User Manual

NANOSCAN NPC-D-6xx0 SERIES NANOMECHANISM CONTROLLER COMMAND SET AND CONTROL SYSTEM

Description of control system and command set allowing configuration and control from customer software

Document No: EN-014429-UM Rev 2.0

Safety Precautions

WARNINGS

HAZARDOUS VOLTAGES

The Product generates high voltages and relies on the provision of a protective earth (ground) conductor to prevent user accessible components developing a hazardous potential in the event of an insulation failure. This protective earth is provided by the external power supply and only an approved power supply should be used with the product. Additional protection is provided by special NanoMechanism interface connectors and cable assembles. To maintain the integrity of the operator safety systems only approved NanoMechanisms and cables should be used with the product. The product should not be used if there are any signs of damage or if the equipment is believed to be faulty. It should be returned to the manufacturer for investigation and repair.

DO NOT remove the equipment’s protect covers. There are no user serviceable parts within the equipment and removal of the cover will expose the user to potential high voltage hazards and will invalidate the Warranty.

Do read the manual before using the controller to understand how to correctly and safely operate the product. Incorrect use of the equipment may lead to personal injury or damage to property. Always turn the equipment off and remove the mains plug when not in use. Always use the equipment as specified in this manual.

CAUTIONS

ELECTROSTATIC SENSITIVE DEVICES (ESD)

The unit contains components that are susceptible to damage through electrostatic discharge at the NanoMechanism and interface connectors. Removal of protective connector covers and connection of cables should be performed in a static safe environment using approved static safety handling procedures.

ENVIRONMENT

The unit is designed for use in-doors in a dry environmentally controlled manufacturing facility, office or laboratory. The temperature and relative humidity should be kept within those specified in Table 2.1. Significant dust or acoustic/mechanical vibration may cause faulty operation or damage to components so should be avoided. Maintain adequate cooling of the controller by not restricting the air flow to and from the fan cooling vents. For prolonged periods of operation it is advisable to keep the environmental humidity to a minimum.

DIRECTIVE AND STANDARDS APPLIED:

2004/108/EC	EMC Directive BS EN 61326-1:2006 Electrical equipment for measurement, control and laboratory use EMC requirements — Part 1: General requirements FCC part 15, subpart B
2006/95/EC	Low Voltage Directive BS EN 61010-1:2010 Safety requirements for electrical equipment for measurement, control, and laboratory use - Part 1: General requirements
2003/108/EC	WEEE Directive

Table of Contents

Related documents

Document Ref	Title	Usage
EN-014217-UM	NanoScan NPC-D Series NanoMechanism Controller - Controller Interface Library	Details the usage of controller interface DLL in order to send commands to the controller
EN-014635-UM	NanoScan NPC-D-6xx0 NanoMechanism Controller - User Manual	User manual for the 6000 series controller

Revision history

Revision	Changes
1.0	First version.
2.0	General: Reformatted for Prior Scientific and NanoScan rebranding. General: Added revision history. General: Added "Command" section for every command, to make it clearer how to construct command strings. General: Added "Supported in" section for every command, so that this document remains relevant for older versions of firmware and controller interface library software. General: Renamed "Security" section for every command to "Minimum security level", to make this clear. 2, "System overview": Substantial rework. 4.2, "Controller protection": Added details of fan control and associated commands for firmware application 6.4.1 and controller interface 2.4.1. 5.1, "Control system architecture": Substantial rework to diagrams, leading to some commands changing locations within the document. 5.2.1, "stage.position.absolute-command.get/set": Added for firmware application 6.2.12 and controller interface 2.2.7. 6, "Stepped inputs, outputs and position command": Added for firmware application 6.2.12 and controller interface 2.2.7. 7, "Function playback": Substantial rework to add details of waveform generator and associated commands for firmware application 6.4.1 and controller interface 2.4.1, and to integrate this with sample-based waveform. Some changes to sample-based commands for commonality with waveform generator. Waveform generator is now considered the main method to use when setting up function playback, and is documented as such.

Revision	Changes
	7, “Function playback”: Added “in-position” trigger output type for firmware application 6.4.1 and controller interface 2.4.1. 9, “Scope data signal selection”: Updated to support new signals available. 11, “”Deprecated commands”: Deprecated some function playback commands for sample-based waveform configuration. These have been superseded by equivalent waveform-generator-style commands. 11, “”Deprecated commands”: Added details of which versions of firmware and controller interface library software these commands were deprecated in, and which commands have replaced them.

1 Introduction

Queensgate NanoMechanisms combine a flexure-guided mechanism, a piezo actuator and a precision capacitive position sensor to allow highly-accurate position measurement. These NanoMechanisms are generally referred to as “stages”. Where multi-axis control is required, multiple stages may be assembled together, or a single stage may be designed to provide control in two or more axes.

The NanoScan NPC-D-6xx0 series digital controllers provide closed-loop positioning control for one or more stage axes. The controller incorporates a high-voltage power amplifier to drive the piezo actuator, a precision measurement circuit to calculate the stage position, digital signal processing to provide closed-loop control of the stage(s), and various interfacing methods to set the commanded position and system calibration.

To allow straightforward user control of the system, the controller interface DLL provides a standard method for a PC running Windows or Linux to interact with the controller. The user need only tell the DLL how the PC is connecting to the controller, and the DLL handles all details of the communications link. Requests/commands are provided to the DLL as text strings; the DLL translates these into the appropriate messages to/from the controller over the comms link; and responses from the controller are returned by the DLL as text strings.

Document EN-014217-UM describes the API for the DLL. This document describes the details of commands passed to the **`DoCommand()`** function from the DLL, to configure and control the digital controller’s operation, and the values passed back from the DLL for these commands.

2 System overview

2.1 Controller management

There are a number of status reports available from the controller, relating to controller sample rate, synchronisation between multiple controllers, protection against controller faults, and others. These are covered in sections 4.1 and 4.2.

2.2 Controller and stage identification

The user needs to be able to identify the controller and stage connected, and other features of the controller and stage such as the stage axis. Sections 4.3 and 4.4 cover the commands for this.

2.3 Controller security

All users must be clear that the digital controller only provides limited protection against configurations which would cause damage to a connected stage. Customers will naturally wish to configure the system behaviour to meet their own requirements, but configuration must be performed by engineers or technicians who have sufficient expertise to carry it out safely. This is primarily intended to limit accidental misconfiguration of the system. As such, full cryptographic security is not required.

A simple security system is provided which restricts the commands available to users. The controller initially has no security access allowed, which limits operators to commands which read data only. Secret values may then be used to unlock “user” and “superuser” security levels. “User” level security allows control of stage command and selection of the stage calibration preset, but does not allow stage calibration values to be changed. “Superuser” level security allows stage calibration values to be changed and these modified calibrations to be saved back to the stage. The values to unlock these security levels will be provided to customers in a separate document, so that access can be controlled as required.

Multiple users may connect to the controller over different comms links. To ensure security is maintained, security access unlocks a security level for that comms link only. Other users on other comms links do not then get access to additional commands when someone with a higher security level connects to the controller on another comms link.

Section 4.5 describes in greater detail the operation of security and associated commands.

2.4 Control system

The commanded position may be set from a channel’s analogue input or via digital commands from the host PC. Commands from the host PC may be via the comms interfaces, or may be via digital inputs emulating a stepper-motor interface. These may be used simultaneously if required, in which case commands will be summed. If a feedforward edge boost is required on command changes, high-pass filters may be used to add this.

The measured position is corrected for sensor linearity errors, and may have low-pass filtering applied to reduce noise.

The controller may be operated in open-loop or closed-loop modes. Open-loop mode allows direct drive of the actuator from the input(s), with the benefits of digital control of the open-loop gain and a digital notch filter to control resonance, as well as accurate reporting of measured position. Closed-loop mode uses a standard PID to control the measured position to the commanded position, again with a digital notch filter to control resonance. When in closed-loop mode, the “in position” status and digital output may be used for external systems to be alerted that the commanded position has been reached.

To reduce ringing on larger steps, especially with high-mass or resonant loads, feedforward limiting of the commanded position may be used. This applies trapezoidal trajectory control, in the same way as a typical motor controller. Limiting the acceleration and deceleration can greatly reduce ringing. Limiting the commanded velocity to the velocity achievable by the system can also reduce PID integral overshoot in closed-loop mode.

To reduce system noise in some measurement applications, the actuator may be frozen at the current position. This is typically used for fast measurements where actuator drift will be negligible but noise from closed-loop control may be significant.

The controller includes soft-start mechanisms to reduce sudden movements on power-on and on mode changes wherever possible.

Section 5 describes the operation of the control system and associated commands used to control, configure and calibrate the system.

2.5 Stepped inputs, outputs and position command

The controller provides stepped inputs on its “Digital I/O” connector. These allow the controller to be connected to standard quadrature or step-and-direction signals, as if it were a stepper motor controller. With this interface it is possible to carry out fast, high-resolution and low-noise control without the need for expensive DACs, enabling a considerable saving on the system cost compared to traditional analogue inputs.

The controller also provides stepped outputs on the same connector which can be used for feedback of the commanded position (to ensure commands are received correctly) or to report the measured position. Again this allows a saving on system cost compared to analogue outputs by avoiding the need for expensive ADCs.

Section 6 describes the operation of stepped inputs and outputs.

2.6 Function playback and snapshot capture

In addition to direct commands from the analogue input or host PC, the commanded position may also be set by preset waveforms stored for function playback. These may be played once, multiple times, or indefinitely. For multi-channel units, waveforms may be triggered simultaneously for multiple synchronised channels (for example to position a three-axis system), or may be run individually. Waveforms may be triggered from the host PC or from digital inputs, and may also be configured to trigger digital outputs at points during the waveform.

Snapshot capture allows sample-accurate acquisition of measured position over a short period of time, or optionally other parameters within the system, to be read back later by the host PC. Snapshots may be triggered simultaneously with waveforms or independently, either from the host PC or from digital inputs. Snapshots capture data for all channels; there is no provision for running snapshots on separate channels independently. Since a common use of snapshots is in measuring the system's step response, snapshots include the ability to generate a step during the snapshot, which is easier than setting this up with function playback. Function playback can still be used to create more complex stimuli, of course.

Commands from function playback and/or from the snapshot step command are summed with the analogue and digital commands. This allows various more sophisticated modes of operation to be configured which may allow better control of the desired response. For example, if the application is to use an XY stage to scan over a small area, function playback may be pre-programmed with a spiral or raster pattern on the channels controlling the X and Y axes. A direct command may then set the initial coordinates for the scan, and a single command will then trigger the functions for the X and Y channels when the stage reaches that point, which greatly reduces the workload instead of having to directly command every movement in the scan. This is particularly effective where the direct command for initial coordinates may come from the analogue input.

Sections 7 and 8 describe the operation of function playback and snapshot capture respectively and their associated commands.

2.7 Scope signals

In normal operation, the analogue output and snapshot capture for each channel will capture the measured position for that channel.

When calibrating/configuring the system, it may be of use to measure internal state within the control loop. The controller provides a "scope" feature which allows any signal within the control loop to be monitored directly. It also allows signals to be routed to an analogue output and/or captured during a snapshot, instead of reporting measured position for a channel. The ID numbers to select each signal within the control loop are indicated in the diagrams in section 5 using the format `123` next to the relevant data flow.

Section 9 describes the operation of the scope functionality and associated commands.

2.8 Stage/system calibration and configurations

Stage/system calibration and configurations are stored on the stage itself. Controllers and stages may therefore be interchanged with minimal change to stage linearity and scale factor. If the same closed-loop control parameters are set for each stage, stages may also be interchanged with minimal change to dynamic performance.

Stages are pre-programmed with three factory presets (fast, medium and slow closed-loop step response), which may not be changed by customers. Five further preset slots are available for customer calibrations, which may be freely altered. The user may select which preset is loaded by default on startup.

Section 10 describes the operation of stage calibration and associated commands.

3 Notation used in this document

3.1 Format of command descriptions

For clarity, notation conventions within this document must be explained.

Every command has a standard layout for common command details.

3.1.1 Command name

Command are listed by name. This is the command name which is passed to the function **`DoCommand()`**. For example, as described in 4.1.1, a *`command`* string of "`controller.sampling-time.get`" passed to **`DoCommand()`** would read back the control loop sample time.

Frequently we will need to be able to change a value and to read it back. In these cases the two commands are grouped together for clarity. For example, as described in 5.2.1, "`stage.position.command.get/set`" specifies the commands to change and read back the current commanded position. A *`command`* string of "`stage.position.command.set`" passed to **`DoCommand()`** would set the current commanded position, and "`stage.position.command.get`" passed to **`DoCommand()`** would read back the current commanded position.

3.1.2 Description

The command name attempts to be a convenient mnemonic. However some description of the command or the value to be changed is naturally required.

3.1.3 Command

This illustrates the *`command`* string passed to **`DoCommand()`**, showing the command name and parameters.

3.1.4 Parameters

Parameters form part of the *`command`* string passed to **`DoCommand()`**. They identify which value is to be changed or read back, where there is more than one possibility, and will set the new value.

Some commands need no parameters. For example, a *`command`* string of "`controller.sampling-time.get`" as described in 4.1.1 is a complete command in itself, because the system only has one control loop sampling time. In this case the parameters are explicitly specified as "None".

Most commands do need parameters. For example, the command "`stage.position.measured.get`" as described in 5.4.1 requires the channel number to be specified, otherwise it is unclear which channel's position is to be reported. A *`command`* string of "`stage.position.measured.get 2`" will read back the position for the stage on channel 2.

Some commands need more than one parameter. For example, the command "`function.command.start`" as described in 7.9.2 must specify for all channels whether they are to be started or not. In this case the *`command`* string must set each parameter in the order specified, separated by a space. A *`command`* string of "`function.command.start 1 0 0 1 1`" will start a snapshot and function playback on channels 2 and 3.

Where we have commands to change a value and to read it back, the two commands will usually need different parameters. These are specified in separate sections as "Parameters (get)" and "Parameters (set)", or whatever suffix is appropriate for the commands. For example, as described in 5.2.1, "`stage.position.command.get/set`" specifies the commands to change and read back the current commanded position. A *`command`* string of "`stage.position.command.set 2 3500`" passed to **`DoCommand()`** would set the current commanded position for channel 2 to 3500pm, and "`stage.position.command.get 2`" passed to **`DoCommand()`** would read back the current commanded position for channel 2.

Further details are generally required for parameters. Where values must be in a defined range, the minimum and/or maximum range is specified. Some parameters may be integers or floating-point, which will affect whether fractional values are expected or not. Some parameters may also be specified as text rather than numeric values, in which case the relevant text format is described.

If the *`command`* string contains too few parameters for a command, **`DoCommand()`** returns an error (negative value). If the *`command`* string contains more parameters than required for a command, additional parameters are silently ignored.

3.1.5 Results

If a command is sent successfully to the controller and returns data, the function **`DoCommand()`** returns the number of result values passed back by the controller for this command. When changing a value, this will usually mirror back the value changed; or when reading back a value, this will report the value. Some commands may report more than one result.

The function **`GetResultName()`** reads back the name of each result, and the function **`GetResult()`** reads back the result value.

Note that a command may be sent to the controller but may then fail, for example if parameters are out of range. Errors are reported using a standard format of results, as described in 3.2. If the result name for index 0 is "`error`", the user knows the command failed. Common errors for all commands are described in 3.2.

If a command succeeds, the result name and results will be as specified in the "Results" section for the command. For example, as described in 4.1.1, a *`command`* string of "`controller.sampling-time.get`" passed to **`DoCommand()`** would return a value of 1, indicating that there is only one result. Calling **`GetResultName()`** for index 0 would report "`value`" as the result name, and calling **`GetResult()`** for index 0 would report a value of "`2.0e-5`" on an NPC-6000 series controller, which has a 20µs sample period when running at 50kHz.

Where "get" and "set" commands are specified together, typically the results will be the same for both. The "set" command will report a result which mirrors the value set (or will report an error if it cannot be set), and the "get" command will report the value currently set.

Some commands report more than one result value, for example "`controller.status.get`" as described in 4.1.2. The order of results for **`GetResultName()`** and **`GetResult()`** will always match the order specified, so for example for "controller.status.get", result index 0 will report "`security`", result index 1 will report "`channels`", and result index 2 will report "`status`". It is always best practise to check the name of the first result to ensure that a command has been carried out successfully, but results for a successful command will always be in the correct order, so it is not necessary to check the result names for subsequent results.

3.1.6 Possible error reports

The common error result format and common errors for all commands are described in 3.2. Each command will also typically have errors which are specific to the command: for example, the channel number may be invalid, or the commanded value may be out of range.

For each command, the possible error reports in the "`errcode`" result (index 1) are described. The user may assume that only error reports from 3.2 or from this section of the command description will occur.

Where "get" and "set" commands are specified together, typically there will be different error cases for the two commands. In this case the error reports for each are described separately.

3.1.7 Minimum security level

Many commands which only return values do not require security and will always be available. It is usually possible to observe system operation without security.

To avoid accidental modification of key settings though, many commands require a minimum security level to be carried out, as described in 2.3 and 4.5. Where this is required, it is specified for each command.

Where "get" and "set" commands are specified together, typically there will be different security levels required for the two commands. In this case these are specified separately.

3.1.8 Supported in

New features are being continuously developed for the controller. This manual documents features and commands for the latest versions of controller firmware and controller interface library software. In case users have not yet upgraded to the latest firmware or software, it is important that this document specifies the minimum versions of firmware and software necessary to use a particular command.

3.2 Error reporting

3.2.1 Error result fields

As described in 3.1.5, unsuccessful commands do not report the results specified for each command. Instead, all unsuccessful commands report the following results fields to report an error.

Name	Type	Units	Minimum	Maximum	Description
error	string				Always set to “FAILED”
errcode	string				Error code text

3.2.2 Error code text

The following standard error code text strings may be reported for every command.

Error result “`errcode`” string reported	Description
Command invalid	The command is not valid for this firmware. Typically the interface DLL will prevent this happening, by concealing invalid commands from the user.
Command locked by security	The command is not valid for the security permissions unlocked (see section 4.5). Typically the interface DLL will prevent this happening, by concealing not-permitted commands from the user.
Command not available	The command is intended to be implemented for a later version of firmware, but is not currently implemented. Typically this will be seen for beta firmware during development. It will not normally been seen in production firmware.

In addition, every command has error codes specific to that command, as described in 3.1.6.

4 Controller

4.1 Controller system

4.1.1 controller.sampling-time.get

4.1.1.1 Description

Control loop sample time

4.1.1.2 Command

```
controller.sampling-time.get
```

4.1.1.3 Parameters

None

4.1.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s			Control loop sample time

4.1.1.5 Possible error reports

None

4.1.1.6 Minimum security level

controller.sampling-time.get	No security required

4.1.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.1.2 controller.status.get

4.1.2.1 Description

Controller system status

4.1.2.2 Command

```
controller.status.get
```

4.1.2.3 Parameters

None

4.1.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
security	string	enum			Current controller security level "None" : No security "User": User level security "Superuser": Superuser level security
channels	8-bit unsigned integer				Number of controller channels
status	16-bit unsigned integer	Bitfield			Reserved

4.1.2.5 Possible error reports

None

4.1.2.6 Minimum security level

controller.status.get	No security required

4.1.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.1.3 controller.synchronisation.master.get

4.1.3.1 Description

Controller is synchronisation master, or controller synchronisation cable is not connected

4.1.3.2 Command

```
controller.synchronisation.master.get
```

4.1.3.3 Parameters

None

4.1.3.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Controller is synchronisation master

4.1.3.5 Possible error reports

None

4.1.3.6 Minimum security level

controller.synchronisation.master.get	No security required

4.1.3.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.1.4 controller.synchronisation.slave.get

4.1.4.1 Description

Controller is synchronisation slave and is locked to the master

4.1.4.2 Command

```
controller.synchronisation.slave.get
```

4.1.4.3 Parameters

None

4.1.4.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Controller is synchronisation slave

4.1.4.5 Possible error reports

None

4.1.4.6 Minimum security level

controller.synchronisation.slave.get	No security required

4.1.4.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.2 Controller protection

The controller firmware contains safety features which are designed to prevent the controller presenting a hazard to the user. In order to achieve this, the controller has temperature sensors on the internal power supply electronics and on the heatsinks for the piezo actuator output electronics.

The controller has cooling fans which ensure internal temperatures are not excessive. Passive cooling is sufficient when stages are not being driven fast, so the controller is normally configured to only turn the fans on when temperatures become elevated. The acoustic noise from fans may be unacceptable in some applications though, so the user has the option to disable the cooling fans.

In the event that high power is drawn for a long period and the controller overheats, the firmware will immediately carry out a protection shutdown and cut power to all actuators. The front panel LEDs for all channels will flash red to alert the user. The cooling fans will prevent this occurring in normal use, but protection is required in case the user has disabled the fans or the fans are faulty.

When all temperatures have returned to a safe operating range, the controller will automatically restart. All connected stages will have their default configuration loaded and will restart as if the controller had just been powered on.

Note that any stage configuration changed before the protection shutdown will be lost. This is a safety measure, because it is possible that a channel may overheat due to a misconfiguration causing the stage to resonate. Returning to the default configuration should ensure that the controller resumes operation in a safe state.

4.2.1 protection.fan.mode.get/set

4.2.1.1 Description

Fan mode.

When this is set, the setting is stored to the controller and is retrieved at next power-on.

4.2.1.2 Command

```
protection.fan.mode.get
```

```
protection.fan.mode.set <value>
```

4.2.1.3 Parameters (get)

None

4.2.1.4 Parameters (set)

Name	**Type**	**Units**	**Minimum**	**Maximum**	**Description**
value	string	enum			Fan mode “off” : Fan permanently off “on”: Fan permanently on “auto”: Fan turned on and off automatically based on temperature

4.2.1.5 Results

Name	**Type**	**Units**	**Minimum**	**Maximum**	**Description**
value	string	enum			Fan mode “off” : Fan permanently off “on”: Fan permanently on “auto”: Fan turned on and off automatically based on temperature

4.2.1.6 Possible error reports

None

4.2.1.7 Minimum security level

protection.fan.mode.get	No security required
protection.fan.mode.set	Superuser level security required

4.2.1.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

4.2.2 protection.fan.state.get

4.2.2.1 Description

Fan state (on/off)

4.2.2.2 Command

```
protection.fan.state.get
```

4.2.2.3 Parameters

None

4.2.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	boolean	0	1	Fan state 0 = off 1 = on

4.2.2.5 Possible error reports

None

4.2.2.6 Minimum security level

protection.fan.state.get	No security required

4.2.2.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

4.2.3 protection.fan.psu.on-temperature.get/set

4.2.3.1 Description

In auto fan mode, PSU temperature at which fan turns on.

When this is set, the setting is stored to the controller and is retrieved at next power-on.

4.2.3.2 Command

```
protection.fan.psu.on-temperature.get
```

```
protection.fan.psu.on-temperature.set <value>
```

4.2.3.3 Parameters (get)

None

4.2.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	°C			PSU temperature at which fan turns on

4.2.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	°C			PSU temperature at which fan turns on

4.2.3.6 Possible error reports

None

4.2.3.7 Minimum security level

protection.fan.psu.on-temperature.get	No security required
protection.fan.psu.on-temperature.set	Superuser level security required

4.2.3.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

4.2.4 protection.fan.psu.off-temperature.get/set

4.2.4.1 Description

In auto fan mode, PSU temperature at which fan turns off.

When this is set, the setting is stored to the controller and is retrieved at next power-on.

4.2.4.2 Command

```
protection.fan.psu.off-temperature.get
```

```
protection.fan.psu.off-temperature.set <value>
```

4.2.4.3 Parameters (get)

None

4.2.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	°C			PSU temperature at which fan turns off

4.2.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	°C			PSU temperature at which fan turns ooff

4.2.4.6 Possible error reports

None

4.2.4.7 Minimum security level

protection.fan.psu.off-temperature.get	No security required
protection.fan.psu.off-temperature.set	Superuser level security required

4.2.4.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

4.2.5 protection.fan.heatsink.on-temperature.get/set

4.2.5.1 Description

In auto fan mode, heatsink temperature at which fan turns on.

When this is set, the setting is stored to the controller and is retrieved at next power-on.

4.2.5.2 Command

```
protection.fan.heatsink.on-temperature.get
```

```
protection.fan.heatsink.on-temperature.set <value>
```

4.2.5.3 Parameters (get)

None

4.2.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	°C			Heatsink temperature at which fan turns on

4.2.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	°C			Heatsink temperature at which fan turns on

4.2.5.6 Possible error reports

None

4.2.5.7 Minimum security level

protection.fan.heatsink.on-temperature.get	No security required
protection.fan.heatsink.on-temperature.set	Superuser level security required

4.2.5.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

4.2.6 protection.fan.heatsink.off-temperature.get/set

4.2.6.1 Description

In auto fan mode, heatsink temperature at which fan turns off.

When this is set, the setting is stored to the controller and is retrieved at next power-on.

4.2.6.2 Command

```
protection.fan.heatsink.off-temperature.get
```

```
protection.fan.heatsink.off-temperature.set <value>
```

4.2.6.3 Parameters (get)

None

4.2.6.4 Parameters (set)

Name	**Type**	**Units**	**Minimum**	**Maximum**	**Description**
value	32-bit floating-point	°C			Heatsink temperature at which fan turns off

4.2.6.5 Results

Name	**Type**	**Units**	**Minimum**	**Maximum**	**Description**
value	32-bit floating-point	°C			Heatsink temperature at which fan turns ooff

4.2.6.6 Possible error reports

None

4.2.6.7 Minimum security level

protection.fan.heatsink.off-temperature.get	No security required
protection.fan.heatsink.off-temperature.set	Superuser level security required

4.2.6.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

4.2.7 protection.thermal.status.get

4.2.7.1 Description

Status of thermal cutout protection

4.2.7.2 Command

```
protection.thermal.status.get
```

4.2.7.3 Parameters

None

4.2.7.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Bitfield			Bit 0: Protection shutdown triggered by channel 1 heatsink overtemperature Bit 1: Protection shutdown triggered by channel 2 heatsink overtemperature Bit 2: Protection shutdown triggered by channel 3 heatsink overtemperature Bit 7: Protection shutdown triggered by internal power supply overtemperature

4.2.7.5 Possible error reports

None

4.2.7.6 Minimum security level

protection.thermal.status.get	No security required

4.2.7.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.8 protection.thermal.heatsink.temperature.get

4.2.8.1 Description

Heatsink temperature

4.2.8.2 Command

```
protection.thermal.heatsink.temperature.get <channel>
```

4.2.8.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

4.2.8.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	°C			Heatsink temperature

4.2.8.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

4.2.8.6 Minimum security level

protection.thermal.heatsink.temperature.get	No security required

4.2.8.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.9 protection.thermal.heatsink.overtemperature-detect-threshold.get

4.2.9.1 Description

Heatsink overtemperature detect threshold

4.2.9.2 Command

```
protection.thermal.heatsink.overtemperature-detect-threshold.get
<channel>
```

4.2.9.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

4.2.9.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	°C			Detect threshold

4.2.9.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

4.2.9.6 Minimum security level

protection.thermal.heatsink.overtemperature-detect-threshold.get	No security required

4.2.9.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.10 protection.thermal.heatsink.overtemperature-detect-time.get

4.2.10.1 Description

Heatsink overtemperature detect time before cutout

4.2.10.2 Command

```
protection.thermal.heatsink.overtemperature-detect-time.get <channel>
```

4.2.10.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

4.2.10.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	s	0		Detect time before cutout

4.2.10.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

4.2.10.6 Minimum security level

protection.thermal.heatsink.overtemperature-detect-time.get	No security required

4.2.10.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.11 protection.thermal.heatsink.overtemperature-clear-threshold.get

4.2.11.1 Description

Heatsink overtemperature clear threshold

4.2.11.2 Command

```
protection.thermal.heatsink.overtemperature-clear-threshold.get
<channel>
```

4.2.11.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

4.2.11.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	°C			Clear threshold

4.2.11.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

4.2.11.6 Minimum security level

protection.thermal.heatsink.overtemperature-clear-threshold.get	No security required

4.2.11.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.12 protection.thermal.heatsink.overtemperature-clear-time.get

4.2.12.1 Description

Heatsink overtemperature detect threshold

4.2.12.2 Command

```
protection.thermal.heatsink.overtemperature-clear-time.get <channel>
```

4.2.12.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

4.2.12.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	°C			Clear time before restart

4.2.12.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

4.2.12.6 Minimum security level

protection.thermal.heatsink.overtemperature-clear-time.get	No security required

4.2.12.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.13 protection.thermal.psu.temperature.get

4.2.13.1 Description

Internal power supply temperature

4.2.13.2 Command

```
protection.thermal.psu.temperature.get
```

4.2.13.3 Parameters

None

4.2.13.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	°C			Power supply temperature

4.2.13.5 Possible error reports

None

4.2.13.6 Minimum security level

protection.thermal.psu.temperature.get	No security required

4.2.13.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.14 protection.thermal.psu.overtemperature-detect-threshold.get

4.2.14.1 Description

Internal power supply overtemperature detect threshold

4.2.14.2 Command

```
protection.thermal.psu.overtemperature-detect-threshold.get
```

4.2.14.3 Parameters

None

4.2.14.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	°C			Detect threshold

4.2.14.5 Possible error reports

None

4.2.14.6 Minimum security level

protection.thermal.psu.overtemperature-detect-threshold.get	No security required

4.2.14.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.15 protection.thermal.psu.overtemperature-detect-time.get

4.2.15.1 Description

Internal power supply overtemperature detect time before cutout

4.2.15.2 Command

```
protection.thermal.psu.overtemperature-detect-time.get
```

4.2.15.3 Parameters

None

4.2.15.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	s	0		Detect time before cutout

4.2.15.5 Possible error reports

None

4.2.15.6 Minimum security level

protection.thermal.psu.overtemperature-detect-time.get	No security required

4.2.15.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.16 protection.thermal.psu.overtemperature-clear-threshold.get

4.2.16.1 Description

Internal power supply overtemperature clear threshold

4.2.16.2 Command

```
protection.thermal.psu.overtemperature-clear-threshold.get
```

4.2.16.3 Parameters

None

4.2.16.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	°C			Clear threshold

4.2.16.5 Possible error reports

None

4.2.16.6 Minimum security level

protection.thermal.psu.overtemperature-clear-threshold.get	No security required

4.2.16.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.2.17 protection.thermal.psu.overtemperature-clear-time.get

4.2.17.1 Description

Internal power supply overtemperature detect threshold

4.2.17.2 Command

```
protection.thermal.psu.overtemperature-clear-time.get
```

4.2.17.3 Parameters

None

4.2.17.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating point	°C			Clear time before restart

4.2.17.5 Possible error reports

None

4.2.17.6 Minimum security level

protection.thermal.psu.overtemperature-clear-time.get	No security required

4.2.17.7 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

4.3 Controller identification

4.3.1 identity.hardware.part.get

4.3.1.1 Description

Controller part number

4.3.1.2 Command

```
identity.hardware.part.get
```

4.3.1.3 Parameters

None

4.3.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
part	String				Controller part number

4.3.1.5 Possible error reports

None

4.3.1.6 Minimum security level

identity.hardware.part.get	No security required

4.3.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.3.2 identity.hardware.serial.get

4.3.2.1 Description

Controller serial number

4.3.2.2 Command

```
identity.hardware.serial.get
```

4.3.2.3 Parameters

None

4.3.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
serial	32-bit unsigned integer				Controller serial number

4.3.2.5 Possible error reports

None

4.3.2.6 Minimum security level

identity.hardware.serial.get	No security required

4.3.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.3.3 identity.hardware.mandate.get

4.3.3.1 Description

Controller manufacture date

4.3.3.2 Command

```
identity.hardware.mandate.get
```

4.3.3.3 Parameters

None

4.3.3.4 Results

Name	Type	Units	Minimum	Maximum	Description
mandate	32-bit unsigned integer	MS Excel serial day			Controller manufacture date

4.3.3.5 Possible error reports

None

4.3.3.6 Minimum security level

identity.hardware.mandate.get	No security required

4.3.3.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.3.4 identity.hardware.caldate.get

4.3.4.1 Description

Controller calibration date

4.3.4.2 Command

```
identity.hardware.caldate.get
```

4.3.4.3 Parameters

None

4.3.4.4 Results

Name	Type	Units	Minimum	Maximum	Description
caldate	32-bit unsigned integer	MS Excel serial day			Controller calibration date

4.3.4.5 Possible error reports

None

4.3.4.6 Minimum security level

identity.hardware.caldate.get	No security required

4.3.4.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.3.5 identity.hardware.bootloader-version.get

4.3.5.1 Description

Controller software bootloader version number

4.3.5.2 Command

```
identity.hardware.bootloader-version.get
```

4.3.5.3 Parameters

None

4.3.5.4 Results

Name	Type	Units	Minimum	Maximum	Description
version	32-bit unsigned integer				Version number Bits 31-24: Major version Bits 23-16: Minor version Bits 15-0: Build number

4.3.5.5 Possible error reports

None

4.3.5.6 Minimum security level

identity.hardware.bootloader-version.get	No security required

4.3.5.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.3.6 identity.hardware.platform-version.get

4.3.6.1 Description

Controller software platform version number

4.3.6.2 Command

```
identity.hardware.platform-version.get
```

4.3.6.3 Parameters

None

4.3.6.4 Results

Name	Type	Units	Minimum	Maximum	Description
version	32-bit unsigned integer				Version number Bits 31-24: Major version Bits 23-16: Minor version Bits 15-0: Build number

4.3.6.5 Possible error reports

None

4.3.6.6 Minimum security level

identity.hardware.platform-version.get	No security required

4.3.6.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.3.7 identity.software.version.get

4.3.7.1 Description

Control application software version number

4.3.7.2 Command

```
identity.software.version.get
```

4.3.7.3 Parameters

None

4.3.7.4 Results

Name	Type	Units	Minimum	Maximum	Description
version	32-bit unsigned integer				Version number Bits 31-24: Major version Bits 23-16: Minor version Bits 15-0: Build number

4.3.7.5 Possible error reports

None

4.3.7.6 Minimum security level

identity.software.version.get	No security required

4.3.7.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.3.8 identity.software.reldate.get

4.3.8.1 Description

Control application software release date

4.3.8.2 Command

```
identity.software.reldate.get
```

4.3.8.3 Parameters

None

4.3.8.4 Results

Name	Type	Units	Minimum	Maximum	Description
caldate	32-bit unsigned integer	MS Excel serial day			Software release date

4.3.8.5 Possible error reports

None

4.3.8.6 Minimum security level

identity.software.reldate.get	No security required

4.3.8.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.3.9 identity.software.part.get

4.3.9.1 Description

Control application software part number

4.3.9.2 Command

```
identity.software.part.get
```

4.3.9.3 Parameters

None

4.3.9.4 Results

Name	Type	Units	Minimum	Maximum	Description
caldate	32-bit unsigned integer				Software part number

4.3.9.5 Possible error reports

None

4.3.9.6 Minimum security level

identity.software.part.get	No security required

4.3.9.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.4 Stage identification

4.4.1 identity.stage.part.get

4.4.1.1 Description

Stage part number

4.4.1.2 Command

```
identity.stage.part.get <channel>
```

4.4.1.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

4.4.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
part	String				Stage part number

4.4.1.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

4.4.1.6 Minimum security level

identity.stage.part.get	No security required

4.4.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.4.2 identity.stage.serial.get

4.4.2.1 Description

Stage serial number

4.4.2.2 Command

```
identity.stage.serial.get <channel>
```

4.4.2.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

4.4.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
part	32-bit unsigned integer				Stage serial number

4.4.2.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

4.4.2.6 Minimum security level

identity.stage.serial.get	No security required

4.4.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.4.3 identity.stage.axisid.get

4.4.3.1 Description

Stage axis for this controller channel

4.4.3.2 Command

```
identity.stage.axisid.get <channel>
```

4.4.3.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

4.4.3.4 Results

Name	Type	Units	Minimum	Maximum	Description
axisid	String				Stage axis for this controller channel: “x”, “y”, “z”, “theta”, “gamma”, “phi” or “unspecified”

Note that the axis will typically be reported as “unspecified” for single-axis stages. This is primarily intended for axis identification on multi-axis systems.

4.4.3.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

4.4.3.6 Minimum security level

identity.stage.axisid.get	No security required

4.4.3.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.4.4 identity.stage.mandate.get

4.4.4.1 Description

Stage manufacture date

4.4.4.2 Command

```
identity.stage.mandate.get <channel>
```

4.4.4.3 Parameters

Name	**Type**	**Units**	**Minimum**	**Maximum**	**Description**
channel	8-bit unsigned integer				Controller channel for stage

4.4.4.4 Results

Name	**Type**	**Units**	**Minimum**	**Maximum**	**Description**
mandate	32-bit unsigned integer	MS Excel serial day			Stage manufacture date

4.4.4.5 Possible error reports

Error return "errcode" string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

4.4.4.6 Minimum security level

identity.stage.mandate.get	No security required

4.4.4.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.4.5 identity.stage.caldate.get

4.4.5.1 Description

Stage calibration date

4.4.5.2 Command

```
identity.stage.caldate.get <channel>
```

4.4.5.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

4.4.5.4 Results

Name	Type	Units	Minimum	Maximum	Description
caldate	32-bit unsigned integer	MS Excel serial day			Stage calibration date

4.4.5.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

4.4.5.6 Minimum security level

identity.stage.caldate.get	No security required

4.4.5.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.5 Security access and security levels

The controller implements a simple form of security, where each security level is unlocked by entering a secret “key” value corresponding to the security level. This is not intended to prevent determined attacks, but is simply intended to stop accidental changes to the configuration which would adversely affect the system behaviour.

Security is implemented on a per-comms-link basis. It is possible to have multiple operators connected to the same controller on different comms links. In the case of Ethernet comms it is possible to have multiple operators connected over the same comms link, and in this case security is implemented on a per-connection basis.

The intention is that one operator could be in control, with “user” or “superuser” permissions. Other operators without security permissions set would be able to observe the system, but would be prevented from accidentally interfering with its behaviour. This provides a simple level of security if the controller is required to be monitored by a SCADA system.

A further intention is that an operator with “user” permissions may control stage movement, but are prevented from changing parameters which may affect system stability. “User” level operators may load preset calibrations (see section 10) but may not themselves change settings from those calibrations. “Superuser” level operators may configure calibrations to tune the system, and save those calibrations to the stage for use by “users”. This provides a level of security if the controller will typically be operated by software on a production line, where users may not have the necessary skills or equipment to change calibrations safely.

To provide some limited protection against users brute-forcing key values, entering an invalid “key” value removes any current “user” or “superuser” permissions, and prevents further unlock requests for 5s.

The relevant “key” values are:-

Security level	**Key value**
“User” level security	233573869 (hex: 0xDEC0DED)
“Superuser” level security	2954754766 (hex: 0xB01DFACE)

4.5.1 controller.security.user.get/set

4.5.1.1 Description

Controller security level

4.5.1.2 Command

```
controller.security.user.get
```

```
controller.security.user.set <code>
```

4.5.1.3 Parameters (get)

None

4.5.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
code	32-bit unsigned integer				Key value to unlock desired security level

4.5.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
security	string	enum			Current controller security level “None” : No security “User”: User level security “Superuser”: Superuser level security

Note that after a “set” request with an invalid command unlock code, security will always return to level “None”.

4.5.1.6 Possible error reports

Error return “errcode” string reported	Description
Wait for 5s after invalid command unlock code	A previous request to unlock a security level was unsuccessful, and another request has been made before the 5s timeout has elapsed

4.5.1.7 Minimum security level

controller.security.user.get	No security required
controller.security.user.set	No security required

4.5.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

4.5.2 controller.security.lock

4.5.2.1 Description

Lock controller security

4.5.2.2 Command

```
controller.security.lock
```

4.5.2.3 Parameters

None

4.5.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
security	string	enum			Current controller security level Always returns “None” : No security

4.5.2.5 Possible error reports

None

4.5.2.6 Minimum security level

controller.security.lock	No security required

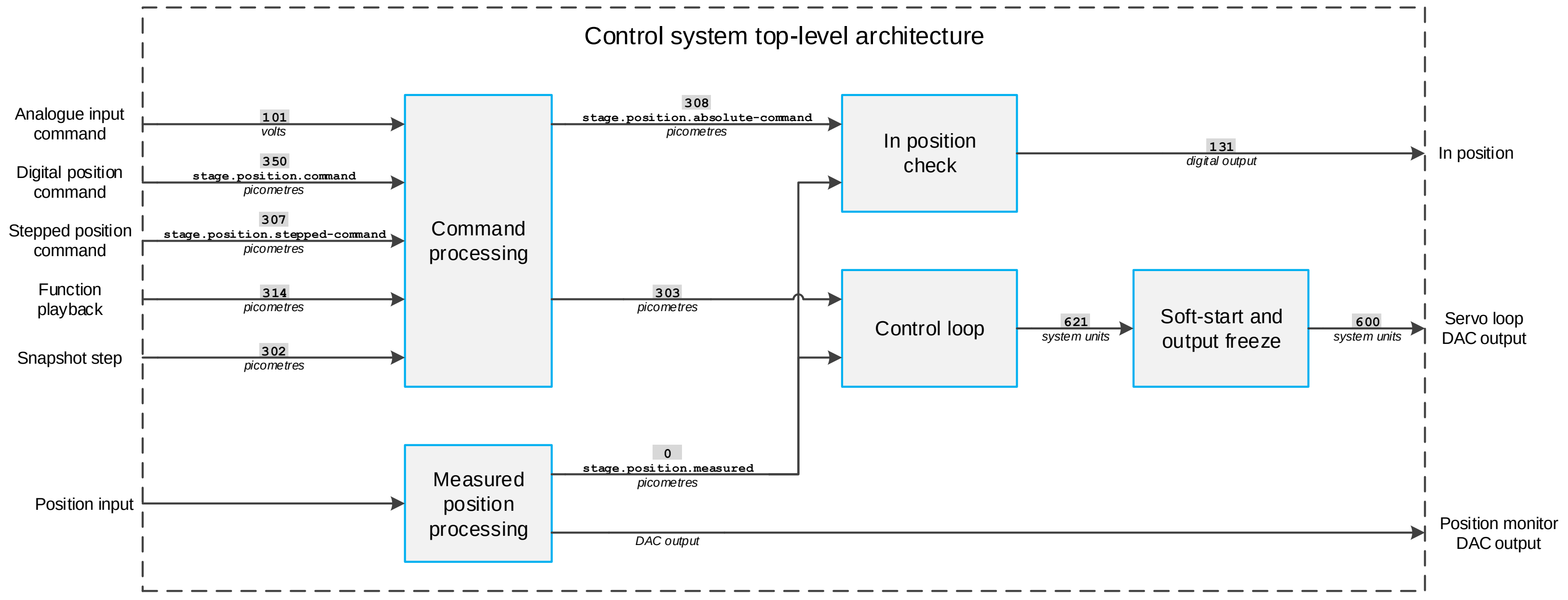
4.5.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

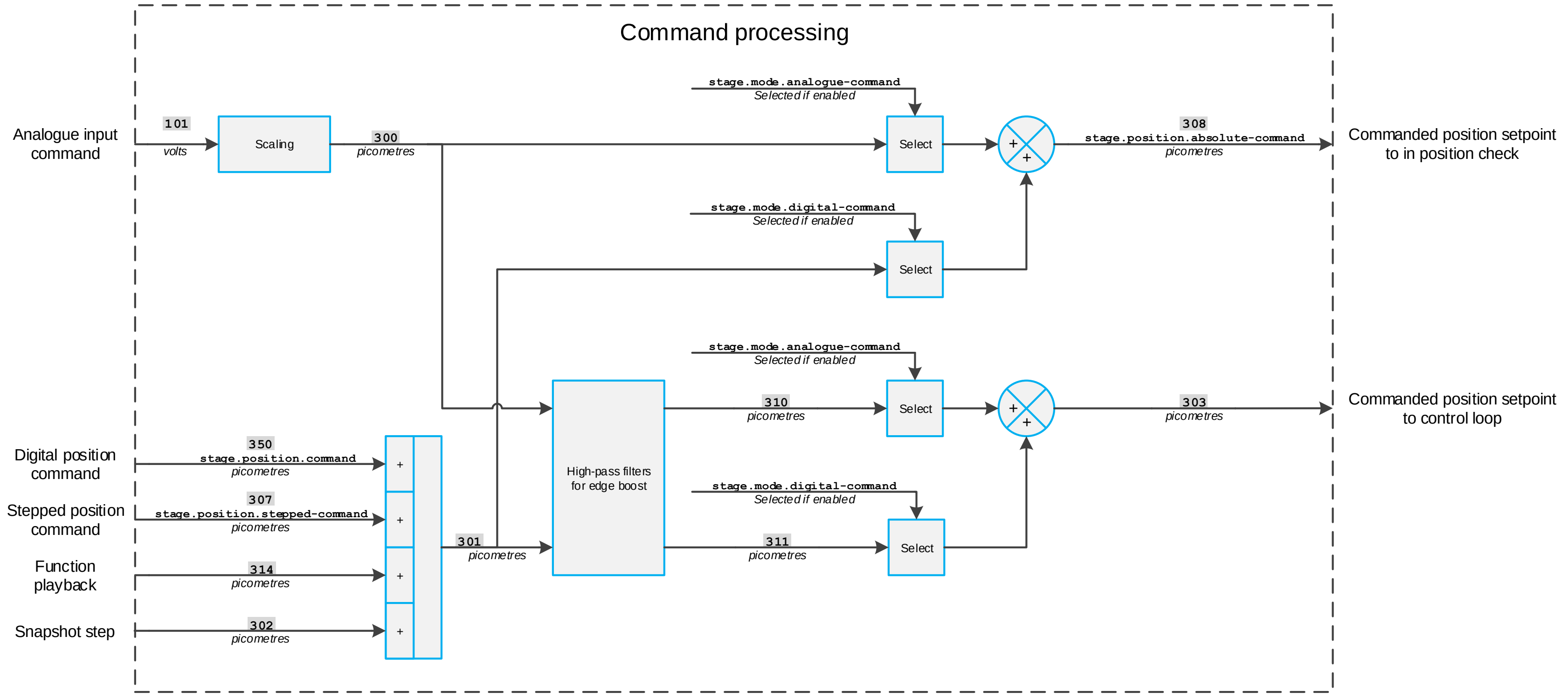
5 Control system

5.1 Control system architecture

5.1.1 Control system top-level architecture



5.1.2 Command processing

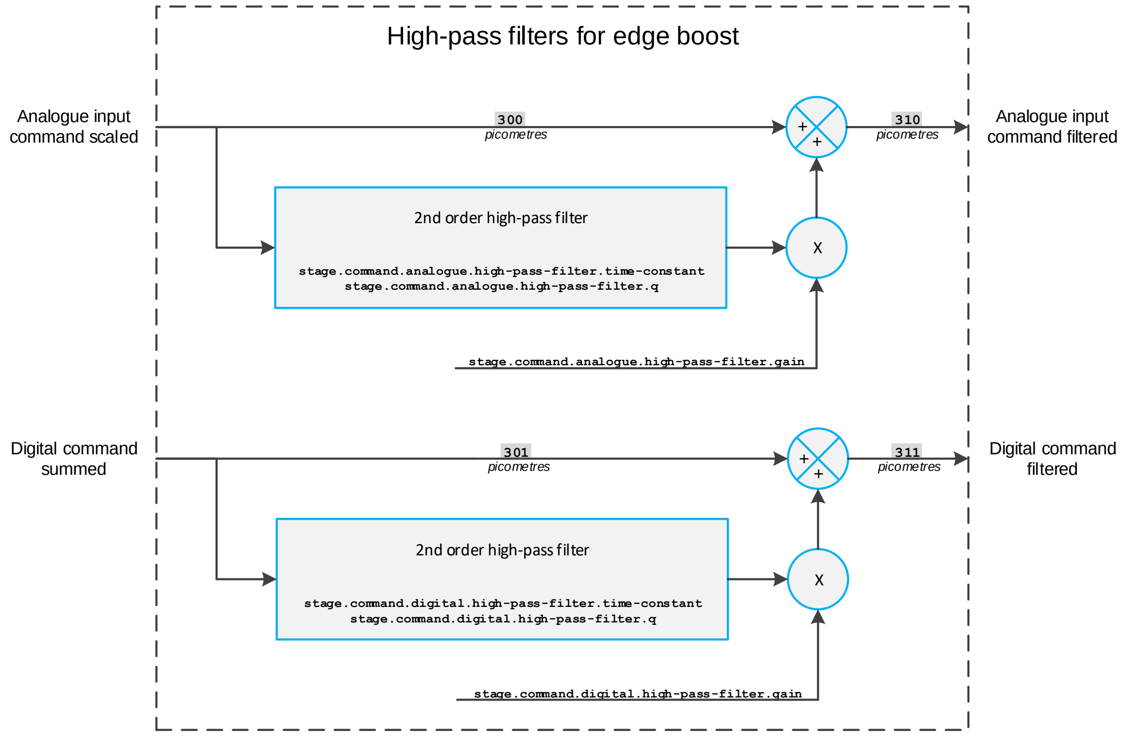


See 6.2 for details of stepped position command, 7 for details of function playback, and 8 for details of snapshot step. These require additional explanation which will not be duplicated here.

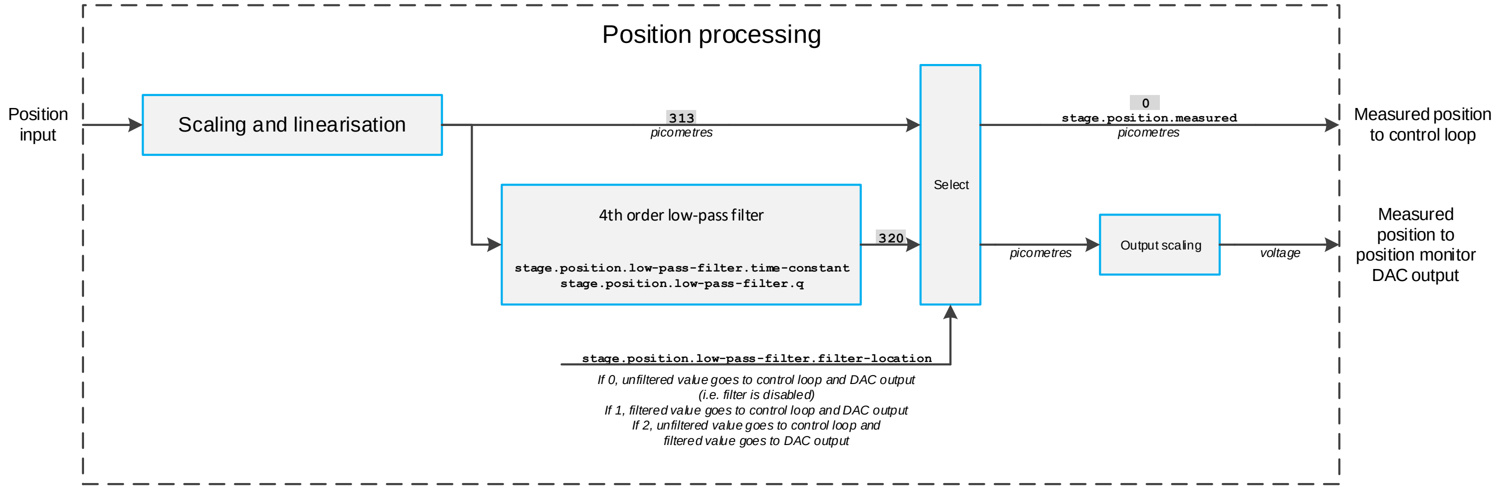
The value **`stage.position.absolute-command`** may be set as well as read. When this is set, the digital position command is set to the difference between this value and the sum of all other (enabled) commands, so that the overall commanded position setpoint is as specified.

Because the analogue input is inherently noisy, the edge boost filters (if enabled) may require different settings for analogue and digital commands. Note that the absolute command used by the in position check only considers the input commands and does not include the edge boost.

5.1.3 High-pass filters for edge boost

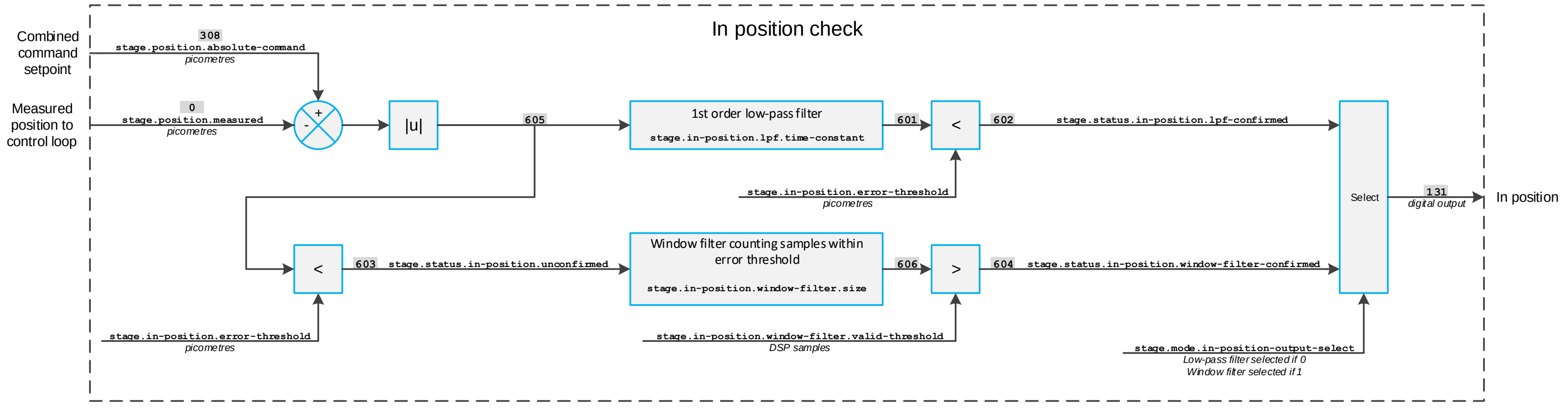

High-pass filters for edge boost
Analogue input command scaled
300
picometres
+
+
310
picometres
Analogue input command filtered
2nd order high-pass filter
stage.command.analogue.high-pass-filter.time-constant
stage.command.analogue.high-pass-filter.q
X
stage.command.analogue.high-pass-filter.gain
Digital command summed
301
picometres
+
+
311
picometres
Digital command filtered
2nd order high-pass filter
stage.command.digital.high-pass-filter.time-constant
stage.command.digital.high-pass-filter.q
X
stage.command.digital.high-pass-filter.gain

5.1.4 Measured position processing


Position processing
Position input
Scaling and linearisation
313
picometres
4th order low-pass filter
stage.position.low-pass-filter.time-constant
stage.position.low-pass-filter.q
320
Select
0
stage.position.measured
picometres
Measured position to control loop
picometres
Output scaling
voltage
Measured position to position monitor DAC output
stage.position.low-pass-filter.filter-location
If 0, unfiltered value goes to control loop and DAC output (i.e. filter is disabled)
If 1, filtered value goes to control loop and DAC output
If 2, unfiltered value goes to control loop and filtered value goes to DAC output

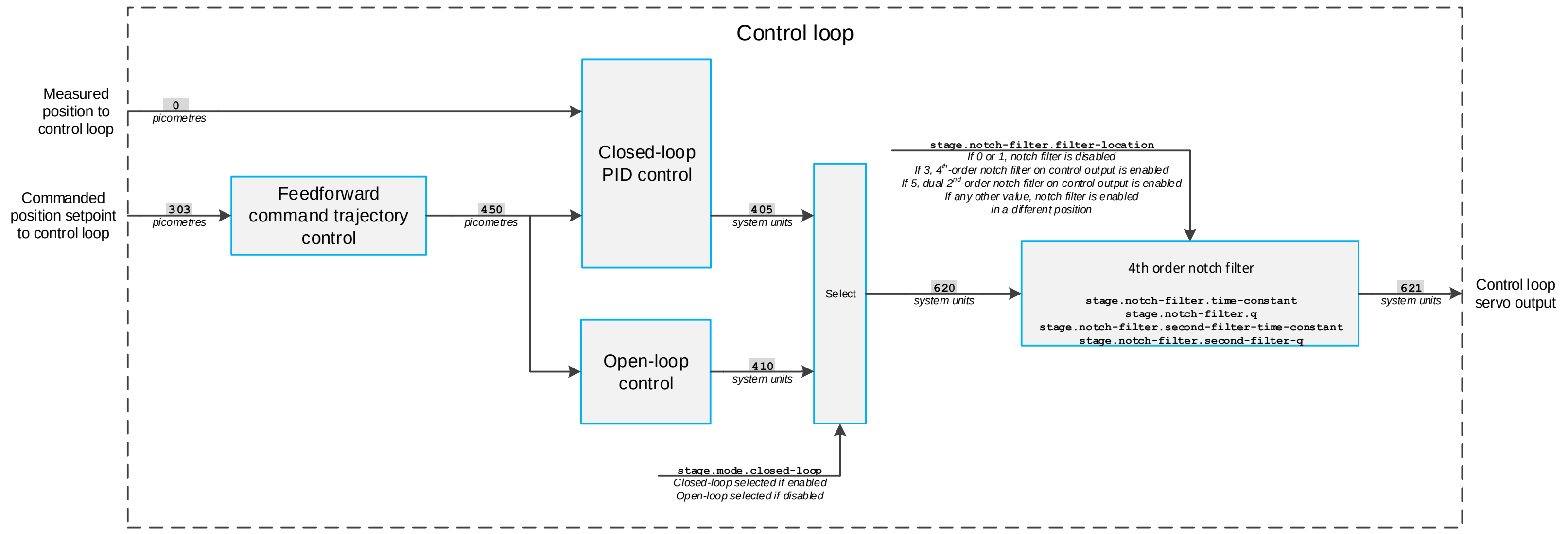
Note that the output scaling from a position in picometres to a voltage on the position monitor output will vary depending on the stage. This scaling is specified in the stage datasheet.

5.1.5 In position check

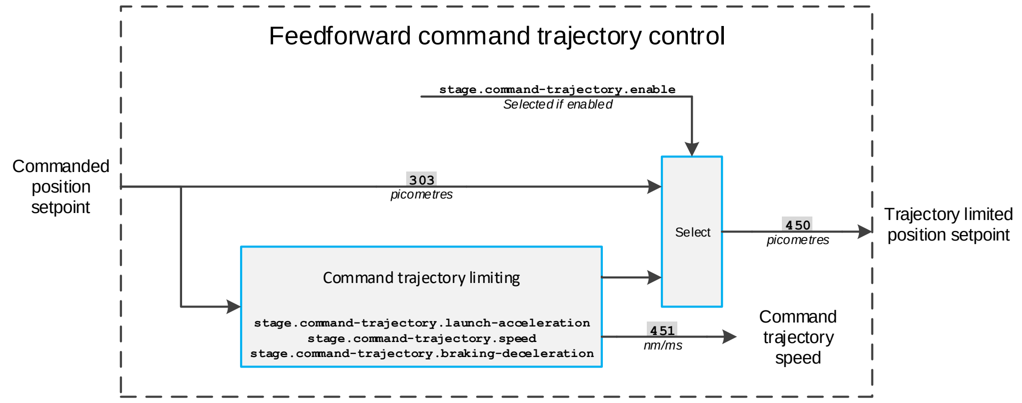

In position check
Combined command setpoint
308
stage.position.absolute-command
picometres
Measured position to control loop
0
stage.position.measured
picometres
+
-
|u|
605
1st order low-pass filter
stage.in-position.lpf.time-constant
601
<
602
stage.status.in-position.lpf-confirmed
stage.in-position.error-threshold
picometres
<
603
stage.status.in-position.unconfirmed
stage.in-position.error-threshold
picometres
Window filter counting samples within error threshold
stage.in-position.window-filter.size
606
>
604
stage.status.in-position.window-filter-confirmed
stage.in-position.window-filter.valid-threshold
DSP samples
Select
131
digital output
In position
stage.mode.in-position-output-select
Low-pass filter selected if 0
Window filter selected if 1

The in position ("ready") check only runs when the control loop for the channel is in closed-loop mode (see 5.11.3, **`stage.mode.closed-loop`**). In open-loop mode, the command does not relate to an actual position, so the in position status/output is always false or logic-low in open-loop mode.

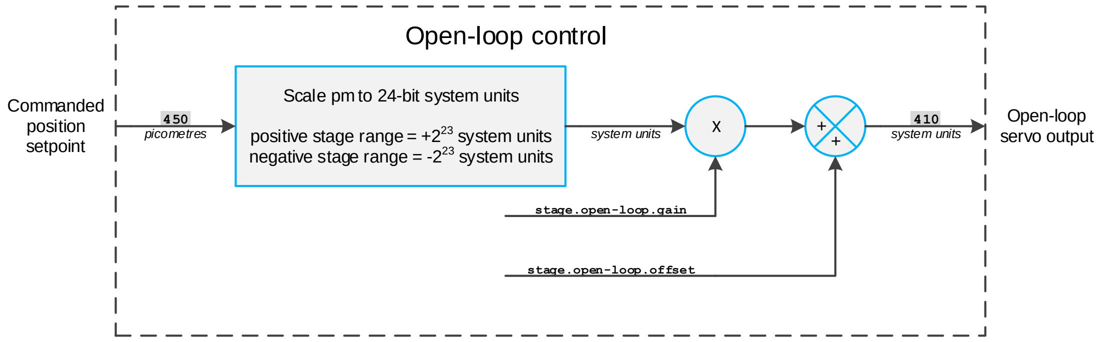
5.1.6 Control loop


Control loop
Measured position to control loop
0
picometres
Commanded position setpoint to control loop
303
picometres
Feedforward command trajectory control
450
picometres
Closed-loop PID control
Open-loop control
405
system units
410
system units
Select
stage.mode.closed-loop
Closed-loop selected if enabled
Open-loop selected if disabled
stage.notch-filter.filter-location
If 0 or 1, notch filter is disabled
If 3, 4th-order notch filter on control output is enabled
If 5, dual 2nd-order notch fitler on control output is enabled
If any other value, notch filter is enabled
in a different position
620
system units
4th order notch filter
stage.notch-filter.time-constant
stage.notch-filter.q
stage.notch-filter.second-filter-time-constant
stage.notch-filter.second-filter-q
621
system units
Control loop servo output

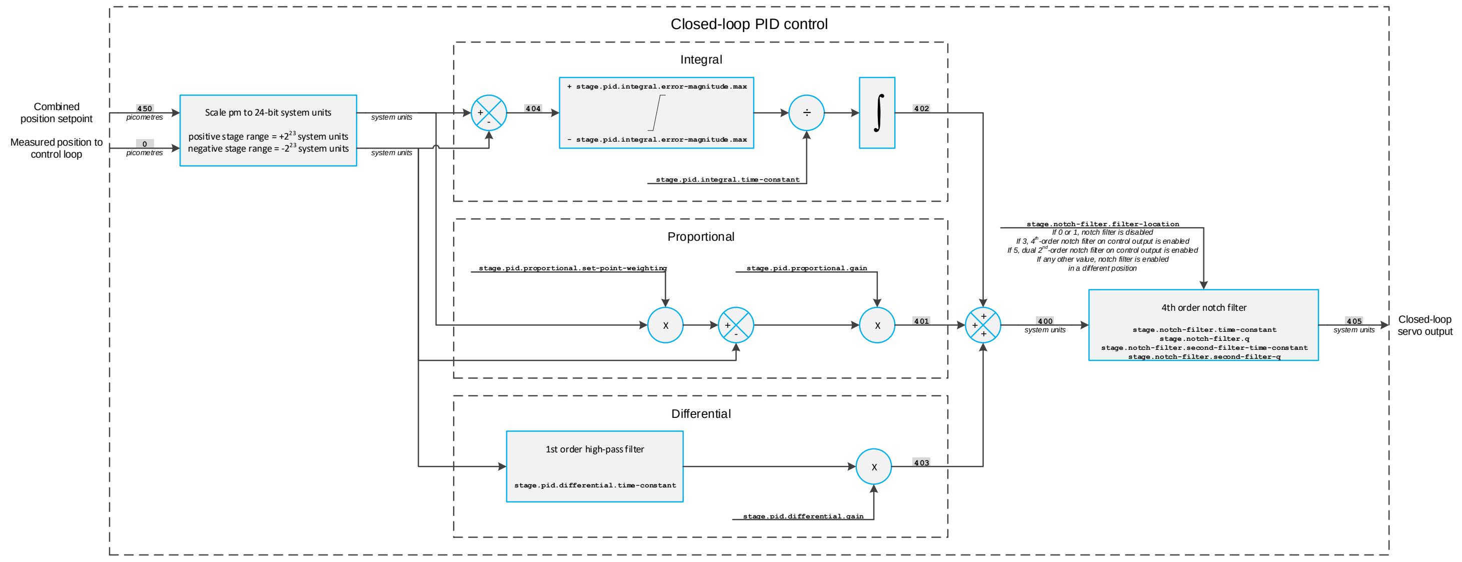
5.1.7 Feedforward command trajectory control


Feedforward command trajectory control
stage.command-trajectory.enable
Selected if enabled
Commanded position setpoint
303
picometres
Select
450
picometres
Trajectory limited position setpoint
Command trajectory limiting
stage.command-trajectory.launch-acceleration
stage.command-trajectory.speed
stage.command-trajectory.braking-deceleration
451
nm/ms
Command trajectory speed

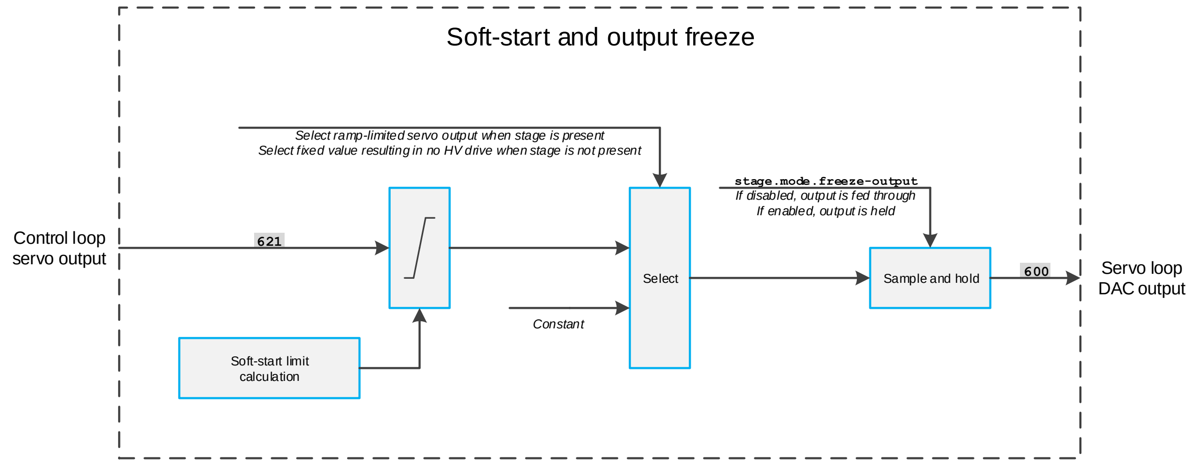
5.1.8 Open-loop control


Open-loop control
Commanded position setpoint
450
picometres
Scale pm to 24-bit system units
positive stage range = +2^23 system units
negative stage range = -2^23 system units
system units
X
+
+
410
system units
Open-loop servo output
stage.open-loop.gain
stage.open-loop.offset

5.1.9 Closed-loop PID control


Closed-loop PID control
Combined position setpoint
Measured position to control loop
450
picometres
0
picometres
Scale pm to 24-bit system units
positive stage range = +2^23 system units
negative stage range = -2^23 system units
system units
system units
Integral
404
+ stage.pid.integral.error-magnitude.max
- stage.pid.integral.error-magnitude.max
÷
∫
402
stage.pid.integral.time-constant
Proportional
stage.pid.proportional.set-point-weighting
stage.pid.proportional.gain
x
x
401
Differential
1st order high-pass filter
stage.pid.differential.time-constant
x
403
stage.pid.differential.gain
stage.notch-filter.filter-location
If 0 or 1, notch filter is disabled
If 3, 4th-order notch filter on control output is enabled
If 5, dual 2nd-order notch filter on control output is enabled
If any other value, notch filter is enabled in a different position
400
system units
4th order notch filter
stage.notch-filter.time-constant
stage.notch-filter.q
stage.notch-filter.second-filter-time-constant
stage.notch-filter.second-filter-q
405
system units
Closed-loop servo output

5.1.10 Soft-start and output freeze


Soft-start and output freeze
Select ramp-limited servo output when stage is present
Select fixed value resulting in no HV drive when stage is not present
stage.mode.freeze-output
If disabled, output is fed through
If enabled, output is held
Control loop servo output
621
Select
Constant
Sample and hold
600
Servo loop DAC output
Soft-start limit calculation

5.2 Command processing

5.2.1 stage.position.absolute-command.get/set

5.2.1.1 Description

Stage absolute commanded position.

When this is set, the value which is actually set is 5.2.1.8 **`stage.position.command`**. The value set for **`stage.position.command`** is the difference between the value set in this command and the sum of all other commands, such that **`stage.position.absolute-`** command is the desired value.

5.2.1.2 Command

```
stage.position.absolute-command.get <channel>
```

```
stage.position.absolute-command.set <channel> <value>
```

5.2.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.2.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	picometres			Stage absolute commanded position

5.2.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	picometres			Stage absolute commanded position

5.2.1.6 Possible error reports

Error return “errcode” string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.2.1.7 Minimum security level

stage.position.absolute-command.get	No security required
stage.position.absolute-command.set	User level security required

5.2.1.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

5.2.2 stage.position.command.get/set

5.2.2.1 Description

Stage commanded position (digital position)

5.2.2.2 Command

```
stage.position.command.get <channel>

stage.position.command.set <channel> <value>
```

5.2.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.2.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	picometres			Stage commanded position

5.2.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	picometres			Stage commanded position

5.2.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.2.2.7 Minimum security level

stage.position.command.get	No security required
stage.position.command.set	User level security required

5.2.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.3 High-pass filters for edge boost

5.3.1 stage.command.analogue.high-pass-filter.gain.get/set

5.3.1.1 Description

Analogue input command feed-forward high-pass filter gain

5.3.1.2 Command

```
stage.command.analogue.high-pass-filter.gain.get <channel>
```

```
stage.command.analogue.high-pass-filter.gain.set <channel> <value>
```

5.3.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.3.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		0		High-pass filter gain

5.3.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		0		High-pass filter gain

5.3.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Gain must be within specified range (set only)

5.3.1.7 Minimum security level

stage.command.analogue.high-pass-filter.gain.get	User level security required
stage.command.analogue.high-pass-filter.gain.set	Superuser level security required

5.3.2 stage.command.analogue.high-pass-filter.time-constant.get/set

5.3.2.1 Description

Analogue input command feed-forward high-pass filter time constant

5.3.2.2 Command

```
stage.command.analogue.high-pass-filter.time-constant.get <channel>
```

```
stage.command.analogue.high-pass-filter.time-constant.set <channel>
<value>
```

5.3.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.3.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	1e-6	10	Filter time constant

5.3.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	1e-6	10	Filter time constant

5.3.2.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Time constant must be within specified range (set only)

5.3.2.7 Minimum security level

stage.command.analogue.high-pass-filter.time-constant.get	User level security required
stage.command.analogue.high-pass-filter.time-constant.set	Superuser level security required

5.3.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.3.3 stage.command.analogue.high-pass-filter.q.get/set

5.3.3.1 Description

Analogue input command feed-forward high-pass filter Q factor

5.3.3.2 Command

```
stage.command.analogue.high-pass-filter.q.get <channel>
```

```
stage.command.analogue.high-pass-filter.q.set <channel> <value>
```

5.3.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.3.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		0		Filter Q factor

5.3.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		0		Filter Q factor

5.3.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Q factor must be within specified range (set only)

5.3.3.7 Minimum security level

stage.command.analogue.high-pass-filter.q.get	User level security required
stage.command.analogue.high-pass-filter.q.set	Superuser level security required

5.3.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.3.4 stage.command.digital.high-pass-filter.gain.get/set

5.3.4.1 Description

Digital command feed-forward high-pass filter gain

5.3.4.2 Command

```
stage.command.digital.high-pass-filter.gain.get <channel>

stage.command.digital.high-pass-filter.gain.set <channel> <value>
```

5.3.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.3.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		0		High-pass filter gain

5.3.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		0		High-pass filter gain

5.3.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Gain must be within specified range (set only)

5.3.4.7 Minimum security level

stage.command.digital.high-pass-filter.gain.get	User level security required
stage.command.digital.high-pass-filter.gain.set	Superuser level security required

5.3.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.3.5 stage.command.digital.high-pass-filter.time-constant.get/set

5.3.5.1 Description

Digital command feed-forward high-pass filter time constant

5.3.5.2 Command

```
stage.command.digital.high-pass-filter.time-constant.get <channel>
```

```
stage.command.digital.high-pass-filter.time-constant.set <channel>
<value>
```

5.3.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.3.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	1e-6	1	Filter time constant

5.3.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	1e-6	1	Filter time constant

5.3.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Time constant must be within specified range (set only)

5.3.5.7 Minimum security level

stage.command.digital.high-pass-filter.time-constant.get	User level security required
stage.command.digital.high-pass-filter.time-constant.set	Superuser level security required

5.3.5.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.3.6 stage.command.digital.high-pass-filter.q.get/set

5.3.6.1 Description

Digital command feed-forward high-pass filter Q factor

5.3.6.2 Command

```
stage.command.digital.high-pass-filter.q.get <channel>
```

```
stage.command.digital.high-pass-filter.q.set <channel> <value>
```

5.3.6.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.3.6.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		0		Filter Q factor

5.3.6.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		0		Filter Q factor

5.3.6.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Q factor must be within specified range (set only)

5.3.6.7 Minimum security level

stage.command.digital.high-pass-filter.q.get	User level security required
stage.command.digital.high-pass-filter.q.set	Superuser level security required

5.3.6.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.4 Measured position processing

5.4.1 stage.position.measured.get

5.4.1.1 Description

Stage measured position

5.4.1.2 Command

```
stage.position.measured.get <channel>
```

5.4.1.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.4.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	picometres			Stage measured position

5.4.1.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.4.1.6 Minimum security level

stage.position.measured.get	No security required

5.4.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.4.2 stage.position.low-pass-filter.filter-location.get/set

5.4.2.1 Description

Enabling and location of measured position low-pass filter in control loop

5.4.2.2 Command

```
stage.position.low-pass-filter.filter-location.get <channel>
```

```
stage.position.low-pass-filter.filter-location.set <channel> <value>
```

5.4.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.4.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit integer	enum	0	2	Location of filter 0=filter is disabled 1=filter is enabled for both control loop and position monitor output 2=filter is enabled only for position monitor output; filter is disabled for control loop

5.4.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit integer	enum	0	2	Location of filter 0=filter is disabled 1=filter is enabled for both control loop and position monitor output 2=filter is enabled only for position monitor output; filter is disabled for control loop

5.4.2.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Low-pass filter location value is invalid (set only)

5.4.2.7 Minimum security level

stage.position.low-pass-filter.filter-location.get	User level security required
stage.position.low-pass-filter.filter-location.set	Superuser level security required

5.4.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.4.3 stage.position.low-pass-filter.time-constant.get/set

5.4.3.1 Description

Measured position low-pass filter time constant

5.4.3.2 Command

```
stage.position.low-pass-filter.time-constant.get <channel>
```

```
stage.position.low-pass-filter.time-constant.set <channel> <value>
```

5.4.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.4.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	1e-6	1	Filter time constant

5.4.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	1e-6	1	Filter time constant

5.4.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

Error return “errcode” string reported	Description
Value out of range	Time constant must be within specified range (set only)

5.4.3.7 Minimum security level

stage.position.low-pass-filter.time-constant.get	User level security required
stage.position.low-pass-filter.time-constant.set	Superuser level security required

5.4.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.4.4 stage.position.low-pass-filter.q.get/set

5.4.4.1 Description

Measured position low-pass filter Q factor

5.4.4.2 Command

```
stage.position.low-pass-filter.q.get <channel>
```

```
stage.position.low-pass-filter.q.set <channel> <value>
```

5.4.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.4.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		0		Filter Q factor

5.4.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		0		Filter Q factor

5.4.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

Error return “errcode” string reported	**Description**
Value out of range	Q factor must be within specified range (set only)

5.4.4.7 Minimum security level

stage.position.low-pass-filter.q.get	User level security required
stage.position.low-pass-filter.q.set	Superuser level security required

5.4.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.5 In position check

5.5.1 stage.in-position.error-threshold.get/set

5.5.1.1 Description

Stage “in position” error threshold (ready limit)

5.5.1.2 Command

```
stage.in-position.error-threshold.get <channel>
```

```
stage.in-position.error-threshold.set <channel> <value>
```

5.5.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.5.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	picometres	0		Stage “in position” error threshold (ready limit)

5.5.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		0		Stage “in position” error threshold (ready limit)

5.5.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Error threshold must be within specified range (set only)

5.5.1.7 Minimum security level

stage.in-position.error-threshold.get	User level security required
stage.in-position.error-threshold.set	Superuser level security required

5.5.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.5.2 stage.in-position.lpf.time-constant.get/set

5.5.2.1 Description

Stage “in position” confirmation low-pass filter time constant

5.5.2.2 Command

```
stage.in-position.lpf.time-constant.get <channel>
```

```
stage.in-position.lpf.time-constant.set <channel> <value>
```

5.5.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.5.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	1e-6	1	Stage “in position” confirmation low-pass filter time constant

5.5.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	1e-6	1	Stage “in position” confirmation low-pass filter time constant

5.5.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Time constant must be within specified range (set only)

5.5.2.7 Minimum security level

stage.in-position.lpf.time-constant.get	User level security required
stage.in-position.lpf.time-constant.set	Superuser level security required

5.5.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.5.3 stage.in-position.window-filter.size.get/set

5.5.3.1 Description

Stage “in position” confirmation window filter size

5.5.3.2 Command

```
stage.in-position.window-filter.size.get <channel>
```

```
stage.in-position.window-filter.size.set <channel> <value>
```

5.5.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.5.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit integer	DSP loop ticks	1	1200	Stage “in position” confirmation window filter size

5.5.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit integer	DSP loop ticks	1	1200	Stage “in position” confirmation window filter size

5.5.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Window filter size must be within specified range (set only)

5.5.3.7 Minimum security level

stage.in-position.window-filter.size.get	User level security required
stage.in-position.window-filter.size.set	Superuser level security required

5.5.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.5.4 stage.in-position.window-filter.valid-threshold.get/set

5.5.4.1 Description

Stage “in position” confirmation window filter valid count threshold to set debounced output

5.5.4.2 Command

```
stage.in-position.window-filter.valid-threshold.get <channel>

stage.in-position.window-filter.valid-threshold.set <channel> <value>
```

5.5.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.5.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit integer	DSP loop ticks	1	1200	Stage “in position” confirmation window filter valid count threshold

5.5.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit integer	DSP loop ticks	1	1200	Stage “in position” confirmation window filter valid count threshold

5.5.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Window filter valid count threshold must be within specified range (set only)

5.5.4.7 Minimum security level

stage.in-position.window-filter.size.get	User level security required
stage.in-position.window-filter.size.set	Superuser level security required

5.5.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.6 Control loop

5.6.1 stage.notch-filter.filter-location.get/set

5.6.1.1 Description

Location and type of notch filter(s) in control loop

5.6.1.2 Command

```
stage.notch-filter.filter-location.get <channel>
```

```
stage.notch-filter.filter-location.set <channel> <value>
```

5.6.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.6.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit integer	Enum			Location and type of notch filter(s) in control loop 0=notch filter disabled 2=single 4th-order notch filter on PID output (applied in closed-loop only) 3=single 4th-order notch filter on control output (applied in open-loop and closed-loop) 4=dual 2nd-order notch filters on PID output (applied in closed-loop only) 5=dual 2nd-order notch filters on control output (applied in open-loop and closed-loop)

5.6.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit integer	Enum			Location and type of notch filter(s) in control loop 0=notch filter disabled 2=single 4th-order notch filter on PID output (applied in closed-loop only) 3=single 4th-order notch filter on control output (applied in open-loop and closed-loop) 4=dual 2nd-order notch filters on PID output (applied in closed-loop only) 5=dual 2nd-order notch filters on control output (applied in open-loop and closed-loop)

5.6.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Notch filter location value is invalid (set only)

5.6.1.7 Minimum security level

stage.notch-filter.filter-location.get	User level security required
stage.notch-filter.filter-location.set	Superuser level security required

5.6.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.6.2 stage.notch-filter.time-constant.get/set

5.6.2.1 Description

First (or single) notch filter time constant

5.6.2.2 Command

```
stage.notch-filter.time-constant.get <channel>

stage.notch-filter.time-constant.set <channel> <value>
```

5.6.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.6.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	1e-6	1	Notch filter time constant

5.6.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	1e-6	1	Notch filter time constant

5.6.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Time constant must be within specified range (set only)

5.6.2.7 Minimum security level

stage.notch-filter.time-constant.get	User level security required
stage.notch-filter.time-constant.set	Superuser level security required

5.6.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.6.3 stage.notch-filter.q.get/set

5.6.3.1 Description

First (or single) notch filter Q factor

5.6.3.2 Command

```
stage.notch-filter.q.get <channel>

stage.notch-filter.q.set <channel> <value>
```

5.6.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.6.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		0		Notch filter Q factor

5.6.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		0		Notch filter Q factor

5.6.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Q factor must be within specified range (set only)

5.6.3.7 Minimum security level

stage.notch-filter.q.get	User level security required
stage.notch-filter.q.set	Superuser level security required

5.6.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.6.4 stage.notch-filter.second-filter-time-constant.get/set

5.6.4.1 Description

Second notch filter time constant

5.6.4.2 Command

```
stage.notch-filter.second-filter-time-constant.get <channel>
```

```
stage.notch-filter.second-filter-time-constant.set <channel> <value>
```

5.6.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.6.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	1e-6	1	Notch filter time constant

5.6.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	1e-6	1	Notch filter time constant

5.6.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Time constant must be within specified range (set only)

5.6.4.7 Minimum security level

stage.notch-filter.second-filter-time-constant.get	User level security required
stage.notch-filter.second-filter-time-constant.set	Superuser level security required

5.6.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.6.5 stage.notch-filter.second-filter-q.get/set

5.6.5.1 Description

Second notch filter Q factor

5.6.5.2 Command

```
stage.notch-filter.second-filter-q.get <channel>

stage.notch-filter.second-filter-q.set <channel> <value>
```

5.6.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.6.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		0		Notch filter Q factor

5.6.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		0		Notch filter Q factor

5.6.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Q factor must be within specified range (set only)

5.6.5.7 Minimum security level

stage.notch-filter.second-filter-q.get	User level security required
stage.notch-filter.second-filter-q.set	Superuser level security required

5.6.5.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.7 Feedforward command trajectory control

5.7.1 stage.command-trajectory.enable.get/set

5.7.1.1 Description

Enable/disable command trajectory limiting

5.7.1.2 Command

```
stage.command-trajectory.enable.get <channel>
```

```
stage.command-trajectory.enable.set <channel> <value>
```

5.7.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.7.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Enable/disable command trajectory limiting

5.7.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Enable/disable command trajectory limiting

5.7.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller; or channel parameter is valid but no stage is connected to the specified channel
Channel not available	No stage is connected to the specified channel

5.7.1.7 Minimum security level

stage.command-trajectory.enable.get	User level security required
stage.command-trajectory.enable.set	Superuser level security required

5.7.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.7.2 stage.command-trajectory.launch-acceleration.get/set

5.7.2.1 Description

Command trajectory maximum launch acceleration limit

5.7.2.2 Command

```
stage.command-trajectory.launch-acceleration.get <channel>
```

```
stage.command-trajectory.launch-acceleration.set <channel> <value>
```

5.7.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.7.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	nm/ms/ms	0		Launch acceleration limit

5.7.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	nm/ms/ms	0		Launch acceleration limit

5.7.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Launch acceleration limit must be within specified range (set only)

5.7.2.7 Minimum security level

stage.command-trajectory.launch-acceleration.get	User level security required
stage.command-trajectory.launch-acceleration.set	Superuser level security required

5.7.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.7.3 stage.command-trajectory.speed.get/set

5.7.3.1 Description

Command trajectory maximum speed limit

5.7.3.2 Command

```
stage.command-trajectory.speed.get <channel>
```

```
stage.command-trajectory.speed.set <channel> <value>
```

5.7.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer	picometres			Controller channel for stage

5.7.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	nm/ms	0		Maximum speed limit

5.7.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	nm/ms	0		Maximum speed limit

5.7.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Maximum speed limit must be within specified range (set only)

5.7.3.7 Minimum security level

stage.command-trajectory.speed.get	User level security required
stage.command-trajectory.speed.set	Superuser level security required

5.7.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.7.4 stage.command-trajectory.braking-deceleration.get/set

5.7.4.1 Description

Command trajectory maximum braking deceleration limit

5.7.4.2 Command

```
stage.command-trajectory.braking-deceleration.get <channel>
```

```
stage.command-trajectory.braking-deceleration.set <channel> <value>
```

5.7.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.7.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	nm/ms/ms	0		Launch acceleration limit

5.7.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	nm/ms/ms	0		Launch acceleration limit

5.7.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Launch acceleration limit must be within specified range (set only)

5.7.4.7 Minimum security level

stage.command-trajectory.braking-deceleration.get	User level security required
stage.command-trajectory.braking-deceleration.set	Superuser level security required

5.7.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.8 Open-loop control

5.8.1 stage.open-loop.gain.get/set

5.8.1.1 Description

Open-loop control scaling gain

5.8.1.2 Command

```
stage.open-loop.gain.get <channel>
```

```
stage.open-loop.gain.set <channel> <value>
```

5.8.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.8.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point				Open-loop control scaling gain

5.8.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point				Open-loop control scaling gain

5.8.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.8.1.7 Minimum security level

stage.open-loop.gain.get	User level security required
stage.open-loop.gain.set	Superuser level security required

5.8.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.8.2 stage.open-loop.offset.get/set

5.8.2.1 Description

Open-loop control scaling offset

5.8.2.2 Command

```
stage.open-loop.offset.get <channel>
stage.open-loop.offset.set <channel> <value>
```

5.8.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.8.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	system units	-16777215	16777215	Open-loop control scaling offset

5.8.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	system units	-16777215	16777215	Open-loop control scaling offset

5.8.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Offset must be within specified range

5.8.2.7 Minimum security level

stage.open-loop.offset.get	User level security required
stage.open-loop.offset.set	Superuser level security required (set only)

5.8.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.8.3 stage.open-loop.output-offset-gain.get

5.8.3.1 Description

Approximate scaling from actuator desired movement in picometres to required open-loop output in bits

5.8.3.2 Command

```
stage.open-loop.output-offset-gain.get <channel>
```

5.8.3.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.8.3.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point				Output scaling

5.8.3.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.8.3.6 Minimum security level

stage.open-loop.output-offset-gain.get	No security required

5.8.3.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.9 Closed-loop PID control

5.9.1 stage.pid.proportional.gain.get/set

5.9.1.1 Description

Closed-loop control PID loop proportional gain

5.9.1.2 Command

```
stage.pid.proportional.gain.get <channel>
```

```
stage.pid.proportional.gain.set <channel> <value>
```

5.9.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.9.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		-100	100	PID loop proportional gain

5.9.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		-100	100	PID loop proportional gain

5.9.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Gain must be within specified range (set only)

5.9.1.7 Minimum security level

stage.pid.proportional.gain.get	User level security required
stage.pid.proportional.gain.set	Superuser level security required

5.9.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.9.2 stage.pid.proportional.set-point-weighting.get/set

5.9.2.1 Description

Closed-loop control PID loop proportional setpoint weighting

5.9.2.2 Command

```
stage.pid.proportional.set-point-weighting.get <channel>
```

```
stage.pid.proportional.set-point-weighting.set <channel> <value>
```

5.9.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.9.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		0	1	PID loop proportional setpoint weighting

5.9.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		0	1	PID loop proportional setpoint weighting

5.9.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Setpoint weighting must be within specified range (set only)

5.9.2.7 Minimum security level

stage.pid.proportional.set-point-weighting.get	User level security required
stage.pid.proportional.set-point-weighting.set	Superuser level security required

5.9.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.9.3 stage.pid.integral.time-constant.get/set

5.9.3.1 Description

Closed-loop control PID loop integral time constant

5.9.3.2 Command

```
stage.pid.integral.time-constant.get <channel>

stage.pid.integral.time-constant.set <channel> <value>
```

5.9.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.9.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	1e-6	1	PID loop integral time constant

5.9.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	1e-6	1	PID loop integral time constant

5.9.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Time constant must be within specified range (set only)

5.9.3.7 Minimum security level

stage.pid.integral.time-constant.get	User level security required
stage.pid.integral.time-constant.set	Superuser level security required

5.9.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.9.4 stage.pid.integral.error-magnitude.max.get/set

5.9.4.1 Description

Closed-loop control PID integral maximum error magnitude

5.9.4.2 Command

```
stage.pid.integral.error-magnitude.get <channel>
```

```
stage.pid.integral.error-magnitude.set <channel> <value>
```

5.9.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.9.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	system units	0	16777215	PID loop integral maximum error magnitude

5.9.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	system units	0	16777215	PID loop integral maximum error magnitude

5.9.4.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Limit must be within specified range (set only)

5.9.4.7 Minimum security level

stage.pid.integral.error-magnitude.max.get	User level security required
stage.pid.integral.error-magnitude.max.set	Superuser level security required

5.9.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.9.5 stage.pid.differential.gain.get/set

5.9.5.1 Description

Closed-loop control PID loop differential gain

5.9.5.2 Command

```
stage.pid.differential.gain.get <channel>

stage.pid.differential.gain.set <channel> <value>
```

5.9.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.9.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point		-10	10	PID loop differential gain

5.9.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point		-10	10	PID loop differential gain

5.9.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Gain must be within specified range (set only)

5.9.5.7 Minimum security level

stage.pid.proportional.differential.gain.get	User level security required
stage.pid.proportional.differential.gain.set	Superuser level security required

5.9.5.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.9.6 stage.pid.differential.time-constant.get/set

5.9.6.1 Description

Closed-loop control PID loop differential time constant

5.9.6.2 Command

```
stage.pid.differential.time-constant.get <channel>

stage.pid.differential.time-constant.set <channel> <value>
```

5.9.6.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.9.6.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	1e-6	1	PID loop differential time constant

5.9.6.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	1e-6	1	PID loop differential time constant

5.9.6.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Time constant must be within specified range (set only)

5.9.6.7 Minimum security level

stage.pid.proportional.differential.time-constant.get	User level security required
stage.pid.proportional.differential.time-constant.set	Superuser level security required

5.9.6.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.10 Control loop status

5.10.1 stage.status.stage-connected.get

5.10.1.1 Description

Stage is connected and ready for use

5.10.1.2 Command

```
stage.status.stage-connected.get <channel>
```

5.10.1.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.10.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Stage is connected and ready for use

5.10.1.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

5.10.1.6 Minimum security level

stage.status.stage-connected.get	No security required

5.10.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.10.2 stage.status.in-position.unconfirmed.get

5.10.2.1 Description

Stage unconfirmed “in position” state

5.10.2.2 Command

```
stage.status.in-position.unconfirmed.get <channel>
```

5.10.2.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.10.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Stage unconfirmed “in position” state

5.10.2.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

5.10.2.6 Minimum security level

stage.status.in-position.unconfirmed.get	No security required

5.10.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.10.3 stage.status.in-position.lpf-confirmed.get

5.10.3.1 Description

Stage “in position” state confirmed by LPF algorithm

5.10.3.2 Command

```
stage.status.in-position.lpf-confirmed.get <channel>
```

5.10.3.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.10.3.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Stage “in position” state confirmed by LPF algorithm

5.10.3.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

5.10.3.6 Minimum security level

stage.status.in-position.lpf-confirmed.get	No security required

5.10.3.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.10.4 stage.status.in-position.window-filter-confirmed.get

5.10.4.1 Description

Stage “in position” state confirmed by window filter algorithm

5.10.4.2 Command

```
stage.status.in-position.window-filter-confirmed.get <channel>
```

5.10.4.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.10.4.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Stage “in position” state confirmed by window filter algorithm

5.10.4.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

5.10.4.6 Minimum security level

stage.status.in-position.window-filter-confirmed.get	No security required

5.10.4.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.11 Control loop configuration

5.11.1 stage.mode.analogue-command.get/set

5.11.1.1 Description

Enable analogue input position commands

5.11.1.2 Command

```
stage.mode.analogue-command.get <channel>
```

```
stage.mode.analogue-command.set <channel> <value>
```

5.11.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.11.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Enable analogue input position commands 1 = enable 0 = disable

5.11.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Enable analogue input position commands 1 = enable 0 = disable

5.11.1.6 Possible error reports

Error return "errcode" string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.11.1.7 Minimum security level

stage.mode.analogue-command.get	No security required
stage.mode.analogue-command.set	Superuser level security required

5.11.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.11.2 stage.mode.digital-command.get/set

5.11.2.1 Description

Enable digital position commands (from host PC, snapshot step and function playback)

5.11.2.2 Command

```
stage.mode.digital-command.get <channel>
```

```
stage.mode.digital-command.set <channel> <value>
```

5.11.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.11.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Enable digital position commands 1 = enable 0 = disable

5.11.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Enable digital position commands 1 = enable 0 = disable

5.11.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.11.2.7 Minimum security level

stage.mode.digital-command.get	No security required
stage.mode.digital-command.set	Superuser level security required

5.11.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.11.3 stage.mode.closed-loop.get/set

5.11.3.1 Description

Select closed-loop or open-loop operation

5.11.3.2 Command

```
stage.mode.closed-loop.get <channel>

stage.mode.closed-loop.set <channel> <value>
```

5.11.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.11.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Select closed/open-loop operation 1 = closed-loop 0 = open-loop

5.11.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Select closed/open-loop operation 1 = closed-loop 0 = open-loop

5.11.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.11.3.7 Minimum security level

stage.mode.closed-loop.get	No security required
stage.mode.closed-loop.set	Superuser level security required

5.11.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.11.4 stage.mode.freeze-servo-output.get/set

5.11.4.1 Description

Freeze/unfreeze servo output

5.11.4.2 Command

```
stage.mode.freeze-servo-output.get <channel>
```

```
stage.mode.freeze-servo-output.set <channel> <value>
```

5.11.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.11.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Freeze servo output 1 = freeze 0 = unfreeze

5.11.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Freeze servo output 1 = freeze 0 = unfreeze

5.11.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.11.4.7 Minimum security level

stage.mode.freeze-servo-output.get	No security required
stage.mode.freeze-servo-output.set	User level security required

5.11.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

5.11.5 stage.mode.in-position-output-select.get/set

5.11.5.1 Description

Select stage "in position" digital output to be confirmed by low-pass filter or window filter algorithm

5.11.5.2 Command

```
stage.mode.in-position-output-select.get <channel>
```

```
stage.mode.in-position-output-select.set <channel> <value>
```

5.11.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

5.11.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Select confirmation algorithm 1 = window filter confirmation selected 0 = LPF confirmation selected

5.11.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Select confirmation algorithm 1 = window filter confirmation selected 0 = LPF confirmation selected

5.11.5.6 Possible error reports

Error return "errcode" string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

5.11.5.7 Minimum security level

stage.mode.in-position-output-select.get	No security required
stage.mode.in-position-output-select.get	Superuser level security required

5.11.5.8 Supported in

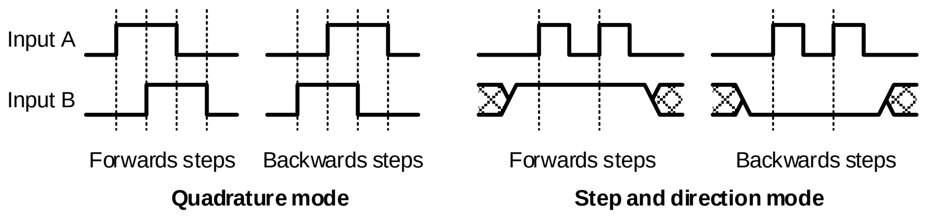
Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

6 Stepped inputs, outputs and position command

6.1 Stepped inputs

The stepped inputs may be configured as quadrature or step and direction interfaces. For each forwards or backwards step received, the commanded position moves by a configurable step in the relevant direction.

For quadrature mode, input A leads input B when commanding forwards movement on the stage; and input B leads input A when commanding backwards movement. With the standard Gray Code system as used for stepper motor controllers, each edge on each input indicates a step in the appropriate direction. For step and direction mode, input B controls whether a step is to be forwards or backwards, and a rising edge on input A triggers a step in the appropriate direction.



To ensure the controller can be easily integrated with other systems, this is also configurable to reverse direction, and for step and direction mode to trigger on rising or falling edge. Where control via stepped inputs is not required, of course it is possible to disable the stepped input.

The stepped input configuration is saved in the stage preset along with other system data (see 10). Different presets may therefore be configured with or without stepped inputs enabled, and with different step sizes, as required.

Debouncing is provided for stepped inputs if required, so that each input must be stable for the specified time before the new value is picked up. The same debounce time is applied for both inputs.

Debouncing is typically not required for quadrature mode, even if driven from a physical switch, because Gray Code ensures the stepped command will merely jitter one step forwards or backwards before settling on the final value. However it may be of value in step and direction mode, where spurious steps on input A will cause significant errors in the system. It may also be required for any system where significant noise on signal cables is expected. Debouncing will affect the rate at which the stepped command can be updated, so it should be used with caution.

Stepped inputs must not be changed faster than they can be read by the controller, otherwise steps may be missed. The following timings must be observed when debouncing is disabled. When debouncing is enabled, the limit will naturally be set by the debounce time instead.

Interface mode	**Timing parameter**	**Absolute minimum**	**Recommended limit**	**Absolute maximum**
Quadrature	Edge on input to edge on opposite input (“hold time”)	20ns	40ns	
	Step rate for both inputs combined		25MHz	50MHz
Step and direction	Change of state on input B to trigger edge on input A (“settle time”)	20ns	40ns	
	Trigger edge on input A to change of state on input B (“hold time”)	20ns	40ns	
	Edge on input A to edge on input A (“hold time”)	20ns	40ns	
	Step rate		12.5MHz	25MHz

Note that these limits relate to the controller electronics only. The cable type and length will cause some degradation of signals, as will other factors such as ferrite beads for EM emissions reduction. Achievable transmission rates in a customer’s application will therefore typically be lower, and customers must carry out appropriate design investigations to ensure signals reach the controller reliably.

6.1.1 stage.stepped.input.enable.get/set

6.1.1.1 Description

Stepped input enable

6.1.1.2 Command

```
stage.stepped.input.enable.get <channel>
```

```
stage.stepped.input.enable.set <channel> <value>
```

6.1.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.1.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Enable/disable stepped input

6.1.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Enable/disable stepped input

6.1.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.1.1.7 Minimum security level

stage.stepped.input.enable.get	No security required
stage.stepped.input.enable.set	Superuser level security required

6.1.1.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.1.2 stage.stepped.input.is-quadrature.get/set

6.1.2.1 Description

Stepped input interface mode

6.1.2.2 Command

```
stage.stepped.input.is-quadrature.get <channel>

stage.stepped.input.is-quadrature.set <channel> <value>
```

6.1.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.1.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Stepped input interface mode 1 = quadrature 0 = step and direction

6.1.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Stepped input interface mode 1 = quadrature 0 = step and direction

6.1.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.1.2.7 Minimum security level

stage.stepped.input.is-quadrature.get	No security required
stage.stepped.input.is-quadrature.set	Superuser level security required

6.1.2.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.1.3 stage.stepped.input.reverse-direction.get/set

6.1.3.1 Description

Stepped input reverse direction

6.1.3.2 Command

```
stage.stepped.input.reverse-direction.get <channel>
```

```
stage.stepped.input.reverse-direction.set <channel> <value>
```

6.1.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.1.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Stepped input reverse direction 1 = reversed (opposite directions from diagram in 6.1) 0 = normal

6.1.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Stepped input reverse direction 1 = reversed (opposite directions from diagram in 6.1) 0 = normal

6.1.3.6 Possible error reports

Error return "errcode" string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.1.3.7 Minimum security level

stage.stepped.input.reverse-direction.get	No security required
stage.stepped.input.reverse-direction.set	Superuser level security required

6.1.3.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.1.4 stage.stepped.input.debounce-time.get/set

6.1.4.1 Description

Stepped input debounce time

6.1.4.2 Command

```
stage.stepped.input.debounce-time.get <channel>
```

```
stage.stepped.input.debounce-time.set <channel> <value>
```

6.1.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.1.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	0	10	Stepped input debounce time

6.1.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	0	10	Stepped input debounce time

6.1.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.1.4.7 Minimum security level

stage.stepped.input.debounce-time.get	No security required
stage.stepped.input.debounce-time.set	Superuser level security required

6.1.4.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.1.5 stage.stepped.input.step-direction.is-rising-edge.get/set

6.1.5.1 Description

Stepped input step and direction interface mode triggers a step on rising or falling edge

6.1.5.2 Command

```
stage.stepped.input.step-direction.is-rising-edge.get <channel>
```

```
stage.stepped.input.step-direction.is-rising-edge.set <channel>
<value>
```

6.1.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.1.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Stepped input step and direction trigger 1 = triggered on rising edge of input A 0 = triggered on falling edge of input A

6.1.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolea n	0	1	Stepped input step and direction trigger 1 = triggered on rising edge of input A 0 = triggered on falling edge of input A

6.1.5.6 Possible error reports

Error return "errcode" string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.1.5.7 Minimum security level

stage.stepped.input.step-direction.is-rising-edge.get	No security required
stage.stepped.input.step-direction.is-rising-edge.set	Superuser level security required

6.1.5.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.2 Stepped position command

If the stepped input is enabled, every input step forwards or backwards will increase or decrease the stepped position command **`stage.position.stepped-command`** by one step. The stepped command may also be set directly by the user from the host PC, either as a position or in steps.

If a stepped position command is present and the step size is changed (either directly by the host PC or indirectly by the host PC selecting a new preset), then the stepped position command in steps will be recalculated such the command in picometres is unchanged, within limits of resolution.

The number of steps is a signed 32-bit value and therefore can count up to $\pm 2\text{x}10^9$ steps. When choosing the step size, the stage resolution and the number of steps available must be considered. More usually though, the step size is limited by the speed of the stage (or the desired speed of response) and the maximum step rate, which is 25MHz for quadrature inputs (see 6.1).

For example, if the stage is to move end-to-end in 0.02s, then the step size must be at least 1/500,000th of the stage range. If the stage range is 100µm, this sets the minimum step size resolution at 200pm.

Because the stepped command may be set by the PC as well, it is also possible to implement a “coarse-fine” scheme where coarse changes are sent from the host PC over the comms link more slowly, and fine changes are sent from the stepped inputs more quickly. This may be of interest in systems with an external control loop, where the “central” position is often changed infrequently but fine correction is required to hold station accurately.

For example, suppose the step size is set at 1pm. Within 1ms, the stepped inputs can command movement of up to ±25nm with very high precision, which may be desirable for accurate control. The host PC can still command coarse movement of up to ±2mm range with the same step size resolution.

Since the stepped position command **`stage.position.stepped-command`** and digital command **`stage.position.command`** are separate entities, the same “coarse-fine” concept may also be used with the stepped position command solely coming from the stepped inputs. The digital command may then be adjusted for coarse positioning. Note also that the absolute position command **`stage.position.absolute-command`** may be used to adjust the digital position to a new “central” position without needing to allow for any current stepped position command.

6.2.1 stage.position.stepped-command.get/set

6.2.1.1 Description

Stage stepped position command

6.2.1.2 Command

```
stage.position.stepped-command.get <channel>
```

```
stage.position.stepped-command.set <channel> <value>
```

6.2.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.2.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	pm			Stage stepped position command

6.2.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm			Stage stepped position command

6.2.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

6.2.1.7 Minimum security level

stage.position.stepped-command.get	No security required
stage.position.stepped-command.set	User level security required

6.2.1.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.2.2 stage.position.stepped-command.increment

6.2.2.1 Description

Stage stepped position command

6.2.2.2 Command

```
stage.position.stepped-command.increment <channel> <value>
```

6.2.2.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	pm			Increment value to add to stage stepped position command

6.2.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm			Increment value to add to stage stepped position command

6.2.2.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

6.2.2.6 Minimum security level

stage.position.stepped-command.increment	User level security required

6.2.2.7 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.2.3 stage.position.stepped-command.steps.get/set

6.2.3.1 Description

Stage stepped position command in steps

6.2.3.2 Command

```
stage.position.stepped-command.steps.get <channel>
```

```
stage.position.stepped-command.steps.set <channel> <value>
```

6.2.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.2.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit signed integer	steps			Stage stepped position command in steps

6.2.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit signed integer	steps			Stage stepped position command in steps

6.2.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

6.2.3.7 Minimum security level

stage.position.stepped-command.steps.get	No security required
stage.position.stepped-command.steps.set	User level security required

6.2.3.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.2.4 stage.position.stepped-command.steps.increment

6.2.4.1 Description

Stage stepped position command in steps

6.2.4.2 Command

```
stage.position.stepped-command.steps.increment <channel> <value>
```

6.2.4.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit signed integer	steps			Increment value to add to stage stepped position command in steps

6.2.4.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit signed integer	steps			Increment value to add to stage stepped position command in steps

6.2.4.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

6.2.4.6 Minimum security level

stage.position.stepped-command.steps.increment	User level security required

6.2.4.7 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.2.5 stage.position.stepped-command.step-size.get/set

6.2.5.1 Description

Stage stepped position command step size

6.2.5.2 Command

```
stage.position.stepped-command.step-size.get <channel>
```

```
stage.position.stepped-command.step-size.set <channel> <value>
```

6.2.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.2.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	pm	0		Stage stepped position command step size

6.2.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm	0		Stage stepped position command step size

6.2.5.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.2.5.7 Minimum security level

stage.position.stepped-command.step-size.get	No security required
stage.position.stepped-command.step-size.set	Superuser level security required

6.2.5.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.3 Stepped outputs

As with stepped inputs, the stepped outputs can also be configured as quadrature or step and direction interfaces. Like the stepped inputs, they have the option to reverse direction, and to select for the step and direction interface whether steps are triggered on rising or falling edges.

A channel's stepped output can be derived from various sources.

- a simple echo of steps from the stepped input;
- reporting the stepped position command **`stage.position.stepped-command`** (see 6.2.1) or **`stage.position.stepped-command.steps`** (see 6.2.3);
- reporting the digital position command **`stage.position.command`** (see 5.2.2);
- reporting the absolute position command **`stage.position.absolute-command`** (see 5.2.1);
- reporting the measured position **`stage.position.measured`** (see 5.4.1).

The stepped output can also be disabled if not required.

The hold time for quadrature interface signals and the settle and hold times for step and direction interface signals are configurable. This allows step rates to be limited to what the receiving hardware or cables are able to handle. The following limits specify the maximum possible transmission speed for stepped outputs.

Interface mode	**Timing parameter**	**Minimum**	**Maximum**
Quadrature	Edge on output to edge on opposite output ("hold time")	20ns	
	Step rate for both outputs combined		50MHz
Step and direction	Change of state on output B to trigger edge on output A ("settle time")	40ns	
	Trigger edge on output A to change of state on output B ("hold time")	40ns	
	Edge on output A to edge on output A ("hold time")	40ns	
	Step rate		12.5MHz

Note that these limits relate to the controller electronics only. The cable type and length will cause some degradation of signals, as will other factors such as ferrite beads for EM emissions reduction. Achievable transmission rates in a customer's application will therefore typically be lower, and customers must carry out appropriate design investigations to ensure signals reach the customer's electronics reliably.

Regardless of the source of the stepped output signal, the output settle and hold times specified by the user will always be followed. In particular, the output settle and hold times will still be followed even if the stepped output is set to simply repeat steps received on the stepped input. It may therefore be possible for the input to change faster than the output can keep up. The output will still send the correct number of steps, but it will lag behind the input. The user must be aware of this if using the stepped output for “handshaking” to verify that the controller has correctly received the stepped input command.

Where the stepped output reports a position, the step size can be set independently of the step size for the stepped input. If the step size is set too small and/or the step rate is set too slow, it is possible that position values reported on the stepped output may change slower than the slew rate of the stage. The user must ensure that the step size and step rate are configured appropriately for the system in which they will be used.

The nature of stepped I/O is that it transmits incremental changes to a value and not the actual value. For example, the initial stage position may be 5 µm and the step size configured to 10nm (0.01 µm) but the controller will not initially send any steps on the stepped output. When the stage position moves to 5.01um, the controller will only then send 1 “forwards” step. Any receiving electronics must therefore use other means to get this initial position, or must not care about the absolute position and only consider relative changes.

This may be inconvenient on startup if the system receiving the stepped output needs to know the actual value. The stepped output can therefore be configured to send steps corresponding to the full position value when the stage is connected (or if settings are changed). In the previous example, the controller would send 500 “forwards” steps on the stepped output when the controller starts. It would then send 1 “forwards” step when the stage position moves to 5.01um. Any receiving electronics must have previously reset its position measurement value, of course, otherwise these 500 steps will be incorrectly added to the previous value. Typically this would be handled by the host PC with some sequencing logic.

6.3.1 stage.stepped.output.select.get/set

6.3.1.1 Description

Stepped output source select

6.3.1.2 Command

```
stage.stepped.output.select.get <channel>
```

```
stage.stepped.output.select.set <channel> <value>
```

6.3.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.3.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Enum			Stepped output source select 0 = stepped output disabled 1 = measured position, scaled by **`stage.stepped.output.step-size`** 2 = echo steps on stepped input 3 = stepped position command in steps 4 = stepped position command, scaled by **`stage.stepped.output.step-size`** 5 = digital position command, scaled by **`stage.stepped.output.step-size`** 6 = absolute position command, scaled by

Name	Type	Units	Minimum	Maximum	Description
					`stage.stepped.output.step-size`

6.3.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Enum			Stepped output source select 0 = stepped output disabled 1 = measured position, scaled by **`stage.stepped.output.step-size`** 2 = echo steps on stepped input 3 = stepped position command in steps 4 = stepped position command, scaled by **`stage.stepped.output.step-size`** 5 = digital position command, scaled by **`stage.stepped.output.step-size`** 6 = absolute position command, scaled by **`stage.stepped.output.step-size`**

6.3.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.3.1.7 Minimum security level

stage.stepped.output.select.get	No security required
stage.stepped.output.select.set	Superuser level security required

6.3.1.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.3.2 stage.stepped.output.is-quadrature.get/set

6.3.2.1 Description

Stepped output interface mode

6.3.2.2 Command

```
stage.stepped.output.is-quadrature.get <channel>

stage.stepped.output.is-quadrature.set <channel> <value>
```

6.3.2.3 Parameters (get)

Name	**Type**	**Units**	**Minimum**	**Maximum**	**Description**
channel	8-bit unsigned integer				Controller channel for stage

6.3.2.4 Parameters (set)

Name	**Type**	**Units**	**Minimum**	**Maximum**	**Description**
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Stepped output interface mode 1 = quadrature 0 = step and direction

6.3.2.5 Results

Name	**Type**	**Units**	**Minimum**	**Maximum**	**Description**
value	32-bit unsigned integer	Boolean	0	1	Stepped output interface mode 1 = quadrature 0 = step and direction

6.3.2.6 Possible error reports

Error return “errcode” string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.3.2.7 Minimum security level

stage.stepped.output.is-quadrature.get	No security required
stage.stepped.output.is-quadrature.set	Superuser level security required

6.3.2.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.3.3 stage.stepped.output.reverse-direction.get/set

6.3.3.1 Description

Stepped output reverse direction

6.3.3.2 Command

```
stage.stepped.output.reverse-direction.get <channel>
```

```
stage.stepped.output.reverse-direction.set <channel> <value>
```

6.3.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.3.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Stepped output reverse direction 1 = reversed (opposite directions from diagram in 6.1) 0 = normal

6.3.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Stepped output reverse direction 1 = reversed (opposite directions from diagram in 6.1) 0 = normal

6.3.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.3.3.7 Minimum security level

stage.stepped.output.reverse-direction.get	No security required
stage.stepped.output.reverse-direction.set	Superuser level security required

6.3.3.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.3.4 stage.stepped.output.step-direction.is-rising-edge.get/set

6.3.4.1 Description

Stepped output step and direction interface mode triggers a step on rising or falling edge

6.3.4.2 Command

```
stage.stepped.output.step-direction.is-rising-edge.get <channel>
```

```
stage.stepped.output.step-direction.is-rising-edge.set <channel>
<value>
```

6.3.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.3.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Stepped output step and direction trigger 1 = triggered on rising edge of output A 0 = triggered on falling edge of output A

6.3.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Stepped output step and direction trigger 1 = triggered on rising edge of output A 0 = triggered on falling edge of output A

6.3.4.6 Possible error reports

Error return "errcode" string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.3.4.7 Minimum security level

stage.stepped.output.step-direction.is-rising-edge.get	No security required
stage.stepped.output.step-direction.is-rising-edge.set	Superuser level security required

6.3.4.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.3.5 stage.stepped.output.step-size.get/set

6.3.5.1 Description

Stepped output position step size

6.3.5.2 Command

```
stage.stepped.output.step-size.get <channel>
```

```
stage.stepped.output.step-size.set <channel> <value>
```

6.3.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.3.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	pm	0		Stepped output position step size

6.3.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm	0		Stepped output position step size

6.3.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.3.5.7 Minimum security level

stage.stepped.output.step-size.get	No security required
stage.stepped.output.step-size.set	Superuser level security required

6.3.5.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.3.6 stage.stepped.output.send-full-value.get/set

6.3.6.1 Description

Stepped output send full value or incremental changes

6.3.6.2 Command

```
stage.stepped.output.send-full-value.get <channel>
```

```
stage.stepped.output.send-full-value.set <channel> <value>
```

6.3.6.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.3.6.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Boolean	0	1	Stepped output send full value or incremental changes 1 = On stage connection or power-on, send steps corresponding to current value of parameter selected for stepped output. Send incremental changes thereafter. 0 = On stage connection or power-on, do nothing. Send incremental changes from initial position thereafter.

6.3.6.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Stepped output send full value or incremental changes

Name	Type	Units	Minimum	Maximum	Description
					1 = On stage connection or power-on, send steps corresponding to current value of parameter selected for stepped output. Send incremental changes thereafter. 0 = On stage connection or power-on, do nothing. Send incremental changes from initial position thereafter.

6.3.6.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.3.6.7 Minimum security level

stage.stepped.output.send-full-value.get	No security required
stage.stepped.output.send-full-value.set	Superuser level security required

6.3.6.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.3.7 stage.stepped.output.quadrature.hold-time.get/set

6.3.7.1 Description

Stepped output quadrature hold time

6.3.7.2 Command

```
stage.stepped.output.quadrature.hold-time.get <channel>
```

```
stage.stepped.output.quadrature.hold-time.set <channel> <value>
```

6.3.7.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.3.7.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	0	10	Stepped output quadrature hold time

6.3.7.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	0	10	Stepped output quadrature hold time

6.3.7.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.3.7.7 Minimum security level

stage.stepped.output.quadrature.hold-time.get	No security required
stage.stepped.output.quadrature.hold-time.set	Superuser level security required

6.3.7.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.3.8 stage.stepped.output.step-direction.settle-time.get/set

6.3.8.1 Description

Stepped output step and direction settle time

6.3.8.2 Command

```
stage.stepped.output.step-direction.settle-time.get <channel>
```

```
stage.stepped.output.step-direction.settle-time.set <channel> <value>
```

6.3.8.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.3.8.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	0	10	Stepped output step and direction settle time

6.3.8.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	0	10	Stepped output step and direction settle time

6.3.8.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.3.8.7 Minimum security level

stage.stepped.output.step-direction.settle-time.get	No security required
stage.stepped.output.step-direction.settle-time.set	Superuser level security required

6.3.8.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

6.3.9 stage.stepped.output.step-direction.hold-time.get/set

6.3.9.1 Description

Stepped output step and direction hold time

6.3.9.2 Command

```
stage.stepped.output.step-direction.hold-time.get <channel>
```

```
stage.stepped.output.step-direction.hold-time.set <channel> <value>
```

6.3.9.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

6.3.9.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit floating-point	s	0	10	Stepped output step and direction hold time

6.3.9.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	0	10	Stepped output step and direction hold time

6.3.9.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data value must be within specified range (set only)

6.3.9.7 Minimum security level

stage.stepped.output.step-direction.hold-time.get	No security required
stage.stepped.output.step-direction.hold-time.set	Superuser level security required

6.3.9.8 Supported in

Controller application firmware	6.2.12 onwards
Controller interface library	2.2.7 onwards

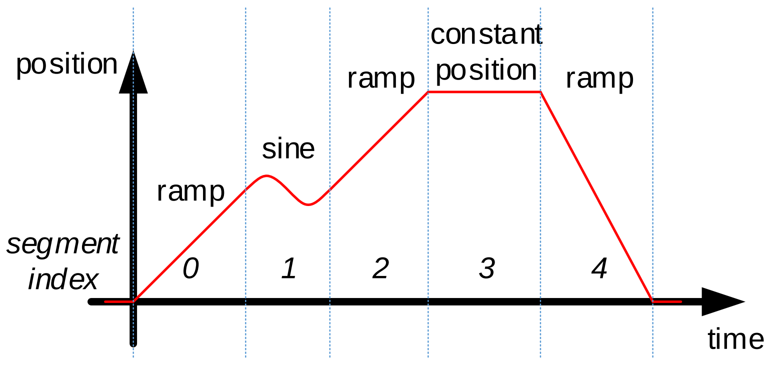
7 Function playback

7.1 Overview

Function playback allows pre-programmed position command waveforms to be played back on one or more channels. This allows the user to construct complex steps or ramps, sine waves, raster scans, or any other waveform as required. For less sophisticated controllers, this would typically require the PC to be fitted with a high-resolution DAC interface to drive the controller's analogue input, and software to be written which would send the appropriate commands to the DAC. With function playback this can be achieved much more easily, with much greater accuracy, and without requiring expensive additional hardware or complex software.

Each channel controlling a stage has its own function playback and waveform setup. Function playback may be run independently for each channel, where stages control elements of a system which are not connected. Alternatively function playback can be run simultaneously for two or three channels on multi-axis systems, allowing complex 2D or 3D paths to be programmed.

The function playback waveform generator allows the construction of complex waveforms with relative ease. Waveforms are constructed from "segments", where each segment of a waveform carries out some action. There are numerous different segment types, covering ramping, stepping to a position, various acceleration and deceleration curves, sine waves, and so on. For example, a waveform which ramps to a position, runs one cycle of a sine wave, ramps to a second position, holds position there for a fixed time and then ramps back to the start might be formed of 5 waveform segments carrying out the sequence "ramp up, sine wave, ramp up, hold a constant position, and ramp down" with appropriate configurations.



7.2 specifies the commands used to configure waveform segments. 7.3 describes the segments types available and the parameters to configure each type. To clarify the waveform design process, 7.4 provides an example of how this can be used in a realistic application.

Part or all of the waveform can also be set up to run repeatedly, as described in 7.5.

In the user's system, external electronics may be required to be triggered during function playback, for example where a camera must be triggered at successive points whilst moving focus through a specimen. Each stage channel has a dedicated digital trigger output which may be used for this. 7.6 describes the trigger output configuration.

Once segments have been entered, the waveform generator must check that segment configurations are valid, and construct the waveform as a series of sampled data points which can be played by the controller. 7.7 describes how to command the waveform generator to carry this out.

When the data points for a waveform have been generated, the user may read back the waveform data points if required, and may edit portions of the generated waveform. Alternatively if the user needs a waveform which cannot be provided by the waveform generator, the user may directly enter their waveform as data points and bypass the waveform generator entirely. 7.8 describes how to configure the waveform data points directly.

When the waveform is ready for use, function playback can be started directly by the PC, or can be started by external electronics using digital trigger inputs. 7.9 and 7.10 describe how to start and stop function playback from the PC and from digital trigger inputs respectively. Function playback will normally run to completion and stop on its own when completed, but it may also be paused or stopped early by the user if required. For obvious reasons, function playback waveforms cannot be modified whilst playback is running for a channel, or whilst digital input triggers have been enabled to allow function playback to be started by external equipment.

Similarly to scope measurement and snapshot (see 8 and 9), the controller provides an additional “internal” function playback waveform which is accessed using channel number 0. This is not associated with a stage channel, and on its own will have no effect. However when routed to a scope output (see 9), it allows the user to turn an unused analogue output into a programmable signal generator, configured in the same way as any other function playback waveform. Since function playback can be run simultaneously for multiple channels, this waveform can be synchronised with stage movement waveforms, providing an integrated solution for controlling external electronics. Alternatively it could be simply used as a means of providing a separate signal to external electronics, saving cost and complexity in the system by removing the need for an additional PC-controlled DAC.

Note that whilst function playback may be used to configure any waveform within the limits available to the controller, the user must ensure that the stage is physically able to follow this position command waveform, as is the case for any other position command input. If the user specifies a ramp or step which is faster than the maximum speed possible for the stage with the current control loop settings, for example, the stage will only move as fast as it is able to go and will not track the function playback ramp correctly.

7.2 Waveform generator segment configuration commands

Each segment type has a number of configuration parameters relating to that segment type. Some will be common to all (for example setting the starting position), and some will be specific to the segment type (for example setting the frequency for a sine wave). For each segment, the user also has the option of whether to continue from the previous segment's final commanded position (and commanded velocity for some segment types), making it easy to construct smooth waveforms, or whether to step to a new position. 7.3 specifies the segment types available, and the parameters available for each. A waveform may have up to 1000 segments.

As can be seen in 5.1.2, the function playback command is summed with all other command sources. This allows other complex behaviour to be set up which combines pre-programmed waveforms with other position commands. For example, a fine auto-focus could be implemented by programming a triangular waveform which automatically sweeps the stage over a ±10nm offset from the current commanded position until the correct focus point is found.

Since the function playback command is summed with all other command sources, it is set to zero when function playback is not running. The first segment should therefore always start at zero, or may step from zero to another position if a smooth start is not required. A controlled end to the waveform is often required, in which case the last segment should also return to zero via a ramp or other smooth transition. If this is unimportant, the waveform may finish at any value and the controller will step back to the original position as the function playback command steps back to zero.

If the waveform is required to finish at a different position and hold that position after function playback has completed, it is possible to set a non-zero final position and set the waveform to "soft" stop at the end. In this case the digital position command is set to the combined position command at the end of the waveform (see the diagram in 5.1.2), so that the absolute commanded position holds its value when the function stops. See also 7.9 for details of commanding a "hard" or "soft" stop whilst the function is running, which behaves identically.

The last segment may also finish at a different position if the waveform is set to repeat forever (see 7.5), in which case any position after it stops does not need to be considered.

7.2.1 function.waveform-generator.clear

7.2.1.1 Description

Clear waveform generator settings

7.2.1.2 Command

```
function.waveform-generator.clear <channel>
```

7.2.1.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.2.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
status	32-bit unsigned integer				Returns 1 if clear succeeded

7.2.1.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed

7.2.1.6 Minimum security level

function.waveform.clear	User level security required

7.2.1.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.2.2 function.waveform-generator.segment.type.get/set

7.2.2.1 Description

Function type for waveform generator segment

7.2.2.2 Command

```
function.waveform-generator.segment.type.get <channel> <segment>
```

```
function.waveform-generator.segment.type.set <channel> <segment>
<type>
```

7.2.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.2.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index
type	String	Enum			Function type for segment See 7.3 for available values

7.2.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
type	String	Enum			Function type for segment See 7.3 for available values

7.2.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range

Error return “errcode” string reported	**Description**
Value out of range	Segment type is incorrect (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.2.2.7 Minimum security level

function.waveform-generator.segment.type.get	No security required
function.waveform-generator.segment.type.set	User level security required

7.2.2.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.2.3 function.waveform-generator.segment.parameter.get/set

7.2.3.1 Description

Function parameter for waveform generator segment

7.2.3.2 Command

```
function.waveform-generator.segment.parameter.get <channel> <segment>
```

```
function.waveform-generator.segment.parameter.set <channel> <segment>
<value>
```

7.2.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index
parameter	32-bit unsigned integer		0	7	Parameter index See 7.3 for parameters for each function type, and the index for each parameter

7.2.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index
parameter	32-bit unsigned integer		0	7	Parameter index See 7.3 for parameters for each function type, and the index for each parameter
value	32-bit floating-point				Function parameter for segment See 7.3 for details of each segment type’s parameters

7.2.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point				Function parameter for segment See 7.3 for details of each segment type's parameters

7.2.3.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index or parameter index must be within specified range
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.2.3.7 Minimum security level

function.waveform-generator.segment.parameter.get	No security required
function.waveform-generator.segment.parameter.set	User level security required

7.2.3.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.2.4 function.waveform-generator.segment.continue-position-velocity.get/set

7.2.4.1 Description

Continue previous position and velocity for waveform generator segment.

7.2.4.2 Command

```
function.waveform-generator.segment.continue-position-velocity.get
<channel> <segment>
```

```
function.waveform-generator.segment.continue-position-velocity.set
<channel> <segment> <continue-position> <continue-velocity>
```

7.2.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.2.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
segment	32-bit unsigned integer		0	999	Segment index
continue-position	32-bit unsigned integer		0	1	If set to 1, start position for this segment is taken from the end position for the previous segment, and start position function parameter for function type does not need to be provided. If set to 0, start position for this segment is set by the relevant function parameter for this segment. See 7.3 for details of function types and function parameters.
continue-velocity	32-bit unsigned integer		0	1	If set to 1, **and** segment has a function parameter setting start velocity, then start

Name	Type	Units	Minimum	Maximum	Description
					velocity for this segment is taken from the end velocity for the previous segment. This may only be set if continue-position is also set to 1. If set to 0, start velocity for this segment is set explicitly by the relevant function parameter for this segment, if the segment type uses it. If function type does not have a function parameter to set start position, then this is unused. See 7.3 for details of function types and function parameters.

7.2.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
continue-position	32-bit unsigned integer		0	1	If set to 1, start position for this segment is taken from the end position for the previous segment, and start position function parameter for function type does not need to be provided. If set to 0, start position for this segment is set by the relevant function parameter for this segment. See 7.3 for details of function types and function parameters.
continue-velocity	32-bit unsigned integer		0	1	If set to 1, **and** segment has a function parameter setting start velocity, then start velocity for this segment is

Name	Type	Units	Minimum	Maximum	Description
					taken from the end velocity for the previous segment. This may only be set if continue-position is also set to 1. If set to 0, start velocity for this segment is set explicitly by the relevant function parameter for this segment, if the segment type uses it. If function type does not have a function parameter to set start position, then this is unused. See 7.3 for details of function types and function parameters.

7.2.4.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range
Value out of range	Continue-velocity must not be set to 1 if continue-position is not set to 1 (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.2.4.7 Minimum security level

function.waveform-generator.segment.continue-position-velocity.get	No security required
function.waveform-generator.segment.continue-position-velocity.set	User level security required

7.2.4.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.2.5 function.waveform-generator.count.get/set

7.2.5.1 Description

Number of waveform generator segments

7.2.5.2 Command

```
function.waveform-generator.count.get <channel>
```

```
function.waveform-generator.count.set <channel> <value>
```

7.2.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.2.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	32-bit unsigned integer		1	1000	Number of waveform generator segments

7.2.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer		1	1000	Number of waveform generator segments

7.2.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Number of segments must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.2.5.7 Minimum security level

function.waveform-generator.count.get	No security required
function.waveform-generator.count.set	User level security required

7.2.5.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.2.6 function.waveform-generator.soft-stop-at-end.get/set

7.2.6.1 Description

Waveform soft-stops at end (true) or steps to zero (false)

7.2.6.2 Command

```
function.waveform-generator.soft-stop-at-end.get <channel>
```

```
function.waveform-generator.soft-stop-at-end.set <channel> <value>
```

7.2.6.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.2.6.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	32-bit unsigned integer		0	1	If set to 1, waveform soft-stops at end. If set to 0, waveform stops normally at end.

7.2.6.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer		0	1	If set to 1, waveform soft-stops at end. If set to 0, waveform stops normally at end.

7.2.6.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

Error return "errcode" string reported	Description
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.2.6.7 Minimum security level

function.waveform-generator.soft-stop-at-end.get	No security required
function.waveform-generator.soft-stop-at-end.set	User level security required

7.2.6.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.3 Waveform generator segment function types

The waveform generator has a variety of segment function types, set by `function.waveform-generator.segment.type` (see 7.2.2). Every function type is configured with different function parameters depending on the waveform to be generated, set by `function.waveform-generator.segment.parameter` (see 7.2.3).

The majority of function types have three variants of the same function.

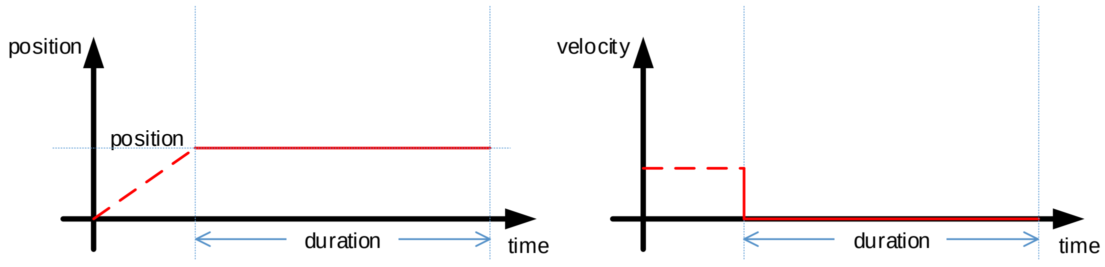
- The "duration" type carries out the function (for example, ramping or following an acceleration trajectory) over a fixed time period. It starts at the start position and continues for the specified time. The distance travelled depends on the time taken and other parameters for the segment.
- The "position" type carries out the function from the start position to an end position. The time taken depends on the distance and other parameters for the segment.
- The "distance" type carries out the function from the start position to some distance relative to the start position. This is important when the start position is not known exactly or cannot be easily calculated, such as when following a complex waveform. As for the "position" type, the time taken depends on the distance and other parameters for the segment.

All three variants will not always be applicable for every function, but it is common to see two or three variants of the same function. Other variants may also exist, where other parameters may be used to control the function.

As described in 7.7, durations are rounded to fit the sample period. This has implications for "position" and "distance" function types if the velocity or acceleration during the segment must be set accurately, because the segment prioritises hitting the correct end position and rounds the velocity and acceleration profiles to ensure this occurs. The "duration" type maintains the specified velocity or acceleration for the segment duration, meaning that it may travel a slightly longer or shorter distance depending on how the duration is rounded. This may affect which function type is most suitable for the user's application. Where parameter rounding occurs, this is specified for each function type.

7.3.1 Constant position

Segment function type	Description
constant-position	Hold a fixed command position for the specified time.



This provides basic constant position behaviour, holding position for a set time.

A square wave can be constructed from two "constant-position" segments at the relevant positions.

7.3.1.1 constant-position

7.3.1.1.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Position	picometres			Position to hold. If "continue-position" is set, the end position for the previous segment is held, and this parameter is unused.
1	Duration	s	> 0		Time to hold fixed position

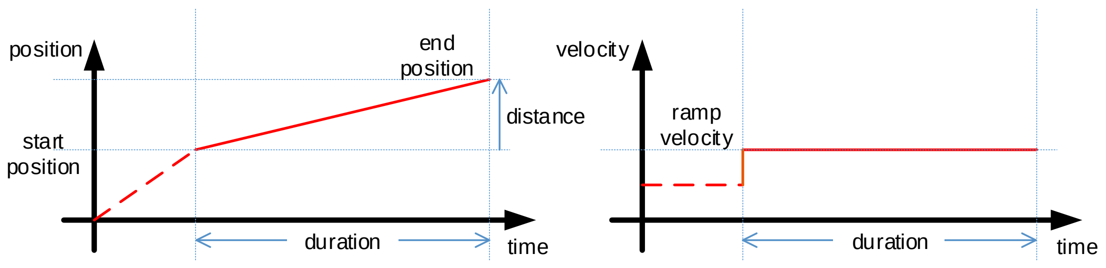
When preparing the waveform, duration is rounded to the nearest sample period (see 7.7).

7.3.1.1.2 Possible errors

Duration must be greater than zero.

7.3.2 Constant velocity

Segment function type	Description
constant-velocity-duration	Ramp command position at a constant velocity for the specified time.
constant-velocity-position	Ramp command position at a constant velocity from start position to end position.
constant-velocity-distance	Ramp command position at a constant velocity from start position for a relative distance.



This provides a simple ramp. The user has the choice of whether to ramp over a fixed time or a fixed distance.

A sawtooth wave can be constructed from successive “constant-velocity” segments, stepping back to the initial position (i.e. “continue-position” is not set).

A triangle wave can be constructed from successive “constant-velocity” segments, ramping up and then ramping down.

7.3.2.1 constant-velocity-duration

7.3.2.1.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	Ramp velocity	picometres/s	Non-zero		Velocity for ramp. If “continue-velocity” is set, the end velocity for the previous segment is used, and this parameter is unused.
2	Duration	s	> 0		Duration of ramp

When preparing the waveform, duration is rounded to the nearest sample period (see 7.7).

If duration is not an exact multiple of the sample period and must be rounded, ramping at the specified velocity for a slightly different duration will cause the end position to be slightly different to that expected.

7.3.2.1.2 Possible errors

Ramp velocity must not be zero.

Duration must be greater than zero.

7.3.2.2 constant-velocity-position

7.3.2.2.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	Ramp velocity	picometres/s	Non-zero		Velocity for ramp. If “continue-velocity” is set, the

Index	Parameter	Units	Minimum	Maximum	Description
					end velocity for the previous segment is used, and this parameter is unused.
2	End position	picometres			End position for ramp

When preparing the waveform, the ramp duration for the specified velocity is rounded to the nearest sample period (see 7.7).

If the ramp duration is not an exact multiple of the sample period and must be rounded, ramping over the specified distance for a slightly different duration will cause the velocity to be slightly different to that specified. If it is critical that the velocity is exactly correct, then the "constant-velocity-duration" segment type (see 7.3.2) should be used instead, maintaining the specified velocity at the cost of a slight change to end position. Alternatively the sample period (see 7.7) may be adjusted so that the ramp duration becomes an exact fit, or a close enough fit for the velocity tolerance.

7.3.2.2.2 Possible errors

Ramp velocity must not be zero.

End position must be reachable with this velocity. If ramp velocity is positive, end position must be greater than start position. If ramp velocity is negative, end position must be lower than start position. End position must not be equal to start position.

7.3.2.3 constant-velocity-distance

7.3.2.3.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If "continue-position" is set, the end position for the previous segment is held, and this parameter is unused.
1	Ramp velocity	picometres/s	Non-zero		Velocity for ramp. If "continue-velocity" is set, the end velocity for the previous segment is used, and this parameter is unused.
2	Distance	picometres	Non-zero		Relative distance of end position from start position.

When preparing the waveform, the ramp duration for the specified velocity is rounded to the nearest sample period (see 7.7).

If the ramp duration is not an exact multiple of the sample period and must be rounded, ramping over the specified distance for a slightly different duration will cause the velocity to be slightly different to that specified. If it is critical that the velocity is exactly correct, then the "constant-velocity-duration" segment type (see 7.3.2) should be used instead, maintaining the specified velocity at the cost of a slight change to end position. Alternatively the sample period (see 7.7) may be adjusted so that the ramp duration becomes an exact fit, or a close enough fit for the velocity tolerance.

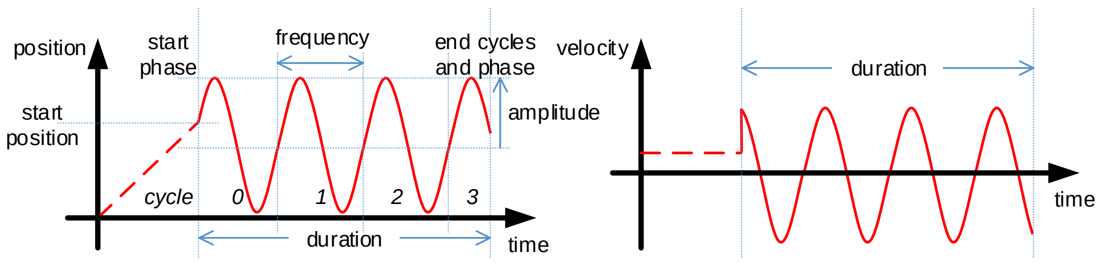
7.3.2.3.2 Possible errors

Ramp velocity must not be zero.

Distance must be reachable with this velocity. If ramp velocity is positive, distance must be positive. If ramp velocity is negative, distance must be negative. Distance must not be zero.

7.3.3 Sine wave

Segment function type	Description
sine	Run a sine wave, or some part of a sine wave.



A sine wave can be run between any phase positions in a cycle, and for multiple cycles. The phase is expressed as a fraction of a complete cycle. For example, starting at a phase of 90 degrees would require a start phase value of 0.25; and finishing at a phase of 270 degrees during the 4th cycle would require an end phase value of 4.75.

Note that the end phase is relative to the sine wave itself, not to the starting phase. Also note that the start phase must be greater than zero. For example, running part of a sine wave from -45 degrees to +45 degrees must be represented by running from a start phase of 0.875 (315 degrees) to an end phase of 1.125 (405 degrees).

The "centre" of the sine wave will depend on the start phase and on the start position at that phase. If the start phase is zero, for example, then the sine wave is starting at the centre and so will be centred around the start position. If the start phase is 0.25 (90 degrees) instead, then the sine wave is starting at a peak and so will be centred around the start position minus the amplitude.

7.3.3.1 sine

7.3.3.1.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If "continue-position" is set, the end position for the previous segment is held, and this parameter is unused.
1	Start phase	Fraction of cycle	0	< 1	Starting phase in cycle, scaled to a fraction of a cycle (0 = 0 degrees, 0.5 = 180 degrees, 1 = 360 degrees)
2	End cycles and phase	Fraction of cycle	0		End cycles and end phase, scaled to a fraction of a cycle (0 = 0 degrees, 0.5 = 180 degrees, 1 = 360 degrees, 2 = 720 degrees i.e. 2 cycles)
3	Amplitude	picometres	0		Sine wave amplitude, from centre to peak
4	Frequency	Hz	> 0		Sine wave frequency

When preparing the waveform, the duration of the sine wave is rounded to the nearest sample period (see 7.7).

If the sine wave duration is not an exact multiple of the sample period and must be rounded, the frequency will be adjusted slightly so that the rounded duration fits the specified start phase, end phase and cycles.

7.3.3.1.2 Possible errors

Start phase must not be negative.

Start phase must be less than 1.

End phase must be greater than start phase.

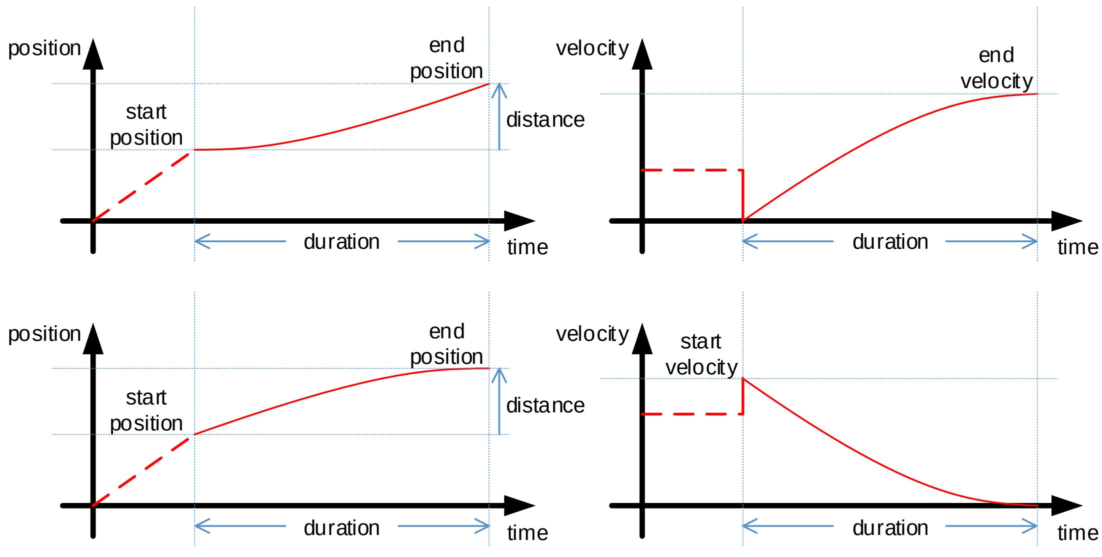
Amplitude must be greater than zero.

Frequency must be greater than zero.

Frequency must be less than half the controller sample rate. For the 6xx0 series controller running at 50kHz, this means the frequency must be less than 25kHz.

7.3.4 Quarter-sine acceleration and deceleration

Segment function type	Description
quarter-sine-accel-duration	Using a quarter-sine profile, accelerate from rest at to specified velocity in specified time.
quarter-sine-accel-position	Using a quarter-sine profile, accelerate from rest at start position to specified velocity at end position.
quarter-sine-accel-distance	Using a quarter-sine profile, accelerate from rest at start position to specified velocity over relative distance.
quarter-sine-decel-duration	Using a quarter-sine profile, decelerate from specified velocity to rest in specified time.
quarter-sine-decel-position	Using a quarter-sine profile, decelerate from specified velocity to rest at end position .
quarter-sine-decel-distance	Using a quarter-sine profile, decelerate from specified velocity to rest over relative distance.



A common technique for minimising resonance during acceleration and deceleration is for the trajectory to follow a portion of a sine wave.

The sine wave quadrant from 270 degrees to 360 degrees starts at rest and finishes at some velocity. Crucially, the final acceleration in this quadrant is zero, making it an appropriate profile for acceleration into a constant-velocity ramp. Similarly, the sine wave quadrant from 0 degrees to 90 degrees starts at some velocity and finishes at rest at a final position, making it appropriate for deceleration from a constant-velocity ramp.

The acceleration types always start from rest, so will ignore the “continue-velocity” setting. The deceleration types may use the “continue-velocity” setting, and will always finish with zero velocity.

Acceleration and deceleration require some time to complete, and some starting/stopping distance. The user has the choice of whether to fix the time taken or the distance travelled, as required for the user’s waveform.

7.3.4.1 quarter-sine-accel-duration

7.3.4.1.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	End velocity	picometres/s	Non-zero		Final velocity after acceleration
2	Duration	s	> 0		Duration of acceleration

When preparing the waveform, the duration of the acceleration is rounded to the nearest sample period (see 7.7).

7.3.4.1.2 Possible errors

End velocity must not be zero.

Duration must be greater than zero.

7.3.4.2 quarter-sine-accel-position

7.3.4.2.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	End position	picometres			Final position after acceleration
2	End velocity	picometres/s	Non-zero		Final velocity after acceleration

When preparing the waveform, the duration of the acceleration is rounded to the nearest sample period (see 7.7).

7.3.4.2.2 Possible errors

End position must be reachable with this velocity. If end velocity is positive, end position must be greater than start position. If end velocity is negative, end position must be lower than start position.

End velocity must not be zero.

7.3.4.3 quarter-sine-accel-distance

7.3.4.3.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	End velocity	picometres/s	Non-zero		Final velocity after acceleration
2	Distance	picometres	Non-zero		Relative distance of end position from start position.

When preparing the waveform, the duration of the acceleration is rounded to the nearest sample period (see 7.7).

7.3.4.3.2 Possible errors

Distance must be reachable with this velocity. If end velocity is positive, distance must be positive. If end velocity is negative, distance must be negative. Distance must not be zero.

End velocity must not be zero.

7.3.4.4 quarter-sine-decel-duration

7.3.4.4.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	Start velocity	picometres/s	Non-zero		If “continue-velocity” is set, the end velocity for the previous segment is held, and this parameter is unused.
2	Duration	s	> 0		Duration of deceleration

When preparing the waveform, the duration of the deceleration is rounded to the nearest sample period (see 7.7).

7.3.4.4.2 Possible errors

Start velocity must not be zero.

Duration must be greater than zero.

7.3.4.5 quarter-sine-decel-position

7.3.4.5.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	Start velocity	picometres/s	Non-zero		If “continue-velocity” is set, the end velocity for the previous segment is held, and this parameter is unused.
2	End position	picometres			Final position after deceleration

When preparing the waveform, the duration of the deceleration is rounded to the nearest sample period (see 7.7).

7.3.4.5.2 Possible errors

Start velocity must not be zero.

End position must be reachable with this velocity. If start velocity is positive, end position must be greater than start position. If start velocity is negative, end position must be lower than start position.

7.3.4.6 quarter-sine-decel-distance

7.3.4.6.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	Start velocity	picometres/s	Non-zero		If “continue-velocity” is set, the end velocity for the previous segment is held, and this parameter is unused.
2	Distance	picometres	Non-zero		Relative distance of end position from start position.

When preparing the waveform, the duration of the deceleration is rounded to the nearest sample period (see 7.7).

7.3.4.6.2 Possible errors

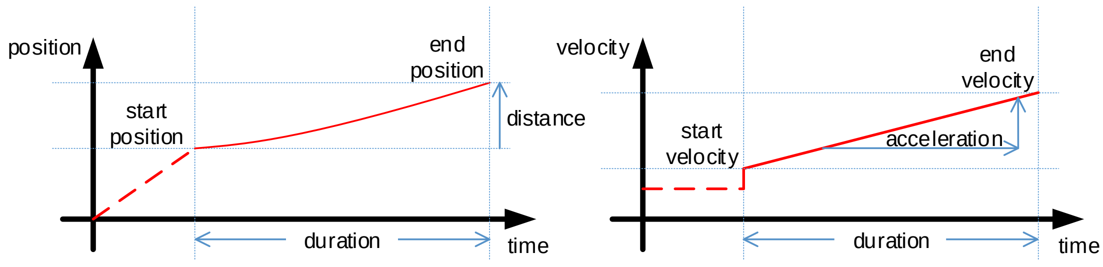
Distance must be reachable with this velocity. If start velocity is positive, distance must be positive. If start velocity is negative, distance must be negative. Distance must not be zero.

Start velocity must not be zero.

7.3.5 Constant acceleration and deceleration (triangular velocity profile)

Segment function type	Description
accel-to-velocity-constant-accel	Accelerate (or decelerate) from current velocity to new velocity with specified acceleration rate.
accel-to-velocity-constant-accel-duration	Accelerate (or decelerate) from current velocity to new velocity in specified time.
accel-to-velocity-constant-accel-position	Accelerate (or decelerate) from current velocity to new velocity at end position.

Segment function type	Description
accel-to-velocity-constant-accel-distance	Accelerate (or decelerate) from current velocity to new velocity to rest over relative distance.



A common acceleration profile for accelerating and decelerating is to apply a constant acceleration/deceleration until the desired change in velocity is achieved. The constant acceleration results in a triangular velocity profile. Acceleration may be specified directly, or the required acceleration may be calculated from the acceleration duration or distance travelled, as required for the user's waveform.

Unlike the quarter-sine acceleration and deceleration functions specified in 7.3.4, this allows changes between non-zero velocities, whereas the quarter-sine functions only handle acceleration from rest and deceleration to rest.

A trapezoidal command trajectory can be constructed from a constant-acceleration segment to accelerate from rest to the desired velocity, a constant-velocity ramp segment to travel the majority of the distance, and a further constant-acceleration segment to decelerate from the ramp velocity to rest at the end position.

7.3.5.1 accel-to-velocity-constant-accel

7.3.5.1.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If "continue-position" is set, the end position for the previous segment is held, and this parameter is unused.
1	Start velocity	picometres/s			If "continue-velocity" is set, the end velocity for the previous segment is held, and this parameter is unused.

Index	Parameter	Units	Minimum	Maximum	Description
2	End velocity	picometres/s			Final velocity after acceleration
3	Acceleration	picometres/s/s	Non-zero		Acceleration rate

Note that both velocity and acceleration are signed. For positive velocity, positive acceleration equates to increased speed in the direction of travel and negative acceleration equates to decreased speed (braking). For negative velocity, negative acceleration equates to increased speed in the direction of travel and positive acceleration equates to decreased speed (braking).

When preparing the waveform, the duration of the acceleration/deceleration is rounded to the nearest sample period (see 7.7).

If the duration is not an exact multiple of the sample period and must be rounded, acceleration will be adjusted slightly so that the rounded duration gives the specified end velocity.

7.3.5.1.2 Possible errors

One of start velocity or end velocity may be zero, but not both.

Acceleration must not be zero.

End velocity must be reachable with this acceleration. If acceleration is positive, end velocity must be greater than start velocity. If acceleration is negative, end position must be lower than start position.

7.3.5.2 accel-to-velocity-constant-accel-duration

7.3.5.2.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	Start velocity	picometres/s			If “continue-velocity” is set, the end velocity for the previous segment is held, and this parameter is unused.
2	End velocity	picometres/s			Final velocity after acceleration

Index	Parameter	Units	Minimum	Maximum	Description
3	Duration	s	> 0		Duration of acceleration

When preparing the waveform, the duration of the acceleration/deceleration is rounded to the nearest sample period (see 7.7).

If the duration is not an exact multiple of the sample period and must be rounded, the expected acceleration will be adjusted slightly so that the rounded duration gives the specified end velocity.

7.3.5.2.2 Possible errors

One of start velocity or end velocity may be zero, but not both.

Duration must be greater than zero.

7.3.5.3 accel-to-velocity-constant-accel-position

7.3.5.3.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	Start velocity	picometres/s			If “continue-velocity” is set, the end velocity for the previous segment is held, and this parameter is unused.
2	End position	picometres			Final position after acceleration
3	End velocity	picometres/s			Final velocity after acceleration

When preparing the waveform, the duration of the acceleration/deceleration is rounded to the nearest sample period (see 7.7).

If the duration is not an exact multiple of the sample period and must be rounded, the expected acceleration will be adjusted slightly so that the rounded duration gives the specified end velocity at the specified end position.

7.3.5.3.2 Possible errors

One of start velocity or end velocity may be zero, but not both.

End position must be reachable from start position with the specified velocities. The limits on this are more complex than for a simple ramp.

7.3.5.4 accel-to-velocity-constant-accel-distance

7.3.5.4.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If "continue-position" is set, the end position for the previous segment is held, and this parameter is unused.
1	Start velocity	picometres/s			If "continue-velocity" is set, the end velocity for the previous segment is held, and this parameter is unused.
2	End velocity	picometres/s			Final velocity after acceleration
3	Distance	picometres	Non-zero		Relative distance of end position from start position.

When preparing the waveform, the duration of the acceleration/deceleration is rounded to the nearest sample period (see 7.7).

If the duration is not an exact multiple of the sample period and must be rounded, the expected acceleration will be adjusted slightly so that the rounded duration gives the specified end velocity at the specified end position.

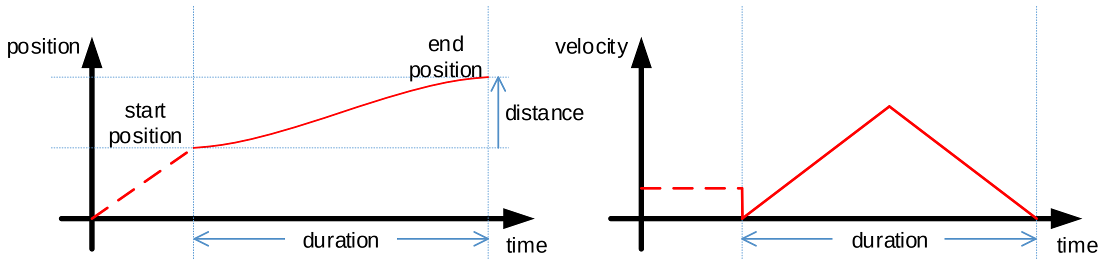
7.3.5.4.2 Possible errors

One of start velocity or end velocity may be zero, but not both.

Distance must be reachable from start position with the specified velocities. The limits on this are more complex than for a simple ramp.

7.3.6 Step with constant acceleration/deceleration (triangular velocity)

Segment function type	Description
step-triangular-velocity-position	Move from rest at start position to rest at end position in specified time, with constant acceleration and deceleration.
step-triangular-velocity-distance	Move from rest at start position to rest at a relative distance in specified time, with constant acceleration and deceleration.



This uses constant symmetrical acceleration and deceleration to round off the edges of a step, which can reduce system resonances. This is an extremely simple profile, but can give reasonable results where more complex trajectory profiles are not required.

7.3.6.1 step-triangular-velocity-position

7.3.6.1.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If “continue-position” is set, the end position for the previous segment is held, and this parameter is unused.
1	End position	picometres			Final position after acceleration
2	Duration	s	> 0		Duration of acceleration

When preparing the waveform, the duration of the step is rounded to the nearest sample period (see 7.7).

7.3.6.1.2 Possible errors

Duration must be greater than zero.

End position must not equal start position.

7.3.6.2 step-triangular-velocity-distance

7.3.6.2.1 Function parameters

Index	Parameter	Units	Minimum	Maximum	Description
0	Start position	picometres			If "continue-position" is set, the end position for the previous segment is held, and this parameter is unused.
1	Distance	picometres	Non-zero		Relative distance of end position from start position.
2	Duration	s	> 0		Duration of acceleration

When preparing the waveform, the duration of the step is rounded to the nearest sample period (see 7.7).

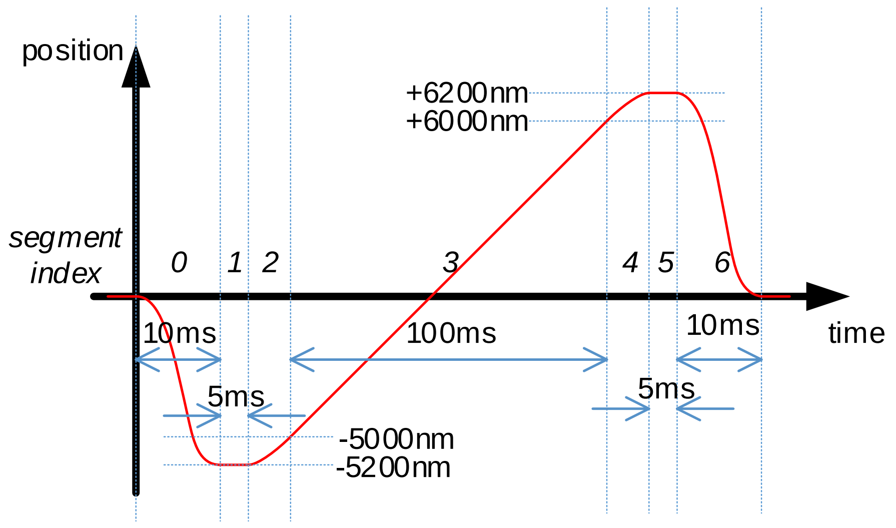
7.3.6.2.2 Possible errors

Duration must be greater than zero.

Distance must not be zero.

7.4 Waveform design example

It may not be clear how to construct a waveform from the component segment types. This section provides a simple worked example as a demonstration, using the commands specified in 7.2 and the function types specified in 7.3.



The example above shows a unidirectional linear ramp from -5000nm to +6000nm over a 100ms duration, with start and end considerations around the ramp. This is a typical operation for many applications, for example focussing a microscope. The waveform carries out the following operations:-

- Travel from the quiescent position to some position before the start of the ramp, using a step with “rounded-off” edges to reduce resonances. Whilst we do not want this to excite system resonances, we are not concerned about the trajectory during this step being particularly accurate, so long as it reaches the position correctly.
- Hold position for a short time to allow any resonances to settle.
- Accelerate to the required ramp rate by the start position for the ramp.
- Ramp at constant velocity, during which time the system will generally be taking measurements or capturing images, depending on the application.
- At the end of the ramp, decelerate to rest.
- Hold position for a short time to allow any resonances to settle.
- Travel back to the quiescent position and stop.

This clearly divides into seven waveform segments.

Segment	Segment type and operation
0	Step from 0nm to -5200nm in 10ms.
1	Hold position for 5ms.
2	Accelerate to required velocity with 200nm “run-up” distance. The ramp covers 11,000nm in 100ms, so the velocity required is 110nm/ms.

Segment	Segment type and operation
3	Ramp from -5000nm to +6000nm in 100ms.
4	Decelerate to rest with 200nm stopping distance.
5	Hold position for 5ms.
6	Step back to 0nm in 10ms.

The example uses constant acceleration profiles for segments 2 and 4, so that the waveform follows a simple trapezoidal trajectory. If this turns out to excite system resonances, the waveform generator provides alternative acceleration profiles which can be used instead. Alternatively it would be possible to set longer "run-up" and stopping distances, allowing acceleration and deceleration to be more progressive.

Similarly, if testing discovered that the "rounded-off" steps in segments 0 and 6 at the start and end did excite system resonances, it would be easy to replace that single-segment step with three segments comprising an alternative acceleration type, a linear ramp and an alternative deceleration type, in the same way as segments 2-4 do. This would give finer control of the waveform trajectory. Alternatively the step could be allowed to take place over a longer time, allowing acceleration and deceleration to be more progressive.

Setting this up in the waveform generator requires the following commands. Note that the controller requires positions to be specified in pm, times to be specified in seconds and velocities to be specified in pm/s, so values are scaled accordingly. For this example, we will assume the stage is connected to channel 1.

Segment	Setting	Command
	Clear waveform	`function.waveform-generator.clear 1`
0	Step to position	`function.waveform-generator.segment.type.set 1 0 step-triangular-velocity-position`
	Start from 0nm	`function.waveform-generator.segment.parameter.set 1 0 0 0`
	Finish at -5200nm	`function.waveform-generator.segment.parameter.set 1 0 1 -5200e+3`
	Duration 10ms	`function.waveform-generator.segment.parameter.set 1 0 2 10e-3`
1	Hold position	`function.waveform-generator.segment.type.set 1 1 constant-position`

Segment	Setting	Command
	Hold at current position	`function.waveform-generator.segment.continue-position-velocity.set 1 1 1 0`
	Hold for 5ms	`function.waveform-generator.segment.parameter.set 1 1 1 5e-3`
2	Accelerate from rest to velocity	`function.waveform-generator.segment.type.set 1 2 accel-to-velocity-constant-accel-position`
	Start at current position and velocity	`function.waveform-generator.segment.continue-position-velocity.set 1 2 1 1`
	Complete acceleration at -5000nm	`function.waveform-generator.segment.parameter.set 1 2 2 -5000e+3`
	Accelerate to 110nm/ms	`function.waveform-generator.segment.parameter.set 1 2 3 110e+6`
3	Linear ramp	`function.waveform-generator.segment.type.set 1 3 constant-velocity-position`
	Continue at this velocity	`function.waveform-generator.segment.continue-position-velocity.set 1 3 1 1`
	Ramp to +6000nm	`function.waveform-generator.segment.parameter.set 1 3 2 +6000e+3`
4	Decelerate to rest	`function.waveform-generator.segment.type.set 1 4 accel-to-velocity-constant-accel-position`
	Start at current position and velocity	`function.waveform-generator.segment.continue-position-velocity.set 1 4 1 1`
	Complete deceleration at +6200nm	`function.waveform-generator.segment.parameter.set 1 4 2 +6200e+3`
	Decelerate to rest (0nm/ms)	`function.waveform-generator.segment.parameter.set 1 4 3 0`

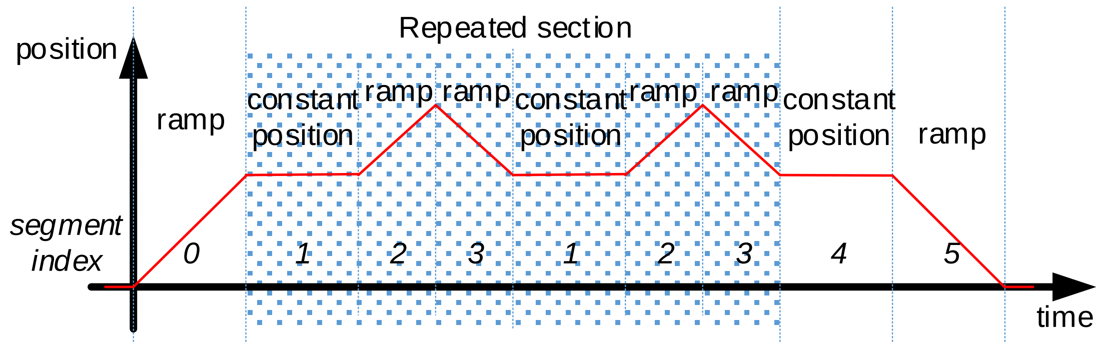
Segment	Setting	Command
5	Hold position	`function.waveform-generator.segment.type.set 1 5 constant-position`
	Hold at current position	`function.waveform-generator.segment.continue-position-velocity.set 1 5 1 0`
	Hold for 5ms	`function.waveform-generator.segment.parameter.set 1 5 1 5e-3`
6	Step to position	`function.waveform-generator.segment.type.set 1 6 step-triangular-velocity-position`
	Start from current position	`function.waveform-generator.segment.continue-position-velocity.set 1 6 1 0`
	Finish at 0nm	`function.waveform-generator.segment.parameter.set 1 6 1 0`
	Duration 10ms	`function.waveform-generator.segment.parameter.set 1 6 2 10e-3`
	Use 7 segments for waveform	`function.waveform-generator.count.set 1 7`

After the waveform has been set up, the user may also wish to set up parts of the waveform to repeat (see 7.5), or have function playback control the digital trigger output (see 7.6).

Once all elements of the waveform have been configured, it must be prepared for playback (see 7.7). It can then be started and stopped with PC commands (see 7.9) or digital trigger inputs (see 7.10).

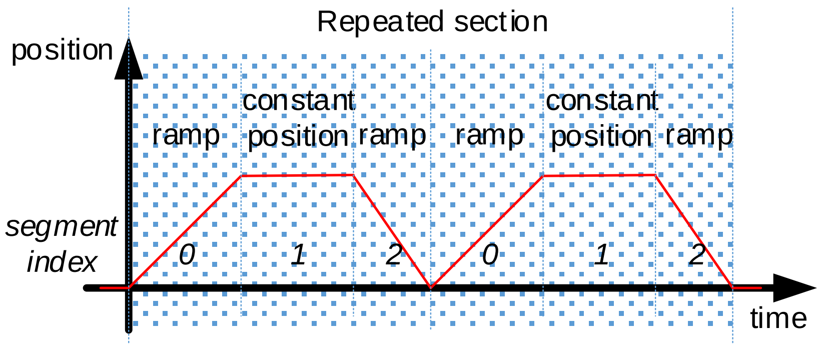
7.5 Waveform design with repeating waveforms

It is often necessary to run some part of the waveform repeatedly. In the example below, the repeated section of the waveform holds a constant position for some time, then carries out a triangular ramp up and down.



The waveform generator allows a section of the waveform to be repeated a number of times as required. The start and end segments of the repeat section and the repeat count are specified. After the section has carried out the specified number of repeats, the remainder of the waveform is run. As the example above shows, repeated waveforms often require an initial section which moves the system to the required position, and a final section which returns the system to zero. Setting the start and end segments for the repeated section allows this to be configured. In the example above, the start segment is 1, the end segment is 3, and the repeat count is 2.

Alternatively if the entire waveform is to be repeated, start or end sections need not be present.



In the example above, the waveform has 3 segments. The start of the repeat section is segment 0 (the first segment) and the end of the repeat section is segment 2 (the last segment), with a repeat count of 2.

Repeated waveforms may also be required to repeat forever. This is selected by setting the repeat count to zero. If the waveform needs to be stopped, the user must stop it manually using an appropriate “stop” command (see 7.9 and 7.10).

Note that the start of the first repeated segment and the end of the last repeated segment must usually be the same position, otherwise there will be a step back to the start position on the first repeated segment. Setting “continue-position” (see 7.2.4) for the first repeated segment is only effective from the segment immediately before, and not when repeating.

7.5.1 function.waveform-generator.repeat-start.get/set

7.5.1.1 Description

Waveform generator start segment for repeating

7.5.1.2 Command

```
function.waveform-generator.repeat-start.get <channel>
```

```
function.waveform-generator.repeat-start.set <channel> <value>
```

7.5.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.5.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	32-bit unsigned integer		0	999	Waveform generator start segment for repeating

7.5.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer		0	999	Waveform generator start segment for repeating

7.5.1.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Segment index must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.5.1.7 Minimum security level

function.waveform-generator.repeat-start.get	No security required
function.waveform-generator.repeat-start.set	User level security required

7.5.1.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.5.2 function.waveform-generator.repeat-end.get/set

7.5.2.1 Description

Waveform generator end segment for repeating

7.5.2.2 Command

```
function.waveform-generator.repeat-end.get <channel>
```

```
function.waveform-generator.repeat-end.set <channel> <value>
```

7.5.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.5.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	32-bit unsigned integer		0	999	Waveform generator end segment for repeating

7.5.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer		0	999	Waveform generator end segment for repeating

7.5.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Segment index must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.5.2.7 Minimum security level

function.waveform-generator.repeat-end.get	No security required
function.waveform-generator.repeat-end.set	User level security required

7.5.2.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.5.3 function.waveform-generator.repeat-count.get/set

7.5.3.1 Description

Waveform generator number of repeats of waveform (0=repeat forever)

7.5.3.2 Command

```
function.waveform-generator.repeat-count.get <channel>
```

```
function.waveform-generator.repeat-count.set <channel> <value>
```

7.5.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.5.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	32-bit unsigned integer		0		Waveform generator number of repeats of waveform 0 = repeat forever 1 = play once (no repeat) 2+ = repeat specified number of times

7.5.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer		0		Waveform generator number of repeats of waveform 0 = repeat forever 1 = play once (no repeat) 2+ = repeat specified number of times

7.5.3.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.5.3.7 Minimum security level

function.waveform-generator.repeat-count.get	No security required
function.waveform-generator.repeat-count.get	User level security required

7.5.3.8 Supported in

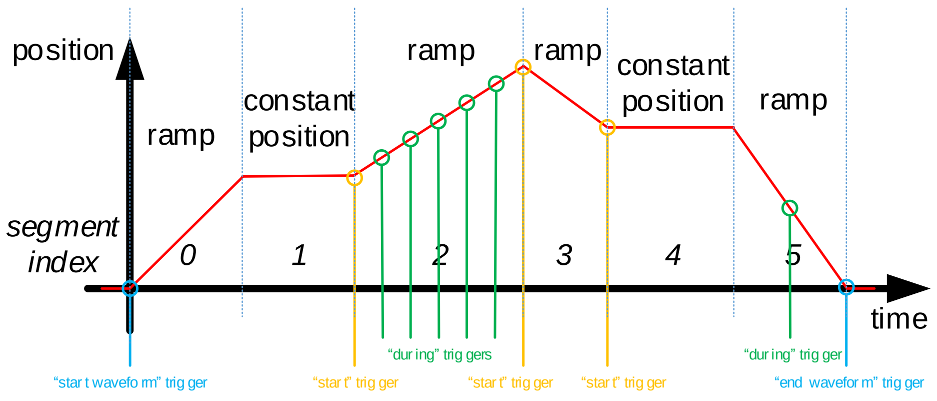
Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.6 Digital trigger outputs

7.6.1 Trigger points during waveform

The controller has one digital trigger output per channel which may be pulsed synchronously with the waveform during function playback to activate external equipment. The controller manual (EN-014635-UM for the 6000 series) specifies digital output connectors and logic levels, which will not be repeated here. Note that each channel only has one trigger output; there is no facility for one channel to trigger multiple outputs.

The trigger output may be activated at the start of a segment, or at evenly-spaced intervals during a segment. There is no provision for triggering at the end of a segment; set a trigger at the start of the next segment instead. Triggers can also be set at the start and end of the entire waveform.



The example above demonstrates the various trigger points available within the function playback waveform. Triggers are set at the start and end of the entire waveform. Segment 2 has a "start" trigger set, and also has "during" triggers set. The "during" triggers are evenly spaced, so setting 5 triggers as in the example above will divide the segment into 6 equal parts. A "start" trigger in segment 3 reports the end of the ramp in segment 2. Segments 4 and 5 show that it is possible to set just a "start" trigger or just a "during" trigger. Using the trigger output is optional, and segment 1 shows that it is also possible to have no triggers set at all.

There is no facility to set triggers at arbitrary intervals in a segment, instead of the evenly spaced "during" triggers. If this is required, the solution is to replace a single segment with two or more segments of the same time, setting the segment starts at the trigger intervals required. For example, if the waveform is required to ramp from 5µm to 10µm and produce one trigger pulse as the ramp reaches 6µm, the waveform generator would be set up with one ramp segment from 5µm to 6µm, and a second ramp segment from 6µm to 10 µm with a "start" trigger set. Note that setting "continue position" and "continue velocity" (see 7.2.4) can make segment configuration easier in this case.

7.6.2 Trigger output event types

Trigger output event type	**Description**
none	No trigger is set at this trigger position
pulse	Trigger output is activated immediately. This is the “traditional” time-based triggering method.
above-command	Trigger output is activated when measured position is greater than trigger command. If measured position is already greater than trigger command, trigger output is activated immediately.
below-command	Trigger output is activated when measured position is lower than trigger command. If measured position is already lower than trigger command, trigger output is activated immediately.
rising-edge	Trigger output is activated when measured position becomes greater than trigger command. If measured position is already greater than trigger command, measured position must fall below trigger command and then cross trigger command again for the trigger output to be activated.
falling-edge	Trigger output is activated when measured position becomes lower than trigger command. If measured position is already lower than trigger command, measured position must fall below trigger command and then cross trigger command again for the trigger output to be activated.
either-edge	Trigger output is activated when measured position crosses trigger command in either direction.
in-position	Trigger output is activated when measured position is within the in-position error threshold (`stage.in-position.error-threshold`, see 5.5.1) of the trigger command. If measured position is already within this range, trigger output is activated immediately.

Older controllers typically only provided simple time-based triggering, where trigger outputs were pulsed at specified times during the waveform. Since the stage movement lags behind the commanded position, the user had to estimate how long this lag would be and allow for it in their trigger times. This was generally a source of inaccuracy in systems, causing issues such as imperfect image focusing.

Whilst time-based triggering is available, the Queensgate controller can also trigger based on position. The controller does not activate the trigger output immediately. Instead it notes the commanded position at the trigger point, and activates the trigger output only when the measured position reaches that trigger position. The system no longer needs to consider the lag between commanded position and stage movement, and accuracy is greatly improved.

Whilst position-based triggering improves trigger accuracy, it is important to select the correct trigger event for the waveform trajectory, otherwise the trigger output may be pulsed too soon or may never be activated at all.

For a positive-going trajectory, the trigger events “above-command” or “rising-edge” would typically be used. Inadvertently using “below-command” would generally result in the trigger output being activated immediately; and using “falling-edge” would generally result in no trigger output at all. Similarly the trigger events “below-command” or “falling-edge” would generally be used for a negative-going trajectory.

The threshold-based and edge-based trigger events all require that the measured position crosses the command position. When stepping to a position with an underdamped system it may take a long time for this to occur; and at the peaks of a sine or sawtooth wave (such as the “start” trigger for segment 3 in the example above) it is possible that the stage may never fully reach that peak before the command changes to pull the stage back down again. As an alternative to threshold-based or edge-based triggering, the “in-position” trigger event activates the trigger output when the measured position is close enough to the command position for the current application’s purposes. This uses the same “in position” error threshold as the control loop (see 5.5.1). In optical applications for example, focusing only needs to be accurate to half a wavelength of the relevant light source.

It is possible that the conditions for a trigger may not have been met before a second trigger event is set later in the waveform. In that case the controller discards the first trigger event and switches to using the second trigger event. This is usually undesirable for external equipment since it generally leads to missed data captures or incorrect operation, so the user must ensure the waveform is constructed such that this is extremely unlikely to occur.

7.6.3 Trigger output pulse duration

To ensure the trigger output pulse can be received by external equipment, the duration of the trigger output pulse can be set.

If the trigger output pulse time is set too long and two or more trigger points are close together, there is a risk that a second trigger will occur whilst the trigger output is already active. In that case the trigger output will be retriggered and will remain active for the specified time from the second trigger. This is usually undesirable for external equipment since it generally leads to missed data captures or incorrect operation, so the user must ensure the waveform is constructed such that this is extremely unlikely to occur.

7.6.4 function.waveform-generator.trigger-output.start-trigger.get/set

7.6.4.1 Description

Waveform generator trigger output event at start, if required

7.6.4.2 Command

```
function.waveform-generator.trigger-output.start-trigger.get <channel>
```

```
function.waveform-generator.trigger-output.start-trigger.set <channel>
<value>
```

7.6.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.6.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	String	Enum			Waveform generator trigger output event at start See 7.6.2 for available trigger events

7.6.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	String	Enum			Waveform generator trigger output event at start See 7.6.2 for available trigger events

7.6.4.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Trigger event must be as specified (set only)

Error return "errcode" string reported	**Description**
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.6.4.7 Minimum security level

function.waveform-generator.trigger-output.start-trigger.get	No security required
function.waveform-generator.trigger-output.start-trigger.set	User level security required

7.6.4.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.6.5 function.waveform-generator.trigger-output.end-trigger.get/set

7.6.5.1 Description

Waveform generator trigger output event at end, if required

7.6.5.2 Command

```
function.waveform-generator.trigger-output.end-trigger.get <channel>
```

```
function.waveform-generator.trigger-output.end-trigger.set <channel>
<value>
```

7.6.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.6.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	String	Enum			Waveform generator trigger output event at end See 7.6.2 for available trigger events

7.6.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	String	Enum			Waveform generator trigger output event at end See 7.6.2 for available trigger events

7.6.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Trigger event must be as specified (set only)

Error return “errcode” string reported	**Description**
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.6.5.7 Minimum security level

function.waveform-generator.trigger-output.end-trigger.get	No security required
function.waveform-generator.trigger-output.end-trigger.set	User level security required

7.6.5.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.6.6 function.waveform-generator.segment.trigger-output.start-trigger.get/set

7.6.6.1 Description

Waveform generator trigger output event at start of segment, if required

7.6.6.2 Command

```
function.waveform-generator.trigger-output.segment.start-trigger.get
<channel> <segment>
```

```
function.waveform-generator.trigger-output.segment.start-trigger.set
<channel> <segment> <value>
```

7.6.6.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.6.6.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index
value	String	Enum			Waveform generator trigger output event at start of segment See 7.6.2 for available trigger events

7.6.6.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	String	Enum			Waveform generator trigger output event at start of segment See 7.6.2 for available trigger events

7.6.6.6 Possible error reports

Error return "errcode" string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range
Value out of range	Trigger event must be as specified (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.6.6.7 Minimum security level

function.waveform-generator.segment.trigger-output.start-trigger.get	No security required
function.waveform-generator.segment.trigger-output.start-trigger.set	User level security required

7.6.6.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.6.7 function.waveform-generator.segment.trigger-output.during-trigger.get/set

7.6.7.1 Description

Waveform generator trigger output events during segment, if required

7.6.7.2 Command

```
function.waveform-generator.trigger-output.segment.during-trigger.get
<channel> <segment>
```

```
function.waveform-generator.trigger-output.segment.during-trigger.set
<channel> <segment> <value>
```

7.6.7.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.6.7.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index
value	String	Enum			Waveform generator trigger output events during segment See 7.6.2 for available trigger events

7.6.7.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	String	Enum			Waveform generator trigger output events during segment See 7.6.2 for available trigger events

7.6.7.6 Possible error reports

Error return “errcode” string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range
Value out of range	Trigger event must be as specified (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.6.7.7 Minimum security level

function.waveform-generator.segment.trigger-output.during-trigger.get	No security required
function.waveform-generator.segment.trigger-output.during-trigger.set	User level security required

7.6.7.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.6.8 function.waveform-generator.segment.trigger-output.during-trigger-count.get/set

7.6.8.1 Description

Waveform generator number of trigger output events during segment

7.6.8.2 Command

```
function.waveform-generator.trigger-output.segment.during-trigger-
count.get <channel> <segment>
```

```
function.waveform-generator.trigger-output.segment.during-trigger-
count.set <channel> <segment> <value>
```

7.6.8.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.6.8.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index
value	32-bit unsigned integer				Waveform generator number of trigger output events during segment

7.6.8.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer				Waveform generator number of trigger output events during segment

7.6.8.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range
Value out of range	Number of trigger output events cannot be greater than the maximum number of samples for function playback (see 7.7)
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.6.8.7 Minimum security level

function.waveform-generator.segment.trigger-output.during-trigger-count.get	No security required
function.waveform-generator.segment.trigger-output.during-trigger-count.set	User level security required

7.6.8.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.6.9 function.trigger-output.pulse-time.get/set

7.6.9.1 Description

Function playback trigger output pulse length

7.6.9.2 Command

```
function.trigger-output.pulse-time.get <channel>
```

```
function.trigger-output.pulse-time.set <channel> <value>
```

7.6.9.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.6.9.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	32-bit floating-point	s	0		Function playback trigger output pulse length Setting this to exactly zero disables trigger output pulses

7.6.9.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	0		Function playback trigger output pulse length Setting this to exactly zero disables trigger output pulses

7.6.9.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

Error return “errcode” string reported	**Description**
Value out of range	Pulse length must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.6.9.7 Minimum security level

function.trigger-output.pulse-time.get	No security required
function.trigger-output.pulse-time.set	User level security required

7.6.9.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7 Waveform check, waveform preparation and sampled data points

7.7.1 Waveform generation and sample rates

Waveform segments are not played directly. Instead, the waveform generator must generate a series of sampled data points for the waveform which can be played back by the controller. The controller allows up to 500,000 points of sampled data in a function generator waveform, which equates to 10s of sample-accurate waveform playback at the controller's 50kHz sample rate.

If a longer waveform is required, the sample rate for function generator waveform playback can be set slower. 500,000 points at a sample rate of 1ms would allow up to 500s of waveform playback, for instance.

A slower sample rate may also be chosen to allow the waveform generator to prepare data points more quickly. The more data points the waveform generator is required to calculate, the longer the preparation process takes. Typically a waveform using all 500,000 points can be prepared within 2-3s, but if the user requires a faster response time between configuring the waveform generator and being able to start function playback, it may be necessary to use a slower sample rate and reduce the number of points in the waveform. At slower sample rates, the controller ramps linearly between data points to reduce the impact of the slower sample rate.

Each waveform generator segment must be fitted to the sample rate. As described in 7.3, this may cause some rounding of segment durations to fit the sample rate. The user may notice some small differences in waveform duration with different sample rates; and may also notice some small differences in velocities and accelerations depending on the function type. As described in 7.3, if the sample rate is reduced to a point where rounding can affect the waveform then different function types may be more appropriate depending on the application.

The effects of reduced sample rate will be most significant in nonlinear waveform segments such as sine waves, where substantially reducing the number of data points will affect how accurately the controller can plot the trajectory. In this case the sample rate should be set faster than the bandwidth of the system, so that errors due to the slower sample rate are naturally "smoothed out" by the system.

7.7.2 Waveform generator check

The first step in waveform generation is to check that the waveform is valid. Parameters for segments must be validated, the time to carry out each segment must be calculated, and the entire waveform must be checked to ensure it fits within the number of sample points available. This check may be done separately, or may be combined with preparing data points.

When the check is complete, the waveform generator allows the user to check each segment's duration, start and end positions, and start and end velocities, as well as the duration for the entire waveform. This may be of use when reporting back details of the waveform on a user interface.

If the check fails, the waveform generator reports which segment has a fault. Again, this will be useful for a user interface to provide guidance.

7.7.3 Waveform generator sampled data point preparation

The preparation step generates sampled data points for each segment, building up the waveform. If the waveform has not been checked, this is done automatically before the data points are generated. Data points cannot be generated if the check fails.

The preparation process can take up to 2-3s, which is too long for a single command to wait for results. Preparation is started with one command, and the user must then check the preparation status repeatedly until it completes.

Once the preparation process has been completed successfully, function playback can be started using the commands in 7.9 and 7.10.

7.7.4 function.waveform-generator.sample-period.get/set

7.7.4.1 Description

Period of samples in generated waveform

7.7.4.2 Command

```
function.waveform-generator.sample-period.get <channel>
```

```
function.waveform-generator.sample-period set <channel> <value>
```

7.7.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.7.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	32-bit floating-point	s	0		Period of samples in generated waveform

The sample period will be rounded to the nearest multiple of the controller’s sample period (20µs for the controller’s 50kHz sample rate). The result returned may therefore not be exactly the value set, if rounding is required.

7.7.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	0		Period of samples in generated waveform

7.7.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Sample period must be greater than zero (set only)

Error return “errcode” string reported	Description
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.7.4.7 Minimum security level

function.waveform-generator.sample-period.get	No security required
function.waveform-generator.sample-period.set	User level security required

7.7.4.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.5 function.waveform-generator.check-waveform

7.7.5.1 Description

Check waveform segments and calculate segment boundaries

7.7.5.2 Command

```
function.waveform-generator.check-waveform <channel>
```

7.7.5.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.7.5.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer		1	1	Always returns 1 if check succeeds. If check fails, returns an error.

7.7.5.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	The waveform is not configured correctly. Use `function.waveform-generator.prepare-waveform-status.get` (see 7.7.8) to find which waveform segment is causing the problem
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed

7.7.5.6 Minimum security level

function.waveform-generator.check-waveform	User level security required

7.7.5.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.6 function.waveform-generator.prepare-waveform

7.7.6.1 Description

Prepare waveform for playback, calculating sample data points

7.7.6.2 Command

```
function.waveform-generator.prepare-waveform <channel>
```

7.7.6.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.7.6.4 Results

Name	Type	Units	Minimum	Maximum	Description
status	32-bit unsigned integer		1	1	Always returns 1 if prepare can be started

7.7.6.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed

7.7.6.6 Minimum security level

function.waveform-generator.prepare-waveform	User level security required

7.7.6.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.7 function.waveform-generator.prepare-waveform-status.get

7.7.7.1 Description

Status of waveform preparation

7.7.7.2 Command

```
function.waveform-generator.prepare-waveform-status.get <channel>
```

7.7.7.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.7.7.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	String	Enum			Status of waveform preparation “idle”: Not running, and generated waveform sample data points successfully “error”: Not running, and waveform has an error which caused waveform preparation to fail. Use `function.waveform-generator.prepare-waveform-status.get` (see 7.7.8) to find which waveform segment is causing the problem. “in-progress”: Waveform generator is still generating sample data points

7.7.7.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

7.7.7.6 Minimum security level

function.waveform-generator.prepare-waveform-status.get	No security required

7.7.7.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.8 function.waveform-generator.failed-at-segment-index.get

7.7.8.1 Description

If waveform generation failed, reports the segment index at which an error occurred when last running waveform check (see `function.waveform-generator.check-waveform`, 7.7.5) or prepare (see `function.waveform-generator.prepare-waveform`, 7.7.6)

7.7.8.2 Command

```
function.waveform-generator.failed-at-segment-index.get <channel>
```

7.7.8.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.7.8.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit signed integer				Segment index at which an error occurred Returns -1 if last check or prepare completed successfully

7.7.8.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

7.7.8.6 Minimum security level

function.waveform-generator.failed-at-segment-index.get	No security required

7.7.8.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.9 function.waveform-generator.waveform-duration.get

7.7.9.1 Description

Total duration for generated waveform

7.7.9.2 Command

```
function.waveform-generator.waveform-duration.get <channel>
```

7.7.9.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.7.9.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	0		Total duration for generated waveform

7.7.9.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Waveform check (see `function.waveform-generator.check-waveform`, 7.7.5) or prepare (see `function.waveform-generator.prepare-waveform`, 7.7.6) has not been run successfully since waveform configuration was last changed.

7.7.9.6 Minimum security level

function.waveform-generator.waveform-duration.get	No security required

7.7.9.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.10 function.waveform-generator.segment.duration.get

7.7.10.1 Description

Waveform generator segment duration

7.7.10.2 Command

```
function.waveform-generator.segment.duration.get <channel> <segment>
```

7.7.10.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.7.10.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	0		Waveform generator segment duration

7.7.10.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range
Command could not be carried out	Waveform check (see `function.waveform-generator.check-waveform`, 7.7.5) or prepare (see `function.waveform-generator.prepare-waveform`, 7.7.6) has not been run successfully since waveform configuration was last changed.

7.7.10.6 Minimum security level

function.waveform-generator.segment.duration.get	No security required

7.7.10.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.11 function.waveform-generator.segment.start-position.get

7.7.11.1 Description

Waveform generator segment start position

7.7.11.2 Command

```
function.waveform-generator.segment.start-position.get <channel>
<segment>
```

7.7.11.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.7.11.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm			Waveform generator segment start position

7.7.11.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range
Command could not be carried out	Waveform check (see `function.waveform-generator.check-waveform`, 7.7.5) or prepare (see `function.waveform-generator.prepare-waveform`, 7.7.6) has not been run successfully since waveform configuration was last changed.

7.7.11.6 Minimum security level

function.waveform-generator.segment.start-position.get	No security required

7.7.11.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.12 function.waveform-generator.segment.start-velocity.get

7.7.12.1 Description

Waveform generator segment start velocity

7.7.12.2 Command

```
function.waveform-generator.segment.start-velocity.get <channel>
<segment>
```

7.7.12.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.7.12.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm/s			Waveform generator segment start velocity

7.7.12.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range
Command could not be carried out	Waveform check (see `function.waveform-generator.check-waveform`, 7.7.5) or prepare (see `function.waveform-generator.prepare-waveform`, 7.7.6) has not been run successfully since waveform configuration was last changed.

7.7.12.6 Minimum security level

function.waveform-generator.segment.start-velocity.get	No security required

7.7.12.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.13 function.waveform-generator.segment.end-position.get

7.7.13.1 Description

Waveform generator segment end position

7.7.13.2 Command

```
function.waveform-generator.segment.end-position.get <channel>
<segment>
```

7.7.13.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.7.13.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm			Waveform generator segment end position

7.7.13.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range
Command could not be carried out	Waveform check (see `function.waveform-generator.check-waveform`, 7.7.5) or prepare (see `function.waveform-generator.prepare-waveform`, 7.7.6) has not been run successfully since waveform configuration was last changed.

7.7.13.6 Minimum security level

function.waveform-generator.segment.end-position.get	No security required

7.7.13.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.7.14 function.waveform-generator.segment.end-velocity.get

7.7.14.1 Description

Waveform generator segment end velocity

7.7.14.2 Command

```
function.waveform-generator.segment.end-velocity.get <channel>
<segment>
```

7.7.14.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
segment	32-bit unsigned integer		0	999	Segment index

7.7.14.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm/s			Waveform generator segment end velocity

7.7.14.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Segment index must be within specified range
Command could not be carried out	Waveform check (see `function.waveform-generator.check-waveform`, 7.7.5) or prepare (see `function.waveform-generator.prepare-waveform`, 7.7.6) has not been run successfully since waveform configuration was last changed.

7.7.14.6 Minimum security level

function.waveform-generator.segment.end-velocity.get	No security required

7.7.14.7 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.8 Waveform configuration using sampled data points

As well as using the waveform generator to construct function playback waveforms, it is also possible to directly edit the sampled data points in the waveform. Waveform data may be read back or modified after the waveform generator has generated the sampled data points. Alternatively the entire waveform may be set from scratch, point by point.

The commands described in this section to inspect and modify the waveform data points are exactly equivalent to commands and features already covered for the waveform generator, so descriptions of the features available for waveforms will not be repeated. Where the waveform generator's index values refer to a segment index, these are replaced in the following commands by a data point index within the sampled data. Similarly the "count" refers to a number of data points and not to a number of segments. As described in 7.7.1, the waveform for a channel is limited to a maximum of 10s of sample-accurate data, which on the 6000-series controller is 500,000 data points at a maximum sample rate of 50kHz.

One feature available at the sampled data point level which is not available for the waveform generator is the ability to set trigger points (see 7.6) at arbitrary positions within the waveform. Trigger points in the waveform generator are constrained by the limits of segments. When using sampled data points though, trigger output events can be set freely for any points within the waveform.

Another feature available at the sampled data point level which is not available for the waveform generator is the ability to select "staircase steps" for samples. When the sample rate is reduced, by default the waveform ramps linearly between sample points (as described in 7.7.1). It is possible to select on a point-by-point basis whether to use linear ramps between points, or whether to step directly to each new point and hold that value until the next point. The majority of applications will not use this, because it greatly increases the inaccuracy of any waveform following a ramp or curve. However it can be useful for applications which require a hard-edged step between points, for example if setting up a square wave to test the system's slew rate.

7.8.1 function.waveform.data.get/set

7.8.1.1 Description

Function playback waveform data.

7.8.1.2 Command

```
function.waveform.data.get <channel> <index>
```

```
function.waveform.data.set <channel> <index> <value>
```

7.8.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
index	32-bit unsigned integer	samples	0	See 7.8	Sample index for data point

7.8.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
index	32-bit unsigned integer	samples	0	See 7.8	Sample index for data point
value	32-bit floating-point	pm			Data value at this point

7.8.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm			Data value at this point

7.8.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Sample index must be within specified range
Value out of range	Data value must be within specified range (set only)

Error return "errcode" string reported	**Description**
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.8.1.7 Minimum security level

function.waveform.data.get	No security required
function.waveform.data.set	User level security required

7.8.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.8.2 function.waveform.count.get/set

7.8.2.1 Description

Function playback waveform data length

7.8.2.2 Command

```
function.waveform.count.get <channel>
```

```
function.waveform.count.set <channel> <value>
```

7.8.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.8.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	samples	1	See 7.8	Waveform data length

7.8.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	1	See 7.8	Waveform data length

7.8.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Data length must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.8.2.7 Minimum security level

function.waveform.count.get	No security required
function.waveform.count.set	User level security required

7.8.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.8.3 function.waveform.soft-stop-at-end.get/set

7.8.3.1 Description

Function playback soft-stops at end (true) or steps to zero (false)

7.8.3.2 Command

```
function.waveform.soft-stop-at-end.get <channel>
```

```
function.waveform.soft-stop-at-end.set <channel> <value>
```

7.8.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.8.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer		0	1	Function playback soft-stops at end (true) or steps to zero (false)

7.8.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer		0	1	Function playback soft-stops at end (true) or steps to zero (false)

7.8.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Value must be as specified
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.8.3.7 Minimum security level

function.waveform.soft-stop-at-end.get	No security required
function.waveform.soft-stop-at-end.set	User level security required

7.8.3.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.8.4 function.waveform.repeat-start.get/set

7.8.4.1 Description

Function playback waveform start point for repeating

7.8.4.2 Command

```
function.waveform.repeat-start.get <channel>

function.waveform.repeat-start.set <channel> <value>
```

7.8.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.8.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	samples	0	See 7.8	Function playback waveform start point for repeating

7.8.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	0	See 7.8	Function playback waveform start point for repeating

7.8.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Start index must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.8.4.7 Minimum security level

function.waveform.repeat-start.get	No security required
function.waveform.repeat-start.set	User level security required

7.8.4.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.8.5 function.waveform.repeat-end.get/set

7.8.5.1 Description

Function playback waveform end point for repeating

7.8.5.2 Command

```
function.waveform.repeat-end.get <channel>
```

```
function.waveform.repeat-end.set <channel> <value>
```

7.8.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.8.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	samples	0	See 7.8	Function playback waveform end point for repeating

7.8.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	0	See 7.8	Function playback waveform end point for repeating

7.8.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	End index must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.8.5.7 Minimum security level

function.waveform.repeat-end.get	No security required
function.waveform.repeat-end.set	User level security required

7.8.5.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.8.6 function.waveform.repeat-count.get/set

7.8.6.1 Description

Function playback number of repeats of waveform (0=repeat forever)

7.8.6.2 Command

```
function.waveform.repeat-count.get <channel>
```

```
function.waveform.repeat-count.set <channel> <value>
```

7.8.6.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.8.6.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer				Function playback number of repeats of waveform 0 = repeat forever 1 = run once (as normal) 2+ = repeat this many times

7.8.6.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer				Function playback number of repeats of waveform 0 = repeat forever 1 = run once (as normal) 2+ = repeat this many times

7.8.6.6 Possible error reports

Error return “errcode” string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.8.6.7 Minimum security level

function.waveform.repeat-count.get	No security required
function.waveform.repeat-count.set	User level security required

7.8.6.8 Supported in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

7.8.7 function.waveform.trigger-out-event.get/set

7.8.7.1 Description

Function playback waveform trigger output event

7.8.7.2 Command

```
function.waveform.trigger-out-event.get <channel> <index>
```

```
function.waveform.trigger-out-event.set <channel> <index> <value>
```

7.8.7.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
index	32-bit unsigned integer	samples	0	See 7.8	Sample index for data point

7.8.7.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
index	32-bit unsigned integer	samples	0	See 7.8	Sample index for data point
value	String	Enum			Trigger output event See 7.6 for available trigger event types

7.8.7.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	String	Enum			Trigger output event See 7.6 for available trigger event types

7.8.7.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller

Error return "errcode" string reported	**Description**
Index out of range	Sample index must be within specified range
Value out of range	Value must be as specified
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.8.7.7 Minimum security level

function.waveform.trigger-out-event.get	No security required
function.waveform.trigger-out-event.set	User level security required

7.8.7.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.8.8 function.waveform.sample-period.get/set

7.8.8.1 Description

Period of samples for function playback

7.8.8.2 Command

```
function.waveform.sample-period.get <channel>
```

```
function.waveform.sample-period.set <channel> <value>
```

7.8.8.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel

7.8.8.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Function playback channel
value	32-bit floating-point	s	0		Period of samples for function playback

The sample period will be rounded to the nearest multiple of the controller's sample period (20µs for the controller's 50kHz sample rate). The result returned may therefore not be exactly the value set, if rounding is required.

7.8.8.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	s	0		Period of samples for function playback

7.8.8.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Sample period must be greater than zero (set only)

Error return “errcode” string reported	Description
Command could not be carried out	Function playback is running or waiting for digital input trigger, or waveform generator prepare is in progress, so configuration cannot be changed (set only)

7.8.8.7 Minimum security level

function.waveform.sample-period.get	No security required
function.waveform.sample-period.set	User level security required

7.8.8.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.8.9 function.waveform.command-transition.get/set

7.8.9.1 Description

Function playback waveform ramp or step to command

7.8.9.2 Command

```
function.waveform.command-transition.get <channel> <index>
```

```
function.waveform.command-transition.set <channel> <index> <value>
```

7.8.9.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
index	32-bit unsigned integer	samples	0	See 7.8	Sample index for data point

7.8.9.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
index	32-bit unsigned integer	samples	0	See 7.8	Sample index for data point
value	Boolean		0	1	Sample transition 0 = step to sample value 1 = ramp to sample value over `function.waveform.command-transition` period

7.8.9.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	Boolean		0	1	Sample transition 0 = step to sample value 1 = ramp to sample value over `function.waveform.command-transition` period

7.8.9.6 Possible error reports

Error return "errcode" string reported	**Description**
Channel number invalid	Channel parameter is larger than number of channels on controller
Index out of range	Sample index must be within specified range
Value out of range	Number of steps must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.8.9.7 Minimum security level

function.waveform.command-transition.get	No security required
function.waveform.command-transition.set	User level security required

7.8.9.8 Supported in

Controller application firmware	6.2.8 onwards
Controller interface library	2.2.4 onwards

7.9 Function playback control from host PC

Function playback may be started, paused, unpaused and stopped from the host PC. This may be carried out for a single channel individually, or for multiple channels synchronously. Snapshot capture may also be triggered synchronously with function playback.

Once function playback is started, a function with a fixed number of waveform iterations will run to completion and will automatically stop at that point, unless the user chooses to stop it before it has completed. A function which is set to run indefinitely will never complete, of course, and therefore must be stopped by the user.

When stopping function playback, the user has the option of a “hard” or “soft” stop in the same way as when playback stops naturally at the end (see 7.2). On a “soft” stop, the digital position command value is set to the combined position command (see the diagram in 5.1.2), so that the absolute commanded position holds its value at the time the function stops. A “hard” stop simply sets the function playback command (see the diagram in 5.1.2) to zero, so that the commanded position has a step-change back to the digital position command. The relevant stop command to use will depend on the user’s requirements.

Function playback may be paused at any time. The function playback commanded position will hold its value until function playback is unpaused, and will then resume from that point. Function playback may be stopped (see below) whilst the function is paused, if required. Note that pausing function playback does not affect other position command sources, which will still contribute to the absolute position command as shown in 5.1.2.

Commands allow all channels to be controlled simultaneously, setting flags for the channels to be controlled as required. This ensures function playback for multiple channels and/or snapshot capture is entirely synchronous. When only a subset of channels require function playback to be controlled, care must be taken to ensure the state of other channels is not accidentally affected.

Repeated commands have no effect. In particular, if function playback is currently running then a start command will have no effect. There is no facility to retrigger a function, only to stop it and restart it as separate commands.

The number of parameters and return values for the following commands will vary depending on the number of controller channels. Where the controller only has 2 channels, the parameter for channel 3 is not required, and there will be no return value for channel 3. Similarly for a single-channel controller, parameters for channels 2 and 3 are not required, and there will be no return values for channels 2 and 3. The interface DLL ignores redundant parameters, so providing extra parameters for missing channels will not cause a problem. However calling code will need to be aware that the number of return values will be dependent on the number of controller channels, and must not assume that return values for channels 2 and 3 will always be present.

7.9.1 function.state.get

7.9.1.1 Description

Report the state of function playback and snapshot capture

7.9.1.2 Command

```
function.state.get
```

7.9.1.3 Parameters

None

7.9.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
running-snapshot	Boolean		0	1	Running snapshot capture
running-internal-channel0	Boolean		0	1	Running function playback for internal channel 0
running-channel1	Boolean		0	1	Running function playback for channel 1
running-channel2	Boolean		0	1	Running function playback for channel 2 (**only returned for 2- or 3-channel controller)**
running-channel3	Boolean		0	1	Running function playback for channel 3 (**only returned for 3-channel controller)**
paused-internal-channel0	Boolean		0	1	Paused function playback for internal channel 0
paused-channel1	Boolean		0	1	Paused function playback for channel 1
paused-channel2	Boolean		0	1	Paused function playback for channel 2 (**only returned for 2- or 3-channel controller)**
paused-channel3	Boolean		0	1	Paused function playback for channel 3 (**only returned for 3-channel controller)**

7.9.1.5 Possible error reports

None

7.9.1.6 Minimum security level

function.state.get	No security required

7.9.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.9.2 function.command.start

7.9.2.1 Description

Start function playback and/or snapshot

7.9.2.2 Command

```
function.command.start <start-snapshot> <start-internal-channel0>
<start-channel1> [start-channel2] [start-channel3]
```

7.9.2.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
start-snapshot	Boolean		0	1	Start snapshot capture
start-internal-channel0	Boolean		0	1	Start function playback for internal channel 0
start-channel1	Boolean		0	1	Start function playback for channel 1
start-channel2	Boolean		0	1	Start function playback for channel 2 (**only required for 2- or 3-channel controller)**
start-channel3	Boolean		0	1	Start function playback for channel 3 (**only required for 3-channel controller)**

7.9.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
start-snapshot	Boolean		0	1	Started snapshot capture
start-internal-channel0	Boolean		0	1	Started function playback for internal channel 0
start-channel1	Boolean		0	1	Started function playback for channel 1
start-channel2	Boolean		0	1	Started function playback for channel 2 (**only returned for 2- or 3-channel controller)**

Name	Type	Units	Minimum	Maximum	Description
start-channel3	Boolean		0	1	Started function playback for channel 3 (**only returned for 3-channel controller)**

7.9.2.5 Possible error reports

None

Note that if a flag is set for a channel which has no stage present, the command for that channel will be ignored. Since this command controls multiple channels, this is preferable to reporting an error when one channel cannot be controlled but other channels can. The return value for that channel will return zero, to indicate that the command could not be carried out.

7.9.2.6 Minimum security level

function.command.start	User level security required

7.9.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.9.3 function.command.stop

7.9.3.1 Description

Hard-stop function playback and/or stop snapshot

7.9.3.2 Command

```
function.command.stop <stop-snapshot> <stop-internal-channel0> <stop-
channel1> [stop-channel2] [stop-channel3]
```

7.9.3.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
stop-snapshot	Boolean		0	1	Stop snapshot
stop-internal-channel0	Boolean		0	1	Stop function playback for internal channel 0
stop-channel1	Boolean		0	1	Stop function playback for channel 1
stop-channel2	Boolean		0	1	Stop function playback for channel 2 (**only required for 2- or 3-channel controller)**
stop-channel3	Boolean		0	1	Stop function playback for channel 3 (**only required for 3-channel controller)**

7.9.3.4 Results

Name	Type	Units	Minimum	Maximum	Description
stop-snapshot	Boolean		0	1	Stopped snapshot capture
stop-internal-channel0	Boolean		0	1	Stopped function playback for internal channel 0
stop-channel1	Boolean		0	1	Stopped function playback for channel 1
stop-channel2	Boolean		0	1	Stopped function playback for channel 2 (**only returned for 2- or 3-channel controller)**

Name	Type	Units	Minimum	Maximum	Description
stop-channel3	Boolean		0	1	Stopped function playback for channel 3 (**only returned for 3-channel controller)**

7.9.3.5 Possible error reports

None

Note that if a flag is set for a channel which has no stage present, the command for that channel will be ignored. Since this command controls multiple channels, this is preferable to reporting an error when one channel cannot be controlled but other channels can. The return value for that channel will return zero, to indicate that the command could not be carried out.

7.9.3.6 Minimum security level

function.command.stop	User level security required

7.9.3.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.9.4 function.command.soft-stop

7.9.4.1 Description

Soft-stop function playback and/or snapshot

7.9.4.2 Command

```
function.command.soft-stop <stop-snapshot> <stop-internal-channel0>
<stop-channel1> [stop-channel2] [stop-channel3]
```

7.9.4.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
stop-snapshot	Boolean		0	1	Stop snapshot
stop-internal-channel0	Boolean		0	1	Stop function playback for internal channel 0
stop-channel1	Boolean		0	1	Stop function playback for channel 1
stop-channel2	Boolean		0	1	Stop function playback for channel 2 (**only required for 2- or 3-channel controller)**
stop-channel3	Boolean		0	1	Stop function playback for channel 3 (**only required for 3-channel controller)**

7.9.4.4 Results

Name	Type	Units	Minimum	Maximum	Description
stop-snapshot	Boolean		0	1	Stopped snapshot capture
stop-internal-channel0	Boolean		0	1	Stopped function playback for internal channel 0
stop-channel1	Boolean		0	1	Stopped function playback for channel 1
stop-channel2	Boolean		0	1	Stopped function playback for channel 2 (**only returned for 2- or 3-channel controller)**

Name	Type	Units	Minimum	Maximum	Description
stop-channel3	Boolean		0	1	Stopped function playback for channel 3 (**only returned for 3-channel controller)**

7.9.4.5 Possible error reports

None

Note that if a flag is set for a channel which has no stage present, the command for that channel will be ignored. Since this command controls multiple channels, this is preferable to reporting an error when one channel cannot be controlled but other channels can. The return value for that channel will return zero, to indicate that the command could not be carried out.

7.9.4.6 Minimum security level

function.command.stop	User level security required

7.9.4.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.9.5 function.command.pause

7.9.5.1 Description

Pause function playback

7.9.5.2 Command

```
function.command.pause <pause-snapshot> <pause-internal-channel0>
<pause-channel1> [pause-channel2] [pause-channel3]
```

7.9.5.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
pause-internal-channel0	Boolean		0	1	Pause function playback for internal channel 0
pause-channel1	Boolean		0	1	Pause function playback for channel 1
pause-channel2	Boolean		0	1	Pause function playback for channel 2 (**only required for 2- or 3-channel controller)**
pause-channel3	Boolean		0	1	Pause function playback for channel 3 (**only required for 3-channel controller)**

7.9.5.4 Results

Name	Type	Units	Minimum	Maximum	Description
pause-internal-channel0	Boolean		0	1	Paused function playback for internal channel 0
pause-channel1	Boolean		0	1	Paused function playback for channel 1
pause-channel2	Boolean		0	1	Paused function playback for channel 2 (**only returned for 2- or 3-channel controller)**

Name	Type	Units	Minimum	Maximum	Description
pause-channel3	Boolean		0	1	Paused function playback for channel 3 (**only returned for 3-channel controller)**

7.9.5.5 Possible error reports

None

Note that if a flag is set for a channel which has no stage present, the command for that channel will be ignored. Since this command controls multiple channels, this is preferable to reporting an error when one channel cannot be controlled but other channels can. The return value for that channel will return zero, to indicate that the command could not be carried out.

7.9.5.6 Minimum security level

function.command.pause	User level security required

7.9.5.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.9.6 function.command.unpause

7.9.6.1 Description

Unpause function playback

7.9.6.2 Command

```
function.command.unpause <unpause-snapshot> <unpause-internal-
channel0> <unpause-channel1> [unpause-channel2] [unpause-channel3]
```

7.9.6.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
unpause-internal-channel0	Boolean		0	1	Unpause function playback for internal channel 0
unpause-channel1	Boolean		0	1	Unpause function playback for channel 1
unpause-channel2	Boolean		0	1	Unpause function playback for channel 2 (**only required for 2- or 3-channel controller)**
unpause-channel3	Boolean		0	1	Unpause function playback for channel 3 (**only required for 3-channel controller)**

7.9.6.4 Results

Name	Type	Units	Minimum	Maximum	Description
unpause-internal-channel0	Boolean		0	1	Unpaused function playback for internal channel 0
unpause-channel1	Boolean		0	1	Unpaused function playback for channel 1
unpause-channel2	Boolean		0	1	Unpaused function playback for channel 2 (**only returned for 2- or 3-channel controller)**

Name	Type	Units	Minimum	Maximum	Description
unpause-channel3	Boolean		0	1	Unpaused function playback for channel 3 (**only returned for 3-channel controller)**

7.9.6.5 Possible error reports

None

Note that if a flag is set for a channel which has no stage present, the command for that channel will be ignored. Since this command controls multiple channels, this is preferable to reporting an error when one channel cannot be controlled but other channels can. The return value for that channel will return zero, to indicate that the command could not be carried out.

7.9.6.6 Minimum security level

function.command.unpause	User level security required

7.9.6.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.9.6.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10 Function playback control from digital trigger inputs

The controller has three digital inputs which may be used to trigger function playback and snapshot capture. The manual for each controller specifies digital input connectors and logic levels.

Digital trigger inputs may be used to control function playback for individual channels, or may be used to control function playback on multiple channels synchronously, as well as controlling snapshot capture. All control commands from the host PC as identified in section 7.9 may be configured to be triggered by digital inputs, which potentially allows function playback and snapshot capture to be controlled from external electronics without requiring the host PC. Host PC control commands may still be used as required, in addition to any digital trigger input actions.

Each control command (start, hard-stop, soft-stop, pause, unpause) for each function playback channel and snapshot capture may be linked to a digital input trigger. Available triggers are:-

- Input high
- Input low
- Input rising-edge
- Input falling-edge

To clarify the difference in behaviour between “high” and “rising-edge”, if the input is initially high and function playback is set to start on “high”, then function playback will be started as soon as “high” is selected. If the input is high when function playback completes, then function playback will immediately be restarted. Conversely, if the input is initially high and “rising-edge” is selected, the input must go low and high again before function playback is started; and after function playback completes, the input must again go low and high again before function playback is restarted. “Low” and “falling-edge” behave similarly.

As with control commands from the host PC, repeated commands have no effect. Once function playback is running, further start commands (whether from a “high” or “low” input held in that state, or from transitions on a “rising-edge” or “falling-edge” input) have no effect until the function completes or stops.

Using a trigger input for one command does not prevent it also being used for other commands. An obvious example would be configuring “high” on one input to trigger start, and “low” on the same input to trigger stop. The same input could even be used to trigger different behaviour on different channels; for example, if movement on one axis requires the other axis to be stationary, high and low could trigger start and stop on one channel, and could trigger pause and unpause on the second channel.

Once trigger inputs are configured, digital input triggers for that channel must be enabled, at which point the function playback waveform and configuration may not be changed. To make changes to function playback for this channel, digital input triggers for that channel must be disabled and any running function playback must have completed or stopped.

When capturing a snapshot, the area of interest may be some time after a function has started. When commanding function playback and snapshot capture from the host PC, the host PC can simply wait for some time before commanding a snapshot, but this may not be possible when commanding function playback and snapshot capture from digital inputs. Snapshot capture may therefore have a delay set between an input trigger and snapshot capture starting, so that both function playback and snapshot capture can be triggered from the same input.

7.10.1 function.trigger-inputs.start.get/set

7.10.1.1 Description

Function playback channel start trigger inputs

7.10.1.2 Command

```
function.trigger-inputs.start.get <channel>
```

```
function.trigger-inputs.start.set <channel> <value>
```

7.10.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.10.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.10.1.7 Minimum security level

function.trigger-inputs.start.get	No security required
function.trigger-inputs.start.get	User level security required

7.10.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.2 function.trigger-inputs.stop.get/set

7.10.2.1 Description

Function playback channel hard-stop trigger inputs

7.10.2.2 Command

```
function.trigger-inputs.stop.get <channel>
```

```
function.trigger-inputs.stop.set <channel> <value>
```

7.10.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.10.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.10.2.7 Minimum security level

function.trigger-inputs.stop.get	No security required
function.trigger-inputs.stop.get	User level security required

7.10.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.3 function.trigger-inputs.soft-stop.get/set

7.10.3.1 Description

Function playback channel soft-stop trigger inputs

7.10.3.2 Command

```
function.trigger-inputs.soft-stop.get <channel>
```

```
function.trigger-inputs.soft-stop.set <channel> <value>
```

7.10.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.10.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.10.3.7 Minimum security level

function.trigger-inputs.soft-stop.get	No security required
function.trigger-inputs.soft-stop.get	User level security required

7.10.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.4 function.trigger-inputs.pause.get/set

7.10.4.1 Description

Function playback channel pause trigger inputs

7.10.4.2 Command

```
function.trigger-inputs.pause.get <channel>
```

```
function.trigger-inputs.pause.set <channel> <value>
```

7.10.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.10.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.10.4.7 Minimum security level

function.trigger-inputs.pause.get	No security required
function.trigger-inputs.pause.get	User level security required

7.10.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.5 function.trigger-inputs.unpause.get/set

7.10.5.1 Description

Function playback channel unpause trigger inputs

7.10.5.2 Command

```
function.trigger-inputs.unpause.get <channel>
```

```
function.trigger-inputs.unpause.set <channel> <value>
```

7.10.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

7.10.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

7.10.5.7 Minimum security level

function.trigger-inputs.unpause.get	No security required
function.trigger-inputs.unpause.get	User level security required

7.10.5.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.6 snapshot.trigger-inputs.start.get/set

7.10.6.1 Description

Snapshot capture start trigger inputs

7.10.6.2 Command

```
snapshot.trigger-inputs.start.get
```

```
snapshot.trigger-inputs.start.set <value>
```

7.10.6.3 Parameters (get)

None

7.10.6.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.6.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high

Name	Type	Units	Minimum	Maximum	Description
					Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.6.6 Possible error reports

Error return “errcode” string reported	Description
Command could not be carried out	Snapshot capture is running or waiting for digital input trigger, so configuration cannot be changed

7.10.6.7 Minimum security level

snapshot.trigger-inputs.start.get	No security required
snapshot.trigger-inputs.start.get	User level security required

7.10.6.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.7 snapshot.trigger-inputs.stop.get/set

7.10.7.1 Description

Snapshot capture stop trigger inputs

7.10.7.2 Command

```
snapshot.trigger-inputs.stop.get
```

```
snapshot.trigger-inputs.stop.set <value>
```

7.10.7.3 Parameters (get)

None

7.10.7.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.7.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	bitfield			Trigger inputs Bit 0: Input 1 high Bit 1: Input 2 high Bit 2: Input 3 high

Name	Type	Units	Minimum	Maximum	Description
					Bit 8: Input 1 low Bit 9: Input 2 low Bit 10: Input 3 low Bit 16: Input 1 rising-edge Bit 17: Input 2 rising-edge Bit 18: Input 3 rising-edge Bit 24: Input 1 falling-edge Bit 25: Input 2 falling-edge Bit 26: Input 3 falling-edge

7.10.7.6 Possible error reports

Error return “errcode” string reported	Description
Command could not be carried out	Snapshot capture is running or waiting for digital input trigger, so configuration cannot be changed

7.10.7.7 Minimum security level

snapshot.trigger-inputs.stop.get	No security required
snapshot.trigger-inputs.stop.get	User level security required

7.10.7.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.8 snapshot.trigger-inputs.start-delay.get/set

7.10.8.1 Description

Snapshot start trigger delay

7.10.8.2 Command

```
snapshot.trigger-inputs.start-delay.get
```

```
snapshot.trigger-inputs.start-delay.set <value>
```

7.10.8.3 Parameters (get)

None

7.10.8.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples			Snapshot start trigger delay

7.10.8.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples			Snapshot start trigger delay

7.10.8.6 Possible error reports

Error return “errcode” string reported	Description
Command could not be carried out	Snapshot capture is running or waiting for digital input trigger, so configuration cannot be changed

7.10.8.7 Minimum security level

snapshot.trigger-inputs.start-delay.get	No security required
snapshot.trigger-inputs.start-delay.get	User level security required

7.10.8.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.9 function.trigger-inputs.enabled.get

7.10.9.1 Description

Trigger inputs enabled for function playback and/or snapshot

7.10.9.2 Command

```
function.trigger-inputs.enabled.get
```

7.10.9.3 Parameters

None

7.10.9.4 Results

Name	Type	Units	Minimum	Maximum	Description
enabled-snapshot	Boolean		0	1	Enabled trigger inputs for snapshot capture
enabled-internal-channel0	Boolean		0	1	Enabled trigger inputs for function playback for internal channel 0
enabled-channel1	Boolean		0	1	Enabled trigger inputs for function playback for channel 1
enabled-channel2	Boolean		0	1	Enabled trigger inputs for function playback for channel 2 (**only returned for 2- or 3-channel controller)**
enabled-channel3	Boolean		0	1	Enabled trigger inputs for function playback for channel 3 (**only returned for 3-channel controller)**

7.10.9.5 Possible error reports

None

7.10.9.6 Minimum security level

function.trigger-inputs.enabled.get	No security required

7.10.9.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.10 function.trigger-inputs.enabled.enable-inputs

7.10.10.1 Description

Enable trigger inputs for function playback and/or snapshot

Note that after trigger inputs are enabled for function playback or for snapshot capture, the relevant configuration may not be changed until trigger inputs are disabled.

7.10.10.2 Command

```
function.trigger-inputs.enabled.enable-inputs <enable-snapshot>
<enable-internal-channel0> <enable-channel1> [enable-channel2]
[enable-channel3]
```

7.10.10.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
enable-snapshot	Boolean		0	1	Enable trigger inputs for snapshot capture
enable-internal-channel0	Boolean		0	1	Enable trigger inputs for function playback for internal channel 0
enable-channel1	Boolean		0	1	Enable trigger inputs for function playback for channel 1
enable-channel2	Boolean		0	1	Enable trigger inputs for function playback for channel 2 (**only required for 2- or 3-channel controller)**
enable-channel3	Boolean		0	1	Enable trigger inputs for function playback for channel 3 (**only required for 3-channel controller)**

7.10.10.4 Results

Name	Type	Units	Minimum	Maximum	Description
enable-snapshot	Boolean		0	1	Enable trigger inputs for snapshot capture

Name	Type	Units	Minimum	Maximum	Description
enable-internal-channel0	Boolean		0	1	Enable trigger inputs for function playback for internal channel 0
enable-channel1	Boolean		0	1	Enable trigger inputs for function playback for channel 1
enable-channel2	Boolean		0	1	Enable trigger inputs for function playback for channel 2 (**only returned for 2- or 3-channel controller)**
enable-channel3	Boolean		0	1	Enable trigger inputs for function playback for channel 3 (**only returned for 3-channel controller)**

7.10.10.5 Possible error reports

None

Note that if a flag is set for a channel which has no stage present, the command for that channel will be ignored. Since this command controls multiple channels, this is preferable to reporting an error when one channel cannot be controlled but other channels can. The return value for that channel will return zero, to indicate that the command could not be carried out.

7.10.10.6 Minimum security level

function.trigger-inputs.enabled.enable-inputs	User level security required

7.10.10.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

7.10.11 function.trigger-inputs.enabled.disable-inputs

7.10.11.1 Description

Disable trigger inputs for function playback and/or snapshot

7.10.11.2 Command

```
function.trigger-inputs.enabled.disable-inputs <disable-snapshot>
<disable-internal-channel0> <disable-channel1> [disable-channel2]
[disable-channel3]
```

7.10.11.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
disable-snapshot	Boolean		0	1	Disable trigger inputs for snapshot capture
disable-internal-channel0	Boolean		0	1	Disable trigger inputs for function playback for internal channel 0
disable-channel1	Boolean		0	1	Disable trigger inputs for function playback for channel 1
disable-channel2	Boolean		0	1	Disable trigger inputs for function playback for channel 2 (**only required for 2- or 3-channel controller)**
disable-channel3	Boolean		0	1	Disable trigger inputs for function playback for channel 3 (**only required for 3-channel controller)**

7.10.11.4 Results

Name	Type	Units	Minimum	Maximum	Description
disable-snapshot	Boolean		0	1	Disable trigger inputs for snapshot capture
disable-internal-channel0	Boolean		0	1	Disable trigger inputs for function playback for internal channel 0

Name	Type	Units	Minimum	Maximum	Description
disable-channel1	Boolean		0	1	Disable trigger inputs for function playback for channel 1
disable-channel2	Boolean		0	1	Disable trigger inputs for function playback for channel 2 (**only returned for 2- or 3-channel controller)**
disable-channel3	Boolean		0	1	Disable trigger inputs for function playback for channel 3 (**only returned for 3-channel controller)**

7.10.11.5 Possible error reports

None

Note that if a flag is set for a channel which has no stage present, the command for that channel will be ignored. Since this command controls multiple channels, this is preferable to reporting an error when one channel cannot be controlled but other channels can. The return value for that channel will return zero, to indicate that the command could not be carried out.

7.10.11.6 Minimum security level

function.trigger-inputs.enabled.disable-inputs	User level security required

7.10.11.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

8 Snapshot capture

8.1 Snapshot configuration and control

The snapshot captures the measured positions for all channels for up to 10s at the controller sample rate. A 6000 series controller with a 50kHz internal control loop will therefore store up to 500,000 samples. The value `controller.sample-time` (see 4.1.1) reports the controller's internal control loop period, so that PC software can represent this data appropriately. Once started, the snapshot will run to completion unless stopped by the user.

By default, snapshot channels 1-3 capture measured position for stages on channels 1-3, where these physical channels are present on the controller. Where physical channels are not present (i.e. on a single-channel or two-channel controller), snapshot defaults to capturing measured position for the stage on channel 1. Snapshot captures the measured position to the control loop, as shown in the diagram in section 5.1.4.

A major use of the snapshot is to measure the system step response. To enable this without the complexity of setting up a function playback waveform, the snapshot includes a simple step generator. More complex waveforms may be created using function playback if required (see 6).

Commands to start and stop snapshot work identically to the function playback commands or digital trigger inputs commanding snapshot start or stop. If function playback is not required, it is convenient to have commands which allow snapshot to be controlled separately.

If desired, the scope feature (see section 9) may be used to capture other signals instead. An additional snapshot channel 0 is available which is primarily intended for scope signal capture, for a total of 4 snapshot channels. Single-channel and two-channel controllers still provide snapshot channels 2 and 3, even when those physical channels are not present, so all controllers provide 4 snapshot channels which the scope feature may freely use to capture any signal within the controller. Section 9 describes the scope feature in more detail.

8.1.1 snapshot.fire

8.1.1.1 Description

Start a snapshot capture

8.1.1.2 Command

```
snapshot.fire
```

8.1.1.3 Parameters

None

8.1.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer				Always returns 1

8.1.1.5 Possible error reports

None

8.1.1.6 Minimum security level

snapshot.fire	User level security required

8.1.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

8.1.2 snapshot.stop

8.1.2.1 Description

Stop a snapshot capture

8.1.2.2 Command

```
snapshot.stop
```

8.1.2.3 Parameters

None

8.1.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer				Always returns 0

8.1.2.5 Possible error reports

None

8.1.2.6 Minimum security level

snapshot.stop	User level security required

8.1.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

8.1.3 snapshot.capture.count.get/set

8.1.3.1 Description

Snapshot capture duration

8.1.3.2 Command

```
snapshot.capture.count.get
```

```
snapshot.capture.count.set <value>
```

8.1.3.3 Parameters (get)

None

8.1.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	0	See 8.1	Capture duration

8.1.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	0	See 8.1	Capture duration

8.1.3.6 Possible error reports

Error return “errcode” string reported	Description
Value out of range	Capture duration must be within specified range (set only)

8.1.3.7 Minimum security level

snapshot.capture.count.get	No security required
snapshot.capture.count.get	User level security required

8.1.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

8.1.4 snapshot.trigger.to-target.get/set

8.1.4.1 Description

Time from start to start of snapshot commanded position step

8.1.4.2 Command

```
snapshot.trigger.to-target.get <channel>
```

```
snapshot.trigger.to-target.set <channel> <value>
```

8.1.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Snapshot channel

8.1.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Snapshot channel
value	32-bit unsigned integer	samples	0	See 8.1	Start of step

8.1.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	0	See 8.1	Start of step

8.1.4.6 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of snapshot channels
Value out of range	Start of step must be within specified range (set only)

8.1.4.7 Minimum security level

snapshot.trigger.to-target.get	No security required
snapshot.trigger.to-target.set	User level security required

8.1.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

8.1.5 snapshot.trigger.from-target.get/set

8.1.5.1 Description

Time from start to end of snapshot commanded position step

8.1.5.2 Command

```
snapshot.trigger.from-target.get <channel>
```

```
snapshot.trigger.from-target.set <channel> <value>
```

8.1.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Snapshot channel

8.1.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Snapshot channel
value	32-bit unsigned integer	samples	0	See 8.1	End of step

8.1.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	0	See 8.1	End of step

8.1.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of snapshot channels
Value out of range	End of step must be within specified range (set only)

8.1.5.7 Minimum security level

snapshot.trigger.from-target.get	No security required
snapshot.trigger.from-target.set	User level security required

8.1.5.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

8.1.6 snapshot.trigger.step.get/set

8.1.6.1 Description

Snapshot commanded position step size

8.1.6.2 Command

```
snapshot.trigger.step.get <channel>
```

```
snapshot.trigger.step.set <channel> <value>
```

8.1.6.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Snapshot channel

8.1.6.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Snapshot channel
value	32-bit floating-point	pm			Step size

8.1.6.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	pm			Step size

8.1.6.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of snapshot channels
Value out of range	Start of step must be within specified range (set only)

8.1.6.7 Minimum security level

snapshot.trigger.step.get	No security required
snapshot.trigger.step.set	User level security required

8.1.6.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

8.2 Snapshot results

8.2.1 snapshot.response.data.get

8.2.1.1 Description

Snapshot captured data

8.2.1.2 Command

```
snapshot.response.data.get <channel> <index>
```

8.2.1.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Snapshot channel
index	32-bit unsigned integer	samples	0	See 8.1	Sample index for data point

8.2.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point				Snapshot data

Where the snapshot channel reports measured position for a channel (i.e. default operation), units for snapshot data will be pm as for other commands reporting position.

Where the snapshot channel is used by the scope feature to report other data (see section 9), units will depend on the signal being monitored.

8.2.1.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of snapshot channels
Index out of range	Sample index must be within specified range

8.2.1.6 Minimum security level

snapshot.response.data.get	No security required

8.2.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

8.2.2 snapshot.response.count.get

8.2.2.1 Description

Snapshot captured data duration

8.2.2.2 Command

```
snapshot.response.count.get
```

8.2.2.3 Parameters

None

8.2.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	0	See 8.1	Capture duration

8.2.2.5 Possible error reports

None

8.2.2.6 Minimum security level

snapshot.response.count.get	No security required

8.2.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

8.2.3 snapshot.response.data-select.get

8.2.3.1 Description

Snapshot captured data selected

8.2.3.2 Command

```
snapshot.response.data-select.get <channel>
```

8.2.3.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Snapshot channel

8.2.3.4 Results

Name	Type	Units	Minimum	Maximum	Description
data-selection	16-bit unsigned integer	enum			Scope data selected
physical-channel	8-bit unsigned integer				Physical channel

Where the snapshot channel reports measured position for a channel (i.e. default operation), data-selection will be zero and physical-channel will be the channel number.

Where the snapshot channel is used by the scope feature to report other data (see section 9), these will specify the signal being monitored.

8.2.3.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of snapshot channels

8.2.3.6 Minimum security level

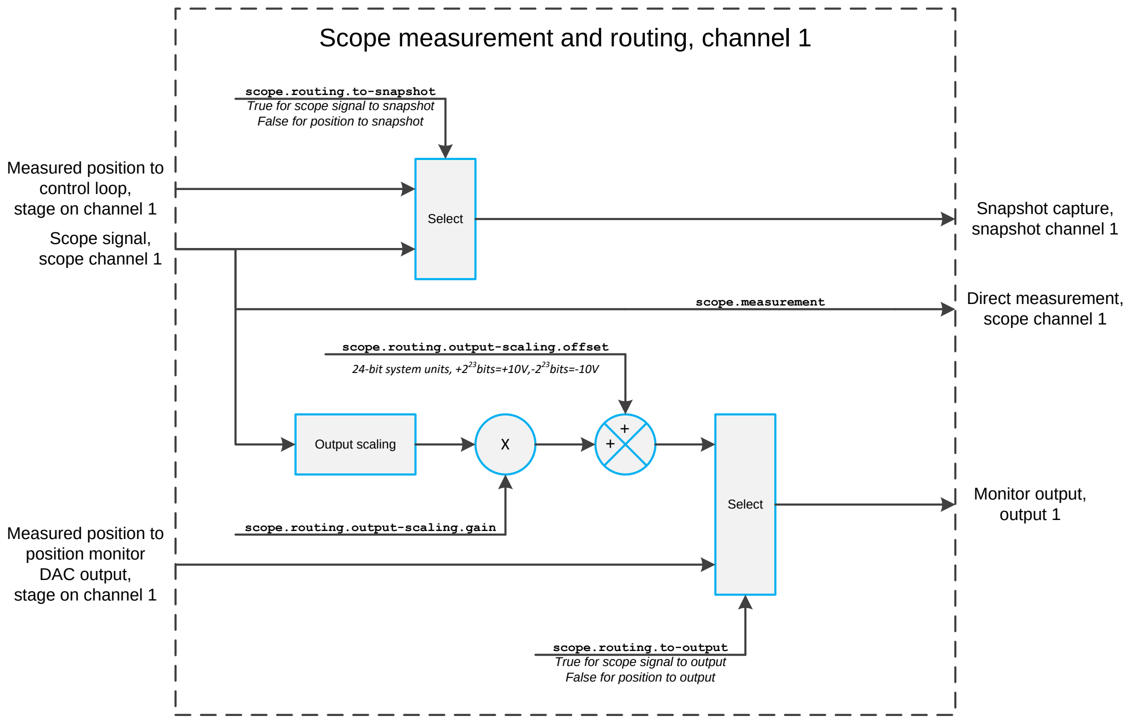
snapshot.response.data.get	No security required

8.2.3.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

9 Scope monitoring of internal data

9.1 Scope measurement and routing


Scope measurement and routing, channel 1
scope.routing.to-snapshot
True for scope signal to snapshot
False for position to snapshot
Measured position to control loop, stage on channel 1
Scope signal, scope channel 1
Select
Snapshot capture, snapshot channel 1
scope.measurement
Direct measurement, scope channel 1
scope.routing.output-scaling.offset
24-bit system units, +2^23bits=+10V,-2^23bits=-10V
Output scaling
X
+
+
Select
Monitor output, output 1
scope.routing.output-scaling.gain
Measured position to position monitor DAC output, stage on channel 1
scope.routing.to-output
True for scope signal to output
False for position to output

Users may need access to internal signals for calibration and monitoring purposes. It is frequently useful to be able to directly observe the response of a filter, for example, or to observe how the commanded velocity and acceleration are controlled by the feedforward command trajectory limiting. As with snapshots (see section 8), there are four channels of scope measurement, numbered 0-3. The number of scope channels is not reduced for controllers with fewer physical channels.

As the diagram above shows, the signal for scope channel 1 may be routed to snapshot channel 1 and/or to monitor output 1, as well as reported directly on request. If the scope signal is not routed to the snapshot or monitor output, the measured position for the stage on channel 1 will be routed to that snapshot and monitor output instead. Similarly for scope channels 2 and 3, these may be routed to snapshot channels 2 and 3 and/or to monitor outputs 2 and 3 respectively.

Scope channel 0 is not associated with a physical channel for measured position or monitor output. The same routing options exist, but routing to monitor output has no effect. The same applies for channels 2 and 3 on single-channel controllers, or channel 3 on dual-channel controllers, where the corresponding physical channel does not exist.

Scope signals are identified by an ID number and a physical channel number. Each scope signal may come from any physical channel on the controller; scope channel 1 could be assigned to report unfiltered measured position for the stage on channel 3, for example. The ID number for each signal is shown in the diagrams in section 5 in the format `123`; to continue the previous example, the unfiltered measured position is shown as ID 313 in the diagram in section 5.1.4. See the list in section 0 for full details of scope signals available.

The stage datasheet specifies the standard scaling from measured position (in picometres/nanometres/microns) to volts. Where a scope signal reports a position and that signal is routed to the monitor output, this scaling is applied in all cases. Other scope signals may report values in signed 24-bit system units (i.e. a range +/-2^{23}), which are sent to the output scaled as +/-2^{23} bits = +/-10V on the monitor output.

Particularly small or large signals may require further scaling in order to be easily observable on the monitor output. A gain and offset may be applied to the scope signal if required.

The original intention of the scope functionality was to facilitate stage calibration, but customers are free to use this for other purposes as required. For example:-

- For some applications, it may be more appropriate for the position setpoint to be available on the monitor output instead of the measured position.
- Customers may wish to disable the analogue input as a position command source and inject an unrelated signal which can be read by the host PC, avoiding the need for an extra ADC in the system.
- Function playback may require an extra output signal synchronised to the function, for example generating a sine wave as the stage moves. Function playback channel 0 can be used to generate this extra waveform (see 7.1), and scope channel routing allows this to be sent to a monitor output.

Note that since this is primarily intended as a debugging and calibration tool, scope routing selection is not saved on power-off. If scope routing is to be used as a feature of the control system, the host PC must configure the required routing options every time the controller powers on.

9.1.1 scope.data-select.get/set

9.1.1.1 Description

Scope data signal selected

9.1.1.2 Command

```
scope.data-select.get <channel>
```

```
scope.data-select.set <channel> <data-selection> <physical-channel>
```

9.1.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel

9.1.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel
data-selection	16-bit unsigned integer	enum			Scope signal ID in control system
physical-channel	8-bit unsigned integer				Stage channel for scope signal

9.1.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
data-selection	16-bit unsigned integer	enum			Scope signal ID in control system
physical-channel	8-bit unsigned integer				Stage channel for scope signal

9.1.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of scope channels

Error return “errcode” string reported	**Description**
Value out of range	Physical channel parameter is larger than number of controller channels, or data selection is not a valid signal ID

9.1.1.7 Minimum security level

scope.data-select.get	No security required
scope.data-select.set	User level security required

9.1.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

9.1.2 scope.measurement.get

9.1.2.1 Description

Current value of scope data signal

9.1.2.2 Command

```
scope.measurement.get <channel>
```

9.1.2.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel

9.1.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point				Scope data signal

9.1.2.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of scope channels

9.1.2.6 Minimum security level

scope.measurement.get	No security required

9.1.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

9.1.3 scope.routing.to-snapshot.get/set

9.1.3.1 Description

Scope data signal routed to snapshot

9.1.3.2 Command

```
scope.routing.to-snapshot.get <channel>
```

```
scope.routing.to-snapshot.set <channel> <value>
```

9.1.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel

9.1.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel
value	32-bit unsigned integer	Boolean	0	1	Scope channel routed to snapshot

9.1.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Scope channel routed to snapshot

9.1.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of scope channels

9.1.3.7 Minimum security level

scope.routing.to-snapshot.get	No security required
scope.routing.to-snapshot.set	User level security required

9.1.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

9.1.4 scope.routing.to-output.get/set

9.1.4.1 Description

Scope data signal routed to monitor output

9.1.4.2 Command

```
scope.routing.to-output.get <channel>
```

```
scope.routing.to-output.set <channel> <value>
```

9.1.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel

9.1.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel
value	32-bit unsigned integer	Boolean	0	1	Scope channel routed to monitor output

9.1.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean	0	1	Scope channel routed to monitor output

9.1.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of scope channels

9.1.4.7 Minimum security level

scope.routing.to-output.get	No security required
scope.routing.to-output.set	User level security required

9.1.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

9.1.5 scope.routing.output-scaling.gain.get/set

9.1.5.1 Description

Gain applied to scope data signal when routed to monitor output

9.1.5.2 Command

```
scope.routing.output-scaling.gain.get <channel>

scope.routing.output-scaling.gain.set <channel> <value>
```

9.1.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel

9.1.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel
value	32-bit floating-point				Monitor output gain

9.1.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point				Monitor output gain

9.1.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of scope channels

9.1.5.7 Minimum security level

scope.routing.output-scaling.gain.get	No security required
scope.routing.output-scaling.gain.set	User level security required

9.1.5.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

9.1.6 scope.routing.output-scaling.offset.get/set

9.1.6.1 Description

Offset applied to scope data signal when routed to monitor output

9.1.6.2 Command

```
scope.routing.output-scaling.offset.get <channel>
```

```
scope.routing.output-scaling.offset.set <channel> <value>
```

9.1.6.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel

9.1.6.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Scope channel
value	32-bit floating-point	24-bit DAC bits	-16777215	16777215	Monitor output offset

9.1.6.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit floating-point	24-bit DAC bits	-16777215	16777215	Monitor output offset

9.1.6.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of scope channels

9.1.6.7 Minimum security level

scope.routing.output-scaling.offset.get	No security required
scope.routing.output-scaling.offset.set	User level security required

9.1.6.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

9.2 Scope data signal selection

The following scope data signals are available for each physical channel, with the specified scalings when signals are routed to monitor outputs. Where the scaling is listed as “position scaling for stage”, the scaling from microns to voltage on the monitor output is as specified in the stage datasheet.

Signal ID	Signal description	Diagram or reference	Output scaling
0	Measured position to control loop **`stage.position.measured`**	5.1.1, 5.1.4, 5.1.5, 5.1.6, 5.1.9, 5.4.1	Position scaling for stage
101	Analogue input	5.1.1, 5.1.2	Voltage
111	Channel heatsink temperature **`protection.thermal.heatsink.temperature`**	4.2, 4.2.8	±10V=±200°C
112	Internal power supply temperature **`protection.thermal.psu.temperature`**	4.2, 4.2.13	±10V=±200°C
120	Function playback trigger input	7.10	+5V=true,0V=false
122	Stepped input A	6.1	+5V=true,0V=false
123	Stepped input B	6.1	+5V=true,0V=false
124	Stepped input steps received each control loop period	6.1	+5V=one or more forward steps received 0V=no steps received -5V=one or more backward steps received
130	Function playback trigger output	7.6	+5V=true,0V=false
131	“In position” output	5.1.1, 5.1.5	+5V=true,0V=false
134	Stepped output steps sent each control loop period	6.3	+5V=one or more forward steps sent 0V=no steps sent -5V=one or more backward steps sent

Signal ID	Signal description	Diagram or reference	Output scaling
300	Analogue command scaled	5.1.2, 5.1.3	Position scaling for stage
301	Combined digital command	5.1.2, 5.1.3	Position scaling for stage
302	Snapshot step command	5.1.1, 5.1.2, 8	Position scaling for stage
303	Combined position setpoint to control loop	5.1.1, 5.1.2, 5.1.6, 5.1.7	Position scaling for stage
307	Stepped position command **`stage.position.stepped-command`**	5.1.1, 5.1.2, 6.1, 6.2, 6.2.1	Position scaling for stage
308	Absolute position command **`stage.position.absolute-command`**	5.1.1, 5.1.2, 5.1.5, 5.2.1	Position scaling for stage
310	Analogue position command filtered	5.1.2, 5.1.3	Position scaling for stage
311	Digital position command filtered	5.1.2, 5.1.3	Position scaling for stage
313	Measured position scaled and linearised	5.1.4	Position scaling for stage
314	Function playback command	5.1.1, 5.1.2, 6	Position scaling for stage
320	Measured position filtered	5.1.4	Position scaling for stage
350	Digital position command **`stage.position.command`**	5.1.1, 5.1.2, 5.2.2	Position scaling for stage
400	PID output	5.1.9	+10V=full-scale positive HV -10V=full-scale negative HV
401	PID proportional component	5.1.9	+10V=full-scale positive HV -10V=full-scale negative HV
402	PID integral component	5.1.9	+10V=full-scale positive HV -10V=full-scale negative HV

Signal ID	Signal description	Diagram or reference	Output scaling
403	PID differential component	5.1.9	+10V=full-scale positive HV -10V=full-scale negative HV
404	PID integral error	5.1.9	+10V=full-scale positive HV -10V=full-scale negative HV
405	PID output after filtering	5.1.6, 5.1.9	+10V=full-scale positive HV -10V=full-scale negative HV
410	Open-loop control output	5.1.6, 5.1.8	+10V=full-scale positive HV -10V=full-scale negative HV
450	Combined command after command trajectory limiting (if selected)	5.1.6, 5.1.7, , 5.1.8, 5.1.9	Position scaling for stage
451	Command velocity after command trajectory limiting	5.1.7	Position scaling for stage per ms
600	Control output	5.1.1, 5.1.10	+10V=full-scale positive HV -10V=full-scale negative HV
601	“In position” error after LPF smoothing	5.1.5	Position scaling for stage
602	“In position” confirmed flag, using LPF error smoothing for debouncing	5.1.5	+5V=true,0V=false
603	“In position” unconfirmed flag	5.1.5	+5V=true,0V=false
604	“In position” confirmed flag, using window filter for debouncing	5.1.5	+5V=true,0V=false
605	“In position” error	5.1.5	Position scaling for stage
606	“In position” window filter depth	5.1.5	Proportion of window filter depth full 0V=window filter empty +2.5V=window filter half full +5V=window filter full

Signal ID	Signal description	Diagram or reference	Output scaling
620	Control output before filtering	5.1.6	+10V=full-scale positive HV -10V=full-scale negative HV
621	Control output after filtering	5.1.1, 5.1.6, 5.1.10	+10V=full-scale positive HV -10V=full-scale negative HV

10 Stage calibration

10.1 Stage calibration storage

Calibration settings for stage control (i.e. all settings whose commands start with “stage.”) may be stored as preset calibrations. The intention is that users with “superuser” security level will configure the stage to known-good values, and users with “user” security level will simply load an appropriate preset for their application. Presets may be tuned for different loads, to trade off speed of response versus noise level, or for any other reasons that may be appropriate.

Calibration settings are stored in non-volatile memory contained within the stage connector. Each stage’s calibration settings therefore travel with the stage, and controllers may be swapped out freely for maintenance without affecting system behaviour.

A total of 8 calibration presets are available. Three factory-set “fast”, “medium” and “slow” presets are configured during production for reference, and may not be changed by customers. Five further customer presets may be set at will by customers. Presets may be freely loaded and saved during normal operation as required. The default preset on startup may also be selected.

After a preset has been loaded, calibration settings for stage control may be changed as required. These changes are not automatically stored, and will be lost on power-off or if the user loads a different preset. The user must save these to a calibration preset if they are required to be kept.

As calibration presets are developed, customers will typically rotate through the five customer presets to save the latest settings. This may make it hard to know which is the latest calibration preset. To allow customers to keep track of this more easily, each preset includes a “configuration ID” which may be freely set to any value. This may correspond to an entry in a customer database or a test number, or simply be incremented each time the customer saves a new configuration. Customers may then read the configuration ID for a preset when they load it, so that they know the correct preset is selected.

When a calibration preset is saved, it takes a short period of time for the data to be fully written back to the stage. Controller operation is unaffected, but if power is lost during this period then this calibration preset may be corrupted and its data lost. It is vital after saving a calibration preset that the user reads the command `stage.calibration.status.get` until it reports “idle”. While it reports “busy”, the controller is still saving data to the stage and controller power must not be removed.

10.1.1 stage.calibration.status.get

10.1.1.1 Description

Get status of current stage operation

10.1.1.2 Command

```
stage.calibration.status.get <channel>
```

10.1.1.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

10.1.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Enum			Stage operation status 0 = Loading stage data 1 = Idle 2 = Busy 4 = Load failure 5 = Write failure 6 = Verify failure

10.1.1.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

10.1.1.6 Minimum security level

stage.calibration.status.get	No security required

10.1.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

10.1.2 stage.calibration.preset.current.get

10.1.2.1 Description

Current calibration preset selected for stage

10.1.2.2 Command

```
stage.calibration.preset.current.get <channel>
```

10.1.2.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

10.1.2.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Enum			Current calibration preset 3=preset fast 4=preset medium 5=preset slow 6=customer 1 7=customer 2 8=customer 3 9=customer 4 10=customer 5

10.1.2.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

10.1.2.6 Minimum security level

stage.calibration.preset.current.get	No security required

10.1.2.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

10.1.3 stage.calibration.preset.default.get

10.1.3.1 Description

Default (startup) calibration preset selected for stage

10.1.3.2 Command

```
stage.calibration.preset.default.get <channel>
```

10.1.3.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

10.1.3.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Enum			Default calibration preset 3=preset fast 4=preset medium 5=preset slow 6=customer 1 7=customer 2 8=customer 3 9=customer 4 10=customer 5

10.1.3.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

10.1.3.6 Minimum security level

stage.calibration.preset.default.get	No security required

10.1.3.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

10.1.4 stage.calibration.preset.default.save

10.1.4.1 Description

Set the current calibration preset as the default (startup) calibration preset for stage

10.1.4.2 Command

```
stage.calibration.preset.default.save <channel>
```

10.1.4.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

10.1.4.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean			Carried out successfully 1 = started No other values possible

10.1.4.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Stage calibration data storage fault	Non-volatile memory storing stage calibration has failed so stage calibration data cannot be loaded or saved

10.1.4.6 Minimum security level

stage.calibration.preset.default.save	Superuser level security required

10.1.4.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

10.1.5 stage.calibration.preset.exists

10.1.5.1 Description

Check whether calibration preset exists

10.1.5.2 Command

```
stage.calibration.preset.exists <channel> <value>
```

10.1.5.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Enum			Calibration preset 3=preset fast 4=preset medium 5=preset slow 6=customer 1 7=customer 2 8=customer 3 9=customer 4 10=customer 5

10.1.5.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean			Calibration preset exists 1 = exists 0 = does not exist

10.1.5.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

Error return "errcode" string reported	**Description**
Value out of range	Calibration preset value is not a valid preset ID

10.1.5.6 Minimum security level

stage.calibration.preset.exists	No security required

10.1.5.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

10.1.6 stage.calibration.preset.load

10.1.6.1 Description

Load a calibration preset for stage

10.1.6.2 Command

```
stage.calibration.preset.load <channel> <value>
```

10.1.6.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Enum			Load calibration preset 3=preset fast 4=preset medium 5=preset slow 6=customer 1 7=customer 2 8=customer 3 9=customer 4 10=customer 5

10.1.6.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean			Carried out successfully 1 = started No other values possible

10.1.6.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

Error return “errcode” string reported	Description
Value out of range	Calibration preset value is not a valid preset ID, or preset does not exist
Stage calibration data storage fault	Non-volatile memory storing stage calibration has failed so stage calibration data cannot be loaded or saved

10.1.6.6 Minimum security level

stage.calibration.preset.load	User level security required

10.1.6.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

10.1.7 stage.calibration.preset.save

10.1.7.1 Description

Save current calibration as a calibration preset for stage

10.1.7.2 Command

```
stage.calibration.preset.save <channel> <value>
```

10.1.7.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Enum			Save calibration preset 6=customer 1 7=customer 2 8=customer 3 9=customer 4 10=customer 5

10.1.7.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean			Carried out successfully 1 = started No other values possible

10.1.7.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Calibration preset value is not a valid preset ID
Stage calibration data storage fault	Non-volatile memory storing stage calibration has failed so stage calibration data cannot be loaded or saved

10.1.7.6 Minimum security level

stage.calibration.preset.save	Superuser level security required

10.1.7.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

10.1.8 stage.calibration.preset.delete

10.1.8.1 Description

Delete a calibration preset for stage

10.1.8.2 Command

```
stage.calibration.preset.delete <channel> <value>
```

10.1.8.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	Enum			Delete calibration preset 6=customer 1 7=customer 2 8=customer 3 9=customer 4 10=customer 5

10.1.8.4 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	Boolean			Carried out successfully 1 = started No other values possible

10.1.8.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel
Value out of range	Calibration preset value is not a valid preset ID
Stage calibration data storage fault	Non-volatile memory storing stage calibration has failed so stage calibration data cannot be loaded or saved

Error return “errcode” string reported	**Description**
Command could not be carried out	Delete command attempted to delete current or default calibration preset, which is not permitted

10.1.8.6 Minimum security level

stage.calibration.preset.delete	Superuser level security required

10.1.8.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

10.1.9 stage.calibration.preset.configuration-id.get/set

10.1.9.1 Description

Configuration ID for stage calibration preset

10.1.9.2 Command

```
stage.calibration.preset.configuration-id.get <channel>
```

```
stage.calibration.preset.configuration-id.set <channel> <value>
```

10.1.9.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

10.1.9.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer				Configuration ID

10.1.9.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer				Configuration ID

10.1.9.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

10.1.9.7 Minimum security level

stage.calibration.preset.configuration-id.get	No security required
stage.calibration.preset.configuration-id.get	Superuser level security required

10.1.9.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11 Deprecated commands

The following commands have been replaced with versions which are considered more user-friendly. However they are retained in the interface DLL for backwards compatibility with older code.

11.1 Control loop configuration

11.1.1 stage.mode.get

11.1.1.1 Description

Stage mode and status

11.1.1.2 Command

```
stage.mode.get <channel>
```

11.1.1.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.1.1.4 Results

Name	Type	Units	Minimum	Maximum	Description
status	16-bit unsigned integer	Bitfield			Bit 12: Controller is synchronisation slave and synchronisation is locked to master Bit 11: Controller is synchronisation master Bit 10: Protection shutdown active (see 4.2) Bit 4: Stage in position unconfirmed Bit 2: Stage “in position” confirmed by window filter Bit 1: Stage connected Bit 0: Stage “in position” confirmed by low-pass filter

Name	Type	Units	Minimum	Maximum	Description
mode	16-bit unsigned integer	Bitfield			Bit 10: Stage "in position" output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

11.1.1.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

11.1.1.6 Minimum security level

stage.mode.get	No security required

11.1.1.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.1.8 Deprecated in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.1.9 Replaced by

Equivalent `stage.mode` commands as described in 5.11.

11.1.2 stage.mode-only.get/set

11.1.2.1 Description

Stage mode

11.1.2.2 Command

```
stage.mode-only.get <channel>
```

```
stage.mode-only.set <channel> <mode>
```

11.1.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.1.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
mode	16-bit unsigned integer	Bitfield			Bit 10: Stage "in position" output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

This allows the user to change multiple settings in one operation.

Each bit within the bitfield is protected by security, corresponding with security levels for corresponding commands. No error will be reported if bits are set without having the correct security level, but that setting will not be changed. The return value reports the actual state of all these modes.

Stage "in position" output confirmation	Superuser level security required
Digital command enabled	Superuser level security required

Analogue input command enabled	Superuser level security required
Servo output frozen	User level security required
Closed loop control enabled	Superuser level security required

11.1.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
mode	16-bit unsigned integer	Bitfield			Bit 10: Stage “in position” output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

11.1.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

11.1.2.7 Minimum security level

stage.mode-only.get	No security required
stage.mode-only.set	User level required

11.1.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.2.9 Deprecated in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.2.10 Replaced by

Equivalent `stage.mode` commands as described in 5.11.

11.1.3 stage.mode-mask.set

11.1.3.1 Description

Set stage mode using mask to select changes

11.1.3.2 Command

```
stage.mode-mask.set <channel> <mask> <mode>
```

11.1.3.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
mask	16-bit unsigned integer	Bitfield			Mode setting mask Bit 10: Stage "in position" output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled
mode	16-bit unsigned integer	Bitfield			Mode setting states Bit 10: Stage "in position" output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

This allows the user to change multiple settings in one operation, but leave other settings unchanged if required. Bits set in the “mask” cause the settings in the “mode” to take effect. Where bits are cleared for settings in the “mask”, those mode settings are left unchanged.

Each bit within the bitfield is protected by security, corresponding with security levels for corresponding commands. No error will be reported if bits are set without having the correct security level, but that setting will not be changed. The return value reports the actual state of all these modes.

Stage “in position” output confirmation	Superuser level security required
Digital command enabled	Superuser level security required
Analogue input command enabled	Superuser level security required
Servo output frozen	User level security required
Closed loop control enabled	Superuser level security required

11.1.3.4 Results

Name	Type	Units	Minimum	Maximum	Description
status	16-bit unsigned integer	Bitfield			Bit 12: Controller is synchronisation slave and synchronisation is locked to master Bit 11: Controller is synchronisation master Bit 10: Protection shutdown active (see 4.2) Bit 4: Stage in position unconfirmed Bit 2: Stage “in position” confirmed by window filter Bit 1: Stage connected Bit 0: Stage “in position” confirmed by low-pass filter
mode	16-bit unsigned integer	Bitfield			Bit 10: Stage “in position” output confirmation, 0=low-

Name	Type	Units	Minimum	Maximum	Description
					pass filter selected, 1=window filter selected Bit 9: Digital command enabled Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

11.1.3.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

11.1.3.6 Minimum security level

stage.mode-mask.set	User level required

11.1.3.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.3.8 Deprecated in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.3.9 Replaced by

Equivalent `stage.mode` commands as described in 5.11.

11.1.4 stage.mode.analogue-command.enable/disable

11.1.4.1 Description

Enable/disable analogue input position commands

11.1.4.2 Command

```
stage.mode.analogue-command.enable <channel>
```

```
stage.mode.analogue-command.disable <channel>
```

11.1.4.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.1.4.4 Results

Name	Type	Units	Minimum	Maximum	Description
status	16-bit unsigned integer	Bitfield			Bit 12: Controller is synchronisation slave and synchronisation is locked to master Bit 11: Controller is synchronisation master Bit 10: Protection shutdown active (see 4.2) Bit 4: Stage in position unconfirmed Bit 2: Stage “in position” confirmed by window filter Bit 1: Stage connected Bit 0: Stage “in position” confirmed by low-pass filter
mode	16-bit unsigned integer	Bitfield			Bit 10: Stage “in position” output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled

Name	Type	Units	Minimum	Maximum	Description
					Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

11.1.4.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

11.1.4.6 Minimum security level

stage.mode.analogue-command.enable/disable	Superuser level security required

11.1.4.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.4.8 Deprecated in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.4.9 Replaced by

Equivalent `stage.mode.analogue-command.get/set` commands as described in 5.11.1.

11.1.5 stage.mode.digital-command.enable/disable

11.1.5.1 Description

Enable/disable digital position command (direct command from PC, function playback and snapshot step)

11.1.5.2 Command

```
stage.mode.digital-command.enable <channel>
```

```
stage.mode.digital-command.disable <channel>
```

11.1.5.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.1.5.4 Results

Name	Type	Units	Minimum	Maximum	Description
status	16-bit unsigned integer	Bitfield			Bit 12: Controller is synchronisation slave and synchronisation is locked to master Bit 11: Controller is synchronisation master Bit 10: Protection shutdown active (see 4.2) Bit 4: Stage in position unconfirmed Bit 2: Stage “in position” confirmed by window filter Bit 1: Stage connected Bit 0: Stage “in position” confirmed by low-pass filter
mode	16-bit unsigned integer	Bitfield			Bit 10: Stage “in position” output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled

Name	Type	Units	Minimum	Maximum	Description
					Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

11.1.5.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

11.1.5.6 Minimum security level

stage.mode.digital-command.enable/disable	Superuser level security required

11.1.5.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.5.8 Deprecated in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.5.9 Replaced by

Equivalent `stage.mode.digital-command.get/set` commands as described in 5.11.2.

11.1.6 stage.mode.closed-loop.enable/disable

11.1.6.1 Description

Enable/disable closed-loop control operation (and hence disable/enable open-loop operation)

11.1.6.2 Command

```
stage.mode.closed-loop.enable <channel>

stage.mode.closed-loop.disable <channel>
```

11.1.6.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.1.6.4 Results

Name	Type	Units	Minimum	Maximum	Description
status	16-bit unsigned integer	Bitfield			Bit 12: Controller is synchronisation slave and synchronisation is locked to master Bit 11: Controller is synchronisation master Bit 10: Protection shutdown active (see 4.2) Bit 4: Stage in position unconfirmed Bit 2: Stage “in position” confirmed by window filter Bit 1: Stage connected Bit 0: Stage “in position” confirmed by low-pass filter
mode	16-bit unsigned integer	Bitfield			Bit 10: Stage “in position” output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled

Name	Type	Units	Minimum	Maximum	Description
					Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

11.1.6.5 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

11.1.6.6 Minimum security level

stage.mode.closed-loop.enable/disable	Superuser level security required

11.1.6.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.6.8 Deprecated in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.6.9 Replaced by

Equivalent `stage.mode.closed-loop.get/set` commands as described in 5.11.3.

11.1.7 stage.mode.freeze-servo-output.enable/disable

11.1.7.1 Description

Freeze/unfreeze servo output

11.1.7.2 Command

```
stage.mode.freeze-servo-output.enable <channel>

stage.mode.freeze-servo-output.disable <channel>
```

11.1.7.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.1.7.4 Results

Name	Type	Units	Minimum	Maximum	Description
status	16-bit unsigned integer	Bitfield			Bit 12: Controller is synchronisation slave and synchronisation is locked to master Bit 11: Controller is synchronisation master Bit 10: Protection shutdown active (see 4.2) Bit 4: Stage in position unconfirmed Bit 2: Stage “in position” confirmed by window filter Bit 1: Stage connected Bit 0: Stage “in position” confirmed by low-pass filter
mode	16-bit unsigned integer	Bitfield			Bit 10: Stage “in position” output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled

Name	Type	Units	Minimum	Maximum	Description
					Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

11.1.7.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

11.1.7.6 Minimum security level

stage.mode.freeze-servo-output.enable/disable	User level security required

11.1.7.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.7.8 Deprecated in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.7.9 Replaced by

Equivalent `stage.mode.freeze-servo-output.get/set` commands as described in 5.11.4.

11.1.8 stage.mode.in-position-output-select.lpf/window-filter

11.1.8.1 Description

Select stage “in position” (ready) digital output to be confirmed by low-pass filter or window filter

11.1.8.2 Command

```
stage.mode.in-position-output-select.lpf <channel>
```

```
stage.mode.in-position-output-selct.window-filter <channel>
```

11.1.8.3 Parameters

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.1.8.4 Results

Name	Type	Units	Minimum	Maximum	Description
status	16-bit unsigned integer	Bitfield			Bit 12: Controller is synchronisation slave and synchronisation is locked to master Bit 11: Controller is synchronisation master Bit 10: Protection shutdown active (see 4.2) Bit 4: Stage in position unconfirmed Bit 2: Stage “in position” confirmed by window filter Bit 1: Stage connected Bit 0: Stage “in position” confirmed by low-pass filter
mode	16-bit unsigned integer	Bitfield			Bit 10: Stage “in position” output confirmation, 0=low-pass filter selected, 1=window filter selected Bit 9: Digital command enabled

Name	Type	Units	Minimum	Maximum	Description
					Bit 6: Analogue input command enabled Bit 3: Servo output frozen Bit 1: Closed loop control enabled

11.1.8.5 Possible error reports

Error return "errcode" string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Channel not available	No stage is connected to the specified channel

11.1.8.6 Minimum security level

stage.mode.in-position-output-select.lpf/window-filter	Superuser level security required

11.1.8.7 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.8.8 Deprecated in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.1.8.9 Replaced by

Equivalent `stage.mode.in-position-output-select.get/set` commands as described in 5.11.5.

11.2 Function playback configuration

11.2.1 function.waveform.steps-per-sample.get/set

11.2.1.1 Description

Function playback control loop sample time steps per waveform data point

11.2.1.2 Command

```
function.waveform.steps-per-sample.get <channel>
```

```
function.waveform.steps-per-sample.set <channel> <value>
```

11.2.1.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.2.1.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	samples	1		Steps per data point

11.2.1.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	1		Steps per data point

11.2.1.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Number of steps must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

11.2.1.7 Minimum security level

function.waveform.steps-per-sample.get	No security required
function.waveform.steps-per-sample.set	User level security required

11.2.1.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.2.1.9 Deprecated in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

11.2.1.10 Replaced by

Equivalent `function.waveform.sample-period.get/set` commands to set value in seconds as described in 7.8.8.

11.2.2 function.waveform.steps-per-trigger-out-pulse.get/set

11.2.2.1 Description

Function playback waveform trigger output pulse duration

11.2.2.2 Command

```
function.waveform.steps-per-trigger-out-pulse.get <channel>
```

```
function.waveform.steps-per-trigger-out-pulse.set <channel> <value>
```

11.2.2.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.2.2.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	samples	1		Pulse duration

11.2.2.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	1		Pulse duration

11.2.2.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Value must be as specified
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

11.2.2.7 Minimum security level

function.waveform.trigger-out-event.get	No security required
function.waveform.trigger-out-event.set	User level security required

11.2.2.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.2.2.9 Deprecated in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

11.2.2.10 Replaced by

Equivalent `function.trigger-output.pulse-time.get/set` commands to set value in seconds as described in 7.6.9.

11.2.3 function.waveform.waveform-start.get/set

11.2.3.1 Description

Function playback waveform data index for start of waveform section (and end of opening section)

11.2.3.2 Command

```
function.waveform.waveform-start.get <channel>
```

```
function.waveform.waveform-start.set <channel> <value>
```

11.2.3.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.2.3.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	samples	0	See 7.8	Waveform start index

11.2.3.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	0	See 7.8	Waveform start index

11.2.3.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	Start index must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

11.2.3.7 Minimum security level

function.waveform.waveform-start.get	No security required
function.waveform.waveform-start.set	User level security required

11.2.3.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.2.3.9 Deprecated in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

11.2.3.10 Replaced by

Equivalent `function.waveform.repeat-start.get/set` commands to match waveform generator commands as described in 7.8.4.

11.2.4 function.waveform.waveform-end.get/set

11.2.4.1 Description

Function playback waveform data index for start of closing section (and end of waveform section)

11.2.4.2 Command

```
function.waveform.waveform-end.get <channel>
```

```
function.waveform.waveform-end.set <channel> <value>
```

11.2.4.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.2.4.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer	samples	0	See 7.8	Waveform end index

11.2.4.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer	samples	0	See 7.8	Waveform end index

11.2.4.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Value out of range	End index must be within specified range (set only)
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

11.2.4.7 Minimum security level

function.waveform.waveform-end.get	No security required
function.waveform.waveform-end.set	User level security required

11.2.4.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.2.4.9 Deprecated in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

11.2.4.10 Replaced by

Equivalent `function.waveform.repeat-end.get/set` commands to match waveform generator commands as described in 7.8.5.

11.2.5 function.waveform.iterations.get/set

11.2.5.1 Description

Function playback number of waveform iterations

11.2.5.2 Command

```
function.waveform.iterations.get <channel>
```

```
function.waveform.iterations.set <channel> <value>
```

11.2.5.3 Parameters (get)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage

11.2.5.4 Parameters (set)

Name	Type	Units	Minimum	Maximum	Description
channel	8-bit unsigned integer				Controller channel for stage
value	32-bit unsigned integer				Number of iterations 0 = run indefinitely 1+ = run this many iterations

11.2.5.5 Results

Name	Type	Units	Minimum	Maximum	Description
value	32-bit unsigned integer				Number of iterations 0 = run indefinitely 1+ = run this many iterations

11.2.5.6 Possible error reports

Error return “errcode” string reported	Description
Channel number invalid	Channel parameter is larger than number of channels on controller
Command could not be carried out	Function playback is running or waiting for digital input trigger, so configuration cannot be changed

11.2.5.7 Minimum security level

function.waveform.iterations.get	No security required
function.waveform.iterations.set	User level security required

11.2.5.8 Supported in

Controller application firmware	6.2.1 onwards
Controller interface library	2.2.1 onwards

11.2.5.9 Deprecated in

Controller application firmware	6.4.1 onwards
Controller interface library	2.4.1 onwards

11.2.5.10 Replaced by

Equivalent `function.waveform.repeat-count.get/set` commands to match waveform generator commands as described in 7.8.6.