as limit on the target to generate an error (M64=66) when the movement starts. Used with K61=1			
K37				Mask of the digital output (DOUT) that must be cleared when the controller is in error. When the controller is not in error any more, the digital outputs have the DOUT value.	0	0	3
K38				Duration of the window (used with WTW command)	0	0	393210
K39				Position range of the window (used with WTW command)	0	0	1073741823
K40				Homing mode	8	0	39
		0		Homing on mechanical end stop			
		1	0	Homing with a negative movement			
		2		Homing on home switch			
		4		Homing on limit switch			
		6		Homing on home switch with detection of limit switch			
		8		Homing on a single index			
		10		Homing on a single index with detection of limit switch			
		12		Homing on a multi-index			
		14		Homing on a multi-index with detection of limit switch			
		16		Homing on a single index with DIN2 at 1			
		18		Homing on a single index with DIN2 at 1 with detection of limit switch			
		20		Homing on multi-index with a defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
		22		Immediate homing at the current position (it is used to take K53 at a known position into account)			
		24		Single index homing with defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
		26		Homing on single index. If the motor reaches a mechanical end stop before an index, it changes its movement direction. If after the mechanical end stop it does not meet an index before a stroke defined by K46, the controller generates an error (M64=62)			

DSCDL parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		34		Homing on a single index with detection of limit switch coming from the encoder			
		36		Homing on home switch. If the motor reaches a mechanical end stop before home switch, it changes its movement direction. If after the mechanical end stop it does not meet home switch before stroke defined by K46, the controller generates an error (M64=62)			
		38		Homing on single index after having found mechanical end stop with defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
K41				Homing speed	2000000	1	2147483647
K42				Homing acceleration	1000000	256	2147483647
K43				Homing tracking limit for mechanical end stop detection	10000000	0	2147483647
K44				Homing force limit for mechanical end stop detection	4096	0	32767
K45				Offset on absolute position	0	-2147483648	2147483647
K46				Stroke for K40 = 20, 21, 24, 25 homing mode	0	0	2147483647
K47				Movement to go out of a limit switch or mechanical end stop at the end of the homing	0	0	2147483647
K48				Movement to go out of an index or home switch if the motor is on the top of it when starting the homing	0	0	2147483647
K50				Set point calculator shift value: 1dpi = 2^K50 * 1upi	0	0	8
K52				Enables fine phase adjustment (takes K53 into account) after homing	0	0	1
K53				Motor commutation phase adjustment after homing (is taken into account only if K52=1)	0	0	2048
K54				Pairs of pole of the motor (=1 for linear motor)	1	1	2147483647
K55				Motor commutation encoder: number of dpi per revolution for rotary motor or number of dpi per magnetic period for linear motor	0	0	2147483647
K56				Motor commutation phase inversion enabled	0	0	1
K58				Limit switch mode	0	0	2
		0		Limit switch mode DIN9 and DIN10			
		1		Limit switch mode L1/L2			
		2		Limit switch mode L/H			
K59				Theoretical software force/torque limit for stepper, when no movement is required	20000	0	31000
K60				Theoretical software force/torque limit (regulator output)	20000	0	31000
K61				Reference mode	1	0	39
		0		Force/torque reference mode. The reference value is defined by K220 to K224 (32767 is equivalent to the current defined by M82)			
		1		Standard position profile mode			
		3		Controller controlled by a speed reference defined by K220 to K224			
		4		Controller controlled by a position reference defined by K220 to K224			
		36		Controller controlled by a position reference defined by K220 to K224. After a power on, it takes into account the actual motor position as reference			
K66				Display mode	1	1	64
		1		Displays normal informations			

DSCDL parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		2		Displays temperature of the controller			
		4		Displays analog encoder amplitude and index position			
		8		Displays sequence line number			
		16		Displays optional board message			
K68				Inverts positive/negative way	0	0	3
		1	0	Inverts analog 1Vptp encoder			
		2	1	Inverts TTL encoder			
K69				Encoder shift value for TTL encoder (K79=1)	0	0	12
K70				Analog encoder sine offset	0	-1024	1024
K71				Analog encoder cosine offset	0	-1024	1024
K72				Analog encoder sine factor	32767	16384	32767
K73				Analog encoder cosine factor	32767	16384	32767
K75				Distance between two indexes for multi-indexes encoder	0	0	2147483647
K77				Encoder interpolation shift value for analog encoder	0	0	12
K79				Encoder type selection	0	0	120
		0		Analog sine/cosine encoder 1Vptp			
		1		TTL encoder			
		4		EnDat encoder			
		20		Stepper in open loop without encoder			
		21		Stepper in open loop with TTL encoder as secondary			
		23		Stepper in open loop with 1Vptp encoder as secondary			
		24		Stepper in open loop with EnDat encoder as secondary			
K80				Current loop proportional gain	100	0	16383
K81				Current loop integrator gain	0	0	127
K83				Current loop software overcurrent limit	16000	0	32000
K84				Current loop i2t rms current limit	510	0	8192
K85				Current loop i2t time limit	20966400	0	2147483647
K87				Synchronization mode and frequency of the controller	0	0	1023
K89				Motor phase number	20	10	41
		10		1 phase motor			
		20		2 phase motor			
K90				Phasing mode. Depth 1: phasing mode for command AUT.x=8	2	0	6
		0		No phasing (with 1-phase)			
		2		Phasing by sending constant current to the motor (ironcore and ironless motors)			
		6		Small movements phasing			
K91				Phasing pulse level (with K90=1 or K90=6)	12000	0	28000
K92				Phasing constant current level (with K90=2)	12000	0	28000
K93				Phasing final phase (with K90=2)	1024	0	2048
K94				Phasing time process (with K90=2)	5000	0	25000
K97				Phasing initial phase (with K90=2)	512	0	2048
K100				SPD, ACC and MMD after IND selection mode	0	0	18
		2	1	Sets SPD, ACC and MMD parameter that was set before IND command at the end of the homing process			

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL parameters	K101				Phasing time process (with K90=6)	1000	0	25000
	K102				Enables or disables the TEB Real-Time Monitoring (RTM)	0	0	1
			1	0	Disables slave to master RTM			
	K104				First advanced filter coefficient for term yk-1	0	-2147483648	2147483647
	K105				First advanced filter coefficient for term yk-2	0	-2147483648	2147483647
	K106				First advanced filter coefficient for term xk	1073741824	-2147483648	2147483647
	K107				First advanced filter coefficient for term xk-1	0	-2147483648	2147483647
	K108				First advanced filter coefficient for term xk-2	0	-2147483648	2147483647
	K114				Second advanced filter coefficient for term yk-1	0	-2147483648	2147483647
	K115				Second advanced filter coefficient for term yk-2	0	-2147483648	2147483647
	K116				Second advanced filter coefficient for term xk	1073741824	-2147483648	2147483647
	K117				Second advanced filter coefficient for term xk-1	0	-2147483648	2147483647
	K118				Second advanced filter coefficient for term xk-2	0	-2147483648	2147483647
	K120				Enables trace synchronization mode. See EBL2 manual	0	0	1
	K121				Trace register selection type (depth 0 for trace 0; depth 1 for trace 1,...). See EBL2 manual.	3	0	8
			1		Users variable register (X)			
			2		Parameter register (K)			
			3		Monitoring register (M)			
			8		LKT register (L)			
	K122				Trace register number and depth selection (bit#0-15 for register number, bit#16-23 for depth). See EBL2 manual	0	-2147483648	2147483647
	K123				Trace time in sti between two trace acquisition points. See EBL2 manual	0	0	65535
	K125				Trace trigger mode. See EBL2 manual	0	0	5
			0		No trigger			
			1		Start of movement			
			2		End of movement			
			3		Trigger at a position selected by K127			
			4		Trigger at a register value defined by K127. K126=0 means that register defined in depth 0 is chosen for trigger, and K126=1 for depth 1,…			
			5		Trigger never starts; only ZFT command can start trigger acquisition			
	K126				The value indicates that the register selected for the trigger condition is defined in depth of K121 and K122 parameters. See EBL2 manual	0	0	3
	K127				Value of the register selected by K121, K122 and K126 parameters for acquisition start. See EBL2 manual	0	-2147483648	2147483647
	K128				Number of points for the trace. See EBL2 manual	1024	0	1024
	K133				AUT command definition mode	0	0	1
			0		Principle for finding the fine phase adjustment 0:old principle 1:new principle			
	K135				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
	K136				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
	K137				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
	K138				Optional board parameter (see optional board manual)	0	-2147483648	2147483647

DSCDL parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
K139				Optional board parameter (see optional board manual)	0	-2147483648	2147483647
K140				Mask for fuse control	0	0	1
		1	0	Disables the test of F2 fuse			
K141				Enables test of motor overtemperature protection connected to DIN and test of TEB time-out error. K141 is a global parameter (for both axes), it is only available on pair axis (for example: K141.0=2 is available, K141.1=2 is not available)	0	0	122
		2	1	Enables time-out TEB error test			
		8	3	Enables test of motor overtemperature protection, connected to DIN1			
		16	4	Enables test of motor overtemperature protection, connected to DIN2			
		32	5	Enables test of motor overtemperature protection, connected to DIN9			
		64	6	Enables test of motor overtemperature protection, connected to DIN10			
K145				Searches limit stroke (SLS) mode	0	0	3
		0		Positive movement to search mechanical end stop			
		1		Negative movement to search mechanical end stop			
		2		Positive movement to search mechanical limit switch			
		3		Negative movement to search mechanical limit switch			
K146				Limit of Vpower to activate undervoltage warning [V/100]. When Vpower[V] * 100 is lower than K146, the controller generates an undervoltage warning (M66=10). If K146=0, the limit detection is disabled	1200	0	4000
K147				Limit of Vpower to activate undervoltage error [V/100]. When Vpower[V] * 100 is lower than K147, the controller generates an undervoltage error (M64=9). If K147=0, the limit detection is disabled	800	0	4000
K148				Limit of Vpower to activate overvoltage warning [V/100]. When Vpower[V] * 100 is greater than K148, the controller generates an overvoltage warning (Set bit#19 of SD1). If K148=0, the limit detection is disabled. For DSC2P the limit is reduced by firmware	3700	0	4000
K149				Limitt of Vpower to activate overvoltage error [V/100]. When Vpower[V] * 100 is greater than K149, the controller generates an overvoltage error (M64=6). If K149=0, the limit detection is disabled.	3800	0	4000
K150	MST			Optional board source register type. It has four depths to monitor four registers per sti (slow time interrupt)	0	0	3
		1		Source type is a user's variable X			
		2		Source type is a parameter K			
		3		Source type is a monitoring M			
K151	MSN			Optional board source register index	0	0	255
K154	MOF			Optional board source register offset	0	-2147483647	2147483647
K155	MAM			Optional board source register gain	16777216	-2147483647	2147483647
K156	DAO			Optional board analog offset. It has 4 depths for DSO-HIO analog output, each depth corresponds to the depths of K173	0	-32767	32767
K157	DAA			Optional board analog gain. It has 4 depths, for DSO-HIO analog output, each depth corresponds to the depths of K173	32767	-32767	32767

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL parameters	K158	ASE			Optional board source register selection. It has four depths for the four analog outputs. Choose the axis to link with the monitoring source	0	0	1
	K159				Inverts the digital inputs for position capture or with DSO-MAC optional board. If K33=0 (DIN1 is a safety input), DIN1 cannot be inverted	0	0	771
	K160				Syncronization input mask that will be tested for synchronized start movement on digital input (STI command)	0	0	771
	K161				Syncronization input mask value (used with STI command)	0	0	771
	K162				Syncronization output mask (used with STI command)	0	0	3
	K163				Syncronization output value (used with STI command)	0	0	3
	K164				Syncronization time-out (used with STI command)	0	0	600000
	K165				Enables scaling/mapping correction	0	0	4
			0		Mapping and scaling deactivated			
			1		Mapping activated			
			2		Scaling activated			
			3		Mapping and scaling activated			
			4		Rotary mapping activated			
	K166				Position (dpi) where the mapping correction starts	0	-2147483648	2147483647
	K167				Length (dpi) where the mapping correction is active	0	0	2147483647
	K168				Position (dpi) where the scaling correction is null	0	-2147483648	2147483647
	K169				This value/100000 multiplied by the position M8 gives the scaling correction	0	-50000	50000
	K171	DOUT			Activates ot deactivates the digital outputs of the controller (DOUT1 and DOUT2). K37 (if different from 0) changes the value of the digital outputs in case of error, but not the value of DOUT (K171)	0	0	3
	K172	XDOUT			Optional board digital outputs (8 for DSO-HIO)	0	0	255
	K173	XAOUT			Optional board analog outputs of the DSO-HIO (8191 = -20V between XAOUT+ and XAOUT-, -8192 = +20V between XAOUT+ and XAOUT-). It has 4 depths that corresponds to the 4 analog outputs of the DSO-HIO	0	-8192	8192
	K177				User word of M63 (bit#0 to 15) and user byte of SD2 (bit#8 to 15)	0	0	65535
	K178				Depth 0: rising edge mask on digital input for position capture. Depth 2: mask on digital input that must be at 1 for K60/K31 limitation	0	0	771
	K179				Depth 0: falling edge mask on digital input for position capture. Depth 2: mask on digital input that must be at 0 for K60/K31 limitation	0	0	771
	K182				Enables position capture on input according to K178 and K179. Writing a 1 in K182 enables the capture and resets bit#2 of SD2 and bit#30 of M63. Writing a 0 stops the process	0	0	1
	K183				Digital output mask on optional board for trigger	0	0	255
	K184				Mask on user word of M63 (bit#0 to 15) and user byte of SD2 (bit#8 to 15) for trigger	0	0	65535
	K185	TMK			Digital output mask for trigger	0	0	3
	K186	TRS			Trigger mapping number. -1 deactivates the trigger	-1	-1	191
	K187	TNB			Trigger mapping size	0	0	192

DSCDL parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
K190				Global enable of real-time function	0	0	1
K191				List of valid real-time lines in the RTI table	0	0	255
K192				List of enable real-time lines in the RTI table	0	0	255
K193				List of pending real-time lines in the RTI table	0	0	255
K194				Time added at the end of the movement where K60 is kept as force/torque reference. At the end of this time, K59 is kept as force/torque reference	0	0	600000
K198				The register number is given by K198 when it is equal to Y or y (ex: Ky.1=...). See EBL2 manual	0	0	65535
K201	MMC			Concatenated movement selection	0	0	3
		0		Concatenated movement disabled			
		1		Concatenated movement enabled for MMD=1, 17 and 24			
		2		Continuous back and forth movement enabled for MMD = 10, 26, 3 and 19 movement (could be stopped by POS, BRK, HLT, HLB and HLO commands)			
		3		One back and forth MMD = 10, 26, 3 and 19 movement enabled for look-up table movement			
K202	MMD			Movement type	1	1	32
		1		S-curve (jerk time) linear movement			
		3		Selects a predefined linear movement according to K230 value and executes the trajectory in a time given by K229			
		10		Look-up table linear movement			
		17		S-curve (jerk time) rotary movement			
		19		Selects a predefined rotary movement according to K230 value and executes the trajectory in a time given by K229			
		24		Infinite rotary movement			
		26		LKT rotary movement			
K203	LTN			Look-up table number movement	0	0	7
K204	LTI			Time to execute a look-up table movement	10000	4	500000
K205	CAM			Came value (in percent). Stretches the user time scale	100	1	100
K206				Brake deceleration (with BRK and HLB command)	1000000	256	2147483647
K207				LKT mode selection	0	0	1
		0		LKT movement running to target defined by POS command			
		1		LKT movement with same starting and end point. K208 defines the amplitude of the movement			
K208				Maximum stroke for LKT movement with K207=1 and for MMD=10 & 26	0	-2147483648	2147483647
K209				Rotary movement type selection	0	0	2
		0		Movement always positive			
		1		Movement always negative			
		2		Movement minimum distance			
K210	POS			Starts movement (only depth 0) and gives the target position	0	-2147483648	2147483647
K211	SPD			Absolute maximum speed	2000000	1	2147483647
K212	ACC			Absolute maximum acceleration and deceleration	1000000	256	2147483647

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL parameters	K213	JRT			Jerk time	0	0	500
	K219				TEB real-time slave to master monitoring pointer	0	-2147483648	2147483647
	K220				Control source type	1	1	3
			1		Source type is a user's variable X			
			2		Source type is a parameter K			
			3		Source type is a monitoring M			
	K221				Control source index	0	0	255
	K222				Control source shift factor	0	0	16
	K223				Control source offset	0	-2147483648	2147483647
	K224				Control source gain	16777216	-2147483648	2147483647
	K229				Execution time of the movement selected by K230	10000	4	500000
	K230				Movement type selection for MMD = 3 or 19	0	0	3
			0		Triangular speed movement			
			1		S-curve (full jerk) movement			
			2		Sine modified movement			
			3		Real sine movement			
	K240				Motor type. 0: linear motor, 1:rotary motor	0	0	1
	K241				Encoder period in [nm] for linear encoder or number of period for rotary encoder	1	-2147483648	2147483647
	K242				Position multiplication factor (is used by the DLL to calculate the position unit with indirect encoder)	1	1	2147483647
	K243				Position division factor (is used by the DLL to calculate the position unit with indirect encoder)	1	1	2147483647

16.4 Parameters K for DSCDM

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM parameters	K1				Position loop proportional gain	100	0	2147483647
	K2				Position loop speed feedback gain	20	0	2147483647
	K3				Position loop force feedback gain	0	0	2147483647
	K4				Position loop integrator gain	0	0	2147483647
	K5				Position loop anti-windup gain	10	0	2147483647
	K6				Position loop integrator limitation	1073741823	0	2147483647
	K7				Position loop integrator mode	0	0	2
			0		Position loop integrator gain (K4) always on			
			2		Position loop integrator gain (K4) always off			
	K8				Position loop speed filter	0	0	511
	K9				Force reference filter on the position regulator output	0	0	511
	K11				Speed smooth filter for TTL encoder	0	0	256
	K20				Position loop speed feedforward gain	0	0	2147483647
	K21				Position loop acceleration feedforward gain	0	0	2147483647
	K23				Commutation phase advance according to the speed	0	0	65535
	K27				Maximum position range limit for rotary movement	1	0	2000000000

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM parameters	K30				Absolute tracking error limit. When the tracking error is greater than K30, the controller generates the error M64=23	10000000	0	1073741823
	K31				Absolute velocity limit. When the velocity is greater than K31, the controller generates an overspeed error M64=24	100000000	0	2147483647
	K32				Limit switch and home switch inversion	0	0	63
			1	0	Enables the use of the limit switches			
			2	1	Inverts home switch			
			4	2	Inverts limit switches from the encoder			
			8	3	Inverts limit switches			
			16	4	Enables the use of the dynamic braking controlled by the transistors			
			32	5	Enables limitation of K60/K31 according to DIN. K178:2 defines the mask of the DIN that should be at 1 to limit K60 and K31. K179:2 defines the mask of the DIN that should be at 0 to limit K60 and K31. K60:3 and K31:3 are the limitation			
	K33				Enables input mode	125	0	125
			0		Enabled signal is necessary to power on the controller on DIN1. In this case this input must be at 1 when a PWR.#=1 command is executed. If this input is cleared, the controller generates an error (M64=26)			
			125		Enabled signal not used (DIN1 is not taken into account). PWR.#=1 command switches the power on the motor			
	K34				Minimum software position limit	0	-2147483648	2147483647
	K35				Maximum software position limit	0	-2147483648	2147483647
	K36				Enables position limit (K34, K35) generating an error depending on the value	0	0	7
			1	0	Use of K34 and K35 as limit on the target the motor can reach. Used with K61=1			
			2	1	Use of K34 and K35 as limit on the actual position of the motor. If the motor reaches these limits, it generates an error (M64=65). These limits are tested on every sti but only if a homing has been previously done. Used for all values of K61			
			4	2	Use of K34 and K35 as limit on the target to generate an error (M64=66) when the movement starts. Used with K61=1			
	K37				Mask of the digital output (DOUT) that must be cleared when the controller is in error. When the controller is not in error any more, the digital outputs have the DOUT value.	0	0	7
	K38				Duration of the window (used with WTW command)	0	0	393210
	K39				Position range of the window (used with WTW command)	0	0	1073741823
	K40				Homing mode	8	0	41
			0		Homing on mechanical end stop			
			1	0	Homing with a negative movement			
			2		Homing on home switch			
			4		Homing on limit switch			
			6		Homing on home switch with detection of limit switch			
			8		Homing on a single index			

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM parameters			10		Homing on a single index with detection of limit switch			
			12		Homing on a multi-index			
			14		Homing on a multi-index with detection of limit switch			
			16		Homing on a single index with choosen DIN (K58 bit3:DIN1, bit4:DIN9) at 1			
			18		Homing on a single index with choosen DIN (K58 bit3:DIN1, bit4:DIN9) at 1 with detection of limit switch			
			20		Homing on multi-index with a defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
			22		Immediate homing at the current position (It is used to take K53 at a known position into account)			
			24		Single index homing with defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
			26		Homing on single index. If the motor reaches a mechanical end stop before an index, it changes its movement direction. If after the mechanical end stop it does not meet an index before a stroke defined by K46, the controller generates an error (M64=62)			
			34		Homing on a single index with detection of limit switch coming from the encoder			
			36		Homing on home switch. If the motor reaches a mechanical end stop before home switch, it changes its movement direction. If after the mechanical end stop it does not meet home switch before stroke defined by K46, the controller generates an error (M64=62)			
			38		Homing on single index after having found mechanical end stop with defined stroke (K46). If no index has been met, the controller generates an error (M64=62)			
	K41				Homing speed	2000000	1	2147483647
	K42				Homing acceleration	1000000	256	2147483647
	K43				Homing tracking limit for mechanical end stop detection	10000000	0	2147483647
	K44				Homing force limit for mechanical end stop detection	4096	0	32767
	K45				Offset on absolute position	0	-2147483648	2147483647
	K46				Stroke for K40 = 20, 21, 24, 25 homing mode	0	0	2147483647
	K47				Movement to go out of a limit switch or mechanical end stop at the end of the homing	0	0	2147483647
	K48				Movement to go out of an index or home switch if the motor is on the top of it when starting the homing	0	0	2147483647
	K50				Set point calculator shift value: 1dpi = 2^K50 * 1upi	0	0	8
	K52				Enables fine phase adjustment (takes K53 into account) after homing	0	0	1
	K53				Motor commutation phase adjustment after homing (is taken into account only if K52=1)	0	0	2048
	K54				Pairs of pole of the motor (=1 for linear motor)	1	1	2147483647
	K55				Motor commutation encoder: number of dpi per revolution for rotary motor or number of dpi per magnetic period for linear motor	0	0	2147483647
	K56				Motor commutation phase inversion enabled	0	0	1
	K58				Limit switch mode	0	0	10
			0		Limit switch mode DIN9 and DIN10			
			1		Limit switch mode L1/L2			

DSCDM parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		2		Limit switch mode L/H			
		4	2	Home switch on DIN1			
		8	3	Home switch on DIN9			
K59				Theoretical software force/torque limit for stepper, when no movement is required	20000	0	31000
K60				Theoretical software force/torque limit (regulator output)	20000	0	31000
K61				Reference mode	1	0	39
		0		Force/torque reference mode. The reference value is defined by K220 to K224 (32767 is equivalent to the current defined by M82)			
		1		Standard position profile mode			
		3		Controller controlled by a speed reference defined by K220 to K224			
		4		Controller controlled by a position reference defined by K220 to K224			
		36		Controller controlled by a position reference defined by K220 to K224. After a power on, it takes into account the actual motor position as reference			
K66				Display mode	1	1	64
		1		Displays normal informations			
		2		Displays temperature of the controller			
		4		Displays analog encoder amplitude and index position			
		8		Displays sequence line number			
K68				Inverts positive/negative way	0	0	7
		1	0	Inverts analog 1Vptp encoder			
		2	1	Inverts TTL encoder			
K69				Encoder shift value for TTL encoder (K79=1)	0	0	12
K70				Analog encoder sine offset	0	-1024	1024
K71				Analog encoder cosine offset	0	-1024	1024
K72				Analog encoder sine factor	32767	16384	32767
K73				Analog encoder cosine factor	32767	16384	32767
K75				Distance between two indexes for multi-indexes encoder	0	0	2147483647
K77				Encoder interpolation shift value for analog encoder	0	0	12
K79				Encoder type selection	0	0	120
		0		Analog sine/cosine encoder 1Vptp			
		1		TTL encoder			
		4		EnDat encoder			
		20		Stepper in open loop without encoder			
		21		Stepper in open loop with TTL encoder as secondary			
		23		Stepper in open loop with 1Vptp encoder as secondary			
		24		Stepper in open loop with EnDat encoder as secondary			
K80				Current loop proportional gain	500	0	2147483647
K81				Current loop integrator gain	0	0	2147483647
K82				Current loop output filter	0	0	511
K83				Current loop software overcurrent limit	16000	0	32000
K84				Current loop i2t rms current limit	510	0	8192

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM parameters	K85				Current loop i2t time limit	20966400	0	2147483647
	K87				Synchronization mode and frequency of the controller	0	0	16777215
	K88				Reserved	18	0	32
	K89				Motor phase number and PWM type selection	30	10	41
			10		1 phase motor, PWM at 18KHz			
			20		2 phase motor, PWM at 18KHz			
			30		3 phase motor, PWM at 18KHz			
	K90				Phasing mode. Depth 1: phasing mode for command AUT.x=8	2	0	6
			0		No phasing (with 1-phase)			
			1		Phasing with current pulses (with ironcore motors only)			
			2		Phasing by sending constant current to the motor (ironcore and ironless motors)			
			3		Phasing with digital Hall sensors until the index is found then commutation by position encoder. The value stored in K53 is used if K52 = 1			
			4		Phasing with digital Hall sensors until the first edge of signal then commutation by position encoder. When index is found, the value stored in K53 is used if K52 = 1			
			5		Phasing and commutation with digital Hall sensors only			
			6		Small movements phasing			
	K91				Phasing pulse level (with K90=1 or K90=6)	12000	0	28000
	K92				Phasing constant current level (with K90=2)	12000	0	28000
	K93				Phasing final phase (with K90=2)	1024	0	2048
	K94				Phasing time process (with K90=2)	5000	0	25000
	K97				Phasing initial phase (with K90=2)	512	0	2048
	K98				Phasing power voltage rate in percent with K90=1. This is useful for slow inductance motor	100	25	100
	K100				SDP, ACC and MMD after IND selection mode	0	0	18
			2	1	Sets SPD, ACC and MMD parameter that was set before IND command at the end of the homing process			
	K101				Phasing time process (with K90=6)	1000	0	25000
	K102				Enables or disables the TEB Real-Time Monitoring (RTM)	0	0	1
			1	0	Disables slave to master RTM			
	K104				First advanced filter coefficient for term yk-1	0	-2147483648	2147483647
	K105				First advanced filter coefficient for term yk-2	0	-2147483648	2147483647
	K106				First advanced filter coefficient for term xk	1073741824	-2147483648	2147483647
	K107				First advanced filter coefficient for term xk-1	0	-2147483648	2147483647
	K108				First advanced filter coefficient for term xk-2	0	-2147483648	2147483647
	K114				Second advanced filter coefficient for term yk-1	0	-2147483648	2147483647
	K115				Second advanced filter coefficient for term yk-2	0	-2147483648	2147483647
	K116				Second advanced filter coefficient for term xk	1073741824	-2147483648	2147483647
	K117				Second advanced filter coefficient for term xk-1	0	-2147483648	2147483647
	K118				Second advanced filter coefficient for term xk-2	0	-2147483648	2147483647
	K120				Enables trace synchronization mode. See EBL2 manual	0	0	1

DSCDM parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
K121				Trace register selection type (depth 0 for trace 0; depth 1 for trace 1,...). See EBL2 manual.	3	0	8
		1		Users variable register (X)			
		2		Parameter register (K)			
		3		Monitoring register (M)			
		8		LKT register (L)			
K122				Trace register number and depth selection (bit#0-15 for register number, bit#16-23 for depth). See EBL2 manual	0	-2147483648	2147483647
K123				Trace time in sti between two trace acquisition points. See EBL2 manual	0	0	65535
K125				Trace trigger mode. See EBL2 manual	0	0	5
		0		No trigger			
		1		Start of movement			
		2		End of movement			
		3		Trigger at a position selected by K127			
		4		Trigger at a register value defined by K127. K126=0 means that register defined in depth 0 is chosen for trigger, and K126=1 for depth 1,…			
		5		Trigger never starts; only ZFT command can start trigger acquisition			
K126				The value indicates that the register selected for the trigger condition is defined in depth of K121 and K122 parameters. See EBL2 manual	0	0	3
K127				Value of the register selected by K121, K122 and K126 parameters for acquisition start. See EBL2 manual	0	-2147483648	2147483647
K128				Number of points for the trace. See EBL2 manual	1024	0	1024
K133				AUT command definition mode	0	0	1
		0		Principle for finding the fine phase adjustment 0:old principle 1:new principle			
K140				Mask for fuse control	0	0	1
		1	0	Disables the test of F2 fuse			
K141				Enables test of motor overtemperature protection connected to DIN and test of TEB time-out error. K141 is a global parameter (for both axes), it is only available on pair axis (for example: K141.0=2 is available, K141.1=2 is not available)	0	0	106
		2	1	Enables time-out TEB error test			
		8	3	Enables test of motor overtemperature protection, connected to DIN1			
		32	5	Enables test of motor overtemperature protection, connected to DIN9			
		64	6	Enables test of motor overtemperature protection, connected to DIN10			
K145				Searches limit stroke (SLS) mode	0	0	3
		0		Positive movement to search mechanical end stop			
		1		Negative movement to search mechanical end stop			
		2		Positive movement to search mechanical limit switch			
		3		Negative movement to search mechanical limit switch			
K159				Inverts the digital inputs for position capture. If K33=0 (DIN1 is a safety input), DIN1 cannot be inverted	0	0	769

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM parameters	K160				Syncronization input mask that will be tested for synchronized start movement on digital input (STI command)	0	0	769
	K161				Syncronization input mask value (used with STI command)	0	0	769
	K162				Syncronization output mask (used with STI command)	0	0	7
	K163				Syncronization output value (used with STI command)	0	0	7
	K164				Syncronization time-out (used with STI command)	0	0	600000
	K165				Enables scaling/mapping correction	0	0	4
			0		Mapping and scaling deactivated			
			1		Mapping activated			
			2		Scaling activated			
			3		Mapping and scaling activated			
			4		Rotary mapping activated			
	K166				Position (dpi) where the mapping correction starts	0	-2147483648	2147483647
	K167				Length (dpi) where the mapping correction is active	0	0	2147483647
	K168				Position (dpi) where the scaling correction is null	0	-2147483648	2147483647
	K169				This value/100000 multiplied by the position M8 gives the scaling correction	0	-50000	50000
	K171	DOUT			Activates ot deactivates the digital outputs of the controller (DOUT1 to DOUT3). K37 (if different from 0) changes the value of the digital outputs in case of error, but not the value of DOUT (K171)	0	0	7
	K177				User word of M63 (bit#0 to 15) and user byte of SD2 (bit#8 to 15)	0	0	65535
	K178				Depth 0: rising edge mask on digital input for position capture. Depth 2: mask on digital input that must be at 1 for K60/K31 limitation	0	0	769
	K179				Depth 0: falling edge mask on digital input for position capture. Depth 2: mask on digital input that must be at 0 for K60/K31 limitation	0	0	769
	K182				Enables position capture on input according to K178 and K179. Writing a 1 in K182 enables the capture and resets bit#2 of SD2 and bit#30 of M63. Writing a 0 stops the process	0	0	1
	K184				Mask on user word of M63 (bit#0 to 15) and user byte of SD2 (bit#8 to 15) for trigger	0	0	65535
	K185	TMK			Digital output mask for trigger	0	0	3
	K186	TRS			Trigger mapping number. -1 deactivates the trigger	-1	-1	191
	K187	TNB			Trigger mapping size	0	0	192
	K190				Global enable of real-time function	0	0	1
	K191				List of valid real-time lines in the RTI table	0	0	255
	K192				List of enable real-time lines in the RTI table	0	0	255
	K193				List of pending real-time lines in the RTI table	0	0	255
	K194				Time added at the end of the movement where K60 is kept as force/torque reference. At the end of this time, K59 is kept as force/torque reference	0	0	600000
	K195				Enables the selection of ETEL-Bus-Lite2 baudrate. It is taken into account only at the first switch on. If you want to change it, you have to set the value in K195 and then save it into the controller with SAV.#=2. Then switch off and on the controller	0	0	115200

DSCDM parameters

K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		0		ETEL-Bus-Lite2 at 115200 bauds (default value)			
		9600		ETEL-Bus-Lite2 at 9600 bauds			
		19200		ETEL-Bus-Lite2 at 19200 bauds			
		38400		ETEL-Bus-Lite2 at 38400 bauds			
		57600		ETEL-Bus-Lite2 at 57600 bauds			
		115200		ETEL-Bus-Lite2 at 115200 bauds			
K198				The register number is given by K198 when it is equal to Y or y (ex: Ky.1=...). See EBL2 manual	0	0	65535
K201	MMC			Concatenated movement selection	0	0	3
		0		Concatenated movement disabled			
		1		Concatenated movement enabled for MMD=1, 17 and 24			
		2		Continuous back and forth movement enabled for MMD = 10, 26, 3 and 19 movement (could be stopped by POS, BRK, HLT, HLB and HLO commands)			
		3		One back and forth MMD = 10, 26, 3 and 19 movement enabled for look-up table movement			
K202	MMD			Movement type	1	1	32
		1		S-curve (jerk time) linear movement			
		3		Selects a predefined linear movement according to K230 value and executes the trajectory in a time given by K229			
		10		Look-up table linear movement			
		17		S-curve (jerk time) rotary movement			
		19		Selects a predefined rotary movement according to K230 value and executes the trajectory in a time given by K229			
		24		Infinite rotary movement			
		26		LKT rotary movement			
K203	LTN			Look-up table number movement	0	0	7
K204	LTI			Time to execute a look-up table movement	10000	4	500000
K205	CAM			Came value (in percent). Stretches the user time scale	100	1	100
K206				Brake deceleration (with BRK and HLB command)	1000000	256	2147483647
K207				LKT mode selection	0	0	1
		0		LKT movement running to target defined by POS command			
		1		LKT movement with same starting and end point. K208 defines the amplitude of the movement			
K208				Maximum stroke for LKT movement with K207=1 and for MMD=10 & 26	0	-2147483648	2147483647
K209				Rotary movement type selection	0	0	2
		0		Movement always positive			
		1		Movement always negative			
		2		Movement minimum distance			
K210	POS			Starts movement (only depth 0) and gives the target position	0	-2147483648	2147483647
K211	SPD			Absolute maximum speed	2000000	1	2147483647
K212	ACC			Absolute maximum acceleration and deceleration	1000000	256	2147483647

	K	Alias	Val <P1>	Bit # <P1>	Comment for parameters K and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM parameters	K213	JRT			Jerk time	0	0	500
	K219				TEB real-time slave to master monitoring pointer	0	-2147483648	2147483647
	K220				Control source type	1	1	3
			1		Source type is a user's variable X			
			2		Source type is a parameter K			
			3		Source type is a monitoring M			
	K221				Control source index	0	0	255
	K222				Control source shift factor	0	0	16
	K223				Control source offset	0	-2147483648	2147483647
	K224				Control source gain	8388608	-2147483648	2147483647
	K229				Execution time of the movement selected by K230	10000	4	500000
	K230				Movement type selection for MMD = 3 or 19	0	0	3
			0		Triangular speed movement			
			1		S-curve (full jerk) movement			
			2		Sine modified movement			
			3		Real sine movement			
	K240				Motor type. 0: linear motor, 1:rotary motor	0	0	1
	K241				Encoder period in [nm] for linear encoder or number of period for rotary encoder	1	-2147483648	2147483647
	K242				Position multiplication factor (is used by the DLL to calculate the position unit with indirect encoder)	1	1	2147483647
	K243				Position division factor (is used by the DLL to calculate the position unit with indirect encoder)	1	1	2147483647

17. Monitorings M

17.1 Monitorings M for DSC2P and DSC2V

DSC2P and DSC2V monitorings

M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
M0				Theoretical position (dpi). Takes the scaling/mapping correction into account but does not take care of SET command and K50	0	-2147483648	2147483647
M1				Real position (dpi). Takes the scaling/mapping correction into account but does not take care of SET command and K50	0	-2147483648	2147483647
M2				Tracking error. This is the difference between M0 and M1	0	-2147483648	2147483647
M3				Maximum tracking error during movement	0	0	2147483647
M4				Offset between dpi and upi. [upi] = ([dpi] + M4) >>K50	0	-2147483648	2147483647
M5				Offset due to the homing (dpi)	0	-2147483648	2147483647
M6				Theoretical position (upi). Takes SET command, K50 and the scaling/mapping correction into account	0	-2147483648	2147483647
M7				Real position (upi). Takes SET command, K50 and the scaling/mapping correction into account	0	-2147483648	2147483647
M8				Theoretical position (dpi). Does not take SET command, K50 and scaling/mapping correction into account	0	-2147483648	2147483647
M10				Theoretical velocity (dsi)	0	-2147483648	2147483647
M11				Real velocity (dsi)	0	-2147483648	2147483647
M12				Real position (upi) captured on DIN. Takes SET command, K50 and scaling/mapping correction into account	0	-2147483648	2147483647
M13				Position given by the secondary encoder (dpi)	0	-2147483648	2147483647
M14				Theoretical acceleration (dai)	0	-2147483648	2147483647
M17				Reference value for mode K61=0, 1, 3, 4 and 36	0	-2147483648	2147483647
M20				Real current in phase 1	0	-32768	32767
M21				Real current in phase 2	0	-32768	32767
M22				Real current in phase 3	0	-32768	32767
M23				Current reference in phase 1	0	-32768	32767
M24				Current reference in phase 2	0	-32768	32767
M25				Current loop look-up table value of phase 1	0	-32768	32767
M26				Current loop look-up table value of phase 2	0	-32768	32767
M27				Current loop look-up table value of phase 3	0	-32768	32767
M29				PWM value of phase 1	0	-512	511
M30				Theoretical force after advanced filter 2	0	-32768	32767
M31				Real force	0	-32768	32767
M32				Theoretical force after K9 filter and before advanced filters	0	-32768	32767
M36				Inferior position after a SLS command	0	-2147483648	2147483647
M37				Superior position after a SLS command	0	-2147483648	2147483647
M40				Analog encoder sine signal	0	-2048	2047
M41				Analog encoder cosine signal	0	-2048	2047
M42				Analog encoder index signal	0	-2048	2047
M43				Analog encoder sine^2 + cosine^2	0	0	4194304

DSC2P and DSC2V monitorings

M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
M44				Encoder limit switch	0	0	3
		1	0	Encoder limit switch ELS (L1 or L)			
		2	1	Encoder limit switch EHO (L2 or H)			
M48				Digital Hall effect sensor signal	0	0	255
M50	DIN			Value of the 9 digital inputs: DIN1 to DIN7 (bits# 0 to 6), DIN 9 and DIN10 (bits# 8 to 9). Caution: bits# 4 to 7 are set to 0 if DIN5 to 7 are connected to a Hall effect sensor (K90 = 3, 4 or 5)	0	0	895
M51	AI1			Reads the voltage on the analog input	0	-2048	2047
M55	XDIN			Optional board digital inputs (8 inputs for DSO-HIO)	0	0	255
M56	XAIN1			Optional board analog input 1 (for DSO-HIO)	0	-8192	8191
M57	XAIN2			Optional board analog input 2 (for DSO-HIO)	0	-8192	8191
M58	XAIN3			Optional board analog input 3 (for DSO-HIO)	0	-8192	8191
M59	XAIN4			Optional board analog input 4 (for DSO-HIO)	0	-8192	8191
M60	SD1			Drive status 1	8	0	4294967295
		1	0	The controller is in power on			
		2	1	This bit is set when the controller has been initialized once (first PWR.# = 1)			
		4	2	This bit is set when the controller has finished the homing process with success			
		8	3	Bit present, always 1			
		16	4	The motor is executing a trajectory			
		32	5	This bit is set when the motor is in the position/time window defined by K38 and K39			
		64	6	Controller in u-master mode. This mode is set at the power on of the controller when K170 = 1			
		128	7	Controller is waiting			
		256	8	The controller is executing an internal sequence			
		512	9	The controller is in edition of sequence mode (EDI command). This mode allows the writing of S (sequence) register			
		1024	10	The controller is in error mode			
		2048	11	Trace busy flag is set during a register trace acquisition			
		8192	13	This bit is set during the homing process			
		8388608	23	Global warning			
M61	SD2			Drive status 2	0	0	4294967295
		1	0	Sequence error label pending			
		4	2	Position captured on digital input (see K182/K178/K179). This bit is set when the conditions of the digital input allow the capture of the position. It is reset when 1 is written in K182			
		256	8	User bit 0, could be modified by trigger functions or by K177			
		512	9	User bit 1, could be modified by trigger functions or by K177			
		1024	10	User bit 2, could be modified by trigger functions or by K177			
		2048	11	User bit 3, could be modified by trigger functions or by K177			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSC2P and DSC2V monitorings			4096	12	User bit 4, could be modified by trigger functions or by K177			
			8192	13	User bit 5, could be modified by trigger functions or by K177			
			16384	14	User bit 6, could be modified by trigger functions or by K177			
			32768	15	User bit 7, could be modified by trigger functions or by K177			
	M63				TEB status	524288	0	4294967295
			1	0	User word bit 0, could be modified by trigger functions or by K177			
			2	1	User word bit 1, could be modified by trigger functions or by K177			
			4	2	User word bit 2, could be modified by trigger functions or by K177			
			8	3	User word bit 3, could be modified by trigger functions or by K177			
			16	4	User word bit 4, could be modified by trigger functions or by K177			
			32	5	User word bit 5, could be modified by trigger functions or by K177			
			64	6	User word bit 6, could be modified by trigger functions or by K177			
			128	7	User word bit 7, could be modified by trigger functions or by K177			
			256	8	User word bit 8, could be modified by trigger functions or by K177			
			512	9	User word bit 9, could be modified by trigger functions or by K177			
			1024	10	User word bit 10, could be modified by trigger functions or by K177			
			2048	11	User word bit 11, could be modified by trigger functions or by K177			
			4096	12	User word bit 12, could be modified by trigger functions or by K177			
			8192	13	User word bit 13, could be modified by trigger functions or by K177			
			16384	14	User word bit 14, could be modified by trigger functions or by K177			
			32768	15	User word bit 15, could be modified by trigger functions or by K177			
			65536	16	The controller is in power on			
			524288	19	Bit present, always 1			
			1048576	20	The motor is executing a trajectory			
			2097152	21	This bit is set when the motor is in the position/time window defined by K38 and K39			
			8388608	23	The controller is in warning mode			
			16777216	24	The controller is executing an internal sequence			
			67108864	26	The controller is in error mode			
			134217728	27	Trace busy flag is set during a register trace acquisition			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSC2P and DSC2V monitorings			1073741824	30	Position captured according to the digital input (see K182/K178/K179). This bit is set when the conditions on the digital input allow the capture of the position. It is reset when 1 is written in K182			
	M64				Gives the error code	0	0	255
			2		The current measured in phase 1 is greater than K83			
			3		The current measured in phase 2 is greater than K83			
			4		This occurs when M67 becomes greater than K85. This is a power protection (of the motor and/or the controller)			
			5		The temperature of the controller is greater than 70°C. This is measured by a thermostat mounted on the heat sink			
			6		The power voltage (Vpower) in volt is greater than K149/100			
			7		Power supply inrush or power voltage too low when motor power is enabled (for rack version, it occurs when inrush signal on d14 of JC15 is not at 0V; this signal is given by ETEL's power supply or must be given by the user in case of another power supply)			
			8		Error that occurs when the +5V of the controller becomes lower than 4.5V			
			9		The power voltage (Vpower) in volt in lower than K147/100			
			10		Error in offset of current measurement. The offset measured if greater than 8% of the maximum current of the controller (M82)			
			13		The temperature sensor does not respond			
			16		The EnDat calculated position is too big. Decrease K77			
			17		Zero position of the EnDat encoder not found			
			18		Error on CRC from the EnDat			
			19		Error during the reading position of EnDat			
			20		The amplitudes of the analog encoder signals are too small			
			21		The encoder has lost the position acquisition, the frequency is too high			
			23		The tracking position error is greater than K30			
			24		The velocity is greater than K31			
			26		Error when power on with DIN1 is equal to 0 when K33 = 0			
			29		The temperature of the motor is too high, measured by temperature sensor connected to pin TSD (see Hardware Manual)			
			30		The controller generates this error when the controller reaches a limit switch during a movement (except during IND and SLS), if bit #0 of K32 is set			
			33		Error when the same label is defined several times. If the error appears, checks the sequence and if it is correct, erases sequence (NEW.#=1 command) and then download again the sequence			
			35		The fuse of the encoder supply is broken			
			36		Error when the label of a jump or a call is greater than 511			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSC2P and DSC2V monitorings			37		Error when the controller executes a command at a sequence line number greater than 8190			
			38		This error occurs when the register number is greater than its maximum value			
			39		This error occurs when the stack is overflowed (256 times)			
			40		This error occurs when K79 as a wrong value			
			41		This error occurs when K89 as a wrong value			
			52		No slave present on the TEB. The command or parameter axis number is wrong (axis does not exist) on the u-master chain			
			54		TEB has not been initialized correctly in u-master mode (K170=1 at power up of the controller)			
			56		Time-out error in TEB communication. This error is checked only if bit#1 of K141 is set			
			59		There are several nodes that have the same number on the TEB			
			60		Error in homing process when leaving home switch or limit switch. Checks K48 parameter or DIN 2, 9 or 10. This error occurs when after the trip from K48 the home switch or limit switch is present			
			61		Error in multi-index homing process. Checks K43, K44 and K75 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)			
			62		Error in single index homing process. Checks K43 and K44 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)			
			63		Error in start synchro on input (STI command)			
			65		Error when the real position (M7) is out of the limits defined by K34 and K35 when bit#1 of K36 is set			
			66		Error when the reference position is out of the limits defined by K34 and K35 when bit#2 of K36 is set			
			67		Movement not possible because the controller is not in power on mode			
			68		PWR ON not possible because the controller is in interpolation mode (by ITP=1)			
			69		Homing is not possible due to the configuration of the controller and the setting			
			80		Framing error on EBL2			
			81		Overrun error on EBL2			
			82		Checksum error on EBL2 message			
			84		Input buffer full on EBL2			
			85		Bad CRC on EBL2 in received message in CRC Mode			
			88		Other axis error on EBL2			
			89		Overrun error with Macro			
			90		Violation error with Macro			
			91		Parity error with Macro			
			92		Underrun error with Macro			
			93		Synchro error with Macro			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSC2P and DSC2V monitorings			94		Error auxiliary channel second command Macro			
			95		Error auxiliary channel Macro aux 2			
			96		Error auxiliary channel Macro aux 3			
			97		Error auxiliary channel Macro aux 4			
			100		Optional board error 100. See optional board manual			
			101		Optional board error 101. See optional board manual			
			102		Optional board error 102. See optional board manual			
			103		Optional board error 103. See optional board manual			
			104		Optional board error 104. See optional board manual			
			105		Optional board error 105. See optional board manual			
			106		Optional board error 106. See optional board manual			
			107		Optional board error 107. See optional board manual			
			108		Optional board error 108. See optional board manual			
			109		Optional board error 109. See optional board manual			
			110		Optional board error 110. See optional board manual			
			111		Optional board error 111. See optional board manual			
			112		Optional board error 112. See optional board manual			
			113		Optional board error 113. See optional board manual			
			114		Optional board error 114. See optional board manual			
			115		Optional board error 115. See optional board manual			
			116		This error is generated by the ERR command. In general, this command is sent by the DSMAX or the u-master			
			130		Hardware overcurrent. The current has reached the maximum admissible value by the controller			
			141		Watchdog error			
			144		The difference between the oscillator and quartz is too big			
			150		Bad measure during the phasing process when K90=1			
			151		Bad time measurement during phasing process when K90=1			
			152		Error when the controller is disabled during phasing process when K90 = 1 or 2			
			153		Too low force when K90=6			
			154		Too high force when K90=6			
			155		Too low time when K90=6			
			156		Time-out during AUT command			
			157		Error during the phasing process when K90=2			
			190		The controller has executed a save operation (SAV command)			
	M66				Gives the warning code	0	0	255
			1		This warning occurs when M67 is greater than K85/2			
			2		This warning occurs when the temperature of the controller is greater than 60°C			

DSC2P and DSC2V monitorings

M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		3		Power supply inrush or power voltage too low (for rack version, it occurs when inrush signal on d14 of JC15 is not at 0V. This signal is given by ETEL's power supply or must be given by the user in case of another power supply)			
		4		This warning occurs when the power voltage (Vpower) in volt in greater than K148/100			
		5		This warning occurs when the amplitude of the analog encoder signals are too small (2x the error)			
		6		This warning occurs when the tracking error is greater than K30/2			
		8		Wrong value coming from the digital Hall sensor			
		9		Loss of synchronization between the TEB signal and the controller interrupt			
		10		This warning occurs when the power voltage (Vpower) in volt is lower than K146/100			
M67				Current loop i2t value. When M67 is greater than K85, the controller generates an I2T error (M64=4)	0	0	2147483647
M70				Indicates the type of controller	6	6	6
M71				Gives the software boot version of the controller. A special ETEL procedure allows the conversion of this value in the software boot version (format is the same than M72)	0	0	4294967295
M72	VER			Gives the firmware version of the controller. A special ETEL procedure allows the conversion of this value in the firmware version	0	0	4294967295
M73	SER			Gives the serial number of the controller	0	0	4294967295
M76				Optional board type	0	0	255
		0		No optional board			
		4		DSO-MAC optional Macro protocol			
		7		DSO-HIO optional board with 8 digital I/O and 4 analog I/O			
		16		DSO-CAN optional w/CANetel protocol			
		24		DSO-SER optional Sercos protocol			
		32		DSO-PRO optional Profibus protocol			
M77				Optional board boot revision	0	0	255
M78				Optional board firmware version	0	0	4294967295
M79				Optional board serial number	0	0	4294967295
M81				Mask of axis present given by the u-master (only on u-master: axis 0)	1	1	2147483647
M82				Controller maximum current. M82/100 gives the maximum current in Ampere	2500	1250	6667
M85				Gives the article number string. The 16 strings of the article number are read using the 4 depths of M85. Each depth shows 4 strings (in ASCII)	0	0	4294967295
M86				Gives the article number string of the optional board. This is available only with DSO-CAN, DSO-SER, DSO-PRO. The 16 strings of the artcle number are read using the 4 depths of M86. Each depth shows 4 strings (in ASCII)	0	0	4294967295
M87				Gives the axis number	1	0	30
M90				Temperature of the controller measured by a thermostat on the heat sink	0	-40	100

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSC2P and DSC2V	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSC2P and DSC2V monitorings	M91				DC power voltage (Vpower) [V*100]. M91/100 gives the tension in Volt	0	0	100000
	M95				Shows the strings on the display. The 16 strings of the controller display are read using the 4 depths of M95. Each depth shows 4 string (in ASCII)	0	0	4294967295
	M96				Gives the line number being executed in the sequence	0	0	8190
	M110				Gives the interpolation mode of the position controller	0	0	255
			0		Interpolation mode disabled			
			1		Interpolation mode at every sti interrupt. Takes jerk time (IJT) and encoder scaling/mapping into account			
			2		Interpolation mode on fti interrupt. Does not take jerk time (IJT) and encoder scaling/mapping into account			
	M140				Fuse status	0	0	1
			1	0	Fuse F3 broken (from DSC2Pxxx-xxxC-xxxx)			
	M145				Type of EnDat encoder	0	0	65535
	M146				EnDat measuring step per turn (for rotary encoder) or in [mm] (for linear encoder)	0	0	65535
	M147				EnDat analog encoder period per turn (for rotary encoder) or in [mm] (for linear encoder)	0	0	65535
	M148				EnDat pulse number	0	0	65535
	M149				EnDat turn number	0	0	65535
	M171				Gives the state of the digital outputs of the controller. Takes DOUT (K171) and K37 into account	0	0	15
	M173				Real state of DSO-HIO analog output 1. Takes K173, K150, K151 and K154 to K157 into account	0	-8192	8191
	M174				Real state of DSO-HIO analog output 2. Takes K173, K150, K151 and K154 to K157 into account	0	-8192	8191
	M175				Real state of DSO-HIO analog output 3. Takes K173, K150, K151 and K154 to K157 into account	0	-8192	8191
	M176				Real state of DSO-HIO analog output 4. Takes K173, K150, K151 and K154 to K157 into account	0	-8192	8191
	M230				Real-time monitoring 1 slave to slave	0	0	4294967295
	M231				Real-time monitoring 2 slave to slave	0	0	4294967295
	M239				Encoder period	1	1	2147483647
	M240				Motor type	0	0	1
	M241				Encoder interpolation factor	0	0	32768
	M242				Controller quartz frequency [Hz]	30000000	30000000	30000000
	M243				Controller current loop time factor (cti). M243/M242 = time in second	1250	1250	1250
	M244				Controller position loop time factor (fti). M244/M242 = time in second	1250	1250	1250
	M245				Controller slow interrupt time factor (sti). M245/M242 = time in second	5000	5000	5000

17.2 Monitorings M for DSCDP

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP monitorings	M0				Theoretical position (dpi). Takes the scaling/mapping correction into account but does not take care of SET command and K50	0	-2147483648	2147483647
	M1				Real position (dpi). Takes the scaling/mapping correction into account but does not take care of SET command and K50	0	-2147483648	2147483647
	M2				Tracking error. This is the difference between M0 and M1	0	-2147483648	2147483647
	M3				Maximum tracking error during movement	0	0	2147483647
	M4				Offset between dpi and upi. [upi] = ([dpi] + M4) >>K50	0	-2147483648	2147483647
	M5				Offset due to the homing (dpi)	0	-2147483648	2147483647
	M6				Theoretical position (upi). Takes SET command, K50 and the scaling/mapping correction into account	0	-2147483648	2147483647
	M7				Real position (upi). Takes SET command, K50 and the scaling/mapping correction into account	0	-2147483648	2147483647
	M8				Theoretical position (dpi). Does not take SET command, K50 and scaling/mapping correction into account	0	-2147483648	2147483647
	M10				Theoretical velocity (dsi)	0	-2147483648	2147483647
	M11				Real velocity (dsi)	0	-2147483648	2147483647
	M12				Real position (upi) captured on DIN. Takes SET command, K50 and scaling/mapping correction into account	0	-2147483648	2147483647
	M13				Position given by the secondary encoder (dpi)	0	-2147483648	2147483647
	M14				Theoretical acceleration (dai)	0	-2147483648	2147483647
	M17				Reference value for mode K61=0, 1, 3, 4 and 36	0	-2147483648	2147483647
	M20				Real current in phase 1	0	-32768	32767
	M21				Real current in phase 2	0	-32768	32767
	M22				Real current in phase 3	0	-32768	32767
	M23				Current reference in phase 1	0	-32768	32767
	M24				Current reference in phase 2	0	-32768	32767
	M25				Current loop look-up table value of phase 1	0	-32768	32767
	M26				Current loop look-up table value of phase 2	0	-32768	32767
	M27				Current loop look-up table value of phase 3	0	-32768	32767
	M29				PWM value of phase 1	0	-512	511
	M30				Theoretical force after advanced filter 2	0	-32768	32767
	M31				Real force	0	-32768	32767
	M32				Theoretical force after K9 filter and before advanced filters	0	-32768	32767
	M36				Inferior position after a SLS command	0	-2147483648	2147483647
	M37				Superior position after a SLS command	0	-2147483648	2147483647
	M40				Analog encoder sine signal	0	-2048	2047
	M41				Analog encoder cosine signal	0	-2048	2047
	M43				Analog encoder sine^2 + cosine^2	0	0	4194304
	M44				Encoder limit switch	0	0	3
			1	0	Encoder limit switch ELS (L1 or L)			
			2	1	Encoder limit switch EHO (L2 or H)			

DSCDP monitorings

M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
M48				Digital Hall effect sensor signal	0	0	255
M50	DIN			Value of the 4 digital inputs: DIN1 and DIN2 (bits# 0 to 1), DIN9 and DIN10 (bits# 8 to 9). Caution: bits# 2 to 7 are set to 0	0	0	771
M55	XDIN			Optional board digital inputs (8 inputs for DSO-HIO)	0	0	255
M56	XAIN1			Optional board analog input 1 (for DSO-HIO)	0	-8192	8191
M57	XAIN2			Optional board analog input 2 (for DSO-HIO)	0	-8192	8191
M58	XAIN3			Optional board analog input 3 (for DSO-HIO)	0	-8192	8191
M59	XAIN4			Optional board analog input 4 (for DSO-HIO)	0	-8192	8191
M60	SD1			Drive status 1	8	0	4294967295
		1	0	The controller is in power on			
		2	1	This bit is set when the controller has been initialized once (first PWR.# = 1)			
		4	2	This bit is set when the controller has finished the homing process with success			
		8	3	Bit present, always 1			
		16	4	The motor is executing a trajectory			
		32	5	This bit is set when the motor is in the position/time window defined by K38 and K39			
		128	7	Controller is waiting			
		256	8	The controller is executing an internal sequence			
		512	9	The controller is in edition of sequence mode (EDI command). This mode allows the writing of S (sequence) register			
		1024	10	The controller is in error mode			
		2048	11	Trace busy flag is set during a register trace acquisition			
		8192	13	This bit is set during the homing process			
		8388608	23	Global warning			
M61	SD2			Drive status 2	0	0	4294967295
		1	0	Sequence error label pending			
		4	2	Position captured on digital input (see K182/K178/K179). This bit is set when the conditions of the digital input allow the capture of the position. It is reset when 1 is written in K182			
		256	8	User bit 0, could be modified by trigger functions or by K177			
		512	9	User bit 1, could be modified by trigger functions or by K177			
		1024	10	User bit 2, could be modified by trigger functions or by K177			
		2048	11	User bit 3, could be modified by trigger functions or by K177			
		4096	12	User bit 4, could be modified by trigger functions or by K177			
		8192	13	User bit 5, could be modified by trigger functions or by K177			
		16384	14	User bit 6, could be modified by trigger functions or by K177			
		32768	15	User bit 7, could be modified by trigger functions or by K177			

DSCDP monitorings

M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
M63				TEB status	524288	0	4294967295
		1	0	User word bit 0, could be modified by trigger functions or by K177			
		2	1	User word bit 1, could be modified by trigger functions or by K177			
		4	2	User word bit 2, could be modified by trigger functions or by K177			
		8	3	User word bit 3, could be modified by trigger functions or by K177			
		16	4	User word bit 4, could be modified by trigger functions or by K177			
		32	5	User word bit 5, could be modified by trigger functions or by K177			
		64	6	User word bit 6, could be modified by trigger functions or by K177			
		128	7	User word bit 7, could be modified by trigger functions or by K177			
		256	8	User word bit 8, could be modified by trigger functions or by K177			
		512	9	User word bit 9, could be modified by trigger functions or by K177			
		1024	10	User word bit 10, could be modified by trigger functions or by K177			
		2048	11	User word bit 11, could be modified by trigger functions or by K177			
		4096	12	User word bit 12, could be modified by trigger functions or by K177			
		8192	13	User word bit 13, could be modified by trigger functions or by K177			
		16384	14	User word bit 14, could be modified by trigger functions or by K177			
		32768	15	User word bit 15, could be modified by trigger functions or by K177			
		65536	16	The controller is in power on			
		524288	19	Bit present, always 1			
		1048576	20	The motor is executing a trajectory			
		2097152	21	This bit is set when the motor is in the position/time window defined by K38 and K39			
		8388608	23	The controller is in warning mode			
		16777216	24	The controller is executing an internal sequence			
		67108864	26	The controller is in error mode			
		134217728	27	Trace busy flag is set during a register trace acquisition			
		1073741824	30	Position captured according to the digital input (see K182/K178/K179). This bit is set when the conditions on the digital input allow the capture of the position. It is reset when 1 is written in K182			
M64				Gives the error code	0	0	255
		2		The current measured in phase 1 is greater than K83			
		3		The current measured in phase 2 is greater than K83			
		4		This occurs when M67 becomes greater than K85. This is a power protection (of the motor and/or the controller)			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP monitorings			5		The temperature of the controller is greater than 70°C. This is measured by a thermostat mounted on the heat sink			
			6		The power voltage (Vpower) in volt is greater than 425Volts			
			7		Power supply inrush or power voltage too low when motor power is enabled (for rack version, it occurs when inrush signal on d14 of JC15 is not at 0V; this signal is given by ETEL's power supply or must be given by the user in case of another power supply)			
			8		Error that occurs when the +5V of the controller becomes lower than 4.5V			
			10		Error in offset of current measurement. The offset measured if greater than 8% of the maximum current of the controller (M82)			
			13		The temperature sensor does not respond			
			16		The EnDat calculated position is too big. Decrease K77			
			17		Zero position of the EnDat encoder not found			
			18		Error on CRC from the EnDat			
			19		Error during the reading position of EnDat			
			20		The amplitudes of the analog encoder signals are too small			
			21		The encoder has lost the position acquisition, the frequency is too high			
			23		The tracking position error is greater than K30			
			24		The velocity is greater than K31			
			26		Error when power on with DIN1 is equal to 0 when K33 = 0			
			29		The temperature of the motor is too high, measured by temperature sensor connected to digital input according to K141			
			30		The controller generates this error when the controller reaches a limit switch during a movement (except during IND and SLS), if bit #0 of K32 is set			
			33		Error when the same label is defined several times. If the error appears, checks the sequence and if it is correct, erases sequence (NEW.#=1 command) and then download again the sequence			
			36		Error when the label of a jump or a call is greater than 511			
			37		Error when the controller executes a command at a sequence line number greater than 8190			
			38		This error occurs when the register number is greater than its maximum value			
			39		This error occurs when the stack is overflowed (256 times)			
			40		This error occurs when K79 as a wrong value			
			41		This error occurs when K89 as a wrong value			
			56		Time-out error in TEB communication. This error is checked only if bit#1 of K141 is set			
			59		There are several nodes that have the same number on the TEB			

DSCDP monitorings

M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		60		Error in homing process when leaving home switch or limit switch. Checks K48 parameter or DIN 2, 9 or 10. This error occurs when after the trip from K48 the home switch or limit switch is present			
		61		Error in multi-index homing process. Checks K43, K44 and K75 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)			
		62		Error in single index homing process. Checks K43 and K44 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)			
		63		Error in start synchro on input (STI command)			
		65		Error when the real position (M7) is out of the limits defined by K34 and K35 when bit#1 of K36 is set			
		66		Error when the reference position is out of the limits defined by K34 and K35 when bit#2 of K36 is set			
		67		Movement not possible because the controller is not in power on mode			
		68		PWR ON not possible because the controller is in interpolation mode (by ITP=1)			
		69		Homing is not possible due to the configuration of the controller and the setting			
		80		Framing error on EBL2			
		81		Overrun error on EBL2			
		82		Checksum error on EBL2 message			
		84		Input buffer full on EBL2			
		85		Bad CRC on EBL2 in received message in CRC Mode			
		88		Other axis error on EBL2			
		89		Overrun error with Macro			
		90		Violation error with Macro			
		91		Parity error with Macro			
		92		Underrun error with Macro			
		93		Synchro error with Macro			
		94		Error auxiliary channel second command Macro			
		95		Error auxiliary channel Macro aux 2			
		96		Error auxiliary channel Macro aux 3			
		97		Error auxiliary channel Macro aux 4			
		100		Optional board error 100. See optional board manual			
		101		Optional board error 101. See optional board manual			
		102		Optional board error 102. See optional board manual			
		103		Optional board error 103. See optional board manual			
		104		Optional board error 104. See optional board manual			
		105		Optional board error 105. See optional board manual			
		106		Optional board error 106. See optional board manual			
		107		Optional board error 107. See optional board manual			
		108		Optional board error 108. See optional board manual			
		109		Optional board error 109. See optional board manual			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP monitorings			110		Optional board error 110. See optional board manual			
			111		Optional board error 111. See optional board manual			
			112		Optional board error 112. See optional board manual			
			113		Optional board error 113. See optional board manual			
			114		Optional board error 114. See optional board manual			
			115		Optional board error 115. See optional board manual			
			116		This error is generated by the ERR command. In general, this command is sent by the DSMAX or the u-master			
			130		Hardware overcurrent. The current has reached the maximum admissible value by the controller			
			141		Watchdog error			
			144		The difference between the oscillator and quartz is too big			
			150		Bad measure during the phasing process when K90=1			
			151		Bad time measurement during phasing process when K90=1			
			152		Error when the controller is disabled during phasing process when K90 = 1 or 2			
			153		Too low force when K90=6			
			154		Too high force when K90=6			
			155		Too low time when K90=6			
			156		Time-out during AUT command			
			157		Error during the phasing process when K90=2			
			190		The controller has executed a save operation (SAV command)			
	M66				Gives the warning code	0	0	255
			1		This warning occurs when M67 is greater than K85/2			
			2		This warning occurs when the temperature of the controller is greater than 60°C			
			3		Power supply inrush or power voltage too low (for rack version, it occurs when inrush signal on d14 of JC15 is not at 0V. This signal is given by ETEL's power supply or must be given by the user in case of another power supply)			
			5		This warning occurs when the amplitude of the analog encoder signals are too small (2x the error)			
			6		This warning occurs when the tracking error is greater than K30/2			
			8		Wrong value coming from the digital Hall sensor			
			9		Loss of synchronization between the TEB signal and the controller interrupt			
			11		This warning occurs when the power voltage is greater than 425V and no motor in Power On			
	M67				Current loop i2t value. When M67 is greater than K85, the controller generates an I2T error (M64=4)	0	0	2147483647
	M70				Indicates the type of controller	7	7	7

DSCDP monitorings

M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
M71				Gives the software boot version of the controller. A special ETEL procedure allows the conversion of this value in the software boot version (format is the same than M72)	0	0	4294967295
M72	VER			Gives the firmware version of the controller. A special ETEL procedure allows the conversion of this value in the firmware version	0	0	4294967295
M73	SER			Gives the serial number of the controller	0	0	4294967295
M76				Optional board type	0	0	255
		0		No optional board			
		4		DSO-MAC optional Macro protocol			
		7		DSO-HIO optional board with 8 digital I/O and 4 analog I/O			
		16		DSO-CAN optional w/CANetel protocol			
		32		DSO-PRO optional Profibus protocol			
M77				Optional board boot revision	0	0	255
M78				Optional board firmware version	0	0	4294967295
M79				Optional board serial number	0	0	4294967295
M82				Controller maximum current. M82/100 gives the maximum current in Ampere	2500	1250	6667
M85				Gives the article number string. The 16 strings of the article number are read using the 4 depths of M85. Each depth shows 4 strings (in ASCII)	0	0	4294967295
M86				Gives the article number string of the optional board. This is available only with DSO-CAN, DSO-SER, DSO-PRO. The 16 strings of the article number are read using the 4 depths of M86. Each depth shows 4 strings (in ASCII)	0	0	4294967295
M87				Gives the axis number	1	0	30
M90				Temperature of the controller measured by a thermostat on the heat sink	0	-40	100
M95				Shows the strings on the software display. The 16 strings of the controller display are read using the 4 depths of M95. Each depth shows 4 string (in ASCII)	0	0	4294967295
M96				Gives the line number being executed in the sequence	0	0	8190
M110				Gives the interpolation mode of the position controller	0	0	255
		0		Interpolation mode disabled			
		1		Interpolation mode at every sti interrupt. Takes jerk time (IJT) and encoder scaling/mapping into account			
		2		Interpolation mode on fti interrupt. Does not take jerk time (IJT) and encoder scaling/mapping into account			
M145				Type of EnDat encoder	0	0	65535
M146				EnDat measuring step per turn (for rotary encoder) or in [mm] (for linear encoder)	0	0	65535
M147				EnDat analog encoder period per turn (for rotary encoder) or in [mm] (for linear encoder)	0	0	65535
M148				EnDat pulse number	0	0	65535
M149				EnDat turn number	0	0	65535
M171				Gives the state of the digital outputs of the controller. Takes DOUT (K171) and K37 into account	0	0	3
M173				Real state of DSO-HIO analog output 1. Takes K173, K150, K151 and K154 to K158 into account	0	-8192	8191

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDP	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDP monitorings	M174				Real state of DSO-HIO analog output 2. Takes K173, K150, K151 and K154 to K158 into account	0	-8192	8191
	M175				Real state of DSO-HIO analog output 3. Takes K173, K150, K151 and K154 to K158 into account	0	-8192	8191
	M176				Real state of DSO-HIO analog output 4. Takes K173, K150, K151 and K154 to K158 into account	0	-8192	8191
	M239				Encoder period	1	1	2147483647
	M240				Motor type	0	0	1
	M241				Encoder interpolation factor	0	0	32768
	M242				Controller quartz frequency [Hz]	30000000	30000000	30000000
	M243				Controller current loop time factor (cti). M243/M242 = time in second	1667	1667	1667
	M244				Controller fast interrupt time factor (fti). M244/M242 = time in second	1667	1667	1667
	M245				Controller slow interrupt time factor (sti). M245/M242 = time in second	15000	15000	15000

17.3 Monitorings M for DSCDL

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL monitorings	M0				Theoretical position (dpi). Takes the scaling/mapping correction into account but does not take care of SET command and K50	0	-2147483648	2147483647
	M1				Real position (dpi). Takes the scaling/mapping correction into account but does not take care of SET command and K50	0	-2147483648	2147483647
	M2				Tracking error. This is the difference between M0 and M1	0	-2147483648	2147483647
	M3				Maximum tracking error during movement	0	0	2147483647
	M4				Offset between dpi and upi. [upi] = ([dpi] + M4) >>K50	0	-2147483648	2147483647
	M5				Offset due to the homing (dpi)	0	-2147483648	2147483647
	M6				Theoretical position (upi). Takes SET command, K50 and the scaling/mapping correction into account	0	-2147483648	2147483647
	M7				Real position (upi). Takes SET command, K50 and the scaling/mapping correction into account	0	-2147483648	2147483647
	M8				Theoretical position (dpi). Does not take SET command, K50 and scaling/mapping correction into account	0	-2147483648	2147483647
	M10				Theoretical velocity (dsi)	0	-2147483648	2147483647
	M11				Real velocity (dsi)	0	-2147483648	2147483647
	M12				Real position (upi) captured on DIN. Takes SET command, K50 and scaling/mapping correction into account	0	-2147483648	2147483647
	M13				Position given by the secondary encoder (dpi)	0	-2147483648	2147483647
	M14				Theoretical acceleration (dai)	0	-2147483648	2147483647
	M17				Reference value for mode K61=0, 1, 3, 4 and 36	0	-2147483648	2147483647
	M20				Real current in phase 1	0	-32768	32767
	M21				Real current in phase 2	0	-32768	32767
	M23				Current reference in phase 1	0	-32768	32767
	M24				Current reference in phase 2	0	-32768	32767

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL monitorings	M25				Current loop look-up table value of phase 1	0	-32768	32767
	M26				Current loop look-up table value of phase 2	0	-32768	32767
	M29				Voltage consigne phase 1	0	-2147483648	2147483647
	M30				Theoretical force after advanced filter 2	0	-32768	32767
	M31				Real force	0	-32768	32767
	M32				Theoretical force after K9 filter and before advanced filters	0	-32768	32767
	M36				Inferior position after a SLS command	0	-2147483648	2147483647
	M37				Superior position after a SLS command	0	-2147483648	2147483647
	M40				Analog encoder sine signal	0	-32768	32767
	M41				Analog encoder cosine signal	0	-32768	32767
	M43				Analog encoder sine^2 + cosine^2	0	0	4194304
	M44				Encoder limit switch	0	0	3
			1	0	Encoder limit switch ELS (L1 or L)			
			2	1	Encoder limit switch EHO (L2 or H)			
	M50	DIN			Value of the 4 digital inputs: DIN1 and DIN2 (bits# 0 to 1), DIN9 and DIN10 (bits# 8 to 9). Caution: bits# 2 to 7 are set to 0	0	0	771
	M51	AI1			Analog input	0	-32768	32767
	M55	XDIN			Optional board digital inputs (8 inputs for DSO-HIO)	0	0	255
	M56	XAIN1			Optional board analog input 1 (for DSO-HIO)	0	-8192	8191
	M57	XAIN2			Optional board analog input 2 (for DSO-HIO)	0	-8192	8191
	M58	XAIN3			Optional board analog input 3 (for DSO-HIO)	0	-8192	8191
	M59	XAIN4			Optional board analog input 4 (for DSO-HIO)	0	-8192	8191
	M60	SD1			Drive status 1	8	0	4294967295
			1	0	The controller is in power on			
			2	1	This bit is set when the controller has been initialized once (first PWR.# = 1)			
			4	2	This bit is set when the controller has finished the homing process with success			
			8	3	Bit present, always 1			
			16	4	The motor is executing a trajectory			
			32	5	This bit is set when the motor is in the position/time window defined by K38 and K39			
			128	7	Controller is waiting			
			256	8	The controller is executing an internal sequence			
			512	9	The controller is in edition of sequence mode (EDI command). This mode allows the writing of S (sequence) register			
			1024	10	The controller is in error mode			
			2048	11	Trace busy flag is set during a register trace acquisition			
			8192	13	This bit is set during the homing process			
			8388608	23	Global warning			
	M61	SD2			Drive status 2	0	0	4294967295
			1	0	Sequence error label pending			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL monitorings			4	2	Position captured on digital input (see K182/K178/ K179). This bit is set when the conditions of the digital input allow the capture of the position. It is reset when 1 is written in K182			
			256	8	User bit 0, could be modified by trigger functions or by K177			
			512	9	User bit 1, could be modified by trigger functions or by K177			
			1024	10	User bit 2, could be modified by trigger functions or by K177			
			2048	11	User bit 3, could be modified by trigger functions or by K177			
			4096	12	User bit 4, could be modified by trigger functions or by K177			
			8192	13	User bit 5, could be modified by trigger functions or by K177			
			16384	14	User bit 6, could be modified by trigger functions or by K177			
			32768	15	User bit 7, could be modified by trigger functions or by K177			
	M63				TEB status	524288	0	4294967295
			1	0	User word bit 0, could be modified by trigger functions or by K177			
			2	1	User word bit 1, could be modified by trigger functions or by K177			
			4	2	User word bit 2, could be modified by trigger functions or by K177			
			8	3	User word bit 3, could be modified by trigger functions or by K177			
			16	4	User word bit 4, could be modified by trigger functions or by K177			
			32	5	User word bit 5, could be modified by trigger functions or by K177			
			64	6	User word bit 6, could be modified by trigger functions or by K177			
			128	7	User word bit 7, could be modified by trigger functions or by K177			
			256	8	User word bit 8, could be modified by trigger functions or by K177			
			512	9	User word bit 9, could be modified by trigger functions or by K177			
			1024	10	User word bit 10, could be modified by trigger functions or by K177			
			2048	11	User word bit 11, could be modified by trigger functions or by K177			
			4096	12	User word bit 12, could be modified by trigger functions or by K177			
			8192	13	User word bit 13, could be modified by trigger functions or by K177			
			16384	14	User word bit 14, could be modified by trigger functions or by K177			
			32768	15	User word bit 15, could be modified by trigger functions or by K177			
			65536	16	The controller is in power on			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL monitorings			524288	19	Bit present, always 1			
			1048576	20	The motor is executing a trajectory			
			2097152	21	This bit is set when the motor is in the position/time window defined by K38 and K39			
			8388608	23	The controller is in warning mode			
			16777216	24	The controller is executing an internal sequence			
			67108864	26	The controller is in error mode			
			134217728	27	Trace busy flag is set during a register trace acquisition			
			1073741824	30	Position captured according to the digital input (see K182/K178/K179). This bit is set when the conditions on the digital input allow the capture of the position. It is reset when 1 is written in K182			
	M64				Gives the error code	0	0	255
			2		The current measured in phase 1 is greater than K83			
			3		The current measured in phase 2 is greater than K83			
			4		This occurs when M67 becomes greater than K85. This is a power protection (of the motor and/or the controller)			
			5		The temperature of the controller is greater than 70°C. This is measured by a thermostat mounted on the heat sink			
			6		The absolute value of the power voltage (Vpower) in volt is greater than K149/100			
			8		Error that occurs when the +5V of the controller becomes lower than 4.5V			
			9		The absolute value of the power voltage (Vpower) in volt is lower than K147/100			
			10		Error in offset of current measurement. The offset measured if greater than 8% of the maximum current of the controller (M82)			
			13		The temperature sensor does not respond			
			16		The EnDat calculated position is too big. Decrease K77			
			17		Zero position of the EnDat encoder not found			
			18		Error on CRC from the EnDat			
			19		Error during the reading position of EnDat			
			20		The amplitudes of the analog encoder signals are too small			
			21		The encoder has lost the position acquisition, the frequency is too high			
			23		The tracking position error is greater than K30			
			24		The velocity is greater than K31			
			26		Error when power on with DIN1 is equal to 0 when K33 = 0			
			29		The temperature of the motor is too high, measured by temperature sensor connected to digital input according to K141			
			30		The controller generates this error when the controller reaches a limit switch during a movement (except during IND and SLS), if bit #0 of K32 is set			
			33		Error when the same label is defined several times. If the error appears, checks the sequence and if it is correct, erases sequence (NEW.#=1 command) and then download again the sequence			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL monitorings			35		The fuse of the encoder supply is broken			
			36		Error when the label of a jump or a call is greater than 511			
			37		Error when the controller executes a command at a sequence line number greater than 8190			
			38		This error occurs when the register number is greater than its maximum value			
			39		This error occurs when the stack is overflowed (256 times)			
			40		This error occurs when K79 as a wrong value			
			41		This error occurs when K89 as a wrong value			
			56		Time-out error in TEB communication. This error is checked only if bit#1 of K141 is set			
			59		There are several nodes that have the same number on the TEB			
			60		Error in homing process when leaving home switch or limit switch. Checks K48 parameter or DIN 2, 9 or 10. This error occurs when after the trip from K48 the home switch or limit switch is present			
			61		Error in multi-index homing process. Checks K43, K44 and K75 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)			
			62		Error in single index homing process. Checks K43 and K44 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)			
			63		Error in start synchro on input (STI command)			
			65		Error when the real position (M7) is out of the limits defined by K34 and K35 when bit#1 of K36 is set			
			66		Error when the reference position is out of the limits defined by K34 and K35 when bit#2 of K36 is set			
			67		Movement not possible because the controller is not in power on mode			
			68		PWR ON not possible because the controller is in interpolation mode (by ITP=1)			
			69		Homing is not possible due to the configuration of the controller and the setting			
			80		Framing error on EBL2			
			81		Overrun error on EBL2			
			82		Checksum error on EBL2 message			
			84		Input buffer full on EBL2			
			85		Bad CRC on EBL2 in received message in CRC Mode			
			88		Other axis error on EBL2			
			100		Optional board error 100. See optional board manual			
			101		Optional board error 101. See optional board manual			
			102		Optional board error 102. See optional board manual			
			103		Optional board error 103. See optional board manual			
			104		Optional board error 104. See optional board manual			
			105		Optional board error 105. See optional board manual			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL monitorings			106		Optional board error 106. See optional board manual			
			107		Optional board error 107. See optional board manual			
			108		Optional board error 108. See optional board manual			
			109		Optional board error 109. See optional board manual			
			110		Optional board error 110. See optional board manual			
			111		Optional board error 111. See optional board manual			
			112		Optional board error 112. See optional board manual			
			113		Optional board error 113. See optional board manual			
			114		Optional board error 114. See optional board manual			
			115		Optional board error 115. See optional board manual			
			116		This error is generated by the ERR command. In general, this command is sent by the DSMAX or the u-master			
			130		Hardware overcurrent. The current has reached the maximum admissible value by the controller			
			131		Error on the power bridge			
			141		Watchdog error			
			144		The difference between the oscillator and quartz is too big			
			152		Error when the controller is disabled during phasing process when K90 = 2			
			153		Too low force when K90=6			
			154		Too high force when K90=6			
			155		Too low time when K90=6			
			156		Time-out during AUT command			
			157		Error during the phasing process when K90=2			
			190		The controller has executed a save operation (SAV command)			
	M66				Gives the warning code	0	0	255
			1		This warning occurs when M67 is greater than K85/2			
			2		This warning occurs when the temperature of the controller is greater than 60°C			
			4		This warning occurs when the power voltage (Vpower) in volt is greater than K148/100			
			5		This warning occurs when the amplitude of the analog encoder signals are too small (2x the error)			
			6		This warning occurs when the tracking error is greater than K30/2			
			9		Loss of synchronization between the TEB signal and the controller interrupt			
			10		This warning occurs when the power voltage (Vpower) in volt is lower than K146/100			
	M67				Current loop i2t value. When M67 is greater than K85, the controller generates an I2T error (M64=4)	0	0	2147483647
	M70				Indicates the type of controller	8	8	8
	M71				Gives the software boot version of the controller. A special ETEL procedure allows the conversion of this value in the software boot version (format is the same than M72)	0	0	4294967295

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL monitorings	M72	VER			Gives the firmware version of the controller. A special ETEL procedure allows the conversion of this value in the firmware version	0	0	4294967295
	M73	SER			Gives the serial number of the controller	0	0	4294967295
	M76				Optional board type	0	0	255
			0		No optional board			
			7		DSO-HIO optional board with 8 digital I/O and 4 analog I/O			
	M77				Optional board boot revision	0	0	255
	M78				Optional board firmware version	0	0	4294967295
	M79				Optional board serial number	0	0	4294967295
	M82				Controller maximum current. M82/100 gives the maximum current in Ampere	600	600	600
	M85				Gives the article number string. The 16 strings of the article number are read using the 4 depths of M85. Each depth shows 4 strings (in ASCII)	0	0	4294967295
	M87				Gives the axis number	1	0	30
	M90				Temperature of the controller measured by a thermostat on the heat sink	0	-40	100
	M91				DC positive power voltage (+Vpower) [V*100]. M91/100 gives the tension in Volt	0	0	5000
	M92				DC negative power voltage (-Vpower) [V*100]. M92/100 gives the tension in Volt	0	-5000	0
	M95				Shows the strings on the software display. The 16 strings of the controller display are read using the 4 depths of M95. Each depth shows 4 string (in ASCII)	0	0	4294967295
	M96				Gives the line number being executed in the sequence	0	0	8190
	M110				Gives the interpolation mode of the position controller	0	0	255
			0		Interpolation mode disabled			
			1		Interpolation mode at every sti interrupt. Takes jerk time (IJT) and encoder scaling/mapping into account			
			2		Interpolation mode on fti interrupt. Does not take jerk time (IJT) and encoder scaling/mapping into account			
	M140				Fuse status	0	0	1
			1	0	Fuse F2 broken			
	M145				Type of EnDat encoder	0	0	65535
	M146				EnDat measuring step per turn (for rotary encoder) or in [mm] (for linear encoder)	0	0	65535
	M147				EnDat analog encoder period per turn (for rotary encoder) or in [mm] (for linear encoder)	0	0	65535
	M148				EnDat pulse number	0	0	65535
	M149				EnDat turn number	0	0	65535
	M171				Gives the state of the digital outputs of the controller. Takes DOUT (K171) and K37 into account	0	0	3
	M173				Real state of DSO-HIO analog output 1. Takes K173, K150, K151 and K154 to K158 into account	0	-8192	8192
	M174				Real state of DSO-HIO analog output 2. Takes K173, K150, K151 and K154 to K158 into account	0	-8192	8192
	M175				Real state of DSO-HIO analog output 3. Takes K173, K150, K151 and K154 to K158 into account	0	-8192	8192

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDL	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDL monitorings	M176				Real state of DSO-HIO analog output 4. Takes K173, K150, K151 and K154 to K158 into account	0	-8192	8192
	M239				Encoder period	1	1	2147483647
	M240				Motor type	0	0	1
	M241				Encoder interpolation factor	0	0	32768
	M242				Controller quartz frequency [Hz]	30000000	30000000	30000000
	M243				Controller current loop time factor (cti). M243/M242 = time in second	417	417	417
	M244				Controller fast interrupt time factor (fti). M244/M242 = time in second	1667	1667	1667
	M245				Controller slow interrupt time factor (sti). M245/M242 = time in second	15000	15000	15000

17.4 Monitorings M for DSCDM

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM monitorings	M0				Theoretical position (dpi). Takes the scaling/mapping correction into account but does not take care of SET command and K50	0	-2147483648	2147483647
	M1				Real position (dpi). Takes the scaling/mapping correction into account but does not take care of SET command and K50	0	-2147483648	2147483647
	M2				Tracking error. This is the difference between M0 and M1	0	-2147483648	2147483647
	M3				Maximum tracking error during movement	0	0	2147483647
	M4				Offset between dpi and upi. [upi] = ([dpi] + M4) >>K50	0	-2147483648	2147483647
	M5				Offset due to the homing (dpi)	0	-2147483648	2147483647
	M6				Theoretical position (upi). Takes SET command, K50 and the scaling/mapping correction into account	0	-2147483648	2147483647
	M7				Real position (upi). Takes SET command, K50 and the scaling/mapping correction into account	0	-2147483648	2147483647
	M8				Theoretical position (dpi). Does not take SET command, K50 and scaling/mapping correction into account	0	-2147483648	2147483647
	M10				Theoretical velocity (dsi)	0	-2147483648	2147483647
	M11				Real velocity (dsi)	0	-2147483648	2147483647
	M12				Real position (upi) captured on DIN. Takes SET command, K50 and scaling/mapping correction into account	0	-2147483648	2147483647
	M13				Position given by the secondary encoder (dpi)	0	-2147483648	2147483647
	M14				Theoretical acceleration (dai)	0	-2147483648	2147483647
	M17				Reference value for mode K61=0, 1, 3, 4 and 36	0	-2147483648	2147483647
	M20				Real current in phase 1	0	-32768	32767
	M21				Real current in phase 2	0	-32768	32767
	M22				Real current in phase 3	0	-32768	32767
	M23				Current reference in phase 1	0	-32768	32767
	M24				Current reference in phase 2	0	-32768	32767
	M25				Current loop look-up table value of phase 1	0	-32768	32767
	M26				Current loop look-up table value of phase 2	0	-32768	32767

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM monitorings	M27				Current loop look-up table value of phase 3	0	-32768	32767
	M29				PWM value of phase 1	0	-512	511
	M30				Theoretical force after advanced filter 2	0	-32768	32767
	M31				Real force	0	-32768	32767
	M32				Theoretical force after K9 filter and before advanced filters	0	-32768	32767
	M36				Inferior position after a SLS command	0	-2147483648	2147483647
	M37				Superior position after a SLS command	0	-2147483648	2147483647
	M40				Analog encoder sine signal	0	-2048	2047
	M41				Analog encoder cosine signal	0	-2048	2047
	M43				Analog encoder sine^2 + cosine^2	0	0	4194304
	M44				Encoder limit switch	0	0	3
			1	0	Encoder limit switch ELS (L1 or L)			
			2	1	Encoder limit switch EHO (L2 or H)			
	M48				Digital Hall effect sensor signal	0	0	255
	M50	DIN			Value of the digital inputs: DIN1 (bits# 0), DIN9 and DIN10 (bits# 8 to 9). Caution: bits# 1 to 7 are set to 0	0	0	769
	M60	SD1			Drive status 1	8	0	4294967295
			1	0	The controller is in power on			
			2	1	This bit is set when the controller has been initialized once (first PWR.# = 1)			
			4	2	This bit is set when the controller has finished the homing process with success			
			8	3	Bit present, always 1			
			16	4	The motor is executing a trajectory			
			32	5	This bit is set when the motor is in the position/time window defined by K38 and K39			
			128	7	Controller is waiting			
			256	8	The controller is executing an internal sequence			
			512	9	The controller is in edition of sequence mode (EDI command). This mode allows the writing of S (sequence) register			
			1024	10	The controller is in error mode			
			2048	11	Trace busy flag is set during a register trace acquisition			
			8192	13	This bit is set during the homing process			
			8388608	23	Global warning			
	M61	SD2			Drive status 2	0	0	4294967295
			1	0	Sequence error label pending			
			4	2	Position captured on digital input (see K182/K178/K179). This bit is set when the conditions of the digital input allow the capture of the position. It is reset when 1 is written in K182			
			256	8	User bit 0, could be modified by trigger functions or by K177			
			512	9	User bit 1, could be modified by trigger functions or by K177			
			1024	10	User bit 2, could be modified by trigger functions or by K177			

DSCDM monitorings

M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
		2048	11	User bit 3, could be modified by trigger functions or by K177			
		4096	12	User bit 4, could be modified by trigger functions or by K177			
		8192	13	User bit 5, could be modified by trigger functions or by K177			
		16384	14	User bit 6, could be modified by trigger functions or by K177			
		32768	15	User bit 7, could be modified by trigger functions or by K177			
M63				TEB status	524288	0	4294967295
		1	0	User word bit 0, could be modified by trigger functions or by K177			
		2	1	User word bit 1, could be modified by trigger functions or by K177			
		4	2	User word bit 2, could be modified by trigger functions or by K177			
		8	3	User word bit 3, could be modified by trigger functions or by K177			
		16	4	User word bit 4, could be modified by trigger functions or by K177			
		32	5	User word bit 5, could be modified by trigger functions or by K177			
		64	6	User word bit 6, could be modified by trigger functions or by K177			
		128	7	User word bit 7, could be modified by trigger functions or by K177			
		256	8	User word bit 8, could be modified by trigger functions or by K177			
		512	9	User word bit 9, could be modified by trigger functions or by K177			
		1024	10	User word bit 10, could be modified by trigger functions or by K177			
		2048	11	User word bit 11, could be modified by trigger functions or by K177			
		4096	12	User word bit 12, could be modified by trigger functions or by K177			
		8192	13	User word bit 13, could be modified by trigger functions or by K177			
		16384	14	User word bit 14, could be modified by trigger functions or by K177			
		32768	15	User word bit 15, could be modified by trigger functions or by K177			
		65536	16	The controller is in power on			
		524288	19	Bit present, always 1			
		1048576	20	The motor is executing a trajectory			
		2097152	21	This bit is set when the motor is in the position/time window defined by K38 and K39			
		8388608	23	The controller is in warning mode			
		16777216	24	The controller is executing an internal sequence			
		67108864	26	The controller is in error mode			
		134217728	27	Trace busy flag is set during a register trace acquisition			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM monitorings			1073741824	30	Position captured according to the digital input (see K182/K178/K179). This bit is set when the conditions on the digital input allow the capture of the position. It is reset when 1 is written in K182			
	M64				Gives the error code	0	0	255
			2		The current measured in phase 1 is greater than K83			
			3		The current measured in phase 2 is greater than K83			
			4		This occurs when M67 becomes greater than K85. This is a power protection (of the motor and/or the controller)			
			5		The temperature of the controller is greater than 75°C. This is measured by a thermostat mounted on the board			
			7		The power voltage (Vpower) in volt is lower than 10V			
			8		Error that occurs when the +5V of the controller becomes lower than 4.5V			
			10		Error in offset of current measurement. The offset measured if greater than 8% of the maximum current of the controller (M82)			
			13		The temperature sensor does not respond			
			16		The EnDat calculated position is too big. Decrease K77			
			17		Zero position of the EnDat encoder not found			
			18		Error on CRC from the EnDat			
			19		Error during the reading position of EnDat			
			20		The amplitudes of the analog encoder signals are too small			
			21		The encoder has lost the position acquisition, the frequency is too high			
			23		The tracking position error is greater than K30			
			24		The velocity is greater than K31			
			26		Error when power on with DIN1 is equal to 0 when K33 = 0			
			29		The temperature of the motor is too high, measured by temperature sensor connected to digital input according to K141			
			30		The controller generates this error when the controller reaches a limit switch during a movement (except during IND and SLS), if bit #0 of K32 is set			
			33		Error when the same label is defined several times. If the error appears, checks the sequence and if it is correct, erases sequence (NEW.#=1 command) and then download again the sequence			
			35		The fuse of the encoder supply is broken			
			36		Error when the label of a jump or a call is greater than 511			
			37		Error when the controller executes a command at a sequence line number greater than 8190			
			38		This error occurs when the register number is greater than its maximum value			
			39		This error occurs when the stack is overflowed (256 times)			
			40		This error occurs when K79 as a wrong value			
			41		This error occurs when K89 as a wrong value			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM monitorings			56		Time-out error in TEB communication. This error is checked only if bit#1 of K141 is set			
			59		There are several nodes that have the same number on the TEB			
			60		Error in homing process when leaving home switch or limit switch. Checks K48 parameter or DIN 2, 9 or 10. This error occurs when after the trip from K48 the home switch or limit switch is present			
			61		Error in multi-index homing process. Checks K43, K44 and K75 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)			
			62		Error in single index homing process. Checks K43 and K44 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)			
			63		Error in start synchro on input (STI command)			
			65		Error when the real position (M7) is out of the limits defined by K34 and K35 when bit#1 of K36 is set			
			66		Error when the reference position is out of the limits defined by K34 and K35 when bit#2 of K36 is set			
			67		Movement not possible because the controller is not in power on mode			
			68		PWR ON not possible because the controller is in interpolation mode (by ITP=1)			
			69		Homing is not possible due to the configuration of the controller and the setting			
			80		Framing error on EBL2			
			81		Overrun error on EBL2			
			82		Checksum error on EBL2 message			
			84		Input buffer full on EBL2			
			85		Bad CRC on EBL2 in received message in CRC Mode			
			88		Other axis error on EBL2			
			116		This error is generated by the ERR command. In general, this command is sent by the DSMAX or the u-master			
			130		Hardware overcurrent. The current has reached the maximum admissible value by the controller			
			141		Watchdog error			
			144		The difference between the oscillator and quartz is too big			
			150		Bad measure during the phasing process when K90=1			
			151		Bad time measurement during phasing process when K90=1			
			152		Error when the controller is disabled during phasing process when K90 = 1 or 2			
			153		Too low force when K90=6			
			154		Too high force when K90=6			
			155		Too low time when K90=6			
			156		Time-out during AUT command			

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM monitorings			157		Error during the phasing process when K90=2			
			190		The controller has executed a save operation (SAV command)			
	M66				Gives the warning code	0	0	255
			1		This warning occurs when M67 is greater than K85/2			
			2		This warning occurs when the temperature of the controller is greater than 65°C			
			3		This warning occurs when the power voltage (Vpower) in volt is lower than 10V			
			5		This warning occurs when the amplitude of the analog encoder signals are too small (2x the error)			
			6		This warning occurs when the tracking error is greater than K30/2			
			8		Wrong value coming from the digital Hall sensor			
			9		Loss of synchronization between the TEB signal and the controller interrupt			
	M67				Current loop i2t value. When M67 is greater than K85, the controller generates an I2T error (M64=4)	0	0	2147483647
	M70				Indicates the type of controller	10	10	10
	M71				Gives the software boot version of the controller. A special ETEL procedure allows the conversion of this value in the software boot version (format is the same than M72)	0	0	4294967295
	M72	VER			Gives the firmware version of the controller. A special ETEL procedure allows the conversion of this value in the firmware version	0	0	4294967295
	M73	SER			Gives the serial number of the controller	0	0	4294967295
	M82				Controller maximum current. M82/100 gives the maximum current in Ampere	2500	1250	6667
	M85				Gives the article number string. The 16 strings of the article number are read using the 4 depths of M85. Each depth shows 4 strings (in ASCII)	0	0	4294967295
	M87				Gives the axis number	1	0	30
	M90				Temperature of the controller measured by a thermostat on the heat sink	0	-40	100
	M95				Shows the strings on the software display. The 16 strings of the controller display are read using the 4 depths of M95. Each depth shows 4 string (in ASCII)	0	0	4294967295
	M96				Gives the line number being executed in the sequence	0	0	8190
	M110				Gives the interpolation mode of the position controller	0	0	255
			0		Interpolation mode disabled			
			1		Interpolation mode at every sti interrupt. Takes jerk time (IJT) and encoder scaling/mapping into account			
			2		Interpolation mode on fti interrupt. Does not take jerk time (IJT) and encoder scaling/mapping into account			
	M140				Fuse status	0	0	1
			1	0	Fuse F2 broken			
	M145				Type of EnDat encoder	0	0	65535
	M146				EnDat measuring step per turn (for rotary encoder) or in [mm] (for linear encoder)	0	0	65535
	M147				EnDat analog encoder period per turn (for rotary encoder) or in [mm] (for linear encoder)	0	0	65535

	M	Alias	Values	Bit # <P1>	Comment for monitorings M and <P1> of the DSCDM	Def. Val <P1>	Min. Val <P1>	Max. Val <P1>
DSCDM monitorings	M148				EnDat pulse number	0	0	65535
	M149				EnDat turn number	0	0	65535
	M171				Gives the state of the digital outputs of the controller. Takes DOUT (K171) and K37 into account	0	0	7
	M239				Encoder period	1	1	2147483647
	M240				Motor type	0	0	1
	M241				Encoder interpolation factor	0	0	32768
	M242				Controller quartz frequency [Hz]	30000000	30000000	30000000
	M243				Controller current loop time factor (cti). M243/M242 = time in second	1667	1667	1667
	M244				Controller fast interrupt time factor (fti). M244/M242 = time in second	1667	1667	1667
	M245				Controller slow interrupt time factor (sti). M245/M242 = time in second	15000	15000	15000

18. Warnings reference lists

18.1 Warnings for DSC2P and DSC2V

M66	Displayed message	Description
1	W I2T WARNING	This warning occurs when M67 is greater than K85/2
2	W OVER TEMPERAT	This warning occurs when the temperature of the controller is greater than 60°C
3	W INRUSH P.SUPPLY	Power supply inrush or power voltage too low (for rack version, it occurs when inrush signal on d14 of JC15 is not at 0V; this signal is given by ETEL's power supply or must be given by the user in case of another power supply)
4	W OVER VOLTAGE	This warning occurs when the power voltage (Vpower) in volt is greater than K148/100
5	W ENCODE AMPLITU	This warning occurs when the amplitude of the analog encoder signals is too small (2x the error)
6	TRACKING WARNING	This warning occurs when the tracking error is greater than K30/2
8	W DIGIT. HALL	Wrong value coming from the digital Hall sensor
9	WuCONTRO LSYNCHRO	Loss of synchronization between the TEB signal and the controller interrupt
10	W UNDER VOLTAGE	This warning occurs when the power voltage (Vpower) in volt is lower than K147/100

18.2 Warnings for DSCDP

M66	Displayed message	Description
1	W I2T WARNING	This warning occurs when M67 is greater than K85/2
2	W OVER TEMPERAT	This warning occurs when the temperature of the controller is greater than 60°C
3	W INRUSH P.SUPPLY	Power supply inrush or power voltage too low (for rack version, it occurs when inrush signal on d14 of JC15 is not at 0V; this signal is given by ETEL's power supply or must be given by the user in case of another power supply)
5	W ENCODE AMPLITU	This warning occurs when the amplitude of the analog encoder signals is too small (2x the error)
6	TRACKING WARNING	This warning occurs when the tracking error is greater than K30/2
8	W DIGIT. HALL	Wrong value coming from the digital Hall sensor
9	WuCONTRO LSYNCHRO	Loss of synchronization between the TEB signal and the controller interrupt
11	W OVER VOLTAGE	This warning occurs when the power voltage is greater than 425V and no motor in Power On

18.3 Warnings for DSCDL

M66	Displayed message	Description
1	W I2T WARNING	This warning occurs when M67 is greater than K85/2
2	W OVER TEMPERAT	This warning occurs when the temperature of the controller is greater than 60°C
4	W OVER VOLTAGE	This warning occurs when the power voltage (Vpower) in volt in greater than K148/100
5	W ENCODE AMPLITU	This warning occurs when the amplitude of the analog encoder signals is too small (2x the error)
6	TRACKING WARNING	This warning occurs when the tracking error is greater than K30/2
9	WuCONTRO LSYNCHRO	Loss of synchronization between the TEB signal and the controller interrupt
10	W UNDER VOLTAGE	This warning occurs when the power voltage (Vpower) in volt is lower than K146/100

18.4 Warnings for DSCDM

M66	Displayed message	Description
1	W I2T WARNING	This warning occurs when M67 is greater than K85/2
2	W OVER TEMPERAT	This warning occurs when the temperature of the controller is greater than 65°C
3	W UNDER VOLTAGE	Power voltage too low
5	W ENCODE AMPLITU	This warning occurs when the amplitude of the analog encoder signals is too small (2x the error)
6	TRACKING WARNING	This warning occurs when the tracking error is greater than K30/2
8	W DIGIT. HALL	Wrong value coming from the digital Hall sensor
9	WuCONTRO LSYNCHRO	Loss of synchronization between the TEB signal and the controller interrupt

19. Errors reference lists

Note: **Troubleshooting is permitted only for ETEL technicians and agreed distributors!**

The points to check in the **errors reference list** are indicated as follows (possible reasons for the error are listed, but the list is non-exhaustive):

Enc = **x**	Error may be due to the **encoder** and its cable.
Mot = **x**	Error may be due to the **motor** and its cable.
Hrd = **x**	Error may be due to the **hardware of the controller**.
Hrd	Error may be due to the a part of the hardware of the controller (F7= fuse7, DSP = Sharc,...)
K value(s)	Error may be due to a bad setting of the listed parameter(s).
PS = **x**	Error may be due to the **power supply** (DSO-PWS).
TEB = **x**	Error may be due to the **Turbo-ETEL-Bus** communication protocol.
EBL2 = **x**	Error may be due to the **ETEL-Bus-Lite 2** communication protocol.
Other	Error may be due to the reason described in the cell.
SW Res = **x**	It is **possible to reset** the error by software (RST command).
HW Res = **x**	It is r**ecommended or compulsory to reset** the error by hardware (RSD command or switch off/on)
Brk = OFF/ON	OFF or ON means that this error activates or deactivates the dynamic braking (when used).

19.1 Errors for DSC2P and DSC2V

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSC2P and DSC2V errors	2	OVER CURRENT1	The current measured in phase 1 is greater than K83		x	x	K60, K80, K81, K82, K83, K98					x		OFF
	3	OVER CURRENT2	The current measured in phase 2 is greater than K83		x	x	K60, K80, K81, K82, K83, K98					x		OFF
	4	I2T ERROR	This occurs when M67 becomes greater than K85. This is a power protection (of the motor and/or the controller)				K1, K2, K4, K52, K53, K56, K84, K85				Friction / duty cycle	x		OFF
	5	OVER TEM-PERAT	The temperature of the controller is greater than 70°C. This is measured by a thermostat mounted on the heat sink								Heat evacuation	x		ON
	6	OVER VOLTAGE	The power voltage (Vpower) in volt is greater than K149/100				K149	x				x		OFF
	7	INRUSH P.SUPPLY	Power supply inrush or power voltage too low when motor power is enabled (for rack version, it occurs when inrush signal on d14 of JC15 is not at 0V; this signal is given by ETEL's power supply or must be given by the user in case of another power supply)					x				x		ON
	8	+5V SHUTDOWN	Error that occurs when the +5V of the controller becomes lower than 4.5V			x		x					x	ON
	9	UNDER VOLTAGE	The power voltage (Vpower) in volt is lower than K147/100				K147					x		ON
	10	OFFSET CURERROR	Error in offset of current measurement. The offset measured is greater than 8% of the maximum current of the controller (M82)			x							x	OFF
	13	SENSOR TEMP ERR	The temperature sensor does not respond			x						x		OFF

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSC2P and DSC2V errors	16	ENDAT OVERFLOW	The calculated EnDat position is too big. Decrease K77	x			K77						x	ON
	17	ENDAT POS LOST	Zero position of the EnDat encoder not found	x		F3					Bad cable type or cable too long		x	ON
	18	ENDAT BAD CRC	Error on CRC from the EnDat	x		F3					Bad cable type or cable too long		x	ON
	19	ENDAT POS ERR	The absolute position of EnDat is wrong	x		F3					Bad cable type or cable too long		x	ON
	20	ENCODER AMPLITUD	The amplitudes of the analog encoder signals are too small	x		F3					Speed too high, bad shield con.		x	ON
	21	ENCODER POS LOST	The encoder has lost the position acquisition, the frequency is too high	x		F3					Speed too high, bad shield con.		x	ON
	23	TRACKING ERROR	The tracking position error is greater than K30	x		F4	K30, K52, K53, K56				Friction	x		ON
	24	OVER SPEED	The velocity is greater than K31	x			K31, K52, K53, K56				SW	x		ON
	26	POWER OFF/ON	Error when power on with DIN1 is equal to 0 when K33 = 0			x	K33				I/O	x		ON
	29	MOTOR OVERTEMP	The temperature of the motor is too high, measured by temperature sensor connected to TSD pin (see Hardware Manual)		x		K141					x		OFF
	30	SWITCH LIMIT	The controller generates this error when the controller reaches a limit switch during a movement (except during IND and SLS) if bit#0 of K32 is set				K32, K40				I/O, shorted limit switch	x		ON
	33	MULT LAB ERROR	Error when the same label is defined many times. If the error appears, checks the sequence and if correct, erases sequence (NEW.#=1) and download again the sequence. It could be possible that a part of an old sequence is still present in the controller								Same label # used several times. May appear when an old sequence has not been erased with a NEW cmd.	x		ON
	35	ENCODER FUSE KO	The fuse for the encoder is broken	x		F3					Encoder cable	x		ON
	36	LABEL ERROR	Error when the label of a jump or a call is greater than 511								Sequence bad programming.	x		ON
	37	MAX SEQ LINE ERR	Error when the controller executes a command at a sequence line number greater than 8190								Sequence bad programming.	x		ON
	38	REGISTER NUM ERR	This error occurs when the register number is greater than its maximum value									x		ON
	39	STACK OVERFLOW	This error occurs when the stack is overflowed (256 times)										x	ON
	40	K79 BAD VALUE	This error occurs when K79 as a wrong value				K79						x	ON
	41	K89 BAD VALUE	This error occurs when K89 as a wrong value				K89						x	ON
	52	NO SLAVE PRESENT	No slave present on the TEB. The command or parameter axis number is wrong (axis does not exist) on the u-master chain						x		Bad TEB command, sequence bad programming	x		ON

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSC2P and DSC2V errors	54	SELFTEST TEB ERR	TEB has not been initialized correctly in u-Master mode (K170=1 at the power on of the controller)						x		Bad TEB connection, sequence bad programming	x		ON
	56	TIMEOUT TEB ERR	The controller does not receive TEB command any more. This error is enabled by setting bit#1of K141			x	K141		x		Bad TEB connection	x		ON
	59	BAD NODE TEB ERR	There are several nodes that have the same number on the TEB			x			x		Duplicated node number			ON
	60	LEAVEREF ERROR	Error in homing process when leaving home switch or limit switch. Checks K48 parameter or DIN 2, 9 or 10. This error occurs when after the trip from K48 the home switch or limit switch is present			F3 F4	K40, K41, K42, K45, K48					x		ON
	61	MULT IDX SEARCH	Error in multi-index homing process. Checks K43, K44 and K75 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)	x		F3 F4	K30, K31, K40, K41, K42, K43, K44, K45					x		ON
	62	SING IDX SEARCH	Error in single index homing process. Checks K43 and K44 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)	x		F3 F4	K30, K31, K40, K41, K42, K43, K44, K45					x		ON
	63	SYNCHRO START	Error in start synchro on input (STI command)				K164				SW, user sequence	x		ON
	65	OUT OF STROKE	Error when the real position (M7) is out of the limits defined by K34 and K35 when bit #1 of K36 is set				K34, K35, K36					x		ON
	66	REF OUT OFSTROKE	Error when the reference position is out of the limits defined by K34 and K35 when bit #2 of K36 is set				K34, K35, K36					x		ON
	67	MVT NOT POSSIBLE	Movement not possible because the controller is not in power on mode									x		ON
	68	NO PWRON IN ITP	PWR ON not possible because the controller is in interpolation mode (by ITP=1)									x		ON
	69	HOME NOT POSSIBLE	Homing is not possible due to the configuration of the controller and the setting			x	K58 K40					x		ON
	80	EBUSLITE FRAMING	Framing error on EBL2			x	K195			x	SW	x		ON
	81	EBUSLITE OVERRUN	Overrun error on EBL2			x				x		x		ON
	82	EBUSLITE CHECKSUM	Checksum error on EBL2 message			x				x		x		ON
	84	EBUSLITE INBUFFER	Input buffer full on EBL2									x		ON
	85	EBUSLITE BAD CRC	Bad CRC on EBL2 in received message in CRC Mode			x				x	Noise on line, bad user program	x		ON

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSC2P and DSC2V errors	88	EBUSLITO THERAXIS	Other axis error on EBL2							x		x		ON
	89	MACRO ER OverRun	Overrun error with MACRO									x		ON
	90	MACRO ER Violatio	Violation error with MACRO									x		ON
	91	MACRO ER Parity	Parity error with MACRO									x		ON
	92	MACRO ER UnderRun	Underrun error with MACRO									x		ON
	93	MACRO ER SyncLost	Synchro with MACRO is lost									x		ON
	94	MACRO ER Aux 2cmd	Error auxiliary chanel second command MACRO									x		ON
	95	MACRO ER Aux 2	Error auxiliary chanel MACRO error 2									x		ON
	96	MACRO ER Aux 3	Error auxiliary chanel MACRO error 3									x		ON
	97	MACRO ER Aux 4	Error auxiliary chanel MACRO error 4									x		ON
	100-115	XXXXXXXX XXXXXXXX	See optional board's manual									x		OFF
	116	EXTERNAL ERROR	This error is generated by the ERR command. In general, this command is sent by the DSMAX									x		ON
	130	HARDWARE OVERCURR	Hardware overcurrent. The current has reached the maximum admissible value by the controller		x	x					Motor cable, external relay,…		x	OFF
	141	FPGA ERROR 2	DSP watchdog error			x							x	OFF
	144	QUARTZ OSCILLAT	Difference between the oscillator and quartz is too big			x							x	OFF
	150	INITIALI MOTOR 1	Bad measure during initialization process when K90=1		x		K90, K91, K98				Motor cable	x		OFF
	151	INITIALI MOTOR 2	Bad time measurement during initialization process when K90=1		x		K90, K91, K98				Motor cable	x		OFF
	152	INITIALI POWER OF	Error when the controller is disabled during initialization process when K90=1 or 2									x		OFF
	153	INITIALI LOW CUR	Too low force when K90 = 6. Increase K91				K91, K101					x		OFF
	154	INITIALI HIGH CUR	Too high force when K90 = 6. Decrease K91				K91, K101					x		OFF
	155	INITIALI LOW TIME	Too low time when K90 = 6. Increase K101				K91, K101					x		OFF
	156	TIMEOUT AUT CMD	Time-out during AUT command	x	x		K90 to K98 K79				Motor or encoder cable	x		OFF
	157	INIT BAD PHASING	Error during the initialization process when K90=2	x	x		K90 to K98 K79				Motor or encoder cable	x		OFF
	190	SWITCH OFF & ON	The controller has executed a save operation (SAV command)										x	OFF

19.2 Errors for DSCDP

DSCDP errors

M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
2	OVER CURRENT1	The current measured in phase 1 is greater than K83		x	x	K60, K80, K81, K82, K83, K98					x		OFF
3	OVER CURRENT2	The current measured in phase 2 is greater than K83		x	x	K60, K80, K81, K82, K83, K98					x		OFF
4	I2T ERROR	This occurs when M67 becomes greater than K85. This is a power protection (of the motor and/or the controller)				K1, K2, K4, K52, K53, K56, K84, K85				Friction / duty cycle	x		OFF
5	OVER TEMPERAT	The temperature of the controller is greater than 70°C. This is measured by a thermostat mounted on the heat sink								Heat evacuation	x		ON
6	OVER VOLTAGE	The power voltage (Vpower) in volt is greater than 425V and one motor in Power ON					x				x		OFF
7	INRUSH P.SUPPLY	Power supply inrush or power voltage too low when motor power is enabled (for rack version, it occurs when inrush signal on d14 of JC15 is not at 0V; this signal is given by ETEL's power supply or must be given by the user in case of another power supply)					x				x		ON
8	+5V SHUTDOWN	Error that occurs when the +5V of the controller becomes lower than 4.5V			x		x					x	ON
10	OFFSET CURERROR	Error in offset of current measurement. The offset measured is greater than 8% of the maximum current of the controller (M82)			x							x	OFF
13	SENSOR TEMP ERR	The temperature sensor does not respond			x						x		OFF
16	ENDAT OVERFLOW	The calculated EnDat position is too big. Decrease K77	x			K77						x	ON
17	ENDAT POS LOST	Zero position of the EnDat encoder not found	x		F3					Bad type cable or cable too long		x	ON
18	ENDAT BAD CRC	Error on CRC from the EnDat	x		F3					Bad type cable or cable too long		x	ON
19	ENDAT POS ERR	The absolute position of EnDat is wrong	x		F3					Bad type cable or cable too long		x	ON
20	ENCODER AMPLITUD	The amplitudes of the analog encoder signals are too small	x		F3					Speed too high, bad shield con.		x	ON
21	ENCODER POS LOST	The encoder has lost the position acquisition, the frequency is too high	x		F3					Speed too high, bad shield con.		x	ON
23	TRACKING ERROR	The tracking position error is greater than K30	x		F4	K30, K52, K53, K56				Friction	x		ON
24	OVER SPEED	The velocity is greater than K31	x			K31, K52, K53, K56				SW	x		ON
26	POWER OFF/ ON	Error when power on with DIN1 is equal to 0 when K33 = 0			x	K33				I/O	x		ON

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDP errors	29	MOTOR OVERTEMP	The temperature of the motor is too high, measured by temperature sensor connected to digital input according to K141		x		K141					x		OFF
	30	SWITCH LIMIT	The controller generates this error when the controller reaches a limit switch during a movement (except during IND and SLS) if bit#0 of K32 is set				K32, K40				I/O, shorted limit switch	x		ON
	33	MULT LAB ERROR	Error when the same label is defined many times. If the error appears, checks the sequence and if correct, erases sequence (NEW.#=1) and download again the sequence. It could be possible that a part of an old sequence is still present in the controller								Same label # used several times. May appear when an old sequence has not been erased with a NEW cmd.	x		ON
	36	LABEL ERROR	Error when the label of a jump or a call is greater than 511								Sequence bad programming.	x		ON
	37	MAX SEQ LINE ERR	Error when the controller executes a command at a sequence line number greater than 8190								Sequence bad programming.	x		ON
	38	REGISTER NUM ERR	This error occurs when the register number is greater than its maximum value									x		ON
	39	STACK OVERFLOW	This error occurs when the stack is overflowed (256 times)										x	ON
	40	K79 BAD VALUE	This error occurs when K79 as a wrong value				K79						x	ON
	41	K89 BAD VALUE	This error occurs when K89 as a wrong value				K89						x	ON
	56	TIMEOUT TEB ERR	The controller does not receive TEB command any more. This error is enabled by setting bit#1of K141			x	K141		x		Bad TEB connection	x		ON
	59	BAD NODE TEB ERR	There are several nodes that have the same number on the TEB			x			x		Duplicated node number			ON
	60	LEAVEREF ERROR	Error in homing process when leaving home switch or limit switch. Checks K48 parameter or DIN 2, 9 or 10. This error occurs when after the trip from K48 the home switch or limit switch is present			F3 F4	K40, K41, K42, K45, K48					x		ON
	61	MULT IDX SEARCH	Error in multi-index homing process. Checks K43, K44 and K75 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)	x		F3 F4	K30, K31, K40, K41, K42, K43, K44, K45					x		ON
	62	SING IDX SEARCH	Error in single index homing process. Checks K43 and K44 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)	x		F3 F4	K30, K31, K40, K41, K42, K43, K44, K45					x		ON
	63	SYNCHRO START	Error in start synchro on input (STI command)				K164				SW, user sequence	x		ON

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDP errors	65	OUT OF STROKE	Error when the real position (M7) is out of the limits defined by K34 and K35 when bit #1 of K36 is set				K34, K35, K36					x		ON
	66	REF OUT OFSTROKE	Error when the reference position is out of the limits defined by K34 and K35 when bit #2 of K36 is set				K34, K35, K36					x		ON
	67	MVT NOT POSSIBLE	Movement not possible because the controller is not in power on mode									x		ON
	68	NO PWRON IN ITP	PWR ON not possible because the controller is in interpolation mode (by ITP=1)									x		ON
	69	HOME NOT POSSIBLE	Homing is not possible due to the configuration of the controller and the setting			x	K58 K40					x		ON
	80	EBUSLITE FRAMING	Framing error on EBL2			x				x	SW	x		ON
	81	EBUSLITE OVERRUN	Overrun error on EBL2			x				x		x		ON
	82	EBUSLITE CHECKSUM	Checksum error on EBL2 message			x				x		x		ON
	84	EBUSLITE INBUFFER	Input buffer full on EBL2									x		ON
	85	EBUSLITE BAD CRC	Bad CRC on EBL2 in received message in CRC Mode			x				x	Noise on line, bad user program	x		ON
	88	EBUSLITO THERAXIS	Other axis error on EBL2							x		x		ON
	89	MACRO ER OverRun	Overrun error with MACRO									x		ON
	90	MACRO ER Violatio	Violation error with MACRO									x		ON
	91	MACRO ER Parity	Parity error with MACRO									x		ON
	92	MACRO ER UnderRun	Underrun error with MACRO									x		ON
	93	MACRO ER SyncLost	Synchro with MACRO is lost									x		ON
	94	MACRO ER Aux 2cmd	Error auxiliary chanel second command MACRO									x		ON
	95	MACRO ER Aux 2	Error auxiliary chanel MACRO error 2									x		ON
	96	MACRO ER Aux 3	Error auxiliary chanel MACRO error 3									x		ON
	97	MACRO ER Aux 4	Error auxiliary chanel MACRO error 4									x		ON
	100-115	XXXXXXXX XXXXXXXX										x		OFF
	116	EXTERNAL ERROR	This error is generated by the ERR command; in general, this command is sent by the DSMAX		x	x						x		ON
	130	HARDWARE OVERCURR	Hardware overcurrent. The current has reached the maximum admissible value by the controller		x	x					Motor cable, external relay,…		x	OFF
	141	FPGA ERROR 2	DSP watchdog error			x							x	OFF

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDP errors	144	QUARTZ OSCILLAT	Difference between the oscillator and quartz is too big			x							x	OFF
	150	INITIALI MOTOR 1	Bad measure during initialization process when K90=1		x		K90, K91, K98				Motor cable	x		OFF
	151	INITIALI MOTOR 2	Bad time measurement during initialization process when K90=1		x		K90, K91, K98				Motor cable	x		OFF
	152	INITIALI POWER OF	Error when the controller is disabled during initialization process when K90=1 or 2									x		OFF
	153	INITIALI LOW CUR	Too low force when K90 = 6. Increase K91				K91, K101					x		OFF
	154	INITIALI HIGH CUR	Too high force when K90 = 6. Decrease K91				K91, K101					x		OFF
	155	INITIALI LOW TIME	Too low time when K90 = 6. Increase K101				K91, K101					x		OFF
	156	TIMEOUT AUT CMD	Time-out during AUT command	x	x		K90 to K98 K79				Motor or encoder cable	x		OFF
	157	INIT BAD PHASING	Error during the initialization process when K90=2	x	x		K90 to K98 K79				Motor or encoder cable	x		OFF
	190	SWITCH OFF & ON	The controller has executed a save operation (SAV command)										x	OFF

19.3 Errors for DSCDL

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDL errors	2	OVER CURRENT1	The current measured in phase 1 is greater than K83		x	x	K60, K80, K81, K83, K98					x		OFF
	3	OVER CURRENT2	The current measured in phase 2 is greater than K83		x	x	K60, K80, K81, K83, K98					x		OFF
	4	I2T ERROR	This occurs when M67 becomes greater than K85. This is a power protection (of the motor and/or the controller)				K1, K2, K4, K52, K53, K56, K84, K85				Friction / duty cycle	x		OFF
	5	OVER TEMPERAT	The temperature of the controller is greater than 70°C. This is measured by a thermostat mounted on the heat sink								Heat evacuation	x		OFF
	6	OVER VOLTAGE	The power voltage (Vpower) in volt in greater than K149/100				K149					x		OFF
	8	+5V SHUTDOWN	Error that occurs when the +5V of the controller becomes lower than 4.5V			x		x					x	OFF
	9	UNDER VOLTAGE	The power voltage (Vpower) in volt is lower than K147/100				K147					x		OFF
	10	OFFSET CURERROR	Error in offset of current measurement. The offset measured is greater than 8% of the maximum current of the controller (M82)			x							x	OFF
	13	SENSOR TEMP ERR	The temperature sensor does not respond			x						x		OFF
	16	ENDAT OVERFLOW	The calculated EnDat position is too big. Decrease K77	x			K77						x	ON

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDL errors	17	ENDAT POS LOST	Zero position of the EnDat encoder not found	x		F3					Bad type cable or cable too long		x	ON
	18	ENDAT BAD CRC	Error on CRC from the EnDat	x		F3					Bad type cable or cable too long		x	ON
	19	ENDAT POS ERR	The absolute position of EnDat is wrong	x		F3					Bad type cable or cable too long		x	ON
	20	ENCODER AMPLITUD	The amplitudes of the analog encoder signals are too small	x		F3					Speed too high, bad shield con.		x	OFF
	21	ENCODER POS LOST	The encoder has lost the position acquisition, the frequency is too high	x		F3					Speed too high, bad shield con.		x	OFF
	23	TRACKING ERROR	The tracking position error is greater than K30	x		F4	K30, K52, K53, K56				Friction	x		OFF
	24	OVER SPEED	The velocity is greater than K31	x			K31, K52, K53, K56				SW	x		OFF
	26	POWER OFF/ ON	Error when power on with DIN1 is equal to 0 when K33 = 0			x	K33				I/O	x		OFF
	29	MOTOR OVERTEMP	The temperature of the motor is too high, measured by temperature sensor, connected to digital input according to K141		x		K141							OFF
	30	SWITCH LIMIT	The controller generates this error when the controller reaches a limit switch during a movement (except during IND and SLS) if bit#0 of K32 is set				K32, K40				I/O, shorted limit switch	x		OFF
	33	MULT LAB ERROR	Error when the same label is defined many times. If the error appears, checks the sequence and if correct, erases sequence (NEW.#=1) and download again the sequence. It could be possible that a part of an old sequence is still present in the controller								Same label # used several times. May appear when an old sequence has not been erased with a NEW cmd.	x		OFF
	35	ENCODER FUSE KO	The fuse for the encoder is broken	x		F2					Encoder cable	x		OFF
	36	LABEL ERROR	Error when the label of a jump or a call is greater than 511								Sequence bad programming.	x		OFF
	37	MAX SEQ LINE ERR	Error when the controller executes a command at a sequence line number greater than 8190								Sequence bad programming.	x		OFF
	38	REGISTER NUM ERR	This error occurs when the register number is greater than its maximum value									x		OFF
	39	STACK OVERFLOW	This error occurs when the stack is overflowed (256 times)										x	OFF
	40	K79 BAD VALUE	This error occurs when K79 as a wrong value				K79						x	ON
	41	K89 BAD VALUE	This error occurs when K89 as a wrong value				K89						x	ON
	56	TIMEOUT TEB ERR	The controller does not receive TEB command any more. This error is enabled by setting bit#1of K141			x	K141				Bad TEB connection	x		OFF
	59	BAD NODE TEB ERR	There are several nodes that have the same number on the TEB			x			x		Duplicated node number			ON

DSCDL errors

M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
60	LEAVEREF ERROR	Error in homing process when leaving home switch or limit switch. Checks K48 parameter or DIN 2, 9 or 10. This error occurs when after the trip from K48 the home switch or limit switch is present			F3 F4	K40, K41, K42, K45, K48					x		OFF
61	MULT IDX SEARCH	Error in multi-index homing process. Checks K43, K44 and K75 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)	x		F3 F4	K30, K31, K40, K41, K42, K45					x		OFF
62	SING IDX SEARCH	Error in single index homing process. Checks K43 and K44 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)	x		F3 F4	K30, K31, K40, K41, K42, K45					x		OFF
63	SYNCHRO START	Error in start synchro on input (STI command)				K164				SW, user sequence	x		OFF
65	OUT OF STROKE	Error when the real position (M7) is out of the limits defined by K34 and K35 when bit #1 of K36 is set				K34, K35, K36					x		OFF
66	REF OUT OFSTROKE	Error when the reference position is out of the limits defined by K34 and K35 when bit #2 of K36 is set				K34, K35, K36					x		ON
67	MVT NOT POSSIBLE	Movement not possible because the controller is not in power on mode									x		OFF
68	NO PWRON IN ITP	PWR ON not possible because the controller is in interpolation mode (by ITP=1)									x		OFF
69	HOME NOT POSSIBLE	Homing is not possible due to the configuration of the controller and the setting			x	K58 K40					x		ON
80	EBUSLITE FRAMING	Framing error on EBL2			x				x	SW	x		OFF
81	EBUSLITE OVERRUN	Overrun error on EBL2			x				x		x		OFF
82	EBUSLITE CHECKSUM	Checksum error on EBL2 message			x				x		x		OFF
84	EBUSLITE INBUFFER	Input buffer full on EBL2									x		OFF
85	EBUSLITE BAD CRC	Bad CRC on EBL2 in received message in CRC Mode			x				x	Noise on line, bad user program	x		OFF
88	EBUSLITO THERAXIS	Other axis error on EBL2							x		x		OFF
100-115	XXXXXXXX XXXXXXXX										x		OFF
116	EXTERNAL ERROR	This error is generated by the ERR command; in general, this command is sent by the DSMAX		x	x						x		OFF
130	HARDWARE OVERCURR	Hardware overcurrent. The current has reached the maximum admissible value by the controller		x	x					Motor cable, external relay,…		x	OFF

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDL errors	131	BRIDGE FAILURE	Error on the bridge		x	x					Power Bridge		x	OFF
	141	FPGA ERROR 2	DSP watchdog error			x							x	OFF
	144	QUARTZ OSCILLAT	Difference between the oscillator and quartz is too big			x							x	OFF
	152	INITIALI POWER OF	Error when the controller is disabled during initialization process when K90 = 2									x		OFF
	153	INITIALI LOW CUR	Too low force when K90 = 6. Increase K91				K91, K101					x		OFF
	154	INITIALI HIGH CUR	Too high force when K90 = 6. Decrease K91				K91, K101					x		OFF
	155	INITIALI LOW TIME	Too low time when K90 = 6. Increase K101				K91, K101					x		OFF
	156	TIMEOUT AUT CMD	Time-out during AUT command	x	x		K90 to K98 K79				Motor or encoder cable	x		OFF
	157	INIT BAD PHASING	Error during the initialization process when K90=2	x	x		K90 to K98 K79				Motor or encoder cable	x		OFF
	190	SWITCH OFF & ON	The controller has executed a save operation (SAV command)										x	OFF

19.4 Errors for DSCDM

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDM errors	2	OVER CURRENT1	The current measured in phase 1 is greater than K83		x	x	K60, K80, K81, K82, K83, K98					x		OFF
	3	OVER CURRENT2	The current measured in phase 2 is greater than K83		x	x	K60, K80, K81, K82, K83, K98					x		OFF
	4	I2T ERROR	This occurs when M67 becomes greater than K85. This is a power protection (of the motor and/or the controller)				K1, K2, K4, K52, K53, K56, K84, K85				Friction / duty cycle	x		OFF
	5	OVER TEMPERAT	The temperature of the controller is greater than 75°C. This is measured by a thermostat mounted on the board								Heat evacuation	x		ON
	7	UNDER VOLTAGE	The power voltage (Vpower) in volt is less than 10V									x		ON
	8	+5V SHUTDOWN	Error that occurs when the +5V of the controller becomes lower than 4.5V			x		x					x	ON
	10	OFFSET CURERROR	Error in offset of current measurement. The offset measured is greater than 8% of the maximum current of the controller (M82)			x							x	OFF
	13	SENSOR TEMP ERR	The temperature sensor does not respond			x						x		OFF
	16	ENDAT OVERFLOW	The calculated EnDat position is too big. Decrease K77	x			K77						x	ON
	17	ENDAT POS LOST	Zero position of the EnDat encoder not found	x		F3					Bad type cable or cable too long		x	ON

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDM errors	18	ENDAT BAD CRC	Error on CRC from the EnDat	x		F3					Bad type cable or cable too long		x	ON
	19	ENDAT POS ERR	The absolute position of EnDat is wrong	x		F3					Bad type cable or cable too long		x	ON
	20	ENCODER AMPLITUD	The amplitudes of the analog encoder signals are too small	x		F3					Speed too high, bad shield con.		x	ON
	21	ENCODER POS LOST	The encoder has lost the position acquisition, the frequency is too high	x		F3					Speed too high, bad shield con.		x	ON
	23	TRACKING ERROR	The tracking position error is greater than K30	x		F4	K30, K52, K53, K56				Friction	x		ON
	24	OVER SPEED	The velocity is greater than K31	x			K31, K52, K53, K56				SW	x		ON
	26	POWER OFF/ ON	Error when power on with DIN1 is equal to 0 when K33 = 0			x	K33				I/O	x		ON
	29	MOTOR OVERTEMP	The temperature of the motor is too high, measured by temperature sensor connected to digital input according to K141		x		K141							OFF
	30	SWITCH LIMIT	The controller generates this error when the controller reaches a limit switch during a movement (except during IND and SLS) if bit#0 of K32 is set				K32, K40				I/O, shorted limit switch	x		ON
	33	MULT LAB ERROR	Error when the same label is defined many times. If the error appears, checks the sequence and if correct, erases sequence (NEW.#=1) and download again the sequence; it could be possible that a part of an old sequence is still present in the controller								Same label # used several times. May appear when an old sequence has not been erased with a NEW cmd.	x		ON
	35	ENCODER FUSE KO	The fuse for the encoder is broken	x		F2					Encoder cable	x		OFF
	36	LABEL ERROR	Error when the label of a jump or a call is greater than 511								Sequence bad programming.	x		ON
	37	MAX SEQ LINE ERR	Error when the controller executes a command at a sequence line number greater than 8190								Sequence bad programming.	x		ON
	38	REGISTER NUM ERR	This error occurs when the register number is greater than its maximum value									x		ON
	39	STACK OVERFLOW	This error occurs when the stack is overflowed (256 times)										x	ON
	40	K79 BAD VALUE	This error occurs when K79 as a wrong value				K79						x	ON
	41	K89 BAD VALUE	This error occurs when K89 as a wrong value				K89						x	ON
	56	TIMEOUT TEB ERR	The controller does not receive TEB command any more. This error is enabled by setting bit#1of K141			x	K141		x		Bad TEB con-nection	x		ON
	59	BAD NODE TEB ERR	There are several nodes that have the same number on the TEB			x			x		Duplicated node number			ON

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDM errors	60	LEAVEREF ERROR	Error in homing process when leaving home switch or limit switch. Checks K48 parameter or DIN 2, 9 or 10. This error occurs when after the trip from K48 the home switch or limit switch is present			F3 F4	K40, K41, K42, K45, K48					x		ON
	61	MULT IDX SEARCH	Error in multi-index homing process. Checks K43, K44 and K75 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)	x		F3 F4	K30, K31, K40, K41, K42, K43, K44, K45					x		ON
	62	SING IDX SEARCH	Error in single index homing process. Checks K43 and K44 parameters. This error occurs when the motor has reached the two mechanical end stops (detections are done by K43 and K44) without having finished the homing process (for linear motors)	x		F3 F4	K30, K31, K40, K41, K42, K43, K44, K45					x		ON
	63	SYNCHRO START	Error in start synchro on input (STI command)				K164				SW, user sequence	x		ON
	65	OUT OF STROKE	Error when the real position (M7) is out of the limits defined by K34 and K35 when bit #1 of K36 is set				K34, K35, K36					x		ON
	66	REF OUT OFSTROKE	Error when the reference position is out of the limits defined by K34 and K35 when bit #2 of K36 is set				K34, K35, K36					x		ON
	67	MVT NOT POSSIBLE	Movement not possible because the controller is not in power on mode									x		ON
	68	NO PWRON IN ITP	PWR ON not possible because the controller is in interpolation mode (by ITP=1)									x		ON
	69	HOME NOT POSSIBLE	Homing is not possible due to the configuration of the controller and the setting			x	K58 K40					x		ON
	80	EBUSLITE FRAMING	Framing error on EBL2			x	K195			x	SW	x		ON
	81	EBUSLITE OVERRUN	Overrun error on EBL2			x				x		x		ON
	82	EBUSLITE CHECKSUM	Checksum error on EBL2 message			x				x		x		ON
	84	EBUSLITE INBUFFER	Input buffer full on EBL2									x		ON
	85	EBUSLITE BAD CRC	Bad CRC on EBL2 in received message in CRC Mode			x				x	Noise on line, bad user program	x		ON
	88	EBUSLITO THERAXIS	Other axis error on EBL2							x		x		OFF
	116	EXTERNAL ERROR	This error is generated by the ERR command; in general, this command is sent by the DSMAX		x	x						x		ON
	130	HARDWARE OVERCURR	Hardware overcurrent. The current has reached the maximum admissible value by the controller		x	x					Motor cable, external relay,…		x	OFF
	141	FPGA ERROR 2	DSP watchdog error			x							x	OFF

	M64	Displayed message	Description	Enc	Mot	Hrd	Bad K	PS	TEB	EBL2	Other	SW res	Hw res	Brk
DSCDM errors	144	QUARTZ OSCILLAT	Difference between the oscillator and quartz IS too big			x							x	OFF
	150	INITIALI MOTOR 1	Bad measure during initialization process when K90=1		x		K90, K91, K98				Motor cable	x		OFF
	151	INITIALI MOTOR 2	Bad time measurement during initialization process when K90=1		x		K90, K91, K98				Motor cable	x		OFF
	152	INITIALI POWER OF	Error when the controller is disabled during initialization process when K90=1 or 2									x		OFF
	153	INITIALI LOW CUR	Too low force when K90 = 6. Increase K91				K91, K101					x		OFF
	154	INITIALI HIGH CUR	Too high force when K90 = 6. Decrease K91				K91, K101					x		OFF
	155	INITIALI LOW TIME	Too low time when K90 = 6. Increase K101				K91, K101					x		OFF
	156	TIMEOUT AUT CMD	Time-out during AUT command	x	x		K90 to K98 K79				Motor or encoder cable	x		OFF
	157	INIT BAD PHASING	Error during the initialization process when K90=2	x	x		K90 to K98 K79				Motor or encoder cable	x		OFF
	190	SWITCH OFF & ON	The controller has executed a save operation (SAV command)										x	OFF

20. Units conversion

The controller does not accept ISO unit values system (meter, kilogram, amp, second) but those from a system with **increment**. In this chapter, the conversion formulae are given to calculate an increment value from an ISO one and inversely. First, the **cinematic** quantities such as distance, speed, acceleration and jerk will be looked at and then **time** quantities units.

20.1 Cinematic quantities units

There are always two cinematic quantities systems side by side in the controller. The first is the user increment [ui], and the second is the drive increment [di]. Some values have to be given in user increment and others in drive increment. All internal calculations done by the controller are in drive increment because the resolution is better.

In the next part of the manual, the different cinematic quantities are abbreviated like described in the following table, otherwise the complete conversion formula will be given from case to case.

	Notation	Signification	Equivalent in the ISO system
Linear motors	[upi]	User Position Increment	[m]
	[usi]	User Speed Increment	[m/s]
	[uai]	User Acceleration Increment	[m/s^2]
	[dpi]	Drive Position Increment	[m]
	[dsi]	Drive Speed Increment	[m/s]
	[dai]	Drive Acceleration Increment	[m/s^2]
Rotary motors	[rupi]	Rotary User Position Increment	[turn]
	[rusi]	Rotary User Speed Increment	[turn/s]
	[ruai]	Rotary User Acceleration Increment	[turn/s^2]
	[rdpi]	Rotary Drive Position Increment	[turn]
	[rdsi]	Rotary Drive Speed Increment	[turn/s]
	[rdai]	Rotary Drive Acceleration Increment	[turn/s^2]

Remark: Read also §13.3.10 about parameter **K50** for units scales definition.

Offset between [dpi] and [upi]: [upi] = ([dpi] + M4) / 2^{K50}

20.1.1 Linear motors

20.1.1.1 User increments, linear motors

Abbreviation used in the table below: PCod = Encoder period [m]
h = Slow time interrupt [s] (refer to §4.1 for more information)

Unit	Quantity	Conversion formulae	Concerned quantities
[upi]	**Distance** user position increment	**[m] → [upi]** $\text{Distance [upi]} = \text{Distance [m]} \cdot \frac{1024 \cdot 2^{K77}}{PCod \cdot 2^{K50}}$ (for analog encoder) $\text{Distance [upi]} = \text{Distance [m]} \cdot \frac{64 \cdot 2^{K69}}{PCod \cdot 2^{K50}}$ (for TTL encoder) **[upi] → [m]** $\text{Distance [m]} = \text{Distance [upi]} \cdot \frac{PCod \cdot 2^{K50}}{1024 \cdot 2^{K77}}$ (for analog encoder) $\text{Distance [m]} = \text{Distance [upi]} \cdot \frac{PCod \cdot 2^{K50}}{64 \cdot 2^{K69}}$ (for TTL encoder)	• POS (K210), SET, STE, WTP • K27, K34, K35, K39, K45, K46, K47, K48, K208 • M6, M7, M12, M13, M36, M37
[usi]	**Speed** user speed increment	**[m/s] →[usi]** $\text{Speed [usi]} = \text{Speed [m/s]} \cdot \frac{256 \cdot 1024 \cdot 2^{K77} \cdot h}{PCod \cdot 2^{K50}}$ (for analog encoder) $\text{Speed [usi]} = \text{Speed [m/s]} \cdot \frac{256 \cdot 64 \cdot 2^{K69} \cdot h}{PCod \cdot 2^{K50}}$ (for TTL encoder) **[usi] → [m/s]** $\text{Speed [m/s]} = \text{Speed [usi]} \cdot \frac{PCod \cdot 2^{K50}}{256 \cdot 1024 \cdot 2^{K77} \cdot h}$ (for analog encoder) $\text{Speed [m/s]} = \text{Speed [usi]} \cdot \frac{PCod \cdot 2^{K50}}{256 \cdot 64 \cdot 2^{K69} \cdot h}$ (for TTL encoder)	• SPD (K211) • K41
[uai]	**Acceleration** user acceleration increment	**[m/s^2] → [uai]** $\text{Acceleration [uai]} = \text{Acceleration [m/}s^2] \cdot \frac{65536 \cdot 1024 \cdot 2^{K77} \cdot h^2}{PCod \cdot 2^{K50}}$ (for analog encoder) $\text{Acceleration [uai]} = \text{Acceleration [m/}s^2] \cdot \frac{65536 \cdot 64 \cdot 2^{K69} \cdot h^2}{PCod \cdot 2^{K50}}$ (for TTL encoder) **[uai] → [m/s^2]** $\text{Acceleration [m/}s^2] = \text{Acceleration [uai]} \cdot \frac{PCod \cdot 2^{K50}}{65536 \cdot 1024 \cdot 2^{K77} \cdot h^2}$ (for analog encoder) $\text{Acceleration [m/}s^2] = \text{Acceleration [uai]} \cdot \frac{PCod \cdot 2^{K50}}{65536 \cdot 64 \cdot 2^{K69} \cdot h^2}$ (for TTL encoder)	• ACC (K212) • K42, K206

Remark: 1024 * 2^{K77} and 64 * 2^{K69} can be replaced by the value given by monitoring M241.

Example:

If a 12 cm movement is wanted with a linear motor, the POS command is used. The value has to be calculated in increments corresponding to 12 cm with the above table.
With a 40µm period of analog encoder and parameters K50 and K77 equal to 4, the value in increment corresponding to a 12cm movement is obtained as follows:

$$\text{Distance [upi]} = \text{Distance [m]} \cdot \frac{1024 \cdot 2^{K77}}{PCod \cdot 2^{K50}} = 0.12 \cdot \frac{1024 \cdot 2^4}{40 \cdot 10^{-6} \cdot 2^4} = 3072000$$

The value which has to be programmed is 3072000 increments for this movement.

20.1.1.2 Drive increments, linear motors

Abbreviation used in the table below: PCod = Encoder period [m].
k = Fast time interrupt [s] (refer to §4.2 for more information).

Unit	Quantity	Conversion formulae	Concerned quantities
[dpi]	**Distance** drive position increment	**[m] → [dpi]** $\text{Distance [dpi]} = \text{Distance [m]} \cdot \frac{1024 \cdot 2^{K77}}{PCod}$ (for analog encoder) $\text{Distance [dpi]} = \text{Distance [m]} \cdot \frac{64 \cdot 2^{K69}}{PCod}$ (for TTL encoder) **[dpi] → [m]** $\text{Distance [m]} = \text{Distance [dpi]} \cdot \frac{PCod}{1024 \cdot 2^{K77}}$ (for analog encoder) $\text{Distance [m]} = \text{Distance [dpi]} \cdot \frac{PCod}{64 \cdot 2^{K69}}$ (for TTL encoder)	• All internal distances • K30, K166, K167, K168, K189 • M0 to M5, M8
[dsi]	**Speed** drive speed increment	**[m/s] → [dsi]** $\text{Speed [dsi]} = \text{Speed [m/s]} \cdot \frac{256 \cdot 1024 \cdot 2^{K77} \cdot k}{PCod}$ (for analog encoder) $\text{Speed [dsi]} = \text{Speed [m/s]} \cdot \frac{256 \cdot 64 \cdot 2^{K69} \cdot k}{PCod}$ (for TTL encoder) **[dsi] → [m/s]** $\text{Speed [m/s]} = \text{Speed [dsi]} \cdot \frac{PCod}{256 \cdot 1024 \cdot 2^{K77} \cdot k}$ (for analog encoder) $\text{Speed [m/s]} = \text{Speed [dsi]} \cdot \frac{PCod}{256 \cdot 64 \cdot 2^{K69} \cdot k}$ (for TTL encoder)	• All internal speeds of the controller • K31 • M10, M11
[dai]	**Acceleration** drive acceleration increment	**[m/s²] → [dai]** $\text{Acceleration [dai]} = \text{Acceleration [m/s}^2\text{]} \cdot \frac{65536 \cdot 1024 \cdot 2^{K77} \cdot k^2}{PCod}$ (for analog encoder) $\text{Acceleration [dai]} = \text{Acceleration [m/s}^2\text{]} \cdot \frac{65536 \cdot 64 \cdot 2^{K69} \cdot k^2}{PCod}$ (for TTL encoder) **[dai] → [m/s²]** $\text{Acceleration [m/}s^2\text{]} = \text{Acceleration [dai]} \cdot \frac{PCod}{65536 \cdot 1024 \cdot 2^{K77} \cdot k^2}$ (for analog encoder) $\text{Acceleration [m/}s^2\text{]} = \text{Acceleration [dai]} \cdot \frac{PCod}{65536 \cdot 64 \cdot 2^{K69} \cdot k^2}$ (for TTL encoder)	• All internal acc. of controller • M14

Remark: 1024 * 2^{K77} and 64 * 2^{K69} can be replaced by the value given by monitoring M241.

20.1.2 Rotary motors

20.1.2.1 User increments, rotary motors

Abbreviation used in the table below: NPCod = Encoder periods number per turn [p/r]
h = Slow time interrupt[s] (refer to §4.1 for more information).

Unit	Quantity	Conversion formulae	Concerned quantities
[rupi]	**Distance** rotary user position increment	**[turn] → [rupi]** $\text{Distance [rupi]} = \text{Distance [turn]} \cdot \frac{1024 \cdot 2^{K77} \cdot NPCod}{2^{K50}}$ (for analog encoder) $\text{Distance [rupi]} = \text{Distance [turn]} \cdot \frac{64 \cdot 2^{K69} \cdot NPCod}{2^{K50}}$ (for TTL encoder) **[rupi] → [turn]** $\text{Distance [turn]} = \text{Distance [rupi]} \cdot \frac{2^{K50}}{1024 \cdot 2^{K77} \cdot NPCod}$ (for analog encoder) $\text{Distance [turn]} = \text{Distance [rupi]} \cdot \frac{2^{K50}}{64 \cdot 2^{K69} \cdot NPCod}$ (for TTL encoder)	• POS (K210), SET, STE, WTP • K27, K34, K35, K39, K45, K46, K47, K48, K208 • M6, M7, M12, M13, M36, M37
[rusi]	**Speed** rotary user speed increment	**[turn/s] →[rusi]** $\text{Speed [rusi]} = \text{Speed [turn/s]} \cdot \frac{256 \cdot 1024 \cdot 2^{K77} \cdot NPCod \cdot h}{2^{K50}}$ (for analog encoder) $\text{Speed [rusi]} = \text{Speed [turn/s]} \cdot \frac{256 \cdot 64 \cdot 2^{K69} \cdot NPCod \cdot h}{2^{K50}}$ (for TTL encoder) **[rusi] → [turn/s]** $\text{Speed [turn/s]} = \text{Speed [rusi]} \cdot \frac{2^{K50}}{256 \cdot 1024 \cdot 2^{K77} \cdot NPCod \cdot h}$ (for analog encoder) $\text{Speed [turn/s]} = \text{Speed [rusi]} \cdot \frac{2^{K50}}{256 \cdot 64 \cdot 2^{K69} \cdot NPCod \cdot h}$ (for TTL encoder)	• SPD (K211) • K41
[ruai]	**Acceleration** rotary user acceleration increment	**[turn/s^2] → [ruai]** $\text{Acceleration[ruai]} = \text{Acceleration } [turn/s^2] \cdot \frac{65536 \cdot 1024 \cdot 2^{K77} \cdot NPCod \cdot h^2}{2^{K50}}$ (for analog encoder) $\text{Acceleration[ruai]} = \text{Acceleration } [turn/s^2] \cdot \frac{65536 \cdot 64 \cdot 2^{K69} \cdot NPCod \cdot h^2}{2^{K50}}$ (for TTL encoder) **[ruai] → [turn/s^2]** $\text{Acceleration } [turn/s^2] = \text{Acceleration [ruai]} \cdot \frac{2^{K50}}{65536 \cdot 1024 \cdot 2^{K77} \cdot NPCod \cdot h^2}$ (for analog encoder) $\text{Acceleration } [turn/s^2] = \text{Acceleration [ruai]} \cdot \frac{2^{K50}}{65536 \cdot 64 \cdot 2^{K69} \cdot NPCod \cdot h^2}$ (for TTL encoder)	• ACC (K212) • K42, K206

Remark: 1024 * 2^{K77} and 64 * 2^{K69} can be replaced by the value given by monitoring M241.

20.1.2.2 *Drive increments, rotary motors*

Abbreviation used in the table below: NPCod = Encoder periods number per turn [p/r]
k = Fast time interrupt [s] (refer to §4.2 for more information).
h = Slow time interrupt [s] (refer to §4.1 for more information).

Unit	Quantity	Conversion formulae	Concerned quantities
[rdpi]	**Distance** rotary drive position increment	**[turn] → [rdpi]** $\text{Distance [rdpi]} = \text{Distance [turn]} \cdot 1024 \cdot 2^{K77} \cdot NPCod$ (for analog encoder) $\text{Distance [rdpi]} = \text{Distance [turn]} \cdot 64 \cdot 2^{K69} \cdot NPCod$ (for TTL encoder) **[rdpi] → [turn]** $\text{Distance [turn]} = \text{Distance [rdpi]} \cdot \frac{1}{1024 \cdot 2^{K77} \cdot NPCod}$ (for analog encoder) $\text{Distance [turn]} = \text{Distance [rdpi]} \cdot \frac{1}{64 \cdot 2^{K69} \cdot NPCod}$ (for TTL encoder)	• All internal distances of controller • K30, K166, K168, K167, K189 • M0 to M5, M8
[rdsi]	**Speed** rotary drive speed increment	**[turn/s] → [rdsi]** $\text{Speed [rdsi]} = \text{Speed [turn/s]} \cdot 256 \cdot 1024 \cdot 2^{K77} \cdot NPCod \cdot k$ (for analog encoder) $\text{Speed [rdsi]} = \text{Speed [turn/s]} \cdot 256 \cdot 64 \cdot 2^{K69} \cdot NPCod \cdot k$ (for TTL encoder) **[rdsi] → [turn/s]** $\text{Speed [turn/s]} = \text{Speed [rdsi]} \cdot \frac{1}{256 \cdot 1024 \cdot 2^{K77} \cdot NPCod \cdot k}$ (for analog encoder) $\text{Speed [turn/s]} = \text{Speed [rdsi]} \cdot \frac{1}{256 \cdot 64 \cdot 2^{K69} \cdot NPCod \cdot k}$ (for TTL encoder)	• All internal speeds of controller • K31 • M10, M11
[rdai]	**Acceleration** rotary drive acceleration increment	**[turn/s²] → [rdai]** $\text{Acceleration [rdai]} = \text{Acc } [turn/s^2] \cdot 65536 \cdot 1024 \cdot 2^{K77} \cdot NPCod \cdot k^2$ (for analog encoder) $\text{Acceleration [rdai]} = \text{Acc } [turn/s^2] \cdot 65536 \cdot 64 \cdot 2^{K69} \cdot NPCod \cdot k^2$ (for TTL encoder) **[rdai] → [turn/s²]** $\text{Acceleration } [turn/s^2] = \text{Acc [rdai]} \cdot \frac{1}{65536 \cdot 1024 \cdot 2^{K77} \cdot NPCod \cdot k^2}$ (for analog encoder) $\text{Acceleration } [turn/s^2] = \text{Acc [rdai]} \cdot \frac{1}{65536 \cdot 64 \cdot 2^{K69} \cdot NPCod \cdot k^2}$ (for TTL encoder	• All internal acc. of controller • M14

Remark: $1024 * 2^{K77}$ and $64 * 2^{K69}$ can be replaced by the value given by monitoring M241.

20.1.3 Resolution

The **resolution** is the value of a single increment expressed in the ISO unit system A distinction between the user resolution and the controller resolution as in the unit system is also necessary.

Caution: The resolution is high when the value of the resolution is small and vice-versa.

For example (with an analog encoder), a user resolution value for speed is:

$$\text{Speed [m/s]} = \underbrace{\text{Speed [usi]}}_{=1} \cdot \frac{PCod \cdot 2^{K50}}{256 \cdot 1024 \cdot 2^{K77} \cdot h} \Rightarrow \text{User resolution [m/s]} = \frac{PCod \cdot 2^{K50}}{256 \cdot 1024 \cdot 2^{K77} \cdot h}$$

And for a controller resolution:

$$\text{Speed [m/s]} = \underbrace{\text{Speed [dsi]}}_{=1} \cdot \frac{PCod}{256 \cdot 1024 \cdot 2^{K77} \cdot k}$$

$$\Rightarrow \text{Controller resolution [m/s]} = \frac{PCod}{256 \cdot 1024 \cdot 2^{K77} \cdot k}$$

There is a better resolution with the drive than the user. In the above example, the motor can reach a maximum speed (SPD command) with a certain amount of user resolution, but the controller will calculate the speed trajectory with a higher resolution.

20.2 Current, force and torque units

The current, force and torque are in increment too. The linear motors are expressed in force, and rotary motors in torque. Both quantities are measured from the motor current.

The **force** F from a **linear motor** is expressed in Newton and is proportional to the current according to the following formula:

$$F\ [N] = K_t \cdot I$$

with F = force [N]
K_t = motor force constant [N/A]
I = current [A]

The **torque** T from a **rotary motor** is expressed in Newton meter and is proportional to the current according to the following formula:

$$T\ [N \cdot m] = K_t \cdot I$$

with T = torque [N · m]
K_t = motor force constant [N · m/A]
I = current [A]

The constant K_t is given for every motor (linear or rotary). The formulae which calculates the currents and forces in ISO units from their increment values and vice-versa are given bellow.

For the next part of this chapter, the units will be abbreviated as follows. In all other cases, a formula will be given.

	Notation	Signification	Equivalent in ISO unit
Current	**[ci]**	Current increment	**[A]**
Force	**[foi]**	Force increment (linear motors)	**[N]**
Torque	**[toi]**	Torque increment (rotary motors)	**[N · m]**

Abbreviation used in the table below: $I_{max,\ controller}$ = [A] maximum current delivered by the drive = M82/100
K_t = [N/A] Force constant for a linear motor.
K_t = [N · m/A] Torque constant for a rotary motor.

Unit	Quantity	Conversion formulae	Concerned quantities
[ci]	**Current** increment	**[A] → [ci]** $\text{Current [ci]} = \text{Current [A]} \cdot \frac{32768}{I_{max,\ controller}}$ **[ci] → [A]** $\text{Current [A]} = \text{Current [ci]} \cdot \frac{I_{max,\ controller}}{32768}$	• K83, K91, K92, K44 • M20 to M24
[foi]	**Force** increment (linear motors)	**[N] → [foi]** $\text{Force [foi]} = \text{Force [N]} \cdot \frac{32768}{K_t \cdot I_{max,\ controller}}$ **[foi] → [N]** $\text{Force [N]} = \text{Force [foi]} \cdot \frac{K_t \cdot I_{max,\ controller}}{32768}$	• K60 • M30 to M32

Unit	Quantity	Conversion formulae	Concerned quantities
[toi]	**Torque** increment (rotary motors)	**[N · m] → [foi]** $\text{Torque [toi]} = \text{Torque [N} \cdot \text{m]} \cdot \frac{32768}{K_t \cdot I_{\text{max, controller}}}$ **[foi] → [N · m]** $\text{Torque [N} \cdot \text{m]} = \text{Torque [toi]} \cdot \frac{K_t \cdot I_{\text{max, controller}}}{32768}$	• K60 • M30 to M32

20.3 Time quantities units

Some parameters and commands are time-related. The formulae which calculates the time in ISO unit from the time in increments are given below.

In the next part of the chapter, the time units will be abbreviated as follows and the conversion formulae given from case to case.

Notation	Signification	Equivalent in SI Unit
[sti]	Slow time increment	**[s]**
[fti]	Fast time increment	**[s]**
[cti]	Current loop time increment	**[s]**

Abbreviation used in the table below: h = Slow time interrupt [s] (refer to §4.1 for more information).
k = Fast time interrupt [s] (refer to §4.2 for more information).
m = Current time interrupt[s] (refer to §4.3 for more information).

Unit	Quantity	Conversion formulae	Concerned quantities
[sti]	**Time** Slow time increment	**[sti] → [s]** $\text{Time [sti]} = \frac{\text{Time [s]}}{h}$ **[s] → [sti]** $\text{Time [s]} = \text{Time [sti]} \cdot \text{h}$	• K38, K164, K229 • LTI (K204) • WTT • JRT (K213)
[fti]	**Time** Fast time increment	**[fti] → [s]** $\text{Time [fti]} = \frac{\text{Time [s]}}{k}$ **[s] → [fti]** $\text{Time [s]} = \text{Time [fti]} \cdot k$	-
[cti]	**Time** Current loop time increment	**[cti] → [s]** $\text{Time [cti]} = \frac{\text{Time [s]}}{m}$ **[s] → [cti]** $\text{Time [s]} = \text{Time [cti]} \cdot m$	-

$$\Rightarrow \text{Controller resolution [m/s]} = \frac{PCod}{256 \cdot 1024 \cdot 2^{K77} \cdot k} \quad \text{(for analog encoder)}$$

$$\Rightarrow \text{Controller resolution [m/s]} = \frac{PCod}{256 \cdot 64 \cdot 2^{K69} \cdot k} \quad \text{(for TTL encoder)}$$