List of commands for the USB interface

Parameters for the communication port :

Baud rate : 115200 Bauds Data bits : 8 Stop bits : 1

No parity Flow control : none Character for transmission end : LF (0xA)

List of the commands :

Important note: the commands have always 5 characters and it is important to respect the syntax of each of those (i.e. capital letters have to be used).

«HELP_» :

This command allows to know all the information related to the different commands.

«_RAZ_» :

This commands sets all the outputs of the DAC to 0V.

« INFOS » :

This commands allows to know all the basic informations of the nanopositionning system (name of the stage, number of axis, maximum stroke, etc...)

« MOVRX » : This commands allows to move the X axis relatively to the current position (the same function : MOVRY et MOVRZ exist for the Y axis and the Z axis).

For example, if I want to move 30µm in the positive direction, one must write : « MOVRX +30u » ,

If I want to move in the other direction, one must write « MOVRX -30u ».

« MOVEX » :

This command allows to move the stage to an exact position (similar function exist for the Y axis and the Z axis : MOVEY and MOVEZ)

As an example, if one wants to move to the 200 µm position (ie 200000 nm), one must write : « MOVEX 200u » or « MOVEX 200000n »

« GET_X » :This command allows to read the position of the X axis (similar function exist for the Y axis and the Z axis : GET_Y and GET_Z)

« STIME» :
This command allows the modification of the time between 2 points which are sent by the RAM of the USB interface to the nanopositioner

Consequently,
If I want to have a time between points of 200ms, I must write : « STIME 200m ».
If I want to have a time between points of 20µs, I must write « STIME 20u ».
If I want to have a time between points of 2s, I must write « STIME 2s ».

« GTIME » :
This command allows you to know the time between each point.

« SWF_X »
This command allows you to set up a ramp waveform for the X axis. Example:
"SWF_X 100 0u 100u" loads a waveform of 100 equal sized steps from 0u to 100u.

« SWF_Y »
« SWF_Z »
As per SWF_X.

« ARBWF»

This command prepares the controller for receiving arbitrary waveform data.
Data storage is allocated for the data when ARBWF is executed, data points must then be loaded using ADDPn.

Example:
« ARBWF 100 20 10 » This allocates storage for 100 X axis data points, 20 Y axis data points and 10 Z axis data points.

« ADDPX »

This command allows adding of data points to a waveform.
This command is only valid when used after ARBWF has been used to allocate data storage for the appropriate waveform data.
Example:
«ADDPX 100u» This command adds a point to the end of the X waveform data set. The command accepts the same format of position as MOVEX.

« ADDPY »
« ADDPZ »

As per ADDPX.

« RUNWF »

This command allows to launch the scan defined by the function SETWF. X axis is the first axis. Y axis is the second axis. Z axis is the third axis. One should write « RUNWF ».

« RUXYZ»

This command is the same as RUNWF but also works for all axis in every sequences
Example: RUZXY, RUYXZ ,...

« RUXY_»

This command is the same as RUNWF but works for 2 axes in every sequences

Example: RUZX_, RUYX_ ,…

« RUX__»

This command is the same as RUNWF but works only for 1 axe in all orders

Example: RUZ__, RUY__ ,...

« ARB3D »

This command prepares the controller for receiving a sequence of arbitrary 3D locations. Data storage is allocated for the data when ARB3D is executed, data points must then be loaded using ADD3D.

Example: «ARB3D 10» This command allocates storage for 10 3D locations.

« ADD3D »

This command allows adding a 3D location to the 3D positions list.

Example: «ADD3D 10u 10u 0u» This command allocates storage for one location. Specify all axes locations using the same number format as MOVEX.

« RUN3D »

This command runs the stored sequence of 3D locations.

Example: «RUN3D»

« DISIO »

This command displays the setup of the TTL ports. DISIO must be used with a TTL port number (1-4).

Example :

« DISIO 1 »

Returns

«

status:

input

position:

Channel1

RisingEdge

»

« CHAIO »

This command allows setting the parameters of the 4 TTL IO ports.

Each TTL Port can be individually set to one of the follow states :

Disabled

Input Rising – responds to the rising edge of the incoming TTL signal

Input Falling – responds to the falling edge of the incoming TTL signal

Pulse Out Start – Provides a TTL pulse at the start of motion on the chosen axis

Pulse Out End – Provides a TTL pulse at the end of motion on the chosen axis

Step Number Pulse – Provides a TTL pulse at the start of motion of the chosen axis on the specified step number

Gate pulse – Provides a TTL gate signal, switches on and off at the start of the specified steps

Output mode provides a TTL pulse at the start or end of motion of a chosen axis, or an output pulse on a specified step of motion on a given axis.

« CHAIO 1n » Disables TTL port 1

« CHAIO 1i1r » Sets TTL 1 as an input, triggering motion on the 1st axis on the rising edge

« CHAIO 1i2f » Sets TTL 1 as an input, triggering motion on the 2nd axis on the falling egde

« CHAIO 1o1s » Sets TTL 1 as an output, providing pulses at the start of motion of the first axis

« CHAIO 1o1e » Sets TTL 1 as an output, providing pulses at the end of motion of the first axis

« CHAIO 1o1n6 » Sets TTL 1 as an output, providing a pulse at the start of step number 6 motion on the first axis.

« CHAIO 1o1g5-10 » Sets TTL 1 as an output, provide a gate pulse controlled by the first axis, TTL output turns on at step 5 and turns off at step 10.

Format of disable commands :

« CHAIO [channel]d » where [channel] is TTL port 1-4

Format of Input commands :

« CHAIO [channel]i[axis][modifier] » where

[channel] is TTL port 1-4,

[axis] is the axis to trigger 1-3 and

[modifier] is either 'f' or 'r' designating triggering of the axis occurring on falling or rising edge of the TTL signal respectively.

Specific command have to be used for input TTL : the function to run a waveform with an input TTL is « RUTTL». Use «CHAIA» for removing an input TTL CHAIO's command.

Format of the Output commands :

TTL pulses on every step:

« CHAIO [channel]o[axis][modifier] » where

[channel] is TTL port 1-4,

[axis] is the axis that generates TTL pulses and

[modifier] is either 's' or 'e' designating TTL pulses occurring at the start or end of motion respectively.

Or

TTL pulses on a given step number:

« CHAIO [channel]o[axis]n[step number] » where

[channel] is TTL port 1-4,

[axis] is the axis that generates TTL pulses and

'n' designating step number is to be provided

[step number] is an integer number within the step range as specified in ARBWF or SWF_n commands (eg for a waveform of 50 steps, the valid range of step numbers is 0-49)

Or

TTL Gate signal turned on and off at specified step numbers:

« CHAIO [channel]o[axis]g[start step number]-[end step number] » where

[channel] is TTL port 1-4,

[axis] is the axis that generates TTL pulses and

'g' designating TTL gate behaviour

[start step number] is an integer number within the step range as specified in ARBWF or SWF_n commands (eg for a waveform of 50 steps, the valid range of step numbers is 0-49)

[end step number] is an integer number within the step range as specified in ARBWF or SWF_n commands (eg for a waveform of 50 steps, the valid range of step numbers is 0-49)