SMC VIs

Content

1 General

This manual is applicable when the Tango controller is switched to the VENUS command set. These VIs have been created using LabVIEW 8.6 and will not run with older versions of LabVIEW.

The communication with the controller SMC is done by sending and receiving ASCII strings, this command language is called VENUS language. Depending on the controller, VENUS 1 or VENUS 2 is used. For details about the VENUS language please read the corresponding documentation.

These VIs encapsulate the most important VENUS commands like movement commands, position and status interrogation, velocity, or finding limit switches and index marks. The more low level VENUS commands for configuration of the controller are not implemented because the controller already is configured for your mechanics. If you have problems, please contact Märzhäuser GmbH.

The SMC VIs itself use another VI called SMC_Manager, this SMC_Manager stores information like the number of axes. It also stores the last acquired position and status, so if you need these information you can get it directly from the SMC_Manager. It also allows a simulation mode which is useful for writing software without a controller connected.

The SMC VIs also use a VI named Communication Manager, which itself does all the communication with the various communication channels (RS232, USB, PCI and PCI-Express).

2 SMC VIs

2.1 Initialization

2.1.1 SMC_Init

Initialize the communication with the controller and prepare the SMC_Manager.

Input Parameters:

- Comm Mode: select always RS232 for Tango using USB, PCI or PCIe interfaces
- Com Port: use 1 for COM1 and so on
- Baud Rate: use 57600 for all physical Tango interfaces. The installed Tango driver will handle the correct baud rate automatically.
- Number of Axis: use 2 or 3 depending on your order option
- VENUS language: select VENUS 1
- Simulation: use true if you want to work without a connected Tango controller
- Error

Output Parameters:

- Error

Remark:
This VI calls the VIs Communication Manager to open the communication, the SMC_Manager, and SMC_SetDim.

2.1.2 SMC_Close

Closes the communication with the controller.

Remark:
This VI calls the VIs Communication Manager to close the communication with the controller.

2.2 Interrogation

2.2.1 SMC_GetDim

Get the number of axes (dimension) which the SMC controller uses by sending the VENUS command getdim. For VENUS 2 the result is always 1.

Input Parameters:

- Error

Output Parameter:

- Error
- Dimension of the SMC controller (number of axis)

2.2.2 SMC_GetPos

Get the actual position by sending the VENUS command pos or np.

Input Parameters:

- Error

Output Parameter:

- Error
- Answer string from the SMC

Remark:
This VI calls the VI SMC_Manager with the mode SetPostion so that the SMC_Manager stores the actual position (as array of double).

2.2.3 SMC_GetSt

Get the actual status of the controller by sending the VENUS command st or nst.

Input Parameters:

- Error

Output Parameter:

- Error
- Answer string from the SMC

Remark:
This VI calls the VI SMC_Manager with the mode SetStatus so that the SMC_Manager stores the actual status and moving mode.

2.2.4 SMC_GetSwSt

Get the condition of the limit switches (touched or not touched) by sending the VENUS command getswst.

Input Parameters:
- Error

Output Parameter:
- Error
- Switches min: array of boolean with the information about the backward limit switches
- Switches max: array of boolean with the information about the forward limit switches

2.2.5 SMC_GetVel

Get the velocity setting by sending the VENUS command gv or gnv.

Input Parameters:
- Error

Output Parameter:
- Error
- Velocity

2.2.6 SMC_GetAcc

Get the acceleration setting by sending the VENUS command ga or gna.

Input Parameters:
- Error

Output Parameter:
- Error
- Acceleration

2.2.7 SMC_GetLimit

Get the possible travel range by sending the VENUS command getlimit or getnlimit.

Input Parameters:
- Error

Output Parameter:
- Error
- MinValues: array of double with the minimum travel ranges
- MaxValues: array of double with the maximum travel ranges

2.2.8 SMC_GetError

Get the last occurred error of the SMC controller by sending the VENUS command ge or gne.
0 means no error.

Input Parameters:

- Error

Output Parameter:

- Error
- Error Number
- Error String

Remark:
The Error String is available in English and in German, the language you can select in the block diagram.

2.2.9 SMC_GetAI

Get the value of the analogue input.

Input Parameters:

- Error

Output Parameter:

- Error
- Value from the analogue input
- Average value

Remark:
The analogue input is a special option and not available for all controllers. Also included is an averaging of the measurement value.

2.3 Finding switches and index marks

2.3.1 SMC_Calibrate

Find the backward limit switches and set the position to zero by sending the VENUS command cal or ncal.

Input Parameters:
- Error

Output Parameter:
- Error

Remark:
For some controllers (e.g. SMC Corvus) this VI exits only after the movement has been finished. For other controllers (e.g. SMC Pollux) it returns immediately.

2.3.2 SMC_RangeMeasure

Find the forward limit switches by sending the VENUS command rm or nrm.

Input Parameters:
- Error

Output Parameter:
- Error

Remark:
For some controllers this VI exits only after the movement has been finished.

2.3.3 SMC_RefMove

Find the index marks of the encoders by sending the VENUS command refmove or nrefmove.

Input Parameters:
- Error
- Number of revolutions

Output Parameter:
- Error

Remark:
This is only possible with the corresponding order option. This VI exits only after the movement has been finished.

2.4 Movement

2.4.1 SMC_AxisAbs

Move one axis absolute by sending the VENUS command m or nm.

Input Parameters:
- Error
- Axis number
- New position
- WaitUntilReady

Output Parameter:
- Error

Remark:
Before starting the movement, the moving mode in the SMC_Manager is set to true. Depending on the input WaitUntilReady, the VI returns immediately or waits until the end of the movement. The input Axis number is used only for VENUS 1, it is ignored for VENUS 2.

2.4.2 SMC_AxisRel

Move one axis relative by sending the VENUS command r or nr.

Input Parameters:
- Error
- Axis number
- Distance
- WaitUntilReady

Output Parameter:
- Error

Remark:
Before starting the movement, the moving mode in the SMC_Manager is set to true. Depending on the input WaitUntilReady, the VI returns immediately or waits until the end of the movement. The input Axis number is used only for VENUS 1, it is ignored for VENUS 2.

2.4.3 SMC_MoveAbs

Move all axes absolute by sending the VENUS command m or nm.

Input Parameters:

- Error
- Array of double with the new positions
- WaitUntilReady

Output Parameter:

- Error

Remark:
Before starting the movement, the moving mode in the SMC_Manager is set to true. Depending on the input WaitUntilReady, the VI returns immediately or waits until the end of the movement.

2.4.4 SMC_MoveRel

Move all axes relative by sending the VENUS command r or nr.

Input Parameters:

- Error
- Array of double with the distances
- WaitUntilReady

Output Parameter:

- Error

Remark:
Before starting the movement, the moving mode in the SMC_Manager is set to true. Depending on the input WaitUntilReady, the VI returns immediately or waits until the end of the movement.

2.4.5 SMC_Stop

Stop all movements by sending CTRL C (ASCII #3).

Input Parameters:

- Error

Output Parameter:

- Error

2.5 Configuration

2.5.1 SMC_SetDim

Sets the dimension (number of axes) the controller uses by sending the VENUS command setdim. For VENUS 2, this VI does not send a command to the controller.

Input Parameters:

- Error
- Number of dimensions

Output Parameter:

- Error

Remark:
You can configure a three axis controller to act like a two axis controller, but you cannot use a two axis controller as a three axis controller. This is an extra order option.

2.5.2 SMC_SetVel

Sets the velocity by sending the VENUS command sv or snv.

Input Parameters:

- Error
- Velocity value

Output Parameter:

- Error

2.5.3 SMC_SetAcc

Sets the acceleration by sending the VENUS command sa or sna.

Input Parameters:

- Error
- Acceleration value

Output Parameter:

- Error

2.6 Miscellaneous

2.6.1 SMC_SetZero

Sets the coordinates to zero by sending the VENUS command setpos or setnpos.

Input Parameters:

- Error

Output Parameter:

- Error

2.6.2 SMC_SendCommand

Sends directly a VENUS command to the controller.

Input Parameters:

- Error
- VENUS command

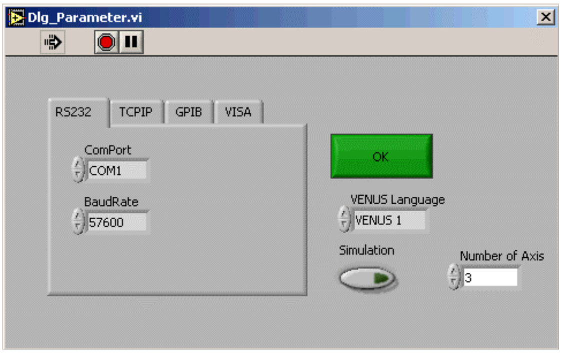
Output Parameter:

- Error
- Answer string

Remark:
This VI expects one line as answer. To avoid a timeout error if you send a command which gives no answer back, a " st" or " nst" command is automatically added. If the answer consists of more than one line, only the first one is given back, the other lines are ignored.
If you want to send a command which gives back an answer consisting of more than one line, you should do it in a way like in the SMC_GetLimit VI.

3 TestProgram

TestProgram is a simple demo program which shows the use of the SMC VIs. In the first frame in the block diagram you can enter the communication parameters and the number of axis manually, or you can use a dialog box at every start of the program.

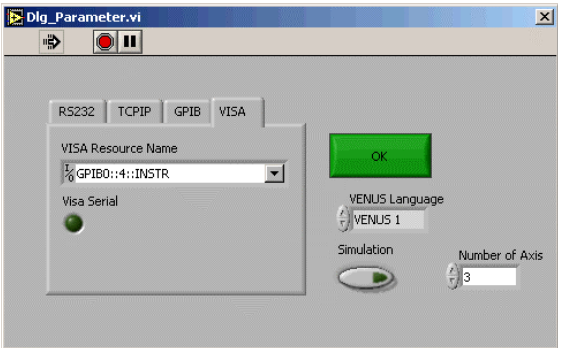


ComPort COM1

BaudRate 57600

VENUS Language VENUS 1

Number of Axis 3



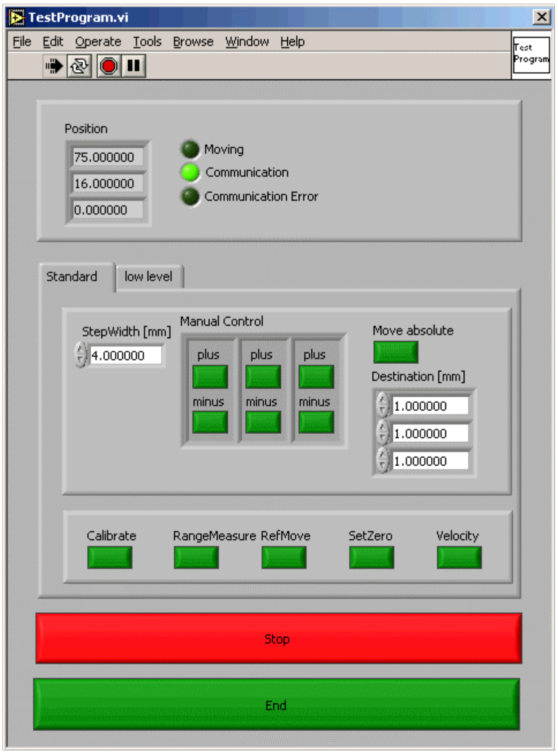
VISA Resource Name GPIB0::4::INSTR

VENUS Language VENUS 1

Number of Axis 3

Dialog box with RS232 parameters

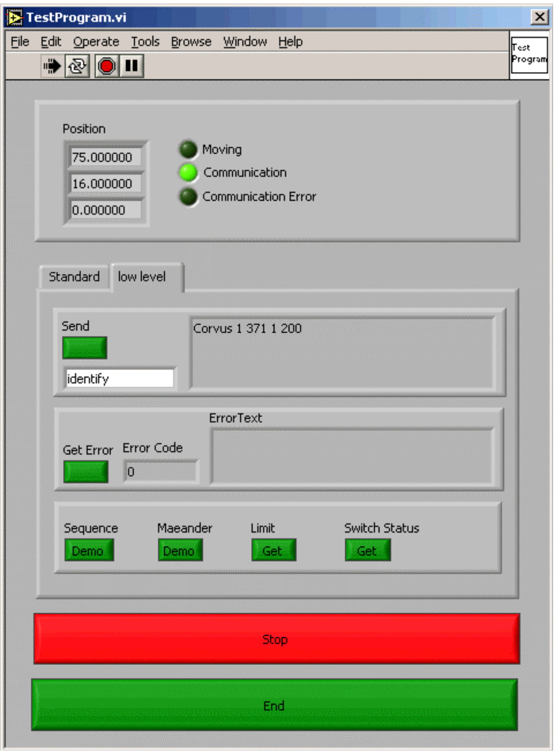
Dialog box with VISA parameters



Position 75.000000, 16.000000, 0.000000

StepWidth [mm] 4.000000

Destination [mm] 1.000000, 1.000000, 1.000000



Position 75.000000, 16.000000, 0.000000

Send: Corvus 1 371 1 200

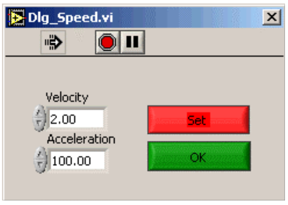
identify

Error Code 0

Main program, standard page

Main program, low level page

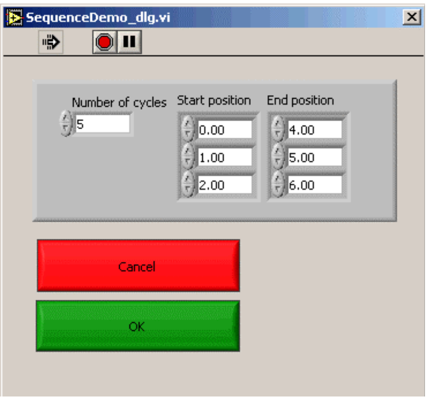
In the main program, you can start relative movements with the corresponding “plus” and “minus” buttons and absolute movements with the “Move Absolute” button. “Calibrate” looks for the backward limit switches, “RangeMeasure” looks for the forward limit switches. If available, “RefMove” looks for the index mark of an optional encoder. “Velocity” opens a dialog box which shows and sets the movement velocity and acceleration.



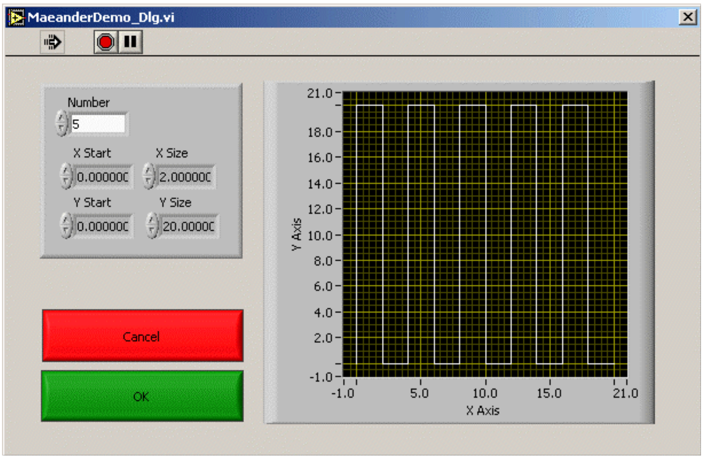
Velocity: 2.00
Acceleration: 100.00

Dialog box for setting the movement velocity and acceleration

In the low level page, you can directly send VENUS commands like “pos” or “identify”. Please be careful, here you have full access to all parameters of the controller. Also in the low level page, you can start some movement demos.



Number of cycles	Start position	End position
5	0.00	4.00
	1.00	5.00
	2.00	6.00



Number: 5
X Start: 0.000000
X Size: 2.000000
Y Start: 0.000000
Y Size: 20.00000

Dialog boxes for the movement demos “Sequence” and “Maeander”

The second frame in the block diagram consists of two loops. The upper one reacts on the buttons, or if you don’t press a button, it queries the actual position and status which are stored in the SMC_Manager and also written in two global variables. The lower loop runs always and has two jobs: it checks the Stop button and updates the position and moving display from the global variables.

The global variables are used for the display only (which is updated in the lower loop of the test program). Since the three VIs SMC_Calibrate, SMC_RangeMeasure and SMC_RefMove do return only after the movement has been finished, the global variable Moving is set manually in the test program.

The advantage of the concept of the SMC_Manager who stores the actual position and moving status is that if you need these informations in your application, you get these informations much faster from the SMC_Manager than from the controller itself.