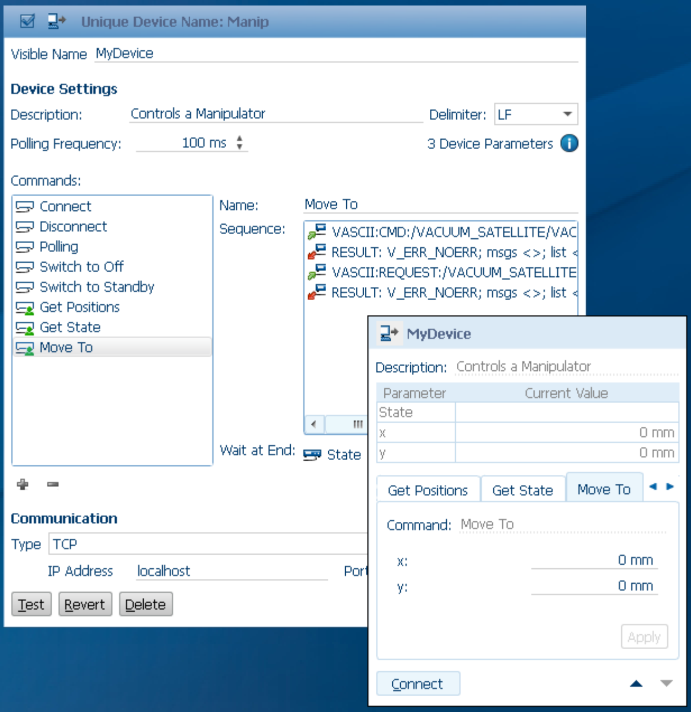
SpecsLab Prodigy

Configuration and Use of Remote Out Devices



Unique Device Name: Manip

Visible Name: MyDevice

Device Settings

Description: Controls a Manipulator

Delimiter: LF

Polling Frequency: 100 ms

3 Device Parameters

Commands:

- Connect
- Disconnect
- Polling
- Switch to Off
- Switch to Standby
- Get Positions
- Get State
- Move To

Name: Move To

Sequence:

```
VASCII:CMD:/VACUUM_SATELLITE/VAC
RESULT: V_ERR_NOERR; msgs <>; list <
VASCII:REQUEST:/VACUUM_SATELLITE
RESULT: V_ERR_NOERR; msgs <>; list <
```

Wait at End: State

Communication

Type: TCP

IP Address: localhost

MyDevice

Description: Controls a Manipulator

Parameter	Current Value
State	
x	0 mm
y	0 mm

Command: Move To

x: 0 mm

y: 0 mm

User Manual 4.106.0-r114022 | August 28, 2023

A member of SPECSGROUP

SPECS Surface Nano Analysis GmbH
Voltastrasse 5
13355 Berlin
Germany
Tel: +49 30 46 78 24-0
Fax: +49 30 46 42 08-3
Email: support@specs.com
Web: http://www.specs-group.com

SPECS User Manual
Remote Out Devices—Configuration and Use of Remote Out Devices
SpecsLab Prodigy Version 4.106.0-r114022

August 28, 2023

SPECS order number reference: N/A

Table of Contents

This page intentionally left blank.

Chapter 1 – Remote Out Devices

The Remote Out device feature allows you to run scripts and control devices that are not directly supported by SPECS. It uses a text-based protocol to send device commands over TCP, UDP and serial ports or to executables via standard input / output. Any instrument which can be controlled in this way can be adapted for use in SpecsLab Prodigy.

Since the aim is for a close integration of such devices into SpecsLab Prodigy, adding and using devices is performed using the standard applications:

- SpecsLab Prodigy Configuration Tool for adding and configuring the device.
- SpecsLab Prodigy for using the device.

This manual describes the special features in the SpecsLab software for adding additional devices. A certain amount of familiarity with the SpecsLab Prodigy Configuration Tool and SpecsLab Prodigy is expected—you should feel comfortable with the procedures covered in the SpecsLab Prodigy Quick Guide before reading this manual.

You are also expected to have experience at programming your device over a network or serial connection. The facilities offered by SpecsLab Prodigy essentially allow you to trigger standard commands to your device, while adding some additional features for greater interaction through the SpecsLab Prodigy interface.

Programming examples are presented at the end of this manual to help you write programs or scripts, which can be executed via the remote interface. This way arbitrary actions or communication with devices can be triggered if more complex protocols have to be implemented.

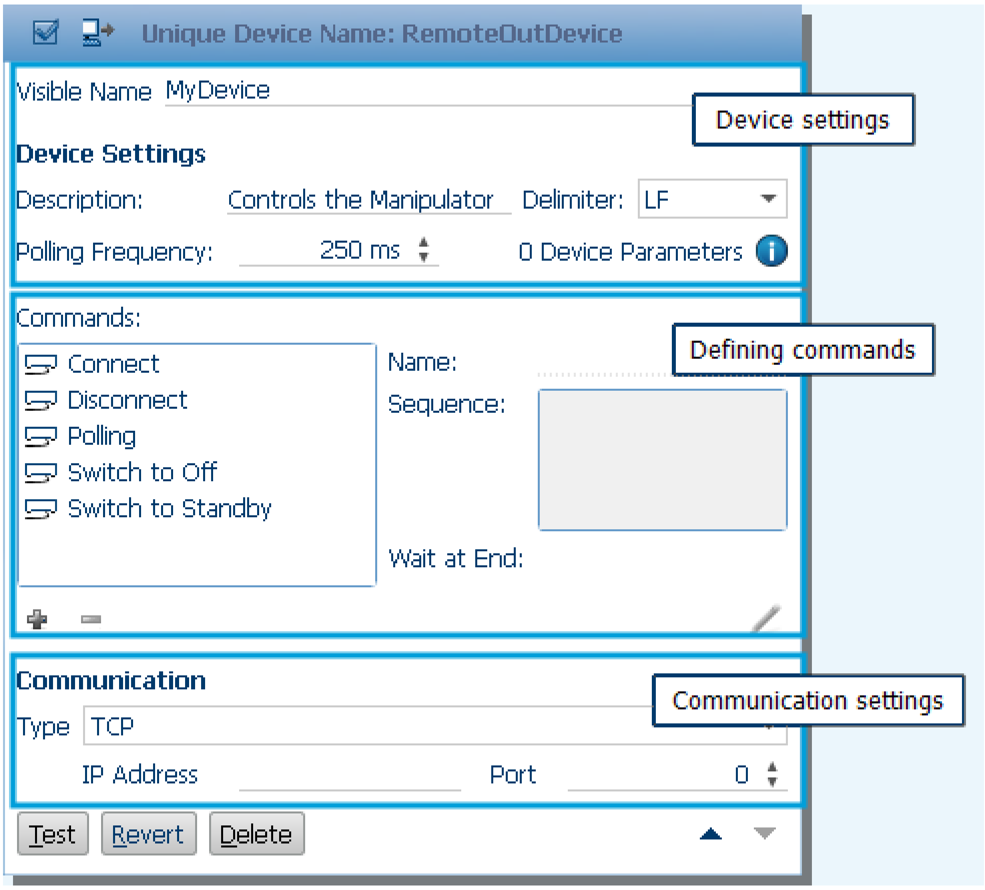
This page intentionally left blank.

Chapter 2 – Adding a Remote Out Device

All devices in SpecsLab Prodigy are added and configured in the SpecsLab Prodigy Configuration Tool. This is also the case for remote out devices. The difference is that more configuration is required by the user so that all necessary interactions between SpecsLab Prodigy and the device are covered.

To add a remote device:

1. Open the SpecsLab Prodigy Configuration Tool.
2. Select the *Device Configuration* tab.
3. Click *Add Device* and select *Remote Out* from the list. A new remote out device will be added to the defined devices. If a device already exists with this name, you will be prompted to enter a new name for the device.



Unique Device Name: RemoteOutDevice

Visible Name MyDevice

Device Settings

Description: Controls the Manipulator
Delimiter: LF
Polling Frequency: 250 ms
0 Device Parameters

Commands:
- Connect
- Disconnect
- Polling
- Switch to Off
- Switch to Standby

Name:
Sequence:
Wait at End:

Communication

Type TCP
IP Address
Port 0

Device settings

Defining commands

Communication settings

When you select the new remote out device in the list of devices. its properties are shown in the right pane. The screenshot below shows these properties divided into three parts. Each part is covered in the following sections. There is also a section covering the test of a remote device.

Note
You can remove the device by clicking *Delete*. The *Revert* button allows you to return all settings to those currently stored in the registry.

2.1 Defining the Device Settings

The device settings determine basic information about the interaction between SpecsLab Prodigy and the device, as well as providing descriptions that are used in the device control in SpecsLab Prodigy. The table below lists the device settings.

Feature	Description
Visible name	The name that appears in the title bar of the device control in SpecsLab Prodigy.
Description	A text description label that is shown in the device control for information.
Delimiter	Sets the delimiter that is sent at the end of each command: • LF—line feed. • CR—carriage return. • CRLF—carriage return and line feed. This delimiter is also used for the received data when the message is completed.
Polling frequency	Sets the interval for polling the device parameters. This time is for each polling event. Only one parameter is polled in each polling event. Examples: • For 1 parameter, a polling frequency of 100 ms means that the parameter is updated every 100 ms in SpecsLab Prodigy. • For 4 parameters, a polling frequency of 100 ms means that there is a single polling event every 100 ms, so each parameter is updated in intervals of 400 ms in SpecsLab Prodigy.
ℹ	Shows the number of device parameters. Hovering the mouse over the icon produces a tooltip listing all device parameters.

2.2 Defining Commands

Commands are strings of text that contain instructions to a device. Responses are also text. This section describes the facilities offered by SpecsLab Prodigy to allow you to integrate this control into SpecsLab Prodigy.

The following topics are covered:

- Pre-defined commands that are triggered by events in SpecsLab Prodigy.
- Details of how to enter sequences in a command.
- Using placeholders in commands.
- Adding other commands so that users can trigger certain actions.

Note
It is helpful to read this section in conjunction with "Appearance in SpecsLab Prodigy" on page 15. This gives you an idea of how the commands are used in the normal interface of SpecsLab Prodigy.

2.2.1 Adding Instructions to Commands

For the most part, the instructions to a device are the same text strings that you would send when controlling it over TCP, UDP or a serial connection. There are however placeholders for handling special characters and parameters. These are described in the following section.

Note the following points when writing commands:

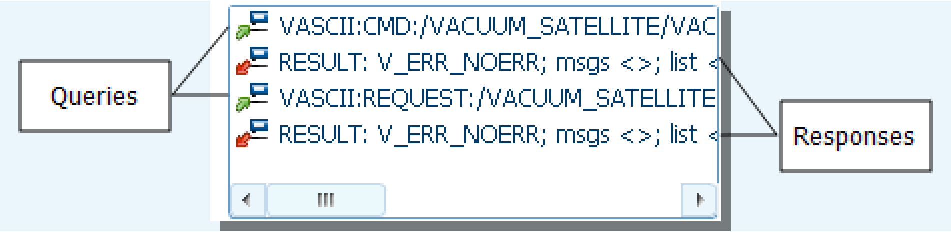
- The protocol is text-based. Communication with the device contains a set of instructions that will be understood by the instruction set of the device.
- A command consists of one or more sequences.
- A sequence consists of a query and the expected response.
- Queries and responses can contain placeholders for variables and special characters.

The following description is the same for pre-defined and user-defined commands. To add instructions to a command:

1. Select the command in the list of commands. If any sequences have been defined, they will be shown in the sequence pane.
2. Click the ✏ icon to start editing.

   Note
   You can double-click existing sequences to edit them.

3. Enter a sequence in the pane. This consists of alternating queries and replies, as shown in the screenshot below.
4. Click outside the editing pane when you have finished. The pane will show the sequences with icons to distinguish the queries and responses.



```
VASCII:CMD:/VACUUM_SATELLITE/VAC
RESULT: V_ERR_NOERR; msgs <>; list <
VASCII:REQUEST:/VACUUM_SATELLITE
RESULT: V_ERR_NOERR; msgs <>; list <
```

2.2.2 Placeholders

Placeholders are expressions in commands that represent parameters or special characters. SpecsLab Prodigy expands placeholders when executing the code.

The following sections list the uses of placeholders for remote out devices in SpecsLab Prodigy as well as including some examples showing their use.

Parameters

Placeholders allow SpecsLab Prodigy to recognize a device parameter. The value of the parameter is then displayed in the interface of SpecsLab Prodigy accordingly. It is also possible to use a placeholder so that users can enter the value of a parameter which will then be sent to the device.

The table below summarizes the placeholders that can be used to represent variables.

Note

Only parameters being specified with units can be logged over time (for example with the Live Data View).

Note

Parameters defined in the commands *Connect* and *Polling* are known as device parameters. When using device parameters, do not use the same name for different types, e.g. do not use the same VarName for a float and a string. Such conflicts will lead to an error.

Placeholder	Description
%{VarName:float}*	Numerical variable of type float.
%{VarName[Unit]:float}*	Numerical variable of type float with a unit.
%{VarName:enum:Selection1,Selection2,...}	List of variables.
%{VarName:string}	String.

*The identifier "float" may be omitted. If specified, it must be separated by a colon.

Special characters

The % sign is used as an escape character for various control characters, as well as for the % sign itself. The table below summarizes the available special characters.

Special character	Equivalent	Comment
%%	%	Percent sign (in command and response).
%{%n}	\n	Line feed.
%{%r}	\r	Carriage return.
%{%t}	\t	Tab.

Example 1—variables with units

Imagine that we have a manipulator, which translates the numerical value of a received position as an axis destination on a millimeter scale. For example, the manipulator would be able to move its X axis to 5 mm with the command:

```
SET X_POS 5
```

We further assume that the manipulator device returns its current X axis position likewise as millimeter value, represented by:

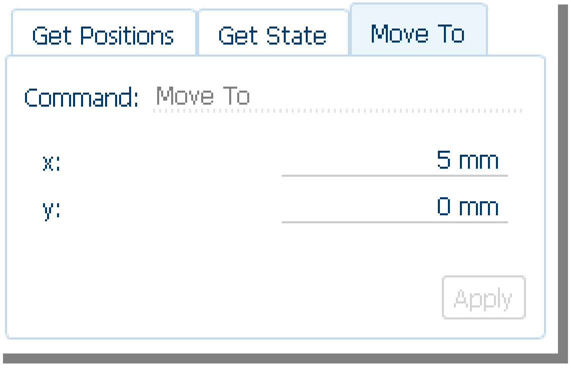
```
X_POS 5
```

In this context, it would be straightforward for the user to specify the position in the same manner, i.e. with the unit of length millimeter. We can hence define the positioning command in the SpecsLab configuration using a placeholder:

```
SET X_POS %{x[mm]:float}
```

The command defines the physical quantity for the user input to be expressed in millimeters; it will be reduced at a later stage to the form given at the beginning of the example. Notably, the %-placeholder does not contain a scaling factor. This is because using the same unit on the part of the manipulator and the user eliminates the need for conversion.

The device control in SpecsLab Prodigy will now appear with a corresponding entry field marked *x*, where the user can enter a millimeter value:



Command: Move To

x: 5 mm

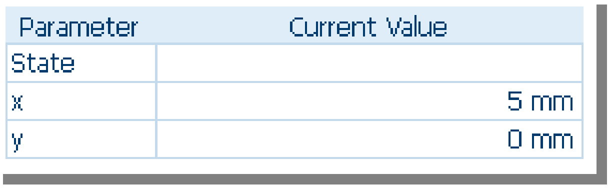
y: 0 mm

Clicking *Apply*, causes the placeholder `%{…}` to expand with value “5”, which results in the command `SET X_POS 5` being sent. The same goes for the manipulator’s Y axis if we proceed in a similar fashion.

In order to evaluate the response of the manipulator properly, we also use the placeholder:

```
X_POS %{x[mm]:float}
```

As before, SpecsLab Prodigy is aware of `X_POS` being a float, and converts the numeric value correctly. In the device control, "mm" is shown next to the return value "5":



Parameter	Current Value
State	
x	5 mm
y	0 mm

The unit specification within the placeholder may even be omitted. The conversion still works the same way. However, in this case the position can no longer be logged by the software.

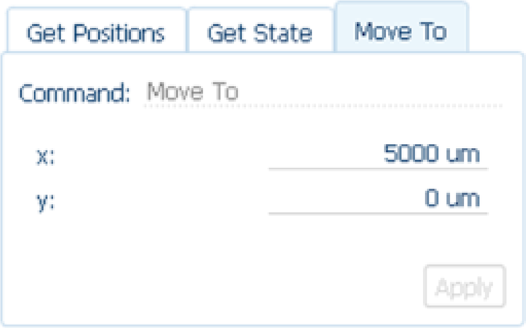
Note
Parameters without a unit are not available for logging.

Example 2—scaled variables with units

Imagine that we have the same manipulator as in example 1 using a millimeter scale internally, but we want the user to specify the position in micrometers (µm). For that matter, we perform the conversion "x[um] * 0.001 = x[mm]" with the scaling factor `0.001`. We can redefine the positioning command by adding the scaling factor to the placeholder:

```
SET X_POS %{x[um]*0.001}
```

Like before, the device control in SpecsLab Prodigy will feature an entry field marked *x*. Rather than setting the position in mm, the user is be prompted for a value in µm:



Command: Move To

x: 5000 um

y: 0 um

This time, SpecsLab Prodigy will expand the placeholder with the scaling factor `0.001`. Assuming the user input is “5000”, the command that is sent to the device will be the same as in example 1 due to the scaling:

```
SET X_POS 5
```

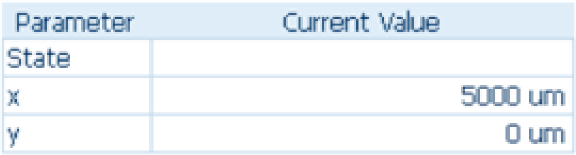
Sticking with the example, in the opposite direction the manipulator X axis position is returned in mm:

```
X_POS 5
```

This needs to be converted into a µm value for the device control to display it suitably. Since, by definition, the conversion is declarative, the placeholder uses the same scaling factor to obtain the desired output:

```
X_POS %{x[um]*0.001}
```

In the device control, the position now appears in the unit µm:



Parameter	Current Value
State	
x	5000 um
y	0 um

Example 3—lists

For this example, we shall assume that we have an X-ray source (or other similar device), which can be switched into an operating state using the command:

```
SET DEV_STATE On
```

Since we know the operating states that the user can select, we can put them into a list:

```
SET DEV_STATE %{Status:enum:On,Off,Standby}
```

The device control of SpecsLab Prodigy will show this as a list. The user can select the desired state from the list and send it to the device by clicking *Apply*.

You can also use enumerated lists in a response. However, it often makes more sense to use a string in such cases. When reading the status of a device, for example, there may be a *number* of error codes that are returned. Rather than listing all the error codes in the sequence, a string placeholder will simply accept any string and display it in SpecsLab Prodigy.

An advantage of using an enumeration type for the device status is that you can wait for specific states during the execution of the experiment and offer a list of valid states to the user.

2.2.3 Predefined Commands

There are five predefined commands that are performed in response to actions triggered by SpecsLab Prodigy. These are listed in the table below.

It is important to note that any parameters defined in the *Connect* or *Polling* commands are treated as "device parameters". These have special properties:

- All device parameters are listed in a tooltip when you hover the mouse over the ⓘ icon.
- Device parameters can be used in the *Wait at end* section when defining your own commands—see "Adding New Commands" on page 11.
- Although device parameters can be used in other commands, there must be no conflict in placeholder type. For example, if a device parameter is defined as a string, the same name cannot be used as a float in another command.
- When device parameters appear as responses, they are shown in the main section of the device control in SpecsLab Prodigy.

Note

Parameters defined in other commands are "local". The same name can appear with different types in different commands, as long as they do not also appear in *Connect* or *Polling*.

Command	Description
Connect	Executed when a connection is made.
Disconnect	Executed when disconnecting from the device.
Polling	Executed regularly (according to the polling interval in the device settings section) to retrieve information about the device.
Switch to off	Executed when SpecsLab Prodigy closes.
Switch to standby	Executed in the following cases: • When the command has been completed in the schedule of SpecsLab Prodigy. The Experiment Editor contains the message "The devices will be switched to a safe state after this item has been processed"—this command defines the "safe state". • While sputtering during a sputter depth profile experiment.

2.2.4 Adding New Commands

In addition to the predefined commands, you can define other commands which can be run by the user in SpecsLab Prodigy. In the SpecsLab Prodigy interface, each command appears in its own tab. Parameters defined in the command are shown in the tab and can be set by the user (for a query) or display the value (for a response).

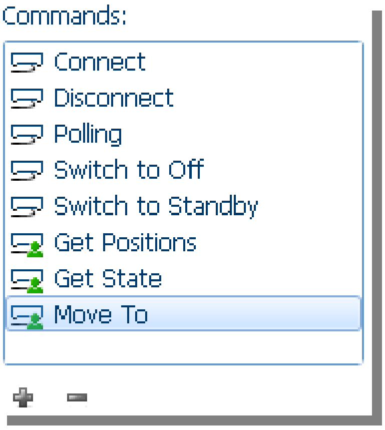
Definition of user-defined commands follows essentially the same principles as the "Predefined Commands".

Examples of such commands include:

- Setting the position of a manipulator axis.
- Setting the anode HV.
- Switching on a bias voltage for the duration of an experiment.

To add a user-defined command:

1. Click the ➕ icon below the pane containing the list of commands. The *Add Command* dialog will open.
2. Enter a name for the new command and click *OK*. User-defined commands are marked with a green "user" on the icon.



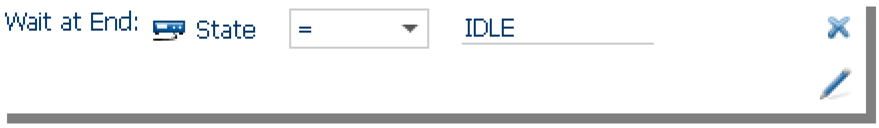
Select the new command in the *Commands* pane.

3. Click the ✎ icon to start editing the command.
4. Define the command in the same way as for the pre-defined commands.

You can optionally set a condition for a device parameter in the *Wait at end* section. The command will wait until this condition is reached before exiting. This is useful if you want to ensure that the device has a particular status or that a parameter has reached a defined value.

To use the *Wait at end* feature:

5. Click *Add Parameter*. A menu will appear containing all the device parameters that you have defined in the pre-defined commands.
6. Select the parameter you want to use for the condition.
7. Select an operator to define the relation:
   - For strings or items from enumerated lists, you can select = or ≠.
   - For numerical variables, you can select =, ≠, ≥ or ≤.
8. Enter the condition for the parameter. The example in the screenshot below means that the command will wait until the device reports the status *IDLE* before exiting.



Wait at End: State = IDLE

2.3 Communication Settings

SpecsLab Prodigy needs to know how to connect to a device. This information is set in the communication settings. You should make sure that your device is available under the supplied settings.

The remote out interface supports four types of communication protocols:

- TCP: Transfer control protocol over a LAN.
- UDP: User datagram protocol over a LAN.
- Serial: For an RS-232 connection, e.g. over COM 1.
- Executable: Standard input / output streams.

TCP and UDP

For the TCP and UDP protocols, you need to set the IP address and port number for the device.

Serial

When controlling devices over a serial interface, you need to select the COM port and its communication settings.

Executable

This setting is used to start executables or batch-scripts which will receive the commands on the standard input stream and have to write the reply to the standard output stream. You have to specify the executable, which will be started when going online, and it will be stopped when going offline. Optionally you can specify command line arguments.

Note

Executables will be started with administrator permissions as user SYSTEM.

2.4 Testing the Remote Out Device

After defining your commands, you can test the device by clicking the *Test* button. This performs the following actions:

- Establish a connection to the device according to the communication settings.
- Run the sequences defined in the *Connect* command, if any.
- Run the sequences defined in the *Polling* command, if any.
- Run the sequences defined in the *Disconnect* command, if any.

The test is successful if all three steps run correctly. In case of any problems, an error message will appear with details.

2.5 Storing the Remote Out Settings

After defining the remote out device, you need to store the settings:

- Click *Store* in the main toolbar of the SpecsLab Prodigy Configuration Tool. This will write the settings to the registry. The configuration is available the next time you start SpecsLab Prodigy.

Note

The *Revert* button returns the settings in the remote out device to the current registry settings.

Chapter 3 – Appearance in SpecsLab Prodigy

As described in the previous chapter, configuration of remote out devices is performed in the SpecsLab Prodigy Configuration Tool. This chapter shows how your device will appear in SpecsLab Prodigy and how it can be used.

Operation is basically the same as for all SPECS devices that are controlled from SpecsLab Prodigy—there are two ways to access and control the device:

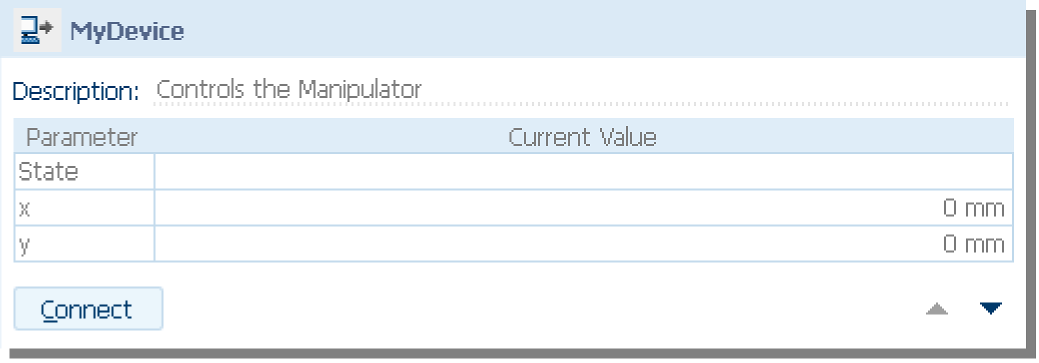
- Device Control view.
- Experiment schedule.

3.1 Remote Out Devices in the Device Control View

The Device Control view contains all devices that SpecsLab Prodigy can connect to and control. Your remote out device should be listed in the pane on the left side of the view. Clicking this will take you to the device.

In order to access the device, you need to click the *Connect* button. This will establish the connection.

The initial view of the remote out device shows the device parameters which were defined in the *Connect* and *Polling* commands in the SpecsLab Prodigy Configuration Tool. The name of the device, here "MyDevice" is set in the *Visible Name* section when configuring the device.

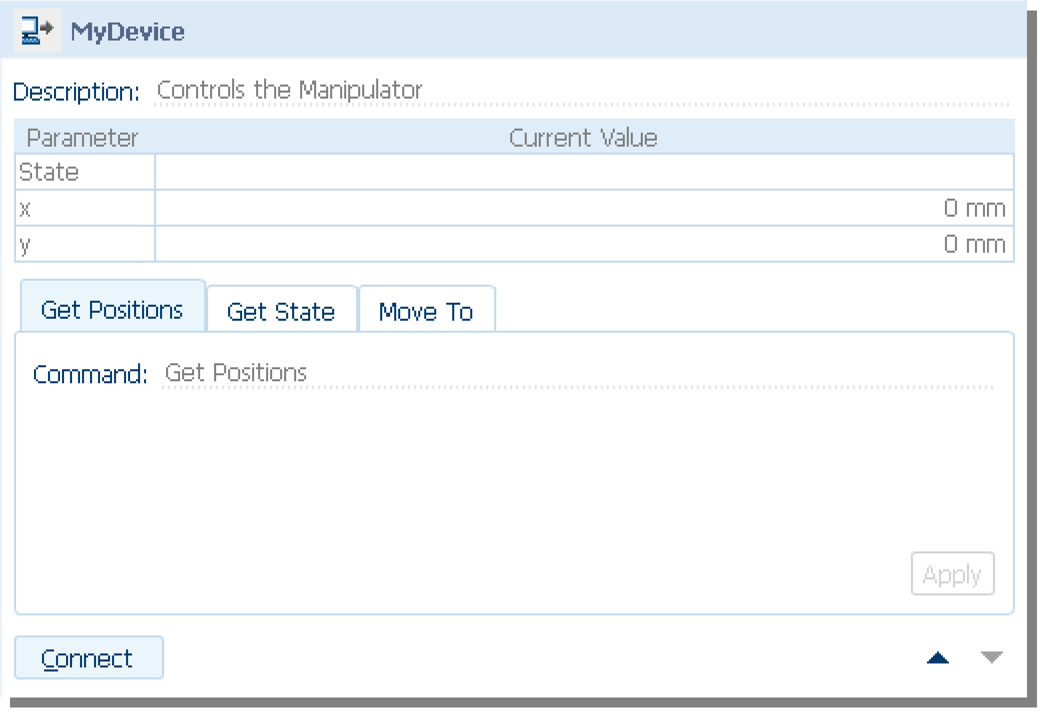


MyDevice

Description: Controls the Manipulator

Parameter	Current Value
State	
x	0 mm
y	0 mm

When you expand the device control by clicking the ▼ button, the user-defined commands are shown. Each command is in its own tab. You can set the parameters for the command as necessary and execute the command by clicking *Apply*.



Description: Controls the Manipulator

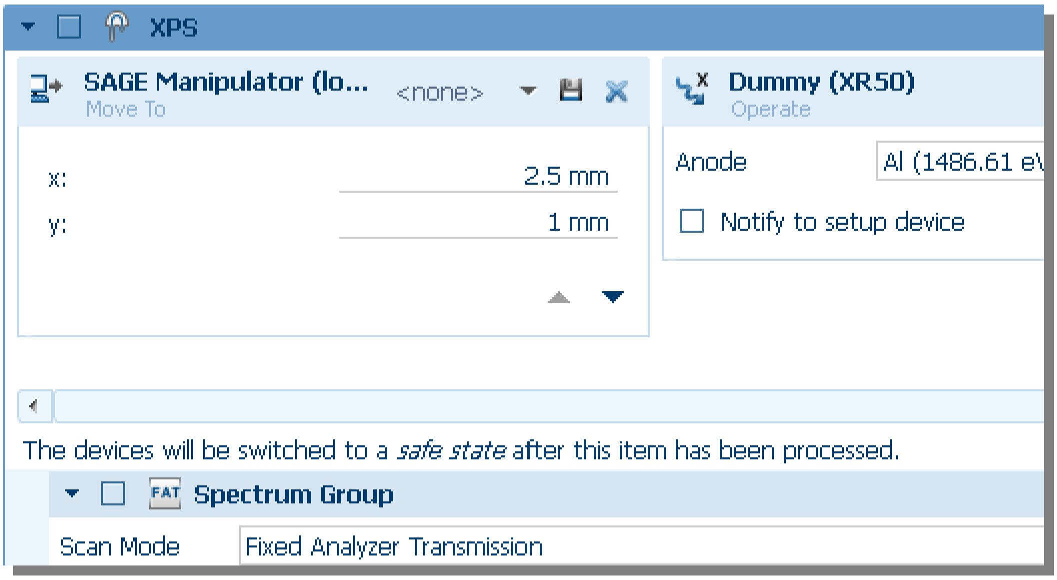
Parameter	Current Value
State	
x	0 mm
y	0 mm

Command: Get Positions

3.2 Using Remote Out Devices in Experiments

User-defined commands can be added to the schedule and run as part of experiments. The remote out device appears in the Add Device list with the other SPECS devices installed on your system. On clicking the device, a menu appears containing all the user-defined commands.

You can select one or more of these commands to add to the experiment. The screenshot below shows part of the definition of an XPS experiment.



x: 2.5 mm
y: 1 mm

Anode: Al (1486.61 eV

□ Notify to setup device

The devices will be switched to a *safe state* after this item has been processed.

Scan Mode: Fixed Analyzer Transmission

Setting the parameters is the same as in the device control. When you run the experiment, SpecsLab Prodigy will automatically connect to the device and execute the command with the selected parameters. After the XPS section (and its subsections, such as the Spectrum Group) has been processed, the device will be set to the safe state as determined by the predefined command *Switch to Standby* in the configuration.

This page intentionally left blank.

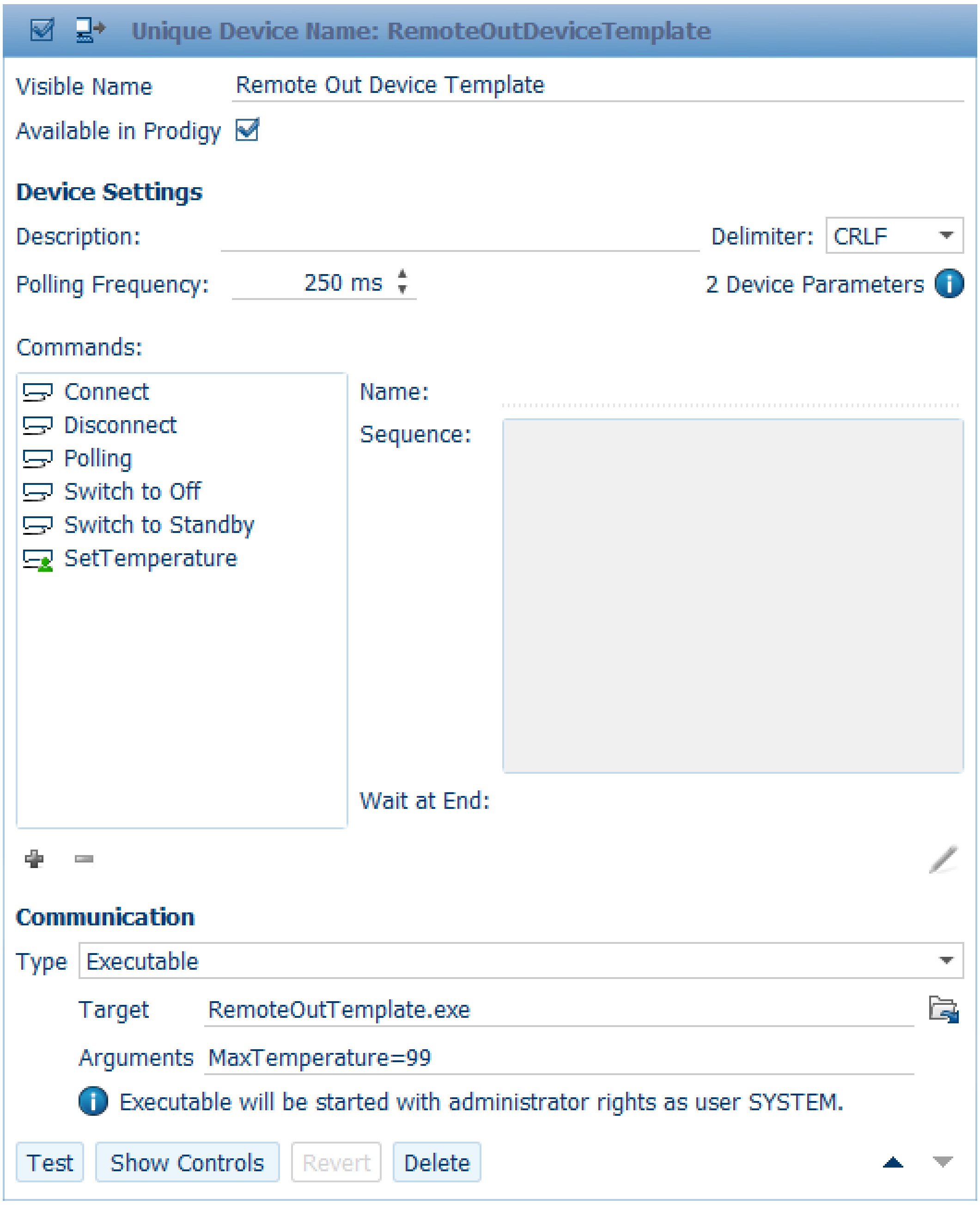
Chapter 4 – Programming Examples for Communication with Executables

Some programming examples are included in the SpecsLab Prodigy installation under the path:
C:\Program Files\SPECS\SpecsLab Prodigy\Programming Examples.

The examples demonstrate how to use executables and scripts with the Remote Out module:

- [...]\C++\RemoteOut
  Visual C++ example of a simple heater.
- [...]\C++\RemoteOutTango:
  Visual C++ example for communicating with devices via TANGO; the TANGO test device is used which is part of the TANGO environment by default.
- [...]\Python\RemoteOut:
  Python example to echo the command parameters.
- [...]\Scripts\RemoteOut:
  Batch file example to echo the command parameters.

All examples have a cfg file which can be imported into the system through the *Import File* button in the *SpecsLab Prodigy Configuration Tool* to simplify the setup. Please note that the path to the executables and scripts may have to be adapted if you have a 32 bit installation running on a 64 bit operating system. Also, the path to the python.exe may have to be adapted.



Unique Device Name: RemoteOutDeviceTemplate

Visible Name: Remote Out Device Template

Available in Prodigy ☑

Device Settings

Description:

Delimiter: CRLF

Polling Frequency: 250 ms

2 Device Parameters

Commands:

- Connect
- Disconnect
- Polling
- Switch to Off
- Switch to Standby
- SetTemperature

Name:

Sequence:

Wait at End:

Communication

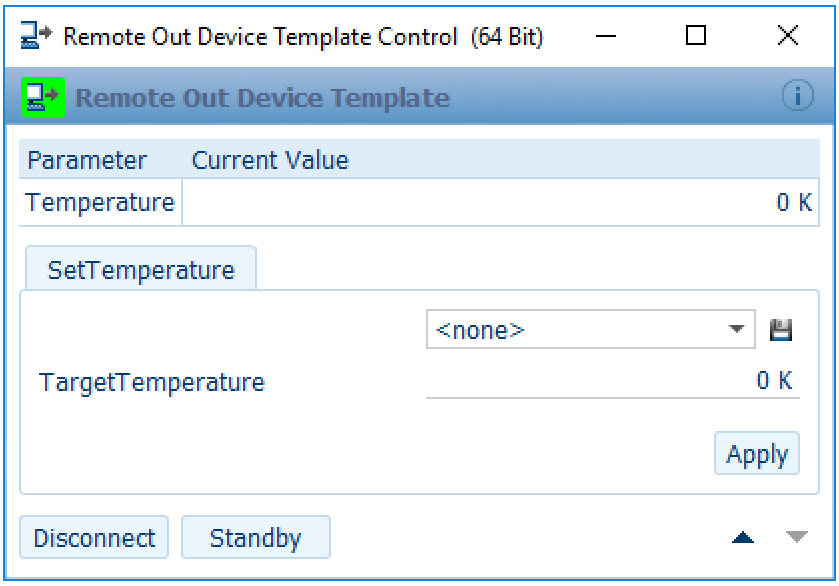
Type: Executable

Target: RemoteOutTemplate.exe

Arguments: MaxTemperature=99

Executable will be started with administrator rights as user SYSTEM.

After importing and storing the configuration, click on the *Show Controls* button to launch the device controls user interface to test it:



Remote Out Device Template Control (64 Bit)

Remote Out Device Template

Parameter	Current Value
Temperature	0 K

SetTemperature

TargetTemperature <none> 0 K

This page intentionally left blank.

Index

This page intentionally left blank.