

The Instruction Set of the TANGO Controller



In der Murch 15
35579 Wetzlar
Germany
Tel.: +49/6441/9116-0
www.marzhauser.com

1. Table of Contents

1.	Table of Contents	2
2.	Introduction.....	6
3.	Hint for controller initialization.....	8
4.	Brief Description of the Tango Instruction Set	9
5.	Instruction Syntax Description.....	14
6.	Error Numbers and their possible Root Cause	14
7.	Controller Informations.....	15
7.1.	version (Read detailed Version information)	15
7.2.	det (Read detailed Configuration)	16
7.3.	readsn (Read Serial Number)	16
7.4.	ver (Read default Version Number)	17
7.5.	iver (Read internal Version Number)	17
7.6.	uptime (Read Controller Up Time)	17
7.7.	temp (Read Case Temperature)	17
8.	Communication Interface Settings.....	18
8.1.	baud (Baud Rate)	18
8.2.	cts (Enable/Disable RS232 Hardware Handshake)	18
9.	System Instructions	19
9.1.	save (Save Parameters)	19
9.2.	restore (Restore Saved Parameters)	19
9.3.	reset (Force a Software Reset).....	19
9.4.	pa (Enable or Disable the Power Amplifiers)	20
9.5.	ipreter (Select Instruction Set)	20
10.	Operating Modes	21
10.1.	Extended Mode.....	21
10.1.1	extmode (Switch to Extended Mode)	21
10.2.	Scan Mode	22
10.2.1	scanmode (Switch to Scan Mode).....	22
10.2.2	scanvel (Scanmode Vector Velocity).....	22
11.	Controller States and Error Messages.....	23
11.1.	autostatus (Set Autostatus to required behavior)	23
11.2.	statusaxis (Query State of Axis).....	23
11.3.	status (Query the Controller Error State)	24
11.4.	err (Query Error Number)	24
11.5.	help (Query Error Number with Description String).....	24
11.6.	service (Print Service Information to Terminal)	25
11.7.	pci (Is PCI Bus).....	25
11.8.	lock (Select Parameters to Lock)	25
11.9.	isvel (Query Actual Velocities)	26
11.10.	maxpos (Maximum Position)	26
12.	General Adjustments.....	27
12.1.	dim (Unit for Positions)	27
12.2.	pitch (Spindle Pitch).....	27
12.3.	gear (Gear Ratio).....	28
12.4.	motorsteps (Motor Steps Per Revolution)	28
12.5.	accel (Acceleration)	29
12.6.	accelfunc (Acceleration Ramp Function)	29



12.7.	stopaccel (Emergency Stop Deceleration).....	30
12.8.	vel (Velocity)	30
12.9.	velfac (Velocity Factor)	31
12.10.	sevel (Secure Velocity)	31
12.11.	maxcur (Query Maximum Motor Current)	31
12.12.	cur (Motor Current)	32
12.13.	reduction (Motor Current Reduction Factor)	32
12.14.	curdelay (Delay for Current Reduction)	33
12.15.	axis (Enable, Disable, Switch Off Axis).....	33
12.16.	axisdir (Axis Direction)	34
12.17.	motortable (Motor Correction Table).....	34
12.18.	usteps (Microstep Resolution)	35
12.19.	resolution (Position Number Format).....	35
12.20.	backlash (Mechanical Backlash Compensation)	35
12.21.	lockaxis (Apply the Parameter Lock to Axes)	36
12.22.	lockstate (Query all internal Lock States)	36
12.23.	stout (Select Status Signal Output).....	36
13.	Limit Switch Instructions (Hardware and Software)	37
13.1.	lim (Software Limits)	37
13.2.	limctr (Enable or Disable Limit Control)	37
13.3.	nosetlimit (Do not set limits by cal/rm)	38
13.4.	swtyp (Type of Limit Switch)	38
13.5.	swpol (Polarity of Limit Switch)	39
13.6.	swact (enable or disable limit switches)	39
13.7.	swdir (swap assignment of cal and rm switch).....	40
13.8.	readsw (Read Status of Limit Switches)	40
13.9.	swin (Read Limit Switch Input Level)	41
13.10.	statuslimit (Limit Status)	42
14.	Calibration and Range Measure Instructions	43
14.1.	cal (Command a Calibration)	43
14.2.	rm (Command a Range Measure)	43
14.3.	calmode (Closed Loop/Calibration Behavior)	44
14.4.	caltimeout (Calibration Timeout)	44
14.5.	caliboffset (Calibration Offset).....	45
14.6.	rmoffset (Range Measure Position Offset).....	45
14.7.	caldir (Calibration Direction)	45
14.8.	calbspeed (Calibration Speed for Retraction)	46
14.9.	calrefspeed (Reference Signal Calibration Speed)	46
14.10.	calpos (Calibration Position)	46
14.11.	refdir (Direction for Searching Reference Signal)	47
14.12.	calvel (Calibration Velocities for CAL Instruction)	47
14.13.	rmvel (Range Measure Velocities for RM Instruction)	47
14.14.	autopitch (Measure Pitch after CAL Instruction)	48
15.	Move Instructions	49
15.1.	moa (Move Absolute).....	49
15.2.	mor (Move Relative)	49
15.3.	m (Move Relative Shortcut)	50
15.4.	distance (Distance for m).....	50
15.5.	moc (Move to Center)	50
15.6.	go (Go To Position).....	51
15.7.	speed (Speed Move)	51



15.8.	a (Abort the Current Move)	52
15.9.	delay (Set the Delay Time for Consecutive Moves)	52
15.10.	pause (Set the Pause after Position Reached).....	52
15.11.	pos (Read or Set Position)	53
15.12.	zero (Set Internal Position to Zero).....	54
15.13.	clearpos (Set Internal Position to Zero)	54
16.	HDI: Joystick, Tackball and Handwheel Instructions	55
16.1.	joy (Generally Enable/Disable/Set Joystick Mode)	55
16.2.	joydir (Joystick Direction or Assign Joystick)	56
16.3.	joywindow (Joystick Window)	56
16.4.	joyvel (Joystick Velocity)	57
16.5.	joyspeed (Joystick Speed Presets for BPZ Device)	57
16.6.	keymode (Joystick Key Mode)	58
16.7.	keyspeed (Joystick Key Speed Presets).....	59
16.8.	joycurve (Joystick Characteristic)	59
16.9.	key (Read HDI Device Key State).....	59
16.10.	keyl (Read HDI Device Latched Key State)	60
16.11.	hwfactor (Handwheel Transmission Factor)	60
16.12.	hwfactorb (Alternate Handwheel Factor)	60
16.13.	hwfilter (Handwheel Noise Filter).....	61
16.14.	tbfactor (Handwheel Transmission Factor).....	61
16.15.	zwheel (Is Z-Wheel Available)	61
16.16.	zwtravel (Z-Wheel Travel per wheel revolution)	62
16.17.	tvrjoy (Pulse and Direction Joystick Functionality).....	63
16.18.	tvrjoyf (Pulse and Direction Joystick Factor).....	63
16.19.	hdi (Read HDI ID)	63
16.20.	hdimode (HDI Mode Options)	64
17.	Digital and Analogue I/O	65
17.1.	digin (Digital Input)	65
17.2.	digout (Digital Output).....	65
17.3.	adigin (AUX-I/O Digital Input)	66
17.4.	adigout (AUX-I/O Digital Output).....	66
17.5.	anain (Analogue Input)	67
17.6.	anaout (Analogue Output)	68
17.7.	stoppol (Mode and Polarity of Stop Input Signal).....	68
17.8.	stop (Release Stop Condition)	68
17.9.	shutter (Shutter Out Signal of AUX-IO).....	69
18.	Encoder Instructions	70
18.1.	encmask (Encoder Mask)	70
18.2.	enc (Encoder Active)	70
18.3.	encperiod (Encoder Signal Period)	71
18.4.	encdir (Encoder Counting Direction)	71
18.5.	encvel (Encoder Auto-Ajust Velocity).....	71
18.6.	encttl (Encoder has TTL Signal)	72
18.7.	encref (Use Encoder Reference Signal)	72
18.8.	encnas (Use Encoder NAS Error Signal).....	73
18.9.	encrefstatus (Encoder REF Signal State)	73
18.10.	encrefstatusl (Latched Encoder REF Signal State)	73
18.11.	encnasstatus (Encoder NAS Error Signal State)	74
18.12.	encerr (Encoder Error State)	74
18.13.	encamp (Encoder Signal Amplitude)	74



18.14.	encpos (Encoder Position)	75
18.15.	hwcount (Hardware Counter)	75
18.16.	clearhwcount (Clear Hardware Counter)	75
19.	MR Encoder Instructions	76
19.1.	mra (MR Amplitude Correction Factor)	76
19.2.	mro (MR Offset Correction Value)	76
19.3.	mrp (MR Signal Peak-To-Peak Measuring Result)	77
19.4.	mrt (MR Signal Level)	77
20.	Closed Loop Instructions	78
20.1.	ctr (Control Enable).....	78
20.2.	ctrf (Control Factor).....	79
20.3.	ctrff (Extended Control Factor)	79
20.4.	ctrc (Control Call).....	80
20.5.	ctrd (Control Target Window Delay).....	80
20.6.	ctrt (Control Timeout).....	80
20.7.	twi (Target Window).....	81
20.8.	ctrstatus (Control Status)	81
20.9.	ctrdiff (Control Position Difference)	82
21.	Trigger Signal Configuration	83
21.1.	trig (Trigger)	83
21.2.	triga (Trigger Axis)	83
21.3.	trigm (Trigger Mode)	84
21.4.	trigger (Force Trigger Signal).....	85
21.5.	trigs (Trigger Signal Length)	85
21.6.	trigd (Trigger Distance)	85
21.7.	trigf (Trigger Frequency)	87
21.8.	trigcount (Trigger Counter)	87
22.	Snapshot Signal Configuration	88
22.1.	sns (Snapshot).....	88
22.2.	sns1 (Snapshot Level / Polarity)	88
22.3.	snsf (Snapshot Filter).....	89
22.4.	sns1 (Snapshot Mode).....	89
22.5.	sns1 (Snapshot Counter)	89
22.6.	sns1 (Snapshot Position)	91
22.7.	sns1 (Snapshot Array)	91
22.8.	sns1 (Snapshot Event)	92
22.9.	prehome (Snapshot PreHome Position)	93
22.10.	home (Snapshot Home Position).....	93
23.	Document Revision History	94

2. Introduction

Instruction syntax:

All Tango controllers communicate via a standard serial COM port interface, independent of the controller type (RS232C, USB, PCI, PCI-E). The instructions and parameters are sent as cleartext ASCII strings with a terminating carriage return [CR], which is 0x0d hex. Characters may be upper-, lower- or camel-case. The parameters are separated by a space character.

This provides easy access to all functions by using a simple terminal program such as HyperTerminal. A typical instruction syntax is as follows:

[!,?][instruction][SP][optional axis] [parameter1][SP][parameter2] [etc...] [CR]

[!,?] Read/write specifier, required by most instructions:

! (exclamation mark) = to write parameter, execute an instruction etc.

? (question mark) = to read data (returns settings, or status, etc.)

[instruction] Is the instruction word itself.

[SP] Space (ASCII 0x20 hex) as separation.

[optional axis] Axis character x, y, z or a if only one axis must be addressed.

[parameter] Usually integer or floating point numbers, floating point uses decimal point, no comma.

[CR] Termination (ASCII 0x0d hex), causes instruction execution.

Syntax examples:

```
!vel 10 1.5      set velocities for the first two axes
!cal            command all axes to perform a calibration move
!moa y 10.1     move y axis to absolute position 10.1
?pos           returns all axis positions
```

Moves:

Move commands are executed as a vector move, so if several axes are involved they will reach their destinations at the same time. This means that - depending on velocity, acceleration and travel distance - one leading axis travels at its full velocity while the others follow synchronously. To move axes independently with their individual velocities, they may be started separately by using single axis instructions. Please refer to the "move" instruction descriptions.

Settings:

Most settings can be stored permanently in the Tango Controller, so they are available from power on. When stored once, this reduces initialization overhead of the application software. Refer to the "save" instruction for further information. Parameters that are saved can be identified by a 'Y' in the Save column of the brief instruction set description later in this document.

Character limits:

To prevent the input buffer from overflow, please do not send more than 255 characters at once.

Such may occur when sending the setup sequence to the Tango controller. A good practice is to request the "?err" state after each setup instruction. This will return the information if the parameters were accepted or not while preventing overflow.

Another solution is to activate the "!cts" handshake (available only with Desktop RS232C and some USB versions). This will automatically halt the PC transmission for as long as the input buffer is full. The PC COM port then must be opened with hardware handshake on, too. Please refer to the "!cts" command description.

Important: Security speed limitation!

The Tango controllers have a built in security function, which reduces the maximum travel velocity to a secure 10mm/s for as long as no initial "cal" and "rm" moves have been executed. This is to preserve the microscope stage from damage that could be caused by moving fast into its end positions. After calibrating the axis into its endswitches (cal and/or rm if switches are mounted and enabled) the travel velocity is no longer limited.

If it is not wanted or impossible to do a calibration and range measure move on each power on, the speed limit may be increased to up to 100mm/s at own risk. Please refer to the "**secvel**" instruction for further information.

Important: Measuring units!

The measuring unit is set by the "dim" instruction, where dim 2 [mm] is the default value.

In all dim settings, except of dim 9 the velocities are in motor turns per second! Only dim 9 provides the millimeter unit for most parameters¹, positions and velocities.

Extended mode:

In addition to the improvements when using dim 9 units, there is an option to enable extended mode behavior. It enables more functionality, like separate calibration, rm and joystick velocities, which else are the same as the axis velocity (vel). Please refer to the "extmode" instruction for further details.

¹ Only '**calbspeed**' and '**calrefspeed**' are always in 1/100 turns/sec, even in dim mode 9

3. Hint for controller initialization

Please make sure that first of all the following parameters have to be set:

- The axis units (here called "**dim**")
- If the controller firmware version is 1.32 or above: the **extmode**
- The axis **pitch** and - if used - also the **gear**, which are always in [mm] independend of dim

Using *dim=9* and *extmode=1* will turn all (even the vel and joyvel) units to stage related [mm] and [mm/s]. Extmode=1 also offers bugfixes, more features and flexibility. But it has a slightly different behavior. Please refer to the **Extended Mode** description in this document.

4. Brief Description of the Tango Instruction Set

Controller Informations					
Instruction		Example	Save	Brief description	Page
(?)	version	version	fix	Read detailed firmware and controller version	15
(?)	det	det	fix	Read detailed configuration information	16
?	readsn	?readsn	fix	Read the controller serial number	16
(?)	ver	ver	fix	Read default version number	17
(?)	iver	iver	fix	Read further version number information	17
(?)	uptime	uptime	-	Read how long the controller is running	17
(?)	temp	temp	-	Read case temperature (available with encoder option)	17

Communication Interface Settings					
Instruction		Example	Save	Brief description	Page
? !	baud	!baud 9600	Y	Set RS232 baud rate to 9600 Bd (default=57600)	18
? !	cts	!cts 1	Y	Switch on CTS hardware handshake	18

System Instructions					
Instruction		Example	Save	Brief description	Page
(!)	save	save	-	Save parameters to controller nonvolatile memory	19
(!)	restore	restore	-	Reload controller parameters from saved values	19
(!)	reset	reset	-	Reset controller (forces restart, similar to cycle power)	19
!	pa	!pa 1	-	Enable power amplifiers (disable = 0), see 'axis' cmd. too	20
? !	ipreter	!ipreter 2	Y	Select optional Venus instruction set	20

Operating Modes					
Instruction		Example	Save	Brief description	Page
? !	extmode	!extmode 1	-	Enable extended controller behavior	21
? !	scanmode	!scanmode 1	-	Set controller positioning behavior to scanmode	22
? !	scanvel	!scanvel 20 20	-	Set scanmode vector velocity to 20mm/s for X and Y	22

Controller States and Error Messages					
Instruction		Example	Save	Brief description	Page
? !	autostatus	!autostatus 0	-	Select autostatus response type 0 (=disabled), range: [0-4]	23
(?)	statusaxis	statusaxis	-	Read axis state [@,M,J,C,S,A,D,-]	23
(?)	status	status	-	Read controller error state	24
(?)	err	err	-	Read error number	24
(?)	help	help	-	Read error number with additional text description	24
(?)	service	service	-	Returns a detailed parameter and state list, for debugging	25
(?)	pci	pci	-	Returns 1 if controller is plugged in a PCI slot (desktop=0)	25
!?	lock	!lock 2 1	Y	Set write protection for parameter 2 (here: motor current)	25
(?)	isvel	?isvel x	-	Query actual velocity of the X axis	26
?	maxpos	?maxpos x	-	Query maximal available position range for X axis	26

General Adjustments					
Instruction		Example	Save	Brief description	Page
? !	dim	!dim 1 1 1	Y	Set position units of X Y Z to μm	27
? !	pitch	!pitch 1 1 1	Y	Set spindle pitch of X Y Z to 1 [mm/revolution]	27
? !	gear	!gear 1 1 1	Y	Set gear factor of X Y Z to 1	28
? !	motorsteps	!motorsteps x 200	Y	Set X axis motor has 200s steps per revolution	28
? !	accel	!accel 0.1 0.1 0.1	Y	Set acceleration of X Y Z to 0.1m/s ²	29

General Adjustments

Instruction		Example	Save	Brief description	Page
? !	accelfunc	!accelfunc 1 1 0	Y	Set acceleration function X and Y to sin², Z to linear	29
? !	stopaccel	!stopaccel 2 2	Y	Set X and Y deceleration during stop condition to 2m/s²	30
? !	vel	!vel 10 10 10	Y	Adjust speed of X Y Z to 10 [revolutions/s]	30
? !	velfac	!velfac 1 1 1	Y	Set velocity reduction factor for X Y Z to 1 (= no reduction), range is [0.01-1]	31
? !	secvel	!secvel x 20	Y	Set secure speed limit X to 20mm/s (unit is always mm/s)	31
?	maxcur	?maxcur	fix	Show the maximum possible motor currents of all axes	31
? !	cur	!cur 0.5 0.6 1	Y	Set motor current in Ampere: X=0.5 Y=0.6 and Z=1 A	32
? !	reduction	!reduction 0.5 0.5 0.5	Y	Select 50% motor current reduction for X Y Z	32
? !	curdelay	!curdelay 1000	Y	Delay X axis motor current reduction by 1000 [ms]	33
? !	axis	!axis 1 0 -1	Y	Enable X, disable Y and switch off Z axis	33
? !	axisdir	!axisdir 0 1 0	Y	Reverse rotating direction of Y motor (caution!)	34
? !	motortable	!motortable x 2	Y	Select custom motor correction table type 2 for X axis	34
? !	usteps	!usteps 50000	Y	Set microstepping resolution to 50000/rev for all axes	35
? !	resolution	!resolution 6	Y	Set pos query return string resolution to 6 decimal places	35
? !	backlash	!backlash 12.3 0 0	Y	Set backlash compensation to 12.3µm in X and 0 in Y & Z	35
!?	lockaxis	!lockaxis 0 0 0 0	Y	Remove lock protection from all axes (lock has no effect)	36
?	lockstate	?lockstate x	-	Query extended locked parameters, including internal limitations currently applied to X axis	36
? !	stout	!stout 2	Y	Make Status LED state available at AUX-I/O Pin VR_OUT	36

Limit Switch Instructions (Hardware and Software)

Instruction		Example	Save	Brief description	Page
? !	lim	!lim 0 10 0 10 0 10	-	Set lower position limit to 0 and upper limit to 10 (assume unit is [mm] if dim was set to 2) for X Y Z	37
? !	limctr	!limctr x 1	-	Enable hardware limit switches for X axis, default = 1	37
? !	nosetlimit	!nosetlimit 1 1 1 1	Y	Disable setting/overwriting of software limits during cal and rm for all axes (here: X Y Z A), default = 0	38
? !	swtyp	!swtyp 1 0 1 !swtyp y 0 0 0	Y	Set limit switch type for all axes to NPN (pull-up) Set limit switch type for Y to PNP (pull-down)	38
? !	swpol	!swpol 1 0 1 !swpol z 1 0 1	Y	Set polarity of limit switches for all axes to active high (=1) Set polarity of limit switches for Z to active high	39
? !	swact	!swact 1 0 1 !swact y 1 0 0	Y	Enable cal and rm limit switches for all axes Enable cal limit switch for Y, disable ref and rm	39
? !	swdir	!swdir x 1	Y	Swap reference- and endswitch assignment for X axis	40
?	readsw	?readsw	-	Read states of all limit switches (1=active and actuated)	40
(?)	swin	swin	-	Read TTL signal level of all limit switch inputs (1=high)	41
(?)	statuslimit	statuslimit	-	Read current limit status „A“ = calibration done „D“ = rm done „L“ = limit switch modified by software „-“ = not yet modified	42

Calibration and Range Measure Instructions

Instruction		Example	Save	Brief description	Page
(!)	cal	cal	-	Perform a calibration move for all enabled axes, see 'axis'	43
(!)	rm	rm x	-	Perform a range measure move in X	43
? !	calmode	!calmode 2 2	Y	Set calibration/closed loop behavior X, Y to type 2	44
? !	caltimout	!caltimout 60 60 10	Y	Set calibration timeout for X and Y to 1 minute, Z to 10s	44
? !	caliboffset	!caliboffset 1 1 1	Y	Set the cal zero-point 1mm aside lower limit switch (dim 2)	45
? !	rmoffset	!rmoffset 1 1 1	Y	Set rm end-position 1mm aside upper limit switch (dim 2)	45
? !	caldir	!caldir z 1	Y	Calibrate the Z-axis in positive direction	45

**Calibration and Range Measure Instructions**

Instruction	Example	Save	Brief description	Page
? ! calbspeed	!calbspeed 20	Y	Set the speed for move out of 'cal' and 'rm' limit switches for all axes to 0.2 [revolutions/s], range is [1...100]	46
? ! calrefspeed	!calrefspeed 10	Y	Set the speed for calibrating to the encoder reference for all axes to 0.1 [revolutions/s], range is [1...100]	46
? ! calpos	calpos	-	Read back the encoder position where the calibration switch was released	46
? ! refdir	?refdir y	Y	Read the direction for encoder reference search in Y axis	47
? ! calvel	!calvel x 10 0.5	Y	Only if extmode = 1: Set calibration velocities in X	47
? ! rmvel	!rmvel x 10 0.5	Y	Only if extmode = 1: Set range measure velocities in X	47
? ! autopitch	!autopitch x 1	Y	Measure pitch after cal move of X axis	48

Move Instructions

Instruction	Example	Save	Brief description	Page
(!) moa	moa 10 10 10 moa y 20	-	Move X Y Z absolute to positions 10 10 10 Move Y axis to position 20 (unit depends on dim seting)	49
(!) mor	mor 4 4 4 mor y -10.5	-	Move X Y Z relative by 4 (unit depends on dim seting) Move Y axis relative 10.5 backwards	49
(!) m	m	-	Move relative again (use same parameters as defined by last '!mor' or '!distance' instruction)	50
? ! distance	!distance 1 1 1	-	Set distance for X Y Z 'm'-move (start with 'm' or '!m')	50
(!) moc	moc x	-	Move X to center position between lower and upper limit switch, or between lower and upper software limits	50
(!) go	go x 12.5	-	Move X to pos. 12.5, overwritable, for tracking applications	51
? ! speed	!speed 5 5 5 !speed y 0	-	Digital joystick: move X Y Z axis with 5 [revolutions/s] Stop the Y axis speed move	51
(!) a	a	-	Abort move (Stop)	52
? ! delay	!delay 1000	Y	Delay all consecutive moves by 1000 ms	52
? ! pause	!pause 10	Y	Delay "position reached" autostatus response by 10 ms	52
? ! pos	!pos 0 0 0 !pos z 1.2	-	Set current X Y Z position to 0 Set current Z position to 1.2	53
(!) zero	!zero z	-	Set Z position and internal counter to 0 (e.g. filter wheel application)	54
(!) clearpos	!clearpos z	-	Set Z position and internal counter to 0 (e.g. filter wheel application), not executable with measuring system	54

HDI: Joystick, Tackball and Handwheel Instructions

Instruction	Example	Save	Brief description	Page
? ! joy	!joy 0 !joy 2	Y	Switch joystick on(=2) or off(=0)	55
? ! joydir	!joydir 1 1 -1	Y	Set motor direction for joystick operation (Z reversed)	56
? ! joywindow	!joywindow 14	Y	Set idle window of the joystick center position, where a joystick deflection has no effect [0..100]	56
? ! joyvel	!joyvel z 1.5	Y	Only if extmode = 1: Set joystick velocity for Z to 1.5	57
? (!) joyspeed	joyspeed 2 25	Y	Set joystick speed for speed button 2 "medium" to 25 rev/s	57
? ! keymode	!keymode 2	Y	Select joystick key mode 2 = high speed preselection	58
? ! keyspeed	!keyspeed x 5 20	Y	Set keymode joystick speed X low=5mm/s, high=20mm/s	59
? (!) joycurve	!joycurve z 1	Y	Set joystick characteristic for Z ot linear	59
(?) key	key	-	Read state of all joystick buttons (0=released, 1=pressed)	59
(?) keyl	keyl	-	Read and clear latched state of all joystick buttons	60
? ! hwfactor	!hwfactor x 1	Y	One handwheel revolution in X is 1mm stage travel	60
? ! hwfactorb	!hwfactorb x 14	Y	One handwheel revolution in X is 14mm stage travel	60
? ! hwfilter	!hwfilter 0	Y	Switch off handwheel noise reduction	61
? ! tbfactor	!tbfactor 1 1	Y	Set trackball transmission factor in X and Y to default	61

**HDI: Joystick, Tackball and Handwheel Instructions**

Instruction	Example	Save	Brief description	Page
(?) zwheel	?zwheel	-	Returns 1 if HDI device has a Z-Wheel attached	61
? (!) zwtravel	!zwtravel 1 0.25	Y	Set default Z-Wheel travel to 2.5 mm/rev	62
? ! tvrjoy	!tvrjoy z	Y	Assign AUX-IO pulse&direction joystick to Z axis	63
? ! tvrjoyf	!tvrjoyf 1	Y	Set tvrjoy transmission factor to 1	63
(?) hdi	hdi	-	Read ID number of the connected HDI device	63
! ? hdimode	!hdimode 0 1	Y	Set hdimode bit 0 to 1 for ErgoDrive Toggle Mode	64

Digital and Analogue I/O

Instruction	Example	Save	Brief description	Page
(?) digin	digin digin 8	-	I/O Extension board: Read all digital inputs I/O Extension board: Read digital input 8	65
? ! digout	!digout 5 1 ?digout	-	I/O Extension board: Set digital output 5 to logic level 1 I/O Extension board: Read back all digital output levels	65
(?) adigin	adigin adigin 2	-	Read all AUX-I/O digital inputs Read logic level of AUX-I/O digital input 2 only	66
? ! adigout	!adigout 3 1 ?adigout	-	Set AUX-I/O digital output 3 to logic level 1 Read back all digital output levels	66
(?) anain	anain c 2	-	Read input of analogue channel 2	67
? ! anaout	!anaout c 1 17.5	-	Set analogue voltage of channel 1 to 17.5 percent (1.75V)	68
? ! stoppol	!stoppol 1	Y	Set AUX-IO stop input to active high	68
! stop	!stop 0	-	Release stop condition (in stoppol modes 4 or 5)	68
? ! shutter	!shutter 1	-	Set AUX-IO shutter out signal to TTL high	69

Encoder Instructions

Instruction	Example	Save	Brief description	Page
? ! encmask	!encmask 1 1 0	Y	Enable activation of X and Y encoders, disable Z	70
? ! enc	!enc 1 0	-	Manually activate X encoder (caution!), set Y to inactive	70
? ! encperiod	!encperiod 0.1	Y	Set signal period of X encoder to 100 µm	71
? ! encctl	!encctl x 1	Y	X encoder is TTL type (has no analogue sin/cos signal)	72
? ! encdir	!encdir y 1	(Y)	Reverse counting direction for Y encoder	71
? ! encvel	!encvel x 0.5	Y	Set auto-adjust velocity of X encoder to 0.5mm/s	71
? ! encref	!encref 0	Y	Disable usage of X encoder reference signal	72
? ! encnas	!encnas 1 0 0	Y	Enable NAS error signal input encoding for X encoder only	72
(?) encrefstatus	encrefstatus x	-	Read X encoder reference signal state (1=on reference)	73
(?) encrefstatusl	encrefstatusl x	-	Read latched X encoder reference signal state	73
(?) encnasstatus	encnasstatus x	-	Read X encoder NAS signal state (1=NAS error)	74
? ! encerr	!encerr 0	-	Clear encoder error state for X axis (? response is 0 or e)	74
? ! encamp	?encamp x	-	Read X encoder signal amplitude in percent	74
? ! encpos	!encpos 1	-	?pos instruction returns for the encoder positions, if enc=1	75
(?) hwcount	hwcount	-	Read all encoder positions (TTL counter, no interpolation)	75
(!) clearhwcount	clearhwcount x	-	Set X axis hwcount to zero	75

MR Encoder Instructions

Instruction	Example	Save	Brief description	Page
? ! mra	?mra x	-	Read amplitude correction factor (sin/cos ratio) of X	76
? ! mro	?mro	-	Read offset correction value for all encoders	76
? ! mrp	!mrp x 0 0 0 0	-	Reset MR-signal peak-to-peak measurement result of X	77
? mrt	?mrt z 2	-	List two measurement results of the Z input signals	77

Closed Loop Instructions

Instruction	Example	Save	Brief description	Page
? ! ctr	!ctr 1 1 1	Y	Set closed loop circuit X Y Z to "active until reached" mode	78
? ! ctrf	!ctrf 2.0	Y	Closed loop factor for X axis is set to 2.0	79
? ! ctrff	!ctrf 2 3.5	Y	Closed loop factors for X axis are set to 2 and 3.5	79
? ! ctrc	!ctrc 3	Y	Closed loop control is called every 3 millisecond	80
? ! ctrd	!ctrd 100	Y	Closed loop in target window for 100 milliseconds	80
? ! ctrt	!ctrt 200	Y	Closed loop control timeout after 200 milliseconds	80
? ! twi	!twi 0.01 0.01 0.01	Y	Set target window for X Y Z to 10µm (assume dim=2)	81
? ctrstatus	?ctrstatus 1	-	Get Closed Loop active state of all axes	81
? ctrdiff	?ctrdiff	-	Get Closed Loop position difference of all axes	82

Trigger Signal Configuration¹

Instruction	Example	Save	Brief description	Page
? ! trig	!trig 1	-	Enable trigger functionality (should be the last command)	83
? ! triga	!triga x	-	Trigger function is related to X axis	83
? ! trigm	!trigm 0	-	Select trigger mode 0	84
(!) trigger	trigger	-	Manually set trigger output (available in trigm 102, 103)	85
? ! trigs	!trigs 40	-	Set trigger output signal length to 40 microseconds	85
? ! trigd	!trigd 10	-	Set trigger distance to 10 (mm if dim=2)	85
? ! trigf	!trigf 1000	-	Generate periodic trigger pulses with 1kHz	87
? ! trigcount	?trigcount	-	Read number of generated trigger events	87

Snapshot Signal Configuration¹

Instruction	Example	Save	Brief description	Page
? ! sns	!sns 1	1	Enable snapshot functionality (should be last command)	88
? ! snsI	!snsI 0	Y	Set snapshot input signal to active low	88
? ! snsf	!snsf 10	Y	Set snapshot signal debounce filter to 10 milliseconds	89
? ! snsM	!snsM 0	Y	Set snapshot mode to 0(=capture, 1=move)	89
? ! snsc	?snsc	-	Read number of snapshot events (=array fill size)	89
? ! snsp	?snsp x	-	Read last captured X position	91
? ! snsA	?snsA 1	-	Read first position entry of snapshot array (all axes)	91
(!) snse	snse 2	-	Generate SnapShot event F2	92
? ! prehome	!prehome 10 20 1	-	Set prehome positions X Y Z to 10 20 1 (unit depends on dim seting)	93
? ! home	!home 5 5 0	-	Set home positions X Y Z to 5 5 0 (unit depends on dim seting)	93

¹ Function has to be enabled by factory, it is not available per default.

5. Instruction Syntax Description

Most instructions work in both directions (reading and writing). (?)! means the instruction accepts write and read. The controller identifies a read command by the preceded '?', or '!' for writing parameters.

Some examples for legal instruction syntax:

```
(?)!Command parameter1 parameter2 parameter3 parameter4  
(?)!Command parameter1 parameter2  
(?)!Command axis parameter  
(?)!Command
```

6. Error Numbers and their possible Root Cause

```
0    no error  
1    no valid axis name  
2    no executable instruction  
3    too many characters in command line  
4    invalid instruction  
5    number is not inside allowed range  
6    wrong number of parameters  
7    either ! or ? is missing  
8    no TVR possible, while axis active  
9    no ON or OFF of axis possible, while TVR active  
10   function not configured  
11   no move instruction possible, while joystick enabled  
12   limit switch active  
13   function not executable, because encoder detected  
21   multiple axis moves are forbidden (e.g. during initialization)  
22   automatic or manual move is not allowed (e.g. door open or initialization)  
27   emergency STOP is active  
29   servo amplifier are disabled (switched OFF)  
30   safety circuit out of order  
  
70   wrong CPLD data  
71   ETS error  
72   parameter is write protected (check lock bits)
```

7. Controller Informations

You may read the firmware version by sending the instruction 'version' to the controller. The instruction 'det' gives you further details of which options are enabled. Each controller has its own unique serial number readable with the instruction 'readsn'.

7.1. version (Read detailed Version information)

Syntax: ?version or version
Parameter: none or 1

Description: This instruction gives detailed information about the firmware version.

Sending the version instruction with parameter 1 returns the tango firmware version number only.

Example: ?version
TANGO-DT-S, Version 1.37, Aug 12 2008 , 16:39:01

 ?version 1
1.37

Response syntax: Character string including controller type, firmware version and build date separated by a comma:
TANGO Fixed string identifying the Tango controller
-DT Desktop version
-PCI PCI card version
-S Tango short card version (PCI-S, DT-S)
-MINI TANGOMini
-C Motorized Stage with integrated Controller
e Tango PCI-E card version (PCIE, DTe)
Version 1.37 Firmware version number
Aug 12 2008 Firmware build date
16:39:01 Firmware build time

7.2. det (Read detailed Configuration)

Syntax: ?det or det
Parameter: none
Description: This instruction delivers detailed information about the current controller configuration.
Response: The controller returns a decimal integer number. Its hexadecimal value represents the configuration, like following table shows:

0x0	- - - 1	1Vss-encoder is configured
0x0	- - - 2	MR-encoder is configured
0x0	- - - 4	TTL-encoder is configured
0x0	- - 3 -	this is the number of configured axes (e.g. 3)
0x0	- 1 - -	Display is configured
0x0	- 2 - -	Speedpoti is configured
0x0	- 4 - -	Hand wheel is configured
0x0	- 8 - -	Snapshot is configured
0x0	1 - - -	TVRin is configured
0x0	2 - - -	Trigger out is configured
0x0	8 - - -	TVRout is configured
0x1	- - - -	16 digital I/O are configured
0x2	- - - -	32 digital I/O are configured
0x4	- - - -	Trackball is configured
0x8	- - - -	ETS available

The current configuration results as a logical 'or' of these bits.

Example: Assume ?det delivers the response 81697 which is 13F21 hexadecimal. This number means in detail, that the controller is configured for:

1 => 16 digital I/O
3 => TVRin and Trigger out
F => Display, Speedpoti, Hand wheel and Snapshot
2 => 2 axis
1 => 1Vss encoder

7.3. readsn (Read Serial Number)

Syntax: ?readsn or readsn
Parameter: none
Description: The instruction ?readsn delivers the current serial number.
Example: ?readsn
Response: The controller transmits its unique serial number as ASCII characters like YYWWNNXXX.

YY year of manufacturing
DD week of manufacturing
NN available axes (in hardware)
XXX Index number

7.4. ver (Read default Version Number)

Syntax: ?ver or ver
Parameter: none

Description: This instruction reads back the default firmware version info. The first number is the number of configured axes. The second number is the maximum possible motor current in ampere. For Tango firmware version information please use "version".

Example: ?ver
Response syntax: Vers:LSnm.xx.xxx
 (in some cases Vers:ESnm.xx.xxx)

"Vers:LS"	Fixed character string
n	Number of configured axes: 1, 2, 3, or 4
m	Maximum Current: 1=1.25A, 2=2.5A, 3=3.75A
x	Fixed numbers

7.5. iver (Read internal Version Number)

Syntax: ?iver or iver
Parameter: none

Description: This instruction reads the internal version information string. Mostly unused. Please use the "version" command to read the Tango firmware version.

Response syntax: 14 characters, e.g. T[DD].[WW].[YY]-[NNNN]
 [DD] = Day of Week, [WW] = Week, [YY] = Year
 [NNNN] = Number

Example of ?iver response: T04.35.02-0004

7.6. uptime (Read Controller Up Time)

Syntax: ?uptime or uptime
Parameter: none

Description: This instruction reads how long the controller is running since power on or reset.

Response: Time in seconds.

Example: uptime

7.7. temp (Read Case Temperature)

Syntax: ?temp or temp
Parameter: none

Description: This instruction reads the temperature inside the controller. Only available with the encoder interface.

Response: Temperature in [°C] with one decimal place.

Example of temp response: 28.9

8. Communication Interface Settings

8.1. baud (Baud Rate)

Syntax: `!baud` or `?baud`

Parameter: 9600, 19200, 38400, 57600 or 115200

Description: This instruction sets or reads the serial communication transfer rate (baudrate). After sending this command first make sure the controlling device (e.g. a PC) has the same setting again. Then a save command may be sent to permanently store the new baudrate.

For PCI bus communication this instruction has no effect.

Response: Current baud rate.

Examples:

`!baud 57600` The baud rate is set to 57600 [Bd].

`?baud` query controller for current baud rate

8.2. cts (Enable/Disable RS232 Hardware Handshake)

Syntax: `?cts` or `!cts`

Parameter: 0 or 1

Description: Writing a 1 enables additional hardware handshake of the RS232 or USB interface. A 0 disables this function.

For PCI bus communication this instruction has no effect.

Please note that the PC COM port has to be opened in hardware handshake mode, too.

Response: Current state of CTS (0=disabled or 1=enabled)

Examples:

`?cts` query controller for current state of CTS

`!cts 0` disable CTS handshake

`!cts 1` enable CTS handshake

9. System Instructions

The controller provides two different instruction sets.

> The default instruction set as described in this manual.

> The second optional instruction set is a subset of the Venus command set. The following instruction let you select your required instruction set (if the option is installed).

9.1. save (Save Parameters)

Syntax: !save or save

Parameter: none

Description: The instruction !save stores your favorite parameter settings (like spindle pitch) in a permanent and safe data area. These parameters will be taken by the controller after each consecutive reset or power on as default values. Executing a save comand always returns the "OK..." string when writing is completed.

Response: The save instruction returns the response string "OK..."

Example:

save => The currently used controller parameters are saved as default.

9.2. restore (Restore Saved Parameters)

Syntax: !restore or restore

Parameter: none

Description: The controller reloads the saved parameters from its nonvolatile memory. The current controller parameters get overwritten by the saved defaults. Refer to the "save" instruction. Similar to a software "reset", but without affecting the hardware.

Response: none

Example: restore

9.3. reset (Force a Software Reset)

There are two possibilities to reset the controller:

- . The power on reset
- . The Software Reset

Syntax: !reset or reset

Parameter: none

Description: The controller is forced to perform a software reset. It is a restart similar to power on. Rebooting from reset will take more than 1 second, where the controller is not responding. There is no reply to a software reset. So for knowing if the controller is rebooted and ready, it may be neccessary to poll data until it responds again.

Response: none

Example: reset

9.4. pa (Enable or Disable the Power Amplifiers)

Syntax: !poweramplifier or !pa
Parameter: 0 or 1

Description: This instruction switches all motor amplifiers on(=1) or off(=0). If switched off, no motor current is flowing. To switch off axes individually, please use the 'axis' command.

Response: none

Example: !pa 1 Switch on all amplifiers.

9.5. ipreter (Select Instruction Set)

Syntax: !ipreter or ?ipreter
Parameter: 1, 2, 3 or 4

Description: 0 => Prohibited. Register command set is no longer provided.

1 => Default instruction set (Native), as described in this manual

2 => VENUS-1 and VENUS-2

3 => LUDL MAC5000

4 => Do not use for Firmware \leq 1.45!
From Firmware 1.46: ASI MS-2000

To return from the VENUS instruction set (2), please enter the string "1 setipreter" and press enter (or send an ASCII [CR]). For other instruction sets please refer to the corresponding instruction set description.

Response: 1, 2, 3 or 4

Example:
!ipreter 2 => The controller switches to the VENUS instruction set.
?ipreter => Responds the currently selected interpreter.

10. Operating Modes

10.1. Extended Mode

Activating Extended Mode will change some of the controllers behavior. Also there are new instructions for calibrate and range measure velocities.

Note: When initializing the controller, the desired Extended Mode should be set directly after setting **dim** and before setting gear, pitch, vel etc.

Calibration in extmode = 0:

!vel --> The velocity for moving towards an endswitch has to be set everytime before starting a **cal** / **rm** move.
!calbspeed --> There is only one velocity for all axes to travel out of the endswitch. The unit is 1/100 rev/s.

Calibration in extmode = 1:

!vel has no influence to the cal / rm move, same to **calbspeed**.
 Now the calibrate (**cal**) and range measure (**rm**) velocities can be assigned once and will be used for all time.
!calvel --> Set velocities for moving towards and out of the cal endswitch (E0)
!rmvel --> Set velocities for moving towards and out of the rm endswitch (EE)

Additional differences when in extmode = 1:

All the parameters stored in revolutions/s (e.g. vel) are recalculated if the pitch or gear values have changed. So the stage velocities will remain the same even when pitch or gear changed.

joyvel --> The joystick velocity can (and has to be) set independently from **vel** by the **joyvel** instruction.

The **?lim** command, when requested without an axis specifier, now returns all limits in a correctly formatted way.

10.1.1 extmode (Switch to Extended Mode)

Syntax: **!extmode** or **?extmode**

Parameter: 0 or 1

Description: This instruction switches the Tango controller into extended mode. This mode offers improved behavior and some more sophisticated commands then the standard interpreter. For further information please refer to the **Extended Mode** Chapter 10.1.

0 = default, compatible interpreter mode
 1 = extended interpreter mode.

Response: currently used extmode.

Examples:

!extmode 1 Set controller into extended mode.
?extmode Query extended mode.

10.2. Scan Mode

In Scan Mode the controller executes move instructions with a vector velocity.

10.2.1 scanmode (Switch to Scan Mode)

Syntax: !scanmode or ?scanmode
Parameter: 0, 1 or 2

Description: This instruction switches the Tango controller into scan mode. In this mode the vector velocity for automatic moves (moa, mor) is constant and can be set by 'scanvel'.

0 = normal operation (no scan mode)
1 = scan mode 1
2 = scan mode 2

Scan mode 1:

All axes have the same scanvel velocity. Individual 'vel' settings are ignored. If started individually (e.g. "!moa z 10") the axis will also travel at scanvel.

Scan mode 2:

When the automatic move is started as a vector or without an axis character (e.g. "moa 10 5", "moa 10") the axis/axes travel at scanvel.

If started individually (with specified axis, "moa z -10") the axis specific 'vel' velocity is used.

Response: Scanmode (automatic move mode) as integer.

Examples:
!scanmode 1 Set controller into scanmode 1.
?scanmode Query controller scanmode.

10.2.2 scanvel (Scanmode Vector Velocity)

Syntax: !scanvel or ?scanvel
Parameter: 0.000001 to 1000 [mm/s]

Description: This instruction sets or reads the scanmode vector velocity in millimeter per second. As this is a vector mode there is only one velocity parameter. Please also refer to the 'scanmode' instruction.

Response: Currently selected velocity in [mm/s]

Examples:
!scanvel 10 Set scanmode vector velocity to 10mm/s
?scanvel Query scan mode velocity.

11. Controller States and Error Messages

11.1. autostatus (Set Autostatus to required behavior)

Syntax: !autostatus or ?autostatus

Parameter: 0, 1, 2, 3 or 4

Description:

- 0 => Sending of any automatic state messages is switched OFF, except 'save'.
- 1 => After each automatic move the message 'position reached' (this is the character '@' for each configured axis) is automatically transmitted by the controller. It is the default configuration after power on.
- 2 => The controller transmits the message 'position reached' plus other status messages.
- 3 => Instead of 'position reached' string a simple <CR> is transmitted (=fast).
- 4 => Echoes all input instructions including parameters.

Autostatus can not be saved. After power on or reset it is always set to 1.

Example: Assume there are three axes configured and autostatus is set to 1.

After completion of a move (moa, mor, m, a) the controller will return a "@@@-." which means position reached.

!autostatus 0 Switch off autostatus (now the position reached info has to be polled by using the "statusaxis" instruction).
?autostatus Reads back the selected autostatus value.

11.2. statusaxis (Query State of Axis)

Syntax: ?statusaxis or statusaxis

Parameter: none

Description:

Statusaxis responds the state of each axis.
Similar to the 'autostatus 1' response of move commands, but with an additional '-' after the dot.
It can be used for polling axes in 'autostatus 0' mode, where no automatic response is generated.
Every response except of 'M' means the axis has stopped for some reason and may be ready for a new move command.

Response: 6 ASCII characters: [STATUS X][STATUS Y][STATUS Z][STATUS A].-

- @ => Axis is not moving and ready
- M => Axis is moving
- J => Axis is controlled manually (by joystick)
- C => Axis is in closed loop
- S => Limit switches are triggered and prevent further automatic move
- A => ok response after cal instruction
- D => ok response after rm instruction
- E => not o.k. response after cal or rm, if an error occurred during cal instruction (e.g. a limitswitch is not working properly)
- U => manual adjustment (e.g. 1st setup)
- T => Timeout (refer to 'caltimeout' instruction)
- => Axis is not enabled

Example: Assume ?statusaxis delivers the response @@@-.-

This means three axes are enabled and ready to move.

11.3. status (Query the Controller Error State)

Syntax: ?status or status
Parameter: none

Description: The ?status instruction responds with the current state of the controller. Which is either 'OK...' or an 'ERR' with error number. Also see 'err' instruction.

Response: OK... or ERR with error number

Example: ?status => ERR 4
 ?status => OK...

11.4. err (Query Error Number)

Syntax: ?err or err, !err
Parameter: none

Description: The instructions err or ?err return the controller error state or 0, if no error occurred. The error state will be updated or re-set by the next instruction. Additionally the error state may be cleared to zero by sending !err.

Response: Error number as decimal value
 (refer to Chapter 6. "Error Numbers")

Example: err => 0
 !err (clear error state if no permanent error)

11.5. help (Query Error Number with Description String)

Syntax: ?help or help
Parameter: none or requested error number

Description: The instruction help returns a text string. It contains the error state with appended error description. The error state is not cleared to zero. Please also refer to the 'err' instruction.

When called without a parameter:
It returns the controller's error state with description

When called with a parameter (error number):
It returns this error number with the corresponding comment

Response: Error number as decimal value, error description as ASCII text

Example: help => ERROR 0,no error
 (controller state, assumed to be ok here)

 help 29 => ERROR 29,servo amplifier off

11.6. service (Print Service Information to Terminal)

Syntax: ?service or service
Parameter: none

Description: The instruction service returns a multi-line parameter and state list of the controller. It may be used for debugging or in case of service requests. Either a terminal program or SwitchBoard version 1.19 and above can be used.

Response: Many lines of text including e.g. serial number, parameters, states etc.

Example: service

11.7. pci (Is PCI Bus)

Syntax: ?pci or pci
Parameter: none

Description: The instruction pci returns:

0 = Controller is a desktop version
1 = Controller is a PCI card and plugged in a PCI slot

Response: 0 or 1

Example: pci => 0

11.8. lock (Select Parameters to Lock)

Syntax: ?lock or !lock
Parameter: 0 bis 15, 0 oder 1

Description: Write protection for parameter. The lock bits must be applied to one or more axes by the lockaxis command. Else they have no effect.

Response: Lock bit state or entire lock bit field

0: Pitch
1: Gear
2: Cur
3: MotorSteps
4: SwPol
5: SwTyp
6: SwDir
7: EncTTL
8: EncPeriod
9: AxisDir
10: MotorTable
11: BackLash

Example: !lock 111 => Set lock bits 0 1 and 2, leave others unaffected
 !lock 2 0 => Clear lock condition for parameter 2 (=current)
 !lock 0 1 => Set lock bit for parameter 0 (pitch)
 ?lock => Query lock bit field (e.g. "0000000000000000")
 ?lock 5 => Query lock bit 5 state

11.9. isvel (Query Actual Velocities)

Syntax: `?isvel` or `isvel`

Parameter: `x,y,z,a` or none

Description: Read the actual velocity(ies) with which the axis is currently traveling. Unlike `'?vel'` or `'?speed'` this instruction returns the current real speed of the axes.

Response: Actual motor velocity in [mm/s]

Example: `?isvel` => Query actual velocity of all axes
`?isvel y` => Query actual velocity of the X axis

11.10. maxpos (Maximum Position)

Syntax: `?maxpos`

Parameter: none

Description: Query the maximum position value which the controller can accept due to internal limitations. It depends on e.g. the selected pitch, gear or motorsteps.

Response: Theoretical maximum position value of the axes
(unit depends on dim setting)

Example:
`?maxpos x` => 2600.0000 (X axis accepts position values in the range of ± 2600 mm, assuming dim = 2)

12. General Adjustments

With the following instructions the parameters of the controller are widely scalable to the given mechanic construction and to customer requirements. The controller is adaptable to the requested requirements.

12.1. dim (Unit for Positions)

Syntax: !dim or ?dim
Parameter: x, y, z or a
 0 to 9

Description: The dim instruction sets the unit (or "dimension") of all input and output parameters related to length, e.g. position or move commands.

The provided units for length (parameters for dim) are:

0 => Micro steps
1 => μm
2 => mm (Tango default)
3 => 360°
4 => revolutions
5 => cm
6 => m
7 => inch
8 => mil
9 => mm (difference to mode 2: all velocity instructions in mm/s)

Examples:

!dim 4 1 the selected dimension for X is [revolutions] and for Y is [μm].
!dim z 2 the selected dimension for Z is [mm]
?dim responds the dimensions for all axes.
?dim a responds the dimension of the a-axis.
dim 2 2 2 2 set dimension for all axes to [mm]

Response: Current settings

Hint: For dimensions 3 ($=360^\circ$) and 4 (=revolutions) it is recommended to set the **spindle pitch** to 1mm.

12.2. pitch (Spindle Pitch)

Syntax: !pitch or ?pitch
Parameter: x, y, z or a
 0.0001 to 68

Description: This instruction sends the spindle pitch (given by mechanic components) to the controller. It will be taken for all further calculations.

Response: current spindle pitch

Examples:

!pitch 4.0 1.0 set spindle pitch X=4[mm] and Y=1[mm]
!pitch z 2.0 set spindle pitch Z=2[mm]
?pitch query all axes for their spindle pitch
?pitch a query spindle pitch for a-axis

12.3. gear (Gear Ratio)

Syntax: !gear or ?gear
Parameter: x, y, z or a
 0.001 to 1000

Description: This instruction transmits the gear ratio to and from the controller. The ratio is 1, if the motor is directly mounted on the spindle.

Response: current gear ratio

Examples:

```
!gear 10           set gear ratio X=1/10
!gear 4.0 1.0      set gear ratio X=1/4 and Y=1/1
!gear z 10.0       set gear ratio Z=1/10
?gear              query all axes for their gear ratio
?gear a            query A-axis for its gear ratio
```

12.4. motorsteps (Motor Steps Per Revolution)

Syntax: !motorsteps or ?motorsteps
Parameter: x, y, z or a
 [multiples of 4]

Description: This instruction sets or reads back the steps per revolution of the attached motor. Commonly the most motors have 200 steps per revolution (which is 1.8° full step). Other motors may have 400 or 500 steps per revolution. The motor steps parameter must be a multiple of 4.

Response: Currently used motorsteps

Examples:

```
!motorsteps 200 200 400  set motor steps for X and Y to 200 and Z to 400
!motorsteps x 500        set motor steps for X to 500
?motorsteps               query all axes for their motorsteps
?motorsteps a             query A-axis for its motorsteps
```

12.5. accel (Acceleration)

Syntax: !accel or ?accel
Parameter: none, x, y, z or a
 0.0001 to 20 [m/s²]

Description: This instruction sets or reads the maximum acceleration which is used for all moves, the speed instruction and HDI devices.

Remarks: In case of a stop event, '**stopaccel**' is used instead.

Response: Currently used acceleration in m/s²

Examples:
!accel 0.5 set acceleration X=0.5[m/s²]. Other axes are not affected.
!accel 1 0.55 set acceleration X=1.0[m/s²] and Y=0.55[m/s²]
!accel z 0.2 set acceleration Z=0.2[m/s²]. Other axes are not affected.
?accel query all axes for their current acceleration.
?accel z query Z axis for its acceleration.

Additional information:

The standard ?vel query returns numbers with a resolution of 0.01 m/s², but the instruction accepts higher resolutions.

In case of Firmware 1.46 or higher: In order to read out more decimal places it is possible to add the requested decimal places to the query (valid is 0 to 6).

Examples:
?accel 6 => 0.123456 0.123456 0.123456
?accel y 4 => 0.1235 0.1235

12.6. accelfunc (Acceleration Ramp Function)

Syntax: !accelfunc or ?accelfunc
Parameter: none, x, y, z or a
 0, 1 or 2

Description: Select the acceleration ramp type for automatic moves (e.g. m, moa, mor, cal, rm).

0 = Linear acceleration/deceleration ramp
1 = sin² acceleration/deceleration ramp
2 = reserved, currently same as 1: sin²

Response: Currently used acceleration function

Examples:
!accelfunc 1 set accel function X to sin². Other axes are not affected.
!accelfunc x 1 set accel function X to sin². Other axes are not affected.
!accelfunc 1 1 0 set accel function in X and Y to sin², Z to linear.
?accelfunc query all axes for their current acceleration ramp functions.
?accelfunc z query Z axis for its acceleration ramp function.

12.7. stopaccel (Emergency Stop Deceleration)

Syntax: !stopaccel or ?stopaccel

Parameter: x, y, z or a
0.01 to 200 m/s²

Description: This instruction sets the deceleration for emergency stop conditions. It will be used by abort commands or when detecting an unexpected limit switch (e.g. no cal/rm move was performed).

Response: Currently used deceleration for stop

Examples:

!stopaccel 1 1 2 Set the stop deceleration for X and Y to 1 and Z to 2 [m/s²]

!stopaccel x 1.5 Set the X stop deceleration to 1.5[m/s²]

?stopaccel Returns the currently used stop deceleration for all axes

12.8. vel (Velocity)

Syntax: !vel or ?vel

Parameter: x, y, z or a
0.000001 to 200 [rev/s] or [mm/s] if dim = 9

Description: Velocity in motor revolutions per second, which is used for all consecutive automatic moves.

Remarks: If **extmode**=0 (default), the vel is also used for the HDI (joystick) velocity.

Response: Currently selected velocity

Examples:

!vel 10 set velocity X=10[revolution/s]. Other axis are not affected.

!vel 1.0 15 set velocity X=1[revolution/s] and Y=15[revolution/s].

!vel z 0.1 set velocity Z=0.1[revolution/s].

?vel query all axes for their velocities.

?vel x query x axis for its velocity.

Additional information:

The standard ?vel query returns numbers with a resolution of 0.001 rev/s, but the instruction accepts higher resolutions.

In order to read out more decimal places it is possible to add the requested decimal places to the query (valid is 0 to 16).

Examples:

?vel 6 => 0.123456 0.123456 0.123456

?vel y 4 => 0.1235 0.1235

If a higher resolution or slower velocity is needed, the '**velfac**' instruction can be used in addition.

12.9. velfac (Velocity Factor)

Syntax: !velfac or ?velfac
Parameter: x, y, z or a
 0.01 to 1.00

Description: This instruction sets or reads the velocity factor, which is used for all consecutive automatic moves. It is internally multiplied to the velocity (vel).

Response: Currently used velocity factor [0.01 to 1.00]

Examples:
?velfac z query X axis for its current velocity factor.
!velfac x 0.1 set velocity X to 1/10 of current velocity.
?velfac query all axes for their current velocity factors.

12.10. secvel (Secure Velocity)

Syntax: !secvel or ?secvel
Parameter: none, x, y, z or a
 1 to 100 [mm/s]

Description: The security speed limitation is used as long as the axis is not calibrated and range measured ("cal", "rm"). The velocity unit is always mm/s and does not depend on the "dim" state. It prevents the microscope stage from mechanical damage as long as the controller does not know the mechanical limits. It may also be used as a workaround, if running a cal/rm is not wanted.

Response: Currently used secure velocity [1 to 100 mm/s]

Examples:
!secvel 100 100 100 => Set maximum possible velocity of X Y Z
!secvel y 14.5 => Set maximum possible velocity of Y to 14.5 mm/s

12.11. maxcur (Query Maximum Motor Current)

Syntax: ?maxcur
Parameter: none, x, y, z or a

Description: This instruction reads the maximum possible motor current.

Response: maximum motor current in Ampere [A]

Examples:
?maxcur y query Y axis for its maximum motor current
?maxcur query all axes for their maximum motor currents

12.12. cur (Motor Current)

Syntax: !cur or ?cur
Parameter: x, y, z or a
 0.1 to [maximum current]

Description: This instruction sets or reads the motor current. The maximum current is limited by hardware and may be checked by the "**maxcur**" instruction.

Response: Motor current in Ampere

Examples:
!cur 1.0 set X motor current to 1[A]
!cur 1.0 2 set motor current for X=1[A] and Y=2[A]. Other axes are not affected.
!cur z 0.3 set Z motor current to 0.3[A].
?cur query all axes for their motor currents.
?cur x query X for its motor current.

12.13. reduction (Motor Current Reduction Factor)

Syntax: !reduction or ?reduction
Parameter: x, y, z or a
 0 to 1.0

Description: This instruction sets or reads the current reduction factor. When the axis is idle (stopped), the motor current is reduced by this factor. The floating point values from 0 to 1 represent a current of 0 to 100% of the selected motor current (**cur**). A value of 1 disables the reduction. Motor current reduction can be used to keep the motor temperature low, but as a side effect it may add noise, decrease performance and position accuracy. A delayed current reduction can be achieved by the "**curdelay**" instruction.

Response: Reduction factor(s) [0.00 to 1.00]

Examples:
!reduction .1 .7 Set idle currents X=0.1*cur[A] and Y=0.7*cur[A]
!reduction z 0.5 Set Z idle current to 0.5*cur[A]
?reduction Query all axes for their current reduction factors.
?reduction x Query X for its reduction factor.

12.14. curdelay (Delay for Current Reduction)

Syntax: !curelay or ?curelay

Parameter: x, y, z or a
0 to 10000 [ms]

Description: At the end of each move the axis enters the idle state. If the motor current reduction factor is set to a value less than 1.0 this reduction will take effect after the curelay time.

Response: Selected delay of current reduction in [ms]

Examples:

!curelay 100 300 set delay for motor current reduction X=100[ms] and Y=300[ms]

!curelay z 450 set delay for motor current reduction Z=450[ms]

?curelay query all axes for their motor current reduction delay.

?curelay x query X-axis for its motor current reduction delay.

12.15. axis (Enable, Disable, Switch Off Axis)

Syntax: !axis or ?axis

Parameter: x, y, z or a
-1, 0, 1

Description: This instruction enables, disables, switches off axes. Or reads its current state. A disabled axis still powers the motor with its current, while a switched off axis loses its torque.

Response: Current axis state (1=enabled, 0=disabled, -1=power stage off)

Examples:

!axis 1 1 1 1 enable all axes.

?axis x query X-axis for its state.

!axis 1 0 1 0 disable Y and A axis while X and Z are enabled.

!axis y -1 switch off Y axis: power stage Y off.

?axis query all axes for their state.

12.16. axisdir (Axis Direction)

Syntax: !axisdir or ?axisdir
Parameter: x, y, z or a
0 or 1

Description: This command reverses the specified axes. The meaning of the limit switches (E0 and EE) is also automatically exchanged against each other:
The E0 and EE switches are treated as virtual switches, E0 is the switch in negative direction, EE in positive. The hardware is reassigned to the opposite switch. Also for: swact, swpol, swtyp, readsw. Exception: The 'swin' function is not affected. Please make sure to first set the desired axis direction before setting the end switch types, polarity etc.!
It is not recommended to change direction during operation!

0 = Normal direction, CAL switch => E0, RM switch => EE
1 = Reversed direction, CAL switch => EE, RM switch => E0

Hint: If you need to change the assignment of E0 and EE

Response: Current axis direction is 0=not changed or 1=changed

Examples:
!axisdir 0 1 0 1 Axis directions of Y and A are reversed.
?axisdir x Query X for its current axis direction.

12.17. motortable (Motor Correction Table)

Syntax: !motortable or ?motortable
Parameter: x, y, z or a
0 or number specified by factory

Description: This instruction adds a motor correction. The motor has to be measured for the specific application by factory. There a table number will be assigned and the customer may activate it by setting the corresponding motortable number. Using a wrong motortable will lead to increased noise and position error.

0 = No correction

Response: Currently used motortable(s)

Examples:
!motortable 1 1 2 0 Select motortable 1 for X and Y, 2 for Z and no for A
!motortable x 0 Disable correction in x
?motortable Returns the currently used tables for all axes

12.18. usteps (Microstep Resolution)

Syntax: !usteps or ?usteps
Parameter: 360 ... 819200

Description: This instruction sets the microstep resolution of one motor revolution. Only one resolution for all axes. It is used when dimension Micro steps (dim 0) is selected.

Response: Currently used microstepping resolution

Examples:
!usteps 40000 Set microstep resolution to 40000/revolution
?usteps Query microstep resolution

12.19. resolution (Position Number Format)

Syntax: !resolution or ?resolution
Parameter: 0 ... 6

Description: This instruction sets the number of decimal places for "?pos" instructions in dim modes 2 and 9.
One value applies to all axes, default = 4 (100nm resolution).

Response: Currently responded decimal places for the pos instruction.

Examples:
!resolution 6 Set position string resolution to 6 decimal places (0.000000)
? resolution Query decimal places

12.20. backlash (Mechanical Backlash Compensation)

Syntax: !backlash or ?backlash
Parameter: x, y, z, a or none
 -100.0 ... 100.0 [µm]

Description: This instruction compensates mechanical backlash separately for each axis. Unit is µm, independent from dim.

0 = Backlash compensation off

Response: Currently used backlash in µm

Examples:
!backlash 12.7 21.3 0 Set backlash for X to 12.7µm, Y=21.3µm and Z= none.
!backlash x 0 No backlash compensation for X
!backlash x 5 Compensate a backlash in X is of 5µm
?backlash Returns the currently used backlash for all axes

12.21. lockaxis (Apply the Parameter Lock to Axes)

Syntax: ?lockaxis or !lockaxis

Parameter: x,y,z,a or none

Description: Apply the lock to an axis. If the lock instruction is set to all zero, there is no effect and vice versa.

Response: Axes to which the lock bits are currently applied.

Example: !lockaxis y 1 => Apply lock bits to Y axis
!lockaxis 1 1 => Apply lock bits to X and Y axis
?lockaxis x => Query if lock bits are applied to the X axis
?lockaxis => Query all axes (returns e.g. "1 1 0 0");

12.22. lockstate (Query all internal Lock States)

Syntax: ?lockstate

Parameter: x,y,z,a or none

Description: Set/read lock bits corresponding to the parameter listed below
0: Pitch
1: Gear
2: Cur
3: MotorSteps
4: SwPol
5: SwTyp
6: SwDir
7: EncTTL
8: EncPeriod
9: AxisDir
10: MotorTable
11: BackLash

Response: Lock state as 16bits ASCII string ('0' and '1', LSB first)

Example: ?lockstate => Query lock state of all axes
?lockstate x => Lock state of X axis e.g. "1100000000000000"

12.23. stout (Select Status Signal Output)

Syntax: !stout or ?stout

Parameter: 0,1,2,3,4

Description: Makes the state of the Status LED available to the AUX-I/O connector:

0 = Just Status LED, no AUX-I/O used (Standard)
~~1 = AUX-I/O Pin 5 (TAKT_OUT) may not be available!~~
2 = AUX-I/O Pin 6 (VR_OUT)
3 = AUX-I/O Pin 7 (SHUTTER_OUT)
4 = AUX-I/O Pin 8 (TRIGGER_OUT)

Response: Actual status output mode

Example: !stout 0 => Only use Status LED
?stout => Query actual Status output mode (0,1,2,3 or 4)

13. Limit Switch Instructions (Hardware and Software)

13.1. lim (Software Limits)

Syntax: `!lim or ?lim`
Parameter: `x, y, z or a`
`+/- maximum position range`

Description: This instruction sets the maximum allowed positioning range. The upper and lower software limits shall be send together in a single `!lim` instruction. Remember: The unit (dimension) of the transmitted numbers depends on the value of instruction `dim`.

Hint: In **Extended Mode** (`extmode = 1`) the `?lim` command returns the limits as a correctly formatted string)

Response: Currently used software limits

Examples:

```
!lim -1000 1000 -2000 2000    set the software limits for X and Y.
!lim z -500 1700              set the software limits for Z.
?lim y                        query Y-axis for its limits.
?lim                           query all axes for their limits, only recommended
                               in extmode=1

?lim response example for 3 axes in
--> extmode=0: -1000 1000,[CR]-1000 1000, -1000 100[CR]
--> extmode=1: -1000 1000 -1000 1000 -1000 100[CR]
```

13.2. limctr (Enable or Disable Limit Control)

Syntax: `!limctr or ?limctr`
Parameter: `x, y, z or a`
`0 or 1`

Description: This instruction enables or disables the limit control or shows the current state of limit control. Attention, be careful: If limit controls are disabled, the controller doesn't care about limits. In this case the controller may damage system components. Limit control is enabled by default from power on.

Response: Limit control state (0 = not active, 1 = active)

Example:

```
!limctr y 0                  disable Y limit control, Y axis limit switches are ignored
!limctr 1 1 1                enable X,Y and Z limit control
!limctr z 1                  enable Z limit control
?limctr a                    query A-axis for its status of limit control
?limctr                      query all axes for their status of limit control
```

13.3. nosetlimit (Do not set limits by cal/rm)

Syntax: !nosetlimit or ?nosetlimit
Parameter: x, y, z or a
 0 or 1

Description: This command enables or disables the setting of software limit switches during calibration and range measure. The default is nosetlimit=0 which means that the software limits are set by the cal/rm moves to these min/max positions.

Response: 0 = set software limits to !cal and !rm positions
 1 = do not change software limits after !cal and/or !rm

Examples:
!nosetlimit 1 1 X and Y axis do not take software limits after !cal and !rm
!nosetlimit y 1 Y axis is does not set software limits of !cal and !rm move
?nosetlimit query all axes for their nosetlimit state
?nosetlimit a query A axis for its nosetlimit state

13.4. swtyp (Type of Limit Switch)

Syntax: !swtyp or ?swtyp
Parameter: x, y, z or a
 0 or 1

Description: Set/read the type of the limit switches.
 The sequence is E0 REF EE for all axes.
 The REF switch currently not used by the Tango controller.
 Important: When using no axis parameter (x,y,z or a), the 3 values will be used for all axes! To set individual axes, please do this separately, use the axis parameter x,y,z or a.
 Please note that the E0 and EE switch are reassigned by the 'axisdir' command.

0 = PNP, which adds a pull-down resistor to the switch input
1 = NPN, which adds a pull-up resistor (default)

Response: Currently selected type

Examples:
!swtyp 1 0 1 Set all limit switches to NPN type
!swtyp z 0 0 1 Set Z axis limit switches E0=PNP, REF=don't care, EE=NPN
?swtyp y Query Y axis for its switch type

13.5. swpol (Polarity of Limit Switch)

Syntax: !swpol or ?swpol
Parameter: x, y, z or a
 0 or 1

Description: Set/read the polarity of the limit switches.
 The sequence is E0 REF EE for all axes.
 The REF switch currently not used by the Tango controller.
 Important: When using no axis parameter (x,y,z or a), the 3 values will be used for all axes! To set individual axes, please do this separately, use the axis parameter x,y,z or a.
 Please note that the E0 and EE switch are reassigned by the 'axisdir' command.

 0 = switch has active low signal
 1 = switch has active high signal

Response: Polarity of the limit switches

Examples:

!swpol y 1 1 1	set polarity of Y limit switches (E0 REF EE) to positive edge.
!swpol 1 0 1	set polarity of limit switches (E0 REF EE) for all axes.
!swpol z 0 0 0	set polarity of Y limit switches (E0 REF EE) to negative edge.
?swpol a	query limit switch polarity of the A axis

13.6. swact (enable or disable limit switches)

Syntax: !swact or ?swact
Parameter: x, y, z or a
 0 or 1

Description: This instruction enables or disables the limit switches.
 The sequence is always:

E0 REF EE

0 = switch is inactive (actuation state is ignored)
1 = switch is active

The REF switch is not used by the Tango controller.
Disabling limit switches may damage the hardware.
When using no axis parameter, the 3 values will be used for all axes! To set individual axes please do this separately, use the axis parameter x, y, z or a.
If a switch is set to inactive, it is not necessary to cal or rm this switch to abolish the secvel speed limitation.
Inactive switches always return a non actuated state when using the '?readsw' command. But the 'swin' command still returns the switches TTL logic level state.
Please note that the E0 and EE switch are reassigned by the 'axisdir' command.

Response: Current state, if limit switches are enabled or disabled

Examples:

!swact 1 0 1	Enable cal and rm limit switches <u>for all axes</u> (REF disabled)
!swact z 1 0 1	Set Z limit switches E0=enabled REF=disabled EE=enabled
?swact a	Query A-axis, if limit switches are enabled or disabled

13.7. swdir (swap assignment of cal and rm switch)

Syntax: !swdir or ?swdir
Parameter: x, y, z or a
0 or 1

Description: This command swaps the cal(E0) and rm(EE) switch assignment.

0 = switches are not swapped
1 = switches are swapped

In addition to the axisdir command, which swaps motor direction and endswitch assignment, this command only swaps the switches. E0<->EE.

Caution: swdir should only be used to compensate different wiring of the stage endswitches. **Swapping the switches to the wrong assignment may result in microscope stage damage!**

Response: Current state of endswith assignment(s)

Examples:

!swdir 1 1 0 Swap E0<->EE switch assignment in X and Y, not in Z
!swdir x 1 Swap E0<->EE switch assignment in X (E0 switch is now EE etc.)
?swdir Query all axes for their switch assignment
?swdir z Query Z-axis, if its limit switches are swapped

13.8. readsw (Read Status of Limit Switches)

Syntax: ?readsw
Parameter: none

Description: This instruction reads the limit switch state of all axes. The response is a string of 12 characters, either 0 or 1.

0 = limit switch is currently not actuated or disabled
1 = limit switch is currently actuated (axis is in switch)

In opposite to the "swin" command, readsw returns the active/inactive state and not the signal level of the switch. Also the readsw exchanges E0 and EE switches when axisdir is changed.

Please note that the switch state is only valid when the swtyp, swpol parameters are set correctly and the switch is activated by swact.

Sequence of the 12 characters is:

Axis: X Y Z A X Y Z A X Y Z A
Switch: [E0] [E0] [E0] [E0] [Ref] [Ref] [Ref] [Ref] [EE] [EE] [EE] [EE]

E0 = lower limit switch (!cal command)
Ref = Reference switch
EE = upper limit switch (!rm command)

Response: Actuation state of limit switches, 0 if disabled

Examples: ?readsw => 000000000000
(query all limit switch actuation states)

13.9. swin (Read Limit Switch Input Level)

Syntax: ?swin or swin
Parameter: none or 0...7

Description: This instruction reads the limit switch signal directly.
 The response is a string of 8 characters, either 0 or 1.

0 = limit switch input signal is TTL low
1 = limit switch input signal is TTL high

In opposite to the "readsw" command, swin reflects the TTL input levels. Also disabled switches are represented with their current TTL input signal level. Swin is not affected by the axisdir command (does not exchange E0 and EE switches). The Ref signals are not used.

Sequence of the 8 characters is:
Axis: X Y Z A
Switch: [E0][EE][E0][EE][E0][EE][E0][EE]
Index : 0 1 2 3 4 5 6 7
E0 = lower limit switch (!cal)
EE = upper limit switch (!rm)

Response: Switch input signal TTL levelstate of limit switches

Examples: swin => 11111111 (query all 8 limit switch signal levels)
 swin 1 => 1 (query EE of X Axis signal level)

13.10. statuslimit (Limit Status)

Syntax: ?statuslimit or statuslimit

Parameter: none

Description: The instruction statuslimit responds a string with 16 characters. They represent the state of the software limits.

They are arranged in 3 groups:

- 0 - 3: Group 1 => cal state of axis 0-3 (x,y,z,a)
- 4 - 7: Group 2 => rm state of axis 0-3 (x,y,z,a)
- 8 - 11: Group 3 => lower software limit state of axis 0-3 (x,y,z,a)
- 12 - 15: Group 4 => upper software limit state of axis 0-3 (x,y,z,a)

The characters may have one of four values:

- => the software limit has not yet been modified since power on
- A => axis is already calibrated (!cal)
- D => axis is already range measured (!rm)
- L => software limit has been modified by (!lim)

Example: Assume ?statuslimit returns the string "AA-A---D-LL-L--L"

This means in detail:

- [0] A -> X-axis is calibrated
- [1] A -> Y-axis is calibrated
- [2] - -> Z-axis is not calibrated
- [3] A -> A-axis is calibrated
- [4] - -> X-axis is not range measured
- [5] - -> Y-axis is not range measured
- [6] - -> Z-axis is not range measured
- [7] D -> A-axis is range measured
- [8] - -> X-axis lower software limit is not modified
- [9] L -> Y-axis lower software limit is modified
- [10] L -> Z-axis lower software limit is modified
- [11] - -> A-axis lower software limit is not modified
- [12] L -> X-axis upper software limit is modified
- [13] - -> Y-axis upper software limit is not modified
- [14] - -> Z-axis upper software limit is not modified
- [15] L -> A-axis upper software limit is modified

14. Calibration and Range Measure Instructions

After each power on or '!reset' of the controller the operator may execute a calibration (instruction !cal) followed by a range measure (instruction !rm), if the system is equipped with the corresponding limit switches.

This also disables the travel speed limit set by 'secvel'. So the controller is able to smoothly stop the axes automatically before they crash into the microscope stage hardware limits.

The cal/rm instructions set the limits close to the limit switch positions. An additional position offset for the these limits may be specified with the instructions !caliboffset and !rmoffset.

Long axes and/or slow velocities may exceed the default calibration timeout of 40 seconds. Therefore the timeout can be set to the desired value by **caltimeout**. Please also refer to the optional **extmode** enhancements and **calmode** options for calibration.

14.1. cal (Command a Calibration)

Syntax: !cal or cal

Parameter: x, y, z or a

Description: This instruction moves either the specified or all currently enabled axes in negative direction towards lower positions, until the limitswitch E0 is detected. Depending on extmode it then moves with !calbspeed out of the switch. If caloffset=0, the axis will stop moving as soon as the limitswitch E0 is released. If caliboffset>0, the axis will continue moving to this distance. In both cases the reached position will be set to 0 and, if nosetlimit=1 used as lower software limit.

Response: each commanded (and enabled) axis answers either 'A' after a successful calibration or 'E' if an error occurred (cal was unsuccessful).

Examples:

```
!cal          execute a calibration for all enabled axes
cal y         execute a calibration for Y axis only (if Y is enabled)
```

14.2. rm (Command a Range Measure)

Syntax: !rm or rm

Parameter: x, y, z or a

Description: This instruction moves either the specified or all currently enabled axes in positive direction towards higher positions, until the limitswitch EE is detected. Depending on extmode it then moves with !calbspeed out of the switch. If rmoffset=0, the axis will stop moving as soon as the limitswitch EE is released. If rmoffset>0, the axis will continue moving to this distance. If nosetlimit=1, the reached position will be used as upper software limit.

Response: each commanded (and enabled) axis responds either 'D' after a successful range measure or 'E' if an error occurred (rm was unsuccessful).

Examples:

```
!rm          execute a range measure for all enabled axes
rm x         execute a range measure for X axis only (if enabled)
```

14.3. calmode (Closed Loop/Calibration Behavior)

Syntax: !calmode or ?calmode

Parameter: x, y, z, or a
0, 1 or 2

Description: This instruction reads or sets the closed loop behavior on power-up and also affects the calibration behavior:

0 = Cal instruction sets the zero position (default setting)
Closed loop enabled after cal move

1 = the power-up position remains the zero position even after calibration
Closed loop is enabled from power-up

2 = Cal instruction sets the zero position
Closed loop is enabled from power-up

Remarks: For activating the closed loop, the '**encmask**' of the corresponding axes must be set to one. If encmask is not set or no encoder is present, the calmode only affects the axis zero position behavior.

Response: Calmode behavior.

Examples:

!calmode 0 0 0 Calmode behavior of axes X,Y,Z set to normal operation
!calmode 2 2 Axes X and Y enter closed loop after controller power-up
?calmode y Query Y-axis for its calmode

14.4. caltimeout (Calibration Timeout)

Syntax: !caltimeout or ? caltimeout

Parameter: x, y, z or a
maximum time 1 to 120 (seconds as integer, no floating point)

Description: This instruction specifies the timeout for calibration (cal) and range measure (rm) moves. It may be set for each axis individually.

Response: Calibration timeout in seconds

Examples:

!caltimeout x 40 set the calibration timeout in X to 40 seconds
?caltimeout query all axes for their timeout

14.5. caliboffset (Calibration Offset)

Syntax: !caliboffset or ?caliboffset
Parameter: x, y, z or a
 Position

Description: This instruction specifies an extra offset position above the limitswitch E0 (towards higher positions) where to zero the axis and take this position as lower software limit. The unit depends on the current value of instruction dim. Allowed range is 0 to 30mm equivalent.

Response: Current calibration offset

Examples:

```
?caliboffset y          query the Y-axis for its current calibration offset
!caliboffset 1 1 1      set the calibration offset to 1 for X,Y and Z.
```

14.6. roffset (Range Measure Position Offset)

Syntax: !rmoffset or ?rmoffset
Parameter: x, y, z or a
 Position

Description: This instruction specifies an extra offset position below the limitswitch EE (towards lower positions) where to take the upper software limit. The unit depends on the current value of instruction dim. Allowed range is 0 to 30mm equivalent.

Response: current calibration offset

Examples:

```
?rmoffset z            query the Z-axis for its current range measure offset
!rmoffset 1 1 1        set the range measure offset to 1 for X,Y and Z.
```

14.7. caldir (Calibration Direction)

Syntax: !caldir or ?caldir
Parameter: x, y, z or a
 0 or 1

Description: This instruction set the calibration direction to either positive or negative positions. Default is negative direction. If set to positive(=1), the upper software limit is set. This instruction is not possible for systems with encoders.

Response: 0 = cal move to negative direction
 1 = cal move to positive direction

Examples:

```
!caldir y 1            set Y axis calibration direction to positive
!caldir 0 0 1          set Z axis calibration direction to negative
?caldir                query all axes for their current calibration directions
```

14.8. calbspeed (Calibration Speed for Retraction)

Syntax: `!calbspeed` or `?calbspeed`
Parameter: range 1 to 100 [$*0.01$ revolution/s]

Description: **This instruction is not used in extmode 1.**
Set or read the cal,rm calibration speed which is taken for traveling out of the limit switches E0 and EE. There is only one value for all axes.

Hint: Please refer to the **calvel**, **rmvel** instructions for improved **extmode 1** behavior.

Response: Currently used calibration speed

Examples:
`!calbspeed 15` set the retraction speed to 0.15 [revolutions/s] for all axes.
`?calbspeed` query the controller for current retraction speed.

14.9. calrefspeed (Reference Signal Calibration Speed)

Syntax: `!calrefspeed` or `?calrefspeed`
Parameter: range 1 - 100 [$*0.01$ revolution/s]

Description: This instruction transmits the reference mark calibration speed. This speed is taken for systems with encoders, when searching the reference on the scale. The default is 32. There is only one value for all axes.

Response: Currently used calrefspeed [in 1/100 rev/s]

Examples:
`!calrefspeed 5` set referencing speed to 0.05 [revolutions/s] for all axes.

14.10. calpos (Calibration Position)

Syntax: `!calpos` or `?calpos`
Parameter: x, y, z or a
position value

Description: This instruction is used for systems with encoders only. During calibration the encoder signal period is stored as soon as the E0 limit switch is left and may be queried later on with `?calpos`.
This position may also be set to an other value. The value depends on the unit set by "dim". Allowed range is 0 to 30mm equivalent.

Response: within range of one encoder signal period

Examples:
`?calpos y` query Y-axis for its calibration position
`!calpos 0 0 0` set calibration position to zero (X,Y and Z)
`?calpos` query all axes for their read calibration position

14.11. refdir (Direction for Searching Reference Signal)

Syntax: !refdir or ?refdir
Parameter: x, y, z or a
0 or 1

Description: This instruction is intended for systems with encoders and transmits the current or requested direction for searching the reference point on the scale. The default is 0 for minus direction.

Response: 0 = search in negative direction
1 = search in positive direction

Examples:
!refdir y 1 set the Y-axis reference search to positive direction
?refdir query all axes for their reference search directions

14.12. calvel (Calibration Velocities for CAL Instruction)

Syntax: !calvel or ?calvel
Parameter: none, x, y, z or a
two velocities > 0.0 [revolution/s] or [mm/s] if dim=9

Description: **This instruction is accessible in extmode 1 only.**
As a superset of the regular calbspeed command in combination with the vel command it now sets the two calibration velocities for the **cal** instruction (towards and out of cal endswitch). Out speed should be set slow for high accuracy.

Response: Two velocities (towards and out of endswitch) per axis

Examples:
!calvel x 10 0.5 Cal in X moves towards endswitch with velocity 10 [rev/s] or [mm/s], depending on dim and out of the endswitch with velocity 0.5
?calvel query all axes for their cal velocities
?calvel y query Y-axis for its cal velocities

14.13. rmvel (Range Measure Velocities for RM Instruction)

Syntax: !rmvel or ?rmvel
Parameter: x, y, z or a
two velocities > 0.0 [revolution/s] or [mm/s] if dim=9

Description: **This instruction is accessible in extmode 1 only.**
As a superset of the regular calbspeed command in combination with the vel command it now sets the two range measure velocities for the **rm** instruction (towards and out of rm endswitch). Out speed should be set slow for high accuracy.

Response: Two velocities (towards and out of endswitch) per axis

Examples:
!rmvel x 10 0.5 Rm in X moves towards endswitch with velocity 10 [rev/s] or [mm/s], depending on dim and out of the endswitch with velocity 0.5
?rmvel query all axes for their rm velocities
?rmvel y query Y-axis for its rm velocities

14.14. autopitch (Measure Pitch after CAL Instruction)

Syntax: !autopitch or ?autopitch

Parameter: x, y, z or a
0 or 1

Description: Measures and sets the spindle pitch each time when executing a cal instruction.

Hint: Only works if encoders are present.
Not necessary for spindles.

Response: Autopitch enabled (1) or disabled (0, default)

Examples:

!autopitch 1 1 0 Measure and readjust pitch after each cal instruction X and Y
?autopitch x query X-axis for its autopitch setting

15. Move Instructions

All move instructions include an automatic linear interpolation. Axes, which are started together reach their destination at the same time (vector move). Nevertheless, the accelerations and velocities of each axis are not hurt. The user can also start the axis independently from each other. In this case each axis drives with its own maximum parameters and the axes do not reach the destination at the same time.

E.g. `!moa 10 10` followed by `!moa z 1` makes X and Y move as a vector and Z independently.

15.1. moa (Move Absolute)

Syntax: `!moa` or `moa`
Parameter: none or x, y, z, a
position values within $\pm\text{maxpos}$

Description: This instruction moves one or more axes to a requested destination. The unit of the input numbers depends on instruction dim.

Response: Each commanded (and enabled) axis responds either '@' after successfully completing the move, or 'E' if an error occurred. The response depends on the autostatus setting which is assumed to be 1.

Examples:
`moa x 10.2` the X-axis is moved to position 10.2 ([mm] assume dim=2)
`moa 10 0 20` the axes X,Y,Z are moved to the given position
`moa y 34.5` the Y-axis is moved to position 34.5 ([mm] assume dim=2)

15.2. mor (Move Relative)

Syntax: `!mor` or `mor`
Parameter: none or x, y, z, a

Description: This instruction moves one or more axes relative to the current position. The unit of the input numbers depends on instruction dim.

Response: Each commanded (and enabled) axis responds either '@' after successfully completing the move, or 'E' if an error occurred. The response depends on the autostatus setting which is assumed to be 1.

Examples:
`mor x 12.3` the X-axis is moved by 12.3 ([μm] assume dim=1)
`!mor 1 1` the X and Y-axis are moved by 1 ([mm] assume dim=2)

`!mor a 298` the A-axis is moved by 298 (unit depends on dim)
`!mor -10 0 0 -10` the X and A-axis are moved by -10 (unit depends on dim)

15.3. m (Move Relative Shortcut)

Syntax: !m or m
Parameter: none

Description: The instruction is a shortcut (abbreviation) of mor. It is useful to speed up communication especially for consecutive identical vectors. The vector is taken from the preceding !mor or !distance instruction. The instruction will move enabled axes if their distance is not zero.

Response: Depends on state of autostatus, recommended is mode 3.

This is one example of consecutive moves:

```
!moa 1 2 3 4    will position to 1 2 3 4
!mor 1 1 1 1     will move to 2 3 4 5
m              will move to 3 4 5 6
!distance 0 2 0 0
m              will move to 3 6 5 6
```

15.4. distance (Distance for m)

Syntax: !distance or ?distance
Parameter: x, y, z or a
Distance (+-2600mm max.)

Description: This instruction transmits the travel distance for !m instructions. The unit depends on the selected dimension (refer to "dim" instruction).
Hint: The distance value is also set by executing a !mor command.

Response: Currently used value for distance (unit depends on 'dim')

Examples:

```
?distance      query current distance values for all axes
?distance z    query Z-axis for its distance value
!distance 10 20 set X and Y distance
!distance 1 2 3 set X,Y and Z distance
!distance y 20  set Y distance. Other axes keep their distance value.
```

15.5. moc (Move to Center)

Syntax: !moc or moc
Parameter: x, y, z or a

Description: This instruction centers all enabled axis to the midpoint between lower and upper software limits. The recommended precondition is to first execute the instructions !cal and !rm.

Response: each successful centered axis responds with ,@'.

Examples:

```
moc      moves all axes to their centers
moc z    the Z-axis is centered, with no move of other axis
```

15.6. go (Go To Position)

Syntax: !go or go
Parameter: none or x, y, z, a
 position values within $\pm\text{maxpos}$

Description: Used for position tracking applications.
 Similar to the 'moa' instruction 'go' executes a move of one or more axes to an absolute position.

The differences to move instructions are:

- Go can be overwritten anytime, also when moving, by another go position (without the need to abort or stop it first)
- Go is no vector move, each axis moves at its own velocity '**vel**'
- No autostatus reply on completion (polling of '**statusaxis**' necessary)

The unit of the input numbers depends on instruction dim.

Remarks: In order to check for a completed go move, please poll the '**statusaxis**' state, which should not be 'M' then.

Response: None.

Examples:
go x 10.2 the X-axis moves to position 10.2 ([mm] assume dim=2)
go 10.7 14 the axes X,Y move to the given position
go 10.1 -0.5 0 the axes X,Y,Z move to the given position

15.7. speed (Speed Move)

Syntax: !speed or ?speed
Parameter: x, y, z or a
 +/- 100

Description: This instruction moves one or more axes with requested speed, and not to a specified position. The command may be stopped by setting speed back to zero.
 As this is a "digital Joystick" function, this command only affects axes with Joystick enabled.

Response: current value for speed [revolutions/s]

Examples:
!speed 33 0.01 set speed for X= 33[revolutions/s] and Y= 0.01[revolutions/s]
!speed 0 set speed for X-axis to 0[revolutions/s] (stop X axis)
!speed 10 set speed for X-axis to 10[revolutions/s]
!speed y 0.001 set speed for Y-axis
?speed query all axes for their current speed
?speed z query Z-axis for current speed

15.8. a (Abort the Current Move)

Syntax: !a or a
Parameter: none

Description: This instruction stops all axes and sets them into position reached state.
You may also send a "Ctrl+C" (hex 0x03) command instead.

Response: Each configured axis responds an ,@'.

Example: a

15.9. delay (Set the Delay Time for Consecutive Moves)

Syntax: ?delay or !delay
Parameter: 0 to 10000 [ms]

Description: This instruction will insert a delay time between the reception and execution of move commands (delayed start).
There is only one value for all axes.

Response: Delay time in [ms]

Examples:
!delay 500 Set the delay time to 0.5 seconds
?delay Query the current delay time

15.10. pause (Set the Pause after Position Reached)

Syntax: ?pause or !pause
Parameter: 0 to 10000 [ms]

Description: Complementary to "delay", this instruction adds a pause time after the axes have reached their target positions. In autostatus 1 mode the "@@@" response is delayed by this time. It may be used to insert an automatic settling time after a move command.

Response: Pause time currently used, in [ms]

Examples:
!pause 10 Delay the autostatus response by 10 milliseconds
?pause Query the current pause time

15.11. pos (Read or Set Position)

Syntax: !pos or ?pos
Parameter: x, y, z or a
 Position (+- 2600mm max.)

Description: This instruction either reads or sets the current position.
 The used position unit depends on the selected dimension
 (refer to "dim" instruction).
 If an encoder and the 'encpos' is enabled, it returns the
 encoder position of the axis.

Response: Axis position(s) (depends on dim, and enc/encpos state)

Examples:

?pos	Query all axes for their positions
!pos100 200	Set positions of X=100 and Y=200 (unit depends on dim)
!pos 0.1	Set the position X=0.1 (unit depends on dim)
!pos y 2000	Set the position Y=2000 (unit depends on dim)
?pos z	Query Z-axis for its position

15.12. zero (Set Internal Position to Zero)

Syntax: !zero or zero

Parameter: x, y, z, a (or none)

Description: Unlike the command "!pos 0" this "!zero" instruction resets the internal position counter to zero.
It has to be used in applications where axes exceed the position limits, e.g. filter wheels (in such case a "!pos 0" instruction is not sufficient).
The zero instruction should be executed after completing one or several complete revolutions, before reaching the software limits. So the reference point remains at the same position.

Response: none.

Examples:

!zero Set all internal positions to zero
!zero z Set Z axis position to zero

15.13. clearpos (Set Internal Position to Zero)

Syntax: !clearpos or clearpos

Parameter: x, y, z, a (or none)

Description: For compatibility with LStep controllers.
Functionality is almost the same as with the 'zero' instruction.
The only difference is that the clearpos instruction is not executed when in closed loop.

Response: none.

Examples:

!clearpos Set all internal positions to zero
!clearpos x Set X axis position to zero

16. HDI: Joystick, Tackball and Handwheel Instructions

The HDI device velocities are limited by the '**secvel**' velocity as long as no '**cal**'/'**rm**' instruction has been executed. The axis travel will stop on either the hardware limit switches or the adjustable software limits '**lim**'.

The velocities are either taken from the current axis velocity '**vel**' or, if '**extmode**' is enabled as an independent '**joyvel**'.

The human device interface (HDI) tolerates hot plugging of the devices: It is possible to unplug, plug or change the input devices during operation of the controller.

The '**keymode**' functionality enables selection of different '**keyspeed**' or '**zwtravel**' wheel velocities by pressing the function keys of the joystick. Please refer to '**keymode**' for further informations.

16.1. joy (Generally Enable/Disable/Set Joystick Mode)

Syntax: !joy or ?joy

Parameter: 0, 1, 2, 3, 4 or 5

Description: Generally enable / disable the connected HDI device.
It is recommended to only use the values 0 or 2.

!joy 0 disable the HDI device (joystick, trackball etc.)
!joy 2 enables the HDI device (=default)

Response: HDI mode (joystick, trackball, etc.)

Examples:

!joy 2 set HDI mode 2 (HDI on, device e.g. joystick can be used)
?joy query the HDI mode

16.2. joydir (Joystick Direction or Assign Joystick)

Syntax: !joydir or ?joydir
Parameter: none or x, y, z, a
 and 0, 1, 2, -1, -2

Description: In addition to the '**joy**' instruction, this instruction is used to enable/disable single HDI axes (e.g. of a joystick, trackball, ErgoDrive etc.) and set their counting directions. The parameter functions are as follows:
0 = Disable the HDI axis (e.g. joystick deflection is ignored)
1 = Enable the HDI axis, no motor current reduction
2 = Enable the HDI axis, with current reduction support (default)
-1 = Same as 1, direction reversed
-2 = Same as 2, direction reversed

Remarks: Please make sure that the joystick function is globally enabled by the '**joy**' instruction.

When using a 4 axis controller with a 3 axis HDI device, the 3rd axis must be assigned to axis 3(=default setting), 4 or both (3 and 4) by enabling/disabling their joydir!

Response: HDI directions of the axes or specified axis

Examples:
!joydir -1 enable HDI X-axis, reversed, without current reduction.
!joydir z 0 disable HDI Z-axis.
?joydir query all axes for their HDI direction settings.
!joydir 2 2 0 2 set positive direction, allow current reduction, assign the joysticks 3rd axis to the controller A axis instead of Z.

16.3. joywindow (Joystick Window)

Syntax: !joywindow or ?joywindow
Parameter: 0 to 100

Description: This instruction sets the width of middle range, where joystick deflection has no effect to motor movement. Please note that there is only one value for all axes! This value should not be reduced, as this may result in slow unwanted moving of axes even when the joystick is apparently not deflected. Increasing the value will result in a loss of speed resolution.

Response: joywindow value

Examples:
?joywindow query for joystick window
!joywindow 14 set joystick window to 14

16.4. joyvel (Joystick Velocity)

Syntax: !joyvel or ?joyvel

Parameter: x, y, z or a
0.0001 to 100 [revolutions/s] or [mm/s] for dim = 9

Description: **This instruction is accessible in extmode 1 only!**
In extmode=1 this instruction must be used to set the joystick velocities. If so, the vel command has no influence to the joystick velocity. In normal mode (extmode=0) the joystick velocities are derived from the axis vel settings.

Response: Currently used joystick velocities

Examples:

```
!joyvel 12.5 20 0.4    Set joystick velocities for 3 axes
!joyspeed z 1          Set joystick velocities for z to 1 [rev/s], (e.g. dim=2)
                        or [mm/s] if dim=9
?joyvel x              Query X-axis for its joystick velocity
```

16.5. joyspeed (Joystick Speed Presets for BPZ Device)

Syntax: !joyspeed or ?joyspeed

Parameter: 1, 2 or 3 and
0.0001 to 100 [revolutions/s]

Description: Only used by a customer designed external device (called BPZ), this instruction sets the joystick speeds for the three speed buttons (Slow, Medium, Fast). Unit is in motor revolutions per second (like 'vel' instruction). While the velocity applies to all axes, each speed button has to be set individually:
1 = Slow Button speed, one parameter for all axes
2 = Medium Button speed, one parameter for all axes
3 = Fast Button speed, one parameter for all axes

Response: Speed currently assigned to the specified button

Examples:

```
?joyspeed 1            Query for "Slow" joystick button speed
!joyspeed 3 30         Set "fast" joystick button speed to 30 [revolutions/s]
```

16.6. keymode (Joystick Key Mode)

Syntax: !keymode or ?keymode

Parameter: 0, 1 or 2

Description: Assign keyspeed values to the Joystick buttons. The Joystick can be used with two different velocity settings slow/fast. In keymodes 1,2 'vel' or 'joyvel' instructions have no effect. In such case refer to '**keyspeed**', which will be used then. Please note that other special functions which require Joystick buttons (e.g. snapshot modes) should not be used at the same time as keymode.

A) If Joystick toggle mode ('**hdimode**') is selected:

--> F1 toggles XY between the keyspeed values

--> F4 toggles Z keyspeeds (available from firmware 1.56)

B) If '**hdimode**' is not set to toggle mode, the behavior is:

Pressing F1 selects the fast keyspeed values of X and Y axis, while F4 selects the slow keyspeed values of X and Y axis.

Pressing F2 selects the fast keyspeed value of the Z axis, while F3 selects the slow keyspeed value of the Z axis.

In case of joysticks with the optional '**z-wheel**', the wheel velocities can be selected by pressing F1, F4, or no key. Please refer to the '**zwtravel**' description.

Recommended Joystick settings are:

2/3 axis Joystick: hdimode 0 or 1 to either toggle velocities by pressing F2/F3, F1/F4 (0) or to toggle by pressing F1 (1).

Joystick with wheel: hdimode 0 to toggle XY by F2,F3 and the wheel with pressing none/F1/F2.

Parameter description:

0 = Normal key functions

1 = X/Y and Z Joystick velocity, initial value: slow [F4,F3]

2 = X/Y and Z Joystick velocity, initial value: fast [F2,F1]

Response: keymode as decimal number

Examples:

!keymode 1 slow preset keymode

?keymode => 0 (in case keymode is disabled)

16.7. keyspeed (Joystick Key Speed Presets)

Syntax: !keyspeed or ?keyspeed

Parameter: x, y, z or a
0.0000025 to 100 [mm/s]

Description: Two Joystick velocities can be set for each axis individually. The first parameter is the slow value and the second parameter is the fast. Unit is always mm/s, independent from 'dim'. In keymode 1 or 2 the X and Y values (slow/fast) are assigned to F4 and F1, while the Z values are assigned to F3/F2. Please also refer to 'keymode'.

Response: Two floating point values per axis (slow fast)
single axis : => [slow] [fast]
multiple axes: => [slow_x] [fast_x] [slow_y] [fast_y] ...

Examples:

```
?keyspeed x      => 1 10 (Query for X Joystick button velocities)
?keyspeed        => 1 10 1 10 0.1 1 (Query 3 axis controller)
!keyspeed z 0.1 1 (Set fast Joystick button speed to 0.1 and fast to 1 [mm/s])
!keyspeed 5 20 2 10 0.2 2 (Set 3 axes at once)
```

16.8. joycurve (Joystick Characteristic)

Syntax: !joycurve or ?joycurve

Parameter: x, y, z, or a
0, 1, 2

Description: The speed characteristic of Joystick deflection can be independently defined for each axis.

0 = Logarithmic (standard)
1 = Linear
2 = Quadratic

Response: Currently used characteristic

Examples: !joycurve 0 0 0 => set X,Y,Z axes to logarithmic
!joycurve z 1 => set Z axis to linear
?joycurve => query all active axes characteristics

16.9. key (Read HDI Device Key State)

Syntax: ?key or key

Parameter: none or key number 1, 2, 3, 4

Description: This instruction reads the state of up to 4 HDI device keys.

0 = key is currently released or not available
1 = key is currently pressed

Response: 1 or 4 Key states, each either 0 or 1

Examples: key => query all keys, returns 4 numbers, e.g. 0 0 0 0
key 1 => query only key 1 (e.g. F1 Joystick button)

16.10. keyl (Read HDI Device Latched Key State)

Syntax: ?keyl or !keyl

Parameter: none or key number 1, 2, 3, 4

Description: This instruction reads the latched state of up to 4 HDI device keys and clears their latched state. The latch state is cleared after reading.

0 = key is is/was released since last key or keyl instruction

1 = key is is/was pressed since last key or keyl instruction

Response: 1 or 4 Latched key states, each either 0 or 1

Examples: key => query all keys, returns 4 numbers, e.g. 0 0 0 0

key 1 => query only key 1 (e.g. F1 Joystick button)

!key 1 => clear latch state of key 1 (=0)

!key => clear latch state of all keys (=0)

16.11. hwfactor (Handwheel Transmission Factor)

Syntax: !hwfactor or ?hwfactor

Parameter: none or x, y, z, a
and -200.0 to 200.0

Description: The handwheel transmission factor represents the stage travel distance in millimeter per handwheel knob revolution, which is a floating point number between -200.0 and +200.0. Negative factors reverse the travel direction. Higher factors result in a more coarse resolution, as a typical handwheel provides about 100000 steps per revolution.

Response: Currently used handwheel factor(s)

Examples:

!hwfactor 14 14 => One knob revolution in X or Y results in 14mm axis travel

!hwfactor x 100 => One knob revolution in X results in 100mm travel

?hwfactor => Query all axes for their transmission factor

16.12. hwfactorb (Alternate Handwheel Factor)

Syntax: !hwfactorb or ?hwfactorb

Parameter: none or x, y, z, a
and -200.0 to 200.0

Description: Similar to '**hwfactor**', this instruction accesses the alternate parameter for the second travel distance per knob revolution. As available with the ErgoDrive. Negative factors reverse the travel direction.

Response: Currently used alternate handwheel factor(s)

Examples:

!hwfactorb 26.6 26.6 => One knob revolution in X or Y results in 26.6mm travel

?hwfactorb y => Query Y axis for its alternate transmission factor

16.13. hwfilter (Handwheel Noise Filter)

Syntax: !hwfilter or ?hwfilter
Parameter: 0 or 1

Description: This instruction sets or reads the handwheel noise filter state.

1 = Noise filter is active (recommended, default)
0 = Noise filter is deactivated (finer step resolution)

The filter can only be activated/deactivated for all axes. Disabling the filter may result in a finer resolution. But it also may result in some inaccuracy between automatic moves, as its signal noise will cause a permanent slight position jitter.

Response: Current state of handwheel filter

Examples:
!hwfilter 0 => No noise filter for handwheel, increased finer resolution
?hwfilter => Query hwfilter state

16.14. tbfactor (Handwheel Transmission Factor)

Syntax: !tbfactor or ?tbfactor
Parameter: none or x, y, z, a
and -200.0 to 200.0

Description: This instruction sets or reads the trackball transmission factor, which is a floating point number between -200.0 and +200.0. A sign change may be used to change direction.

Response: Currently used trackball factor(s)

Examples:
!tbfactor x 100 => X axis is 10 times more sensitive than the default setting
?tbfactor => Query all axes for their transmission factor

16.15. zwheel (Is Z-Wheel Available)

Syntax: ?zwheel or zwheel
Parameter: none

Description: The instruction zwheel returns:

0 = HDI device has no additional Z-Wheel
1 = HDI device has additional Z-Wheel

Response: 0 or 1

Example: ?zwheel => 0

16.16. zwtravel (Z-Wheel Travel per wheel revolution)

Syntax: !zwtravel or ?zwtravel
Parameter: 1, 2 or 3 and
- 50.0 to 50 [mm/revolution]

Description: Only used when a HDI device with Z-Wheel is connected to the controller. This instruction sets the Z travel distances for one revolution of the Z-Wheel knob.

In order to prevent mechanical damage, please make sure that the correct '**pitch**' and '**gear**' settings are made and '**save**'d. It is recommended to restart or '**reset**' the Tango controller after changing these parameters.

Please also check that '**secvel**' and '**vel**' or '**joyvel**' speed limitations do not prevent faster traveling when turning the Z-Wheel.

1 = Default (used when no HDI function key is pressed)
2 = Used while Joystick F4 button is pressed (suggested slow)
3 = used while Joystick F1 button is pressed (suggested fast)

Presets for travel distance are

1: 0.1 mm/rev
2: 0.01 mm/rev
3: 1.0 mm/rev

If necessary, the default travel may be set to zero (0) for security reasons. So the axis will move only when a key is pressed (F1, F4).

It is also possible to set negative values for direction change. While the '**joydir**' command changes the direction in general.

Response: Stage travel distance currently assigned to the specified function

Examples:

?zwtravel 1 Query for default Z-Wheel travel distance
!zwtravel 3 2.5 Set F1 "fast" Z-Wheel travel to 2.5 [mm/revolution]

16.17. tvrjoy (Pulse and Direction Joystick Functionality)

Syntax: !tvrjoy or ?tvrjoy
Parameter: 0, z, a

Description: This instruction enables and assigns the AUX-IO pulse and direction input to an axis for simple joystick functionality. The behavior is similar to the trackball, which is available as HDI device.
Important: This option must not be used for absolute positioning of axes by an external controller. Please use the tvr functionality for this applications.

0 = Disabled
z = Assigned to Z-axis
a = Assigned to A-axis

Response: Currently assigned axis

Examples:
!tvrjoy 0 Disable AUX-IO tvr joystick function
!tvrjoy z Assign AUX-IO tvr joystick function to Z-axis
?tvrjoy Query assigned axis

16.18. tvrjoyf (Pulse and Direction Joystick Factor)

Syntax: !tvrjoyf or ?tvrjoyf
Parameter: -200 to +200

Description: This instruction sets or reads the tvrjoy transmission factor, which is a floating point number between -200.0 and +200.0. A sign change may be used to change direction.

Response: Currently used tvr factor

Examples:
!tvrjoyf 100 Axis is 10 times more sensitive than the default setting
?tvrjoyf Query tvrjoy transmission factor

16.19. hdi (Read HDI ID)

Syntax: ?hdi or hdi
Parameter: none

Description: This instruction reads the ID number of the connected hdi device. A second number shows how good the hardware ID code matches the theoretical ID value [in %]. This value should be more than 30.

ID range = 0,1,2, ... 16 (=no device connected)
Match = 0 (poor) ... 100 (good)

Response: HDI ID number and the hardware coded ID match in percent.

Example of hdi response: 12 97 (hdi device 12, 97% match)

16.20. hdimode (HDI Mode Options)

Syntax: ?hdimode or !hdimode

Parameter: Set LSB or more bits at once: string of 0s and 1s,
or single bit with two numbers: 0 to 15 and 0 or 1

Description: This instruction gives access to extended HDI device options. Options may be set either by a string of levels (0s and 1s) or by bit number and logic state (on/off = 1/0). The string is LSB first (bit 0 is the first and leftmost).

Bit Function

0:	Toggle Mode for ErgoDrive (0=off, 1=on)
1:	Toggle Mode for Joystick (in KeyMode 1 or 2) 0=select KeySpeed velocity XY with F1+F4, Z with F2+F3 1=toggle KeySpeed velocity XY by just pressing F1 from firmware 1.56 button F4 toggles Z
2:	- reserved -
3:	- reserved -
4:	- reserved -
5:	- reserved -
6:	- reserved -
7:	- reserved -
8:	- reserved -
9:	- reserved -
10:	- reserved -
11:	- reserved -
12:	- reserved -
13:	- reserved -
14:	- reserved -
15:	- reserved -

Response: Single mode bit or all 16 mode bits as ASCII string

Examples:

```
!hdimode 100010  Set mode bits 0 and 4 to "on", bits 1,2,3,5 to "off". Bits
                  6...15 are left unchanged.
!hdimode 0 1      Set mode bit 0 to 1 (on) = ErgoDrive Toggle Mode selected
!hdimode 5 1      Set mode bit 5 to 1 (on)
?hdimode          query the current state of all mode bits (returns 16 digits)
?hdimode 0        query the current state of mode bit 0 (toggle mode)
```


17. Digital and Analogue I/O

The Tango provides several digital I/O, two analogue outputs (channel 0 and 1) and one analogue input. These are available on the optional auxillary I/O port. The analogue output channel 2 is reserved for special purpose. Furthermore the HDI Interface analogue inputs may be read as well, if no HDI-device is connected.

17.1. digin (Digital Input)

Syntax: ?digin or digin
Parameter: none or 0 to 15

Description: Only available with I/O extension board.
This instruction queries the logic state of one or all digital inputs. If no parameter is used all inputs are returned as a string of 16 characters, ASCII 0 or 1, LSB (channel #0) first.

Response: logic state of digital inputs

Examples:
?digin query all 16 digital inputs (response e.g. 0000000000000000)
?digin 8 query digital input 8 (response e.g. 1)

17.2. digout (Digital Output)

Syntax: !digout oder ?digout
Parameter: Set LSB or more bits at once: string of 0s and 1s,
 or single bit with two numbers: 0 to 15 and 0 or 1

Description: Only available with I/O extension board.
This instruction sets or reads back the logic level of the optional digital outputs.
Outputs may be set either by a string of levels (0s and 1s) or by channel number and signal level.
The string is LSB first (channel 0 is the leftmost).

Response: current output state

Examples:
!digout 11110000 The digital outputs 0,1,2,3 are set to logic ,1' and the outputs 4,5,6,7 are set to logic ,0'. Outputs 8...15 are left unchanged.
!digout 5 1 set digital output #5 to logic 1
?digout query the current state of all outputs
?digout 8 query the current state of output 8
!digout 7 0 set output 7 to 0

17.3. adigin (AUX-I/O Digital Input)

Syntax: ?adigin or adigin
Parameter: none or 0 to 3

Description: Available with the AUX-I/O connector.
This instruction queries the logic state of one or all digital inputs. If no parameter is used all inputs are returned as a string of 4 characters, ASCII 0 or 1, LSB first:

~~0 = Bit 0 = AUX-I/O Pin 1 (Takt In) may not be available!~~
1 = Bit 1 = AUX-I/O Pin 2 (V/R In)
2 = Bit 2 = AUX-I/O Pin 3 (Stop)
3 = Bit 3 = AUX-I/O Pin 4 (SnapShot2)

Response: logic state of digital inputs

Examples:
?adigin query all (4) AUX-I/O digital inputs (response e.g. 1111)
?adigin 3 query AUX-I/O digital input 3 ("SnapShot2", response e.g. 1)

17.4. adigout (AUX-I/O Digital Output)

Syntax: !adigout oder ?adigout
Parameter: Set LSB or more bits at once: string of 0s and 1s,
or single bit with two numbers: 0 to 3 and 0 or 1

Description: Available with the AUX-I/O connector.
This instruction sets or reads back the logic level of the AUX-I/O digital outputs.
Outputs may be set either by a string of levels (0s and 1s) or by channel number and signal level:

~~0 = BIT0 = AUX-I/O Pin 5 (TAKT_OUT) may not be available!~~
1 = BIT1 = AUX-I/O Pin 6 (VR_OUT)
2 = BIT2 = AUX-I/O Pin 7 (SHUTTER_OUT)
3 = BIT3 = AUX-I/O Pin 8 (TRIGGER_OUT)

The string is LSB first (channel 0 is the leftmost).

Response: current output state

Examples:
!adigout 1011 All digital outputs except output 1 are set to logic 1.
!adigout 10 Digital outputs 0 and 1 are set to logic 1 (BIT0) and 0 (BIT1),
outputs 2 and 3 are left unchanged.
!adigout 2 1 set digital output #2 to logic 1
?adigout query the current state of all outputs
?adigout 3 query the current state of output 3
!adigout 1 0 set output 1 to logic 0

17.5. anain (Analogue Input)

Syntax: ?anain
Parameter: c (c = channel)
0 to 15 (channel number)

Description: This instruction reads the current value of one analogue input channel. The range is decimal from 0 (=0V) to 1023 (=5V).

Channel No	Connector	Pin	Signal Name
0	HDI	1	Joystick X
1	HDI	2	Joystick Y
2	HDI	3	Joystick Z
3	HDI	4	
4	HDI	5	Speedpoti
5	HDI	6	
6	HDI	7	
7	HDI	8	
8	HDI	9	
9	HDI	10	HDI-ID
10	AUX-IO	9	ANAIN0
11	internal	-	U-HIP
12	internal	-	V-MOT
13	EXT	20	X-ID0
14	EXT	18	X-ID1 / temp
15	internal	-	REF (2.5V)

Calculating the internal motor voltage:

$U_{mot}[V] = (5 / 1023) * [anain\ c\ 12] * (55.7/4.7)$
More accurate:
 $U_{mot}[V] = (2.5 / [anain\ c\ 15]) * [anain\ c\ 12] * (55.7/4.7)$

Calculating the internal PSE voltage:

$U_{mot}[V] = (5 / 1023) * [anain\ c\ 11] * (14.7/4.7)$
More accurate:
 $U_{mot}[V] = (2.5 / [anain\ c\ 15]) * [anain\ c\ 11] * (14.7/4.7)$

Calculating the case temperature (if available):

$T[^{\circ}C] = (250 / [anain\ c\ 11]) * [anain\ c\ 14]$

Example:
?anain c 10 Query level of channel 10 (analogue input of AUX-IO connector)

17.6. anaout (Analogue Output)

Syntax: !anaout or ?anaout
Parameter: 0 to 100 in percent (100% = 10V)
c (c = single channel keyword)
0, 1 or 2 (single channel number)

Description: This instruction sets and reads the analog output signal levels in percent. There are two ways to access the values, with or without the 'c' keyword (see examples below). So it is possible to address a single channel by using the 'c' or channel 0 or all channels by directly writing the percent values. Fractional numbers may be used, too. 100% corresponds to 10 Volts.

Channel No	Connector	Pin	Signal Name
0	AUX-IO	10	ANOUT0
1	AUX-IO	11	ANOUT1
2	reserved	-	-

Response: Analogue output signal level in percent

Examples:
!anaout 100 50.1 Set channel 0 = 100% (10V) and channel 1 = 50.1% (5.01V)
!anaout 75 Set channel 0 = 75% (7.5V)
!anaout c 1 25.3 Set channel 1 to 25.3% (2.53V)
?anaout Query all channels for their output level
?anaout c 0 Query channel 0 output level

17.7. stoppol (Mode and Polarity of Stop Input Signal)

Syntax: !stoppol or ?stoppol
Parameter: 0 (= active low), 1 (= active high)

Response: Mode and polarity of AUX-I/O stop signal input:
0 = active low , Joystick not affected by stop signal
1 = active high, Joystick not affected by stop signal
2 = active low , all moves disabled as long as signal applied
3 = active high, all moves disabled as long as signal applied
4 = active low , all moves disabled until "!stop 0" command
5 = active high, all moves disabled until "!stop 0" command

Description: The stop input has an internal pull-up resistor to 5V. If you connect an NO (normal open) stop switch, you have to select any low active mode. For NC (normal close) switches please select high active.

Example:
!stoppol 1 => Set the polarity of the AUX-I/O stop input to active high.

17.8. stop (Release Stop Condition)

Syntax: !stop 0
Parameter: 0

Response: -
Description: Release stop condition in active stoppol modes 4 or 5.
Example: !stop 0

17.9. shutter (Shutter Out Signal of AUX-IO)

Syntax: !shutter or ?shutter

Parameter: 0 (= low), 1 (= high)

Response: Output level of shutter signal

Description: Manually set the AUX-IO shutter out signal to the desired TTL level.

Example:

!shutter 1 => Set the shutter out signal into TTL high state.

18. Encoder Instructions

To enable encoder functionality, first the encoder mask has to be set for the corresponding axes. After that a cal move will activate the encoders ('enc'=1), so they can be used. Manually setting the encoders 'enc' state to 1 is not recommended. This may cause trouble with when in closed loop mode, and in case of analog encoders the signal correction will be missing also.

18.1. encmask (Encoder Mask)

Syntax: !encmask or ?encmask

Parameter: x, y, z or a,
0 or 1

Response: Encoder enable mask

Description: The instruction reads or sets the encoder globally enable mask. It is necessary to unmask encoders (=1) as a first step in order to use them later. This instruction does not activate the encoders, it just globally enables the usage of them. Please note: Encoders get detected and finally used after a successful calibration command 'cal'. Here the signal correction data is measured, too.

Example:

```
!encmask 1 1 1    Globally enable encoders for X, Y and Z-axis
!encmask z 0      Globally disable encoder for Z-axis
?encmask           Query encoder mask state for all axes
```

18.2. enc (Encoder Active)

Syntax: !enc or ?enc

Parameter: x, y, z or a
0 or 1

Response: Encoder active state

Description: This instruction may be used to determine if a 'cal' command has activated the encoders or not. In order to activate them, the encmask has to be set to 1 first. It is not recommended to activate encoders manually by setting enc to 1. Refer to the 'encmask' description for further information.

0 = Encoder is inactive (not used)
1 = Encoder is activated

Example:

```
?enc              Globally enable encoders for X, Y and Z-axis
?enc x           Globally disable encoder for Z-axis
!enc x 1         Manually activate the X-axis encoder (not recommended!)
```

18.3. encperiod (Encoder Signal Period)

Syntax: !encperiod or ?encperiod

Parameter: x, y, z or a
0.0001 to 1.000 [mm]

Response: Encoder signal period(s)

Description: This command reads or sets the encoder signal period. The unit is always [mm].

Example:

!encperiod 0.5 0.5 0.001	Set encoder period for X and Y to 500µm, Z to 1µm
!encperiod y 0.020	Set encoder period of Y-axis to 20µm
?encperiod	Read encoder period of all axes
?encperiod x	Read encoder period of X-axis

18.4. encdir (Encoder Counting Direction)

Syntax: !encdir or !encdir

Parameter: x, y, z or a
0 or 1

Response: Encoder counting direction

Description: The encoder counting direction is set automatically by the calibration 'cal' move. It is not necessary to set the encoder direction and it must not be set when in closed loop. Only if the axis is not used for closed loop (e.g. only for relative distance measuring) the encdir can be set manually.

0 = Encoder counting direction default
1 = Encoder counting direction reversed

Example:

!encdir 1 1 1	Reverse encoder counting direction for all axes
!encdir x 1	Reverse encoder counting direction for X-axis only
?encdir	Query encoder counting direction of all axes
?encdir y	Query encoder counting direction of Y-axis only

18.5. encvel (Encoder Auto-Adjust Velocity)

Syntax: !encvel or !encvel

Parameter: x, y, z or a
0.01 ... 20.0

Response: Encoder detect and auto-calibration velocity

Description: The velocity for encoder auto-calibration can be set or read by this command. It is recommended to keep the default setting.

Example:

!encvel 0.5 0.5 0.5	Set encoder auto-adjust velocity for all axes
!encvel x 0.5	Set encoder auto-adjust velocity for X-axis only
?encvel	Query encoder auto-adjust velocity of all axes
?encvel y	Query encoder auto-adjust velocity of Y-axis only

18.6. encctl (Encoder has TTL Signal)

Syntax: !encctl or ?encctl
Parameter: x, y, z, a (or none)
0 or 1

Response: Currently selected encoder signal type(s)

Description: This command reads or writes the currently selected type of encoder signal processing.
If digital encoders (A/B-TTL/RS422) are used, the encoder interface has to be set in TTL mode. Else the TTL signal will be detected as invalid and not be used under some conditions.

0 = Encoder has analog sin/cos signals
1 = Encoder has digital quadrature A/B signals (e.g. RS422)

Example:

```
!encctl 0 0 1    Y and Y axis encoders are analog, Z is digital A/B encoder
!encctl z 1      Set Z encoder signal processing to digital
?encctl          Query all axes for their currently used signal type
?encctl x        Query X-axis for its currently used signal type
```

18.7. encref (Use Encoder Reference Signal)

Syntax: !encref or ?encref
Parameter: x, y, z, a (or none)
0 or 1

Response: Encoder reference signal used, not used

Description: The '**cal**' instruction will, after reaching to the lower limit switch, travel to the reference mark and set the axis position to zero.

0 = Encoder reference signal not used
1 = Encoder reference signal used for calibration

Example:

```
!encref 1 1 0    Utilize encoder reference signal for X and Y-axis
!encref x 1      Utilize encoder reference signal for Y-axis
?encref          Query Encoder reference signal utilization state of all axes
?encref x        Query Encoder reference signal utilization state of X-axis
```


18.8. encnas (Use Encoder NAS Error Signal)

Syntax: !encnas or ?encnas
Parameter: x, y, z or a
0 or 1

Response: Encoder NAS error signal used / not used

Description: Before enabling this functionality please make sure that the connected encoder provides a NAS error signal.
If enabled, a encoder NAS error also generates an internal 'err' error state. The NAS input signals an encoder error state by a TTL low level.

0 = NAS encoder input state is ignored (default)
1 = NAS encoder input signal is used for extended error detection

Example:
!encnas 1 1 0 Utilize encoder NAS signal for X and Y-axis
!encnas x 1 Utilize encoder NAS signal for Y-axis
?encnas Query Encoder NAS signal utilization state of all axes
?encnas x Query Encoder NAS signal utilization state of X-axis

18.9. encrefstatus (Encoder REF Signal State)

Syntax: ?encrefstatus or encrefstatus
Parameter: x, y, z, a (or none)

Response: Encoder reference signal state

Description: Returns the REF signal input state.

0 = REF signal is inactive
1 = REF signal is active (encoder is on a reference mark)

Example:
encnasstatus Query REF signal state for all axes
encnasstatus x Query REF signal state for X-axis only

18.10. encrefstatusl (Latched Encoder REF Signal State)

Syntax: ?encrefstatusl or encrefstatusl
Parameter: x, y, z, a (or none)

Response: Latched encoder reference signal state

Description: Returns the latched REF signal input state.
If the REF signal was active since last reading of the encrefstatusl, a 1 is returned.

0 = REF signal is/was inactive
1 = REF signal is/was active (encoder is/was on a reference mark)

Example:
encnasstatus Query latched REF signal state for all axes
encnasstatus x Query latched REF signal state for X-axis only

18.11. encnasstatus (Encoder NAS Error Signal State)

Syntax: ?encnasstatus or encnasstatus

Parameter: x, y, z, a (or none)

Response: Encoder NAS error signal state

Description: Returns the NAS error signal input state.

0 = NAS signal is inactive (encoder signals 'no error')

1 = NAS signal is active (error flag is set by encoder)

Example:

encnasstatus Query NAS signal (error) state for all axes

encnasstatus x Query NAS signal (error) state for X-axis only

18.12. encerr (Encoder Error State)

Syntax: !encerr or ?encerr

Parameter: x, y, z or a

0

Response: Encoder error state

Description: This command reads or resets the encoder error state. On error the encoder signal is invalid and a potentially running closed loop for the corresponding axis is switched off.

0 = No error, normal function

1 = Encoder error

Example:

!encerr 0 Reset encoder error

?encerr Read encoder error states of all axes

?encerr x Read encoder error states of X-axis only

18.13. encamp (Encoder Signal Amplitude)

Syntax: ?encamp

Parameter: x, y, z or a

Response: Encoder signal amplitude in percent as integer

Description: This command reads the encoder signal amplitude. 100% represents the maximum undistorted signal amplitude.

0 = No error, normal function

1 = Encoder error

Example:

?encamp Read all encoder amplitudes

?encamp x Read X encoder amplitude

18.14. encpos (Encoder Position)

Syntax: !encpos or ?encpos
Parameter: x, y, z, a or none
0 or 1

Response: Position output source

Description: If set to 1 and the encoder is activated (corresponding enc = 1), a '?pos' returns the encoder position, else the motor pos. For compatibility, sending a 0 or 1 without specifying an axis applies the setting for all axes. Refer to the 'pos' and 'enc' commands for further information.

0 = pos command reads the motor position (default)
1 = pos command reads the encoder position (if encoder active)

Example:
!encpos 1 from now on a 'pos' instruction returns the encoder position for all axes (if the corresponding encoders are activated)
~~!encpos 1 1 1~~ Invalid!
!encpos x 1 from now on a 'pos' command returns the encoder position for the X-axis (if the encoder is activated)
?encpos Use not recommended: Due to compatibility this instruction reads the "ored" position output source of all axes (returns just one 0 or 1)
?encpos x Read position output source of X-axis

18.15. hwcount (Hardware Counter)

Syntax: ?hwcount or hwcount
Parameter: x, y, z or a

Response: Hardware counter reading(s)

Description: Hwcount returns the position(s) of the independent TTL encoder counter. It is a digital counter that counts the signal slopes (4 per period) and does not provide signal interpolation. So one signal period corresponds to a counter reading of 4. See also the 'clearhwcount' command.

Example:
hwcount Returns the counter readings of all axes
hwcount x Returns the counter readings of X-axis only

18.16. clearhwcount (Clear Hardware Counter)

Syntax: !clearhwcount or clearhwcount
Parameter: x, y, z or a

Response: Reset hardware counter reading(s)

Description: This command resets the hardware counter(s) to zero.

Example:
clearhwcount Reset hwcount position of all axes to zero
clearhwcount x Reset hwcount position of X-axis to zero

19. MR Encoder Instructions

19.1. mra (MR Amplitude Correction Factor)

Syntax: !mra or ?mra
Parameter: x, y, z or a
0.8 to 1.2

Response: Currently used correction factor(s)

Description: This command reads or sets the cosine amplification correction factor of the analogue encoder signal (here: sin/cos amplitude ratio).
This factor is calculated automatically on each calibration move 'cal' and should not be changed. If the axis is manually controlled and only used for relative measurement, so that no 'cal' is possible, the user may determine the ratio itself and then write it into mra for more accurate results. Please also refer to the mro command.

Example:
?mra Read MR signal correction factor of all axes
?mra x Read MR signal correction factor of X-axis only
!mra x 1.0095 Amplify the X cosine signal by *1.0095 compared to the sine

19.2. mro (MR Offset Correction Value)

Syntax: !mro or ?mro
Parameter: x, y, z or a
-2048 to +2048

Response: Currently used correction value(s)

Description: This command reads or sets the sine and/or cosine offset compensation value as 16bit signed digits.
This factor is calculated automatically on each calibration move 'cal' and should not be changed. If the axis is manually controlled and only used for relative measurement, so that no 'cal' is possible, the user may determine the offset itself and then write it into mro for more accurate results. Please also refer to the mra command.

Example:
?mro Read MR signal offset value sine and cosine for all axes
?mro x Read MR signal offset value sine and cosine for X-axis only
!mro 48 -100 0 0 0 0 0 Set X offset to sin=48digit, cos=-100digit, Y, Z = 0
!mro y 16 -28 Set Y offset to sin=16digit, cos=-28digit
!mro y 16 Set only sine offset of Y encoder

19.3. mrp (MR Signal Peak-To-Peak Measuring Result)

Syntax: !mrp or ?mrp
Parameter: x, y, z or a
-2048 to +2048

Response: [sine max] [sine min] [cosine max] [cosine min] result(s)

Description: This command reads or sets the sine and/or cosine peak values, measured since they were reset the last time.
It is just a measurement and has no effect to the signal processing itself. The returned values are signed 16bit digits.

Example:

?mrp x Returns [x_sin max] [x_sin min] [x_cos max] [x_cos min]
?mrp Returns the above, but for all axes (up to 16 values)
!mrp x 0 0 0 0 Reset the peak-to-peak measurement for x
!mrp x 0 0 Reset only the X sine min, max values
!mrp 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 Reset measurement for all 4 axes

19.4. mrt (MR Signal Level)

Syntax: ?mrt
Parameter: x, y, z or a
1 to 32767

Response: [sine] [cosine] result(s)

Description: This command reads the corrected sine and cosine A/D converter results as signed 16bit digits.
If there is no count parameter transmitted, the command returns 10 measurement result lines per default.

Example:

?mrt x 1 Returns one line with [x_sin] [x_cos] signal digits
?mrt 1 Returns the above, but for all axes (up to 8 values):
[x_s] [x_c] [y_s] [y_c] [z_s] [z_c] if 3 axes are configured

?mrt y 2 Returns two lines with [y_sin] [y_cos] signal digits
?mrt Returns 10 lines with all axes (up to 8 values per line)
?mrt x Returns one line with [x_sin] [x_cos] signal digits

20. Closed Loop Instructions

The closed loop control positions the stage to the scale position. So the inaccuracy of the drive is compensated. The closed loop control circuit is activated by the "ctr" command. But first, in order to activate the encoders they have to be unmasked "encmask" and a calibration move "cal" has to be executed (which activates "enc" and detects the counting direction).

20.1. ctr (Control Enable)

Syntax: !ctr or ?ctr

Parameter: x, y, z or a
0,1,2,3,4

Description: This instruction activates the closed loop circuit.
0 = Closed Loop OFF
1 = Closed Loop Auto OFF each time position is reached
2 = Closed Loop always ON (currently not supported!)
3 = (currently not supported!)
4 = (currently not supported!)

Response: Closed loop state(s)

Examples:

!ctr 0 0 0 0	Closed loop off for all axes
!ctr x 1	Closed loop for X-axis switches off after position reached
?ctr	Query closed loop states of all axes
?ctr x	Query closed loop state of X axis

20.2. ctrf (Control Factor)

Syntax: !ctrf or ?ctrf
Parameter: x, y, z or a
 0.1 to 25.0

Description: This instruction reads or sets the closed loop factor.
 Higher values result in more stiffness and faster settlement.
 Above a critical value this may lead to oscillation.
 The default factor of 2.0 mostly results in a good behavior.
 Hint: Using the ctrff instruction instead offers more options.

Response: Closed loop factors as integers (rounded)

Examples:
!ctrf 2 2 2 Set closed loop factor to 2 for all axes
!ctrf x 3 Set closed loop factor for X axis to 3
?ctrf Query closed loop factors of all axes (integers)
?ctrf y Query closed loop factor of Y axis (integer)

20.3. ctrff (Extended Control Factor)

Syntax: !ctrff or ?ctrff
Parameter: x, y, z or a
 0.1 to 25.0
 0.1 to 25.0

Description: This instruction reads or sets 2 closed loop factors per axis.
 Higher values result in more stiffness and faster settlement.
 Above a critical value this may lead to oscillation.
 The default factor of 2.0 mostly results in a good behavior.
 Important: Can only be set per axis (with x,y,z,a parameter)!

Parameter1: Is used for regulation while axis is moving
Parameter2: Is used for regulation when axis is stopped

Parameter 2 can be set to higher values than Parameter1
to achieve smoother axis travel while still having the
stiffness and faster settling times at the end of a move.
(E.g.: "!ctrff x 2 4".)

Response: Closed loop factors (2 per axis)

Examples:
!ctrff 2 2 2 2 Not supported!
!ctrff x 2 4 Set closed loop factors for X axis 2(moving) and 4(reached)
?ctrff Query closed loop factors of all axes (2 parameters per axis)
?ctrff y Query closed loop factors of Y axis (2 parameters)

20.4. ctrc (Control Call)

Syntax: !ctrc or ?ctrc
Parameter: 1 to 100 [ms]

Description: This instruction reads or sets the controller call interval. Unit is milliseconds. Only one parameter for all axes. The default interval of 5 [ms] in most cases leads to the best results. Values of less than 3 [ms] are not recommended.

Response: Closed loop control call interval in milliseconds.

Examples:
!ctrc 5 Closed loop control is executed every 5 milliseconds
?ctrc Query closed loop call interval

20.5. ctrd (Control Target Window Delay)

Syntax: !ctrd or ?ctrd
Parameter: 0 to 250 [ms]

Description: This instruction reads or sets the control delay. This is the time the closed loop has to stay inside the target window "twi" until a position reached state is set. Unit is milliseconds. Only one parameter for all axes. Please also refer to the ctrt timeout.

Response: Closed loop control delay in milliseconds.

Examples:
!ctrd 100 Closed loop controller must be in target window for 100 ms
?ctrd Query closed loop target window delay

20.6. ctrt (Control Timeout)

Syntax: !ctrt or ?ctrt
Parameter: 0 to 10000 [ms]

Description: This instruction reads or sets the control timeout. It specifies the time the closed loop tries to reach the desired encoder position. If the ctrd condition could not be fulfilled within this ctrt time, it will be aborted. If ctrd/ctrt is used, the ctrt timeout must be set to a value which is higher than the ctrd, typically 1+ seconds. Unit is milliseconds. Only one parameter for all axes.

Response: Closed loop control timeout in milliseconds.

Examples:
!ctrt 1000 Closed loop tries to reach the end position for 1 second
?ctrt Query closed loop timeout

20.7. twi (Target Window)

Syntax: !twi or ?twi

Parameter: x, y, z, or a
[value corresponding 0.0001 to 1 mm] in dim units

Description: This instruction reads or sets the closed loop control target window width. While increasing this value leads to position variance, setting a too narrow window may result in oscillation and closed loop timeouts (higher ctrd,ctrtr values necessary).
The unit depends on 'dim'.

Response: Closed loop target window.

Examples:

```
!twi 0.001 0.001  Closed loop target window is 1µm (if dim=2) for X and Y-axis
!twi y 0.005      Closed loop target window is 5µm (if dim=2) for Y-axis
?twi              Query all axes for their target window
?twi z            Query Z-axis for its target window
```

20.8. ctrstatus (Control Status)

Syntax: ?ctrstatus

Parameter: x, y, z, a or none
optional parameter: 1

Description: This instruction returns the one of two possible closed loop states, depending on if the optional Parameter 1 is used.

Just sending the ?ctrstatus request with or without specifying an axis returns the internal **ctr** state which is set when the closed loop gets enabled by the controller, e.g. 0, 1 or 2.
Hint: the internal ctr mode is set to the requested **ctr** mode when the closed loop is activated. The state may be zero when the closed loop has not been activated yet by cal, calmode, encmask or ctr=0.

Sending the ?ctrstatus 1 request with or without specifying an axis returns if the closed loop momentary is active.

0 = Closed loop momentary not active

1 = Closed loop momentary active

Hint: The state may be not active when the internal cal mode is zero or e.g. during calibration, invalid encoder signal, low current **reduction** (≤ 30%) etc.

Response: Closed loop state.

Examples:

```
?ctrstatus      Returns the internally running ctr mode of all axes
?ctrstatus y    Returns the internally running ctr mode of the Y-axis
```

```
?ctrstatus 1    Returns the Closed Loop active state of all axes
?ctrstatus x 1  Returns the Closed Loop active state of the X-axis, e.g. "1"
```

20.9. ctrdiff (Control Position Difference)

Syntax: `?ctrdiff`

Parameter: `x, y, z, a` or none

Description: This instruction returns the momentarily measured closed loop position difference between the motor and encoder position (`mot.pos - enc.pos`). The unit depends on **dim** settings.

Response: Momentary position difference.

Examples:

`?ctrdiff` Returns the position difference of all axes

`?ctrdiff y` Returns the position difference of the Y-axis e.g. "0.0015"

21. Trigger Signal Configuration

These commands read or modify the parameters for the trigger output signal. It may be used for synchronization of an external device like e.g. a video camera. The trigger output signal is available on the optional AUX-I/O connector. Access permission to the trigger functionality has to be enabled by factory. Before enabling the trigger function (by "!trig 1"), please make sure that all trigger settings have been made.

```

Example1:  !trig 0[CR]           Disable trigger
           !trigm 0[CR]         Choose trigger mode 0
           !triga x[CR]         Choose X axis as trigger source
           !trigd 0.100[CR]      Set trigger distance to 100µm (if dim = 2)
           !trigs 400[CR]        Set trigger pulse width to 0.4ms
           !trig 1[CR]          Enable trigger, set start position

Example2:  !trig 0[CR]           Disable trigger
           !trigs 120[CR]        Set trigger pulse width to 120µs
           !trigf 2500[CR]       Set pulse frequency to 2.5kHz
           !trigm 100[CR]        Choose trigger mode 100 (periodic signal)
           !trig 1[CR]          Enable trigger, set start position

```

Optional the "trigcount 0" command may be executed to reset the event counter.

21.1. trig (Trigger)

```

Syntax:      !trig or ?trig
Parameter:   0 (= disabled) or 1 (= enabled)

Description:  This instruction enables or disables the trigger circuit.
              "!trig 1" also sets the trigger start position.

              0 = Trigger function globally disabled
              1 = Trigger function globally enabled

Response:    0 or 1

Examples:
!trig 0      Disable trigger circuit
?trig        Query for current state of trigger circuit

```

21.2. triga (Trigger Axis)

```

Syntax:      !triga or ?triga
Parameter:   x, y, z or a

Description:  This instruction selects the axis on which to trigger

Response:    x, y, z or a


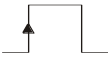








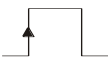











Examples:
!triga x     Select X-axis as trigger source
!triga y     Select Y-axis as trigger source
?triga       Query current trigger axis

```



21.3. trigm (Trigger Mode)

Syntax: !trigm or ?trigm
Parameter: 0 to 11, 100 to 105

Description: This instruction selects the required trigger mode.

Trigger Mode	Trigger Generation	Trigger Signal	Remarks
0		High active 	First pulse when move starts
1		High active 	First pulse when move starts
2		High active 	First pulse when move starts
3	-- See Mode 0 --	Low active 	Same as 0, signal inverted
4	-- See Mode 1 --	Low active 	Same as 1, signal inverted
5	-- See Mode 2 --	Low active 	Same as 2, signal inverted
6		High active 	Triggers shifted by trigm/2
7		High active 	Triggers shifted by trigm/2
8		High active 	Triggers shifted by trigm/2
9	-- See Mode 6 --	Low active 	Same as 6, signal inverted
10	-- See Mode 7 --	Low active 	Same as 7, signal inverted
11	-- See Mode 8 --	Low active 	Same as 8, signal inverted
100	Generates periodic trigger signals with the frequency chosen by the "trigf" parameter.	High active 	Does not depend on position
101	-- See Mode 100 --	Low active 	Same as 100, signal inverted
102	Allows manual forced trigger signals by the "trigger" command.	High active 	Does not depend on position or time
103	-- See Mode 102 --	Low active 	Same as 102, signal inverted



104	Trigger on position reached of specified axis	High active 	Comes with @@@ status response
105	-- See Mode 104 --	Low active 	Same as 102, signal inverted

Response: Trigger mode as integer: 0 to 11, 100 to 105

Examples: `!trigm 3` Set Trigger Mode 3
`?trigm` Query current trigger mode

21.4. trigger (Force Trigger Signal)

Syntax: `!trigger` or `trigger`
Parameter: None

Description: This instruction generates a trigger output pulse. It is available in trigger modes 102 and 103. The pulse width is depending on "trigs" value.

Response: None

Examples:
`trigger` Force trigger pulse now
`!trigger` The same as above

21.5. trigs (Trigger Signal Length)

Syntax: `!trigs` or `?trigs`
Parameter: 0 to 2621400 [μ s] (as multiples of 40 [μ s])

Description: This instruction is used to adjust the trigger pulse width from 40 microseconds to 2.6214 seconds in increments of 40. (0 = shortest trigger signal width)
If the parameter is not a multiple of 40 it will be rounded down to the next lower multiple (e.g. 100 --> 80). When read back, the corrected value is returned (here: 80).

Response: 0 to 2621400 (μ s), as multiple of 40

Examples:
`!trigs 40` Set Trigger pulse width to 40 μ s
`!trigs 2500000` Set Trigger pulse width to 2.5 s
`?trigs` Query current trigger pulse width

21.6. trigd (Trigger Distance)

Syntax: `!trigd` or `?trigd`
Parameter: >0 to 5000000 (unit depends on dim of the selected axis)

Description: This instruction selects the required trigger distance. After passing an interval of trigd with the selected axis, a trigger signal is generated.

Response: Trigger distance

Examples:
`!trigd 3` Set trigger distance to 3mm (if dim of selected axis is 2)
`!trigd 0.010` Set trigger distance to 10 μ m (if dim of selected axis is 2)
`?trigd` Query current trigger distance



21.7. trigf (Trigger Frequency)

Syntax: !trigf or ?trigf
Parameter: 0.01 to 12500

Description: This instruction generates periodic trigger pulses at the desired frequency. It is available at trigger mode 100. Frequency resolution is 1/40µs.

Response: Trigger frequency

Examples:
!trigf 2500 Generate periodic trigger pulses with 2.5kHz (each 0.4ms)
?trigf Query current trigger frequency

21.8. trigcount (Trigger Counter)

Syntax: !trigcount or ?trigcount
Parameter: 0 to 2147483647

Description: This instruction reads or manipulates the counted trigger events.

Response: Number of executed triggers

Examples:
?trigcount Query current trigger count
!trigcount 0 Clear trigger counter

22. Snapshot Signal Configuration

These commands read or modify the parameters for the snapshot input signal, which may be generated from an external device for synchronization purpose. The snapshot input signal is in the moment only available on Joystick button "F2". If snapshot is enabled, each snapshot event will store the current axis positions in an array. Access permission to the snapshot functionality has to be enabled by factory. Please globally enable the snapshot function first ("sns 1") after all snapshot settings have been made.

Example: Three snapshot positions are captured

sns = 3
snsa =

Element Index	Position X	Position Y	Position Z	Position A
1	1.0000	1.2345	1.2345	0
2	2.0000	1.2345	1.2345	0
3	3.0000	1.2345	1.2345	0
4	invalid	invalid	invalid	invalid
5	invalid	invalid	invalid	invalid

... ..

200	invalid	invalid	invalid	invalid
-----	---------	---------	---------	---------

In **sns = 0** the next snapshot will add a new row at index 4.

Same does the **!sns** command.

?sns will return positions of row 3.

?snsa x 2 will return 2.0000

22.1. sns (Snapshot)

Syntax: !sns or ?sns
Parameter: 0 (=disable) or 1 (=enable)

Description: This instruction enables or disables the snapshot input.

Response: Snapshot state

Examples:

!sns 0 Disable snapshot
!sns 1 Enable snapshot
?sns Query state of snapshot circuit

22.2. snsl (Snapshot Level / Polarity)

Syntax: !snsl or ?snsl
Parameter: 0 (=active low) or 1 (active high)

Description: This instruction sets the required snapshot signal polarity.

Response: Currently used snapshot polarity

Examples:

!snsl 0 Set snapshot input to active low
!snsl 1 Set snapshot input to active high
?snsl Query current snapshot input polarity

22.3. snsf (Snapshot Filter)

Syntax: !snsf or ?snsf
Parameter: 0 to 100 [ms]

Description: This instruction reads or modifies the snapshot filter time, which is used to suppress glitches or spikes on noisy signals.

Response: Currently used snapshot filter time

Examples:
!snsf 0 Disable input filter
!snsf 10 Set snapshot filter time to 10 ms
?snsf Query snapshot filter time

22.4. snsnm (Snapshot Mode)

Syntax: !snsnm or ?snsnm
Parameter: 0, 1, 2, 3, 4 or 5

Description: This instruction reads or sets the snapshot mode (default=0).

0 = Capture positions with Joystick key "F2"
1 = Automatic mode: Move to Positions with Joystick key "F2"
2 = Extended move:
 F1: Step/move through position list forward (pointer+1)
 (wraps around at the last element)
 F2: Step/move through position list backward (pointer-1)
 (wraps around at the first element)
 F3: Move to start of list (first element)
 F4: Moves to "prehome" position with "vel",
 then to "home" position with "secvel"
3 = Dissection mode
4 = Like mode 0, AUX-I/O SnapShot input is used instead of F2
5 = Like mode 1, AUX-I/O SnapShot input is used instead of F2

Remarks: Position capture and moves are always executed on all active axes.

Response: Currently selected snapshot mode

Examples:
!snsnm 0 Set snapshot mode to capture
!snsnm 1 Set snapshot mode to move
!snsnm 2 Set snapshot mode to extended move
?snsnm Query current snapshot mode

22.5. snsc (Snapshot Counter)

Syntax: !snsc or ?snsc
Parameter: --

Description: This instruction reads the snapshot counter, which shows the counted snapshots (= snapshot array entries). This instruction may also be used to reset the counter to zero.

Response: Current snapshot array entries (= number of snapshot events)

Example:



?snsc	Query the number of detected snapshots.
!snsc	Clear snapshot counter

22.6. snsp (Snapshot Position)

Syntax: !snsp or ?snsp

Parameter: x, y, z or a

Description: This instruction reads or writes the snapshot position. Writing positions appends them to the current position array. Reading positions returns the last captured position (last array element).

Remark: The position data unit depends on selected dimension 'dim'.

Response: Snapshot position value(s)

Examples:

```
!snsp 100 200    Append snapshot position for X and Y
!snsp 10 20 30   Append snapshot position for X, Y and Z axis
!snsp y 2000     Append snapshot position to Y = 2000
?snsp           Query all axes for their last captured snapshot positions
?snsp z         Query Z axis for its last captured snapshot position
```

22.7. snsas (Snapshot Array)

Syntax: !snsas or ?snsas

Parameter: x, y, z or a
and entry index from 1 to 200

Description: This instruction reads or writes to the snapshot position array, which may contain up to 200 elements. For reading, a valid element index may have a value of 1 to maximum the current snapshot counter value 'snsc'. For writing an index of snsc+1 may be used to append a position element to the array (snsc then gets updated by +1).

Remark: The position data unit depends on selected dimension 'dim'.

Response: Snapshot array position(s)

Examples:

```
?snsas 1         Query 1st snapshot entry for all axes positions
?snsas 33        Query 33rd snapshot entry for all axes positions
?snsas z 99      Query 99th snapshot entry for Z-axis position
?snsas x 199     Query 199th snapshot entry for X-axis position
!snsas 0         Clear the entire snapshot array
!snsas x 1 20.5   Set X position of first element to 20.5 (e.g. mm if dim=2)
!snsas 2 10 10 10 10 Set all axis positions of second array entry to 10
```

22.8. snse (Snapshot Event)

Syntax: !snse or snse

Parameter: 1,2,3 or 4

Description: The Snapshot event instruction can be used to execute the Snapshot functions via the communication interface. Instead of pressing a Joystick button or using the AUX-I/O signal, these snse parameters can be sent by software:

1 = Function normally executed by pressing Joystick F1 key

2 = Function normally executed by pressing Joystick F2 key
or using the AUX-I/O SnapShot input

3 = Function normally executed by pressing Joystick F3 key

4 = Function normally executed by pressing Joystick F4 key

Remark: Behavior is the same as with the function keys. It depends on the snapshot mode settings and only works when Snapshot is enabled.

Response: -

Example:
snse 2 Execute F2 Snapshot event (e.g. store current position in the snapshot array and snapshot position)

22.9. prehome (Snapshot PreHome Position)

Syntax: !prehome or ?prehome

Parameter: x, y, z or a

Description: This instruction sets the prehome position used by the snapshot extended move. The unit of the input position depends on instruction dim.
See "snsm" 2 for more details.

Response: Position value(s)

Examples:

!prehome x 10.2 Set prehome position X-value to 10.2 (e.g. [mm] when dim=2)

!prehome 10 0 20 Set prehome position X,Y,Z

?prehome x Read currently used prehome X-position

?prehome Read currently used prehome positions of all axes

22.10. home (Snapshot Home Position)

Syntax: !home or ?home

Parameter: x, y, z or a

Description: This instruction sets the home position used by the snapshot extended move. The unit of the input position depends on instruction dim.
See "snsm" 2 for more details.

Response: Position value(s)

Examples:

!home x 10.2 Set home position X-value to 10.2 (e.g. [mm] when dim=2)

!home 10 0 20 Set home position X,Y,Z

?home x Read currently used home X-position

?home Read currently used home positions of all axes

23. Document Revision History

No.	Revision	Date	Changes	Remarks
01	A	03. July 2007	New layout, improved and corrected descriptions, added new instructions, re-sorted instructions	Based on Tango firmware revision 1.26
02	B	09. July 2007	Added new instructions	Based on Tango firmware revision 1.26
03	prelim. C	27. July 2007	twi example corrected	Based on Tango firmware revision 1.26
04	C	03. Sept 2007	Added snapshot functions	Based on Tango firmware revision 1.26
05	D	28. Feb. 2008	Added some new instructions of firmware 1.31 and 1.32, Bugfixes in examples and descriptions.	Based on Tango firmware revision 1.32
06	prelim E	17. Jun. 2008	Added new instructions of firmware 1.34	Based on Tango firmware revision 1.34
07	E	07. July 2008	Added new instructions of firmware 1.35	Based on Tango firmware revision 1.35
08	prelim F	23. July 2008	Added encamp instruction	Based on Tango firmware revision 1.35
09	G	14. Aug. 2008	Added instructions keymode, keyspeed, extended help instruction, improved some comments	Based on Tango firmware revision 1.37
10	prelim H	29. Aug. 2008	Added Z-Wheel and extended version instructions	Based on preliminary Tango firmware revision 1.374
11	prelim H	14. Oct. 2008	added backlash instruction, extended lockstate bits	Based on preliminary Tango firmware revision 1.394
12	prelim H	07. Jan. 2009	Improved description of calmode behavior	
13	prelim H	06. Feb. 2009	zwtravel description corrected	
14	H	13. Feb. 2009	Documentation Rev. H released	Based on Tango firmware revision 1.40
12	prelim I	05. May 2009	corrected ?swin- and improved ?readsw command-description, added new MW logo	
13	I	27. May 2009	updated ipreter command description	
14	prelim J	09. July 2009	iver command description	
15	J	26. Aug. 2009	Corrected description of encnasstatus, encrefstatus, encdir. Extended snsm parameter, accel parameter. Improved description of HDI instructions, vel. Added new instructions: go, adigout, adigin, snse, hwfactorb, encrefstatusl, clearpos, maxpos, stout, accelfunc, hdimode, ctrstatus, ctrdiff.	Based on Tango firmware revision 1.46
16	prelim K	31. Aug. 2009	ipreter 1,2 extended up to 5.	Based on Tango firmware revision 1.46
17	prelim L	03. Mar. 2010	description corrected: encpos, refdir, calrefspeed, go resolution example corrected	
18	prelim L	26. Aug. 2010	Trigger modes 104,105 added, Document Title changed from "MST Dokument" to "Tango Instruction Set"	Based on Tango firmware revision 1.515



19	prelim L	05. Nov. 2010	Improved "hdimode" description, toggle mode and keymode	
20	prelim L	01. Dec. 2010	Updated "?version" description for new controllers	
21	prelim L	27. July 2011	Ipreader options changed	
	L	27. July 2011	Changed description for encctl and encrnf instructions Revised Introduction and HDI chapters	Taken from a newer prelim. Document beta version Information based on Tango Firmware 1.52