# Shortform SMC *pollux / SMC pollux NT*

Venus-2 commands consist of ASCII- signs which are interpreted in the controller and immediately executed.
A software development surrounding to produce the control programs is not needed.
The commands can be produced by any Host and whatever programming language you are using, on condition that there is an access to the RS- 232 interface. In the simplest way the commands are directly transmitted to the controller via an ASCII terminal.

## Command syntax

The commands are assembled following this scheme:
[parameter] _ [axis index] _ [command] _ blank, (space) or (SP) blank

## Command ending character while transmitting

The Venus- 2 command must be terminated with a blank (SP). [Parameter] SP [Axes index] SP [Venus- 2 command] SP

## Command ending character while receiving

[1st parameter] SP [2nd parameter] SP [n- parameter] CR LF
Data which are delivered by the controller are always terminated with ASCII (CR) and (LF).

## RS-232 Interface Configuration

| | |
|---|---|
| Data bits | 8 |
| Stop bits | 1 |
| Parity | no |
| Handshake | no |
| Baudrate | 19200 |

## Error numbers:

| | |
|---|---|
| 1.... 4 | Internal error |
| 1001 | Wrong parameter type |
| 1002 | parameter stack underrun |
| 1003 | parameter out of range |
| 1004 | Movement range should be exceeded caused by limit switch |
| 1008 | See 1002 |
| 1010 | RS-232 input buffer lacking space (<30 para left) |
| 1015 | Parameter out of the movement area (softlimit) |
| 1100 | Both Limit Switches are active |
| 2000 | Unknown command |

## First steps:

Normaly the controller is configured for the delived stage. If delivery contains more then one controller, the controllers are labeled with an axisnumber (the address of the controller)

The actual settings are documented in a *.txt file, which is downloadable with our demo-application SMC-Pollux xxx.exe, also documented as pdf-file.

For the first step, hyperterminal, any other terminal-program, or the program smc-pollux xxx.exe could be a good choice.

Due to the fact that the controller is ready configured, the main commands for customers use
are the basic move commands, homing and position and status query:

If a communication is established just type following commands: (axis 1 must be connected)

```
1 np            ; controller returns the actual position, after reset always 0.00000
1 ncal          ; axis searches the limit reverse the release point is the physical zero-position
                of the system
2.0 1 nr        ; axis moves 2 unit relative (usually 2 mm or 2 degree)
1 np            ; returns the actual position, now 2.00000
4.0 1 nm        ; axis moves to absolute 4.0 units
1 nst           ; if stage is just moving return value =1
                ; if stage not moving, means at target, return value = 0
```

## Multiple Axis Application:

Please verify that all daisy-chained controller do have different axis-addresses. If no address labled you should assume that the address is **'1'.**

## Assigning new controller address:

Assigning a new controller address is very easy, connect **only one** controller to the RS-232, connect with program *SMC-pollux xxx.exe* and change the address with dialog *Controller|Assign Address*. After this save the new settings with Button *'nsave'* .

### Command-Overview:

| Command | Description | Parameters | R/W | Range | Example |
|---|---|---|---|---|---|
| nrmove (nr) | move relative, without query status | *relpos axisid* | w | -1000.0 ..+1000.0 | 1.0 1 nr |
| nmove (nm) | move absolute without query status | *abspos axisid* | w | -1000.0 ..+1000.0 | 5.1 1 nm |
| npos (np) | returns actual position | *axisid* | r | | 1 np |
| npush | loads targets on stack (for synchronized start) | *position axisid* | w | -1000.0 ..+1000.0 | 10.0 1 npush |
| npop | removes values from stack | *axisid* | w | | 1 npop |
| setnpos | redefines the actual position | *abspos axisid* | w | -1000.0 ..+1000.0 | 0.0 1 setnpos |
| nstatus (nst) | returns actual status | *axisid* | r | | 1 getaxis |
| getnerror (gne) | returns actual error number | *axisid* | r | | 1 gne |
| getmerror (gme) | returns machine error number | *axisid* | r | | 1 gme |
| nabort | stops a move | *axisid* | w | | 1 nabort |
| <CtrlC> | stops move of all connected axes | | w | | <CTRL-C> hex 3 |
| speed | starts a constant velocity move | *+/-speed axisid* | w | | 2.5 1 speed |
| stopspeed | stops constant velocity move | *axisid* | w | | 1 stopspeed |
| setnpowerup | defines startup behaviour | *value axisid* | w | | 0 1 setnpowerup |
| getnpowerup | returns startup behaviour | *axisid* | | | 1 getnpowerup |
| ncal | homing (search limit reverse) | *axisid* | w | | 1 ncal |
| nrm | rangemeasure (search limit forward) | *axisid* | w | | 1 nrm |
| nversion | returns the firmware-version | *axisid* | r | | 1 nversion |
| nidentify | returns the controller identification | *axisid* | r | | 1 nidentify |
| getnserialno<br>getserialno | returns the serial-number | *axisid* | r | | 1 getnserialno<br>1 getserialno |
| getnoptions | returns the options-code | *axisid* | r | | 1 getnoptions |
| getaxis | returns the wether axis is active or not | *axisid* | r | | 1 getaxis |
| setaxis | defines if axis active or not | *status axisid* | w | 0..2 | 0 1 setaxis |
| getswst | returns the status of limit-inputs | *axisid* | r | | 1 getswst |
| getsw | returns the setting of limit-inputs | *axisid* | r | | 1 getsw |
| setsw | defines the limit-switch-status | *status 0 axisid*<br>*status 1 axisid* | w | 0..2 | 1 0 1 setsw<br>1 1 1 setsw |
| getmotiondir | returns the setting of the direction of motion | *axisid* | r | | 1 getmotiondir |
| setmotiondir | defines the direction of motion | *value axisid* | w | 0..1 | 0 1 setmotiondir |
| getncalswdist | returns the calswitch-distance | *axisid* | r | | 1 getncalswdist |
| setncalswdist | defines the calswitch-distance | *distance axisid* | w | 0..1.0 | 0.5 1 setncalswdist |
| getpitch | returns the pitch of the stage | *axisid* | r | | 1 getpitch |
| setpitch | defines the pitch of the stage | *pitch axisid* | w | 0.1..50 | 1.0 1 setpitch |
| getnvel (gnv) | returns the velocity for move | *vel axisid* | r | | 1 gnv |

| | | | | | |
|---|---|---|---|---|---|
| setnvel (snv) | defines the velocity for move | *axisid* | w | 0.0001...2000.0 | 12.0 1 snv |
| getnaccel (gna) | returns the acceleration for move | *axisid* | r | | 1 gna |
| setnaccel (sna) | defines the acceleration for move | *acc axisid* | w | 1..2000 | 120.0 1 sna |
| getnstopdecel | returns the acceleration for a commanded stop or limit-switch activation | *axisid* | r | | 1 getnstopdecel |
| setnstopdecel | defines the acceleration for a commanded stop or limit-switch activation | *acc axisid* | w | 1..2000 | 400.0 1 setnstopdecel |

| | | | | | |
|---|---|---|---|---|---|
| getncalvel | returns the speed for cal-move | *axisid* | r | | 1 getncalvel |
| setncalvel | defines the speed for cal-move | *value 1 axisid*<br>*value 2 axisid* | w | | 5.0 1 1 setncalvel<br>0.1 2 1 setncalvel |
| getnrmvel | returns the speed for rm-move | *axisid* | r | | |
| setnrmvel | defines the speed for rm-move | *value 1 axisid*<br>*value 2 axisid* | w | | 50 1 1 setnrmvel<br>0.1 2 1 setnrmvel |
| getumotmin | returns the motor-umotmin | *axisid* | r | | 1 getumotmin |
| setumotmin | defines the motor-umotmin (*) | *value axisid* | w | see table | 500 1 setumotmin |
| getumotgrad | returns the motor-umotgrad | *axisid* | r | | 1 getumotgrad |
| setumotgrad | defines the motor-umotgrad (*) | *value axisid* | w | see table | 20 1 setumotgrad |
| | | | | | |
| getnlimit | returns the travel-limits | *axisid* | r | | 1 getnlimit |
| setnlimit | defines the travel-limits | *low high axisid* | w | -1000.0 ..1000.0 | 0.0 100.0 1 setnlimit |
| nsave | save all parameters in flash-memory | *axisid* | w | | 1 nsave |
| nrestore | restores the last saved parameters | *axisid* | w | | 1 nrestore |
| ngsp | returns the stack-counter | *axisid* | r | | 1 ngsp |
| nclear | clear controllers internal stack | *axisid* | w | | 1 nclear |
| | | | | | |
| setaxisno | define address of the controller | *axisid* | w | 1..16 | 2 setaxisno |
| getaxisno | returns address of the controller | *axisid* | r | 1..16 | 2 getaxisno |
| nreset | resets the controller | *axisid* | w | | 1 nreset |

| | | | | | |
|---|---|---|---|---|---|
| setuv | stores uservalue (32-bit int) to memory | *value Id axisid* | w | | 12222 1 2 setuv |
| getuv | load uservalue (32-bit int) from memory | *id axisid* | r | | 1 2 getuv |
| | | | | | |

**Pollux NT (closed-loop) specific commands**:

| | | | | | |
|---|---|---|---|---|---|
| getcloop | returns if closed loop active or not | *axisId* | r | | 1 getcloop |
| setcloop | defines if closed loop active or not | *status axisId* | w | 0..1 | 1 1 setcloop |
| getclwindow | returns the defined closed-loop in-target window | *axisId* | r | | 1 getclwindow |
| setclwindow | defines the closed-loop in-target window | *size axisId* | w | 0.0-1.0 | 0.001 1 setclwindow |
| getclwintime | returns the defined closed-loop in- window time [ms] | *axisId* | r | | 1 getclwintime |
| setclwintime | defines the defined closed-loop in- window time [ms] | *time axisId* | w | 0-8191 | 10 1 setclwintime |
| getnrefvel | returns the speed for refmove (index search) | *axisId* | r | | 1 getnrefvel |
| setnrefvel | defines the speed for refmove (index search) | *value 1 axisid*<br>*value 2 axisid* | w | | 1.0 1 1 setnrefvel<br>2.0 2 1 setnrefvel |
| nrefmove | starts a refmove (index-search) | *abstarget axisId* | w | | 5.0 1 nrefmove |
| getrefst | returns the status of refmove (index-search) | *axisId* | r | | 1 getrefst |
| getref | returns the transition of the index mark | *axisId* | r | | 1 getref |
| setref | defines the transition of the index mark | *transition axisId* | w | | 0 1 setref |
| getemergency | returns the configuration of emergency shortcuts | *axisId* | r | | 1 getemergency |
| setemergency | defines the configuration of emergency shortcuts | *config axisId* | w | 0-3 | 3 1 setemergency |
| getscaleinterface | returns the type of encoder | *axisId* | r | | 1 getscaleinterface |
| setscaleinterface | defines the type of encoder | *type axisId* | w | 0-2 | 1 1 setscaleinterface |

## Some motorsettings for motors used by MICOS:

The values could vary, dependent on the desired load and application!

| Motor | Nominal Current [Amp] | Coil resistance [Ohms] | commonly used with stages | *umotmin* | umotgrad |
|---|---|---|---|---|---|
| Pollux Motor I and II | 1.2 | | VT80,DT80 | *2000..2300* | 110 |
| Pollux Motor III | 1.2 | | VT80,DT80 | *2000* | 110 |
| 4H4018 | 1.7 | 1.7 | VT80, DT80 , HT90 | *3200* | 90 |
| ST-4018 L1206 halfcoil | 1.2 | 3.3 | LS110, PLS85, DT65N, ES65, MA35 | *3500* | 140 |
| PK-245-01B fullcoil | 0.85 | 6.6 | | *4000..5000* | 400 |
| PK-245-01B halfcoil | 1.2 | 3.3 | | *3000..3500* | 150 |
| PK-244-01B halfcoil | 1.2 | 3.3 | ES-100 | *3500* | 140 |
| PK-244M-01B fullcoill | 0.85 | 6.6 | DT-50R DT-80R | *4000-5000* | 140 |
| | | | | | |
| ZSS-43-200-1.2-E parallel | 1.2 | 2.6 | LS110, PLS85, PRS110, HPS170, MS8,DT65N, WT90 | *4300* | 150 |
| ZSS-42-200-1.2-E parallel | 1.2 | 1.6 | NPE200, MA35 | *3500* | 140 |
| ZSS-52-500-2.5E parallel | 2.5 | 0.6 | LS180. UPM160 | | |
| AM1524-A0.25 | 0.25 | 12.5 | MP20S MP20L, MT55, MT60,MT40, ASS5/ADS5 | *2500 –3500* | 140 |
| ZSS-25-200-1.2 parallel | 1.2 | 0.95 | WT85, WT100, MT40,MTS-70 | *2000* | 24 |
| ZSS-32-200-1.2 parallel | 1.2 | 1.3 | MT60 | *3300* | 60 |
| PK266-E2.0 parallel | 2 | 0.9 | DT120, UPL-160, WT120 | *3000..3200* | 160..250 |
| | | | | | |
| CTP11-13 | 1.3 | 3.3 | | *3500* | 140 |
| ST-2818S1006 | 0.95 | 3.4 | | *4000* | 100 |
| LIN-208-17-1 | 0.8 | 5.4 | | *3800-5400* | 20 |
| LIN-211-18-02 | 1.3 | 1.3 | VT-40 MP-20 | *2400* | 60 |
| MICOS 2Ph-018 | 0.24 | 20.4 | VT-21, MP-21, ES-50 | *6500* | 45 |
| | | | | | |

Nominal Current: motors rated continues current, not the real current with the documented settings
Coil resistance: motor single phase resistance (varies depending on wiring type, fullcoil, halfcoil , serial or parallel)

Please note: Without damper motors get stuck in the resonance area of the motor, which is mainly in the range of 4 rev/sec (200 fullstep motor). A damper (oriental) eliminates perfectly!

Read the actual parameters: axisId getumotmin and axisId getumotgrad

Write new parameters: value axisId setumotmin and value axisId setumotgrad

If parameters ok, save flash-memory axisId nsave

## Power-Connector:

| | |
|---|---|
| manufacturer | Binder Connector |
| type | Kabeldose gerade Serie 719 3pol |
| Art.Nr (manufacturer) | 09 9748 70 03 |

| Binder 3 pin | Function |
|---|---|
| 1 | +24 V |
| 2 | - |
| 3 | GND |

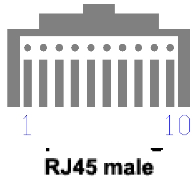
## Motor Interface: DSub9 standard

| DSub9emale | Function |
|---|---|
| 1 | Phase A+ |
| 2 | Phase A- |
| 3 | Phase B+ |
| 4 | Phase B- |
| 5 | Gnd limit-common |
| 6 | cal-switch (limit reverse) |
| 7 | rm-switch (limit forward) |
| 8 | + 5V for active sensors |
| 9 | nc. |

## Interface-Cable RS-232:

| DSub9f | function | color | RJ45 male 10 pin |
|---|---|---|---|
| 2 | RxD | yellow | 5 |
| 3 | TxD | green | 6 |
| 5 | GND | brown | 8 |
| | | | |

Caution: PC-side : connect pin 1+4+6
connect pin 7+8



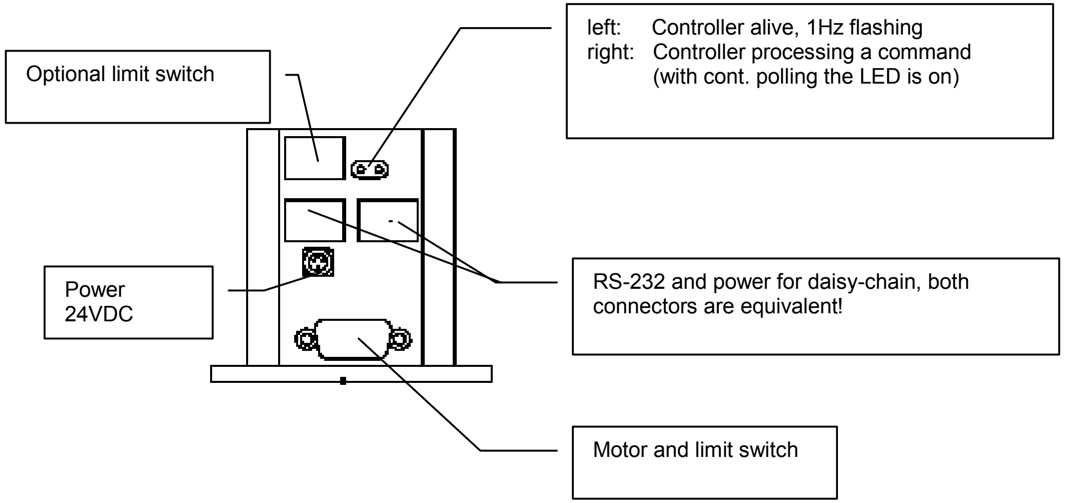
RJ45 10 pin
front view to contacts

## Pollux-Motor Limit Switch Connection

Open-leads to connect directly to the switches (active and passive)

| lead-color | Function |
|---|---|
| white | + 5V for active sensors |
| yellow | cal-switch (limit reverse) |
| green | rm-switch (limit forward) |
| brown | Gnd limit-common |

## Pollux:

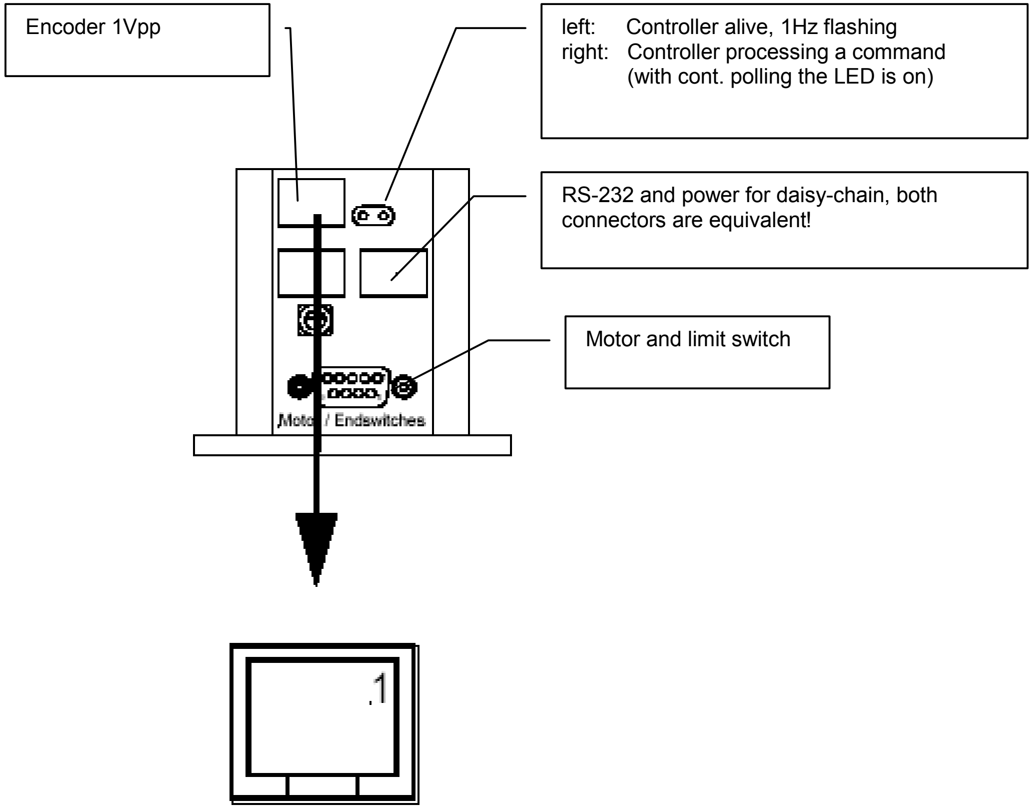


left: Controller alive, 1Hz flashing
right: Controller processing a command
(with cont. polling the LED is on)

RS-232 and power for daisy-chain, both connectors are equivalent!

Motor and limit switch

## Pollux NT (closed-loop):



Encoder 1Vpp

left: Controller alive, 1Hz flashing
right: Controller processing a command
(with cont. polling the LED is on)

RS-232 and power for daisy-chain, both connectors are equivalent!

Motor and limit switch

| RJ-45 10 pin | Function 1Vpp Encoder |
|---|---|
| 1 | 5V |
| 2 | |
| 3 | Sin + (A+) |
| 4 | Sin – (A-) |
| 5 | Cos + (B+) |
| 6 | Cos – (B-) |
| 7 | Ref + (Index+ |
| 8 | Ref- (Index-) |
| 9 | Gnd |
| 10 | |