Command Language

Venus-1
# Contend

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contend</td>
<td>2</td>
</tr>
<tr>
<td>Introduction</td>
<td>5</td>
</tr>
<tr>
<td>Venus-1 is an interpreter language</td>
<td>5</td>
</tr>
<tr>
<td>Conventions</td>
<td>6</td>
</tr>
<tr>
<td>Index of axes</td>
<td>6</td>
</tr>
<tr>
<td>Index of the dimension</td>
<td>6</td>
</tr>
<tr>
<td>Index for On/Off</td>
<td>6</td>
</tr>
<tr>
<td>Get and set commands</td>
<td>6</td>
</tr>
<tr>
<td>Important ASCII characters of the Venus-1 language</td>
<td>9</td>
</tr>
<tr>
<td>Stack</td>
<td>9</td>
</tr>
<tr>
<td>Command syntax</td>
<td>12</td>
</tr>
<tr>
<td>Venus-1 commands</td>
<td>14</td>
</tr>
<tr>
<td>Host / Terminal-Mode</td>
<td>15</td>
</tr>
<tr>
<td>mode</td>
<td>15</td>
</tr>
<tr>
<td>Homing</td>
<td>16</td>
</tr>
<tr>
<td>calibrate (cal)</td>
<td>16</td>
</tr>
<tr>
<td>rangemeasure (rm)</td>
<td>17</td>
</tr>
<tr>
<td>setcalvel</td>
<td>18</td>
</tr>
<tr>
<td>setrmvel</td>
<td>19</td>
</tr>
<tr>
<td>Query : Position, Status and Error</td>
<td>20</td>
</tr>
<tr>
<td>pos (p)</td>
<td>20</td>
</tr>
<tr>
<td>status (st)</td>
<td>21</td>
</tr>
<tr>
<td>geterror (ge)</td>
<td>23</td>
</tr>
<tr>
<td>getmerror (gme)</td>
<td>23</td>
</tr>
<tr>
<td>Joystick-Functions:</td>
<td>24</td>
</tr>
<tr>
<td>joystick (j)</td>
<td>24</td>
</tr>
<tr>
<td>getjoyspeed</td>
<td>25</td>
</tr>
<tr>
<td>setjoyspeed (js)</td>
<td>25</td>
</tr>
<tr>
<td>getjoybspeed</td>
<td>26</td>
</tr>
<tr>
<td>setjoybspeed</td>
<td>26</td>
</tr>
<tr>
<td>getjoysticktype</td>
<td>27</td>
</tr>
<tr>
<td>setjoysticktype</td>
<td>27</td>
</tr>
<tr>
<td>Configuration</td>
<td>28</td>
</tr>
<tr>
<td>getpowerup</td>
<td>28</td>
</tr>
<tr>
<td>setpowerup</td>
<td>28</td>
</tr>
<tr>
<td>getdim</td>
<td>29</td>
</tr>
<tr>
<td>setdim</td>
<td>29</td>
</tr>
<tr>
<td>getpdisplay</td>
<td>30</td>
</tr>
</tbody>
</table>
setpdisplay 30
getlimit 31
setlimit 32
setpos (sp) 33
scale 34
getaxis 35
setaxis 35
getpitch 36
setpitch 36
getsw 37
setsw 37
getswst 38
getunit 38
setunit 39
getvel (gv) 40
setvel (sv) 40
getaccel (ga) 41
setaccel (sa) 41
getaccelfunc 42
setaccelfunc 43

**Positioning-Commands:**  44
move (m) 44
rmove (r) 45
Ctrl C 45

**Velocity-Mode:**  46
speed 46
stopspeed 47

**Parameter**  48
restore 48
save 48

**Identification of the controller:**  49
identify 49
version 50
getserialno 50

**Stack-Commands**  51
pop 51
gsp 51
clear 52
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting of the motor-current:</td>
<td>53</td>
</tr>
<tr>
<td>setumotmin</td>
<td>53</td>
</tr>
<tr>
<td>getumotmin</td>
<td>54</td>
</tr>
<tr>
<td>setumotgrad</td>
<td>55</td>
</tr>
<tr>
<td>getumotgrad</td>
<td>56</td>
</tr>
<tr>
<td>setmp</td>
<td>57</td>
</tr>
<tr>
<td>getmp</td>
<td>57</td>
</tr>
<tr>
<td>gi</td>
<td>58</td>
</tr>
<tr>
<td>Additional Commands</td>
<td>59</td>
</tr>
<tr>
<td>Special Output:</td>
<td>59</td>
</tr>
<tr>
<td>setsout</td>
<td>59</td>
</tr>
<tr>
<td>getsout</td>
<td>59</td>
</tr>
<tr>
<td>Encoder closed-loop:</td>
<td>60</td>
</tr>
<tr>
<td>getcloop</td>
<td>60</td>
</tr>
<tr>
<td>setcloop</td>
<td>61</td>
</tr>
<tr>
<td>getclfactor</td>
<td>62</td>
</tr>
<tr>
<td>setclfactor</td>
<td>62</td>
</tr>
<tr>
<td>getnselpos</td>
<td>63</td>
</tr>
<tr>
<td>setnselpos</td>
<td>63</td>
</tr>
<tr>
<td>Commands for the refmove:</td>
<td>64</td>
</tr>
<tr>
<td>getref</td>
<td>65</td>
</tr>
<tr>
<td>setref</td>
<td>65</td>
</tr>
<tr>
<td>getrefst</td>
<td>66</td>
</tr>
<tr>
<td>getrefvel</td>
<td>67</td>
</tr>
<tr>
<td>setrefvel</td>
<td>67</td>
</tr>
<tr>
<td>refmove</td>
<td>67</td>
</tr>
<tr>
<td>Ethernet-Interface</td>
<td>69</td>
</tr>
<tr>
<td>getipadr</td>
<td>69</td>
</tr>
<tr>
<td>setipadr</td>
<td>69</td>
</tr>
<tr>
<td>getmacadr</td>
<td>70</td>
</tr>
<tr>
<td>Error messages:</td>
<td>71</td>
</tr>
</tbody>
</table>
Introduction

This handbook describes the command language Venus-1. User with program experience are addressed.

All commands are described in detail concerning function and syntax. The handbook is equipped with an extensive look up part, which rapidly obtains a survey and allows a fast finding of the commands. The controllers fundamental function and the different operation types are not reviewed here. Please take these information’s from the appropriate instruction manual.

Venus-1 is an interpreter language

A compiler for translating command language Venus-1 is not required, because Venus-1 is a command language which translates and executes each command immediately in the controller. The commands which are sent to the controller are not intricate escape-sequences, but self-explaining commands which exclusive use printable ASCII-characters.
Conventions

Index of axes
In Venus-1 an index belongs to each axis

<table>
<thead>
<tr>
<th>Axis</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>virtual velocity axis</td>
<td>0</td>
</tr>
<tr>
<td>X-axis</td>
<td>1</td>
</tr>
<tr>
<td>Y-axis</td>
<td>2</td>
</tr>
<tr>
<td>Z-axis</td>
<td>3</td>
</tr>
</tbody>
</table>

Index of the dimension
The dimension can be put in from one-dimensional to three-dimensional.
The assignment for the index is appropriate.

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-dimensional (only X)</td>
<td>1</td>
</tr>
<tr>
<td>Two-dimensional (X-Y)</td>
<td>2</td>
</tr>
<tr>
<td>Tree-dimensional (X-Y-Z)</td>
<td>3</td>
</tr>
</tbody>
</table>

Index for On/Off
Different commands permit switching functions of the controller on or off. The index convention for on/off is shown in the following table.

<table>
<thead>
<tr>
<th>Status</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>On</td>
<td>1</td>
</tr>
<tr>
<td>Off</td>
<td>0</td>
</tr>
</tbody>
</table>

Get and set commands
The convention for the names is following
**set-commands** parameter are written
**get-commands** parameter are read

-**1 parameter**
  Sometimes it is possible to use a value of -1 for the axis parameter. That means that all axes are addressed.

  example:
  The command `2 getaxis` returns parameter of the y-axis.
  The command `-1 getaxis` returns the parameters of all axes.

**Data format:**
Venus-1 commands are sent as ASCII character string, there is distinguished between upper and lower characters.
The command line is put together from parameters and commands as shown above.

**parameter:**
The parameter have to be separated by a space character.
Between parameter and command a space character is recommended, but not always necessary.

**Allowed characters:**
Allowed characters are: a-z  A-Z  0-9  . - + and space character

**Number format:**
Floating point numbers use the point (not a comma).

**command separating character:**
In the Host Mode a command is terminated by a space or <CR> character. The command is executed immediately after receiving the separating character. A following <LF> character is ignored. Normally it is a good method to send <CR><LF> combination after the command.
In the Terminal Mode a command is terminated by a space or <CR> (carriage return) character, but the entire command line is interpreted after receiving the <CR> character. It is possible to edit the command line with the <BS> (backspace) character.

**Command buffer:**
The interface of the controller owns an internal memory with FIFO-structure (First In First Out). The received commands are stored there and they are executed in order of their input. Altogether the FIFO takes up to 1024 characters (not commands). After switching off the power of the controller, this command buffer is erased.

**Replied data messages:**
Returned characters of the controller are always terminated with <CR> and <LF>. (ASCII Decimal 13 and 10)
Different parameter sets are returned in one line, the parameter are separated by a space character and the whole line is terminated with the <CR><LF> combination.
Sometimes a parameter set consists of several lines. Each line contains one parameter terminated with the <CR><LF> combination.
Important ASCII characters of the Venus-1 language

<table>
<thead>
<tr>
<th>ASCII-character</th>
<th>Function</th>
<th>Decimal</th>
<th>Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR</td>
<td>Carriage return</td>
<td>13</td>
<td>D</td>
</tr>
<tr>
<td>LF</td>
<td>Line Feed</td>
<td>10</td>
<td>A</td>
</tr>
<tr>
<td>SP</td>
<td>Space</td>
<td>32</td>
<td>20</td>
</tr>
</tbody>
</table>

Transmitting scheme:

Sending data to the controller

value SP value SP command SP

Receiving data from the controller

value SP value SP value CR LF

or

value CR LF
value CR LF

Stack

All the sent parameters (not any commands) are put on a so-called stack, which altogether can receive 99 entries. The stack works in according to the LIFO principle (Last In First Out) that means, that the last arrived parameter is the first available. According to this principle the required parameter are popped off the stack with each command. The Interpreter knows for each command how many parameter have to be taken from the stack.

With an example of a positioning command the process in the stack shall be made plain.

The controller is here focused on the dimension 3 and expects 3 parameters for the command.
With the execution of this command the 3 parameter are popped off the stack, which is empty again after execution.

```
59 4 12  move
```

<table>
<thead>
<tr>
<th>stack after the data input</th>
<th>stack after the command execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td>99</td>
</tr>
<tr>
<td>59</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>move</td>
<td>move</td>
</tr>
</tbody>
</table>
Another example:

The input of the command
1848223489 move
places all parameter on the stack. With the execution of the command move the upper 3 parameter are removed off the stack (only if dimension 3 is selected). The remaining parameters stay on the stack as long as they are used by another command.
Command syntax

Commands

A single command will be defined as followed.

- [parameter] [Venus-1 command]
- [parameter] [axis index] [Venus-1 command]
- [Venus-1 command]

Example 1:
A positioning command should be as followed. Move to the absolute position $x=10$, $y=20$ and $z=0.023$.

The Venus-1 syntax is

```
[value X] [value Y] [value Z] [Venus-1 command]
```

in this case:

```
Space (SP)

10  20  0.023  move
```

X  Y  Z  Venus-1 command
Example 2:
Disable y-axis:

```
[value] __ [axis index] __ [Venus-1 command] __
```

In this case:

```
0__2__setaxis__
```

Example 3:
To get the current position from the controller the command should be:

```
pos__
```

The returned value is the actual position. If the dimension is set to two then the line returned could be as

```
1.201__100.1234<CR><LF>
```
Venus-1 commands

(in functional order)

are described on the following pages, each command on a separate page.

In the upper right side the controller families are shown for which the command is available. If no family is shown then this command is available for all controllers.
Host / Terminal-Mode

mode

parameter: n 0,1
nv - storable

description: sets the operation mode of the controller
n = 1  Terminal Mode
n = 0  Host Mode

In the Host Mode all reply messages are terminated with the <CR><LF> character. All messages have to be requested by a command. In Host Mode no informations are sent unrequested to the host-computer.

In the Terminal mode a screen mask is send to the VT100 compatible ASCII terminal and unrequested informations are sent to the terminal. A VT100 terminal emulation can be used an a computer instead of a real terminal.

reply message: -

error code: 1003 parameter beyond the value range

example:

0 mode  enable Host Mode
Homing

calibrate (cal)

parameter:  -

description:  moves all enabled axes in direction to smaller position values (reverse) to the cal limit-switch. After detecting the limit-switch the origin (0.0) is set to that point.

reply message:  -

error code:  -

example:  
cal
  calibrate

During the cal command the controller does not execute any other command until the cal command finishes execution. Status inquiries are answered therefore only after the cal finished.

To prevent injuries and damages to persons and equipments the position of the cal-limit-switch has to be investigated after each power up of the controller.
rangemeasure (rm)

parameter: -

description: moves all enabled axes (setaxis = 1) in positive direction to the rm-switch (limit forward). After detecting the limit forward the maximum travel is determined.

reply message: -

error code: -

example: rm

During the rm command the controller does not execute any other command until the rm command finishes execution. Status inquiries are answered therefore only after the rm finished.
setcalvel

parameter:
\[ v \ t \quad v = \text{velocity in rev/s} \]
\[ t = 1 \text{ to switch} \]
\[ t = 2 \text{ from switch} \]

nv – storablen
description: defines the velocities for cal-move. The velocity must given in rev/s (based on a 200 step motor).
t=1 (to switch) means the higher velocity for searching the switch
t=2 (from switch) means the lower velocity for releasing the switch. This value should be slow (<0.5),

reply message: -

error code: 1001 wrong parameter type
1003 value range is crossed

example: 5 1 setcalvel
0.1 2 setcalvel
setrmvel

parameter: \( v \ t \)  
\( v = \) velocity in rev/s  
\( t = 1 \) to switch  
\( t = 2 \) from switch

nv – storable

description: defines the velocities for rm-move. The velocity must be given in rev/s (based on a 200 step motor).
  \( t=1 \) (to switch) means the higher velocity for searching the switch
  \( t=2 \) (from switch) means the lower velocity for releasing the switch. This value should be slow (<0.5),

reply message: -

error code:  
1001 wrong parameter type  
1003 value range is crossed

example:  
5 1 setrmvel  
0.1 2 setrmvel
Query : Position, Status and Error

pos (p)

parameter:  -

description:  returns the current position in the current coordinate system

reply message:  -

error code:  -

example:  pos
          12.5  6.77 2.55 ;returns the x y z position if the current dimension is 3
status (st)

parameter: no

description: returns the important information about the active condition of the controller for the host-communication. The value is returned as a numerical string. The assignment of the bits as a binary value is shown below.

<table>
<thead>
<tr>
<th>Bit #</th>
<th>decimal</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>D0</td>
<td>1</td>
<td>Ready to receive next command Move-complete</td>
</tr>
<tr>
<td>D1</td>
<td>2</td>
<td>Actual joystick condition</td>
</tr>
<tr>
<td>D2</td>
<td>4</td>
<td>User LED1</td>
</tr>
<tr>
<td>D3</td>
<td>8</td>
<td>Machine-error</td>
</tr>
<tr>
<td>D4</td>
<td>16</td>
<td>Speed-mode</td>
</tr>
<tr>
<td>D5</td>
<td>32</td>
<td>In-Position-Window</td>
</tr>
<tr>
<td>D6</td>
<td>64</td>
<td>setinfnc situation active</td>
</tr>
<tr>
<td>D7</td>
<td>128</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

D0

0 command/movement ready
1 command/movement in process

D1

0 joystick not active
1 joystick active

D2

0 User-Led1 Off (toggle with Key A)
1 User-Led1 On
<table>
<thead>
<tr>
<th>D4</th>
<th>0</th>
<th>controller in position-mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>controller in speed-mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D5</th>
<th>0</th>
<th>controller not in-position (clwindow)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>controller in-position</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D6</th>
<th>0</th>
<th>setinfnc situation is active (-axis enabled-led blinking)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

One or more bits of the reply message can be set. If it returns for example the value "6" (Binary: 0 1 1 0) after a status question that means the controller is ready for executing a new command, joystick mode is active, User-LED is switched off.

**Example:**

```
status or st 2
```

(Ready for receiving a new command, joystick mode active)
geterror (ge)

parameter: -

**description:** returns the last occured error code.
At the same time the error code is set to 0.

reply message: n error code

type: error code

example: geterror

getmerror (gme)

parameter: -

**description:** returns the last occured machine-error.
At the same time the error code is set to 0.

reply message: n error code

0 no error
1 overflow-FIFO
13 position-error (7. parameter clpara)
22 encoder-error

example: getmerror
**Joystick-Functions:**

**joystick (j)**

- **parameter:** \( n \in \{0, 1\} \)
- **description:** sets the mode for the joystick.
  - \( n = 0 \): Joystick-operation disabled
  - \( n = 1 \): Joystick-operation enabled

  **Caution:** The joystick must be connected while the controller is booting!

  The joystick is disabled automatically after any move-command!

- **reply message:** -
- **error code:** 1003 value range is exceeded
- **example:** 0 joystick disable manual operation
getjoyspeed

parameter: -

description: returns the current maximum joystick speed

reply message: r the actual maximum joystickspeed

type: -

description: determines the maximum joystick speed at full stroke of the joystick.

type: -

description: maximum speed is 20 mm/s if unit is mm

type: js

example: 20 js

maximum speed is 20 mm/s if unit is mm

(setunit=2)
getjoybspeed

parameter: -

description: returns the current maximum joystick speed if the joystick-button is pressed!

reply message: r the actual joystickspeed

error code: -

example: getjoybspeed

0.2

setjoybspeed

parameter: r

nv - storable

description: determines the maximum joystick speed at full stroke of the joystick, if button is pressed!

example:

20 js maximum speed is 20 mm/s if button pressed
2 joybspeed maximum speed is 2 mm/s
**getjoysticktype**

**parameter:**

- 

**description:** returns the current joystick type

**reply message:** i joystick type

**error code:** -

**example:**

getjoysticktype returns the joystick type

---

**setjoysticktype**

**parameter:**

i 0,2,3

nv - storage

**description:** defines the joysticktype, for obtaining an optimum conduct of the joystick or to select the appropriate hardware

**reply message:** -

**error code:** 1001 wrong value type

**example:**

2 setjoysticktype selects the joysticktype 2
Configuration

getpowerup

parameter: No
description: returns the actual setting of the powerup-function (setpowerup)
reply message: 0..3
description: returns the actual setting of the powerup-function
example:

getpowerup
2 -> calibrate stages joystick off

setpowerup

parameter: i 0..3
nv – storable
description: defines the actual setting of the powerup-function.
0 joystick off
1 joystick on (default)
2 calibrate, joystick off
3 calibrate, joystick on
reply message: no
example:
after powerup the controllers starts the calibrate function, joystick is off!
getdim

parameter: no

description: returns the actual setting of the dimension for the positioning-commands and replies. A two axes system has normally the dimension set to 2

reply message: 1..3

description: sets the dimension of the position for the commands with the parameters in[]. A two axes system has normally the dimension set to 2

event code: 1003 parameter beyond the value range

example:

3 setdim At now 3 parameters are needed for commands with parameters in[]. For example: 20.0 3.0 1.0 move

1 setdim only the x-position is needed and moved, for example: 13.1 move
getpdisplay

parameter: \( i = -1, 1..3 \) axisindex

description: returns the actual position-format (for the query position command pos)

reply message: \( w, p \) \( w \) = number of digits including decimal-point \( p \) = number of digits after decimal-point

example:

1 getpdisplay 10 4;

setpdisplay

parameter: \( w \) width \( p \) decimals \( i = -1, 1..3 \) axisindex

nv - storable

description: defines the actual position-format (query position). The width of the string is defined trough parameter \( w \), the number of digits after the decimal-point trough parameter \( p \)

error code: 1003 parameter beyond the value range

example:

10 3 1 setpdisplay p
12.23433.332
getlimit

parameter: no

description: returns the software limits for the programmed and joystick motion range. The number of return values depends on the actual setting of the dimension (setdim getdim)

reply message: 
x min x max     if dimension = 1
ymin ymax      if dimension = 2
zmin zmax      if dimension = 3

example:
2 setdim
getlimit

   0 54.345    travelrange x axis
   0 99.760    travelrange y axis

After switching on the controller the limits are undefined. After the cal-command and rm-command the limits are set to the moving area of the stage.
setlimit

parameter: [r] [s]

description: sets the software limits for the programmed and joystick motion range. The coordinate values of \( r \) always have to be smaller than the values of \( s \). The limits can not be chosen greater than the hard limits calculated by the commands \textit{cal} and \textit{rm}.

Note: If you define limits then the current position has to be inside of these limits. Otherwise the limit command has no effect.

error code: 1007
1015 current position beyond the allowed limits

example:
2 setdim
0 0 10 10 setlimit sets the limits of the process way in the current coordinate system and units to 0,0 for xmin and ymin and 10,10 for xmax and ymax
setpos (sp)

parameter: [r]

description: Shifts the position-counter to a new value. This command is normally used to define any position as the origin.

reply message: -
error code: -

example:
0 0 setpos the current position is the origin
p $\rightarrow 0 0$
scale

parameter: [r]

description: This command is normally used for a two-point-slope-correction. The parameter is not stored in the flash-memory, so please send the value after a reset! The new parameter is multiplied with the current value, and the resulting value is used. Be aware that your new parameter is not used! If you want to undo the parameter, send the reciprocal value of the parameter or send the reset-command!

reply message: -

error code: -

example:

the linear error of a 100 mm stage connected to the x-axis, is +4 µm. To compensate this we have to calculate the scale-parameter:

\[
\frac{100.000}{100.004} = 0.9999611\text{ scale}
\]
getaxis

parameter: \( i \quad i = \text{axis } -1,1,2,3 \)

description: Returns the status of selected axis.
Look at the command setaxis.

reply message: \( n \quad 0..3 \)

example:

2 getaxis returns the status for the y-axis.
-1 getaxis returns the status of all axes

setaxis

parameter: \( i \quad n \quad i = \text{parameter } 0..2 \)

\( n = \text{axis } 1..3 \)

nv – storabe

description: Sets the mode for the axis \( n \).

\( i = 0 \) The selected axis is disabled
\( i = 1 \) The selected axis is enabled
\( i = 2 \) The selected axis is enabled
cal and rm not active
\( i = 3..5 \) reserved

error code:

1001 wrong parameter type
1003 value range is crossed

example:

0 1 setaxis The x-axis is disabled
With the commands cal, rm and setpos the
position counter of the axis is set to zero and
the limits are changed.
1 1 setaxis The x-axis is enabled.
**getpitch**

**parameter:** 

\[ \text{i} \quad \text{i = axis} -1, 0, 1,2,3 \]

**description**

returns the pitch of the specified axis

**reply message**

1 .. 4 values

**error code:**
none

**example:**

1 getpitch 1 1 getpitch 1 1 1 1

**setpitch**

**parameter:** 

\[ \text{r} \quad \text{i} \quad \text{r = pitch in desired unit} \]
\[ \text{i = axis} 0,1,2,3 \]

**description**

The pitch defines the calculation factor between the rotation of the motor and the movement of the stage. In case of a linear stage it’s equivalent to the pitch of the screw.

The axis 0 is for the calculation of the velocity.

**error code:**
none

**example:**

2 0 setpitch
2 1 setpitch
1 2 setpitch
**getsw**

**parameter:**

i

i = axis -1,1,2,3

nv – storable

**description:**

Returns the actual setting of the limit switches

**reply message:**

n m

n = cal-switch (limit reverse)

m = rm-switch (limit forward)

**error code:**

1003 Value range of the parameter is exceeded

**example:**

1 getsw

**setsw**

**parameter:**

i

i = parameter 0..2

j = switch

n = axis 1..3

nv – storable

**description:**

sets the mode for the limits.

i = 0 normal open

i = 1 normal closed

i = 2 ignore limit

j = 0 cal – sw (limit reverse)

j = 1 rm – sw (limit forward)

**error code:**

1001 wrong parameter type

1003 value range is crossed

**example:**

1 0 1 setsw cal – sw axis 1 as normal closed

2 1 1 setaxis ignore the rm – sw
getswst

**parameter:**

\[ i \quad i = \text{axis} -1,1,2,3 \]

\[ \text{nv – storable} \]

**description:**

Returns the actual status of the limit switches

**reply message:**

\[ n \quad m \]

\[ n = \text{cal-switch (limit reverse)} \]

\[ m = \text{rm-switch (limit forward)} \]

**error code:**

1003 Value range of the parameter is exceeded

**example:**

1 getswst return :1 1 both limit-sw active

getunit

**parameter:**

\[ i \quad \text{axis} 0..3 \]

**description:**

returns the unit of the axis \( i \)

**reply message:**

\[ n \]

code see setunit

**error code:**

1003 parameter beyond the value range

**example:**

getunit
setunit

parameter: \( i \ n \)
\[ i = \text{unit } 0..7 \]
\[ n = \text{axis } 0..2 \]

\( nv \) - storable

**description:** sets the physical units of an axis.

<table>
<thead>
<tr>
<th>( i )-value</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>microsteps only SMC-series</td>
</tr>
<tr>
<td>1</td>
<td>( \mu )m</td>
</tr>
<tr>
<td>2</td>
<td>mm</td>
</tr>
<tr>
<td>3</td>
<td>cm</td>
</tr>
<tr>
<td>4</td>
<td>( m )</td>
</tr>
<tr>
<td>5</td>
<td>inch</td>
</tr>
<tr>
<td>6</td>
<td>mil (1/1000 inch)</td>
</tr>
</tbody>
</table>

\( n = 0 \) defines the unit of the velocity axis divided by a second. I.e. 2 0 setunit sets the unit for the velocity to mm/s.

\( n = 1..3 \) defines the x, y and z axis.

**reply message:** -

**error code:** 1003 parameter beyond the value range

**example:**

2 1 setunit  the position of the x-axis is defined in units of mm
1 0 setunit  the velocity is defined in \( \mu \)m/sec
getvel (gv)

parameter: -

description: returns the last maximum speed value set by the setvel command.

reply message: r

eample:

getvel returns the maximum speed value in the current units

setvel (sv)

parameter: n
nv – storable

description: sets the maximum velocity of the fastest axis (the axis which covers the longer way during a move). If the same way is covered as well in x as in y, then the resulting velocity (the velocity vector length is maximum velocity multiplied by 1.414. The maximum value is 45 U/sec and the minimum value 15 nm/s). If the parameter is negative, then it will be converted into a positive value.

error code: -

eexample:

10.2 setvel sets the velocity value to 10.2 in the current units.
getaccel (ga)

parameter: -

description: returns the maximum acceleration value sent with setaccel or sa in the current units

reply message: n

declaration: N

description: sets the maximum acceleration of the fastest axis at the programmed move. Too large or too small values are limited to the maximum possible values, without giving an error message. The maximal usable value depends on the hardware (i.e. stage, motors)

reply message: -

ever code: -

example:

23.4 setaccel sets the acceleration to 23.4 in the current units.

For standard application the linear acceleration function is used. For application where high acceleration is needed but the mechanical
system has a high resonance frequency, then the sin² function reduce
the amplitude of the resonance significant and the positioning is done
in a shorter time due to a shorter time of mechanical vibration.

For both acceleration function the time of acceleration is the same.
Therefore the maximum of acceleration is in the middle of the
acceleration time higher for the sin² function.

For some mechanical systems it is better to reduce the acceleration to
get a shorter positioning time.

**getaccelfunc**

**parameter:** no

**description:** this command returns the selected acceleration
function

**reply message:** 0 or 1

**error code:** -

**example:**

```
getaccelfunc returns 1 for the sin² acceleration function
```
setaccelfunc

parameter: n
0 = linear acceleration function
1 = \(\sin^2\) acceleration function

nv storable

description: with this command the acceleration function linear or \(\sin^2\) is selected. The selected function work at acceleration and deceleration.

error code: 1003
1008

example:
1 setaccelfunc selects the acceleration function \(\sin^2\)
Positioning-Commands:

move (m)

parameter: 
[r]

description: moves the axes to the absolute position [r]. If the position is beyond the process way, the maximum or minimum position of the appropriate axis is automatically used. It is not possible to move beyond the limits.

reply message: -

table:

<table>
<thead>
<tr>
<th>error code</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1004</td>
<td>travel range should be exceeded</td>
</tr>
</tbody>
</table>

description of example:

12 4 move  moves the x-axis to the absolute position 12, the y-axis to the absolute position 4. (in the current units)
The dimension here is 2.

2 4.3 4.55 move  moves the x-axis to the absolute position 12, the y-axis to the absolute position 4.3 and the z-axis to position 4.55 (in the current units)
The dimension here is 3.
rmove (r)

parameter: [r]

description: moves the axes to the relative position [r]. The absolute position is the current position add by the parameter position[r]. If the absolute position is beyond the process way the maximum or minimum position of the appropriate axis is automatically used. It is not possible to move beyond the process limits.

reply message: -

error code: 1004 process way should be crossed

example:
18 4 82 r

Ctrl C

parameter: no

description: the key combination „Ctrl“ „C“ aborts the current movement. The key combination often is shown as ^C. On some keyboards the key "Ctrl" is also signed with "Strg" on German-keyboard for example.

The ASCII-code is 3!

reply message: no

example:

Ctrl C
Velocity-Mode:

speed

parameter: \( m = \) velocity value
\( n = \) axis
\( m = -45 \) to \(+45\) (rev/s)
\( n = 1, 2, 3\) (x,y,z)

description: initiates a continues move of a determined axis. The move can be stopped with the velocity value 0 or stopspeed. With 0 velocity the speed mode is furthermore active. After the stopspeed command the standard positioning mode is active.

Allowed commands during speed move: status, speed, stopspeed, Ctrl C Use only these commands!

Other commands do lock the command input of the controller. If the command input is locked only Ctrl C can unlock the input.
speed cannot initiate an endless move. When speed mode is active the status reply is 17. The speed velocity unit is revolutions per second

reply message: -
stopspeed

parameter:  no

description:  disables speed mode
With stopspeed the standard positioning mode is active again.

Notice:
Status reply during speedmode is 17.
In standard positioning mode see Venus-Command status.

reply message:  no

example:

stopspeed  speed mode is finished
Parameter

restore

parameter: -

description: overwrites all parameters with the last saved values (see save command).

reply message: -

error code: 1202 Flash memory read error

example: 1202

restore

save

parameter: -

description: saves all system adjustments (configuration) in the nonvolatile memory. This operation takes some seconds and it must not be interrupted by turning power off. With Restore these values are read back. While switching on the controller a „restore“ command is automatically done.

reply message: -

error code: 1200 write error in the flash memory
           1201 erase error in the flash memory

example: 1200 1201

save
Identification of the controller:

**identify**

**parameter:** -

**description:** returns the identification number of the controller. To this belongs: Configuration switch and Firmware-Revision.

**reply message:** 
- $i$ = Controller ident.  
- $j$ = hardware revision number  
- $k$ = Firmware rev.  
- $l$ = options (hex.)  
- $m$ = switch SW1 (hexadecimal)

**error code:** -

**example:**

*identify* returns: model: nnnn  HW-rev: xx  SW-rev: xx  
Board-SW: xx  FP SW: xx  
Board-SW is the hexadecimal representation of options  
FP is the hexadecimal value of the switch SW1  
(=3d if 9600 Baud and Venus-1 is focused)
version

parameter: -

description: returns the version number of the Venus interpreter

reply message: x valid version at the moment

error code: -

getserialno

parameter: -

description: returns the serial-number of the controller

reply message: the number

error code: -

getserialno
Stack-Commands

pop

parameter: -

description: removes the last element from the stack

reply message: -

error code: 1008 no element is on the stack

example:

pop

gsp

parameter: -

description: returns the number of elements on the stack. This value is also called stackpointer, because it works as an index in the stack memory.

reply message: -

error code: -

example:

gsp
**clear**

**parameter:** -

**description:** removes all elements from the stack

**reply message:** -

**error code:** -

**example:**

```
clear
```

stackpointer is set to 0, after that there are no more parameters on the stack.
Setting of the motor-current:

**setumotmin**

**parameter:** \( v_i \) \( i = -1, 1..3 \) axis number \( i \)

and value \( v \)

**nv** - storable

**description:** defines the current for the motor of the selected axis \( i \) at no motion (\( v \) in internal units)

This command should only be used in agreement with the manufacturer.

**reply message:** -

**error code:** 1001,1003

**example:**

3200 1 setumotmin sets the value 3200 (ZZS-42 1.2A) for the axis X
getumotmin

parameter: -1, 1.. 3 axis number or all axes if -1

description: return the value set with the command setumotmin

reply message: for example 3200

error code: 1001, 1003

example:
1 getumotmin return the value for the X-axis

Do not modify these values without consulting the factory!
setumotgrad

parameter:  \( v \) \( i \) \( i = -1,1..3 \)

axis number \( i \)
and value \( v \)

\( nv \) - storable

description:
defines the enhancement of the current in dependency of the velocity for the motor of the selected axis. (internal units)

This command should only be used in agreement with the manufacturer.

replay message:  -

error code  1001,1003

example:

40 1 setumotgrad for ZSS42-1.2A

Do not modify these values without consulting the factory!
getumotgrad

parameter:          -1,1..3   axis number 1 or 2 or all axes (-1)

description:       returns the value set by the command
                   setumotgrad

reply message:     i.e. 40

example:
                   1 getumotgrad returns the value for the x-axis

Settings for mainly used motors:

<table>
<thead>
<tr>
<th>Motortype</th>
<th>umotmin</th>
<th>umotgrad</th>
</tr>
</thead>
<tbody>
<tr>
<td>4H4018</td>
<td>2000..4000</td>
<td>40..60</td>
</tr>
<tr>
<td>ZSS-19.200-1.2 parallel</td>
<td>1500..1900</td>
<td>4..5</td>
</tr>
<tr>
<td>ZSS-25.200-1.2 parallel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZSS-32.200-1.2 parallel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZSS-42.200-1.2 parallel</td>
<td>2500..4200</td>
<td>40..60</td>
</tr>
<tr>
<td>ZSS-43.200-1.2 parallel</td>
<td>3000..5000</td>
<td>60..80</td>
</tr>
<tr>
<td>ZSS-52-200-2.5 parallel</td>
<td>3000-4000</td>
<td>50..70</td>
</tr>
<tr>
<td>ZSS-52-200-2.5 serial</td>
<td>4200-5200</td>
<td>120..130</td>
</tr>
<tr>
<td>ZSS-57-200-2.5 parallel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZSS-57-200-2.5 serial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZSH-57-200-2.8 parallel</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The result of umotmin is very easy to display with command gi!
The measured value is an effective-current in the motor!

It's always good to use the smaller values if the performance is good.
If higher torque is necessary please use the higher values!
**setmp**

**parameter:**

\[ v \; i \; i = 1..3 \]

\[ v = 0 = \text{motor off} \quad 1 = \text{motor on} \]

\[ i = \text{axisnumber} \quad 1..3 \]

**nv** - storable

**description:**

power-on /off the motorpower

**replay message:**

- 

**example:**

0 1 setmp  power axis 1 = off

**getmp**

**parameter:**

\[ i \; i = \text{axisnumber} \quad 1..3 \]

**nv** - storable

**description:**

returns the status of the motorpower

**replay message:**

- 

**example:**

1 getmp  power axis 1 = off
gi

parameter: i i = axisnumber 1..3

nv - storable

description: returns the actual motorcurrent in Amp (rms)

replay message: -

example:
1 gi
1.2 actual current = 1.2 A
Additional Commands

Special Output:

**setsout**

parameter: \( v \) \( i \) \( v = value \) \( 0..1 \)
\( i = output \# \) \( 1 \)

nv - storable

description: switch on or off the special output
24 V 0.8 A for motorbrake

replay message: -

example:
1 1 setsout output = on
0 1 setsout output = off

**getsout**

parameter: \( i \) \( i = output \# \) \( 1 \)

nv - storable

description: returns the status of the special output

replay message: -

example:
1 getsout if output = on value = 1
output = off value = 0
Encoder closed-loop:

getcloop

parameter: i, i = axis -1..3

description: returns the actual setting of the closed-loop.

reply message: 0 – closed-loop off
               1 - closed-loop on

example:
-1 getcloop returns the closed-loop status of all axis
2 getcloop returns the closed-loop status of axis 2
setcloop

parameter: \( p \ i \)  
- \( p = 0,1 \)
- \( i = -1,1...3 \)

\( p = 0 \) disable closed loop
\( p = 1 \) enabled closed loop
\( i = \) axis number

nv – storable

description: Enables or disabled the closed-loop function
open-loop \( \rightarrow \) standard stepper control
closed-loop\( \rightarrow \) advanced control by a linear or
rotary encoder-system.

error code: 1001,1003

remarks: After enabling a axis for closed loop, a
calibration must be done with the cal
command

description: Enables or disabled the closed-loop function
open-loop \( \rightarrow \) standard stepper control
closed-loop\( \rightarrow \) advanced control by a linear or
rotary encoder-system.

example: 0 1 setcloop disables closed loop for the X-axis.
**getclfactor**

**parameter:**  
i  
i = axis {–1,1,2,3}

**description:**  
returns the resolution of the measurement-system. For evaluation of the correct value, please read section common-hints!

**reply message:**  
the clfactor

**error code:**  
none

**example:**  
1 getclfactor 500  
-1 getclfactor

**setclfactor**

**parameter:**  
r i  
r = 1...72000  
i = axis {-1,1..3}

**nv - storable**

**description:**  
defines the resolution of the measurement-system.  
If with one motorrevolution the travel is 1 mm the factor is the number of increments coming from the encoder divided with 4.  
10000 inkrements (100 nm) factor= 2500

**reply message:**  
-

**error code:**  
1002, 1003

**example:**  
2500 1 setclfactor
getnselpos

**parameter:** 1..3 axisnumber

**description:** this command returns the source of the position-reply of the controller.

**reply message:** 0 or 1

**error code:** -

**example:**
1 getnselpos

setnselpos

**nv storable**

**parameter:**

- n 0 = calculated position
- 1 = measured encoderposition
- i 1..3 :axisnumber

**description:** this command determines which position will be displayed with the command `pos (p)`
This command is only useful with the encoder-option.

**error code:** 1001

**example:**
1 1 setnselpos the encoder position will be displayed
Commands for the refmove:

With the refmove the accuracy of the homing of an axis will be increased to an accuracy of +/- 1 increment of the encoder. The command is available if:
- option closed-loop available
- the encoder or linear scale is fitted with an index mark

Normally rotary encoders have one index mark each revolution. Linear encoders have sometimes one index mark in the middle of the travel or sometimes in steps of 25 or 50 mm.

Homing-Sequence: First run the cal-command for the coarse-home, then run the refmove command. If available, the stage will stop at the index-mark! If you know approximately where the index-mark is, it could be faster to send a move command, short before the index-mark, then the refmove.

If option available the controller is configured completely, so you are able to run the refmove command without any configuration!
getref

parameter: n -1; 1..3

description: returns the value, if the axis should take part with the refmove.

n = 0 : active high signal for index-mark
n = 1 : active low signal for index-mark
n = 2 : index-mark not available

error code: 1001, 1003, 1008

example
-1 getref 1 2 2
1 getref 1

setref

parameter:

n n = 0...2
i i = axis 1,2,3

nv - storable
description: defines if the axis should take part with the refmove.

n = 0 : active high signal for index-mark
n = 1 : active low signal for index-mark
n = 2 : index-mark not available

reply message: -

error code: 1001, 1003, 1008, 2101

example
1 1 setref activ low for axis 1
**getrefst**

**parameter:**

\[ i \quad i = \text{axis } 1,2,3 \]

**description:**

returns the result of an refmove
The value is bit-coded

- bit 0 = 0 : index-mark found
  - = 1 : index-mark not found
- bit 1 = 1 : limit switch active
- bit 2 = actual status of the index-input
  - if signal active-low : 1 = input low
  - 0 = input high
  - if signal active-high: 1 = input high
  - 0 = input low

For test of a successful refmove please use only bit 0! Bit 2 is only for internal use.

**error code:**

- 1001,
- 1003,1008,2101

**example**

1 getrefst 0 = index found
getrefvel

parameter:

description: returns the velocities for refmove.

error code:

example:

getrefvel 5
0.01

setrefvel

parameter: v t
v = velocity in rev/s
 t = 1 fast velocity for course search
 t= 2 slow velocity for fine search

nv – storable
description: defines the velocities for refmove
The velocity must given in rev/s (based on a 200 step motor).
t=1 means the higher velocity for coarse search of the index-mark
t=2 means the lower velocity for finding the edge of the index-mark.

error code: 1001 wrong parameter type
1003 value range is crossed

example:

5 1 setrefvel
0.01 2 setrefvel

refmove
**parameter:**  
* i  
number of revolutions

**description:**  
starts the search for the index-mark of the encoder or linear scale  
If index occurs stages stops at the edge!

**error code:**  
-

**example:**  
50 refmove  
seaches for maximum 50 revolutions
Ethernet-Interface

**getipadr**

**parameter:** No

**description:** returns the actual IP-address of the controller

**reply message:** the address

**error code:**

**example:**
getipadr

**setipadr**

**parameter:** n = IP-address of the controller

**description:** defines the IP-address of the controller

**reply message:** -

**error code:** 1003

**example:**
102 168 100 001 setipadr
getmacadr

**parameter:**

**description:** returns the actual MAC address of the controller

**reply message:** the address

**error code:**

**example:**

getmacadr
## Error messages:

<table>
<thead>
<tr>
<th>Error code</th>
<th>Error description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1..3</td>
<td>Internal error = Hardware problem</td>
</tr>
<tr>
<td>1001</td>
<td>Wrong parameter type</td>
</tr>
<tr>
<td>1002</td>
<td>Not enough parameters for the command on the stack</td>
</tr>
<tr>
<td>1003</td>
<td>Value range of the parameter is crossed</td>
</tr>
<tr>
<td>1004</td>
<td>Process range should be crossed</td>
</tr>
<tr>
<td>1005</td>
<td>Reserved</td>
</tr>
<tr>
<td>1006</td>
<td>Wrong parameter value at the restore Operation detected</td>
</tr>
<tr>
<td>1007</td>
<td>Invalid parameter value</td>
</tr>
<tr>
<td>1008</td>
<td>Not enough parameter on the stack</td>
</tr>
<tr>
<td>1009</td>
<td>Not enough room on the stack for all parameters</td>
</tr>
<tr>
<td>1010</td>
<td>No more available memory</td>
</tr>
<tr>
<td>1011</td>
<td>No chain selected</td>
</tr>
<tr>
<td>1012</td>
<td>No next position in the selected chain</td>
</tr>
<tr>
<td>1013</td>
<td>No position in the current chain available</td>
</tr>
<tr>
<td>1014</td>
<td>Index in the chain not available</td>
</tr>
<tr>
<td>1015</td>
<td>Parameter values out of the possible limits</td>
</tr>
<tr>
<td>1016</td>
<td>In the chain the least index is crossed</td>
</tr>
<tr>
<td>1017</td>
<td>In the chain the greatest index is crossed</td>
</tr>
<tr>
<td>1018</td>
<td></td>
</tr>
<tr>
<td>1100</td>
<td>Division through zero</td>
</tr>
<tr>
<td>Year</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| 2001 | undefined command  
  (only SMC series with closed loop) |
| 2011 | Step error in x-axis appeared |
| 2012 | Step error in y-axis appeared |
| 2014 | Step error in z-axis appeared |
| 2013 | Step error in x and y-axis appeared |
| 2015 | Step error in x and z-axis appeared |
| 2016 | Step error in y and z-axis appeared |
| 2017 | Step error in x, y and z-axis appeared |
| 4000 | self test error appeared (only SMC series) |
| 7001 | I/O Port not present (only SMC series with I/O option) |
| 7002 | mode = 1 only for the serial line allowed (SMC series only |
Venus-1 smc-corvus

For your notices: