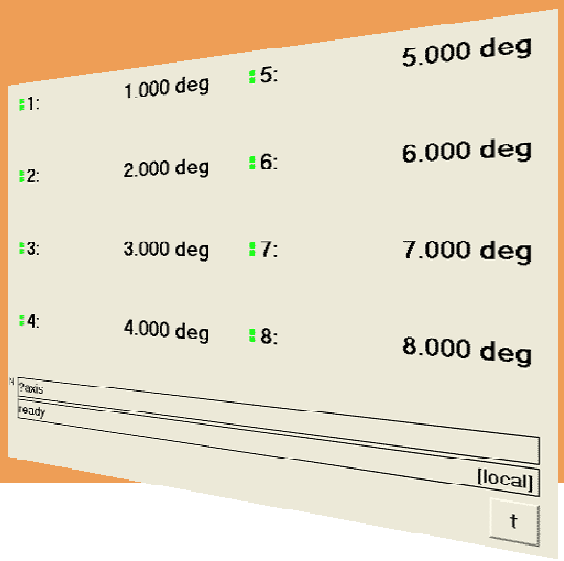
documentation

ppelectronic



motor controller

smc

command reference
smc 1.1.165
february 2012

pp-electronic gmbh
bahnhofstraße 41
d-83253 rimsting
fon +49.8051.96399.0
fax +49.8051.96399.29
www.pp-electronic.de
info@pp-electronic.de

content

general notes

the online documentation of the smc is available here:

http://smc.pp-electronic.de

it contains information about the latest hard- and software changes, updates and other useful information. the access to this site is password-protected: username '***user***', password '***smc***'.

the terms 'command' and 'command line' always denote a single ASCII character string which has to be transferred to the controller. a command must be terminated by <CR> or <CR>+<LF> (<CR>: carriage return 13_h; <LF>: line feed 10_h).

most commands require additional parameters. some are obligatory, a fact which is indicated by *square brackets* [<parameter>], some are optional, which is indicated by *braces* {<Parameter>}.

we distinguish four command categories:

a. configuration commands, which are used to configure the control parameters for the connected positioning hardware.

b. direct commands, to trigger immediate execution of a number of positioning and control functions.

c. query commands, to retriev data or status information from the controller.

d. program commands, which serve to create programs to execute complex positioning-, control- and data collection tasks.

commands of categories a. to c. consist of a single character string which will usually be executed by the controller immediately after arrival. these command will not be stored in the controllers memory. during the execution of a program, only a subset of these commands is available.

the term 'program line' denotes a set of command lines of the category 'program commands'. a program line usually consists of several command lines, followed by the command line 'nl' which indicated the end of a program line.

program lines will not be executed immediately after arrival. they will be stored in the controllers memory instead. the execution of a program can be triggered later on by transmitting a corresponding start command. programs are kept in memory until they are erased, overwritten or the controller is switched off.

some commands require the presence of certain hardware (see manual 'hardware reference'), which is available at option. if one of these commands is transferred to the controller though the necessary hardware is not installed, the command will either be ignored or an error message will be generated.

configuration commands

the controller accepts configuration commands only when it is idle, i. e. not busy with the execution of positioning- or control tasks.

configuration changes apply immediately after command transfer and remain active until the controller is switched off or reset. use command **update** in order to saves configuration changes and make settings permanent.

command list:

acc[axis]:[value]	acceleration ramp
alias[axis]:[text]	axis alias
bln[axis]:[dist]	set backlash loop distance for [-] referencing
blp[axis]:[dist]	set backlash loop distance for [+] referencing
conf|def[0|1|2]	hardware definition
dec[axis]:[value]	deceleration ramp
ecl[axis]:[0|1]	enable closed loop positioning (encoder)
ect[axis]:[value]	define encoder counter type
edev[axis]:[value]	closed loop positioning target window
edir[axis]:[0|1]	ecoder rotation sense
emips[axis]:[-1…255]	enable motion depending on input port status
eres[axis]:[value]	encoder resolution
esh[axis]:[value]	enabel encoder position display
esm[axis]:[1|2|4]	encoder signal multiplication factor
est[axis]:[value]	encoder signal type
ffast|vfast[axis]:[value]	manual positioning speed
fn[axis]:[text]	display additional axis label
frun|vrun[axis]:[value]	manual start-stop speed
gden|gn[axis]:[value]	gear reduction factor denominator
gnum|gz[axis]:[value]	gear reduction factor numerator
hsdm[axis]:[0|1]	slow-down mode of homing procedure
macc[axis]:[value]	acceleration ramp for manual positioning
mdec[axis]:[value]	deceleration ramp for manual positioning
mdir|mdl[0|1]	motor rotation sense
mpr[axis]:[value]	max. number of positioning retries
pqrf[0|1]	position query response format
prst[axis]:[interval]	retry delay for encoder closed-loop operation
prt[axis]:[value]	max. deviation for positioning retries
rofs|nofs[axis]:[value]	reference position offset
sdm[axis]:[status]	**s**low-down signal mode for homing procedure
unit[axis]:[text]	position display unit
update	save configuration parameters

acc

description

configuration of the acceleration ramp of an axis for subsequent **move** and **goto** positioning commands.

syntax

acc[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	1…1000 Hz/ms

usage

during positioning/program execution: no

related commands

dec

example

```
acc1:10
```

alias

description

the stated string will be shown on the display of the controller instad of the axis number.

syntax

alias[axis]:[string]

arguments

[axis]	1 … number of installed axes
[string]	any string, max. 3 characters

usage

during positioning/program execution: no

related commands

none

example

```
alias1:X
alias2:Y
```

bln

description

configuration of the backlash loop distance for negative direction of movement. used during execution of commands 'ref' and 'org'.
after setting the reference position, the controller moves the motor back and forth for the given distance starting into negative direction in order to remove expected drive backlash.

syntax

bln[axis]:[distance]

arguments

[axis]	1 … number of installed axes
[distance]	-1.0 … 1.0

usage

during positioning/program execution: no

related commands

blp, org, ref

example

```
bln1:0.2
```

blp

description

configuration of the backlash loop distance for positive direction of movement. used during execution of commands 'ref' and 'org'.
after setting the reference position, the controller moves the motor back and forth for the given distance starting into positive direction in order to remove expected drive backlash.

syntax

blp[axis]:[distance]

arguments

[axis]	1 … number of installed axes
[distance]	-1.0 … 1.0

usage

during positioning/program execution: no

related commands

bln, org, ref

example

```
blp1:0.1
```

conf | def

description

this command defines the type of positioning device. you may select either circle and linear.

syntax

conf[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	0|1 0: circle positioning device, [deg] unit display 1: linear positioning device, [mm] unit display

usage

during positioning/program execution: no

related commands

none

example

conf1:0

dcpl

description

configuration of the number of decimal places of the position display.

syntax

dcpl[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	1…8

usage

during positioning/program execution: no

related commands

none

example

```
dcpl1:4
dcpl2:3
```

dec

description

configuration of the deceleration ramp of an axis for subsequent **move** and **goto** positioning commands.

syntax

dec[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	1…1000 Hz/ms

usage

during positioning/program execution: no

related commands

acc

example

```
dec1:20
dec2:20
```

ecl

description

enable/disable closed-loop positioning mode. if 'closed-loop' is enabled, positioning commands will be executed according to the position feedback of the connected incremental encoder.

syntax

ecl[axis]:[mode]

arguments

[axis]	1 … number of installed axes
[mode]	0|1 0: closed-loop deactivated 1: closed-loop activated

usage

during positioning/program execution: no

related commands

ect, edev, edir, eres, esm, est, esh, ?e, ?ec

example

```
ecl1:1
```

ect

<only applicable in connection with the smc_pc.pci motor controller board>

description

configuration of the encoder signal evaluation hardware.

syntax

ect[axis]:[type]

arguments

[axis]	1 … number of installed axes
[type]	0|1|2|3 0: none 1: internal 2: external Heidenhain IK220 board, incremental 3: external Heidenhain IK220 board, EnDat

usage

during positioning/program execution: no

related commands

ecl, edev, edir, eres, esm, est, esh, ?e, ?ec

example

```
ect1:1
ect2:1
```

edev

description

configuration of the maximum allowed deviation between actual position and commanded target position for closed-loop positioning. this value must not be less than the resolution of the used encoder.

syntax

edev[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	max. deviation

usage

during positioning/program execution: no

related commands

ecl, ect, edir, eres, esm, est, esh, ?e, ?ec

example

```
edev1:0.002
edev2:0.01
```

edir

description

configuration of the encoder rotation sense. for the case the controller returns the encoder position information with the wrong sign, use this command to change it correpondingly.

syntax

edir[axis]:[value]

arguments

[axis]	1 … number of installed axes.
[value]	0|1 0: normal 1: inverted

usage

during positioning/program execution: no

related commands

ecl, ect, edev, eres, esm, est, esh, ?e, ?ec

example

```
edir1:0
edir2:1
```

emips

description

positioning tasks can be made dependent from the status of the integrated digital input port of the controller. if the input port doesn't show the configured state, the controller will not move the corresponding axis.

syntax

emips[axis]:[status]

arguments

[axis]	1 … number of installed axes.
[status]	-1: motion always allowed, status doesn't matter. 0…255: required input port status for motion.

usage

during positioning/program execution: no

related commands

none

example

axis 1 moves only if input port status =1, i.e. bit 0 = 1.
emips1:1
axis 1 moves only if input port status =128, i.e.bit 7 = 1.
emips2:128

eres

description

configuration of the resolution of the connected incremental encoder. example: the encoder delivers 1000 increments per mm or deg, this corresponds to a resolution of 0.001.

syntax

eres[axis]:[resolution]

arguments

[axis]	1 … number of installed axes
[resolution]	encoder resolution

usage

during positioning/program execution: no

related commands

ecl, ect, edev, edir, esm, est, esh, ?e, ?ec

example

```
eres1:0.001
```

esm

description

configuration of the interpolation factor of the encoder input signal. if you change this value you have to change 'eres'correspondingly.

syntax

esm[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	1, 2 or 4

usage

during positioning/program execution: no

related commands

ecl, ect, edev, edir, eres, est, esh, ?e, ?ec

example

```
esm1:4
```

est

< only applicable in connection with the smc_pc.pci motor controller board >

description

configure the type of input signal of the used encoder signal evaluation hardware.

syntax

est[axis]:[signal type]

arguments

[axis]	1 … number of installed axes
[signal type]	0|1|2
	0: standard (TTL)
	1: 1Vss (Heidenhain IK220)
	2: 11µVss (Heidenhain IK220)

usage

during positioning/program execution: no

related commands

ecl, ect, edev, edir, eres, esm, esh, ?e, ?ec

example

```
est1:0
```

esh

description

configure wether the position of the encoder shall be visible on the controller display or not.

syntax

eshw[axis]:[mode]

arguments

[axis]	1 … number of installed axes
[mode]	0|1 0: encoder position display disabled 1: encoder position display enabled

usage

during positioning/program execution: no

related commands

ecl, ect, edev, edir, eres, esm, est, ?e, ?ec

example

```
esh1:1
```

ffast | vfast

description

configuration of the maximum slew speed used for the execution of manual positioning tasks, i. e. motions controlled through the direction keys [<Neg] and [Pos>] or the positioning commands **`fast`**, **`move`** and **`goto`**.

syntax

ffast[axis]:[speed]

arguments

[axis]	1 … number of installed axes
[speed]	1… fmax, where fmax depends on motor type, driver type and positioning hardware properties.

usage

during positioning/program execution: no

related commands

frun, fref

example

```
ffast1:10000
```

fn

description

this command allows to define an additional axis label, which will be shown on the smc's below the axis number.

syntax

fn[axis]:[text]

arguments

[axis]	1 to number of installed axes
[text]	any character string, 32 characters max.

usage

during positioning/program execution: no

related commands

alias

example

```
fn1:theta mono
fn2:2-theta mono
```

frun | vrun

description

configuration of the start-stop speed used for the execution of manual positioning tasks, i. e. motions controlled through the direction keys [<Neg] and [Pos>] or the positioning commands **`fast`**, **`move`** and **`goto`**.

syntax

frun[axis]:[speed]

arguments

[axis]	1 … number of installed axes
[speed]	1… fmax, where fmax depends on motor type, driver type and positioning hardware properties.

usage

during positioning/program execution: no

related commands

ffast, fref

example

```
frun1:200
```

gden | gn

description

configuration of the denominator of the gear factor, i. e. definition of the relation between number of motor steps and covered distance.

gear factor = numerator (number of steps) / denominator (distance)

syntax

gden[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	distance value

usage

during positioning/program execution: no

related commands

gnum

example

```
;linear axis, 1000 steps = 1 mm
conf1:1
gden1:1
gnum1:1000
```

gnum | gz

description

configuration of the numerator of the gear factor, i. e. definition of the relation between number of motor steps and covered distance.

gear factor = numerator (number of steps) / denominator (distance)

syntax

gnum[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	number of steps/increments

usage

during positioning/program execution: no

related commands

gden

example

```
;circle axis, 400 steps = 2 mm
conf1:0
gnum1:400
gden1:2
```

hsdm

description

set home procedure slow-down mode. by default (i.e. for historical reasons), the smc decelerates with a fixed ramp, which is calculated from the slew speed. if parameter hsdm is set to 1, the smc uses the configured deceleration ramp of the configured speed profile instead.

syntax

hsdm[axis]:[0|1]

arguments

[axis]	1 to number of installed axes
[status]	0|1 0: use fixed deceleration ramp 1: use speed profile deceleration ramp

usage

during positioning/program execution: no

related commands

dec, update

example

```
hsdm1:1
```

macc

description

configuration of the acceleration ramp of an axis use during execution of manual positiong tasks.
a motion always starts at start-stop speed (**frun**), accelerates to slew speed (**ffast**) with **acc** Hz/ms and decelerates again down to **frun** with **dec** Hz/ms when the target position has been reached.

syntax

macc[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	1…1000 Hz/ms

usage

during positioning/program execution: no

related commands

mdec

example

macc1:5

mdec

description

configuration of the deleration ramp of an axis use during execution of manual positiong tasks.
a motion always starts at start-stop speed (**frun**), accelerates to slew speed (**ffast**) with **acc** Hz/ms and decelerates again down to **frun** with **dec** Hz/ms when the target position has been reached.

syntax

mdec[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	1…1000 Hz/ms

usage

during positioning/program execution: no

related commands

macc

example

mdec1:100

mdir | mdl

description

configuration of the motor rotation sense. if the motor doesn't run into the desired direction of movement, change this parameter.

note: if you positioning device is equipped with end/limit switches, you must check the assignment of the switches. make sure EL+ switches when the motor runs into positive direction and vice versa. otherwise you have to re-wire your limit switches.

syntax

mdir[axis]:[mode]

arguments

[axis]	1 … number of installed axes
[mode]	0|1 0: motor rotation sense normal 1: motor rotation sense inverted

usage

during positioning/program execution: no

related commands

none

example

```
mdir1:0
mdir2:1
```

mpr

description

configuration of the maximum number of attempts to move to a certain position for closed-loop operation. if the deviation from the target position exceeds 'edev' after the configured number of of attemps, the controller quits positioning.

syntax

mpr[axis]:[number]

arguments

[axis]	1 … number of installed axes
[number]	max. number of attempts: 1…10

usage

during positioning/program execution: no

related commands

prt, ecl

example

```
mpr1:3
```

pqrf

description

configuration of the maximum number of attempts to move to a certain position for closed-loop operation. if the deviation from the target position exceeds 'edev' after the configured number of of attemps, the controller quits positioning.

syntax

pqrf[format]

arguments

[format]	0|1 0: don't append unit specifier to response on position query. 1: append unit specifier. position query '?p1' returns either 1:1.234; or 1:1.234deg;

usage

during positioning/program execution: no

related commands

prt, ecl

example

```
pqrf1
```

prst

note: applies to encoder 'closed loop' operation.

description

configuration of the positioning retry settling time during closed loop operation. this is a delay between subsequent attempts to reduce the deviation between current position and encoder position.

syntax

prst[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	any value in [sec] > 0

usage

during positioning/program execution: no

related commands

ecl, edev, mpr, prt

example

prst1:0.05

prt

description

configuration of a maximum deviation threshold for closed-loop operation. if the position deviation exceeds this value after the first positioning attempt, the controller quits positioning.

syntax

prt[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	deviation threshold

usage

during positioning/program execution: no

related commands

mpr, ecl

example

```
prt1:0.5
```

rofs | nofs

description

configuration of a reference offset value for the case, the reference position does not coincide with the position of the reference indicator.
after succesful execution of the reference search procedure, the current position value will be set to the configured **`rofs`** value.

syntax

rofs[axis]:[value]

arguments

[axis]	1 … number of installed axes
[value]	position value

usage

during positioning/program execution: no

related commands

ref, org

example

```
rofs1:45
```

sdm

please note: this command is hardware-dependent.
it can be used only in conjunction with PCL control ICs.

description

set slow-down signal mode. by default, the smc assumes the usage of a slow-down signal in homing procedures. if a slow-down signal is not provided, use this command to configure the smc correspondingly.

syntax

sdm[axis]:[status]

arguments

[axis]	1 to number of installed axes
[status]	0|1 0: slow-down signal not provided 1: use slow-down signal (default)

usage

during positioning/program execution: no

related commands

update

example

```
sdm1:0
```

unit

description

change unit specifier of the position display.

syntax

unit[axis]:[text]

arguments

[axis]	1 … number of installed axes
[text]	any text, 3 characters max.

usage

during positioning/program execution: no

related commands

none

example

```
unit1:cm
unit2:dm
unit3:km
```

update

description

changes of configuration parameters will take effect immediately, but for the moment they are only temporary active, i. e. until the controller is switched off or you command a **reset**.
if you transfer the **update** command, the current configuration will be saved permanently and the settings will be reloaded automatically the next time the controller starts.

syntax

update{cfg}

arguments

{cfg}	empty|0: default configuration 1: alternative configuration (gear settings and speed-profile only)

usage

during positioning/program execution: no

related commands

?conf, chconf

example

```
#partial configuration of 2 axes
conf1:0
gnum1:200
gden1:1

conf2:0
gnum2:400
gden2:1

update
```

direct commands

direct commands are usually executed immediately after reception. some commands are not allowed during positioning or program execution.

command list:

block	cancel positioning tasks and block comunication
ccnt	clear counter memory (option)
ccount	start continuous counting (option)
cerr	clear error
clr	clear program memory
changeip	change IP address
count[interval]	start single count interval (option)
date[dd:mm:yy]	set system date
dcpl[axis]:[number]	set number of decimal places
dfi[#]	set filter (option)
dhs[#]	set half screen (option)
doff|lcdoff	deactivate position display
don|lcdon	activate position display
echo[0|1|2]	set status message level
eref[axis]{:[+|-]}	search reference position
fast[axis][+|-]	move countinuously at slew speed
fdelete[file]	delete file
fget[file]	transfer file
fwrite{axis}:[file]*[par]	write parameter to file
goto[axis]:[position]	move axes to absolute position
home	execute homing procedure
iel[0|1]	ignore active limit switch
io|dout[value]	set i/o-port output
list	returns a list of available user programs
load{filename}	load program form harddisk
local|loc	activate LOCAL mode
lpox	load position values
move[axis]:[distance]	execute relative movement
movec[axis]:[distance]	execute relative movement and count
org[axis][+|-]	search reference position
passwd[old]:[new]	set administrator password
pg	switch to position display of axes 9-16
pos[axis]:[position]	set position display
priority|prio|pri[0...3]	set progam execution priority
q|quit{axis}	cancel positioning task
reboot|restart	restart controller
ref{axis}	search reference position
remote|rem	activate REMOTE mode
reset|clear	reset controller to power-on state
restart	restart control program
run[axis][+|-]	move countinuously at start-stop speed
save{filename}	save program to disk
shutdown|off	shutdown controller
sound[0|1]	activate/deactivate acoustic signals

step[axis][+|-]	execute single motor step
sleep	delay command execution
spin	start continuous motion
sync	synchronize position
tbc	clear target buffer
tbg	start positioning task according to the target buffer
tbs	set target buffer
time[hh:mm:ss]	set system time
txdel[0|1|2|3]	set delimiter for data transmission
unblock	unblock smc and resume communication
us[value]	set user status
zero{axis}	zero set position display

block

description

this command terminates all active control- and positioning tasks and blocks all subsequent commands except the following:

?s (status query; bit 14 indicates the block state)
?p (position query)

the purpose of this command is, to stop pending positioning tasks and prevent a continuation. the command buffer will be cleared, the program memory is preserved.

syntax

block

arguments

none

usage

during positioning/program execution: yes

related commands

unblock, ?s

example

```
block
```

ccnt

<requires counter option 9000.07>

description

clear and initialize the counter memory.

syntax

ccnt

arguments

none

usage

during positioning/program execution: no

related commands

none

example

```
ccnt
```

ccount

<requires counter option 9000.07>

description

count TTL pulses signal at the BNC input connector (see 'hardware reference') until reception of a **`q`** or **`quit`** command.

syntax

ccount

arguments

none

usage

during positioning/program execution: no

related commands

none

example

```
ccount
```

cerr

description

the last occurred error and a corresponding message is stored and can be retrieved using command ?status. this command can be used to reset this error and clear the error message.

syntax

cerr{axis}

arguments

{axis}	1 … number of installed axes. 0 or no axis specifier: clear all error messages

usage

during positioning/program execution: no

related commands

?err, ?status

example

```
cerr1
```

changeip

description

change current IP address and net mask setting.

syntax

changeip:[ip-address]{*[net mask]}

arguments

[ip-address] any valid IP-address
[net mask] any valid net mask

if [net mask] is not given, 255.255.255.0 is assumed.

usage

during positioning/program execution: no

related commands

?ip

examples

```
changeip:192.168.250.201
changeip:192.168.250.202*255.255.0.0
```

clr

description

clear and initialize the program memory.

syntax

clr

arguments

none

usage

during positioning/program execution: no

related commands

none

example

```
clr
```

count

<requires counter option 9000.07>

description

count TTL pulses signal at the BNC input connector (see 'hardware reference').

syntax

count[interval]

arguments

[interval] count interval in seconds: 1…1000

usage

during positioning/program execution: no

related commands

ccount

example

```
count10.0
```

ccount

<requires counter option 9000.07>

description

count TTL pulses signal at the BNC input connector (see 'hardware reference') until reception of a **`q`** or **`quit`** command.

syntax

ccount

arguments

none

usage

during positioning/program execution: no

related commands

none

example

```
ccount
```

cerr

description

the last occurred error and a corresponding message is stored and can be retrieved using command ?status. this command can be used to reset this error and clear the error message.

syntax

cerr{axis}

arguments

{axis}	1 … number of installed axes. 0 or no axis specifier: clear all error messages

usage

during positioning/program execution: no

related commands

?err, ?status

example

```
cerr1
```

date

description

set current system date.

syntax

date[dd:mm:yy]

arguments

[dd:mm:yy] day:month:year

usage

during positioning/program execution: yes

related commands

time

example

```
date11:01:04
```

dfi

<requires filter option 9000.08>

description

move filter wheel of the filter device 9000.08 to desired position.

syntax

dfi[filter]

arguments

[filter] filter number 1..6

usage

during positioning/program execution: no

related commands

none

example

```
dfi1
```

dhs

<requires half screen option 9000.09>

description

set desired screen at half screen device 9000.09.

syntax

dhs[value]

arguments

[value]	screen position 0…4 0: beam open 1: left half screened 2: right half screened 3: upper half screened 4: bottom half screened

usage

during positioning/program execution: no

related commands

none

example

dhs4

doff | lcdoff

description

deactivate position display. the position display screen will no longer be updated. this may increase program execution speed.

syntax

doff

arguments

none

usage

during positioning/program execution: yes

related commands

don

example

```
doff
```

don | lcdon

description

activate position display. the position display is visible again and will be continuously updated.

syntax

don

arguments

none

usage

during positioning/program execution: yes

related commands

doff

example

don

dout | io

<requires i/o-port option>

description

set signal output of the 8-bit i/o-port to the stated bit pattern.

syntax

dout[value]

arguments

[value] 0 … 255

usage

during positioning/program execution: yes

related commands

none

example

```
dout255
```

echo

description

activate/deactivate output of error and status messages across the active communication interface.

syntax

echo[0|1|2]

arguments

[0|1|2]	0: no messages 1: normal, critical messages only 2: verbose

usage

during positioning/program execution: yes

related commands

none

example

```
echo0
```

eref

description

start reference search procedure of the stated axis. in contrast to `ref` and `org`, the controller evaluates the index signal of a connected incremental encoder for installation of the reference position.
the procedure starts a motion into the stated direction and stops again upon the occurrence of an input signal change from LO to HI at the index input. after that, the motion starts again at low speed into reverse direction and stops when the index signal is recognized again.

syntax

eref[axis]{[direction]}

arguments

[axis]	1 … number of installed axes.
[direction]	+ | -, controller assumes '-' if no direction is stated.

usage

during positioning/program execution: no

related commands

ref, org

example

`eref1+`

fast

description

moves the corresponding axis at the configured **ffast** speed into the stated direction. the controller starts at **frun** and accelerates with **acc**. the axis continues the motion until **q** or **quit** is received. then the controller decelerates with **dec** to **frun** and stops.
the motion will also stop upon the occurrence of a limit switch event.

syntax

fast[axis][direction]

arguments

[axis]	1 … number of installed axes
[direction]	+|-

usage

during positioning/program execution: no

related commands

run, step

example

fast1+

fdelete

description

delete the given file.

syntax

fdelete:[file]

arguments

[file] any suitable file name.

usage

during positioning/program execution: no

related commands

fget, fwrite

example

```
fdelete:test.dat
```

fget

description

the content of the stated file will be transferred across the interface.

syntax

fget:[file]

arguments

[file] any suitable file name.

usage

during positioning/program execution: no

related commands

fdelete, fwrite

example

```
fget:test.dat
```

fwrite

description

write data to file. if file is not existing, it will be generated. data will be appended.

syntax

fwrite{axis}:[file]*[par]

arguments

{axis}	1 … number of installed axes.	
[file]	any suitable file name.	
[par]	a:	write axis specifier to file
	p:	write current position to file
	e:	write current encoder position to file
	ec:	write current encoder counter content to file
	cr:	write a <cr> to file
	lf:	write a <lf> to file
	crlf:	write a <cr>+<lf> to file
	sp:	write a <space> to file
	tab:	write a <tab> to file

usage

during positioning/program execution: no

related commands

fdelete, fget

example

```
# write single line to file: axis specifier,
# position and encoder position separated by <tab>.
fdelete:data.txt
fwrite1:data.txt*a
fwrite:data.txt*tab
fwrite1:data.txt*p
fwrite:data.txt*tab
fwrite1:data.txt*e
fwrite:data.txt*crlf
```

goto

description

the stated axis will move to the stated absolute position using the configured speed profile.
the controller starts at **frun** and accelerates with **acc** to **ffast**. the controller calculates the slow-down position depending on the stated distance, decelerates with **dec** to **frun** and stops finally at the target position.

you may set **acc, dec, frun** and **ffast** prior to the execution of the command. this speed profile remains valid for all subsequent **goto** and **move** positioning tasks.

syntax

goto[axis]:[position]

arguments

[axis]	1 … number of installed axes
[position]	position value in [mm] or [deg], must not exceed +/- 2^23-1 steps/increments

usage

during positioning:	yes
during program execution:	no

related commands

move

example

goto1:1.234

home

description

execute home position search procedure for given axis.

syntax

home[axis]{:}{options}

arguments

[axis]	1 to number of installed axes
{options}	homing procedure definition string (not case sensitive), options separated by ';'.

options:

{jg}{-}{speed}	search end/limit switch position with given direction and speed. after hitting the limit switch, controller reverses direction and runs until the switch is released.
{hs}	home position = switch at ORG signal input, search direction reverse to jg
{hsd}	home position = switch at SD input, search direction reverse to jg
{he}	home position = encoder ECZ (index) signal
{hm}{pos}	find home position in forward direction and set position to pos
{hr}{pos}	find home position in reverse direction and set position to pos default: no motion, pos = eref default speed is frun ignored with {he}
{vl}	set slow search speed for option hm/hr.

usage

during positioning:	yes
during program execution:	no

related commands

org, ref, eref

examples

```
home1                   # no motion, just set status and position (rofs)

home1:hs                # search ORG and stop

home1:he                # search ECZ and stop

home1:jg1000            # search EL+

home1:hs;jg-500;hr90;   # search EL-, search ORG, run free in reverse
                        # direction and set position to 90

home1:hs;jg1000;vl200;hm90;   # search EL+, search ORG, run free
                              # in forward direction and set position
                              # to 90

home1:hsd;jg1000;hm     # search EL+, search SD, run free in forward
                        # direction and set position to 0

home1:hsd;jg-1200;hr15.5      # search EL-, search SD, run free in
                              #  reverse direction and set position
                              # to 15.5

home1:he;jg1000;  # smc:          requires EL+ and encoder Z0 signal
                  # smc_pc.pci:   requires SD+ signal and encoder Z0
                  #               signal

home1:he;jg-500;  # smc:          requires EL- AND encoder Z0 signal
                  # smc_pc.pci:   requires SD- signal and encoder Z0
                  #               signal

                  # NOTE: SD signals must be active until occurrence
                  # of ORG/Z0.
```

iel

description

ignore end/limit switch status on program start. allows program execution despite of active limit switch signal inputs.

note: axes with active end/limit inputs will not move anyway.

this setting will be stored permanently and is not lost if the controller is switched off.

syntax

iel[status]

arguments

[status]	0|1	0: do not start a program in case of active end/limits. 1: don't care about end/limit status.

usage

during positioning/program execution: no

related commands

none

example

```
iel1
```

list

description

returns a list of currently available user program files (*.smc).

syntax

list

arguments

none

usage

during positioning/program execution: no

related commands

load, save

example

list

load

description

load the stated program file from directory **\up** back into the controllers program memory. if you don't give a filename, the controller tries to load a file named **last.smc**.

syntax

load{:filename}

arguments

{: filename }	any suitable filename *without file extension*. file extension **.smc** is added automatically

usage

during positioning/program execution: no

related commands

none

example

load:test

local | loc

description

switch to LOCAL mode, i. e. activate all functions of the touch screen user interface.

syntax

local

arguments

none

usage

during positioning/program execution: yes

related commands

none

example

```
local
```

lpox

description

during the shutdown procedure, the controller saves the last valid position information to the harddisk. use this command to recall the last position and set the display correspondingly. this description may possibly spare you a time consuming referencing procedure.

syntax

lpox

arguments

none

usage

during positioning/program execution: no

related commands

none

example

lpox

move

description

the stated axis will move the stated distance relative to the current position using the configured speed profile. the controller starts at **frun** and accelerates with **acc** to **ffast**. the controller calculates the slow-down position depending on the stated distance, decelerates with **dec** to **frun** and stops finally at the target position.

you may set **acc, dec, frun** and **ffast** prior to the execution of the command. this speed profile remains valid for all subsequent **goto** and **move** positioning tasks.

syntax

move[axis]:[distance]

arguments

[axis]	1 … number of installed axes
[distance]	distance value in [mm] or [deg], must not exceed an absolute position of +/- 2^23-1 steps/increments

usage

during positioning:	yes
during program execution:	no

related commands

movec

example

```
move1:1.0
move2:2.0
move3:3.0
move4:4.0
```

movec

<requires counter option 9000.07>

description

the stated axis will move the stated distance relative to the current position using the configured speed profile. the controller additionally counts incoming pulses at the counter input during the motion.
the controller starts at **frun** and accelerates with **acc** to **ffast**. the controller calculates the slow-down position depending on the stated distance, decelerates with **dec** to **frun** and stops finally at the target position.

you may set **acc, dec, frun** and **ffast** prior to the execution of the command. this speed profile remains valid for all subsequent **goto** and **move** positioning tasks.

the collected counter value will be automaticaly transferred across the interface after completion of the motion.

syntax

movec[axis]:[distance]

arguments

[axis]	1 … number of installed axes
[distance]	distance value in [mm] or [deg], must not exceed an absolute position of +/- 2^23-1 steps/increments

usage

during positioning:	yes
during program execution:	no

related commands

move

example

```
movec1:50.0
```

org

description

start a search reference procedure of the stated axis. if the reference position of the corresponding axis is already installed, no motion is performed.

syntax

org[axis]{[direction]}

arguments

[axis]	1 … number of installed axes.
[direction]	+ | -, assumes '-' if no direction is stated

usage

during positioning/program execution: no

related commands

ref, eref

example

```
org1+
```

osc

description

start oscillation of an axis with given amplitude at current position.

note: position query of an oscillating axis is not supported. 'osc' will be returned instead of a position value.

syntax

osc[axis]:[amplitude]

arguments

[axis]	1 … number of installed axes.
[amplitude]	oscillation range amplitude = 0: stop oscillation command **q[axis]** will also stop an oscillation.

usage

during positioning/program execution: yes

related commands

osc, q

example

```
# start oscillation
osc1:5

# stop oscillation
osc1:0
```

passwd

description

set administrator password. if password is set, controller shutdown, interface configuration and axis configuration via front panel input requires password input.

syntax

passwd[old] [new]

arguments

[old]	current password
[new]	new password to remove password protection, set new=0. password conventions: must be numeric, range: 10000000...99999999, no spaces, no leading '0' (zero).

usage

during program execution: no

related commands

none

examples

```
# set password (password currently not set)
passwd0:26081964

# change password
passwd26081964:12345678

# remove password
passwd12345678:0
```

pg

description

switch back and forth between position display of axes 1-8 and 9-16. this command ist only available if your controller is equipped with more than eight axes.

syntax

pg

arguments

none

usage

during positioning/program execution: yes

related commands

none

examples

pg

pos

description

set current position of the stated axis to desired value

syntax

pos[axis]:[position]

arguments

[axis]	1 … number of installed axes
[position]	must not exceed an absolute position of +/- 2^23-1 steps/increments

usage

during positioning/program execution: no

related commands

zero

example

```
pos1:90.0
```

priority | prio

description

set execution priority for the smc control program within the operating system environment.

syntax

priority[value]

arguments

[value]	0 … 3 0: idle 1: normal 2: high 3: realtime

usage

during positioning/program execution: yes

related commands

none

example

```
priority3
```

q | quit

description

this command causes the immediate termination of positioning processes. transfer the command without axis specifier to stop all motions at the same time. the controller decelerates with the configured deceleration ramp. in contrast to an emergency stop caused by limit switch events, the position information remains valid in this case.

syntax

q{axis}

arguments

{axis} 1 … number of installed axes

usage

during positioning/program execution: yes

related commands

none

example

q

ref

description

start a search reference procedure of the stated axis with negative direction of movement. if the reference position of the corresponding axis is already installed, no motion is performed.

syntax

ref[axis]

arguments

[axis] 1 … number of installed axes.

usage

during positioning/program execution: no

related commands

org, eref

example

```
ref1
```

reboot | restart

description

restart the controller. active programs and positioning tasks will be aborted.

syntax

reboot

arguments

none

usage

during positioning/program execution: yes

related commands

shutdown, reset

example

```
reboot
```

remote | rem

description

switch to REMOTE mode, i. e. deactivate all functions of the touch screen user interface. manual operation is no longer possible.

syntax

remote

arguments

none

usage

during positioning/program execution: yes

related commands

none

example

```
remote
```

reset | clear

description

reset controller to power-on state.

syntax

reset

arguments

none

usage

during positioning/program execution: yes

related commands

reboot, shutdown

example

```
reset
```

restart

description

this command just re-starts the smc control program. use restart in case a program update to load the new firmware.

syntax

restart

arguments

none

usage

during positioning/program execution: no

related commands

reboot, shutdown

example

```
reset
```

run

description

moves the corresponding axis at the configured **frun** speed into the stated direction. the axis continues the motion until **q** or **quit** is received or a limit switch is hit.

syntax

run[axis][direction]

arguments

[axis]	1 … number of installed axes
[direction]	+|-

usage

during positioning/program execution: no

related commands

fast, step

example

run1+

save

description

saves a program under the stated filename in the directory **\up**. if no filename is stated, the current program memory content will be saved as **last.smc**.

syntax

save{: filename}

arguments

{: filename}	any suitable filename *without file extension*. file extension **.smc** is added automatically

usage

during positioning/program execution: no

related commands

none

example

```
save:stepscan
```

shutdown

description

terminate the smc control program and prepare the operating system for swichtching mains power off. active programs and positioning tasks will be aborted.

syntax

shutdown

arguments

none

usage

during positioning/program execution: yes

related commands

restart

example

```
shutdown
```

sleep

description

delay execution of next command for given interval.

syntax

sleep[interval]

arguments

[interval] delay in [s]

usage

during positioning/program execution no

related commands

none

example

```
move1:1.0
sleep2.5
move1:2.0
```

sound

description

some actions of the smc are accompanied by acoustic signals (if speaker is present). this command activates/deactivates the output of acoustic signals.

syntax

sound[value]

arguments

[value]	0|1	0: acoustic signals deactivated 1: acoustic signals activated

usage

during positioning/program execution: yes

related commands

none

example

```
sound0
```

spin

description

move single axis continuously at given speed into stated direction. the axis continues the motion until q or another spin is received. the motion will also stop upon the occurrence of a limit switch event. if no speed value is given, the controller uses the configured speed profile

syntax

spin[axis]:[dir]{speed}

arguments

[axis]	1 to number of installed axes
[dir]	+|-
{speed}	1 to 2457000 [Hz]

usage

during positioning/program execution: no

related commands

ffast, acc, dec, q

examples

```
spin1:-
spin1:+2000
```

step

description

perform a single motor step with the stated into the corresponding direction.

syntax

run[axis][direction]

arguments

[axis]	1 … number of installed axes
[direction]	+|-

usage

during positioning/program execution: no

related commands

fast, run

example

```
step1+
```

sync

description

synchronize current position display with encoder position.

syntax

sync{[axis]:[mode]}

arguments

[axis]	0 to number of installed axes; 0=all axes
[mode]	'p' or 'e'

usage

during positioning/program execution: no

related commands

pos, zero

example

```
# set position of all axes to current encoder position
sync
sync0:p

# set encoder position of all axes to current position
sync0:e

# set position of axis 1 to current encoder position
sync1:p
```

tbc

description

clear target buffer, i.e. set all target position values to 'r0.00'.

syntax

tbc

arguments

none

usage

during positioning/program execution: no

related commands

tbg, tbs, ?tb

example

```
# set target buffer in order to move axis 1 and 2 simultaneously by 1.5.
# start motion, ignore end/limit status.
tbc
tbs1:r1.5
tbs2:r1.5
tbg:iel
```

tbg

description

execute positioning task depending on the target buffer content.
all axes which have a target position set will start simultaneously using the configured speed profile.

NOTE: moving axes will stop immediately upon the occurrence of a limit switch event. if a limit switch input is already active, no motion is executed. use option 'iel' if limit switch status shall be ignored.

NOTE: axes with active limit switches will not move into direction of the active limit switch even if you use option 'iel'.

syntax

tbg{:iel}

arguments

{:iel}	ignore end/limit status.

usage

during positioning/program execution: no

related commands

tbc, tbs, ?tb

example

```
# set target buffer in order to move axis 1 and 2 simultaneously by 1.5.
# start motion, ignore end/limit status.
tbc
tbs1:r1.5
tbs2:r1.5
tbg:iel
```

tbs

description

set target buffer content.

syntax

tbs[axis]:{a|r}[distance]

arguments

[axis]	1 ... number of installed axes.
{a|r}	specify absolute or relative positioning.
[distance]	distance value in [mm] or [deg].

usage

during positioning/program execution: no

related commands

tbc, tbg, ?tb

example

```
# set target buffer in order to move axis 1 and 2 simultaneously by 1.5.
# start motion, ignore end/limit status.
tbc
tbs1:r1.5
tbs2:r1.5
tbg:iel
```

time

description

set current system time.

syntax

time[hh:mm:ss]

arguments

[hh:mm:ss] hours:minutes:seconds

usage

during positioning/program execution: yes

related commands

date

example

```
time13:30:00
```

txdel

description

select delimiter characters which terminate data transferred by the controller.

syntax

txdel[value]

arguments

[value]	0 … 3 0: no delimiter 1: CR (13h) 2: LF (10h) 3: CR+LF

usage

during positioning/program execution: no

related commands

none

example

```
txdel3
```

unblock

description

this command unblocks the controller again.

syntax

unblock

arguments

none

usage

during positioning/program execution: yes

related commands

block, ?s

example

```
unblock
```

us

description

set user status value. this variable can be used to indicate user specific status or events.

please note: the default value is 0. during startup of the controller or a software reset, it will be set back to 0.

syntax

?us

arguments

[value] -2147483648 to 2147483647

usage

during positioning/program execution: yes

related commands

?us

example

us1024

zero

description

set current position of stated axis to 0.00. if a reference position offset vale is configured (see configuration command **rofs**) the current position is set to this value.

syntax

zero{axis}

arguments

{axis}	1 … number of installed axes. 0 or no argument: all axes at the same time.

usage

during positioning/program execution: no

related commands

pos

examples

```
zero1
zero
```

program commands

the controller stores program commands in the RAM. transmission of the command **`start`** triggers the execution of a program.

as already mentioned, the term 'program line' denotes a set of command lines of the category 'program commands'. a program line may consist of up to five different part jobs which will be executed in the order listed below:

1: i/o-port control (input or output)
2: filter- or half-screen control (if applicable)
3: positioning
4: impulse counting (if applicable)
5: execution delay

the following example shows a program line of an 8-axis controller. it consists of 14 command lines (comments have a leading '#':

```
  msg:starting_program
#set i/o-port
  out255
#positioning commands for the eight axes
  1:a1s500r5000a10
  2:a2s500r5000a10
  3:a3s500r5000a10
  4:a4s500r5000a10
  5:a5s500r5000a10
  6:a6s500r5000a10
  7:a7s500r5000a10
  8:a8s500r5000a10
#set filter #1
  fi1
#set halfscreen #2
  hs2
#count pulses at counter input for 0.1 s
  cnt.1
#5 s program delay
  delay5
#end of program line
  nl
```

command list:
<nya: not yet available>

	[n]:{a|r}{+/-}[dist.][s<value>]{[r<value>][a<value>]{d<value>}};	positioning command
	cnt[interval]	count pulses at counter input
nya	**cntc**	count pulses at counter input until **cnts**
nya	**cnts**	stop counting
	delay[interval]	program execution delay
	end	program ende mark
	fi[number]	filter selection
	gosub**|gsb[line]**	subroutine jump
	hs[number]	half-screen selection
	in{bit}{[.][value]}	query i/o-port state
	jmp|jump[line]	jump to program line
	lin[line]	set line number
	msg	send text message
	nl	program line terminator
	out[value]	set i/o-port byte
	res[bit]	reset i/o-port bit
	ret	end of subroutine mark
	set[bit]	set i/o-port bit
	start{axis}{[:]{rep}}	star program execution

positioning command

description

the motion starts with start-stop speed '**s'**, accelerates with '**a'** to run speed 'r**'**. close to the target position, the controller decelerates with '**d'** to start-stop speed **'s'** and finally stops.
the motion will also stop upon the occurrence of a limit switch event.

- if run speed is not stated, the controller will move the full distance at start-stop speed.

- if no deceleration value is stated, the contoller uses the acceleration value for deceleration (i. e. '**d**' = 'a').
.

syntax

[n:]{a}{+/-}[distance][s<value>]{[r<value>][a<value>]{d<value>}}

arguments

[n:]	axis specifier
{a}	positioning mode 'absolute'
{+/-}	direction of motion
[distance]	distance in [mm] or [deg]
[s<value>]	start-stop speed
[r<value>]	run speed
[a<value>]	acceleration rate
{d<value>}	deceleration rate

usage

during positioning/program execution: no

related commands

none

example

```
1:a1.0s200r2500a20d5
1:a0s200r5000a10
1:2.0s500
```

cnt

<requires counter option 9000.07>

description

count TTL pulses at the counter input (see 'hardware reference'). you can reead out the content of the counter memory after program execution using the query command **?cnt**.

syntax

cnt[interval]

arguments

[interval] 0.1 … 600 seconds

usage

during positioning/program execution: no

related commands

none

example

cnt1.0

cntc

<requires counter option 9000.07>

description

count TTL pulses at the counter input (see 'hardware reference') continuously. use **cntc** to stop the count process. you can reead out the content of the counter memory after program execution using the query command **?cnt**.

syntax

cntc

arguments

none

usage

during positioning/program execution: no

related commands

cnts, ?cnt

example

```
cntc
```

cnts

<requires counter option 9000.07>

description

terminates a count process which has been started previously by **cnts**.

syntax

cnts

arguments

none

usage

during positioning/program execution: no

related commands

cntc

example

```
cnts
```

delay

description

perform a program execution delay.

syntax

delay[interval]

arguments

[interval] 0.1 … 600 seconds

usage

during positioning/program execution: no

related commands

none

example

```
delay5.0
```

end

description

indentifies the end of a program.

syntax

end

arguments

none

usage

during positioning/program execution: no

related commands

none

example

end

fi

<requires filter-option 9000.08>

description

set the stated filter number at the filter device.

syntax

fi[filter]

arguments

[filter] 1 … 6

usage

during positioning/program execution: no

related commands

none

example

```
fi2
```

gosub | gsb

description

executes a subroutine starting at the stated line number. the end of the subroutine must be indicated by a final **ret**.

syntax

gosub[line]*[number]

arguments

[line]	line number of the subroutin
[number]	number of repetitions

usage

during positioning/program execution: no

related commands

jump

example

```
gosub10*100
```

hs

<requires half-screen option 9000.09>

description

set the stated half-screen at the half-screen device.

syntax

hs[value]

arguments

[value]	screen position 0…4 0: beam open 1: left half screened 2: right half screened 3: upper half screened 4: bottom half screened

usage

during positioning/program execution: no

related commands

none

example

hs4

in

<requires i/o-port option>

description

return the i/o-port state of either a single bit or all bits simultaneously. if [value] is stated, program execution will be suspended until the state of the signal input corresponds to the stated value.

syntax

in{bit{[.][value]}}

arguments

[bit]	0 … 7
[value]	0 or 1

usage

during positioning/program execution: no

related commands

out

example

```
in;
in3;
in0.1;
```

jump | jmp

description

unconditional jump to stated program line.

syntax

jump[line]

arguments

[line] program line number

usage

during positioning/program execution: no

related commands

none

example

```
jump10
```

lin

description

set line number of the subsequent program line.

syntax

lin[line]

arguments

[line] line number

usage

during positioning/program execution: no

related commands

none

example

```
lin20
```

msg

description

send a single line text message across the active interface prior to execution of the program line.

syntax

msg:[text]

arguments

[text]	character string note: CR and/or LF is not allowed. TAB, spaces and ';' will be removed. ',' will be replaced by '.'.

usage

during positioning/program execution: no

related commands

none

example

msg:starting_program

nl

description

program line terminator.

syntax

nl

arguments

none

usage

during positioning/program execution: no

related commands

none

example

```
1:a0s100r500a10d10
nl
```

out

<requires i/o-port option>

description

set output of the 8-bit i/o-port to stated byte value.

syntax

out[value]

arguments

[value] 0 … 255

usage

during positioning/program execution: no

related commands

set, res, in

example

out128

res

<requires i/o-port option>

description

reset stated output bit of the i/o-port to 0-level.

syntax

res[bit]

arguments

[bit] 0 … 7

usage

during positioning/program execution: no

related commands

set

example

```
res0
```

ret

description

end-of-subroutine specifier.

syntax

ret

arguments

none

usage

during positioning/program execution: no

related commands

gosub

example

```
ret
```

set

<reqires i/o-port option>

description

set stated output bit of the i/o-port to 1-level.

syntax

set[bit]

arguments

[bit] 0 … 7

usage

during positioning/program execution: no

related commands

res

example

```
set7
```

start

description

start program execution.

syntax

start{axis}:{line}{*<repetitions>}

arguments

{axis}	execute programm only for stated axis.
{line}	start programm execution at stated line number
{repetitions}	repeat program for stated number.

usage

during positioning/program execution: no

related commands

q, stop

examplee

```
start:
start1:10
start:*100
```

query commands

query commands request either status information or date from the controller. command execution depends on the operating state of the controller. some commands are valid only while the cotroller is idle, others are also available during program execution and during running positioning tasks.

command list:

?	get general information
?c	get last counter value (option)
?ccb	get control COM input buffer
?cnt	get counter memory (option)
?conf | ?cfg	get current configuration settings
?dll	
?e{axis}	get current encoder position
?ec{axis}	get current encoder counter value
?err	
?i_eib	
***idn? | ?v**	get id and control program version
?in | ?io	get input port status
?ip	
?ip_eib	
?line | ?ln | ?lin	get current line number
?out	
?p{axis}	get current position
?pgm|?getp	get program memory content
?s{axis}	get status information
?s_eib	
?tb	
?us	
?v	

?

description

query of general system information.

syntax

?{:[command]}

arguments

[command] cmdlist: command list
<cmd>: brief explanation of command <cmd>,

usage

during positioning/program execution: no

related commands

none

examples

```
# get general information
?

# get command list
?:cmdlist

# get help for command 'conf'
?:conf
```

?c

<reqires counter-option 9300.07>

description

query of the recently collected counter value.

syntax

?c

arguments

none

usage

during positioning/program execution: no

related commands

none

example

?c

?ccb

description

query the content of the control COM input buffer.

syntax

?ccb

arguments

none

usage

during positioning/program execution: no

related commands

cc_read
fwrite

example

```
# a digital multimeter is connected to COM1 of the smc. in order to
# execute a voltage measurement, the command ?MEAS must be sent to the
multimeter. finally return the response to the control computer.
cc_open              # open smc's control COM port (usually COM1)
cc_write:?MEAS       # smc sends command to multimeter
cc_read              # smc reads multimeter response
?ccb                 # smc returns response to control computer
```

?cnt

<erfordert impulszähler-option 9300.07>

description

query of the counter memory content. count values will be transferred one by one as ASCII-strings.

syntax

?cnt

arguments

none

usage

during positioning/program execution: no

related commands

none

example

```
?cnt
```

?conf | ?cfg

description

query of the current configuration settings.

syntax

?conf

arguments

none

usage

during positioning/program execution: no

related commands

none

example

?conf

?dll

description

query of the current control DLL version.

syntax

?dll

arguments

none

usage

during positioning/program execution: yes

related commands

?v

example

```
?dll
```

?e

description

query of the current encoder position formatted as follows:
<axis>:<position>;<axis>:<position>;…
e.g.: **1:1.234;**

syntax

?e{axis}

arguments

{axis}	1 … number of installed axes without axis specifier: return position values of all axes

usage

during positioning/program execution: yes

related commands

?p, ?s, ?status

example

?e1

?ec

description

query of the current counter content formatted as follows:
<axis>:<value>;<axis>:<value>;…
e.g.: **1:123400;**

syntax

?ec{axis}

arguments

{axis}	1 … number of installed axes without axis specifier: return position values of all axes

usage

during positioning/program execution: yes

related commands

?p, ?s, ?status

example

?ec1

?err

description

query last occurred error and corresponding error message.

syntax

?err{axis}

arguments

{axis}	1 … number of installed axes without axis specifier: return errors of all axes.

usage

during positioning/program execution: yes

related commands

?status, cerr

example

?err1

?in | ?io

<requires i/o-port option>

description

query of the input port status. the controller returns the port status as a byte value 0…255.

syntax

?in

arguments

none

usage

during positioning/program execution yes

related commands

none

example

```
?in
```

?ip

description

query current network IP address of the controller.

syntax

?ip

arguments

none

usage

during positioning/program execution: yes

related commands

changeip

example

```
?ip
```

?line | ?ln | ?lin

description

quetry of the currently executed program line.

syntax

?line

arguments

none

usage

during positioning/program execution: yes

related commands

none

example

?line

?out

<requires i/o-port option>

description

query of the output-port status. the controller returns the port status as a byte value 0…255.

syntax

?out

arguments

none

usage

during positioning/program execution	yes

related commands

none

example

?out

?p

description

query of the current position formatted as follows:
<axis>:<position>;<axis>:<position>;…
e.g.: **1:1.234;**

note: position query of an oscillating axis (see command 'osc') is not supported. 'osc' will be returned instead of a position value.

syntax

?p{axis}

arguments

{axis}	1 … number of installed axes without axis specifier: return position values of all axes

usage

during positioning/program execution: yes

related commands

?e, ?s, ?status

example

?p1

?pgm | ?getp

description

query of the content of the program memory. program lines will transferred as ASCII-strigns line by line.

syntax

?pgm

arguments

none

usage

during positioning/program execution: no

related commands

none

example

```
?pgm
```

?s

description

query of the current operating state, formatted as follows:
<axis>:<state>;<axis>:<state>;…
e.g.: **1:131;**

syntax

?s{axis}

arguments

{axis} 1 … number of installed axes
without axis specifier: return status information of all axes

return values

bit0: 1 axis ready (i.e. axis stopped)
bit1: 2 reference position installed
bit2: 4 end/limit switch EL- active
bit3: 8 end/limit switch EL+ active
bit4: 16 reserved
bit5: 32 reserved
bit6: 64 program execution in progress
bit7: 128 controller ready (i.e. idle, all axes stopped)
bit8: 256 oscillation in progress
bit9: 512 oscillation positioning error (encoder)
bit10: 1024 encoder reference (index) installed

usage

during positioning/program execution: yes

related commands

?p, ?status

example

?s1

?tb

description

query content of the target buffer.

syntax

?tb

arguments

none

usage

during positioning/program execution: yes

related commands

tbc, tbg, tbs

example

```
?tb
```

?status

description

query of the detailed status of the controller.

syntax

?status{axis}

arguments

{axis}	1 … number of installed axes without axis specifier: return status information of all axes

return values

error message	ErrM
error number	ErrN
position	Pos
encoder position	EPos
end/limit status	EL
reference status	REF
encoder reference status	EREF
controller ready	Rdy
status oscillation	Osc
programm running	Prog

usage

during positioning/program execution: yes

related commands

?p, ?s

example

```
?status1
```

?us

description

query user status value.

please note: the default value is 0. during startup of the controller or a software reset, it will be set back to 0.

syntax

?us

arguments

none

usage

during positioning/program execution: yes

related commands

us

example

?us

?v | *idn?

description

query of the current control program version.

syntax

?v

arguments

none

usage

during positioning/program execution: yes

related commands

?dll

example

?v

special commands and features

upon request, the smc control software can be modified and equipped with additional functions and options in order to meet customer specific requirements. if you have a special application which is not covered by the current functionality of the smc, just let us know.

a number of commands which which have been integrated to support special hardware configurations in order to meet customer specific requirements.

configuration

chconf	change configuration
cf	encoder calibration factor
twen	twin encoder

communication

control of hardware connected to the smc's serial interface

cc_close	close control com port
cc_open	open control com port
cc_read	read from control com port
cc_write	write to control com port
txp	enable communication with STX/ETX

operating system

control of File Based Write Filter FBWF

fbwf_configure	configure FBWF
fbwf_disable	disable FBWF
fbwf_enable	enable FBWF
fbwf_status	query FBWF status

hardware

ffst	find next full-step motor position
sysclk	check system clock setting

external hardware

support for Heidenhain EIB7 encoder counter device (Ethernet based)

?i_eib	query information about the connected EIB7
?ip_eib	query EIB7 IP address
?s_eib	query EIB7 status
changeip_eib	change EIB7 IP address
reset_eib	reset an EIB7

chconf

(i/o-port option required)

description

change between configurations. an alternative configuration (i.e. gear- and speed profile settings) can be defined using command update1.
this command is useful if the motor driver resolution can be changed via remote control. the resolution switching input signal of the motor driver must be wired to the smc's i/o-port.

syntax

chconf{{[axis]}:[cfg]}

arguments

[axis]	none or 1 to number of installed axes.
[cfg]	none or 0: use original configuration 1: use alternative configuration

usage

during positioning/program execution: no

related commands

?conf, update

examples

use alternative configuration for all axes
chconf:1

use alternative configuration just for axis 1
chconf1:1

use default configuration for all axes
chconf:0

cf

(encoder required, 'closed loop' operation must be enabled)

description

define calibration factor for encoder operation. if there exists a linear deviation resulting from distension of the encoder grid tape, you may correct this deviation.

note: setting this value just makes sense, if the step resolution of the positioning device is smaller than the deviation.

syntax

cf[axis]:[value]

arguments

[axis]	1 to number of installed axes
[value]	0.99...1.01

usage

during positioning/program execution: no

related commands

none

example

```
# encoder position information: 80.00000
# true position: 80.00035 (for example measured by laser-interferometer)
# resulting calibration factor: 0.999995625019
cf1:0.999995625019
```

twen

description

this command enables the 'twin-encoder' feature. the encoder position feedback of [axis] will be calculated from two encoder inputs.
the controller uses the average of the encoder position values of [axis] and [axis2].

syntax

twen[axis]:[axis2]

arguments

[axis]	1 to number of installed axes
[axis2]	1 to number of installed axes

usage

during positioning/program execution: no

related commands

?e

example

#use encoder input of axis 4 as secondary encoder of axis 1.
twen1:4

cc_open

description

open control COM port.

syntax

cc_open

arguments

none

usage

during positioning/program execution: no

related commands

cc_close, cc_read, cc_write

example

cc_open

cc_close

description

close control COM port.

syntax

cc_close

arguments

none

usage

during positioning/program execution: no

related commands

cc_open, cc_read, cc_write

example

```
cc_close
```

cc_read

description

read content of control COM-port receive buffer.

syntax

cc_read

arguments

none

usage

during positioning/program execution: no

related commands

cc_open, cc_close, cc_write, fwrite

example

```
cc_read
```

cc_write

description

read content of control COM-port receive buffer.

syntax

cc_write :[string]

arguments

[string] ASCII character string.

please not the following limitations:
1) any ',' will be replaced by '.'
2) character '*' is not allowed.

usage

during positioning/program execution: no

related commands

cc_open, cc_close, cc_read

example

cc_write:?MEAS

txp

description

change transfer protocol for serial interface communication.

syntax

txp[value]

arguments

[value]	0|1 0: default 1: STX/ETX (02h/03h) i.e. <STX><smc's response...><ETX>

usage

during positioning/program execution: no

related commands

none

example

txp1

fbwf_configure

(requires Windows XP SP3 with installed FBWF)

description

configure file based write filter FBWF using command script c:\ppe\fbwf\ConfigureFBWF.cmd.

shut down and reboot controller in order to activate changes.

syntax

fbwf_configure

arguments

none

usage

during positioning/program execution: no

related commands

fbwf_enable, fbwf_disable, fbwf_status, reboot

example

fbwf_configure

fbwf_disable

(requires Windows XP SP3 with installed FBWF)

description

disable file based write filter FBWF using command script c:\ppe\fbwf\DisableFBWF.cmd.

shut down and reboot controller in order to activate changes.

syntax

fbwf_disable

arguments

none

usage

during positioning/program execution: no

related commands

fbwf_enable, fbwf_configure, fbwf_status, reboot

example

```
fbwf_disable
```

fbwf_enable

(requires Windows XP SP3 with installed FBWF)

description

enable file based write filter FBWF using command script c:\ppe\fbwf\EnableFBWF.cmd.

shut down and reboot controller in order to activate changes.

syntax

fbwf_enable

arguments

none

usage

during positioning/program execution: no

related commands

fbwf_disable, fbwf_configure, fbwf_status, reboot

example

```
fbwf_enable
```

fbwf_status

(requires Windows XP SP3 with installed FBWF)

description

query status of file based write filter FBWF using command script c:\ppe\fbwf\StatusFBWF.cmd.

syntax

fbwf_status

arguments

none

usage

during positioning/program execution: no

related commands

fbwf_enable, fbwf_disable, fbwf_configure

example

```
fbwf_status
```

ffst

(customer-specific function, requires i/o-port option and wired full-step signal.)

description

find next full-step position of the motor. move into given direction.

syntax

ffst[axis]:[direction]

arguments

[axis]	1 to 8
[direction]	+|-

usage

during positioning/program execution: no

related commands

none

example

```
ffst1:+
```

sysclk

(for service purpose only)

description

check current system clock setting and adjust value if necessary.

note: make sure axis 1 can move a small distance and no limit switch is active.

syntax

sysclk

arguments

none

usage

during positioning/program execution: no

related commands

none

example

```
sysclk
```

?i_eib

(external Heidenhain EIB7 encoder counter must be available on the network)

description

query Heidenhain EIB7 status information.

syntax

?i_eib

arguments

none

usage

during positioning/program execution: no

related commands

none

example

?i_eib

possible response:
id: <device id>
IP-address: 192.168.253.212
netmask: 255.255.255.0
gateway: 192.168.253.0
DHCP: disabled
hostname: EIB741-33472242
MAC: 00:A0:CD:10:00:EB
bootmode: default firmware with user settings
factory firmware version: 63328108
user firmware version: 00000000

NOTE: if 'bootmode:' is 'default firmware with default settings', the EIB7 uses the default IP address 192.168.1.2 for communication even if 'IP-address:' states a different setting.

?ip_eib

(external Heidenhain EIB7 encoder counter must be available on the network)

description

query current network IP address setting used for communication with Heidenhain EIB7 encoder counter device.

syntax

?ip_eib

arguments

none

usage

during positioning/program execution: no

related commands

changeip_eib

example

```
?ip_eib
```

?s_eib

(external Heidenhain EIB7 encoder counter must be available on the network)

description

query of the current encoder counter status of an axis which uses a Heidenhain EIB7 device.

syntax

?s_eib[axis]

arguments

[axis] 1 to number of installed axes

response:
Heidenhain EIB7 input X1<1..4>
position: <position>
status: <status>

usage

during positioning/program execution: no

related commands

none

example

```
?s_eib1
```

reset_eib

(external Heidenhain EIB7 encoder counter must be available on the network)

description

reset connected Heidenhain EIB7 device. this corresponds to a power off/on cycle.

syntax

reset_eib

arguments

none

usage

during positioning/program execution: no

related commands

none

example

```
reset_eib
```

changeip_eib

(external Heidenhain EIB7 encoder counter must be available on the network)

description

change IP address and net mask setting of a Heidenhain EIB7 encoder counter device. the EIB7 must be active and currently controlable by the smc.

syntax

changeip_eib:[ip-address]{*[net mask]}

arguments

[ip-address]	any valid IP-address
[net mask]	any valid net mask

if [net mask] is not given, 255.255.255.0 is assumed.

usage

during positioning/program execution: no

related commands

?ip_eib

example

```
changeip_eib:192.168.253.210
changeip_eib:192.168.253.211*255.255.0.0
```

appendix