Remote Controller

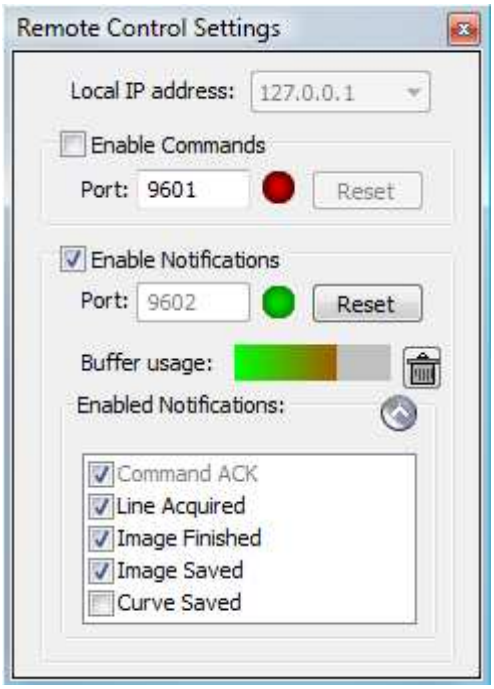
This support note shows how to use Remote Controller of **WSxM** *program.*

Introduction

WSxM Remote Control is a feature included in the WSxM software since version WSxM 5.0 Develop 2.0 that allows a user to give orders to the software from another program. This program can run on the same machine as WSxM or in any other one which is connected to it over a network, using the TCP/IP protocol. In addition to this, WSxM can send notifications to another client application to let it know that a command has been executed; an image has been acquired, etc.

User Interface

The Remote Control Settings dialog bar allows the user to change the connection parameters (such as IP and port).



Local IP address: the IP to be used in the machine running WSxM (the Host) when connecting to the client applications. If several IP addresses are available, the user must choose one. To use WSxM Remote Control over a network, an IP different from loopback address (127.0.0.1) must be used.

Enable Commands: enables or disables the reception of remote commands from the specified Port.

Enable Notifications: enables or disables the sending of remote notifications to the specified Port.

Reset: closes the corresponding connection and waits for a new one. This can be useful when the remote application has finished without closing the connection properly.

Client applications always connect to WSxM, and not on the contrary. That's why the client addresses are not required.

Buffer usage: shows how much of the Pending Notifications Buffer is being used. When the buffer is full, new notifications will be

lost. This can be avoided by reducing the number of notifications, decreasing the frequency, etc.

Trash Button: Empties the Notification Buffer.

Enabled notifications List: lets the user choose which notifications wants to receive.

Commands

The format of the commands accepted by WSxM is the following:

ID Type *Parameters* Delimiter

- ID: optional command identifier. If supplied, it is included in ACK packets to distinguish the executed command. Any character can be used, excepting command delimiter and blanks, and it must be enclosed between curly brackets.
- Type: the type of command the remote user wants to execute. It is not case-sensitive, so control_get_y_offset is the same as control_GET_Y_offset.
- *Parameters*: depending on command type, different parameters may be used.
- Delimiter: a character which indicates the end of the command. This command cannot be used anywhere inside the command. This character is **$.**

Examples:

- {1} wsxm_get_version$
- control_set_xy_offset 23.5 -56.3$
- {ident} control_get_y_offset$
- {id35} control_save_now $
- control_set_scan_freq 3.5 $

* In the rest of examples, the final $ will be assumed and not mentioned.

After a command is sent, it will be executed as soon as possible. If WSxM receives a command while another is being executed, the new command will wait until the previous one has finished. No command is executed unless the "Enable Commands" option is active.

When WSxM finishes executing a command, it will send an ACK packet. These packets, as well as *Information packets*, are only sent if the "Enable Notifications" option is active. They can be useful to know when a command has really finished.

Commands transmitted over TCP/IP are sent as a continuous stream rather than as independent command packets. For WSxM to be able to process these commands, a command delimiter must be chosen (in this case, **$**). This allows the remote controller to send a big amount of data including any number of commands a as a unique logical block.

The following is an example of all the data sent in a simple remote session:
wsxm_get_version$control_set_size
1000$control_set_points_128$control_set_save$

Commands may have any number of parameters, depending on the command type. These parameters can be integer numbers (3, 17, -5…), real numbers (3.15, -2.8, 0.0003…), boolean (true, false) strings (one, continuous…) and ext-strings (c:\my files). A string can contain neither blanks nor double quotes. An ext-string may contain them.

Available Commands

Control

Command	Parameters	ACK values
control_get_points	-	integer
control_get_scan_freq	-	real (Hz)
control_get_size	-	real (nm)
control_get_x_offset	-	real (nm)
control_get_y_offset	-	real (nm)
control_get_z_gain	-	real
control_get_z_offset	-	real (nm)

control_set_points	integer	-
control_set_scan_freq	real (Hz)	-
control_set_size	real (nm)	-
control_set_x_offset	real (nm)	-
control_set_xy_offset	real (nm) real (nm)	-
control_set_y_offset	real (nm)	-
control_set_z_gain	real	-
control_set_z_offset	real (nm)	-

control_up	-	-
control_down	-	-
control_save_now	-	-
control_set_save	boolean (true / false)	-

Scan control

Command	Parameters	ACK values
scan_pause	-	-
scan_resume	-	-

Saving options

Command	Parameters	ACK values
saving_options_set_name	ext-string	-
saving_options_set_type	string	-

Lithography

Command	Parameters	ACK values
litho_run_program	ext-string	-

3d Modes

Command	Parameters	ACK values
3dmodes_start	-	-
3dmodes_finish	-	-
3dmodes_set_acq_point	integer(line) integer(column)	-
3dmodes_set_acq_point_mode	string	-

Curve acquisition

Command	Parameters	ACK values
fz_acquire	-	-
fz_save_all	-	-
iz_acquire	-	-
iz_save_all	-	-
iv_acquire	-	-
iv_save_all	-	-
zv_acquire	-	-
zv_save_all	-	-
gc_acquire	-	-
gc_save_all	-	-

Miscellaneous

Command	Parameters	ACK values
wsxm_get_version	-	ext-string
wait	int	-
wait_image	-	-

Help

Command	Parameters	ACK values
help	-	ext-string
help_remote_command_list	-	ext-string
help_remote_command	string	ext-string

Notification Packets

There are two types of notification packets which can be sent from WSxM to a remote client: **ACK** packets and **Information** packets.

ACK Packets:

ACK packets are sent by WSxM to indicate that a command has been executed. These packets begin with **[ack]**, followed by a string indicating the status of the action and, optionally, any number of values.

All commands will send an ACK packet when executed. If a command is sent and no response is received, it may be because a blocking action requires the user to make something (such a message box waiting to be accepted).

When there is no client application connected to read notifications, WSxM stores them in an internal fixed-size queue (only if Notifications are enabled). They will be sent when such application is connected to prevent loss of information. If these notifications are not required, they can be simply read and ignored.

Information Packets:

Information packets are sent by WSxM to indicate that certain events have occurred (e. g. when an image has been taken).

Information packets begin with **[info]**. The rest of the packet depends on the type of event.

The character used as notification separator is also $.

Available *Information* notifications

Notification	Parameters
Image finished.	-
Image saved.	Any number of ext-string
Curve saved.	Any number of ext-string
Line acquired.	Line information parameters*

* Line information parameters are, at least, the following:

- Channel (ext-string): the channel name.
- Unit (ext-string): the unit in which the data is shown.
- Direction (string): either **forward** or **backward**.

- Index (integer): 0-based index of the acquired line. 0 is the topmost line.
- Points (integer): number of points in the line.
- Data (ext-string): list of values in the acquired line. The string begins and end with a **"** symbol. Each element is a real number, and is separated from the following element by a blank. The first number in the list is always the leftmost point of the line (as shown in the acquired image).

Examples:

←[info] Image finished.

←[info] Image saved. "c:\acquired_data\myfile001.stp"

←[info] Curve saved. "c:\my_data\myfile_0016_Normal force.fz.cur"

←[info] Image saved. "c:\data\file_0001.f.top"
"c:\data\file_0001.b.top"

←[info] Line acquired.Channel: "Lateral force";Unit: "V";Direction:forward;Index:3;Points:8;Data:"2.04437 2.04437 2.04468 2.04437 2.04437 2.04468 2.04468 2.04468"

Command Reference

3dmodes_finish

- Description.
  Finishes the 3dModes acquisition.

- Example:
  → 3d_modes_finish
  ← [ack] Ok.

3dmodes_set_acq_point

- Description.
  Sets the initial acquisition point for 3dModes (only if the acquisition point mode is set to manual). The parameters are the column and row of the desired starting position and so, they must be in the range 0 … number of points – 1.

- Example:
  → 3d_modes_set_acq_point 64, 32
  ← [ack] Ok.

  → 3d_modes_set_acq_point 128, 128
  ← [ack] Invalid value.

  → 3d_modes_start
  ← [ack] Ok.
  → 3d_modes_set_acq_point 32, 32
  ← [ack] Command not available at this moment.

3dmodes_set_acq_point_mode

- Description.
  Sets way in which the initial acquisition point for 3dModes is selected. The parameter can be any of these:
  - **auto**: the center of the image will be used as starting point.
  - **manual**: a user-specified point will be used as starting point.

Note that, under some circumstances, it may be not possible to change the initial acquisition point (e.g. 3d Modes already started, XY channels selected as Fast/Slow channels…)

- Example:
  → 3d_modes_set_acq_mode manual

← [ack] Ok.

3dmodes_start

- Description.
  Starts the 3dModes acquisition.

- Example:
  → 3d_modes_start
  ← [ack] Ok.

control_get_points

- Description.
  Returns the current number of points per line.

- Example:
  → control_get_points
  ← [ack] Ok. 256

control_get_scan_freq

- Description.
  Returns the current scan frequency, in Hertz.

- Example:
  → control_get_scan_freq
  ← [ack] Ok. 1.97

control_get_size

- Description.
  Returns the current scan size, in nanometers.

- Example:
  → control_get_scan_size
  ← [ack] Ok. 1000

control_get_x_offset

- Description.
  Returns the current value of the X offset.

- Example:
  → control_get_x_offset
  ← [ack] Ok. 230

control_get_y_offset

- Description.
  Returns the current value of the Y offset.

- Example:
  → control_get_y_offset
  ← [ack] Ok. 125.50

control_get_z_gain

- Description.
  Returns the current Z Gain.

- Example:
  → control_get_z_gain
  ← [ack] Ok. 10

control_get_z_offset

- Description.
  Returns the current value of the Z offset.

- Example:
  → control_get_z_offset
  ← [ack] Ok. 350

control_set_points

- Description.
  Sets the number of points per line. Since not every value is allowed, the closest permitted value is set.

- Example:
  → control_set_points 128
  ← [ack] Ok.

control_set_scan_freq

- Description.
  Sets the value of the scan frequency, in Hertz. Since not every value is allowed, the closest permitted value is set.

- Example:
  → control_set_scan_freq 4.5
  ← [ack] Ok.

control_set_size

- Description.
  Sets the value of the Scan size, in nanometers.

- Example:
  → control_set_size 1000
  ← [ack] Ok.

control_set_x_offset

- Description.
  Sets the value of the X offset, in nanometers.

- Example:
  → control_set_x_offset 250
  ← [ack] Ok.

control_set_xy_offset

- Description.
  Sets the value of the X offset and Y offset, both in nanometers.

- Example:
  → control_set_xy_offset 250 350
  ← [ack] Ok.

control_set_y_offset

- Description.
  Sets the value of the Y offset, in nanometers.

- Example:
  → control_set_y_offset 250
  ← [ack] Ok.

control_set_z_gain

- Description.
  Sets the value of the Z Gain. Since not every value is allowed, the closest permitted value is set.

- Example:
  → control_set_z_gain 15
  ← [ack] Ok.

control_set_z_offset

- Description.
  Sets the value of the Z offset, in nanometers.

- Example:

→ control_set_z_offset 300
← [ack] Ok.

control_down

- Description.
  Moves the scan position to the bottom of the image.

- Example:
  → control_down
  ← [ack] Ok.

control_up

- Description.
  Moves the scan position to the top of the image.

- Example:
  → control_up
  ← [ack] Ok.

control_save_now

- Description.
  Saves the current image immediately. All channels that are marked for saving will be saved. So, many files may be actually saved. In this case, an information message will also be sent.

- Example:
  → control_save_now
  ← [info] Image saved. "c:\mydata\myfile.f.top"
  ← [ack] Ok.

control_set_save

- Description.
  Activates or deactivates saving. If a **control_set_save** command is executed with a parameter which is the current state, nothing happens.

- Example:
  → control_set_save true
  ← [ack] Ok.

fz_acquire, iz_acquire, iv_acquire, zv_acquire, gc_acquire

- Description.
  Acquires curves of the specified type. The settings for this acquisition will be those currently entered in the corresponding acquisition dialog.

- Example:
  → fz_acquire
  ← [ack] Ok.
  → zv_acquire
  ← [ack] Command not available at this moment.

fz_save_all, iz_save_all, iv_save_all, zv_save_all, gc_save_all

- Description.
  Saves the last acquired set of curves of the specified type, which are shown in the corresponding dialog. The type and number of curves to save can be selected in the dialog. An *information packet* with the names of these curves will be sent if one or more curves were saved.

- Example:
  → iv_save_all
  ← [ack] Ok.

help

- Description.
  Returns some generic help information.

- Example:
  → help
  ← [ack] Ok.
  help_remote_command_list: returns available remote commands.
  help_remote_command command: returns information about the specified command.

help_remote_command

- Description.
  Returns information about a remote command.

- Example:
  → help_remote_command control_get_z_offset
  ← [ack] Ok. Returns the current value of the Z offset.

help_remote_command_list

- Description.

Returns a list of the available remote commands, separated by blanks.

- Example:
  → help_remote_command_list
  ← [ack] Ok.
  control_get_x_offset
  control_get_y_offset
  control_get_z_offset
  control_get_scan_freq
  ...

* Note: any of the help commands may contain any number of DOS newlines (ASCII 0x0A 0x0D or '\r\n') to make the result more readable.

litho_run_program

- Description.
  Executes a lithography program.

- Example:
  → litho_run_program begin setfeedback(active=false)end
  ← [ack] Ok.

saving_options_set_name

- Description.
  Sets the base name of the saved files. The parameter can include blanks.

- Example:
  → saving_options_set_name my_file
  ← [ack] Ok.

  → saving_options_set_name my?file
  ← [ack] Invalid value.

saving_options_set_type

- Description.
  Sets the saving type. The parameter can be any of these:
  - **one**: only saves the following image.
  - **continuous**: saves all the acquired images from now on.
  - **secure**: only saves the following image and then secures the tip withdrawing the piezo and pausing the scan.

Note that this just sets the saving mode. To activate saving, a **control_set_save** instruction must be used.

- Example:
  → saving_options_set_type one
  ← [ack] Ok.

  → saving_options_set_type all
  ← [ack] Invalid value.

scan_pause

- Description.
  Pauses the scan. If it is already paused, nothing happens.

- Example:
  → scan_pause
  ← [ack] Ok.

scan_resume

- Description.
  Resumes the scan. If WSxM is already scanning, nothing happens.

- Example:
  → scan_resume
  ← [ack] Ok.

wait

- Description.
  Waits for the specified time (given in milliseconds), without executing remote commands. WSxM continues operating as usual within this wait time.

- Example:
  → wait
  ← [ack] Ok.

wsxm_get_version

- Description.
  Returns WSxM software version. This command can be useful to check if a connection has been successfully established.

- Example:
  → wsxm_get_version
  ← [ack] Ok. "WSxM 5.0 Develop 1.3 December 2009"

wait_image

- Description.

  Waits until a new image has been taken, without executing remote commands. WSxM continues operating as usual within this wait time.

- Example:

  → wait_image
  ← [ack] Ok.

Nanotec Electrónica S.L.
Centro Empresarial Euronova 3
Ronda de Poniente 12
2ª planta, oficina C
E28760 Tres Cantos (Madrid)
SPAIN

Phone: + 34 91 804 33 47
Fax: + 34 91 804 33 48
Web: www.nanotec.es
CIF: ES000B82006057

06/24/2010